Report: Frame Prediction Model

Dataset

The dataset used for training is a video sourced from YouTube (link: YouTube Video) in .mp4 format. The video frames are extracted for training the model to predict the next frame based on a sequence of previous frames.

Preprocessing Steps

- ✓ **Frame Extraction** => Frames are extracted from the video using OpenCV (VideoCapture) and saved as .png files.
- ✓ **Resizing and Normalization** => Each frame is resized to 64x64 pixels and normalized to the range [0, 1] by dividing pixel values by 255.
- ✓ **Data Augmentation** => Using Keras' ImageDataGenerator, data augmentation techniques like rotation, shifting, and flipping are applied to the frames to increase the dataset's diversity.
- ✓ **Sequence Generation** => Frames are grouped into sequences of 5 frames, where the first 4 are used as inputs and the 5th as the target (next frame).

Model Architecture

The model is designed to predict the next frame using ConvLSTM2D

- ➢ **TimeDistributed Conv2D** => Two convolutional layers (16 and 32 filters) are applied to each frame in the sequence.
- ➢ **ConvLSTM2D** => A ConvLSTM2D layer with 64 filters is used to capture the temporal relationships between frames in the sequence.
- ➢ **Fully Connected Layers** => After flattening the ConvLSTM2D output, the model has dense layers to predict the next frame's pixel values.
- ➢ **Compiling** => The model uses the Adam optimizer and MSE loss, with MAE as the evaluation metric.

Training Process

- ❖ **Data Split** => The frames are split into training (80%) and validation (20%) sets using train_test_split.
- ❖ **Batch Generation** => FrameSequence, a custom data generator, is used to load and augment batches of frame sequences during training.
- ❖ **Model Training** => The model is trained for 10 epochs, with the first two batches of the training set saved as images for inspection.

Evaluation and Results

The model predicts the next frame in the sequence. After prediction, the ground truth and predicted frames are saved side-by-side for comparison.

Sample Output

The result is a side-by-side image showing the predicted frame next to the ground truth frame.

code: https://github.com/Ritik-912/ASSIST_assignment

Improvement and Additional Use Cases

- **Larger Dataset** => Using a larger and more diverse video dataset can improve the model's generalization and performance.
- **Advanced Architectures** => Implementing more advanced architectures like 3D CNNs or Transformer-based models could better capture complex temporal patterns in longer video sequences.
- **Hyperparameter Tuning** => Exploring different hyperparameters (e.g., sequence length, batch size) and augmentations can optimize model performance.
- **Video Compression** => The model can be applied to video compression tasks by predicting key frames and compressing the differences.
- **Video Prediction for Autonomous Vehicles** => The model could be adapted to predict the future frames of a traffic scene, which can be valuable for autonomous driving systems.
- **Augmented Reality** => Predicting the next frame can be useful in AR applications where real-time video predictions can enhance user experience.