BINUS University

Academic Career: ***Undergraduate / ~~Master~~ / ~~Doctoral~~ *)***			**Class Program:** ***~~International~~/Regular/~~Smart Program~~/Global Class*)***		
☐ **Mid Exam** ☑ **Final Exam** ☐ **Short Term Exam** ☐ **Others Exam : _____**			**Term : Odd/~~Even~~/~~Short~~ *)**		
☑ **Kemanggisan** ☑ **Alam Sutera** ☐ **Bekasi** ☐ **Senayan** ☐ **Bandung** ☐ **Malang**			**Academic Year :** **2020 / 2021**		
Faculty / Dept.	:	School of Computer Science	Deadline	Day / Date :	Monday / Feb 15th, 2021
				Time :	13.00
Code - Course	:	COMP7116-Computer Vision	Class	:	All Classes
Lecturer	:	Team	Exam Type	:	Online
*) *Strikethrough the unnecessary items*					
The penalty for CHEATING is DROP OUT!!!					

1. **[20%]** You are expected to implement an instance retrieval algorithm in python notebook using *bag of visual words (BOVW)* method. In order to do so, here are all the tasks that may be carried out:
   a. **[7.5%]** Collect all the *SIFT* descriptors from the image datasets provided (Dataset_P1) and cluster them using *k-Means* into *k* number of clusters where *k* is a number you set (try various different values of *k* in order to improve the precision). The center of these clusters are the "visual words" i.e. representative features of your image datasets. The output of this clustering process is then called a visual codebook or sometimes referred to as dictionary. Draw the histogram of your *k* final visual codebook and save it in *pickle* format. You may need to normalize your histogram.
   b. **[5%]** Load an image from list of image queries and extract its SIFT descriptors. Based on SIFT descriptors, find the visual words of the corresponding query image and draw the histogram of it (histogram of visual words of the query image).
   c. **[7.5%]** Based on compact image features obtained from (a) and (b), Using *cosine similarity* retrieve similar images using Q1, Q2, Q3, and Q4 respectively (you can find Q's from Dataset_P1). Make sure every single query will return 15 most similar images organized in order. Draw the average *precision vs recall curve*.

   Please do not forget to write down (PLEASE USE markdown cells) your analysis, comments, and/or info that may be needed in order to make your codes more understandable.

2. **[20%]** Using Tensorflow library and python notebook, you are supposed to classify scenes into one out of four (4) categories by training and testing on CIFAR-10 image datasets that can be downloaded from (https://www.cs.toronto.edu/~kriz/cifar.html). Your detailed tasks will be as follows:
   a. **[10%]** Build a CNN model to extract features and train a classifier on extracted features. The size of the input is (32x32x3) and the suggested architecture of your CNN is as follows: Conv(16,3x3), Relu, Conv(32,3x3), Relu, Conv(64,3x3), 2FC, and Softmax(4) (10%).
   b. **[5%]** Test your trained model to perform a 4-class classification problems (automobile, cat, dog, and truck).

Verified by,

Ajeng Wulandari (D6422) and sent to Program on Jan 18, 2021

c. **[5%]** Illustrate your classification results using a confusion matrix and report classification accuracy of your classifier.

Again, **please do not forget** to put your analysis, comments, and/or info needed in order to make your work more understandable.

3. **[20%]** You are required to find the location of an object (test1 and test2 images, Dataset_P3) in the collage image using one of your favorite local features (SIFT, SURF, ORB, etc). The tasks in chronological order are as follows:
   a. **[7.5%]** Find keypoints in the images and plot the keypoint matches. Show true and false matches with different colors.
   b. **[7.5%]** Using Direct Linear Transformation algorithm, find the homography matrix for the test images.
   c. **[5%]** Find corners of the bounding box for the test images in the collage image and draw their bounding box. Show your matching result for both test images.

   Implement all the above tasks in python notebook and put your comment/info using markdown cell in order to make your codes more understandable.

4. **[20%]** Using the images of Taj Mahal (Dataset_P4) as it is shown below, create one panoramic image using image stitching technique. Clearly write the algorithm you followed, implement your respective algorithm in python notebook and show the final combined image. You may need to install ***imutils*** python library

5. **[20%]** You are expected to implement a face detection algorithm using HOG features with linear SVM as a classifier in python notebook. So your detailed tasks will be as follows:
   a. **[5%]** Of your choice, build a dataset containing P positive images (face) and N negative images (non-face) where N>>P and save several face images for later use (testing the detector).
   b. **[7.5%]** Extract the HOG features of both positive and negative images and train using linear SVM. Once the training is completed, save the model.
   c. **[7.5%]** Test the model using sliding window, you may get overlapping windows surrounding the face image. If so, how do you get rid of these overlapping windows. Demonstrate the results after the improvement.

-- Good Luck --

Verified by,

Ajeng Wulandari (D6422) and sent to Program on Jan 18, 2021