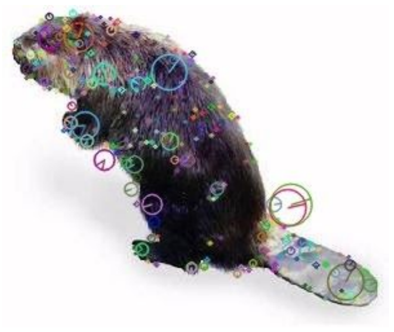
VIP :: Assignment 4: CBIR

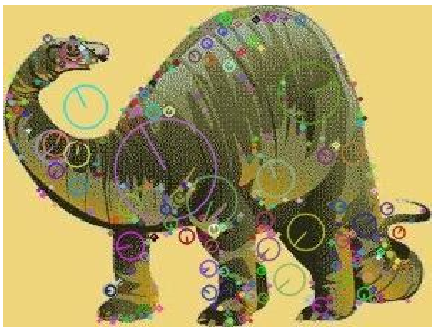
Olga Iarygina (hwk263), Ioannis Manasidis (jdv382), Marina Pinzhakova (ptr273)

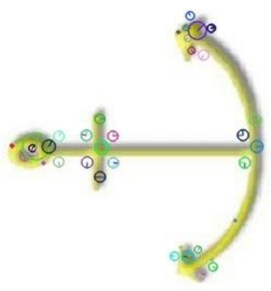
In this assignment we implement a prototypical CBIR system with the CalTech101 image dataset. We decided to use the following image categories: anchor, ant, barrel, bass, beaver, bonsai, brain, brontosaurus 🦕, buddha, camera, ceiling_fan, and watch.

To perform the task we split the dataset images into a train and a test set, sized equally and then extracted visual words from SIFT descriptors. SIFT works in the following way: the main point in the detection of keypoints is the construction of a DoG (difference of gaussians) pyramid to approximate the LoG. We first search for features that seemingly become more and more localized as the blur intensity increases, and afterwards look for max and min values. The next step consists of setting a radius value, based on which we compare the pixels around the min/max values, looking for features that are consistent throughout different scales. So, scale invariance is achieved by finding key points for the original image taken at different scales.

After we have deduced that some point is key, we calculate its orientation. A point can have several directions, and the direction of the key point is calculated based on the direction of the gradients of the points adjacent to the feature. We find the direction of the key point from the direction histogram. The direction of the key point lies in the interval covered by the max component of the histogram. Below there are a few examples of key points that we detected for the images data we used. We can see that the key points are located in the place of clearly prominent features of the object - borders, corners, spots and so on.







In the SIFT method, the descriptor is a vector calculated on the local gradient data weighted by a Gaussian. Before calculating the descriptor, this window is rotated by the angle of the key point direction, which achieves the invariance with respect to rotation. To construct a SIFT descriptor we put the grid over the neighborhood. Then each grid is divided into subgrids and at the center of each subgrid we compute gradient estimates.Then we accumulate them into an orientation histogram. All orientation histograms are concatenated to form a feature vector which is then normalized.

Throughout the assignment, we used the OpenCV library to calculate the SIFT descriptors and the sklearn library to generate the bags of words using KMeans.

