IMAGE CLASSIFICATION
WITH BOW AND LDA FOR TOPIC DETECTION

SERGIO LATROFA - 640584

EXTRACT **VISUAL DESCRIPTORS**

1. *Extract* **maximally stable extremal regions MSER.**
2. *Apply Scale Invariant Feature Transform (SIFT) to regions, obtaining descriptors relative to* ***local features.***
3. *Concatenate all the* ***D****, non-null, computed descriptors obtaining a tensor of shape D x 128.*

```python
def imageToDescriptors(image, save_image=False):
  descriptors = []
  gray = cv.cvtColor(image, cv.COLOR_BGR2GRAY)

  # Use MSER to find ROIs within image
  mser = cv.MSER_create()
  regions, boxes = mser.detectRegions(image)

  # Create SIFT computing object. .
  sift = cv.xfeatures2d.SIFT_create()

  to_save_boxes = []
  to_save_regions = []
  to_save_kp= []

  for i in range(len(boxes)):
    box = boxes[i]
    x, y, w, h = box
    roi = gray[y:y+h,x:x+w]

    #Compute SIFT descriptors for MSER regions
    kp, ds = sift.detectAndCompute(roi, None)

    if np.any(ds):
      descriptors += list(ds)
      if save_image:
        to_save_boxes.append(box)
        to_save_regions.append(regions[i])
        to_save_kp.append(kp)

  return (np.array(descriptors, dtype='d'), to_save_boxes, to_save_regions, to_save_kp)
```


1 J Matas, O Chum, M Urban, T Pajdla (2004). **"Robust wide-baseline stereo from maximally stable extremal regions".** *Image and Vision Computing*. ISSN 0262-8856: 761-767,.

2 David G. Lowe (2004). **"Distinctive Image Features from Scale-Invariant Keypoints".** Computer Science Department University of British Columbia Vancouver, B.C., Canada

CODEBOOK CREATION: **CLUSTERING**

Each image contains a variable number of descriptors (like a document can contain variable number of words), anyway descriptors just consist in fixed dimension array (128 elements for SIFT descriptors), so they can be compared one to each other, using for example, **distance metrics.**

Idea here is to use a clustering algorithm to create, **in unsupervised way**, classes of visual descriptors.

In such a scenario, **k-means** algorithm allow us to maintain a minimal control, imposing the number of clusters. Such number will correspond to the cardinality of the theta vectors , and so will impact on the training (time and quality) of the LDA.

The resulting spit of the data will correspond to our k-visual-term "**dictionary**", mapping all the descriptors found until now and also allow us to assign new descriptors to one of the k class, predicting the closest cluster.

K-Means as a minimization problem

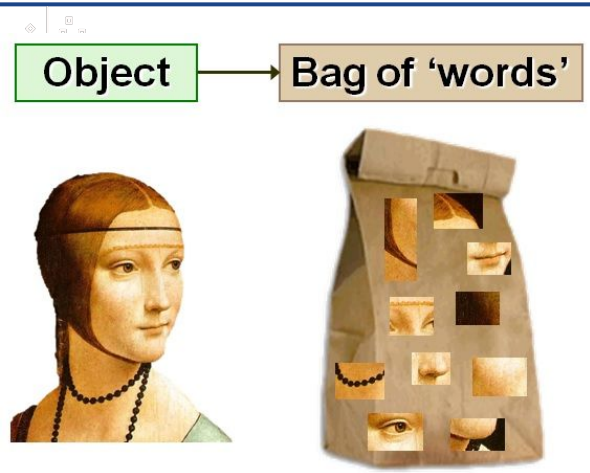
Point-centroid distance is the *cost function*. *Euclidean* distance has been chosen as cost function for simplicity. Anyway, attempts with other metrics, like the *cosine similarity*, would surely be reasonable.

$$\sum_{j=1}^{k} \sum_{i \in C_j} \|x_i - \mu_j\|^2$$

DESCRIPTORS TO **BAG OF WORDS**

Once our dictionary, mapping descriptors in the R^128 to the K=500 possible words, we are ready to compute a new representation for our images, just counting the occurrences of each cluster instance, obtaining histograms, or probabilities if we divide by total number words of each document.

```python
def imageToBOW(image):
  BOW = np.zeros(CLUSTER_NUMBER)
  descriptors = imageToDescriptors(image)[0]

  predicted_labels = clustering.predict(X = descriptors)
  for p in predicted_labels:
    BOW[p] += 1

  return BOW
```



G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray (2004). **"Visual Categorization with Bags of Keypoints".** Workshop on Statistical Learning in Computer Vision. ECCV 2004

LDA - INTUITIVELY

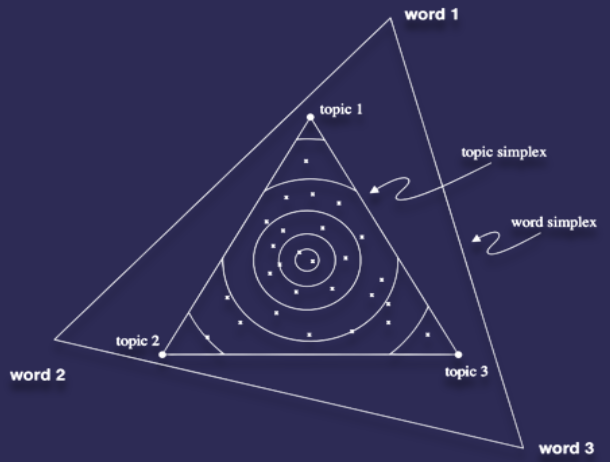
Given a number of **topic** and eventually some knowledge about their prior distribution (expressed by the *alpha* parameter), latent Dirichlet allocation allow us to **map words space to a latent topic space**, basing on the probability distributions observed in the training data.

