Essential Mathematics for Data Scientists

Assessment 5: VisualRank using image similarity

Introduction

This assignment is based on the well-known PageRank algorithm developed by Google to help in ranking webpages. Although much has changed with Google's techniques and it is not known exactly how much of a role the original PageRank algorithm plays in current webpage rankings, it remains an important tool in Google's arsenal.

A different application of PageRank came about when some of Google's engineers suggested applying it to images. They came up with the 'VisualRank' algorithm. We shall attempt something similar in this assignment, albeit with much more simplistic tools!

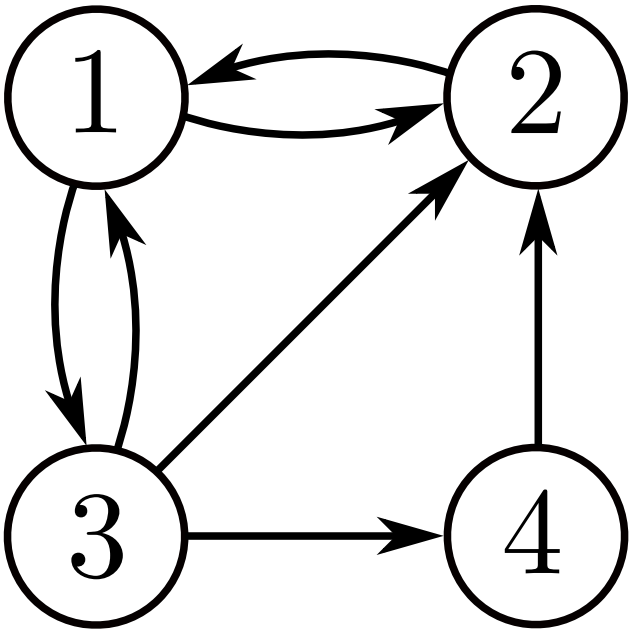
A pedagogical description of the PageRank algorithm is available at:
https://www.amsi.org.au/teacher_modules/pdfs/Maths_delivers/Pagerank5.pdf

Both algorithms are described below.

PageRank

The PageRank algorithm treats the internet as a directed graph, with webpages represented by vertices and hyperlinks to pages represented by directed edges. PageRank then simulates a hypothetical user browsing the internet by clicking on links at random. A rank is then assigned to each page based on the likelihood of finding the user there.

For example, consider the following mini-internet, with four webpages:



The location of the hypothetical user can be expressed using a vector of probabilities. If the user were to start on page 2 above, the initial vector of probabilities would be:

$$\mathbf{v}_0 = [0 \quad 1 \quad 0 \quad 0]$$

Then, after clicking on the only link on page 2, the user would end up on page 1 with certainty:

$$\mathbf{v}_1 = [1 \quad 0 \quad 0 \quad 0]$$

At this point, the user can either follow a link back to page 2 or continue to page 3. The user picks one of these two links at random and ends up on either of those pages with probability 1/2:

$$\mathbf{v}_2 = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}$$

This process can be described using the adjacency matrix for the directed graph:

$$\mathrm{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

The probability of following a link *from* a webpage is distributed equally among all links on that webpage. Since each row of the adjacency matrix corresponds to links *from* a webpage, we can form the required matrix of link-following likelihoods by normalising each row so that they sum to 1:

$$\mathrm{H} = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 1 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

This is called the hyperlink matrix, which describes not only *where* links are but also *how likely* the user is to click on them. Using this matrix, we can simulate a click by right-multiplying the state vector:

$$\mathbf{v}_{n+1} = \mathbf{v}_n \mathrm{H}$$

For example, we can simulate the example clicks above in this way:

$$\mathbf{v}_1 = \mathbf{v}_0 \mathrm{H} = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 1 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{v}_2 = \mathbf{v}_1 \mathrm{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 1 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}$$

Finally, to complete the description of the PageRank algorithm, we also need to consider the case of webpages without *any* links, as such pages would cause our simulated internet user to become stuck. Google addressed this issue by making the user occasionally get bored of the page they're on and go to a random webpage on the internet. To do this they introduced a so-called 'damping factor', d, which modified the hyperlink matrix as follows:

$$\widetilde{\mathrm{H}} = d\mathrm{H} + (1 - d)^* \mathrm{J}$$

where H is the original hyperlink matrix and J is a hyperlink matrix that corresponds to a jump to a random webpage.

