Novel method for Image Classification using Graph approach

Mrs Snehal Amrutkar
Department of Computer Engineering, University of Pune,
Late G.N.Sapkal College of Engineering, Anjaneri,
Nashik, INDIA

Prof. J. V. Shinde
Department of Computer Engineering, University of Pune,
Late G.N.Sapkal College of Engineering, Anjaneri,
Nashik, INDIA

ABSTRACT

Graphs are important in modeling complicated structures, such as natural scenes, animals, and flowers, images in our daily life, social networks and the web. Data representations in graphical forms are an important research topic due to the suitability of this kind of data structure to model entities and the complex relations among them. In order to classify images we are proposing to combine a powerful graph-based image representation and frequent approximate sub-graph (FAS) mining algorithms. From the many approaches for image classification, graph based approach is gaining popularity due to its ability in reflecting global image properties. In this paper, VEAM (Vertex and Edge Approximate graph Miner) algorithm is used for mining frequent connected subgraphs over undirected and labeled graph collections. Some Slight variations of the data, are allowed in this algorithm, by keeping the topology of the graphs. And, Graph-based image representation is by using dynamic region merging technique can tolerate some differences for grouping meaningful regions in an image.

Keyword: *Data Mining, Dynamic region merging, Graph-based image representation, Feature selection, Frequent approximate patterns and Image classification.*

1. INTRODUCTION

In many research fields, graphs have been largely used to model data due to their suitability for applications and their representation expressiveness where some kind of entities and their relationships must be encoded within some data structure. In this paper, our intention is to explore and combine two research fields where graphs are involved, in order to exploit both their advantages.

Our main goal is to classify images using a graph-based representation. The first step for classification of images is to extract low-level features that will encode relevant information for the task, but it has been shown that low-level information by itself cannot provide the high-level perception cues that present in human minds to describe objects or images in general. Within the range of low-level features developed so far, graphs are one of the representations that can provide some kind of high-level information, making them a desirable representation choice for researchers to find new solutions.

And the other research field is Data Mining. Several authors have developed graph-based techniques to convert large volumes of data into useful information. An example of such techniques is frequent subgraph discovery. An important problem in graph mining tasks is classifying information, such as image, text etc.

Within the broad area of information discovery and retrieval, there is an ever increasing requirement for an effective means of indexing, searching and retrieving the information.

To improve the accuracy and effectiveness of images in given databases, we are using the DRM technique for graph generation, starts from an initially over-segmented image. For simplicity, we use the watershed algorithm to obtain the initial segmentation. The neighbouring regions with coherent colours are merged into one. Graph generated by DRM technique is very small in size as compared to quad tree in existing system. So lots of space get saved because of less number of patterns generated from graph.

2. LITERATURE SURVEY

In this section, we will discuss on graph-based image representation techniques. Next, we will see review of previous approximate graph mining methods.

1.1 Brief overview of graph-based image representation techniques:

Images are represented as graphs which is very popular since they are powerful tools to encode different types of information, and may provide a robust and rich representation for many applications. The main issue is how to exploit this information in the graph-based image classification scope. A popular graph-based image representation is the Region Adjacency Graph (RAG) [1] where each vertex denotes a region in the image and an edge exists between two vertices if the underlying regions are adjacent as shown in figure 1.

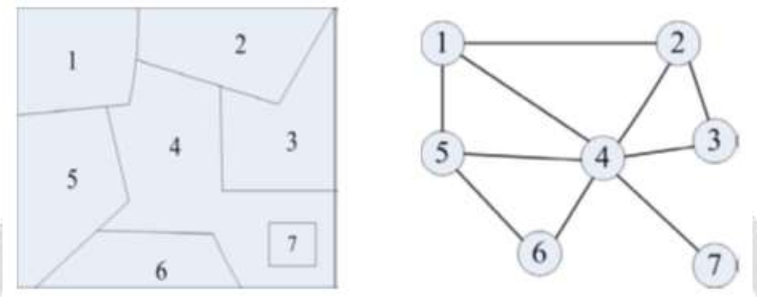


Fig 1. An example of region partition and the corresponding region adjacency graph (RAG)

The results of graph generation using Quad-trees [2] and DRM [3] are compared. Due to modeling images as graphs, the task of image classification get to be one of classifying graphs.