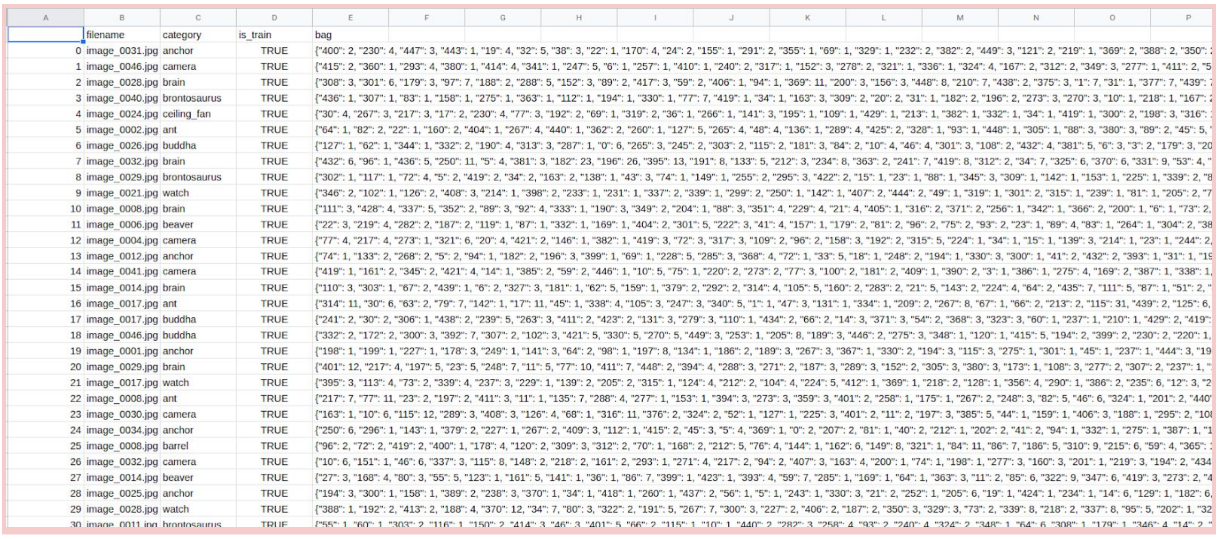
Part 1. Codebook generation

In this step we first extracted SIFT features from the set of training images, using OpenCV's SIFT model implementation. It ignores image scale and rotation and makes the descriptor scale-invariant. Each descriptor is represented as a 128-column row in the resulting matrix. To extract visual word clusters we used k-means. K-means is a method of clustering, which splits the set of elements of the vector space into a known number of clusters k. It seeks to minimize the standard deviation at the points of each cluster. At each iteration, the center of mass for each cluster obtained in the previous step is recalculated, then the vectors are divided into clusters again in accordance with which of the new centers is closer to the chosen metric. The algorithm ends when no cluster changes occur at some iteration or the maximum number of iterations has been reached.
After some experiments, we decided to use 500 clusters, which means that we get 500 visual words that perform as features for image retrieval and make up our "vocabulary". With larger amounts of clusters we noticed an accuracy drop.

We clustered the descriptors using MiniBatchKMeans from Scikit-learn. This allowed us to create bags of visual words (BoW), i.e. unordered sets of prototypical representatives of descriptor groups. A BoW for an image consists of a dictionary, where each word (cluster number from 0 to k) is a key and the total occurrences of that key for the image is the corresponding value. If a word/key has an occurrence of 0, it is omitted and not saved in the csv file.

Part 2. Indexing

In this step, we generate an indexing table for the subsequent usage in image retrieval experiments. The table consists of 566 entries that correspond to the number of images used and include their names, categories, sets (train/test), and bags of words. Below is an example table that was generated after indexing and loaded in a datasheet.