If the total number of webpages is N then we have:

$$\mathrm{J} = \begin{bmatrix} \frac{1}{N} & \cdots & \frac{1}{N} \\ \vdots & \ddots & \vdots \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{bmatrix}$$

It has become standard to use $d = 0.85$, in which case there is a $d = 0.85 = 85\%$ chance the user will follow random links as usual and a $(1 - d) = 0.15 = 15\%$ chance the user will get bored and jump to a random page on the internet.

Now that we have accounted for dead-ends with the modified hyperlink matrix $\tilde{\mathrm{H}}$ above, we can implement the PageRank algorithm by iterating:

$$\mathbf{v}_{n+1} = \mathbf{v}_n \tilde{\mathrm{H}}$$

As stated earlier, the PageRank algorithm is concerned with where the user ends up after enough clicks. The vector of probabilities after an infinite number of clicks is called the PageRank vector:

$$\mathbf{r} = \lim_{n\to\infty} \mathbf{v}_n = \lim_{n\to\infty} \mathbf{v}_0 \tilde{\mathrm{H}}^n = \mathbf{v}_0 \tilde{\mathrm{H}}\tilde{\mathrm{H}}\tilde{\mathrm{H}} \cdots$$

This vector should be the same no matter where the user starts initially ($\mathbf{v}_0$).

For the earlier example with four webpages, we find the following PageRank vector:

$$\mathbf{r} \approx [38\% \quad 33\% \quad 20\% \quad 9\%]$$

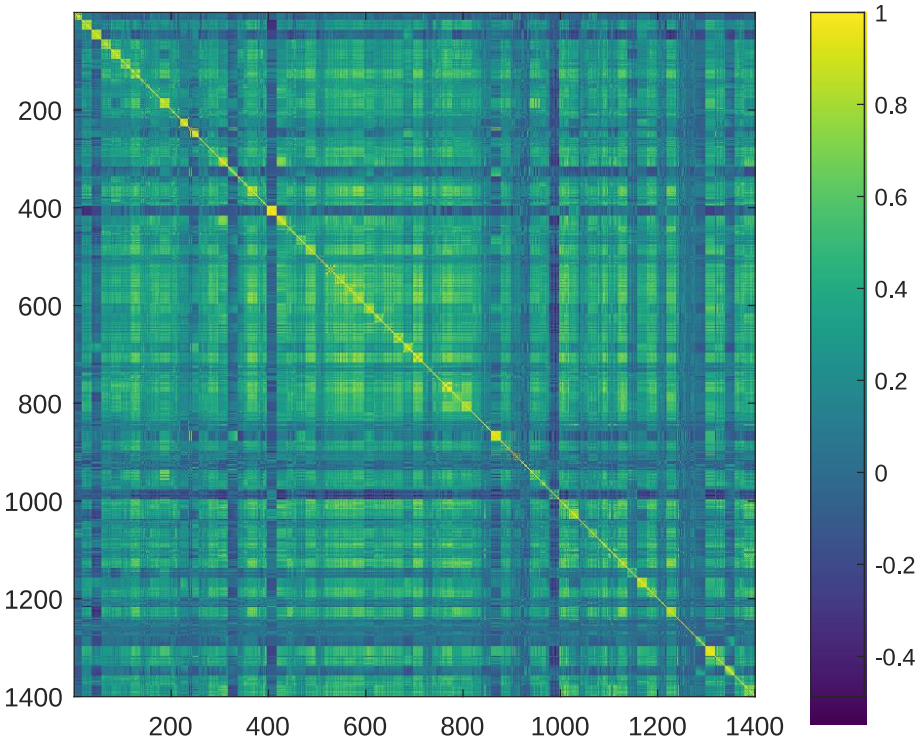
It can be seen in this case that the user is most likely to be found on page 1, so it is given the highest PageRank, followed by pages 2, 3 and 4.

