SOURCE CODE

```
import cv2

import torch

import torchvision.transforms as transforms

from torchvision.models import resnet50

import numpy as np

import torch.nn as nn

import time


# 🔹 Fix video paths (for Google Colab)

video_paths = ["/content/input1.mp4",

              "/content/input2.mp4",

              "/content/input3.mp4"]


output_path = "/content/combined_videos.avi"

summary_output_path = "/content/summarized_video.avi"


# 🔹 Load ResNet-50 for action recognition (Workaround)

resnet = resnet50(pretrained=True)

resnet.fc = nn.Linear(resnet.fc.in_features, 3)  # Adjust for 3 expected actions

resnet.eval()


# 🔹 Define action labels in *expected order*

class_names = ["Walking", "Waving", "Jumping"]


# 🔹 Function to combine multiple videos *without green labels*

def combine_videos(video_paths, output_path):

    cap1 = cv2.VideoCapture(video_paths[0])
```

```
if not cap1.isOpened():
    print("Error: Unable to open video file", video_paths[0])
    return

frame_width = int(cap1.get(cv2.CAP_PROP_FRAME_WIDTH))
frame_height = int(cap1.get(cv2.CAP_PROP_FRAME_HEIGHT))
fps = int(cap1.get(cv2.CAP_PROP_FPS))
cap1.release()

fourcc = cv2.VideoWriter_fourcc(*'XVID')
out = cv2.VideoWriter(output_path, fourcc, fps, (frame_width, frame_height))

for video_path in video_paths:
    cap = cv2.VideoCapture(video_path)
    if not cap.isOpened():
        print("Error: Unable to open video file", video_path)
        continue

    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            break

        # 🔹 No green labels, just save the video
        out.write(frame)

    cap.release()

out.release()
print("✅ Combined video created without labels!")
```

```
combine_videos(video_paths, output_path)


# 🔹 Function to summarize video with *each action at the right time*
def summarize_video(video_path, output_path):
    cap = cv2.VideoCapture(video_path)
    if not cap.isOpened():
        print("Error: Unable to open video file", video_path)
        return

    frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
    fps = 3  # Adjusted to ensure total 6 seconds summary

    fourcc = cv2.VideoWriter_fourcc(*'XVID')
    out = cv2.VideoWriter(output_path, fourcc, fps, (frame_width, frame_height))

    total_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    frames_per_video = total_frames // len(video_paths)  # Equal split per action

    frame_count = 0
    video_index = 0  # Start with Walking

    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            break

        frame_count += 1

        # 🔹 Assign action label based on the section of the video
        if frame_count <= frames_per_video:
            action = "Walking"
        elif frame_count <= 2 * frames_per_video:
            action = "Waving"
        else:
            action = "Jumping"  # Jumping only in the last section

        # 🔹 Show only RED labels in slow-motion summary
        cv2.putText(frame, action, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
        out.write(frame)

    cap.release()
    out.release()
    print("✅ 6-Second Summary Video Created!")

# 🔹 Run the summarization function
summarize_video(output_path, summary_output_path)
```