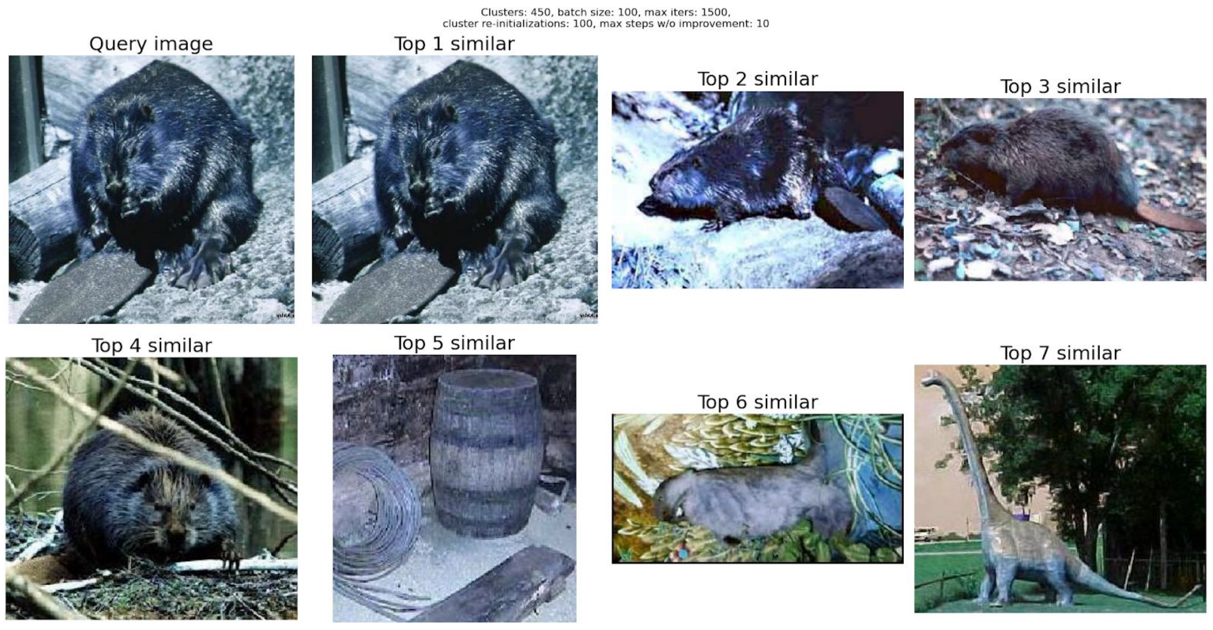
A	B	C	D	E
	filename	category	is_train	bag
0	image_0031.jpg	anchor	TRUE	{"400": 2, "230": 4, "447": 3, "443": 1, "19": 4, "32": 5, "38": 3, "22": 1, "170": 4, "24": 2, "155": 1, "291": 2, "355": 1, "69": 1, "329": 1, "232": 2, "382": 2, "449": 3, "121": 2, "219": 1, "369": 2, "388": 2, "350":
1	image_0046.jpg	camera	TRUE	{"415": 2, "360": 1, "293": 4, "380": 1, "414": 4, "341": 1, "247": 5, "6": 1, "257": 1, "410": 1, "240": 2, "317": 1, "152": 3, "278": 2, "321": 1, "336": 1, "324": 4, "167": 2, "312": 2, "349": 3, "277": 1, "411": 2, "5
2	image_0028.jpg	brain	TRUE	{"308": 3, "301": 6, "179": 3, "97": 7, "188": 2, "288": 5, "152": 3, "89": 2, "417": 3, "59": 2, "406": 1, "94": 1, "369": 11, "200": 3, "156": 3, "448": 8, "210": 7, "438": 2, "375": 3, "1": 7, "31": 1, "377": 7, "439":
3	image_0040.jpg	brontosaurus	TRUE	{"436": 1, "307": 1, "83": 1, "158": 1, "275": 1, "363": 1, "112": 1, "194": 1, "330": 1, "77": 7, "419": 1, "34": 1, "163": 3, "309": 2, "20": 2, "31": 1, "182": 2, "196": 2, "273": 3, "270": 3, "10": 1, "218": 1, "167":
4	image_0024.jpg	ceiling_fan	TRUE	{"30": 4, "267": 3, "217": 3, "17": 2, "230": 4, "77": 3, "192": 2, "69": 1, "319": 2, "36": 1, "266": 1, "141": 3, "195": 1, "109": 1, "429": 1, "213": 1, "382": 1, "332": 1, "34": 1, "419": 1, "300": 2, "198": 3, "316":
5	image_0002.jpg	ant	TRUE	{"64": 1, "82": 2, "22": 1, "160": 2, "404": 1, "267": 4, "440": 1, "362": 2, "260": 1, "127": 5, "265": 4, "48": 4, "136": 1, "289": 4, "425": 2, "328": 1, "93": 1, "448": 1, "305": 1, "88": 3, "380": 3, "89": 2, "45": 5,
6	image_0026.jpg	buddha	TRUE	{"127": 1, "62": 1, "344": 1, "332": 2, "190": 4, "313": 3, "287": 1, "0": 6, "265": 3, "245": 2, "303": 2, "115": 2, "181": 3, "84": 2, "10": 4, "46": 4, "301": 3, "108": 2, "432": 4, "381": 5, "6": 3, "3": 2, "179": 3, "20
7	image_0032.jpg	brain	TRUE	{"432": 6, "96": 1, "436": 5, "250": 11, "5": 4, "381": 3, "182": 23, "196": 26, "395": 13, "191": 8, "133": 5, "212": 3, "234": 8, "363": 2, "241": 7, "419": 8, "312": 2, "34": 7, "325": 6, "370": 6, "331": 9, "53": 4,
8	image_0029.jpg	brontosaurus	TRUE	{"302": 1, "117": 1, "72": 4, "5": 2, "419": 2, "34": 2, "163": 2, "138": 1, "43": 3, "74": 1, "149": 1, "255": 2, "295": 3, "422": 2, "15": 1, "23": 1, "88": 1, "345": 3, "309": 1, "142": 1, "153": 1, "225": 1, "339": 2, "8
9	image_0021.jpg	watch	TRUE	{"346": 2, "102": 1, "126": 2, "408": 3, "214": 1, "398": 2, "233": 1, "231": 1, "337": 2, "339": 1, "299": 2, "250": 1, "142": 1, "407": 2, "444": 2, "49": 1, "319": 1, "301": 2, "315": 1, "239": 1, "81": 1, "205": 2, "7
