Look-a-like: A Fast Content-based Image Retrieval Approach using a Hierarchically Nested Dynamically Evolving Image Clouds and Recursive Local Data Density

Plamen Angelov[a], Pouria Sadeghi-Tehran[1]

[a] *School of Computing and Communications, Data Science Group, Lancaster University, United Kingdom, LA1 4WA*

Abstract

The need to find related images from big data streams is shared by many professionals, such as architects, engineers, designers, journalist, and ordinary people. Users need to quickly find the relevant images from data streams generated from a variety of domains. The challenges in image retrieval are widely recognised and the research aiming to address them led to the area of CBIR becoming a 'hot' area. In this paper, we propose a novel computationally efficient approach which provides a high visual quality result based on the use of local recursive density estimation (RDE) between a given query image of interest and data clouds/clusters which have hierarchical dynamically nested evolving structure. The proposed approach makes use of a combination of multiple features. The results on a data set of 65,000 images organised in two layers of an hierarchy demonstrate its computational efficiency. Moreover, the proposed *Look-a-like* approach is self-evolving and updating adding new images by crawling and from the queries made.

Keywords: recursive density estimation (RDE), dynamically evolving hierarchy of data clouds, content-based image retrieval (CBIR)

1. Introduction

The last decade has witnessed an enormous growth in the amount of digital images on Internet (it was recently estimated that the number of images on the Web is over 100 Billion [1], a figure which some observers consider to

URL: p.angelov@lancaster.ac.uk (Plamen Angelov), pouria.sadeghi-tehran@rothamsted.ac.uk (Pouria Sadeghi-Tehran)

be an underestimate). Everyday millions of new images are being generated creating an enormous multi-dimensional data stream. They play an important role in the fields of entertainment, education, advertising, etc. For instance, photographers or designers often request images with a particular colour or texture; therefore, developing a system which automatically derives requested and similar images is essential. Using the World-wide Web, users are able to access these images from anywhere in the World which creates a huge stimulus to quickly find images that user needs.

Since images are in a digital form this opens new prospects to organise them in a convenient to manipulate form. However, the information that they contain is unstructured and there is no universal established and stable approach to convert this information into an easy to manipulate form. For example, text can be organised alphabetically, music using notes, etc. Images, however, are significantly more unstructured by nature. It is a challenge to organise a huge and dynamically growing amount of images in a structure that is convenient to search quickly. It is possible to identify and retrieve desired images from a small database; however, the difficulties become more vivid for big and dynamically growing data streams with varied images. Another challenge is to use an effective measure of similarity between the query image and another image or a set of similar images.

The methods for retrieving similar images based on features such as colour, texture, or shape are usually referred to as Content-Based Image Retrieval (CBIR). The early use of CBIR system was introduced by Kato [2] in 1992. This research area has since been widely investigated by many researchers.

Although, CBIR technology has started to be used in the form of commercial products - such as TinEye [3], QBIC [4], Yandex Image Search [5], etc. and research projects like NETRA [6], Photobook [7] - still suffers from the lack of maturity. It is obvious from the (lack of) effectiveness of existing CBIR systems, especially when handling real-time scenario on the Web (handling 10^{11} images). There are still many open research issues to be addressed before taking a full advantage of fast and reliable CBIR systems in practice.

Searching images on the Web is a complex problem. Search engines like Google and Yahoo are still not capable of providing efficient CBIR systems and sometimes they are too computationally expensive to operate on the Web, returned results are often not relevant or based on indexing and tagging instead of visual similarity. In addition, storing huge history of images and processing them in the memory is one of the toughest challenge. As a result, users still need to apply considerable efforts to find images they are looking for.

In order to address some of the issues mentioned above, a new fast method called *Look-a-like* for finding visually similar images in big data streams is proposed in this paper which is using a combination of features of different nature, a dynamically evolving hierarchically nested structure of image clouds and a single formula of recursive density estimation (RDE) [8, 9] applied locally (per image cloud). The proposed approach is computationally and time-wise very efficient due to the combination of the hierarchically nested image clouds structure and the use of the local RDE.

Dynamically evolving character of the problem is addressed by a constant update of the proposed nested hierarchical structure using the recently introduced ELM [10] clustering method at the back-end server of the overall system. This approach is very computationally efficient and robust and provides visually meaningful results due to the combination of features of various nature. The local RDE provides the *exact* information about the similarity between any given query image and *all* images from a given image clouds. The proposed approach *Look-a-Like* is capable of real-time image retrieval from a huge number of images. (For example, 10^{12} images which is approximately the amount of images on Internet can be organised automatically using ELM in 6 layers of hierarchy with approximately 100 clusters/data clouds in each layer and the search will then only require to calculate $6 \times 100 = 600$ times the local RDE which takes less than a second on a PC. The performance of the proposed approach was evaluated on a database containing 65,000 images with over 600 classes. The results demonstrate that *Look-a-like* method introduced here is computationally very efficient and fast. Furthermore, the images returned as a result of the search were visually very similar to the query image and the time required was very low, especially with the hierarchically nested structure. In addition, a GUI for a desk-top application was also developed.

The remainder of the paper is organised as follows. First, in section 2 the related work on CBIR is analysed. The proposed approach, *Look-a-like*, is described in section 3. Section 4 details the experimental results. Finally, Section 5 provides conclusions and outlines the future works.

2. State-of-the-art of CBIR

There have been extensive studies to investigate and address the challenges that the CBIR systems face. In this section, we will briefly analyse the proposed techniques in this area.

Eakins and Graham [11] categorised three types of queries in CBIR systems. The first type includes extracting primitive features such as colour, texture, shape or the spatial location of image elements. The most common query is the query by example; for instance, users are interested to find images that are similar to a certain query image. The second type, concerns retrieval of specific object of given type identified by extracted features, with some degree of logical inference [12]. For instance, users intend to find a picture of a bus. The third type includes retrieval by abstract attributes. It involves a considerable amount of high level reasoning regarding the purpose of the objects including pictures with emotional significance, special event, etc. For example, users may want to find pictures of a cheerful crowd. The second and third types are referred to as 'semantic image retrieval' and the difference between them is called 'semantic gap' [11]. Image retrieval of the first type requires users to submit an example/query image; on the other hand, semantic image retrieval supports query by keywords in case users do not have a query image.

In this paper we focus on the first two types of image retrieval where the search is based on a query image and visual similarity, not semantic one. There are three main steps of the process, namely, feature extraction, organisation of the available images, and evaluating the similarity between images.

Feature extraction is a very important element of any CBIR system. Features can be extracted from the specific region of an image or from the entire image. Since colour spaces are closer to the human perception, they are widely used as features in CBIR systems. For different applications different colour spaces can be used such as colour histograms, moments, covariance matrices, dominant colours, etc. [13]. For instance, if objects in an image have homogeneous colour, extracting average colour is not a good option, specifically for face recognition applications [12, 14]. As opposed to colour, texture is not well defined and many systems do not use it as a feature [14, 15]. However, texture refers to the pattern recognitions that have properties of homogeneity that can not be determined from the presence of intensity or a single colour only [16]. Texture provides important information in image classification and describes the content of images such as clouds, sea, fabric, skin, etc. Therefore, it gained popularity in the area of pattern recognition and image processing. Fourier transform, wavelet transform [17], and Gabor filters [18] are used often for texture analysis. Shape is another important feature used in computer vision; however, due to inaccuracy of segmentation it is difficult to determine and has not been as widely used as colour and texture. The representation of the shape can be divided into two categories known as; a) region-based, and b) boundary-based. In region-based techniques, the entire region is used while in the boundary-based approach only the outer boundary is taken into account [19]. For different applications scale, rotation or translation invariance can be used to represent the shape.

