CSE 515 Multimedia and Web Databases

Phase #1

(Due Sept 15th 2019, midnight)

Description: In this project, you will experiment with

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

This project phase will be performed by each group member; but, you will get group grades:

- Each student needs to implement two out of four feature models.
- Each model must be implemented, independently, by at least two students in the group.

PROJECT TASKS:

- **Task 0:** In this project, we will use the data sets associated with the following publication:

  > *Mahmoud Afifi. "11K Hands: Gender recognition and biometric identification using a large dataset of hand images." M. Multimed Tools Appl (2019) 78: 20835.*

  The data sets are available at

  https://sites.google.com/view/11khands

  Note that we will not necessarily solve the specific problems addressed in this paper or use the associated code base. However, fee free to familiarize yourself with both.

  Download and familiarize yourselves with the following data sets provided in the web site:

  - Hand images
  - Metadata
  - Training-testing sets
  - Skin mask images

  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 to create your own file/data structures.

- **Task 1:** Implement a program which, given an image ID and one of the following models, extracts and prints (in a human readable form) the corresponding feature descriptors:

  - *Color moments, CM100x100*: Split the image into 100x100 windows, compute color moments for each window, and concatenate these color moments to obtain a unified feature descriptor. See

`https://en.wikipedia.org/wiki/Color_moments`

for the three color moments (mean, standard deviation, and skewness) to use in this phase. Note that color moments are computed separately for each color channel. Please convert images to the YUV color model before computing the moments.

– *Local binary patterns, LBP100x100*: Split the image into 100x100 windows, compute LBP features for each window, and concatenate these to obtain a unified feature descriptor. See

`https://liris.cnrs.fr/Documents/Liris-5004.pdf`

The following web site has pointers to several libraries that you can use for extracting LBP features:

`https://en.wikipedia.org/wiki/Local_binary_patterns`

Note that LBP is computed on gray scale images thus, depending on the library you pick, you may need to first convert the images to gray scale.

– *Histograms of oriented gradients, HOG*: See

`https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf`

The following web site has pointers to several libraries that you can use for extracting HOG features:

`https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients`

If needed use the following parameter settings:

∗ number of orientation bins = 9,
∗ cell size = 8,
∗ block size = 2,
∗ oriented gradients (yes/no) = no,and 0 otherwise.
∗ L2-norm clipping threshold = 0.2

Down-sample images 1-per-10 (rows and columns) before applying HOG extraction.

– *Scale-invariant feature transform, SIFT*: See

`http://www.cs.ubc.ca/~lowe/keypoints/`

for SIFT feature extraction. Note that this algorithm returns a set of vectors, each of the form

$$[x, y, scale, orientation, a_1, \ldots, a_{128}].$$

Note that, unlike the other models, where you will obtain a single feature vector for each image, when using the SIFT feature, each image will lead to a set of vectors.

- **Task 2:** Implement a program which, given a folder with images, extracts and stores feature descriptors for all the images in the folder.
- **Task 3:** Implement a program which, given an image ID, a model, and a value "k", returns and visualizes the most similar k images based on the corresponding visual descriptors. For each match, also list the overall matching 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.