In figure 4, an example of quad-tree with 4 as depth limit of divisions over an image is shown. The color property taken as an attribute of the nodes in the example. After obtaining quad-tree of an image, we generate a graph to represent the image. Then, a graph is used to represent each image in the collection is created from each respective tree. In Figure 6, we have shown the graph generated from the quad-tree of Figure 4.

The main disadvantage of quad trees is that it is impossible to compare two images that differ in rotation or translation. Because of this quad tree representation of such images will be totally different. The algorithms which are available for rotation of an image are restricted to rotations of 90 degrees (or multiples of there). There is no other rotation or translation facility is available. The rotation will completely change the quad tree of the image as shown in figure 7. Once the quad tree has changed it becomes a difficult task to compare it to an earlier version of quad tree. Therefore quad trees have poor shape analysis and pattern recognition. Graph generated using quad trees contains large number of vertices which will have very large graph size.

A simple perspective to keep the structural and topological information of an image is to use image representation methods; for instance, dynamic region merging [3], etc. By modeling images as graphs, the work of image classification becomes one of classifying graphs.

The DRM algorithm starts from an initially over-segmented image. For simplicity, we use the watershed algorithm to obtain the initial segmentation. It is clear that neighboring regions with coherent colors are merged into one, whereas the boundaries are well located on the reasonable places as shown in figure 2. Some of the large regions have significant variations inside Figure 2, however, with relatively slow changes of colors along the boundaries. This indicates that the DRM algorithm can tolerate some variations for grouping meaningful regions in an image. In case of DRM technique, graph generated is very small in size as compared to quad tree which is shown in figure 3

Advantage of DRM technique are they have better computational efficiency, more convenient to control the performance, can tolerate variation of grouping, can be easily automated, can be applied to discrete regions and the speed is very fast.

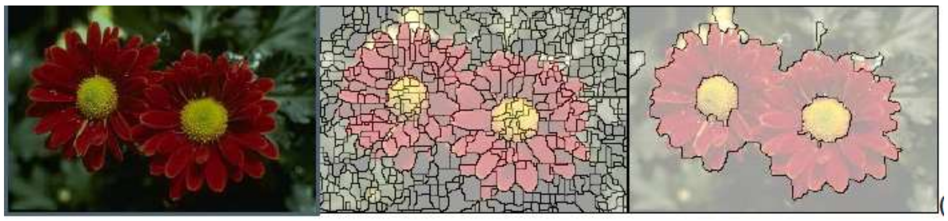


Fig 2. Segmentation results by DRM algorithm. From left to right, the first column shows the original image. The second column shows the over-segmentation produced by watershed algorithm. The third column shows segmentation results using DRM algorithm

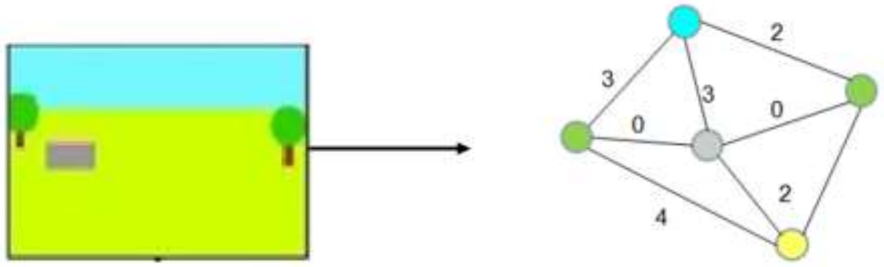


Fig 3: Graph generated by using DRM technique

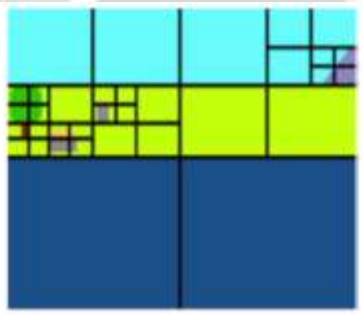
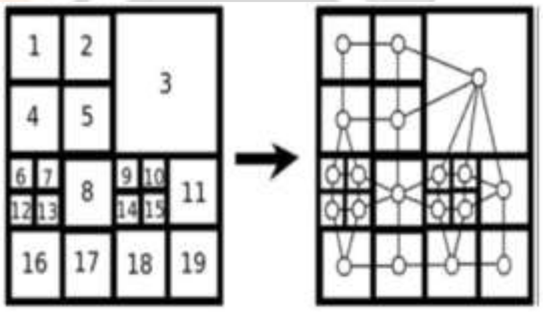



Fig. 4. Example image with the quadrants of quad-tree identified Fig. 5. Example of quadrant divisions of an image.