Another very important element of CBIR, especially, when applied to the Internet is the organisation of images. Different clustering algorithms has been used for this purpose such as the mean-shift, k-means, and hierarchical clustering methods [20]. BenHaim et al. in [21] used HSV colour histogram to extract features and cluster images based on the offline iterative mean-shift clustering algorithm. The cluster that corresponds to the largest number of parent images is selected and referred to as the 'significant' cluster. In [22] the BOO-clustering algorithm and GDBSCAN is utilised to extract colour clusters of each image. Once these are determined, the objects are formed by selecting one or a few colour clusters of the image in an interactive manner. K-means clustering approach and indexing structure B+ tree is used in [23] to group relevant images in a CBIR system. For the retrieval process, images from the closest cluster and from other nearby clusters are considered to retrieve similar images even if the query image is mis-clustered; however, an important drawback of this approach is that the number of clusters, K has to be predefined and is not changing afterwards (is fixed); thus, the number of image groups in the dataset should be known in advance. Another disadvantage is the computational complexity of the k-means approach which is iterative, for large number of images it becomes prohibitive.

In [24] a hybrid clustering technique is used based on k-means clustering and Linde-Buzo-Gray (LBG) clustering methods. Initially, this algorithm assumes that one large Gaussian represents all images in the database. This is later iteratively split and re-estimated to obtain a mixture of Gaussians. The authors tested their algorithm on 12,000 images from 100 classes collected from Google Image search; however, the result of only one class has been illustrated and no comparison with other methods has been done. Same disadvantages as for the previous approach can be attributed to this approach plus the unrealistic assumption of Gaussian distribution.

In order to tackle the 'semantic gap' problem, Chen and Wang [25] proposed an unsupervised learning technique based on clustering. In their approach, image clusters are obtained based on the feature similarity of retrieved images to the query image and also on how the retrieved images are similar to each other. The main drawback of this approach is that the clusters are fixed and not evolving; therefore, if add even a single new image to the database the whole procedure, including the clustering has to be repeated 'from scratch'. In addition, this approach has not been and cannot be applied to a large number of images (e.g. Internet) because it is computationally expensive.

Searching through large image collections especially on the Web with over 100 billion images can be a tedious work. Developing a hierarchical organisation can significantly speed up the search which is essential. In [26] authors developed a hierarchical annular histogram (HAH) and tested it on images from prostate cancer. They consider the hierarchy of image to sub-images and not a hierarchy of nested clusters/image clouds as in the proposed paper and applied their technique to a small amount of images from a specific area only. On the other hand, Distasi et al. [27] applied a hierarchical entropy-based representation (HER) to a database containing several shapes represented by their closest contour in curvilinear coordinates to be used in a CBIR system. A tree-based structure of representation of images was proposed by Chow et al. [28] where a root node contains the global features, as opposed to child nodes which contain the local features. Authors also used multi-layer self-organising map to form the tree structure. In [29] a multi-level hierarchy was proposed and applied to text retrieval and natural language. Finally, in [30] a hierarchical structure to which dynamic indexing and guided search are applied using wavelet-based scheme for multiple features extracted from images in a warehouse. The hierarchy is, however, over the image colour, palm and face etc. Features are not over nested clusters/clouds of images. This approach will also struggle in terms of computational complexity for huge amount of images and sub-images or features.

Although, forming hierarchical structures for retrieving images has been explored by other researchers, their goals for doing so differ from our proposed method. We offer a hierarchy of nested clusters of mean values, not images and sub-images or features.

Last, but not least, it is important to select appropriate proximity and similarity measure used for clustering and search. Traditionally, Euclidean, Mahalonobis, cosine, Manhattan/city distance measures are used. In *Look-a-Like*, we use relative Manhattan (L_1) distance. However, all of these are distances between a given data sample and another data sample (e.g. image). There are also linkages between clusters (distance or dissimilarity measure between groups of images). In addition, the density in the data space as introduced and defined in [8, 9] provides an exact value between 0 and 1 of the similarity between a given data sample (e.g. image) and ***all*** images from a data cloud (or cluster). In the proposed approach, *Look-a-like* we uniquely use such measure of similarity which is not the same as the distance between two data samples (images) nor between groups of images, nor between an image and a mean of a cluster (mean of a cluster is often not an existing image, but an abstraction) only. Data density as defined in [8, 9] is a unique measure which allows quickly to be computed (because is recursive and in the proposed approach can be calculated in a hierarchically nested setting) the *exact* (not approximate) similarity between a given query image and as many other images as needed (e.g. *100* billion or *100* etc.).

3. The Proposed Approach *Look-a-Like*

The aim of the proposed approach is to provide an efficient and fast CBIR system to deal with big data streams in the form of images. *Look-a-like* is a quick strategy for search and retrieval of images in big dynamically evolving data streams. It is subject of a pending patent application [31]. It consist of three main elements:

(a) multiple features extracted from images which represent them in a computationally compact form in a unique way (that is, an image is converted to a vector of less than 700 floating point numbers per image);

(b) a hierarchically nested dynamically evolving data clouds (cluster-like) structure which facilitates the computationally efficient search and logical organisation of the images and is dynamically updated with each new available image including the query image using evolving local means (ELM) algorithm [10]:

(c) computationally efficient RDE formula for evaluating the similarity between a query image and a huge number of other images. The proposed approach is also using relative Manhattan (L_1) distance.

The proposed approach builds automatically a dynamically evolving hierarchically nested image clouds/clusters structure from unstructured big data streams (e.g. billions of images) facilitating the search of most relevant similar images using local density (see Fig. 1). From the computing realisation point of view, the proposed *Look-a-Like* can be realised as a client-server system (see Fig.2) which can be offered as a web service.

Maximum local density indicates the image cloud with mean values (if at the higher levels of the hierarchy) or images (if at the lower hierarchical level). Going down through the levels of the hierarchy, a cloud with a reasonably small (but not pre-defined) number of visually similar images can be identified for a very small amount of time (less than a second) from a big image (billions of images) stream. *Look-a-like* works with vectors of

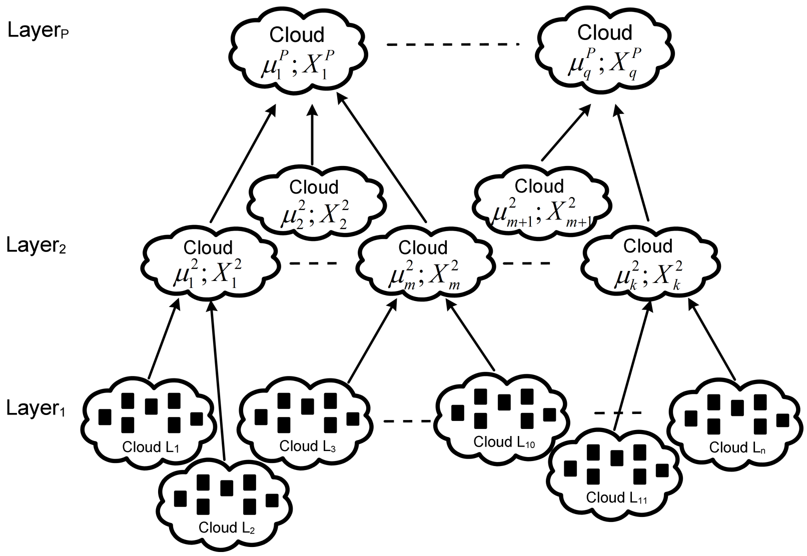


Figure 1: Schematic representation of the hierarchically nested data clouds structure, each square in layer one represents features of an image a described in section 3.1; μ and X denote mean values and scalar products which are abstract values and are described in section 3.3

multi-features (less than 700 floating point numbers per image) and means and accumulated scalar products. It is not using pixels directly; finally, it is using efficient local (per cloud) RDE formula [8] and relative Manhattan distance.

