AIN SHAMS UNIVERSITY
FACULTY OF ENGINEERING
MECHATRONICS ENGINEERING DEPARTMENT
CSE480: Machine Vision
Fall 2025

Milestone 1 Report

Action Recognition with CNN-LSTM

Team 4

Name	ID	Section
Abdelrhman Tarek Mohamed	2101547	Sec.1
Ahmed Mahmoud AbdelAzeem	2100718	Sec.1
El mahdy Salih	2101875	Sec.1
Mahmoud Atef	2001313	Sec.1
Zaid Reda Farouk	2101603	Sec.1

Submitted to:
Eng. Dina

Contents

1 Problem Definition

Human Activity Recognition (HAR) is a core problem in modern computer vision with direct impact on surveillance, healthcare, human-computer interaction, and assistive technologies. In video surveillance, automatic understanding of activities such as walking, waving for attention, or remaining seated allows systems to filter large video streams for anomalous or safety-critical events. In healthcare and ambient assisted living, continuous monitoring of patients or elderly users can detect falls, prolonged inactivity, or unusual motion patterns without requiring manual observation.

Recognizing actions from raw video is challenging because an action is not defined by a single frame but by how the body configuration evolves over time. Static postures such as *Sitting* and *Standing* can often be inferred from a single image, but actions like *Walking* or *Waving* are inherently temporal.

Figure 1: Overview of the Human Activity Recognition (HAR) challenge, distinguishing static vs. dynamic actions.

A hybrid CNN-LSTM approach directly addresses this limitation by combining strong spatial feature extraction with temporal sequence modeling. The CNN branch focuses on *what* is present in each image, while the LSTM branch focuses on *how* these features

change over time.

2 Data Cleaning & Preprocessing

The dataset for Milestone 1 combines a subset of the UCF-101 action dataset with custom recordings tailored to the project classes. UCF-101 provides a rich collection of short, labeled clips recorded in diverse environments. From this dataset, we use clips corresponding to **Walking** and **Waving**. To complement these, we recorded custom long videos for **Standing** and **Sitting**.

All raw videos are stored and processed as follows:

- **Resizing:** Every frame is resized to 128×128 pixels to ensure consistent resolution.
- **Normalization:** Pixel values are converted to `float32` and normalized to the $[0, 1]$ range.
- **Sequence Generation:** We extract fixed-length sequences of **16 frames** per sample. For custom long videos, a sliding window approach is used.
- **Augmentation:** For every sequence, we create a horizontally flipped copy to double the dataset size and improve robustness to viewpoint changes.

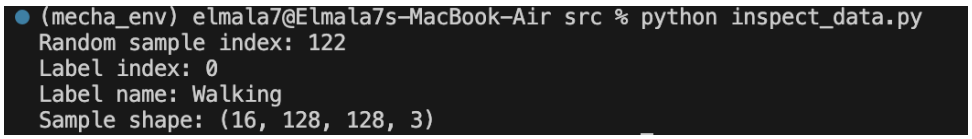

Figure 2: Sample frames from the dataset showing the four classes (Walking, Waving, Standing, Sitting) after preprocessing.

3 Methods & Algorithms

3.1 Model Architecture

The Milestone 1 action model is a hybrid **CNN-LSTM** network designed to leverage both spatial appearance cues and temporal dynamics.

1. **Input:** A tensor of shape $(16, 128, 128, 3)$ representing a sequence of 16 RGB frames.
2. **Spatial Features (MobileNetV2):** We use MobileNetV2 with ImageNet weights (frozen backbone) to extract high-level feature vectors from each frame individually.
3. **Temporal Processing (LSTM):** A `TimeDistributed` layer feeds the sequence of features into an LSTM layer with 64 units and dropout (0.3).
4. **Classification Head:** The final hidden state is passed to a Dense layer with 4 output units and a Softmax activation.

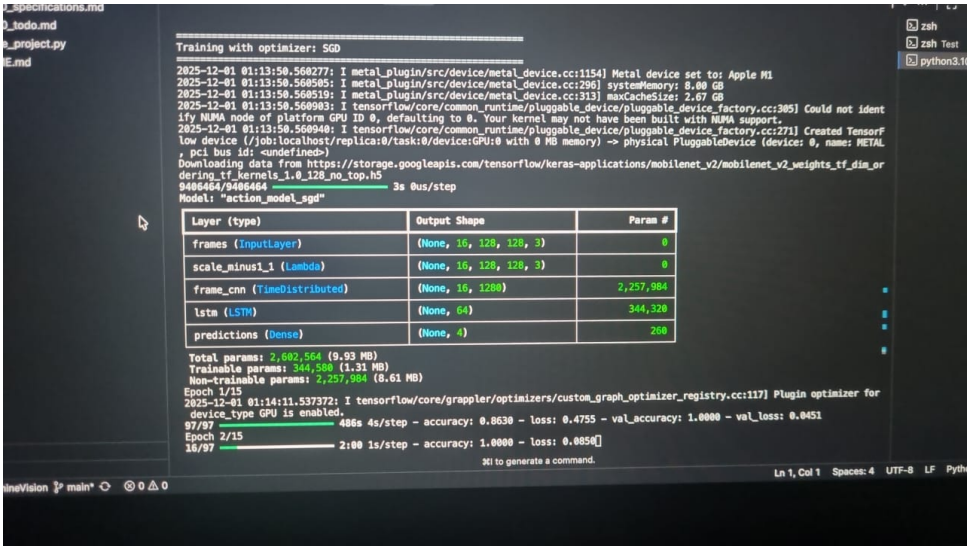

Figure 3: The proposed Hybrid CNN-LSTM Architecture showing the flow from raw frames to class prediction.