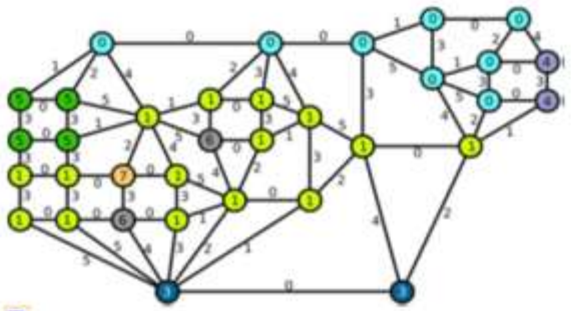


Fig. 6. Graph generated from the quad-tree shown in Fig. 4.

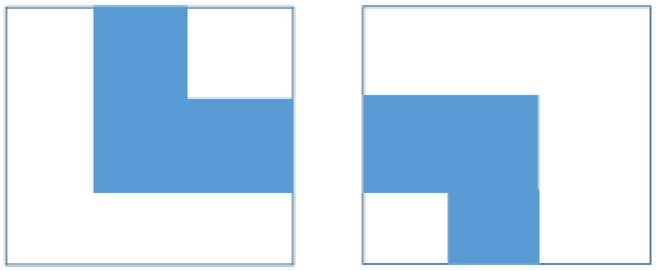


Fig 7: First image and rotated image, different quad trees

1.2 Summary of approximate graph mining:

Different methods have been developed to use graphs for classification tasks.

In the last years, some approximate graph mining algorithms have been researched where several similarity functions are used. For example, the algorithms SUBDUE [5] and RNGV [6] are based on graph edit distance; Monkey [4] is based on β-edge sub- isomorphism; UGRAP [7] and MUSE [8] are based on sub- isomorphism on uncertain graph collections; GREW [9] is based on sub-isomorphism employing ideas of edge contraction and graph rewriting; CSMiner [10] uses node/edge disjoint sub-home-omorfismo; gApprox [11], APGM [12] and VEAM [13] are based on substitution probabilities.

APGM and VEAM algorithms defend the idea that a vertex label or an edge label cannot always be replaced by any other. Therefore, these algorithms states that which vertices, edges or labels can replace others using substitution matrices to perform frequent subgraph mining. Although, only APGM and VEAM perform frequent approximate subgraph mining on graph collections and we are interested in this kind of mining. APGM only deals with the differences between the vertex labels, while VEAM performs the mining process using both the vertex and edge label sets.

In this paper, the last VEAM algorithm is applied in order to create an image representation that will be used for classification purposes. This is because of the need of an algorithm that allows some variations in the data using substitution probabilities and at the same time keeping topology of the graphs.

1.3 Basic Concepts:

This work is focused on simple undirected labeled graphs. Before representing their formal definition, we define the domain of labels.

Let L_V and L_E be label sets, where L_V is a set of vertex labels and L_E is a set of edge labels, the domain of all possible labels is denoted by $L = L_V \cup L_E$.

A labeled graph in L is a 4-tuple, G = (V,E, I, J), where V is a set whose elements are called vertices, $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$ is a set whose elements are called edges (edge {u, v} connecting the vertex u with the vertex v), $I : V \rightarrow L_V$ is a labeling function for assigning labels to vertices and $J : E \rightarrow L_E$ is a labeling function for assigning labels to edges.

Let $G_1 = (V_1, E_1, I_1, J_1)$ and $G_2 = (V_2, E_2, I_2, I_2)$ be two graphs, we say that G_1 is a subgraph of G_2 if $V_1 \subseteq V_2$, $E_1 \subseteq E_2$, $\forall u \in V_1$, $I_1(u) = I_2(u)$, and $\forall e \in E_1$, $J_1(e) = J_2(e)$. In this case, we denoted by notation $G_1 \subseteq G_2$.