10	image_0008.jpg	brain	TRUE	{"111": 3, "428": 4, "337": 5, "352": 2, "89": 3, "92": 4, "333": 1, "190": 3, "349": 2, "204": 1, "88": 3, "351": 4, "229": 4, "21": 4, "405": 1, "316": 2, "371": 2, "256": 1, "342": 1, "366": 2, "200": 1, "6": 1, "73": 2,
11	image_0006.jpg	beaver	TRUE	{"22": 3, "219": 4, "282": 2, "187": 2, "119": 1, "87": 1, "332": 1, "169": 1, "404": 2, "301": 5, "222": 3, "41": 4, "157": 1, "179": 2, "81": 2, "96": 2, "75": 2, "93": 2, "23": 1, "89": 4, "83": 1, "264": 1, "304": 2, "38
12	image_0004.jpg	camera	TRUE	{"77": 4, "217": 4, "273": 1, "321": 6, "20": 4, "421": 2, "146": 1, "382": 1, "419": 3, "72": 3, "317": 3, "109": 2, "96": 2, "158": 3, "192": 2, "315": 5, "224": 1, "34": 1, "15": 1, "139": 3, "214": 1, "23": 1, "244": 2,
13	image_0012.jpg	anchor	TRUE	{"74": 1, "133": 2, "268": 2, "5": 2, "94": 1, "182": 2, "196": 3, "399": 1, "69": 1, "228": 5, "285": 3, "368": 4, "72": 1, "33": 5, "18": 1, "248": 2, "194": 1, "330": 3, "300": 1, "41": 2, "432": 2, "393": 1, "31": 1, "19
14	image_0041.jpg	camera	TRUE	{"419": 1, "161": 2, "345": 2, "421": 4, "14": 1, "385": 2, "59": 2, "446": 1, "10": 5, "75": 1, "220": 2, "273": 2, "77": 3, "100": 2, "181": 2, "409": 1, "390": 2, "3": 1, "386": 1, "275": 4, "169": 2, "387": 1, "338": 1,
15	image_0014.jpg	brain	TRUE	{"110": 3, "303": 1, "67": 2, "439": 1, "6": 2, "327": 3, "181": 1, "62": 5, "159": 1, "379": 2, "292": 2, "314": 4, "105": 5, "160": 2, "283": 2, "21": 5, "143": 2, "224": 4, "64": 2, "435": 7, "111": 5, "87": 1, "51": 2,
16	image_0017.jpg	ant	TRUE	{"314": 11, "30": 6, "63": 2, "79": 7, "142": 1, "17": 11, "45": 1, "338": 4, "105": 3, "247": 3, "340": 5, "1": 1, "47": 3, "131": 1, "334": 1, "209": 2, "267": 8, "67": 1, "66": 2, "213": 2, "115": 31, "439": 2, "125": 6,
17	image_0017.jpg	buddha	TRUE	{"241": 2, "30": 2, "306": 1, "438": 2, "239": 5, "263": 3, "411": 2, "423": 2, "131": 3, "279": 3, "110": 1, "434": 2, "66": 2, "14": 3, "371": 3, "54": 2, "368": 3, "323": 3, "60": 1, "237": 1, "210": 1, "429": 2, "419":
18	image_0046.jpg	buddha	TRUE	{"332": 2, "172": 2, "300": 3, "392": 7, "307": 2, "102": 3, "421": 5, "330": 5, "270": 5, "449": 3, "253": 1, "205": 8, "189": 3, "446": 2, "275": 3, "348": 1, "120": 1, "415": 5, "194": 2, "399": 2, "230": 2, "220": 1,
19	image_0001.jpg	anchor	TRUE	{"198": 1, "199": 1, "227": 1, "178": 3, "249": 1, "141": 3, "64": 2, "98": 1, "197": 8, "134": 1, "186": 2, "189": 3, "267": 3, "367": 1, "330": 2, "194": 3, "115": 3, "275": 1, "301": 1, "45": 1, "237": 1, "444": 3, "19
20	image_0029.jpg	brain	TRUE	{"401": 12, "217": 4, "197": 5, "23": 5, "248": 7, "11": 5, "77": 10, "411": 7, "448": 2, "394": 4, "288": 3, "271": 2, "187": 3, "289": 3, "152": 2, "305": 3, "380": 3, "173": 1, "108": 3, "277": 2, "307": 2, "237": 1,