In what follows, we will, first, describe the set of features that has been used to achieve a high discrimination power. Next, we will recall the evolving local means algorithm, ELM [10] to form the data clouds. ELM is using the similar basic concept as the widely used mean-shift clustering algorithm; however, the local variance and local mean in ELM is calculated recursively and it is a non-iterative, one pass algorithm which makes it significantly faster (in orders of magnitude), especially for big data streams. The search itself is performed by calculating the local recursive density estimation (RDE) in regards to the query image and the data clouds (initially at the top layer of hierarchy, then between the query image and the data clouds that correspond to the winning data cloud of the top layer and all data clouds linked to it and so on going down to the lower layer of the hierarchical structure. Finally, a threshold, ε separates the images that are returned to the user from the data clouds of the lower level of hierarchy associated with the winning data cloud of the higher level of hierarchy as illustrated further.

It has to be stressed that local RDE calculates the exact similarity between a query image and ***all*** images from the winning data clouds recursively and, thus, computationally efficiently. Due to the recursive calculations, the proposed approach is very efficient computation- and time-wise. Furthermore, the proposed method involves search in an ordered multi-layer hierarchy (Fig. 3) such that search process is speeded up by orders of magnitude. The results show that the performance is very high quality and very fast for big data streams even on ordinary laptop using Windows OS and Matlab (using Linux OS and C/C++ language as well as parallelisation or use of GP GPU can further improve significantly the performance). The main reason is that by introducing the hierarchical organisation of the images combined with the RDE the number of comparisons is dramatically reduced yet the ***full*** and ***exact*** information of the comparison with all images from a data cloud is kept intact, Fig. 2.

3.1. Feature Selection

Having a selection of representative features is very important for the quality of the algorithm. In *Look-a-Like* we use a combination of multiple feature sets of different nature, with size of 697 floating point digits: $F = \left\{F^{G}; F^{HSV}; F^{M}; F^{C}; F^{LG}; F^{W}\right\}$ [32].

The first feature is GIST [33] which extracts the global features of the image and gives an impoverished and coarse version of the principal contours and textures of the image which is still detailed enough to recognize the image. It is computationally efficient and there is no need to parse the image or group its components in order to represent the spatial configuration of the scene. The fundament of GIST approach is Gabor filters. Several Gabor filters with selected channels are computed on a grid of the image (4×4) and indexed into an array with 512 features, F^{G}.

The second feature is a colour HSV histogram. To extract colour histogram, each pixel of an image is associated to a specific histogram bin on the basis of its own colour. HSV colour space is used for histogram generation where each pixel contributes its intensity and improves perceptual uniformity. Each image is quantised in the HSV colour space into $8 \times 2 \times 2$ equal bins, which creates a feature vector with 32 features, F^{HSV}.

Since it has been proven [34] that colour moments are more robust and have a better performance in comparison with the colour histogram, they are selected as a third set of extracted features. Three central moments (mean, standard deviation and skewness) can be used for image's colour distributions [35]. In *Look-a-Like*, we define 9 moments (3 moments for

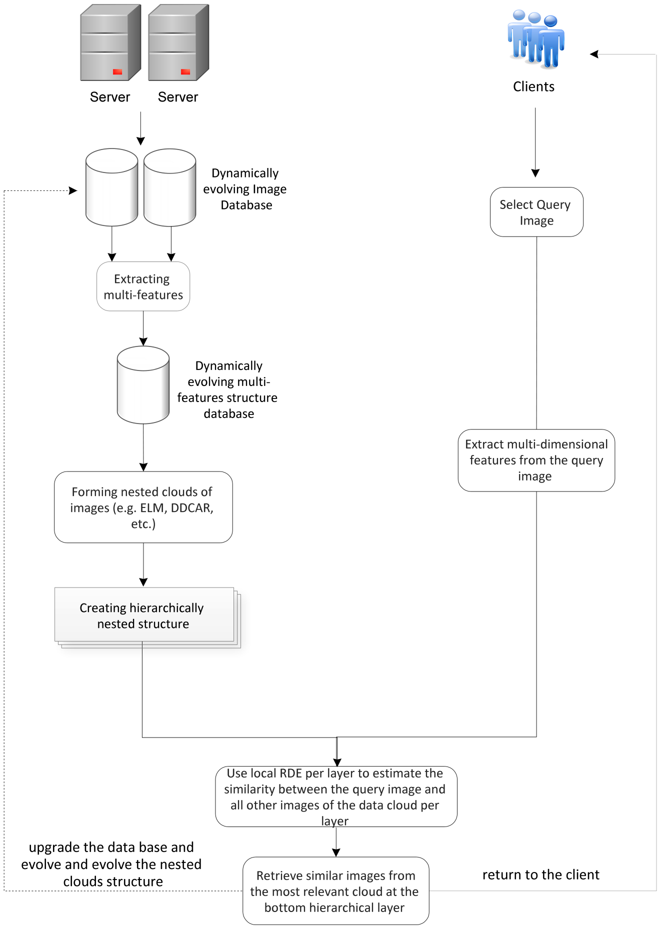

Server
Server
Clients
Dynamically evolving Image Database
Select Query Image
Extracting multi-features
Dynamically evolving multi-features structure database
Extract multi-dimensional features from the query image
Forming nested clouds of images (e.g. ELM, DDCAR, etc.)
Creating hierarchically nested structure
Use local RDE per layer to estimate the similarity between the query image and all other images of the data cloud per layer
upgrade the data base and evolve and evolve the nested clouds structure
Retrieve similar images from the most relevant cloud at the bottom hierarchical layer
return to the client