Given G_1 and G_2, then we say that f is an isomorphism between these graphs if $f : V_1 \rightarrow V_2$ is a bijective function, in this $\forall u \in V_1$; f $\{u\} \in V_2 \wedge I_1\{u\} = I_2(f(u))$ and $\forall\{u, v\} \in E_1$;$\{f(u), f(v)\} \in E_2 \wedge J_1(\{u, v\} = J_2(\{f(u), f(v)\})$. When there is an isomorphism between G_1 and G_2, then we say that G_1 and G_2 are isomorphic.

Let Ω be set of all possible labeled graphs in L, the similarity between two elements $G_1; G_2 \in \Omega$ is defined as a function sim: $\Omega \times \Omega$ [0, 1]. We say that the elements are very different if sim $(G_1, G_2) = 0$, the higher the value of $sim(G_1, G_2)$,then more similar the elements are and if $sim(G_1, G_2) = 1$ then there is an isomorphism between these elements.

Let $D = \{G_1, \ldots, G_D\}$ be a graph collection and let G be a labeled graph in L, support value of G in D is obtained by the following equation:

$$\mathrm{Sup}\{G, D\} = \sum Gi \in D\ \mathrm{sim}(G, Gi) = |D|$$

If Sup(G,D) ≥ δ, then graph G occurs approximately frequent in the collection D, saying that G is a frequent approximate subgraph in D. The value of support threshold δ is in (0, 1] assuming that the similarity is normalized to

1. The frequent subgraph mining does in finding all the connected frequent approximate subgraphs in a collection of graphs D, using a similarity function sim and a support threshold δ.

3. SYSTEM FOR IMAGE CLASSIFICATION

In this section, we present a framework for image classification tasks. Image classification details are introduced with the proposed framework.

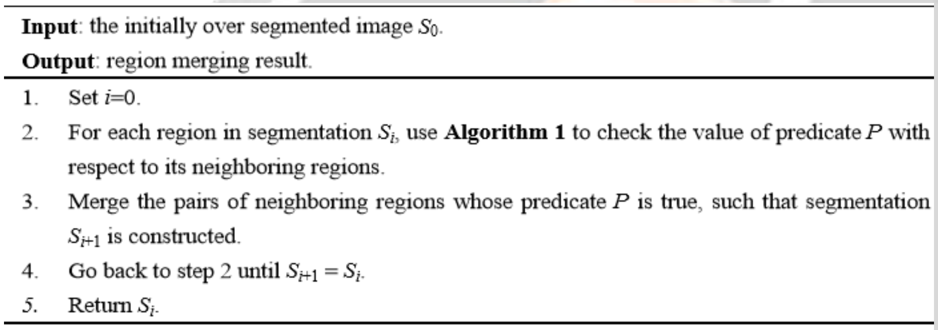
3.1 Proposed Framework:

First of all, we obtain the graph-based image representation of a given set of real images using dynamic region merging, which gives us a graph collection to work with. Afterwards, the algorithms for frequent subgraph mining are implemented with the aim of obtaining all frequent subgraphs over a graph collection. Then, these subgraphs (i.e. patterns) are used as features and the feature vectors of the original images are made. These feature vectors are compared with the feature vector of input query image. Lastly, we employ a classifier generator i.e. SVM using these vectors as data to produce an image classification result. The complete flowchart of our image classification system is shown in Figure 8.

3.2 Watershed algorithm with DRM technique:

DRM algorithm is started from a set of segmented regions. This is because a small region can provide more stable statistical information than a single pixel, and using regions for merging can improve a lot the computational efficiency. The algorithm is as follows:

