Content-Based Image Indexing by Data Clustering and Inverse Document Frequency

Rafał Grycuk, Marcin Gabryel, Marcin Korytkowski, and Rafał Scherer

Institute of Computational Intelligence, Częstochowa University of Technology
Al. Armii Krajowej 36, 42-200 Częstochowa, Poland
{rafal.grycuk,marcin.gabryel,marcin.korytkowski,rafal.scherer}@iisi.pcz.pl
http://iisi.pcz.pl

Abstract. In this paper we present an algorithm for creating and searching large image databases. Effective browsing and searching such collections of images based on their content is one of the most important challenges of computer science. In the presented algorithm, the process of inserting data to the database consists of several stages. In the first step interest points are generated from images by e.g. SIFT, SURF or PCA SIFT algorithms. The resulting huge number of key points is then reduced by data clustering, in our case by a novel, parameterless version of the mean shift algorithm. The reduction is achieved by subsequent operation on generated cluster centers. This algorithm has been adapted specifically for the presented method. Cluster centers are treated as terms and images as documents in the term frequency-inverse document frequency (TF-IDF) algorithm. TF-IDF algorithm allows to create an indexed image database and to fast retrieve desired images. The proposed approach is validated by numerical experiments on images with different content.

Keywords: CBIR, content based image retrieval, image database, keypoints, clustering, inverse document.

1 Introduction

Content image retrieval is one of the greatest challenges of modern computer science. Along with the development of the Internet and the ability of capturing images by devices such as digital cameras and scanners, image databases are growing very rapidly. Effective browsing and retrieving images by users is required in many various fields of life e.g. medicine, architecture, forensic, publishing, fashion, archives and many others. In order to meet those expectations, many general systems of retrieving have been already presented [1,10,12,21].

In the process of image recognition users search through databases which consist of thousands, even millions of images. The aim can be the retrieval of a similar image or images containing certain objects. Retrieving mechanisms use image recognition methods. This is a sophisticated process which requires the use of algorithms from many different areas such as computational intelligence

S. Kozielski et al. (Eds.): BDAS 2014, CCIS 424, pp. 374–383, 2014.

[4,9,17], mathematics [8] and image processing [7]. Those algorithms do not use raster graphics, on the contrary they find certain distinguishing features. In the literature it is possible to find many diverse methods of extracting those features. They extract for example primary features such as: colour, texture, shape and their position on the image. The features can be subjected to additional algorithms which are able to reduce their number. It is possible thanks to background elements exclusion, to eliminate the features absent in other images with the same object etc. These methods facilitate data storing in the database and accelerate future retrieving process.

The presented algorithm was based on the well known and commonly used in the literature algorithms: mean shift [3], TD-IDF [19] and SIFT [13]. The major task of the SIFT algorithm is extraction from an image appropriate local features around key points, independent from size and position of the objects. Those key points are saved as vectors with constant length. Most frequently it is a vector of 128 numbers. Despite this data reduction, the number of key points obtained from all images which are supposed to be saved in the database is still huge. Thus, we resort to another possibility to decrease number of the points, i.e. clustering. There are various commonly used clustering algorithms (k-means, SOM, k-nn). Nevertheless, they need initial number of groups to be determined. In the approach presented here, we use our novel version of the mean shift algorithm for which we do not need this type of knowledge. Certain number of key points will be assigned to each group. These points vectors values will be replaced with the number of the group they belong to. Thanks to this operation, considerable number of data saved in databases will be reduced. Only vectors of the centres of groups and numbers assigned to key points will be saved. It is definitely more advantageous than storing thousands or millions of key points in databases.

Database index as well as quick retrieval are created thanks to the TF-IDF algorithm. The algorithm is widely used as a method of assessment how relevant a given document is. The evaluation takes place for example in Internet search engines. In our case, terms will be treated as centers of groups and documents as images. In this way we obtain a mechanism for quick and efficient similar images retrieval.

In the literature we can find many similar solutions to our approach [11,15,16,20]. Each of them uses some combination of algorithms for image feature extraction, clustering and indexing. Our approach is distinguished by a new, modified method of clustering, which allows to be adjusted the kernel parameters of the mean shift algorithm to the data that are available in the database.

The paper consists of several sections. In further sections we will present available algorithms, modifications in the mean shift algorithm as well as the proposed algorithm. The last section presents the numerical simulations on an original software written in .NET.