21	image_0017.jpg	watch	TRUE	{"395": 3, "113": 4, "73": 2, "339": 4, "237": 3, "229": 1, "139": 2, "205": 2, "315": 1, "124": 4, "212": 2, "104": 4, "224": 5, "412": 1, "369": 1, "218": 2, "128": 1, "356": 4, "290": 1, "386": 2, "235": 6, "12": 3, "2
22	image_0008.jpg	ant	TRUE	{"217": 7, "77": 11, "23": 2, "197": 2, "411": 3, "11": 1, "135": 7, "288": 4, "277": 1, "153": 1, "394": 3, "273": 3, "359": 3, "401": 2, "258": 1, "175": 1, "267": 2, "248": 3, "82": 5, "46": 6, "324": 1, "201": 2, "440'
23	image_0030.jpg	camera	TRUE	{"163": 1, "10": 6, "115": 12, "289": 3, "408": 3, "126": 4, "68": 1, "316": 11, "376": 2, "324": 2, "52": 1, "127": 1, "225": 3, "401": 2, "11": 2, "197": 3, "385": 5, "44": 1, "159": 1, "406": 3, "188": 1, "295": 2, "10
24	image_0034.jpg	anchor	TRUE	{"250": 6, "296": 1, "143": 1, "379": 2, "227": 1, "267": 2, "409": 3, "112": 1, "415": 2, "45": 3, "5": 4, "369": 1, "0": 2, "207": 2, "81": 1, "40": 2, "212": 1, "202": 2, "41": 2, "94": 1, "332": 1, "275": 1, "387": 1, "1
25	image_0008.jpg	barrel	TRUE	{"96": 2, "72": 2, "419": 2, "400": 1, "178": 4, "120": 2, "309": 3, "312": 2, "70": 1, "168": 2, "212": 5, "76": 4, "144": 1, "162": 6, "149": 8, "321": 1, "84": 11, "86": 7, "186": 5, "310": 9, "215": 6, "59": 4, "365":
26	image_0032.jpg	camera	TRUE	{"10": 6, "151": 1, "46": 6, "337": 3, "115": 8, "148": 2, "218": 2, "161": 2, "293": 1, "271": 4, "217": 2, "94": 2, "407": 3, "163": 4, "200": 1, "74": 1, "198": 1, "277": 3, "160": 3, "201": 1, "219": 3, "194": 2, "434
27	image_0014.jpg	beaver	TRUE	{"27": 3, "168": 4, "80": 3, "55": 5, "123": 1, "161": 5, "141": 1, "36": 1, "86": 7, "399": 1, "423": 1, "393": 4, "59": 7, "285": 1, "169": 1, "64": 1, "363": 3, "11": 2, "85": 6, "322": 9, "347": 6, "419": 3, "273": 2, "4
28	image_0025.jpg	anchor	TRUE	{"194": 3, "300": 1, "158": 1, "389": 2, "238": 3, "370": 1, "34": 1, "418": 1, "260": 1, "437": 2, "56": 1, "5": 1, "243": 1, "330": 3, "21": 2, "252": 1, "205": 6, "19": 1, "424": 1, "234": 1, "14": 6, "129": 1, "182": 6,
29	image_0028.jpg	watch	TRUE	{"388": 1, "192": 2, "413": 2, "188": 4, "370": 12, "34": 7, "80": 3, "322": 2, "191": 5, "267": 7, "300": 3, "227": 2, "406": 2, "187": 2, "350": 3, "329": 3, "73": 2, "339": 8, "218": 2, "337": 8, "95": 5, "202": 1, "32
30	image_0011.jpg	brontosaurus	TRUE	{"55": 1, "60": 1, "303": 2, "116": 1, "150": 2, "414": 3, "46": 3, "401": 5, "66": 2, "115": 1, "10": 1, "440": 2, "282": 3, "258": 4, "93": 2, "240": 4, "324": 2, "348": 1, "64": 6, "308": 1, "179": 1, "346": 4, "14": 2,

