3 Track 3: Clustering-Based Action Clips Classification

3.1 Introduction

This assignment is based on **Clustering based Classification**, which includes **15 action categories**: basketball shooting, biking/cycling, diving, etc. The dataset is challenging due to variations in camera motion, object appearance, pose, scale, viewpoint, cluttered backgrounds, and illumination conditions.
You will be clustering the dataset into 15 clusters, and then for new clips, label them based on cluster labelling done in training.
You can find the dataset and kaggle competition here: Data
Use the train_set in the non-comp part

3.2 Non-Competitive Part (13 Marks)

3.2.1 Dataset Split

- **Splits:**
  - **Train:** 60% - features + labels provided.
  - **Validation:** 20% - features + labels provided.
  - **Test:** 20% - features only (labels hidden).
- **Goal:** Predict test labels using clustering-based methods, without supervised classifiers (e.g., SVM, KNN).

3.2.2 Dataset Preprocessing

- Extract frames at a fixed rate (e.g., 1 frame/sec or 5 frames/sec, justify choice in report).
- Resize frames to 224x224 pixels, normalize (e.g., mean [0.485, 0.456, 0.406], std [0.229, 0.224, 0.225] for ResNet50).
- Set a random seed (e.g., `np.random.seed(42)`) for reproducibility.

3.2.3 Feature Extraction

Extract spatiotemporal features from training videos:

1. **Required Methods:**

- **Optical Flow Magnitude:** Use Farneback method (OpenCV) on consecutive frames, compute mean and variance per video.
- **HOG Features:** Extract from keyframes (e.g., every 10th frame) using `skimage.feature.hog`, average across frames.
- **Pre-trained CNN Features:** Use ResNet50 (`torchvision`) to extract 2048D features from the last pooling layer per frame, average per video.

2. **Optional Advanced Methods (Bonus 1 mark):**
   - 3D CNN features (e.g., I3D from `torchvision`).
   - Motion boundary histograms.
3. Normalize features (e.g., L2 normalization).
4. Save features in a CSV file (video filename as index), submit with final deliverables.

3.2.4 Exploratory Data Analysis (EDA)

Analyze features:

- Visualize sample frames and optical flow maps (e.g., `matplotlib`).
- Plot feature distributions (e.g., histograms, box plots).
- Perform statistical tests (e.g., ANOVA via `scipy.stats.f_oneway`) to compare feature means across actions.
- Document observations (e.g., separability of actions).

3.2.5 Dimensionality Reduction

To reduce the dimensionality of the extracted features:

- Apply Principal Component Analysis (PCA) using inbuilt library.
- Mention the number of features chosen and justify your choice in report.
- Analyze feature importance using PCA results.

3.2.6 Clustering

- Apply **K-Means clustering** with $n_{clusters} = 15$, using library.
- Apply **Spectral Clustering**, using library.
- Apply **Agglomerative Clustering**, using library.
- For each clustering technique perform the following:
  - Evaluate clustering performance using the Adjusted Rand Index (ARI).
  - Perform hyperparameter tuning using the validation set.
- Finally compare all the clustering techniques and also make a **dendogram** for agglomerative clustering.
- Document all your observations and insights in report.

3.3 Competitive Part (7 Marks)

3.3.1 Test Set Prediction

- Submit predictions in a CSV file (format: `ID, Label`) to a Kaggle competition.
- Evaluation metrics (Kaggle leaderboard): ARI (clusters vs. hidden labels).

More details are provided on Kaggle.

3.3.2 Alternative Dimensionality Reduction Techniques

Beyond PCA, other dimensionality reduction techniques should be explored, such as:

- t-Distributed Stochastic Neighbor Embedding (t-SNE)
- Uniform Manifold Approximation and Projection (UMAP)
- Experimenting with different numbers of features retained in PCA

3.3.3 Advanced Methods

Enhance performance with:

- **Feature Fusion:** Combine optical flow, HOG, and CNN features (e.g., concatenation, weighted averaging).
- **Deep Clustering:** Use Deep Embedded Clustering (DEC) or Variational Deep Embedding (VaDE).
- **Ensemble Clustering:** Aggregate multiple clustering results (e.g., majority voting).

The grades in this section will primarily depend on the private leaderboard ranking and leaderboard score. Innovative approaches and well-documented findings will be given consideration.

3.4 Useful Suggestions

- Start Early! Feature extraction can take hours!
- Extract features in batches.
- Use platforms like Google Collab, Kaggle notebook etc. for GPUs/more computational power.

3.5 Submission Requirements

Submit:

- **Report:** LaTeX report detailing methodology, validation results, test predictions, visualisations and insights.
- **Code:** Separate files for both Comp and Non Comp (along with output). Code should be Well-commented
- Evaluations of clustering performance using ARI.
- Results of experiments conducted during hyperparameter tuning and alternative techniques.
- The csv file which you used finally (before applying PCA to it) to calculate the ARI score you are mentioning in your report.
- Final 2 submissions you chose on Kaggle for the private leaderboard. (By default these are the best 2 public leaderboard submissions.)

NOTE:

- Using Discussion Section in Kaggle for clarifying doubts is recommended, since there are lots of common queries.
- Use proper naming for all the files. Exact details will be released close to deadline. (Not following them might lead to penalty)
- Form team name and kaggle team name should be same.

In case there are some more track specific instructions/clarifications, then those will also be communicated via Kaggle Discussion section, so do check that regularly!