Figure 2: Schematic representation of the proposed approach *Look-a-like*

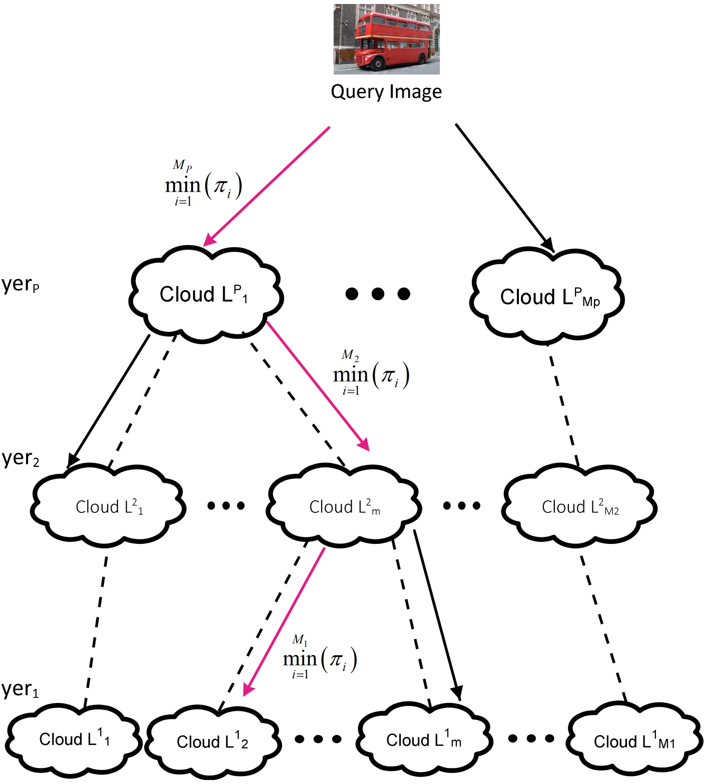


Figure 3: Schematic representation of the hierarchical data cloud structure (in fact, we use matrices of feature vectors instead of the actual images). In layer2 and above we store the mean values of the data clouds in terms of those features which do not necessarily (and usually) represent an image. The principle of '*winner takes all*' based on maximum local RDE value at each layer is used to find the winning data cloud. Finally, **all** images from the layer1 winning data cloud are displayed which may be optionally restricted by a threshold ε (suggested value for ε is around 20).

each colour channel), F^M. The *c-th* colour channel of the *i-th* image pixel P_{ci} is defined by:

i the average colour value in an image:

$$M_c = \frac{1}{A}\sum_{i=1}^{A} p_{ci} \tag{1}$$

where $A = H \times W$ H = height, W = width

ii the variance

$$\sigma_c = \sqrt{\frac{1}{A}\sum_{i=1}^{A} (p_{ci} - M_c)^2} \tag{2}$$

iii the skewness which is a measure of the degree of asymmetry in the distribution:

$$S_c = \sqrt[3]{\frac{1}{A}\sum_{i=1}^{A} (p_{ci} - M_c)^3} \tag{3}$$

The colour auto-correlogram is the fourth extracted feature set which describes how the spatial correlation of colour changes with the distance. If the distance $d \in [n]$ is a fixed *priori*, the correlogram of an image, I is defined for $i, j \in [m]$ positions of pixels,$k \in d$ as [36]:

$$\beta^k_{c_i,c_j}(I) \equiv \Pr\left[|p_1 - p_2| = k, p_2 \in I_{C_j} | p_1 \in I_{C_i}\right] \tag{4}$$

where $|p_1 - p_2| \overset{\Delta}{=} \max\{|x_1 - x_2|, |y_1 - y_2|\}$

Given any pixel of colour C_i in the image I, $\beta^k_{c_i,c_j}$ gives the probability that a pixel at distance k away from the given pixel's colour [36]. For each pixel in the image, the auto-correlogram method applies to all the neighbours of that pixel. If the distance is large, a large area will be covered and more information will be collected from the image; however, the computational complexity will increase. In order to address the computational complexity, the set$S$ is used which is a subset of $d(S = 1; 3; 5; 7)$[34] resulting in a 64 features vector, F^C which are added/appended to F.

The next set of features which is being used is based on the texture representation. Gabor wavelet transform is widely used to represent texture of images and has been demonstrated to be very efficient. However, the bandwidth of the Gabor filter is limited to one octave; therefore, a large number of filters is required to obtain wide spectrum coverage. In addition, their response is symmetrically distributed around the centre frequency, which results in redundant information in the lower frequencies that could instead be devoted to capturing the tails of images in the higher frequencies.

The log-Gabor function is used as an alternative to Gabor function [37] designed as Gaussian functions on the log axes. It has been proven that log-Gabor filter outperforms the standard Gabor filter in order to verify an object inside an image [38]. Their symmetry on the log axes results in a more effective representation of the uneven frequency content of the images. Furthermore, log-Gabor filters do not have a DC component, which allows an increase in the bandwidth which results in fewer filters to cover the same spectrum. The log-Gabor filters are defined in the log-polar coordinates of the Fourier domain as shifted from the origin Gaussians [39]:

$$G_{(s,o)}(\rho,\theta) = \exp\left(-\frac{1}{2}\left(\frac{\rho-\rho_s}{\sigma_\rho}\right)^2\right)\exp\left(-\frac{1}{2}\left(\frac{\theta-\theta_{(s,o)}}{\sigma_\theta}\right)^2\right) \tag{5}$$

$$\begin{aligned} &\rho_s = \log_2(n) - s \\ &\theta_{(s,o)} = \begin{cases} \frac{\pi}{n_o} o \;\; if\ s\ is\ odd \\ \frac{\pi}{n_o}(o+\frac{1}{2}) \; if\ s\ is\ even \end{cases} \\ &(\sigma_\rho,\sigma_\theta) = 0.996\left(\sqrt{\tfrac{2}{3}}, \tfrac{1}{\sqrt{2}}\tfrac{\pi}{n_o}\right) \end{aligned} \tag{6}$$

where s and o specify the scale and orientation of the wavelet, respectively ($s = 0, 1, ..., n_s$;$t = 0, 1, ..., n_o$); and (ρ, θ) are the log-polar coordinates. $\left(\rho_s, \theta_{(s,o)}\right)$ are the coordinates of the centre of the filter and $(\sigma_\rho, \sigma_\theta)$ are the bandwidths. Let F denote the Fourier transform of the input image. The convolution of $G_{s,o}$ and F is obtained by [40]:

Let F denote the Fourier transform of the input image. The convolution of $G_{s,o}$ and F is obtained by [40]:

$$V_{s,o} = F * G_{s,o} \tag{7}$$

An array of magnitudes is obtained as:

$$E_{s,o} = \sum_i \sum_j |V_{s,o}(i,j)| \tag{8}$$

where (i,j) denotes the 2D coordinates of a pixel $p_{i,j}$.

These magnitudes represent the energy content at different scale and orientation of the image. The main goal of the texture-based retrieval is to find images or regions with similar texture. It is assumed that we are interested in images or regions that have homogenous texture; therefore, the following mean, μ_{so} and standard deviation, σ_{so} of the magnitude of the transformed coefficient are used to represent the homogenous texture of the region as a feature:

$$\mu_{so} = \frac{E_{s,o}}{N} \tag{9}$$

$$\sigma_{so} = \frac{\sqrt{\sum_i \sum_j \left(|G_{so}(i,j)| - \mu_{so}\right)^2}}{N} \tag{10}$$

A feature vector is constructed using $\mu_{s,o}$ and $\sigma_{s,o}$. In our experiment the scale was set to 5 and orientation to 6 which results in a feature vector F^{LG}, of size 30 for each $\mu_{s,o}$, $\sigma_{s,o}$.

The wavelet transform is a multi-resolution analysis technique for an image and it has been proven to work well in both space and frequency domain [41]. It is used as the final set of features. Any decomposition of the image into a wavelet involves a pair of waveforms; the high frequency components correspond to the details of an image while the low frequency components correspond to its smooth parts [42]. Discrete Wavelet Transform (DWT) of an image as a 2D signal can be derived from a 1D DWT, implementing 1D DWT to every row then implementing a 1D DWT to every column. Any decomposition of the 2D images into a wavelet involves 4 sub-band elements representing LL (Approximation), HL (Vertical Detail), LH (Horizontal Detail), and HH (Detail), respectively [42]. The DWT of a signal x is calculated by passing it through a low pass filter with impulse response h and high pass filter g. The outputs giving the detail coefficients (from the low pass and high-pass filter) and approximation coefficients.

$$w_{low}[n] = \sum_{k=-\infty}^{\infty} x[k]\, h[2n-k] \tag{11}$$

$$w_{high}[n] = \sum_{k=-\infty}^{\infty} x[k]\, g[2n-k] \tag{12}$$

After resizing the image into 256 × 256 matrix, we applied a 4-level wavelet transformation. The upper left 16 × 16 matrix is stored and also divided into its high and low frequency components, as part of the feature vector. Finally, we calculated the mean and standard deviation of the 16×16 matrix to construct the feature vector. The final size of the feature vector is composed of two sets of 16 features each (32 in total), F^{W}.