Part 3. Retrieving

3.1 Common words

One of the first and most simple metrics in retrieving the photos is the "common words" similarity measure, during which for each query image we count the number of common words relative to every image in the train set, choosing the category of the one with the most counts as our final prediction.

Common words similarity on train set:


Clusters: 450, batch size: 100, max iters: 1500,
cluster re-initializations: 100, max steps w/o improvement: 10
Query image
Top 1 similar
Top 2 similar
Top 3 similar
Top 4 similar
Top 5 similar
Top 6 similar
Top 7 similar

In this case:

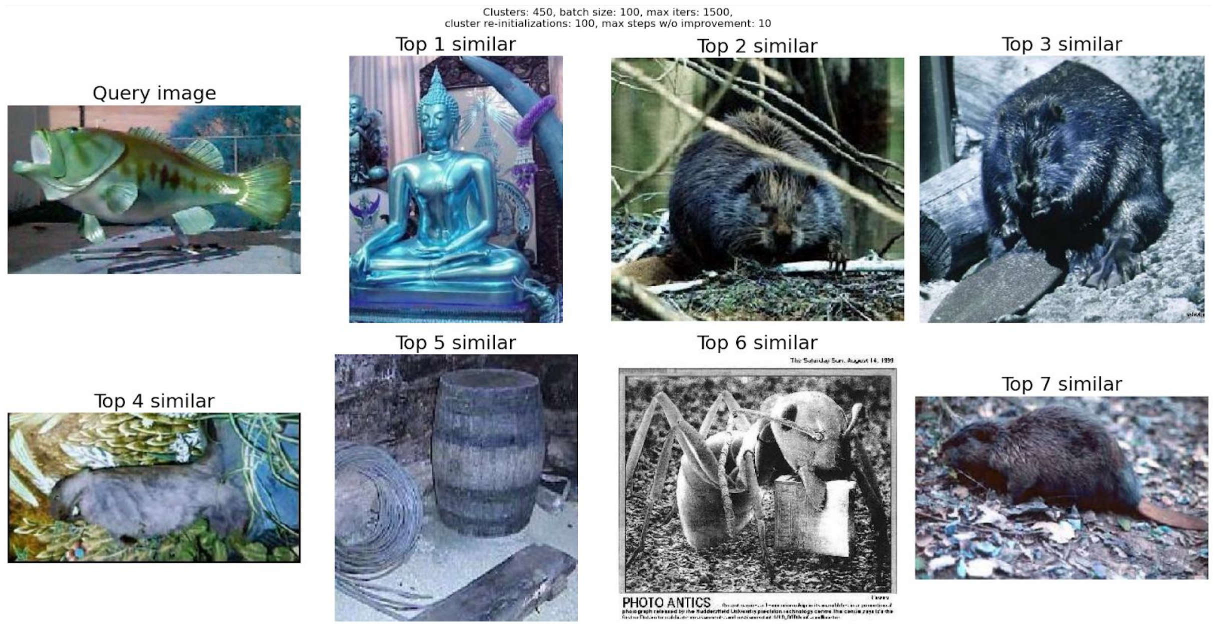
The accuracy (correct prediction percentage) was 100%

