**Image Similarity**

**Objectives**

The objectives of this assignment are:

1. to group similar images out of a set of ten given images, and
2. To formulate a scoring mechanism for images, such that unique images end up getting a higher score.

**Methodology**

This solution builds upon a simplified implementation of the following paper:

<http://wangjiangb.github.io/pdfs/deep_ranking.pdf>

Concretely, the paper proposes the following steps for training a model for **ranking** and **grouping** images based upon similarity

**Step 1: Learning**

The image dataset was obtained from the following link:

<https://sites.google.com/site/imagesimilaritydata/>

This is a subset of the same dataset that was used in the original paper.

Each example in the dataset is a **triplet.** A triplet contains a **query,** a **positive** and a **negative.**

**Distance** between two given images is denoted by

D ( f(image1), f(image2) ).

**f** here is an embedding that will be learned by the neural network. Every image will then be passed through this embedding before distance is calculated between two images.

A **successful prediction** would conclude that

D ( f(query), f(positive) ) < D ( f(query), f(negative) ) (1),

Implying that the **positive** is more similar to the **query** than the **negative.**

For accomplishing this task, the following loss function has been proposed:

**Loss (triplet) = max ( 0, D ( query, positive ) - D ( query, negative ) + margin )**

The use of this loss function ensures that once it is optimized, the inequality (1) is satisfied.

A deep neural network is trained on about 5000 image triplets.

**Step 2: Ranking**

Each of the images is assigned a **rank** based on its **collective similarity** with the rest of the images. The higher the **score,** the higher the image is ranked.

Here, the score of an image (**key)**  is defined in the following way:

Score ( key ) = sum ( D ( f(key), f(image) ) ),

where **image** is a variable that denotes, one at a time, the rest of the images.

**Step 3: Clustering**

The number **k** is a hyperparameter that has to be chosen for this step.

For the purpose of clustering, we need to choose a set of **seeds** first. This is done using the following approach:

1. A random image is chosen as the first seed
2. The image which is the farthest away from the first seed is chosen as the second seed
3. The image which is the farthest away from the first two seeds combined is chosen as the third seed, and so on….