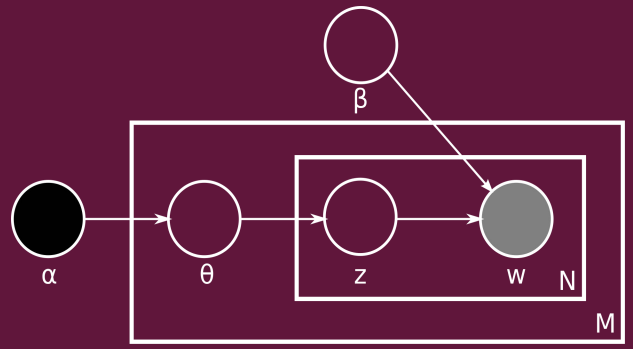
sklearn implementation has been used, executing 10 iterations.

Alpha parameter was not set, assuming a *uniform prior distribution of the topics* towards the words (in our case the visual descriptors clusters).

Number of topic was set **to 8**, according to the number of subset of the dataset.

$$P(\theta \mid \alpha) = \frac{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1}$$





COMMENTING THE **RESULTS**

LIMITATIONS AND POTENTIALITIES OF THE APPROACH

Visual patches of MSER maybe are too superficial

The method proposed here to compute visual descriptors is certainly fast, but meaningful information risk to be lost … SIFT should be computed at least on the whole image!

Not explainable results

Identified object, regions are associated to abstract topics, assign a precise meaning to them only observing images has not been possible for our cases

Global results *seem* **more interesting than patches**

Observing the main topic assigned to each class, we can see that similar images (the ones containing animals, or the ones containing vehicles) are tendentially assigned to the same topic, but such an observation should be better quantified.

OTHER **TEST IMAGES**

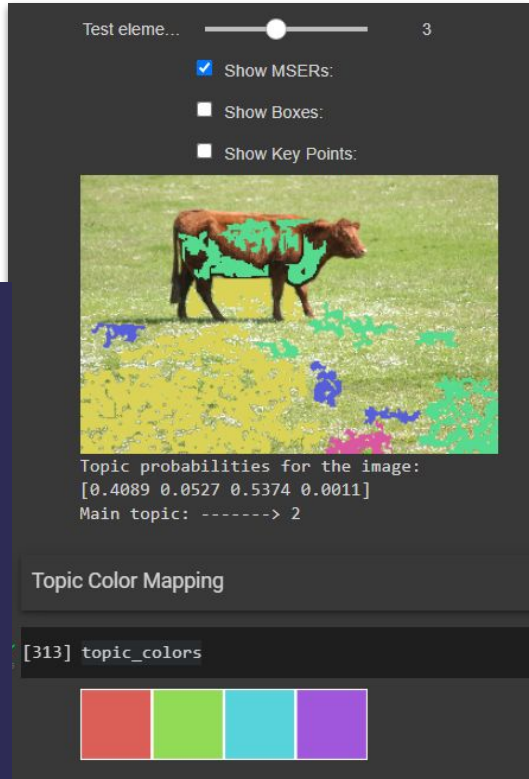
ISPR - MIDTERM 2 - ASSIGNMENT 2

CHECK THE **CODE**

INTERACTIVE WIDGETS

Thanks to the usage of interactive python Widgets, images can be switched and regions and topics can be observed and other results seen. Different attempt changing hyperparameters can also be done, just changing values of the constant and re-executing the code. An hyperlink in in the **Colab Icon**.

```
def draw_descriptors(image, plot_boxes=True, plot_regions=True, plot_keypoints=False):

  descriptors, boxes, regions, kp = imageToDescriptors(image, save_image=True)
  signed_image = image.copy()
  for i in range(len(boxes)):
    word = clustering.predict(X = [descriptors[i]])[0]
    topic = np.argmax(lda.components_[:,word])

    topic_color = color_topic_mapping[topic]

    x, y, w, h = boxes[i]

    if plot_boxes:
      cv.rectangle(signed_image, (x, y), (x+w, y+h), topic_color, 2)

    if plot_regions:
      for p in regions[i]:
        cv.circle(signed_image, (int(p[0]),int(p[1])), radius=0, color=topic_color)

    if plot_keypoints:
      for p in kp[i]:
        resultimage = cv.circle(signed_image, (int(p.pt[0]+x), int(p.pt[1]+y)), radius=8,
color=topic_color, thickness=1)
        resultimage = cv.circle(signed_image, (int(p.pt[0]+x), int(p.pt[1]+y)), radius=4,
color=topic_color, thickness=1)

  cv2_imshow(signed_image)
```



FINAL CONSIDERATION

ABOUT LDA FOR IMAGES AS BOW

Understandable and simple to implement approach, involving instrument familiar concepts coming from affine fields, like **natural language processing** and **information retrieval**.

Topic and visual word remain **“gray” concepts**, not fully explainable in terms of object present in the images. They actually seems to be something useful especially in a data understanding phase, more than for **actual image classification tasks**, especially without additional information.

FURTHER “WEIRD” IMPROVEMENTS

A more deep model selection can certainly be carried on for the proposed model, experimenting with a pletora of **hyperparameters** like **word and topic number** or **training iterations**. Different sets of the images should be used to build the codebook can be analyzed (something K-Fold like), verifying **impact of the sampling**.

Maybe some interesting application with a similar approach can be thought for **generative tasks.**

Feature engineering and problem reformulation

Opportunity to understand importance of feature extraction processes, involving several studied algorithms.

Bayesian Learning and mathematics

Direct application of several mathematical concepts, (probabilities, distributions, conditioning, histograms etc…), which sometimes can appear as “too theoretical stuff”.