The mean reciprocal rank was 1

The correct category was in the top-3 for 100% of cases.

So, on the train set the system works pretty well.

Common words similarity on test set:



In this case:

The **accuracy** (correct prediction percentage) was 9.5406%

The **mean reciprocal rank** was 0.2083.

The correct category was in the **top-3** for 33.9223% of cases.

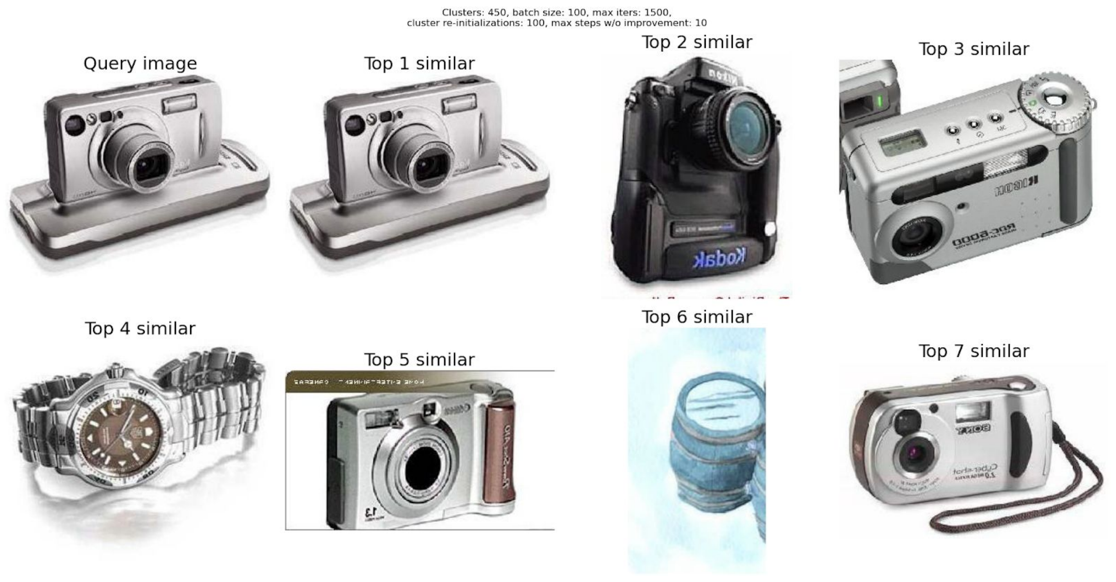
We can see pretty low accuracy. Nevertheless, if you look at the example bove, it is more or less explicable. The system can recognize images as similar because, for instance, top-1 buddha is also iridescent like a query fish and has similar items in the background. Talking about beavers, in the pictures presented, they have a contour similar to that of a fish, their fur glints, and the grass and logs in the background are similar to the bottom and trees of a fish.

3.2 TF-IDF

In this part we tested our retrieval system with the TF-IDF similarity measure. TF-IDF stands for “Term Frequency-Inverse Document Frequency”. This is a simple and convenient way to assess the importance of a term for a document in relation to all other documents. The principle is the following: if a word occurs frequently in a document, while rarely found in all other documents - this word is of great importance for that very document.