As a result of applying these six sets of features a vector with size 697 is formed as $F = \left\{F^{G}, F^{HSV}, F^{M}, F^{C}, F^{LG}, F^{W}\right\}$ [32].

3.2. Forming data clouds

Similarity comparison between the query image and each image from a large collection can be computationally prohibitive and very slow. In addition, it is impossible to compare the query image with all the images in the World Wide Web with its vast and increasing size individually. Therefore, automatically arranging/structuring of the images based on their similarity is essential, especially when the users need to narrow down their requirement to a particular subset. In this sense, it is useful to arrange the images into simple genres forming data clouds. Arranging massive amount of images in the World Wide Web generated every second is the toughest challenge. This is where one of the main innovation aspects of the proposed new Look-a-like approach lies. If we try to implement some of the classical clustering algorithms such as k-means, fuzzy C-means etc. this is not practical due to their fixed structures and pre-defined number of clusters, prohibitive computational costs etc. In addition, storing the huge amount of image data in the memory and processing them is another challenge that needs to be addressed. Moreover, the amount of images in the World Wide Web is not limited or fixed and traditional approaches would require the task to be resolved each time again and again, which is also prohibitive. Therefore, we need a computationally efficient, recursive and dynamically evolving algorithm for data partitioning/forming data clouds.

In *Look-a-like* we use the recently introduced ELM [10] algorithm (using the feature vectors of size 697 floating point values as described above and in [32], however, an alternative is the recently introduced DDCAR method

[43]. The advantage of DDCAR is that it is fully autonomous and does not require any parameter to be pre-specified (for comparison, even ELM does require the radius, r to be pre-specified).

ELM is based on the concept of non-parametric gradient estimate of the density function using local (per data cloud/rubric) means [10]. The local means are being updated for each new coming feature vector/image descriptor allowing for the data set to evolve/expand (as is the case with the World Wide Web, for example). New data clouds are being formed if the density pattern changes, a cloud is created. In that case, the evolving nature of ELM can be useful if new images are added to the database. For each image cloud, i that is being formed we can calculate the local mean, μ_i and variance, σ_i. The mean does not necessarily (and usually) represent a meaningful image but is rather an abstraction/focal point of the cloud. Details of the ELM approach are provided in [10]. Initially a radius of the data cloud is being defined. In terms of the feature vector which was defined in the previous sub-section [32], the initial radius value was chosen to be 150 for the lower hierarchical layer and 250 for the higher/top layer (the units are related to the unnormalised feature vector. As a new image (feature vector) is being processed, the distance/dissimilarity to all existing data clouds is computed. If the following condition is satisfied, then the image I is assigned to the data cloud i:

$$d_i < (\max(\|\sigma_i\|, r) + r) \tag{13}$$

where d_i is the distance from image I to the data cloud mean μ_i. r is a pre-specified radius of the cloud.

If this condition for the image I is true for more than one data cloud, the nearest data cloud is selected. After assigning the new coming image to an existing data cloud, the mean of the data cloud μ_i and the variance, σ_i are updated recursively as detailed in [10].

3.3. Similarity measure based on the local RDE

The next step after forming the clouds and the hierarchical structure is to find the cloud which contains the most similar images to the query. In order to do that, we use local recursive density estimation, RDE [8, 9]. An alternative is the recently introduced typicality measure [44]. Both of them give an estimate of the similarity between the query and ***all*** images from the clouds, Fig. 4. Such a recursive technique makes possible that each image is considered only once and discarded once it has been processed and not kept in the memory, but the information is still exact (not approximate)

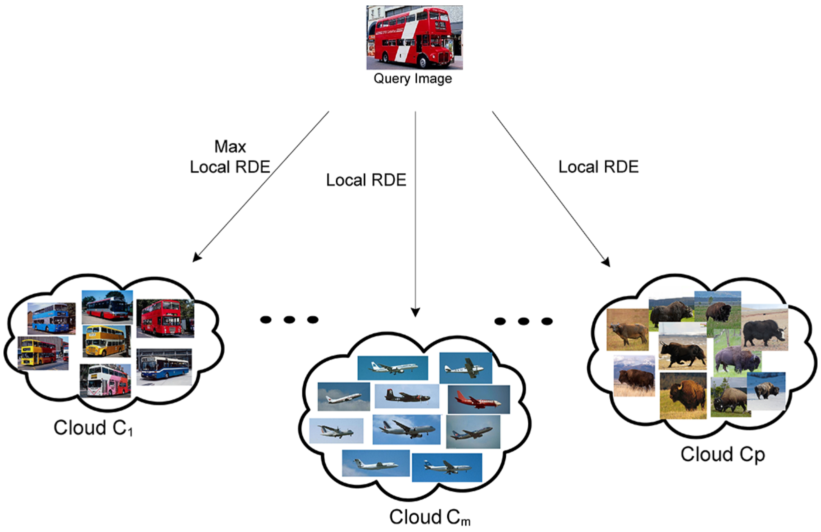


Figure 4: Computing similarity between a given query image, Q and clouds using local density at the highest hierarchical level

in terms of similarity between the query and each individual image from the clouds [8, 9]. Only the information concerning the density (mean, μ and the scalar product, X) is accumulated and stored for each cloud in the memory. Moreover, it makes possible to use a significantly smaller (in orders of magnitude) amount of computations. Due to recursive nature of the algorithm, if compare with the case when the query image is compared with each image from the cloud individually, it is computationally efficient and fast.

In *Look-a-Like*, the degree of similarity of a query image to all images inside a cloud is measured by the relative density in regards to the query image:

$$\gamma_k^i = \frac{1}{1 + \left\| F_k - \mu_k^i \right\|^2 + X_k^i - \left\| \mu_k^i \right\|^2} \tag{14}$$

In practice, it is more convenient (and in accordance with the typicality, [44]) to consider the accumulated proximity, π:

$$\pi_k^i = \frac{1}{\gamma_k^i} - 1 \tag{15}$$

where $F = \{f_1, \ldots, f_{697}\}$ is the representation of the image with its feature vector, $k = 1, 2, \ldots, M_i$; $i = 1, 2, \ldots, C$, M_i is the number of images

