# Assignment-2

Name: Meetakshi Setiya
Roll no: 2019253

**1.**

Resources used:

- https://github.com/backseason/PoolNet : For the image dataset a subset (199) of which used in question 1 and their corresponding saliency maps found using Poolnet Saliency Detection Model (DL-based method).
- Histogram Based Contrast method of salient region detection implemented in part a (equation 3) of question 1 in Assignment 1 to find salient regions of the above image dataset by non-DL based method.

Assumptions:

- The RGB image dataset used in this question is in the "inputs" folder under the name "Images-Q1", the corresponding saliency maps found by DL and non-DL based methods are in the folders "DL-Saliency" and "Non-DL-Saliency" respectively.
- The code assumes that the names of the corresponding RGB image, DL-based saliency map and non-DL based saliency map are all the same.
- The γ (gamma) chosen for calculating ϕ(S) Separation Measure implementation was taken to be **256** since I chose 256 bins to represent the foreground and background distributions.
- The output directory has been created.

Input:

- Call the function WriteSaliencyMetrics() with paths to "Images-Q1", "DL-Saliency", "Non-DL-Saliency" and the "outputs" folder as separate strings. The function will return the Separation and Concentration Measures for the 199 non-DL based saliency maps and the Separation and Concentration Measures for the 199 non-DL-based saliency maps in that order.
- Call the function SignificanceTesting() once with the Separation Measures array of DL and non-DL based saliency maps and once for the Concentration

Measures. The output will be the result of the T-test and Wilcoxon rank sum test.

Outputs:

- The output is a csv file containing the image's filename and the corresponding Separation and Concentration Measures for the DL-based and non-DL based saliency maps. This data is also outputted at the Matlab console.
- The results of the statistical tests.

Assessment of the two saliency map methods:

- After performing the t-test on Separation Measure for the saliency maps of 199 images, the t-test rejected the null hypothesis for 5% significance levels without assuming unequal variances. Moreover, the p-value for the distributions is also very small (5.315446e-138). This low p-value shows that the results are replicable and the two distributions are statistically different.
- The Wilcoxon rank sum test on the two Separation Measures shows that the null hypothesis at the default 5% significance level has been rejected. Thus, for randomly selected values *X* and *Y* from the two populations, the probability of *X* being greater than *Y* is not equal to the probability of *Y* being greater than *X.*
- After performing the t-test on Concentration Measure for the saliency maps of 199 images, the t-test rejected the null hypothesis for 5% significance levels without assuming unequal variances. Moreover, the p-value for the distributions is also very small (6.567233e-92). This low p-value shows that the results are replicable and the two distributions are statistically different.
- The Wilcoxon rank sum test on the two Concentration Measures shows that the null hypothesis at the default 5% significance level has been rejected. Thus, for randomly selected values *X* and *Y* from the two populations, the probability of *X* being greater than *Y* is not equal to the probability of *Y* being greater than *X.*
- Mean value of the concentration measures and the separation measures is higher in case of DL-based saliency maps. **This shows that from the given quality measure, the DL-based method of saliency detection is better.**

**2.**

Assumptions and caveats:

- All images will be grayscale and resized to 256x256.
- The patch sizes used for creating SPP features would be 64x64, 128x128 and 256x256.
- The output folder is assumed to be ../../outputs/ and that the folder exists.
- Hard clustering from fuzzy-c-means values is done based on the maximum membership value.
- The dataset used is the one provided: “dd”. It is present in the inputs directory.
- I have added a very small value (eps on matlab) which would be chosen as the maximum in the modified LBP feature calculation to avoid division by zero scenario.
- 8- nearest neighbours for each pixel are considered to calculate its LBP.
- The LBP 8-bit representation of a pixel is created starting from its top-left (most significant), circling all the way to its left(least significant) i.e. clockwise direction.
- For corner patches, padding with 0 is done to give them neighbours.

Input:

- Run the function q2() by passing the path to the image dataset “dd” and the number of groups k.

Output:

- The program will write the images in folders corresponding to which cluster they belong to. E.g. “Cluster1”, “Cluster2” etc. This can be found in the outputs directory.

**3.**

Assumptions and Caveats:

- All images will be grayscale and resized to 256x256.
- The patch size used is 32x32.
- The dataset used is the one provided: “dd”. It is present in the inputs directory.
- The number of bins in HoG is 15 i.e. 15 key angles.
- K-means was used to cluster the bag of words to k representative features.
- Euclidean distance was used to calculate the distance between each patch-level feature of the target image and the k representative features. The

patch was assigned the ith representative feature's label if it had the minimum euclidean distance from it.
- Signed angles were used to find HoG so that the difference from dark to light and light to dark can be gauged better.

Note:
- Since k-means starts with random centroids, it is very probable that 2 iterations do not give the same k representative features. In this case, the final feature string for the target image may end up being different at each iteration.
- I have outputted the k centroids in the order of 1,2,3,...,k so that interpreting the feature vector of the final image becomes easier.

Input:
- Run the function q3() by passing the path to the image dataset, path to the target image whose feature vector has to be calculated using BoW and k- the number of representative words desired.

Output:
- The k centroids in the order of 1,2,3,...,k so that interpreting the feature vector of the final image becomes easier.
- The feature vector of the target image as per the bag of k words.

**4.**

Assumptions:
- All images will be grayscale and resized to 256x256.
- 8 nearest neighbours for each detected corner are considered i.e. the patch sizes are 3x3 centred around each corner pixel respectively.
- The representative of each patch is the median of LBP value for each pixel in that patch. Thus, for 10 corners per image, each image has a feature vector of length 10.
- For corner patches, padding with 0 is done to give them neighbours.
- Root mean square error was used to calculate the distance between the image-level feature of the target image and the rest of the dataset's features. The top k ones with minimum error were called the k nearest neighbours of the image.

- The LBP 8-bit representation of a pixel is created starting from its top-left (most significant), circling all the way to its left(least significant) i.e. clockwise direction.

Input:

- Run the function q4() by passing the path to the image dataset,  path to the target image whose nearest neighbours have to be found and k- the number of nearest neighbours.

Output:

- The names of the k nearest neighbour images from the dataset inputted and the images are also written to the “../../outputs” file.

Note:

- Since I have just taken 10 significant corner points into consideration, it may be possible that those points encompass the background too. In such a case the image for a different animal might show up if the background is similar or if their fur colours are similar (dd dataset).