VisualRank

By analogy with PageRank, VisualRank makes use of 'visual hyperlinks' between images to determine the most 'important image' in a group (i.e. the image that best represents the group). We can determine the presence and strength of these hyperlinks by using a measure of image similarity like that provided by the SVD-based `compareImages` function introduced in Week 3. In principle, however, any other measure of image similarity could be used and there are certainly more sophisticated alternatives that are used in practice.

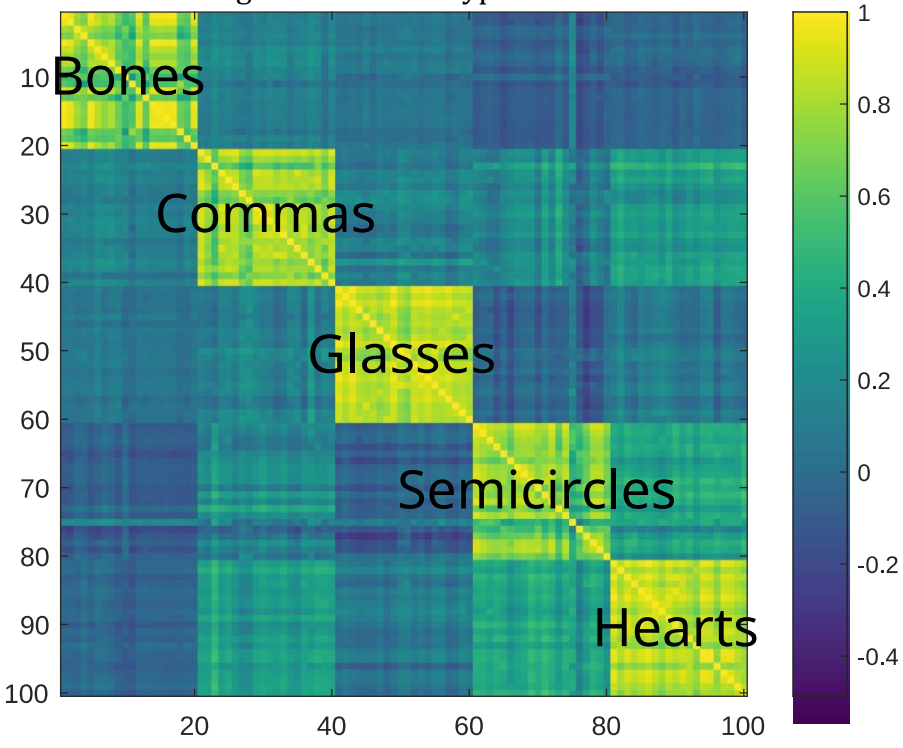
The image set that we will use is the MPEG-7 testing dataset. The entire dataset is provided to you in the archive 'MPEG7.zi'. In Week 3 you compared 10 images from this dataset by tabulating their output from the `compareImages` function in a 10x10 similarity matrix. This week, you will require the similarity matrix for all 1400 images in the dataset. As it is a time-consuming process to fill the entire 1400×1400 similarity matrix, this matrix has been provided for you in the file 'sim.mat'. This file can be loaded into MATLAB and should contain the required similarity matrix `S` and a cell array of corresponding `filenames`. For example, since `filenames{200}` returns `'bell-9.png'` and `filenames{500}` returns `'cup-9.png'`, the similarity between these two images is given by both `S(200,500)` and `S(500,200)`.

The full 1400x1400 similarity matrix `S` looks as follows:



The checkerboard pattern here is due to the dataset of 1400 images being made up of 70 groups containing 20 pictures each (e.g. there are 20 pictures of bats, 20 pictures of apples etc.).

If we zoom in on the top-left corner, we can clearly see the bright 20x20 blocks that indicate that images of the same type are most similar to one another:



To apply VisualRank, we need to form the adjacency matrix A for the graph that connects the images by visual hyperlinks. After this VisualRank proceeds as PageRank did by forming the corresponding (visual) hyperlink matrix H, tweaking it to get $\tilde{H}$ and iterating.