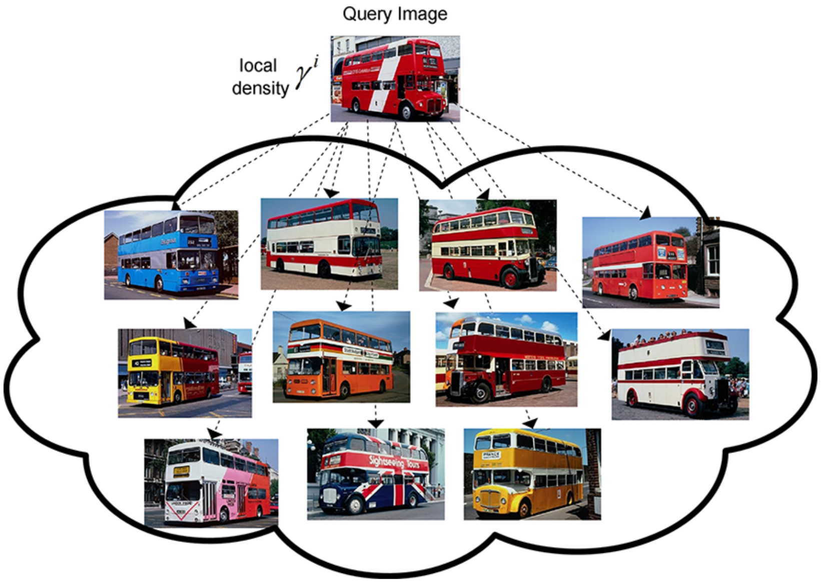


Figure 5: Computing local density γ^i of a query image to **all** images in i^{th} cloud

within the i^{th} cloud; C is the number of clouds.

Both, the mean, μ_k and the scalar product, X_k can be updated recursively as follows [9]:

$$\mu_k = \frac{k-1}{k}\mu_{k-1} + \frac{1}{k}F_k;\ \mu_1 = F_1 \tag{16}$$

$$X_k = \frac{k-1}{k}X_{k-1} + \frac{1}{k}\|F_k\|^2;\ X_1 = \|F_1\|^2 \tag{17}$$

The cloud with the maximum local density in respect to the query image is the winner which contains images that are most relevant/similar to the query image:

$${C_k}^* = \underset{i=1}{\arg\min}\left\{\pi_k^i\right\} \tag{18}$$

Once the winning cloud is selected, the image that are contained in it are re-ranked using relative Manhattan/L_1 distance which yields best results and gives the more significant difference between two images [32]. Small distance implies that the corresponding image is more similar to the query image and vice versa. The relative Manhattan distance between the query image and images inside the selected cloud is computed as follows:

$$d\left(Q, I^j\right) = \sum_{k=1}^{n} \frac{\left|Q_k - I_k^j\right|}{1 + Q_k + I_k^j}; j = 1, \ldots, M_i \tag{19}$$

where M_i is the number of images of a certain cloud; n is the number of extracted features, in this work n=697 [32]. The final result includes all images from the selected/wining cloud up to a certain threshold in terms of d, ε (recommended values $\sim$ 20).

$$IF\ \left(d\left(Q, I^j\right) < \varepsilon\right)\ THEN\ \left(display\ I^j\right) \tag{20}$$

4. Experimental Results

In this section, the experimental results are presented. The proposed approach, *Look-a-like* has been evaluated in terms of the speed and accuracy. It was tested with an image database which includes 65,000 images collected within the WANG database [45] by the visual Geometry group at the University of Oxford [46]. The database contains over 600 classes which makes it an ideal example to evaluate the performance of CBIR systems. Some of the image classes are illustrated in Fig. 6. It should be noted that the number of images is not the same for all classes.

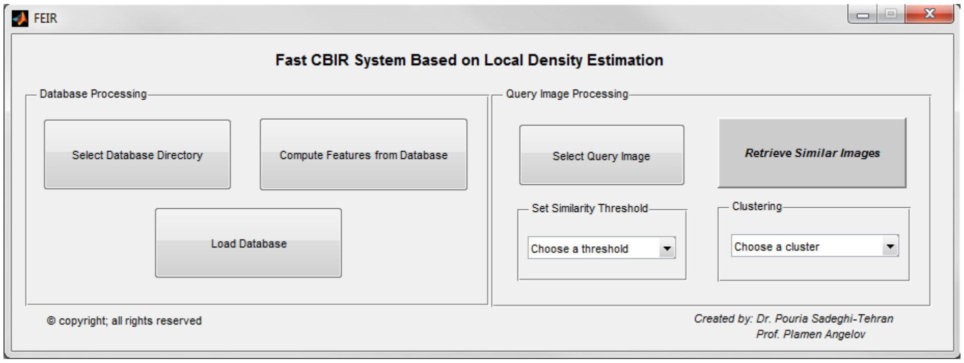
The tests were performed on a standard PC with Intel Core i7, processing power with 3.4 GHz CPU and 8 GB RAM running Windows 7 operating system. A graphical user interface (GUI) application was developed in MATLAB environment (Fig. 7) to facilitate the evaluation work.

The test starts with the user uploading a query image and retrieving the similar images. Users can select a threshold, ε to retrieve the most similar images (we used ε=23). At the end, the final search result was saved in HTML format and ready to publish on the Web. Execution time of the proposed nested hierarchical system and an alternative of clustering and a non-hierarchical system were also compared.

4.1. Speed evaluation of the proposed approach

The execution time of the proposed *Look-a-Like* was tested on several randomly selected queries, such as bikes, planes, cars, and sharks. Figure 9 shows the execution time for the case of; a) direct comparison of the query image with each of the 65.000 images (no clustering); b) with clouds when no hierarchy of nested clouds is built up; c) when two-layer hierarchy is built first using randomly selected query image. In the non-hierarchical system the similarity value is computed between the query image and all the images

Figure 6: Example images from the dataset that was used (65,000 images in total)



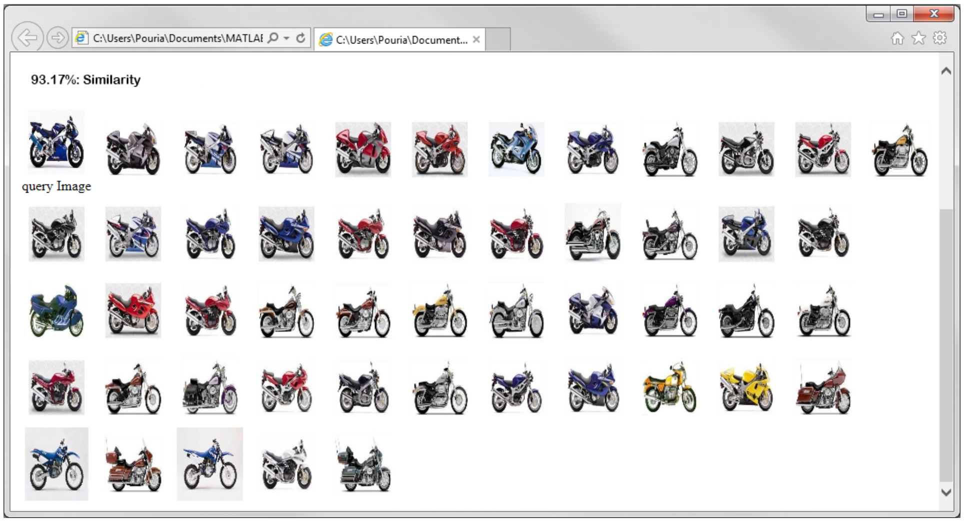
FEIR

Fast CBIR System Based on Local Density Estimation

© copyright; all rights reserved

Created by: Dr. Pouria Sadeghi-Tehran
Prof. Plamen Angelov

Figure 7: The interface of the test environment for the proposed approach



93.17%: Similarity

Figure 8: All images from the winning cloud are shown after re-ranking

Table 1: A two layer example of the proposed hierarchically nested approach

Two Layer Hierarchically Nested Structure		
	Radius	No. Clouds
Layer one	150	697
1-2 Layer two	250	36

or lower layer clouds. In the hierarchical system the comparison is made only with the top layer clouds and after determining the winner cloud the further search at the lower layers is performed only with the clouds which correspond to that cloud significantly reducing the amount of comparisons. ELM method is used for forming the clouds with radius set to 150 for the lower layer and 250 for the upper layer. At the lower layer all 65,000 images were grouped into 697 clouds. It has to be stressed that some of them have a single image and were ignored. At the higher layer the means of the clouds at the lower layer that were not eliminated due to the small number of images which they contain were further grouped suing ELM and a radius of 250. This resulted in 36 higher/top layer clouds, see Table 1.