We first calculate the TF-IDF values for every word in each BoW of every training image. The result is a 2D array, where each row is a vector with the TF-IDF values of a training image’s BoW. We then calculate the TF-IDF word vector for our query image and check how similar it is to each training image (row of the 2D array). We used the cosine similarity method for choosing the most similar BoW and predicted the image category according to that.

TF-IDF similarity on train set:

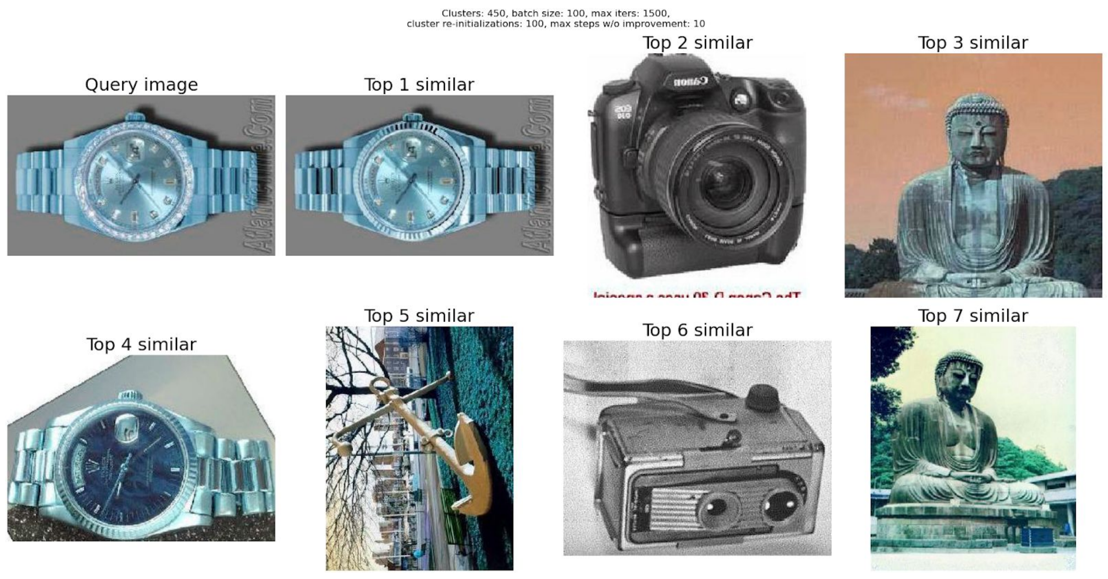

Clusters: 450, batch size: 100, max iters: 1500,
cluster re-initializations: 100, max steps w/o improvement: 10

In this case:

The **accuracy** (correct prediction percentage) was 100%
The **mean reciprocal rank** was 1
The correct category was in the **top-3** for 100% of cases.

In the example above there is an interesting case where the system is mistaken. In top-4 similar we can see a watch whose dial is like a camera lens. The watch lies in such a way that its shape really resembles a camera, and the dial is located relative to the strap in the same way as the lens relative to the camera body. If I looked at this picture without glasses, I might also confuse them.

TF-IDF similarity on test set:


Clusters: 450, batch size: 100, max iters: 1500,
cluster re-initializations: 100, max steps w/o improvement: 10

In this case:

> The **accuracy** (correct prediction percentage) was 38.5159%
> The **mean reciprocal rank** was 0.5090
> The correct category was in the **top-3** for 61.4841% of cases.

In this case the accuracy is much better than in the case with common words. We assume, that in the example, a camera is in top-2, again, because the lens has the same shape as the dial. If you look at a Buddha, his head is similar in shape to a watch strap, and the extension to the shoulders follows the curves as it moves from the watch strap to the dial.

References:

1. Forsyth, D., & Ponce, J. (2011). Computer Vision - A Modern Approach, Second Edition.
2. Lowe, D. (2004) Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60, 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94
3. https://docs.opencv.org/master/da/df5/tutorial_py_sift_intro.html