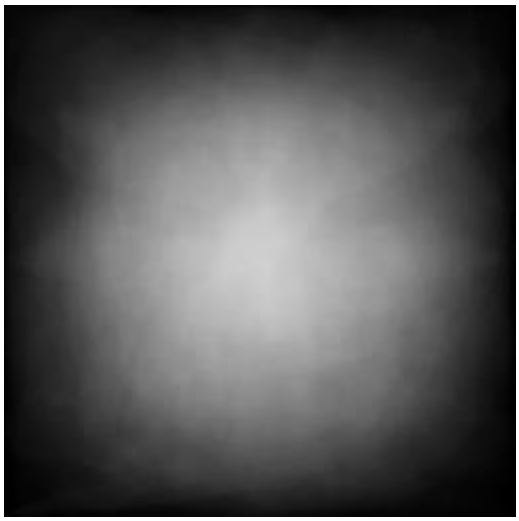
Numerically, the similarities in the above matrix S range in value from -1 to $+1$. Positive values describe *correlations* between images, while negative values describe *anticorrelations* (e.g. an image and its inverse are perfectly anticorrelated). We thus can form a suitable adjacency matrix A by removing elements from S to leave only the connections between images that have some positive correlation.

Lastly, to complete the adjacency matrix A, it is also useful to remove all loop edges in the similarity graph as we are interested in ranking images based on how similar they are to each other, rather than how similar they are to themselves.

Tasks

1. Using the theory above, rank all 1400 MPEG7 shape images using VisualRank.
    a. Load the 'sim.mat' data file into MATLAB. This file contains the similarity matrix `S` for the entire image dataset, as well as a cell array of corresponding `filenames`. **(0.5 marks)**
    b. From `S`, we can form the adjacency matrix `A`. Start by setting `A=S`, then remove elements from `A` to leave visual hyperlinks between only those images that are positively correlated. **(1 mark)**
    c. Finally, remove the elements from `A` that correspond to loop edges in the visual similarity graph. **(1 mark)**
    d. Use `A` to create the hyperlink matrix `H`. **(0.5 marks)**
    e. Form the random jump matrix `J`. **(0.5 marks)**
    f. Given a damping factor of $d = 0.85$, create the modified visual hyperlink matrix `Htilde`. **(0.5 marks)**
    g. Find the VisualRank vector `r` using an initial vector. **(3 marks)**
    h. Given this VisualRank vector, which image is ranked highest? Display this image. **(1 mark)**

    VisualRank ranks images in a group by how representative they are of the group as a whole. This is a difficult task when the group of images is large and varied, as it is in this case. The only common feature among all 1400 images is that the images tend to depict some white object toward the centre of the frame, as can be seen by the average of all the images:



Thus, if you have answered everything above correctly, you should have found a #1 image that looks very vaguely like this.

In practice, VisualRank is much more usefully applied to smaller groups of similar or related images, as is the focus of the following tasks.

2. Rank the 20 heart-shaped images (indices 81 through 100) using VisualRank.
    a. Make a smaller 20x20 adjacency matrix by indexing the necessary rows and columns of the full adjacency matrix from above. **(1 mark)**
    b. Form the corresponding visual hyperlink matrix and find the VisualRank vector for all 20 heart images. **(2 marks)**
    c. Given this ranking, which heart-shaped image is most representative of the group of heart-shaped images? Display this image. **(1 mark)**
    d. Similarly, which heart-shaped image is the least heart-shaped (according to VisualRank)? Also display this image. **(1 mark)**

3. Pretend you are a search engine that has been queried for an image search of the following pentagon-looking shape that is present in the dataset as 'device6-18.png' (at index 650):

MATLAB has a `nearest` function that finds the nearest nodes to a particular node on a graph. We will use this function to search for the images similar (near) to the image above and then refine these search results using VisualRank.

a. We first need to create a graph from the similarity adjacency matrix `A` created in Task 1. However, we cannot use this matrix directly, as it contains edge weights that are *larger* when images are more similar, corresponding to a *further* distance. We would instead like images that are similar to each other to be *nearer* in the graph. This can be achieved simply forming a new adjacency matrix for which the elements are the reciprocal of those in `A`. Do this, and, using the `digraph` function, form the graph `G` corresponding to this new adjacency matrix. **(3 marks)**

b. Using MATLAB's `nearest` function, find the 10 images nearest to 'device6-18.png' in the graph `G`. **(2 marks)**

c. Finally, rank these 10 nearest images using VisualRank and display them on a figure in their ranked order. This is the final search result. **(2 marks)**