2 Clustering Algorithm

The mean shift clustering algorithm [2,5,6] is a method that does not require any parameters such as cluster number or shape. The number of parameters of the

algorithm is limited to the radius h, that determine the range of the clusters [3]. Mean shift determines the points in d-dimensional space as a probability density function, where the denser regions correspond to local maxima. For each data point in the feature space, one performs a gradient ascent procedure on the local estimated density until convergence. Points assigned to one cluster (stationary point) are considered to be a part of the cluster. Given n points $x_i \in R^d$, multivariate kernel density function $K(x)$ is expressed using the following equation

$$\widehat{f}(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right), \tag{1}$$

where h defines the radius of the kernel function. Kernel function is defined as

$$K(x) = c_k k(\| x \|^2), \tag{2}$$

where c_k represents a normalization constant. With a density gradient estimator we can make the following calculations:

$$\nabla \widehat{f}(x) = \frac{2c_{k,d}}{nh^{d+2}} \underbrace{\left[\sum_{i=1}^{n} g\left(\| \frac{x - x_i}{h} \|^2\right)\right]}_{term1} \underbrace{\left[\frac{\sum_{i=1}^{n} x_i g\left(\| \frac{x - x_i}{h} \|^2\right)}{\sum_{i=1}^{n} g\left(\| \frac{x - x_i}{h} \|^2\right)} - x\right]}_{term2}, \tag{3}$$

where $g(x) = -k'(x)$ denotes the derivative a function of the selected kernel. The first $term1$ allows to determine the density, while the second $term2$ defines a mean shift vector $m(x)$. Where $m(x)$ is vector and t is algorithm step. Points toward the direction of maximum density and proportional to the density gradient can be determined at the point x, obtained with the kernel function K. The algorithm can be represented in the following steps:

1. Determine the mean shift vector, expressed by the formula $m(x_t^i)$,
2. Translate density estimation window: $x_{t+1}^i = x_t^i + m(x_t^i)$,
3. Repeat first and second step until: $\nabla f(x_i) = 0$.

3 TF-IDF Algorithm

TF-IDF algorithm (term frequency, inverse document frequency) [19] is an algorithm dedicated to fast document search. In the paper it is used to image key point indexing. TD-IDF algorithm determines the frequency of specific words in a given document and taking account of its occurrence in all documents. This value is calculated from the following formula

$$(tf - idf)_{i,j} = tf_{i,j} \times idf_i \ . \tag{4}$$

The left side of the equation contains the two components tf and idf. The first term is the frequency of the words, expressed by formula

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}, \tag{5}$$

where $n_{i,j}$ is the number of occurrences of the term t_i in the document d_j. The denominator contains the sum of occurrences all the words in the selected document d_j. The second component from (4) is idf_i and it is expressed by the following formula:

$$idf_i = \log \frac{|D|}{|\{d : t_i \in d\}|}, \tag{6}$$

where $|D|$ is the total number of documents and $\{d : t_i \in d\}$ the number of documents that contain at least one instance of document [18,19].

4 SIFT Algorithm

SIFT (Scale-invariant feature transform) is an algorithm used to detect and describe local features of an image. It was presented for the first time in [13] and is now patented by the University of British Columbia. For each key point, which describes the local image feature, we generate feature vector, that can be used for further processing. SIFT contains four main steps [14]:

1. Extraction of potential key points by scanning the entire image,
2. Selection of stable key points (resistant to change of scale and rotation),
3. Finding key point orientation resistant to the image transformation,
4. Generating vectors describing key point.

SIFT key point consists of two vectors. First one contains: point position (x,y), scale (Detected scale), response (Response of the detected feature, strength), orientation (Orientation measured anti-clockwise from +ve x-axis), laplacian (Sign of laplacian for fast matching purposes). The second one contains descriptor of 128 length. In order to generate key points SIFT require one input parameter minHessian. In our experiments we set this variable in 400. This value was obtained empirically.

5 Proposed Method for Fast Image Retrieval

Proposed algorithm in Fig.1 consists of several stages. The first step is to create feature matrix ($n \times 128$), where 128 is the length of the vector generated by the SIFT algorithm and N is the total number of key points from all the images. Then we execute the mean shift clustering algorithm. This algorithm has been modified to our needs by adding a method to automatically match the cluster window. As a result of the algorithm, we obtain a set of features, assigned to specific cluster. The next step is to create a dictionary structure. The dictionary contains: key - cluster number obtained from the mean shift algorithm, value - sorted array of image numbers.

