CS4045 - Deep Learning Course Project

Predicting and Generating Video Sequences Using Deep Learning

---

Instructions:
The assignment deadline is **2nd December 2024**. There is no extension in the deadline. Please submit on or before the deadline. You must verify that your submissions are correct.

You can work in groups of a maximum of 3 members. You need to fill in the group details in the google sheets.
https://docs.google.com/spreadsheets/d/1a82PDTSf_3RDpc7OLhqexLnH-0Ts6KMfBMTOQ60bixo/edit?usp=sharing

Submission Requirements:
1. Code: Submission of all code files in a zipped folder is to be submitted by one member of the group. Please ensure to follow the proper file naming convention when submitting your assignments. **(Format: 21i-0001_21i-0002_21i-0003_DL_Project)**
2. Ensure your code is clean, readable, and well-organized. Use meaningful variable names and comments.
3. Grading will be based on correctness, code quality, efficiency, and adherence to the Instructions.

Problem Statement

In this project, students will develop a video prediction model that generates future frames based on short input sequences from the UCF101 dataset, a dataset capturing various human activities. The primary goal is to predict multiple consecutive frames to create a coherent video clip, effectively "imagining" the continuation of an action based on a short input sequence. By learning motion patterns, the model will simulate ongoing actions in the same scene, with applications in video synthesis, animation, and scene prediction. Students will also build a simple user interface to visualize the generated video sequence and document their findings in a report.

Link: https://www.kaggle.com/datasets/matthewjansen/ucf101-action-recognition/data

Project Objectives

1. Train a deep learning model to predict multiple consecutive frames in a video sequence from the UCF101 dataset.
2. Generate a video by combining the predicted frames into a continuous sequence.
3. Develop a user interface to visualize input frames, predicted frames, and the complete video clip, including runtime inference.
4. Experiment with three different models to compare their effectiveness in generating realistic video sequences.
5. Document the approach, challenges, and results in a detailed report.

Requirements

1. Data Preparation

- Use a subset of the UCF101 dataset, focusing on actions with consistent motion (e.g., "Walking," "Jumping," or "Biking"). You should select at least 5 classes.
- Preprocess video frames by resizing them to 64x64 pixels and converting them to grayscale or RGB to manage computational requirements.

2. Model Development

- **Model Selection**: Implement **(from scratch)** and compare three different deep learning models:
    - **Convolutional LSTM (ConvLSTM)** for capturing spatial-temporal patterns.
    - **PredRNN** for advanced temporal modeling.
    - **Transformer-based model** for frame generation with an emphasis on long-term dependencies.
- Train each model to take a short input sequence (e.g., 10 frames) and predict the next several frames (e.g., 5-10 frames) to simulate continuous motion.

3. Video Generation

- Develop a function to combine the predicted frames into a video file (using OpenCV or moviepy).
- The generated video should demonstrate a smooth transition from the initial input frames through the predicted frames.

4. User Interface

- Create a user interface using Streamlit or Gradio or Flask to:
    - Upload or select a sample input video clip from the dataset.
    - Run each model and display the generated frames and final video clip.

    - Display runtime inference of each model, allowing users to view predictions and generated frames in a single interface.

5. Evaluation and Reporting

- To evaluate frame prediction accuracy, use evaluation metrics like **Mean Squared Error (MSE)** and **Structural Similarity Index (SSIM)**.
- Conduct a comparative analysis of the three models:
    - Assess prediction quality, computational efficiency, and visual coherence of generated frames.
    - Document results, challenges, and insights for each model in a comprehensive report.

Project Deliverables

1. **Model Code**: Python code for data processing, model training, and frame prediction.
2. **Video Generation Script**: Code for combining predicted frames into a video sequence.
3. **User Interface**: UI for visualizing input and generated video sequences, with runtime inference.
4. **Report**: Detailed document summarizing methodology, results, and a comparative analysis of models. It must be written in Overleaf. Use the IEEE Conference Paper Format for composing the report. IEEE Overleaf Template

---

Grading Criteria

- **Model Performance** (35%): Frame prediction quality and coherence in generated videos.
- **Generated Video Quality** (25%): Smoothness and realism of the generated sequences.
- **Runtime Inference and Model Comparison** (15%): Effective runtime inference and comparison of the three models at the demo time.
- **User Interface Functionality** (10%): Usability and effectiveness of the interface.
- **Code Quality and Organization** (10%): Readability, modularity, and organization.
- **Report Quality** (5%): Completeness and clarity of the written analysis.