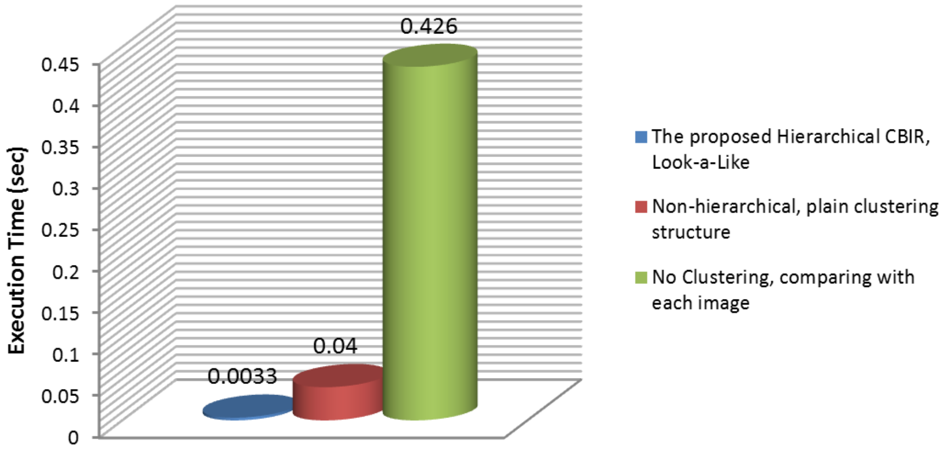


Figure 9: Execution times in seconds on different setups

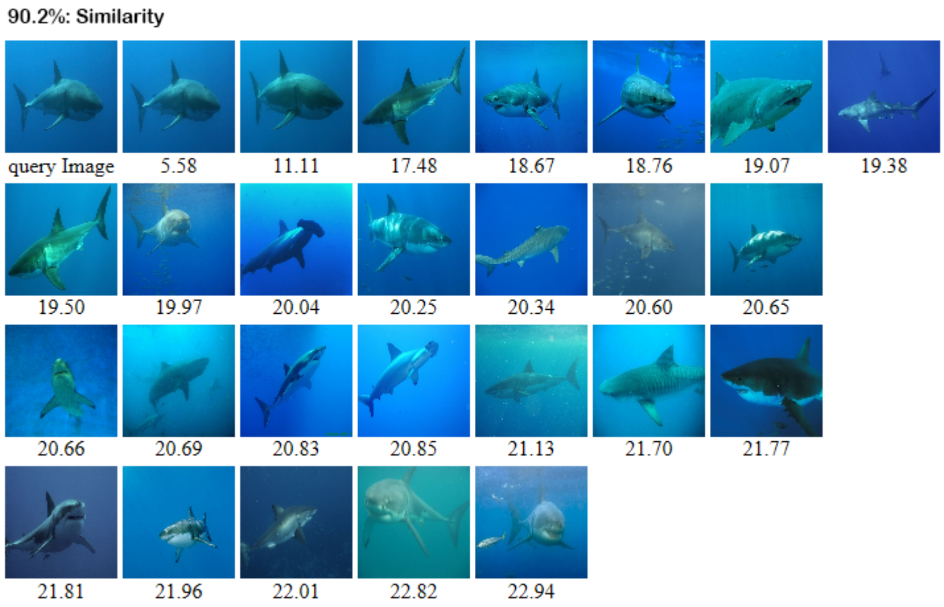


Figure 10: Results for searching sharks (the value under each image represents the Manhattan distance as described earlier) The value of similarity is the local RDE value, eq.(14)

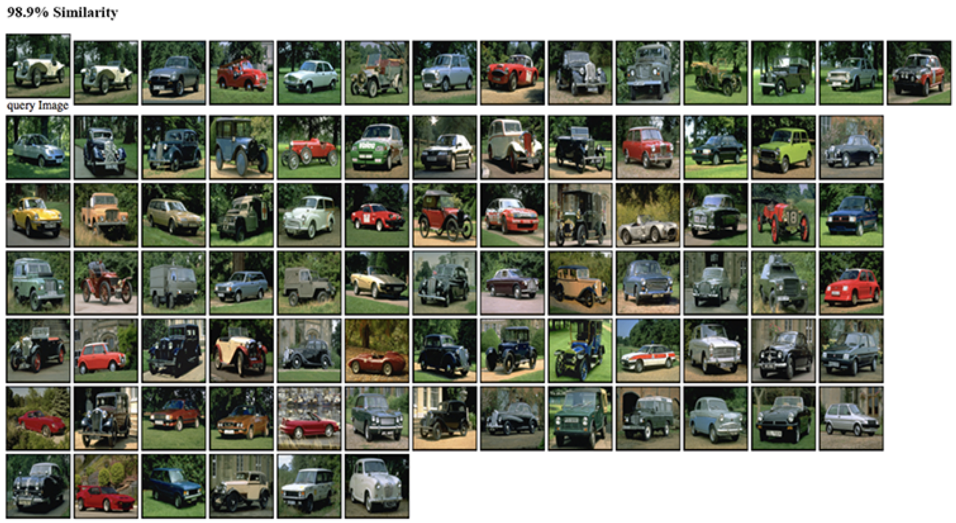


Figure 11: Results for searching cars

5. Conclusion

In this paper, a new fast approach for organisation and search within CBIR context has been proposed. Its main idea is to organise the otherwise unstructured set of complex, multi-dimensional data (images) into a dynamically evolving hierarchically nested clouds structure using a combined multiple sets of features and a computationally efficient local-RDE-based similarity measure. The approach was tested on a data base which contains 65,000 images from about 600 different genres/rubrics. The proposed *Look-a-like* approach was able to automatically form 697 lower layer clouds and 36 higher/top layer clouds and for a given query image it provided visually very relevant results within few milliseconds making only about 50 calculations of the local RDE formula. The approach is scalable and parallelisable in nature (different data clouds can reside on different hardware or multi-core application can benefit from parallelisation, too). It can be realised as a web service. It is also possible to include user feedback in a future application. The method is a subject of a patent application [31].

6. Bibliography

[1] T. Blog, “Internet 2011 in numbers http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers/,” Jan. 2012.

[2] T. Kato, "Database architecture for content-based image retrieval," *Proc. SPIE 1662, Image Storage and Retrieval Systems*, vol. 1662, pp. 112–123, Jan. 1992.

[3] TinEye, "http://www.tineye.com/," Jan. 2012.

[4] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin, "The QBIC project: querying images by content using colors, texture and shape," pp. 173–187, Jan. 1993.

[5] R. Consoli, "http://yandex.ru/images/?d=IBQIKQ."

[6] W. Y. Ma and B. S. Manjunath, "NeTra: a toolbox for navigating large image databases," in *Image Processing, 1997. Proceedings., International Conference on*, pp. 184–198, 1997.

