Programming Assignment 1:
Data Preparation and Understanding
(10 points)

September 8, 2024

1. In this semester, we will be using the "Stanford Dogs" dataset (http://vision.stanford.edu/aditya86/ImageNetDogs/) for all our 4 programming assignments. There are a total of 120 classes (dog breeds). The number of images for each class ranges from 148 to 252.

   Each student will

   (a) be assigned 4 classes to work on the 4 assignments.

   (b) download **Images** (and also **Annotations** - bounding boxes) datasets for the 4 classes to work on.

   (c) create a Github account to share (as collaborator) their solution (Readme, Codes, Processed Dataset for Code to run correctly) with the grader.

2. Use **XML processing modules** (https://docs.python.org/3/library/xml.html) to obtain bounding box information from **Annotations** datasets and **scikit-Image** (Reference: https://scikit-image.org/) to perform image processing and feature extraction.

   (a) **Cropping and Resize Images in Your 4-class Images Dataset**: Use the bounding box information in the **Annotations** dataset relevant to your 4-class Images Dataset to crop the images in your dataset and then resize each image to a 128×128 pixel image. (Hint: https://www.kaggle.com/code/espriella/stanford-dogs-transfer-crop-stack/notebook

   Code Snippet 1:

```
def get_bounding_boxes(annot):
    xml = annot
    tree = ET.parse(xml)
    root = tree.getroot()
    objects = root.findall('object')
    bbox = []
    for o in objects:
        bndbox = o.find('bndbox')
        xmin = int(bndbox.find('xmin').text)
        ymin = int(bndbox.find('ymin').text)
        xmax = int(bndbox.find('xmax').text)
        ymax = int(bndbox.find('ymax').text)
        bbox.append((xmin,ymin,xmax,ymax))
    return bbox
```

   Code Snippet 2:

```
for i in range(len(dog_images)):
    bbox = get_bounding_boxes(annotations[i])
    dog = get_image(annotations[i])
    im = Image.open(dog)
    for j in range(len(bbox)):
        im2 = im.crop(bbox[j])
        im2 = im2.resize((331,331), Image.ANTIALIAS)
        new_path = dog.replace('../input/stanford-dogs-dataset/images/Images/','./Cropped/')
        new_path = new_path.replace('.jpg','-' + str(j) + '.jpg')
        im2=im2.convert('RGB')
        head, tail = os.path.split(new_path)
        Path(head).mkdir(parents=True, exist_ok=True)
        im2.save(new_path)
```

)
(0.5 point)

(b) **Feature Extraction: Edge histogram AND Similarity Measurements**

i. Choose 1 image from each class.

ii. Convert the color images to grayscale images (see https://scikit-image.org/docs/stable/auto_examples/color_exposure/plot_rgb_to_gray.html)

iii. For each image I, use the following

```
import numpy as np
from skimage import filters

def angle(dx, dy):
    """Calculate the angles between horizontal and vertical operators."""
    return np.mod(np.arctan2(dy, dx), np.pi)

angle_sobel = angle(filters.sobel_h(I),
                    filters.sobel_v(I))
```

to obtain an “angle” for each pixel in the images (Intuitively, one can think of the “angle” as the direction of edge gradient at the pixel).

iv. Use **skimage.exposure.histogram** (see https://scikit-image.org/docs/stable/api/skimage.exposure.html#skimage.exposure.histogram) to obtain a histogram with 36 bins.

v. Plot the images with their corresponding edge histogram values (add x-axis label “Bins” and y-axis label “Pixel Count”). (2 points)

vi. Pick 2 edge histograms from the 4 you have constructed (These are the vector representations of the images)

- Perform histogram comparison between the 2 edge histograms using the following metrics/measures. (see https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.distance_metrics.html#sklearn.metrics.pairwise.distance_metrics)
  - Euclidean distance
  - Manhattan distance
  - Cosine distance

  (1.5 points)

(c) **Histogram of Oriented Gradient (HOG) feature descriptor (see https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients)**

i. Pick 1 image and compute its HOG descriptors. Visualise the image and the HOG descriptors for the image (see https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_hog.html#sphx-glr-auto-examples-features-detection-plot-hog-py) (1 point)

(d) **Dimensionality reduction (using Principal Component Analysis, PCA)** (see https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html for PCA. https://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_iris.html for code example. We will use scikit learn more extensively in the next assignment)

i. Use images from all four classes.

ii. Convert all the images from the four classes to edge histograms.(0.5 points)

iii. Perform Principal Component Analysis (PCA) dimensionality reduction on the set of histograms to reduce from 36 to 2 dimensions. (Note: You should not use the class labels) (1 point)

iv. Plot the 2D points using four different colors for data from the four classes (see Figure 1). How many classes are visually separable (i.e., non-overlapping) ? (1 point)

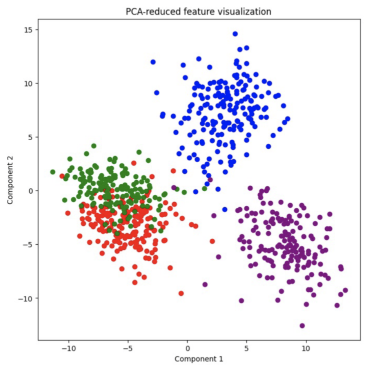

Figure 1:

3. Next, we perform some text processing steps on a tweet (i.e., text) dataset. The dataset file is in json format and each dataset consists of

   - Training Set: 3,000 records
   - Test Set: 1,500 records
   - Validation Set: 400 records

   This is a multi-class dataset with eleven classes (‘anger’, ‘anticipation’, ‘disgust’, ‘fear’, ‘joy’, ‘love’, ‘optimism’, ‘pessimism’, ‘sadness’, ‘surprise’, ‘trust’).

   You will use this dataset again in Assignment 3. **For this assignment, you will just be using the training set**.

   Each student is

   (a) assigned a unique dataset, and

   (b) you need to download your dataset from `https://drive.google.com/drive/folders/1v6ckoWUqFzj23` `we7TEe` (Note: you need to use your Rowan email to login to the google drive)

4. You will use the simple **countvectorizer** and **tfidfvectorizer** in `https://scikit-learn.org/stable/api/sklearn.feature_extraction.html#module-sklearn.feature_extraction.text` to extract (1) token (feature) counts, and (2) TF-IDF feature (counts), respectively. (You will use the default parameters.) What are the dimensionality of the two vector representations?

5. Using the two sets of processed text data in Item 4,

   - Pick four classes which you think will be separable. State the four classes.

- Perform **dimensionality reduction** similar to 2(d) with dimensionality reduced to 2.
- Plot the 2D points using four different colors for data from the four classes (see Figure 1) for both token count features and tf-idf features in two separate plots. (2 points)
- How many classes are visually separable (i.e., non-overlapping) for both plots? (0.5 point)