Algorithm: Segmentation by dynamic region merging

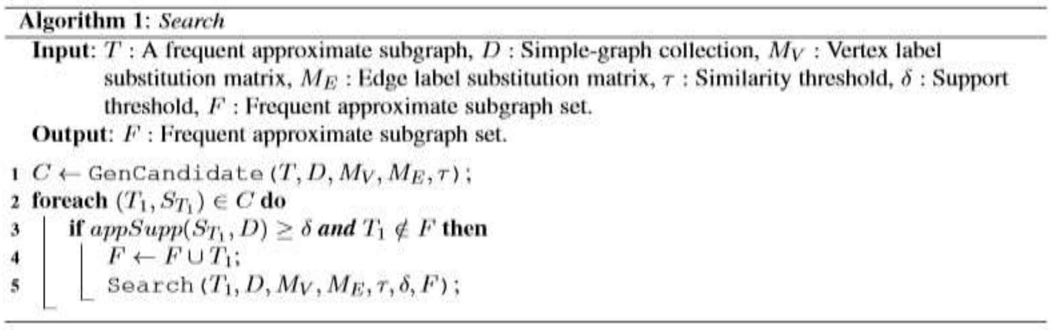


```
Input: the initially over segmented image $S_0$.
Output: region merging result.
1. Set $i$=0.
2. For each region in segmentation $S_i$, use Algorithm 1 to check the value of predicate $P$ with respect to its neighboring regions.
3. Merge the pairs of neighboring regions whose predicate $P$ is true, such that segmentation $S_{i+1}$ is constructed.
4. Go back to step 2 until $S_{i+1} = S_i$.
5. Return $S_i$.
```

3.3 Frequent Approximate Patterns:

The VEAM[14] algorithm starts computing all frequent approximate subgraphs corresponding to single vertex graphs and single-edge graphs. Then, following a Depth-First Search (DFS) approach, each frequent single-edge is extended by adding another single-edge at a time through a recursive function, called "Search". When all frequent approximate single-edges have been extended, then the set of all frequent approximate subgraphs in the multi-graph collection D is returned.

The Search function, shown in Algorithm 1, recursively performs the candidate extension of all frequent graphs. A candidate set for the given frequent graph is obtained from "GenCandidate" function invoked in line 2 of the pseudo-code. Then, the frequency of each candidate is verified, keeping only the candidates that satisfy the support constraint and which are not identified in previous steps; these candidates are stored as output patterns and are extended performing a recursive call to Search function.



```
Algorithm 1: Search
  Input: $T$ : A frequent approximate subgraph, $D$ : Simple-graph collection, $M_V$ : Vertex label substitution matrix, $M_E$ : Edge label substitution matrix, $\tau$ : Similarity threshold, $\delta$ : Support threshold, $F$ : Frequent approximate subgraph set.
  Output: $F$ : Frequent approximate subgraph set.
1 $C \leftarrow$ GenCandidate$(T, D, M_V, M_E, \tau)$;
2 foreach $(T_1, S_{T_1}) \in C$ do
3     if $appSupp(S_{T_1}, D) \geq \delta$ and $T_1 \notin F$ then
4         $F \leftarrow F \cup T_1$;
5         Search$(T_1, D, M_V, M_E, \tau, \delta, F)$;
```

The aim of the GenCandidate function, shown in Algorithm 2, is to compute all candidate extensions of a given frequent graph T. In this candidate generation phase, all child of T and its occurrences in the simple graph collection D are searched. Later, the canonical forms based on adjacency matrices (CAM) for these subgraphs, which satisfy the similarity constraint, are computed by the “ComputeCAM” function invoked in line 3 in Algorithm 2. Finally, each subgraph G with its corresponding CAM code and its similarity value is stored as an output candidate in C.

```
Algorithm 2: $GenCandidate$
  Input: $T$ : A frequent approximate subgraph, $D$ : Simple-graph collection, $M_V$ : Vertex label
         substitution matrix, $M_E$ : Edge label substitution matrix, $\tau$ : Similarity threshold.
  Output: $C$ : A set of candidate graph.
1 $C \leftarrow$ An empty hash table;
2 $O \leftarrow \{(G, T_e) | G$ is a child of $T$ and $T_e$ is an occurrence of $G$ in $G_k \in D\}$;
3 foreach $(G, T_e) \in O$ do
4     $CAM_G =$ ComputeCAM($G$,$e'$) ; // see Algorithm 3
5     $sim(G, T_e)$ is inserted into $C[CAM_G]$;
```

