HW2

Submit Assignment

Due Friday by 12pm **Points** 100 **Submitting** a file upload

HW2 is about matching keypoints in pairs of images for the purposes of image retrieval.

Given a query image, q, and a large dataset of images, $D = \{r_i\}$, find in D a subset $R_q \subset D$ of K most similar images to q , where $|R_q| = K \in \{1, 2, \dots\}$.

Similarity between q and $r \in D$, denoted as $s\,(q, r)$, is computed by matching their keypoints. Given a set of keypoint matches $M = \left\{\left(f_k^q\,,\ f_l^r\right)\right\}$, where f_k^q is a feature in q and f_l^r is a feature in r, $s\,(q, r)$ is defined as a total similarity of the matched keypoints:

$$s\,(q, r) = \sum_{(f_k^q\,,\ f_l^r) \in M}\ s\left(f_k^q\,,\ \ f_l^r\right)$$

In R_q, there could be true and false positives of q. We say that an image r is a true positive of q if r shows the same object as the query image q. Otherwise, r is a false positive.

We could evaluate keypoint matching by estimating precision and recall of image retrieval, defined as

Precision$(q,\ K)$ = (number of true positives of q in R_q) / K

Recall$(q,\ K)$ = (number of true positives of q in R_q) / (number of true positives of q in D)

Tasks:

- Download a set of 140 images, D, and a set of 35 query images, Q, from **image_retrieval.zip.**
- Use your code from HW1 to
  - Detect 30 strongest keypoints in every image of D and Q (e.g., Harris corners). For this, you may use your own code for keypoint detection, or our Pytorch code that we provided as solution for the first part of HW1.
  - Compute a deep feature descriptor f_k^q of every keypoint detected in query images $q \in Q$, and a deep feature descriptor f_l^r of every keypoint detected in images $r \in D$. For this, you may use your best CNN architecture that produced the lowest training loss in your HW1, or our Pytorch code that we provided as solution for the second part of HW1. Note that your image matching and subsequent image retrieval will give poor results if your feature descriptors are poorly estimated. Therefore, we highly recommend that you either use our released Pytorch code, or at least use the same hyper parameters of our CNN for computing feature descriptors.

- For every query image $q \in Q$
  - Match q to every image $r \in D$ by matching their keypoints, using
    - One-to-one keypoint matching:
      1. Compute the cost matrix, $C = [c_{kl}]$, of matching feature pairs (f_k^q, f_l^r) from the two images, where elements of C are defined as $c_{kl} = \| f_k^q - f_l^r \|_2$
      2. Use the Hungarian algorithm with C as input to compute keypoint matches: $M^{\text{one2one}} = \{(f_k^q, f_l^r)\}$
    - Many-to-many keypoint matching:
      1. Compute the similarity vector, $s = [s_{kl}]$, of matching feature pairs (f_k^q, f_l^r) from the two images, where elements of s are defined as $s_{kl} = \exp(-c_{kl}) = \exp(-\| f_k^q - f_l^r \|_2)$
      2. Solve x in the following optimization problem: maximize $s^\top x$, s.t. $\|x\|_2 = 1$, and $x \geq 0$
      3. Use x to find the matching result: $M^{\text{many2many}} = \{(f_k^q, f_l^r)\}$
  - For the one-to-one matching result, compute similarity between q and r as: $s^{\text{one2one}}(q, r) = \sum_{(f_k^q, f_l^r) \in M^{\text{one2one}}} s(f_k^q, f_l^r)$
  - For the many-to-many matching result, compute similarity between q and r as: $s^{\text{many2many}}(q, r) = \sum_{(f_k^q, f_l^r) \in M^{\text{many2many}}} s(f_k^q, f_l^r)$
  - For K=1, 2, 3, 4
    - Identify R_q^{one2one} and $R_q^{\text{many2many}}$, i.e., the two sets of K most similar images to q with respect to the two matching criteria
    - For the two matching criteria, use ground truth, given in image_retrieval.zip, to estimate:
      - Precision: $P^{\text{one2one}}(q, K)$ and $P^{\text{many2many}}(q, K)$
      - Recall: $R^{\text{one2one}}(q, K)$ and $R^{\text{many2many}}(q, K)$
- For the two matching criteria, compute average:
  - Precision: $P^{\text{one2one}}(K) = \frac{1}{|Q|} \sum_{q \in Q} P^{\text{one2one}}(q, K)$ and $P^{\text{many2many}}(K) = \frac{1}{|Q|} \sum_{q \in Q} P^{\text{many2many}}(q, K)$
  - Recall: $R^{\text{one2one}}(K) = \frac{1}{|Q|} \sum_{q \in Q} R^{\text{one2one}}(q, K)$ and $R^{\text{many2many}}(K) = \frac{1}{|Q|} \sum_{q \in Q} R^{\text{many2many}}(q, K)$

What to Turn in?

A zipped folder that contains the following files:

1. (70 points) A pdf file with a maximum page-limit of 2 pages that provides the following details:
   - Type of keypoints that you detected in the images,
   - Hyper-parameters of the CNN architecture for computing feature descriptors,
   - Two recall-precision curves of image retrieval, where each curve uses the corresponding results:

$\{(P^{\text{one2one}}(K), R^{\text{one2one}}(K)) : K = 1, 2, 3, 4\}$ or
$\{(P^{\text{many2many}}(K), R^{\text{many2many}}(K)) : K = 1, 2, 3, 4\}$

2. (20) Two matrices of image similarities with size 35 x 140 for the 35 query images and 140 images from the dataset D. Elements of the two matrices are $s^{\text{one2one}}(q, r)$ and $s^{\text{many2many}}(q, r)$ corresponding to the two matching criteria.
3. (10) Tensor of feature descriptors with size 175 x 30 x 128, for a total of 175 images where the first 35 are query images and the remaining 140 are images from the dataset D, and for the 30 keypoints detected in each image, and for the 128-dimensional feature descriptors

Grading

- 60 points = If you submit everything from the above list.
- 10 points = If your recall and precision are accurately computed from the two 35 x 140 matrices of image similarities that you submitted.
- 20 points = If the two 35 x 140 matrices of image similarities that you submitted are accurate.
- 10 points = If the 175 x 30 x 128 tensor of feature descriptors that you submitted is accurate.