Next, for each dictionary value we calculate $tf - idf$. Values directly show the frequency of each feature in the cluster. Search stage compares the key points of input image with indexed key points stored in the database. Input data of the algorithm is a key points matrix of the new image. Next, each key point

is assigned to a cluster number. Subsequently we calculate $tf - idf$ value. The last step is to compare the $tf - idf$ values with all clusters found on the query image. Another important step in the proposed algorithm is the ability to extend the created index by new images. This method is similar to the previous steps. First, we generate key points from a new image. Next, the algorithm generates new clusters (if needed) for the specified descriptors. Newly acquired clusters are added to the dictionary. The algorithm can be described in the form of steps. The first six steps create an database index that allows to perform content based search.

INPUT: Binary images
OUTPUT: Index for content based image search
foreach $image \in images$ **do**
  Generate Descriptors using SURF,SIFT or PCA SIFT;
  Load each descriptor into $featureArray$;
end
Estimate and set algorithm window (h parameter, see section 5.1)
Generate clusters using mean shift;
foreach $cluster \in clusters$ **do**
  Add Word($wordID, documentID$) to dictionary, where $wordID$ is cluster and $documentID$ is image ID;
end
foreach $cluster \in clusters$ **do**
  Calculate $tfidf$;
end

Algorithm 1. The algorithm steps

In result we obtain database index ready for searching (see Algorithm 2).

INPUT: Query image (model image)
OUTPUT: Retrieved, similar images
Generate descriptors for Query image;
foreach $descriptor \in QueryImageDescriptor$ **do**
  if $descriptor \in clusterCenters$ **then**
    Add cluster number into variable $similarGroups$;
  end
end
Remove distinctive values of $similarGroups$; **foreach** $cluster \in similarGroups$ **do**
  Get dictionary[$cluster$] value passing cluster number; Add these value into $similarImageIDs$;
end
Remove distinctive values of $similarImagesIDs$; Show similar images;

Algorithm 2. The search stage steps

Algorithm of adding new images to the existing knowledge base is as follows:
INPUT: New image
OUTPUT: Index for content based image search with new added image
Generate descriptors for new image;
Execute mean shift on new image descriptors;
foreach $newCluster \in newClusters$ **do**
  Add Word $(newCluster, imageID)$;
  Calculate $TF - IDF$ value;
end

Algorithm 3. Adding new image to existing index

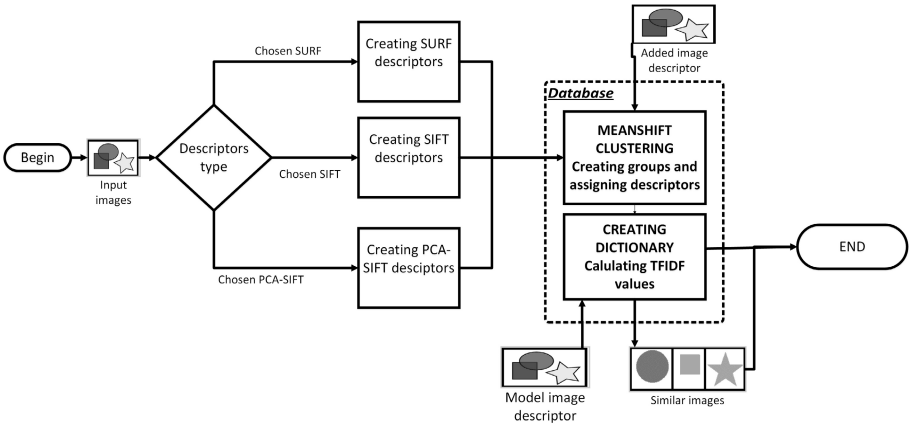

Fig. 1. Block diagram of the proposed algorithm

5.1 Estimation of h Parameter

In this section we present a novel, parameterless version of the mean shift algorithm. Mean shift is an algorithm that requires one input parameter h. Since this parameter depends on the number of generated groups. The small size of this parameter will generate a large number of groups with low coverage. On the other hand, a large range of the parameter will produce many groups which too much generalized data. The choice of this parameter is therefore very significant, because it determines the size of the database. We have developed a novel modification of mean shift consisting in estimation of the h parameter. The h parameter estimation algorithm is as follows:

1. Set the algorithm steps count K.
2. Calculate the distance $d_{i,j}$ between all points x_i and x_j where $i = 1, .., N, j = 1, .., N, i \neq j, N$ - the number of all the key points.
3. Determine the value of m parameter, that allow to increment h parameter by value m.

$$m = \frac{\max\limits_{\substack{i=1,\ldots,N \\ j=1,\ldots,N \\ i \neq j}} d_{i,j} - \min\limits_{\substack{i=1,\ldots,N \\ j=1,\ldots,N \\ i \neq j}} d_{i,j}}{K} \tag{7}$$