3.2 Training Setup

Training is performed using categorical cross-entropy loss with accuracy as the primary metric. The core experiment compares three standard optimizers:

- **SGD:** With momentum to stabilize convergence.
- **Adam:** Adaptive learning rates; showed the steepest initial drop in validation loss.
- **Adagrad:** Fast initial improvement followed by a plateau.

4 Experimental Results

Across all three optimizers, the CNN-LSTM action model achieved **100% test accuracy** on the held-out evaluation set. This strong result is largely attributable to the use of transfer learning from MobileNetV2, which provides highly expressive spatial features even with the backbone frozen.

To differentiate between optimizers, we examine the validation loss curves over the 15 training epochs.

```
(mecha_env) elmala7@Elmala7s-MacBook-Air src % python check_models.py
Searching for .keras files in: /Users/elmala7/Downloads/WorkSpace/Mechatronics_Projects/CSE480_MachineVision/models
Found .keras model files:
 - action_model_adagrad.keras: 11.81 MB (12385407 bytes)
 - action_model_adam.keras: 13.13 MB (13765598 bytes)
 - action_model_sgd.keras: 11.81 MB (12385365 bytes)

Loading model from: /Users/elmala7/Downloads/WorkSpace/Mechatronics_Projects/CSE480_MachineVision/models/action_model_adam.keras
2025-12-02 17:05:07.466418: I metal_plugin/src/device/metal_device.cc:1154] Metal device set to: Apple M1
2025-12-02 17:05:07.466479: I metal_plugin/src/device/metal_device.cc:296] systemMemory: 8.00 GB
2025-12-02 17:05:07.466485: I metal_plugin/src/device/metal_device.cc:313] maxCacheSize: 2.67 GB
2025-12-02 17:05:07.466652: I tensorflow/core/common_runtime/pluggable_device/pluggable_device_factory.cc:305] Could not identify NUMA node of platform GPU ID 0, default
ing to 0. Your kernel may not have been built with NUMA support.
2025-12-02 17:05:07.466671: I tensorflow/core/common_runtime/pluggable_device/pluggable_device_factory.cc:271] Created TensorFlow device (/job:localhost/replica:0/task:0
/device:GPU:0 with 0 MB memory) -> physical PluggableDevice (device: 0, name: METAL, pci bus id: <undefined>)

Model summary:
Model: "action_model_adam"
```

Layer (type)	Output Shape	Param #
frames (InputLayer)	(None, 16, 128, 128, 3)	0
scale_minus1_1 (Lambda)	(None, 16, 128, 128, 3)	0
frame_cnn (TimeDistributed)	(None, 16, 1280)	2,257,984
lstm (LSTM)	(None, 64)	344,320
predictions (Dense)	(None, 4)	260

```
Total params: 3,291,726 (12.56 MB)
Trainable params: 344,580 (1.31 MB)
Non-trainable params: 2,257,984 (8.61 MB)
Optimizer params: 689,162 (2.63 MB)

Using random test sample index: 88
Sample shape: (1, 16, 128, 128, 3)

Running model.predict on the sample...
2025-12-02 17:05:14.851983: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:117] Plugin optimizer for device_type GPU is enabled.
1/1 ━━━━━━━━━━━━━━━━━━━━ 88s 88s/step
Raw output probabilities:
  Class 0 (Walking): 0.0007
  Class 1 (Waving): 0.0003
  Class 2 (Standing): 0.9986
  Class 3 (Sitting): 0.0004

Predicted class index: 2
Predicted class name: Standing
(mecha_env) elmala7@Elmala7s-MacBook-Air src %
```

Figure 4: Training Accuracy vs. Epochs for SGD, Adam, and Adagrad optimizers.

All three optimizers ultimately converge to similar low loss values, but their convergence dynamics differ. Adam exhibited the fastest adaptation to the data distribution. Based on these observations, we select **Adam** as the preferred optimizer for the real-time phase.

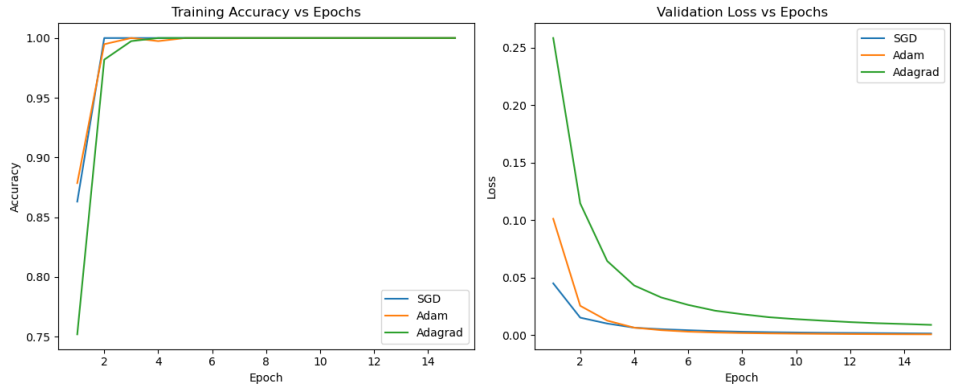

Figure 5: Validation Loss comparison showing the convergence speed of the different optimizers.