CSE 515 Multimedia and Web Databases

September 19, 2024

Phase #2

(Due Oct 23rd 2024, midnight)

Description: In this project, you will experiment with

- video features,
- vector models, and
- dimensionality curse.

Important notes:

- You can use existing libraries for LDA and k-means, as well as for eigenvalue, eigenvector extraction.
- The tasks in this phase involve the feature models and similarity/distance functions developed in the previous phase.
- You may also need numpy and scipy for array manipulation and other low-level mathematical operations.
- In this phase, we will use both *target_videos* and *non_target_videos*. For this phase, the set of *target_videos* include
  - golf
  - shoot_ball
  - brush_hair
  - handstand
  - shoot_bow
  - cartwheel hit
  - shoot_gun
  - cartwheel
  - hug

  - sit
  - catch
  - jump
  - situp
  - chew
  - kick
  - smile
  - clap
  - kick_ball
  - smoke
  - climb
  - somersault
  - climb_stairs
  - laugh
  - stand

- If your group has 3 members instead of 4, please skip tasks 5 and 6.

Project Tasks:

- **Task 0a:** Implement a program which,
  - starting from 0, assigns a unique *videoID* to all videos (*target_videos* and *non_target_videos*);
  - maps the even numbered videos in the *target_videos* data set into each of the 6 visual feature spaces (R3D18-Layer3-512, R3D18-Layer4-512, R3D18-AvgPool-512, BOF-HOG-480, BOF-HOF-480, COL-HIST) from the previous phase and store the resulting data vectors – in the database[1], store not only the video filenames and feature vectors, but also the original video category labels;
  - maps the odd numbered videos in the *target_videos* data set into each of the 6 visual feature spaces from the previous phase and store the resulting data vectors – in the database, store only the video filenames and feature vectors; do not store original video category labels;
  - maps the videos in the *non_target_videos* data set into each of the 6 visual feature spaces from the previous phase and store the resulting data vectors – in the database, store only the video filenames and feature vectors; do not store original video category labels.

[1] As in the first 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.

- **Task 0b:** Implement a program which, given (a) a video file name or videoID (even or odd, target or non_target), (b) a user selected feature space, and (c) positive integer m, identifies and visualizes the most similar m even numbered videos in the *target_videos* data set, along with their scores, under the selected feature space.

- **Task 1:** Implement a program which, given (a) a query video file name or videoID (even or odd, target or non_target), (b) a user selected feature space, and (c) positive integer l, identifies and lists l most likely matching labels, along with their scores, under the selected feature space.

- **Task 2:** Implement a program which (a) given one of the feature models, (b) a user specified value of s, (c) one of the four dimensionality reduction techniques (PCA, SVD, LDA, k-means) chosen by the user, reports the top-s latent semantics extracted from the even numbered *target_videos* under the selected feature space:

  - Store the latent semantics (factor matrices and core matrix, where available) in a properly named output file.
  - List videoID-weight pairs, ordered in decreasing order of weights.

- **Task 3:** Implement a program which, given (a) a video file name or videoID (even or odd, target or non_target), (b) a user selected feature model from Task 0 or latent semantics from Task 2, and (c) positive integer m, identifies and visualizes the most similar m *target_videos*, along with their scores, under the selected model or latent space.

- **Task 4:** Implement a program which, given (a) any (target or non-target) *label*, (b) a user selected latent semantics from Task 2, and (c) positive integer m, identifies and lists m most relevant *target_videos* (even or odd), along with their scores, under the selected latent space.

- **Task 5:** Implement a program which, (a) given one of the feature models and (b) a value s,

  - creates (and saves) a label-label similarity matrix using the even numbered *target_videos*,
  - performs a user selected dimensionality reduction technique (PCA, SVD, LDA, k-means) on this label-label similarity matrix,
  - stores the latent semantics (factor matrices and core matrix, where available) in a properly named output file
  - lists label-weight pairs, ordered in decreasing order of weights.

- **Task 6:** Implement a program which, given (a) any (target or non-target) *label*, (b) a user selected feature model from Task 0 or latent semantics from Tasks 2 or 4, and (c) positive integer l, identifies and lists l most similar *target_videos* labels, along with their scores, under the selected model or latent space.

Deliverables:

- Your (properly commented) code. It is important that
  - the code implements each task as a separate piece of software that can be independently executed from the rest (assuming, of course, that that all the data prerequisites are satisfied), and
  - input values to your software are provided from command line or through a GUI – in particular, avoid designs where constants in the code have to be manually edited to run the software for a given set of input values.
- Your outputs for the provided sample inputs.
- A report describing your work and the results. The report must be informative: It needs to communicate:
  - your relevant assumptions,
  - your design (including "*how*" and "*why*"),
  - your assessment of the results and insights garnered from them.

  Your report must also include sufficient details (such as input and output formats) for me and the TA to
  - deploy and execute your deliverable and
  - interpret the results.

  A README file included with the code helps – but is not sufficient.

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 Canvas digital dropbox.