4. Algorithm step $n = 1$
5. Calculate h, depending on the step of the algorithm n

$$h = \min_{\substack{i=1,\ldots,N \\ j=1,\ldots,N \\ i \neq j}} d_{i,j} + m \cdot n \tag{8}$$

6. Calculation of the derivative of $g_{ij}(x)$ the kernel $k(x)$ for the cartesian product of all the key points i, j, and $i \neq j$. This stage is designed to accelerate the calculations performed during the execution of algorithm.

$$g_{ij} = g\left(\| \frac{x_j - x_i}{h} \|^2 \right) \tag{9}$$

7. Density gradient (3) is transformed to the form:

$$\nabla \widehat{f}(x_j) = \frac{2c_{k,d}}{nh^{d+2}} \left[\sum_{i=1}^{n} g_{ij}\left(x_j\right) \right] \left[\frac{\sum_{i=1}^{n} x_i g_{ij}\left(x_j\right)}{\sum_{i=1}^{n} g\left(x_j\right)} - x_j \right] \tag{10}$$

for $j = 1, .., N$.
8. Execute the mean shift algorithm. Save the clusters and cluster number generated by the algorithm.
9. Increase algorithm step $n = n + 1$,
10. While $h \leq max\ d_{ij}$, repeat steps 5-9,
11. The estimated parameter h is calculated by the formula (8) where n for such n that $G_n = \underset{i=1,\ldots,K}{\text{med}} G_i$.

6 Experimental Results

Experiments were carried out in .NET environment on our own software written C#. Research includes experiments on various objects with background. Test images were taken from [22]. We chose images with different types of objects such as: dinosaurs (100 images), drinks (100 images), landscapes (150 images), castles (230 images), meals (100 images), cars (100 images), cards (100 images), doors (100 images), dogs (100 images), flowers (100 images), mountains (100 images). All 1280 pictures generated a total of 99353 key points that were reduced to 18611 clusters. In experiments we used 90% of each class for index creating and 10% as query images. Our simulations prove effectiveness of the algorithm. Percentage effectiveness is presented in Tab.1. Fig. 2 presents exemplary results of the experiments. In top, left corner we presented a query image (red frame) (Q). The rest of images are retrieved by the the proposed algorithm. In tab. 1 we present the percentage effectiveness of our algorithm for estimated parameter $h = 288$ and fixed parameter h from 100 to 400. Fig. 3 shows a chart of average percentage of correctly found images from all categories versus the h parameter value. As can be seen, the parameter $h = 288$, obtained as a result of our algorithm, provides search results that are near to the optimum found for $h = 250$.

Table 1. Correctly found images

h parameter	dinosaurs	drinks	landscapes	castles	meals	cars	cards	doors	dogs	flowers	mountains
estimated $h = 288$	75%	92%	63%	61%	78%	77%	68%	95%	78%	97%	67%
fixed $h = 100$	7%	0%	0%	2%	1%	0%	1%	0%	1%	0%	0%
fixed $h = 150$	9%	1%	0%	3%	1%	2%	5%	10%	3%	4%	0%
fixed $h = 200$	55%	48%	32%	21%	46%	60%	37%	60%	56%	51%	43%
fixed $h = 250$	78%	90%	71%	59%	79%	80%	66%	93%	83%	95%	68%
fixed $h = 300$	68%	90%	59%	56%	70%	72%	63%	88%	75%	92%	66%
fixed $h = 350$	45%	41%	37%	17%	20%	20%	8%	33%	50%	46%	29%
fixed $h = 400$	37%	40%	35%	11%	18%	14%	5%	30%	47%	32%	24%

The differences in efficiency retrieval on image content is caused by image background. Some key points are located not only on objects, but also in background. Thus, the descriptors of these key points may be assigned by wrong cluster. If we have a similar background in an image class (e.g. dinosaurs) then most of images are correctly found. Otherwise the algorithm can retrieve images from a wrong class.

Fig. 2. Twenty example images from the experiment. Due to lack of space we had to narrow the example to 19 input (learning) images for each class and one query image. Top left image is a query image and the rest is 19 images retrieved by the algorithm. We can observe that 15 of them are retrieved correctly.

