IMAGE RETRIEVAL USING KERNEL METHODS

FOONG JOAN TACK

1. Thesis Abstract

Image retrieval system takes in some texts, images or both, and it searches through its database to find the images that are most relevant to the input. It is content-based if the search is based on the 'visual contents' of the input images, which might be their colours, textures, shapes, or some local features. We can describe the 'visual contents' in the form of vectors, and we call these descriptors.

To search for an image, we must be able to represent it effectively. We are going to use vector space model here. For every image in our database, and for some interesting points in the image, we can describe its local visual feature by a particular local descriptor. Remembering that our descriptors are in the form of vectors, all these instances of local descriptors lives in a large vector space, which we call a feature space. Some of the descriptors might be similar to each other in the sense that they are describing a similar visual feature from different images. So we can use clustering to divide our feature space into discrete regions, vectors in which are all describing similar visual feature. So suppose that our feature space is divided into n regions $[R_1, R_2, ..., R_n]$, and a particular image is described by m descriptors $[D_1, D_2, ..., D_m]$, we can count the number of descriptors that are belonging to R_i, and call it O_i. For instance, if D_1, D_5, D_7 are belonging to R_1, then $O_1 = 3$. We are going to represent our image using $[O_1, O_2, ..., O_n]$, which is inspired by the 'bag of words' representation used successfully in the field of text mining.

After we have represented the images as vectors, we would like to cluster similar images together, that is to serve as labels for using Support Vector Machine(SVM) in the next step. To this end, we are going to use Latent Semantic Indexing(LSI). We do not know how the low level visual features, such as colors and shapes, can map to the high level semantic meanings, such as happiness and sadness. But we suppose that there are some hidden structures, which might be some combination of the low level features, that can reflect the meaning. LSI can reveal the hidden structure and reduce the noise in our raw data.

Suppose that our database contains images $I_1, I_2, ..., I_k$, and the image I_j is represented by $[O_1^j, O_2^j, ..., O_n^j]$ then we can construct the matrix below:

$$O = \begin{pmatrix} O_1^1 & O_1^2 & \cdots & O_1^k \\ O_2^1 & O_2^2 & \cdots & O_2^k \\ \vdots & \vdots & \cdots & \vdots \\ O_n^1 & O_n^2 & \cdots & O_n^k \end{pmatrix}$$

We can then perform Singular Value Decomposition on A, that is to write:

$$O = U\Sigma V^*$$

where U is an $n \times n$ unitary matrix, Σ is a $n \times k$ diagonal matrix, and V is a $k \times k$ unitary matrix. The core of this method is that we truncate the matrix Σ down to a rank r matrix Σ_r, and U, V to U_r, V_r by keeping only their first r columns. By choosing the appropriate rank r, we can reduce the noise and the variation and hence reveal the structure of our data. Part of the aim of this project is to produce empirical data and theoretical foundation for the statement above.

We can then represent the visual features by the column vectors of $U_r\Sigma_r$, and the images by the row vectors of $\Sigma_r V_r^*$. All of them are living in r-dimensional vector space. Given any query image, q, represented as $[O_1, O_2, ..., O_n]$, we can transform it to $\hat{q} = q^* U_r \Sigma_r^- 1$, which is living in the same r-dimensional vector space, and appropriately weighted. From now on, we can perform the query 3 steps:

The first step is to represent the image. This involves calculating the cosine similarity between the query image and all the other images in our database; then we return the images that are most similar to the query image.

The second step is to cluster the images in our database using k-mean clustering, and use the clusters as labels to the images. We can now train the Support Vector Machine using the images and their labels, which later is used to classify the query images. We then return the images that share the same label as the query image.

The third step is to use a graph database. The idea is to represent every images and visual features as a node in a graph. An image node is connected to a features node if it has that particular feature. And the edge of that connect these 2 nodes is labeled by the appropriate entry in the matrix O_r. We can then perform local graph transversal to find the images that shared most of the nodes, with similar labels, to the query image.

To summarize, our image retrieving process consists of 3 steps, namely: representing the images, clustering, and the actual retrieving. The novelty of our

approach are 1. to reduce the semantic gap by using LSI and SVM, and to reveal the hidden structures of the images that can map to their meanings; and 2. to perform the actual retrieval using graph database which scales better than the traditional relational database, especially when the data model has a lot of connection, which is our case.