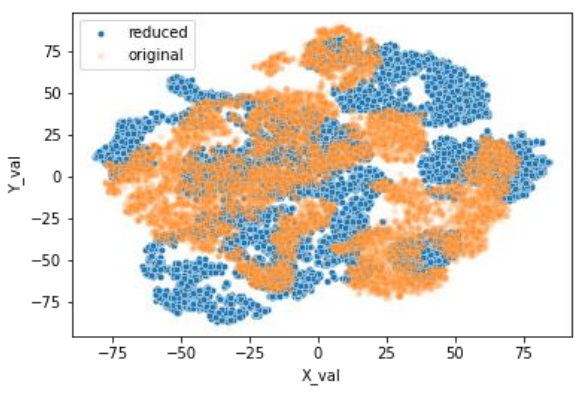
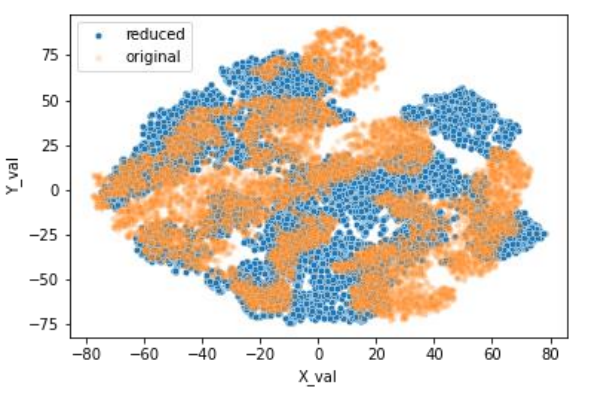


CNN and TSNE based Dimension Reduction



Auto encoder and TSNE based Dimension Reduction

Task 2: The Devil Is in The Detail (50 points)

You successfully solved the first problem and rescued all the wardrobes. Five days later, Sherlock receives an email from Moriarty, which says, “Hello Sherlock, the first game was just a warm-up. This time, I assure you it will be more interesting. I have sent you a bunch of satellite images and I have encoded a 5-letter secret code in them, one alphabet in each of the randomly chosen 5 images. Since you are not that smart to solve it, I have also sent you five images, which are similar to the five images containing the code. I have full confidence that you will fail this time. If you are still wondering what the code might be, that code will give you away my next move in ending you and your little doctor friend. Good Luck!” Since you helped solve the first problem, Sherlock trusts you with this too. You need to search the five images containing code with the help of the given query images. Second time is the charm. Do not let Sherlock down. You can download the database & the five query images at here. Submit the work done (Notebook) to search for these images and the final five images that you think contain the code. Do you know what the code is now?

Note: For all the two tasks, feel free to use any APIs that you see fit. You should document your solutions (and the APIs you used) in detail but concisely as a PDF, which will be used for grading. Your code will be used as an evidence to support your solution document.

Solution:

In this notebook, the SIFT library is used to detect and match features of two sets of images. The dataset contains two sets of images, as per the folders 'database' and 'queries'. These sets of images both relate to satellite images. By visualizing the 'database' dataset, it can be said that a larger satellite picture has been used and smaller overlapping blocks of the image has been cropped out. Before cropping out, alphabets were printed on them. When going through the dataset, instead of finding one single image for one alphabet, ending up finding a continuous series of some images where the alphabet slowly appears and then disappears, travelling horizontally in the real larger satellite image. Also at an offset of 260 images from the blocks where the alphabet is found, the alphabet itself is seen to shift vertically. Therefore, every image represents an overlapping patch from a bigger image. In this Task, it is to be found out that which image from the 'database' actually matches the most with the images from the database. Around that image patch, the alphabet should be present. One

will not get the exact match to the image containing the alphabet as the watermark itself changes the features of the image. Therefore, one will get an image very close to it.

In this Task, first one needs to extract features from all the images present in the 'database' and then compare all those feature arrays with the extracted feature arrays from the 'queries' images. Then after, by Lowe's paper, good matches should be considered only. With that consideration, the percentage of total number of matches with respect to the total number of features taken in consideration is calculated. Finally, taking the highest matching percentage of the good features and relating it to the image file that was considered for feature extraction brings the result. From this relation, getting the most probable image to match with the concerned query image. The procedure is repeated for all the query images. Finally, an added shift is introduced in the image number so that the alphabets are visible.

The data is visualized as:

It is discussed above that with every passing image, one moves in the horizontal direction

(until 259 steps) as the images are overlapping. In addition, with every 260th image, one moves in the vertical direction with respect to the very first image. Such a matrix of small part of the larger picture is plotted in matrix form for visualization purpose above.

The primary function of the Task is dealt with, which is matching the images from the 'database' to the concerned image from 'queries'. Firstly, the images are read in grey scale that is both the query image and the database image set. Thereafter, SIFT (Scale invariant feature transform) library is created from Open CV for detecting and computing features for images. Then onwards, Flann based matcher is initialized with index and search parameters. Using SIFT, features were detected and using Flann, a KNN-Matching was done between the query image and 'database' images, to find out the features and matches. However, one is only concerned with the good matches. Therefore, following Lowe's paper, masking was done to consider only good matches. Thereafter, from the number of good matches and the number of key points detected, the match percentage was calculated and stored in order. In addition, the filenames were stored in order depending upon detection from local host. Each of the 5 query images are fed to the function 'extract_features_and_find_match' and in return two lists of the match percentage for every image in the database and their corresponding file names are received in return.

The index of the element with highest value in the percentage list is found, as it obviously corresponds to the best match and the filename of that match is extracted from the filename list. There onwards, the number of image it is in the 'database' folder is extracted and that number is incremented by 523. As discussed previously, with every 260 images, we get the same image shifting vertically and with every other image; we see the same image shifting horizontally. Therefore, to view the alphabets and have most part of the query image in the matched image, 523 was added. The distribution being 260x2=520 for vertical displacement and 3 was chosen for horizontal displacement. The vertical displacement coefficient 2 and horizontal displacement coefficient 3 was chosen by trial and error method.

The results obtained were:

```
Best match for query 1: image1373.png
Adjusted match with alphabet for query 1: image1896.png
Best match for query 2: image2622.png
Adjusted match with alphabet for query 2: image3145.png
Best match for query 3: image6051.png
Adjusted match with alphabet for query 3: image6574.png
Best match for query 4: image26588.png
Adjusted match with alphabet for query 4: image27111.png
Best match for query 5: image13935.png
Adjusted match with alphabet for query 5: image14458.png
```

Finally, the query and adjusted matches with alphabet were plotted as results:

With this, we get to know that the alphabets for the message are “E”, “U”, “R”, “A” and “S” respectively for the five query images provided.