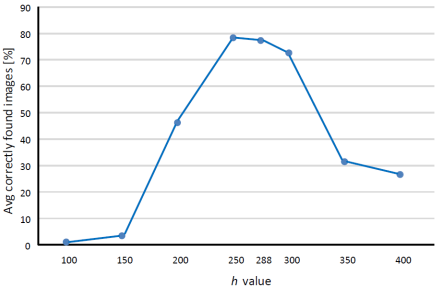

Fig. 3. Average percentage of correctly found images from all categories versus h parameter

7 Final Remarks

The presented algorithm is a contribution to fast content-based image search. Simulations verified the correctness of the algorithm. During initial simulations we discovered a method disadvantage, which is associated with the mean shift

algorithm. Mean shift requires h parameter which is the key points search radius (window). Thus, the algorithm requires an experimental value of this parameter which resulted in the lack of automation during the indexation phase. Presented, improved version of the mean shift algorithm allows for automatic operation of the application. Our paper is a part of a larger system that allows to search and identify specific classes of objects that can be located in different places and in different number of images. In the future we will try to develop a method that removes background from input images. Thus, it will allow to resolve the problem of retrieving images from different classes.

Acknowledgments. The work presented in this paper was supported by a grant from Switzerland through the Swiss Contribution to the enlarged European Union.

References

1. Chang, Y., Wang, Y., Chen, C., Ricanek, K.: Improved image-based automatic gender classification by feature selection. Journal of Artificial Intelligence and Soft Computing Research, 241 (2011)
2. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(8), 790–799 (1995)
3. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(5), 603–619 (2002)
4. Cpałka, K.: On evolutionary designing and learning of flexible neuro-fuzzy structures for nonlinear classification. Nonlinear Analysis: Theory, Methods & Applications 71(12), e1659–e1672 (2009)
5. Derpanis, K.G.: Mean shift clustering. Lecture Notes, `http://www.cse.yorku.ca/kosta/CompVis_Notes/mean_shift.pdf` (2005)
6. Evans, C.: Notes on the opensurf library. University of Bristol, Tech. Rep. CSTR-09-001 (January 2009)
7. Gabryel, M., Korytkowski, M., Scherer, R., Rutkowski, L.: Object detection by simple fuzzy classifiers generated by boosting. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2013, Part I. LNCS, vol. 7894, pp. 540–547. Springer, Heidelberg (2013)
8. Gabryel, M., Nowicki, R.K., Woźniak, M., Kempa, W.M.: Genetic cost optimization of the $GI/M/1/N$ finite-buffer queue with a single vacation policy. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2013, Part II. LNCS, vol. 7895, pp. 12–23. Springer, Heidelberg (2013)
9. Gabryel, M., Woźniak, M., Nowicki, R.K.: Creating learning sets for control systems using an evolutionary method. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) SIDE 2012 and EC 2012. LNCS, vol. 7269, pp. 206–213. Springer, Heidelberg (2012)
10. Górecki, P., Sopyła, K., Drozda, P.: Ranking by K-means voting algorithm for similar image retrieval. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2012, Part I. LNCS, vol. 7267, pp. 509–517. Springer, Heidelberg (2012)

11. Hare, J.S., Samangooei, S., Lewis, P.H.: Efficient clustering and quantisation of sift features: Exploiting characteristics of the sift descriptor and interest region detectors under image inversion. In: Proceedings of the 1st ACM International Conference on Multimedia Retrieval, p. 2. ACM (2011)
12. Lew, M.S., Sebe, N., Djeraba, C., Jain, R.: Content-based multimedia information retrieval: State of the art and challenges. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP) 2(1), 1–19 (2006)
13. Lowe, D.G.: Object recognition from local scale-invariant features. In: The Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 2, pp. 1150–1157. IEEE (1999)
14. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
15. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2161–2168. IEEE (2006)
16. O'Hara, S., Draper, B.A.: Introduction to the bag of features paradigm for image classification and retrieval. arXiv preprint arXiv:1101.3354 (2011)
17. Przybył, A., Cpałka, K.: A new method to construct of interpretable models of dynamic systems. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2012, Part II. LNCS, vol. 7268, pp. 697–705. Springer, Heidelberg (2012)
18. Ramos, J.: Using tf-idf to determine word relevance in document queries. In: Proceedings of the First Instructional Conference on Machine Learning (2003)
19. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing & Management 24(5), 513–523 (1988)
20. Sivic, J., Zisserman, A.: Video google: A text retrieval approach to object matching in videos. In: Proceedings of the Ninth IEEE International Conference on Computer Vision, pp. 1470–1477. IEEE (2003)
21. Veltkamp, R.C., Tanase, M.: Content-based image retrieval systems: A survey. Rapport no UU-CS-2000-34 (2000)
22. Wang, J.Z., Li, J., Wiederhold, G.: Simplicity: Semantics-sensitive integrated matching for picture libraries. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(9), 947–963 (2001)