The similarity function used by VEAM for the graph matching process, but in this case, this function allows performing the approximate matching considering both, vertex and edge labels. The canonical form used by VEAM is known as Canonical Adjacency Matrix (CAM) as it is based on the graph adjacency matrix. Notice that, sin ce the adjacency matrix of an undirected simple-graph is symmetric and this is the kind of graphs treated, only the upper or lower triangular adjacency matrices are needed for computing the CAM code of a graph. In the VEAM algorithm, the CAM code of a graph G is obtained following the ideas, where the adjacency matrix representation of a simple-graph is converted into a sequence of symbols. In this way, the label vertex set, as well as the degree-based partition order are used as vertex invariant. Using this vertex invariant, vertices of a graph can be partitioned into equivalence classes, where vertices of the same class have equivalent vertex invariant values. This criteria allows performing permutations of the vertices into the same cluster (partition) instead to performing permutations into the whole set of vertices. This is because two isomorphic graphs will lead to the same partitioning of the vertices and therefore the same canonical code is computed from them.

In Algorithm 3, the process of computing the CAM code of a given simple-graph is shown. First, the partition set P for the vertices of G is generated according to the vertex labels and the vertex degrees as partitioning criteria. Next, P is sorted in descending way, according to the vertex labels in first place and the degrees as secondary criterion (see lines 2). Later, in each partition an internal sorting taking into account a maximum lexicographical order obtained from its permutations (see lines 3−14) is performed. Finally, the CAM code CAM_G of G is obtained, which is computed according to the final ordering of P.

```
Algorithm 3: ComputeCAM
  Input: $G$ : A candidate subgraph, $e'$ : The last edge added into $G$.
  Output: $CAM_G$ : The CAM code for the candidate subgraph $G$.
1  $P \leftarrow$ Partition of $V_G$ such that: $v_i, v_j \in P_k$, have the same label and degree, where $i \neq j$ and $P_k \in P$;
2  $P$ is lexicographically sorted $[(l_{i-1}, d_{i-1}) \succ (l_i, d_i)]$, being $l_k$ and $d_k$ are the label and the degree of
   $P_k$;
3  foreach $P_k = \{p_1, \ldots, p_{|P_k|}\} \in P$ do
4      $l = |P_k|$;
5      $l_k$ is the label for the clusters $P_k$;
6      while $l \neq 1$ do
7          $newl = 1$;
8          foreach $i = \{2, \ldots, |P_k|\}$ do
9              $X = (x_{(1,1)}, \ldots, x_{(|P_1|,1)}, \ldots, x_{(1,k)}, \ldots, x_{(i-1,k)})$ is computed based on Definition 3.2,
               being $x_{(j,w)}$ the label of the edge between the vertex in $p_{i-1}$ and the vertex in $p_j \in P_w \in P$;
10             $Y = (y_{(1,1)}, \ldots, y_{(|P_1|,1)}, \ldots, y_{(1,k)}, \ldots, y_{(i-2,k)}, y_{(i,k)})$ is computed based on
               Definition 3.2, being $y_{(j,w)}$ the label of the edge between the vertex in $p_i$ and the vertex in
               $p_j \in P_w \in P$;
11             if $X < Y$, following a lexicographical order then
12                 Swap ($p_{i-1}$, $p_i$);
13                 $newl = i$;
14         $l = newl$;
15 The adjacency matrix $M$ of $G$ is built sorting its cells following the lexicographical order achieved in
   $P$;
16 The CAM code is obtained from $M$ and it is stored into $CAM_G$;
```

3.4 Clustering of graphs by K-means:

K-means method is known as the most frequently used and simplest method in unsupervised classification. K-means algorithm is an iterative procedure. The procedure consists of the following steps:

1. Choose K initial clusters $z_1(1), z_2(1), \ldots, z_K(1)$.
2. At the kth iterative step, distribute the samples x among the K clusters using the following relation

$$x \in C_j(k) \; if \; \|x - z_j(k)\| < \|x - z_i(k)\|$$

for all $i = 1,2,\ldots,K, i \neq j$, where $C_j(k)$ denotes the set of samples whose cluster center is $z_j(k)$.

