CSE 515 Multimedia and Web Databases

September 3, 2024

Phase #1

(Due Sept 16th 2024, midnight)

Description: In this project, you will experiment with

- video features,
- vector models, and
- similarity/distance measures

Background: In this phase of the project, we will use

- python as the programming environment,
- pytorch as the deep learning library,
- torchvision, pyav, and opencv as the visual information extraction packages, and
- you may also need numpy and scipy for array manipulation and other mathematical operations.

Familiarize yourself with these tools and languages. In particular, download and familiarize yourselves with the following pre-trained video ResNet (R3D-18) neural architecture available through the torchvision package:

- *pre-trained neural architecture:* R3D-18 (with default weights)

  `https://pytorch.org/vision/main/models/generated/torchvision.models.video.r3d_18.html#torchvision.models.video.r3d_18`

  and the human motion data set, HMDB51:

  `https://serre-lab.clps.brown.edu/resource/hmdb-a-large-human-motion-database/`

  In particular, we will also use the pre-extracted HOG/HOF (STIP) features for this data set, available at:

  `http://serre-lab.clps.brown.edu/wp-content/uploads/2013/10/hmdb51_org_stips.rar`

  Please read the following paper for the STIP features: `http://www.irisa.fr/vista/Papers/2008_cvpr_laptev.pdf`

Please see the following papers for additional information on HOG and HOF features:

- `https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf`
- `https://ieeexplore.ieee.org/document/6523794`
- `https://filebox.ece.vt.edu/~jbhuang/teaching/ece5554-4554/fa16/lectures/Lecture_23_ObjectDetection2.pdf`
- `https://personalpages.surrey.ac.uk/j.collomosse/pubs/Hu-CVIU-2013.pdf`

In this phase, you are free to store the data however you wish: you can use a relational database (such as MySql), a no-SQL database (such as MongoDB), or create your own file/data structures.

Project Tasks:

- **Task 0:** Download the HMDB51 data set as well as the corresponding STIP features.
  - The HMDB51 data set contains videos for 51 action types.
  - We will use the videos for the following 6 actions as inputs for querying, classification, and recommendation tasks:
    - cartwheel
    - drink
    - ride_bike
    - sword
    - sword_exercise
    - wave

  We will refer to these videos as the *target_videos*. Please create a *target_videos* folder and copy the corresponding videos into it.
- Videos for all the other 45 action types will be referred to as *non_target_videos*. Please create a *non_target_videos* folder and copy the corresponding videos into it.
- **Task 1:** Implement a program which, given a video file name and one of the following feature models, visualizes the video and then extracts and prints (in a human readable form) the corresponding NN-based feature descriptors:
  - *R3D18-Layer3-512*: Attach a hook to the output of "layer3" layer of the pre-trained R3D-18 architecture to obtain $256 \times 8 \times 14 \times 14$ dimensional tensor, convert this tensor to a 512 dimensional vector by averaging each $4 \times 14 \times 14$ subtensor.

* *R3D18-Layer4-512*: Attach a hook to the output of "layer4" layer of the pre-trained R3D-18 architecture to obtain $512 \times 4 \times 7 \times 7$ dimensional tensor, convert this tensor to a 512 dimensional vector by averaging each $4 \times 7 \times 7$ subtensor.
* *R3D18-AvgPool-512*: Attach a hook to the output of the "avg-pool" layer of the pre-trained R3D-18 architecture to obtain 512 dimensional vector.

Note: Some of the video files in the data set may not have three channels – if you encounter such a video, you can skip it.

– **Task 2:** In the STIP folder, for each video, we have a set of spatio-temporal interest points (STIPs):
  * *point-type:* you can ignore this
  * *x, y, t:* spatio-temporal position of the point
  * *sigma2*: spatial scale (4,8,16,32,64,128) of the interest region
  * *tau2*: temporal scale (2,4) of the interest region
  * *detector-confidence*: consider only the 400 highest confidence STIPs and ignore the rest.
  * *dscr-hog*: a 72 dimensional HoG feature descriptor
  * *dscr-hof*: a 90 dimensional HoF feature descriptor

  Task 2a: First construct 480 HoG and 480 HoF STIP cluster representatives: More specifically, for each of the 12 $\langle sigma2, tau2 \rangle$ pairs
  * Randomly sample 10000 STIPs from the *non_target_videos*.
  * Apply k-means clustering[1], with $k = 40$, to obtain the corresponding HoG and HoF cluster representatives

  Task 2b: Then, implement a program which, given a video file name, visualizes the video and then extracts and prints (in a human readable form) the corresponding STIP-based **BOF-HOG-480** bag-of-feature descriptor:
  * Create a 40 dimensional HoG STIP histogram for each of the 12 $\langle sigma2, tau2 \rangle$ pairs of the given video by assigning each of the 400 highest confidence STIPs to the closest cluster representative.
  * Concatenate the created 12 histograms to obtain a 480 dimensional HoG bag-of-features vector.

  Task 2c: Finally implement a program which, given a video file name, visualizes the video and then extracts and prints (in a human readable form) the corresponding STIP-based **BOF-HOF-480** bag-of-feature descriptor using a similar process.

– **Task 3:** Implement a program which creates *video color histograms*: Given a video file name, a resolution, $r = 4$, a color histogram size, $n = 12$,

[1] In this phase, you can use a library fro K-means clustering

1. the program selects 3 frames (first, middle, and last),
   (a) divides each selected video frame, f_j into $r \times r$ cells,
   (b) for cell c_l of the frame f_j of the video file, creates an n-bin color histogram[2], $h_{i,j,l}$ and
   (c) outputs (in an human-readable format) the corresponding **COL-HIST** feature.

$$\langle j; l; h_{i,j,l} \rangle$$

   in an human-readable format and visualizes the video.

- **Task 4:** Implement a program which extracts and stores feature descriptors for all the *target_videos* in the data set.
- **Task 5:** Implement a program which, given video file name and a value "m", returns and visualizes the most similar m videos based on each of the visual models -you will select the appropriate distance/similarity measure for each feature model. For each match, also list the corresponding distance/similarity score.

Deliverables:

- Your code (properly commented) and a README file.
- Your outputs for the provided sample inputs.
- A short report describing your work and the results.

Please place your code in a directory titled "Code", the outputs to a directory called "Outputs", and your report in a directory called "Report"; zip or tar all off them together and submit it through the digital dropbox.

[2]Note that you need to use the same color bins for all video frames in the database for results to be comparable.