[7] A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook: Content-based manipulation of image databases," *International Journal of Computer Vision*, vol. 18, pp. 233–254, Jan. 1996.

[8] P. Angelov, "Anomalous system state identification." patent application, GB1208542.9, May 2012.

[9] P. Angelov, *Autonomous Learning Systems: From Data Streams to Knowledge in Real Time.* John Wiley and Sons, Jan. 2012.

[10] R. D. Baruah and P. Angelov, "Evolving Local Means Method for Clustering of Streaming Data," *IEEE World Congress on Computational Intelligence*, pp. 2161–2168, Jan. 2012.

[11] J. Eakins and M. Graham, "Content-based image retrieval," tech. rep., Jan. 1999.

[12] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, "A survey of content-based image retrieval with-level semantics," *Journal of Pattern Recognition Society*, vol. 40, pp. 262–282, Jan. 2007.

[13] W.-T. Chen, W.-C. Liu, and M.-S. Chen, "Adaptive Color Feature Extraction Based on Image Color Distributions," *IEEE Transaction on Image Processing*, vol. 19, pp. 2005–2016, Jan. 2010.

[14] K. Hua, K. Vu, and J. Oh, "SamMatch: a flexible and efficient sampling-based image retrieval technique for large image databases," in *Proceedings of the seventh ACM international*, Jan. 1999.

[15] P. L. Stanchev, D. Green Jr., and B. Dimitrov, “High level color similarity retrieval,” *International Journal on Theories Applications*, vol. 10, pp. 363–369, Jan. 2003.

[16] J. R. Smith and S.-F. Chang, “Automated binary texture feature sets for image retrieval,” in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, pp. 2239–2242, 1996.

[17] I. Daubechies and B. J. Bates, *Ten lectures on wavelets*, vol. 93. The Journal of the Acoustical Society of . . . , 1993.

[18] B. S. Manjunathi and W. Y. Ma, “Texture Features for Browsing and Retrieval of Image Data,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 837–842, Jan. 1996.

[19] Y. Rui, A. C. She, and T. S. Huang, “Modified Fourier descriptors for shape representation-a practical approach,” *Proc of First International Workshop on Image Databases and Multi Media Search*, 1996.

[20] D. Cai, X. He, Z. Li, W. Ma, and J. Wen, “Hierarchical clustering of WWW image search results using visual, textual and link information,” in *Proceedings of the 12th annual ACM*, pp. 952–959, Jan. 2004.

[21] N. Ben-Haim, B. Babenko, and S. Belongie, “ImprovingWeb-based Image Search via Content Based Clustering,” in *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW ’06. Conference on*, pp. 17–22, 2006.

[22] P. Dutta, D. Bhattacharyya, and J. Kalita, “Clustering approach to content based image retrieval,” *Modeling and Imaging*, pp. –, Jan. 1993.

[23] E. Yildizer, A. M. Balci, T. N. Jarada, and R. Alhajj, “Integrating wavelets with clustering and indexing for effective cotent-based image retrieval,” *Knowledge-Based Systems*, vol. 31, pp. 55–66, Jan. 2012.

[24] T. Deselaers, D. Keysers, and H. Ney, “Clustering visually similar images to improve image search engines,” *In Informatiktage 2003 der Gesellschaft fr Informatik, Bad Schussenried, Germany.*, Jan. 2003.

[25] Y. Chen, J. Wang, and R. Krovetz, “Content-based image retrieval by clustering,” *on Multimedia information retrieval*, Jan. 2003.

[26] L. Yang, X. Qi, F. Xing, T. Kurc, J. Saltz, and D. J. Foran, "Parallel content-based sub-image retrieval using hierarchical searching," *Bioinformatics*, Jan. 2013.

[27] R. Distasi, D. Vitulano, and S. Vitulano, "A Hierarchical Representation for Content-based Image Retrieval," *Journal of Visual Languages and Computing*, vol. 11, pp. 369–382, Jan. 2000.

[28] S. Chow, M. Rahman, and S. Wu, "Content-based image retrieval by using treestructured features and multi-layer self-organizing map," *Pattern Analysis & Applications*, vol. 9, pp. 1–20, Jan. 2006.

[29] R. Levinson and G. Ellis, "Multi-level hierarchical retrieval," *In 6th Annual Conceptual Graphs Workshop*, pp. 285–310, Jan. 1996.

[30] J. You and Q. Li, "On hierarchical content-based image retrieval by dynamic indexing and guided search," *Cognitive Informatics, 2009. ICCI '09. 8th IEEE International Conference on*, pp. 188–195, 2009.

[31] P. Angelov and P. Sadeghi-Tehran, "Data Structuring and Searching Method and Apparatus." patent application, GB1417807.3, Oct. 2014.

[32] P. Sadeghi-Tehran and P. Angelov, "An Approach to CBIR using a composite Multi-Feature Vector and Local Recursive Density Estimation," *Journal of Computer Vision and Image Understanding (submitted, October 2014)*.

[33] A. Oliva and A. Torralba, "Building the gist of a scene: The role of global image features in recognition," *Progress in brain research*, vol. 155, pp. 23–36, Jan. 2006.

[34] H. Yu, M. Li, H.-J. Zhang, and J. Feng, "Color texture moments for content-based image retrieval," in *International Conference on Image Processing*, pp. 929–932, 2002.

[35] M. Stricker and M. Orengo, "Similarity of color images," *IS&T/SPIE's*, pp. 381–392, Jan. 1995.

[36] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih, "Image indexing using color correlograms," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 762–768, 1997.

[37] D. J. Field, "Relations between the statistics of natural images and the response properties of cortical cells," *J. Opt. Soc. Amer*, vol. 4, pp. 2379–2394, Jan. 1987.

[38] J. Arróspide and L. Salgado, “Log-Gabor Filters for Image-Based Vehicle Verification,” *IEEE Transaction on Image Processing*, vol. 22, pp. 2286–2295, Jan. 2013.

[39] S. Fischer, F. Šroubek, L. Perrinet, R. Redondo, and G. Cristóbal, “Self-Invertible 2D Log-Gabor Wavelets,” *International Journal of Computer Vision*, vol. 75, pp. 231–246, Nov. 2007.

[40] M. Kuse, Y. F. Wang, and V. Kalasannavar, “Local isotropic phase symmetry measure for detection of beta cells and lymphocytes,” *Journal of pathology Informatics*, vol. 2, no. 2, 2011.

[41] S. Mallat, “Wavelets for a vision,” *Proceedings of the IEEE*, vol. 84, pp. 604–614, Jan. 1996.

[42] K. Arai and C. Rahmad, “Wavelet Based Image Retrieval Method,” *(IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 3, pp. 6–11, Jan. 2012.

[43] R. Hyde and P. Angelov, “DDCAR: Data Density based Clustering with Automated Radius,” in *IEEE International Symposium on Evolving and Autonomous Learning Systems (EALS’14 to appear)*.

[44] P. Angelov, “Anomaly Detection,” *Journal of Automation, Mobile Robotics and Intelligent Systems (JAMRIS)*, vol. 8, pp. 29–35, 2014.

[45] J. Z. Wang, “http://wang.ist.psu.edu/docs/related.shtml,” Jan. 2004.

[46] U. Oxford, “Visual Geometry Group http://www.robots.ox.ac.uk/˜vgg/data/.”