3. Compute the new cluster centers $z_j(k + 1), j = 1,2,\ldots,K$, such that the sum of the squared distance from all points in $C_j(k)$ to the new cluster is minimized. The measure which minimizes this is simply the sample mean of $C_j(k)$. Therefore, the new cluster center is given by

$$z_j(k+1) = \frac{1}{N_j} \sum_{x \in C_j(k)} x, \quad j = 1, 2, \ldots, K$$

where N_j is the number of samples in $C_j(k)$.

4. If $z_j(k + 1)$, $j = 1,2,\ldots,K$, the algorithm has converged and the procedure is terminated. Otherwise go to setp 2.

3.5 SVM Classifier:

In order to evaluate the quality of the patterns identified we use the frequent subgraphs detected on classification tests. For that, the package libSVM is used for image classification through Support Vector Machine (SVM) classifier. Given these frequent patterns, we build feature vectors upon them, then an image is represented as a feature vector $V = (v_1, \ldots, v_x)$, where x is the total number of sub-graphs identified. Thus, we build a matrix where the row number $(1 \leq i \leq G)$ corresponds to the number of graph (images) in the collection, and the number of columns $(1 \leq j \leq s)$ corresponds to the number of frequent subgraphs (features).

4. EXPERIMENTAL RESULTS

4.1 Dataset:

We have considered two well-known databases containing color images of simple objects taken from different viewpoints. The first one is the COIL-100 [15] dataset, which has 100 objects with 72 poses per object. The second dataset is ETH-80 Image Set [16], which has 80 objects from eight categories and each object is represented by 41 different views, giving a total of 3280 images. This database is more challenging than the COIL-100 database because of the viewpoint diversity. Size of all images is no matter. Some example images from both datasets are as shown in Figure 9.Unknown dataset includes randomly selected images of variable sizes from internet of different categories which are present in unknown data set as shown in Figure 10.

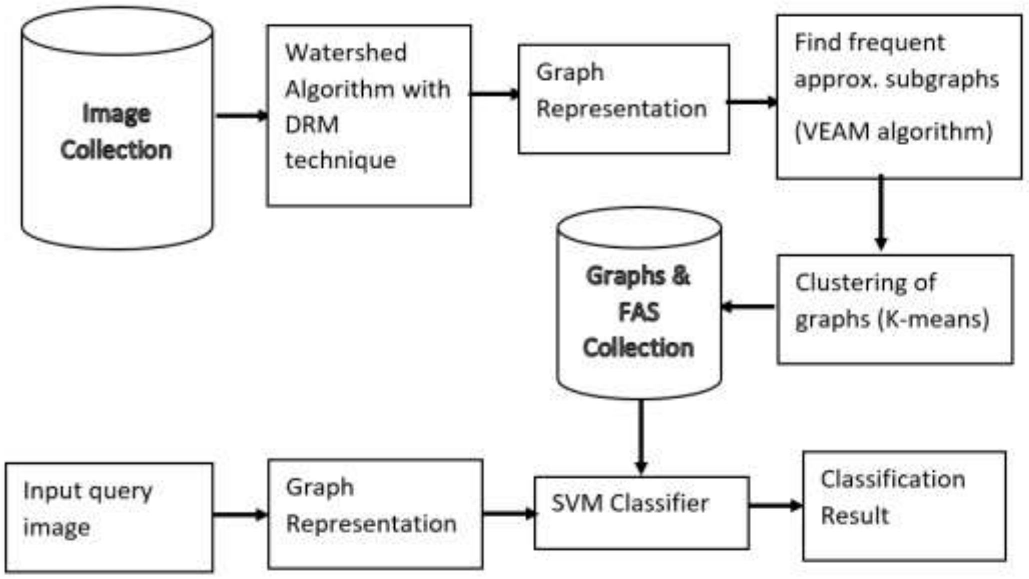


Fig 8: Framework of Graph based Image Classification

Fig 9: Example images from the COIL-100 Image Set database (first six images), and from the ETH-80 Image Set database (last six images)

Fig 10: Example images from unknown dataset

4.2 Graph tables and comparison:

DRM based graph generation method is compared with the quad-tree based graph generation. Comparison of DRM and Quad-tree in graph table is with respect to two parameters i.e. accuracy and precision which as shown in figures 11, figure 12. The values of accuracy and precision are calculated which are better than quad-tree graph generation.

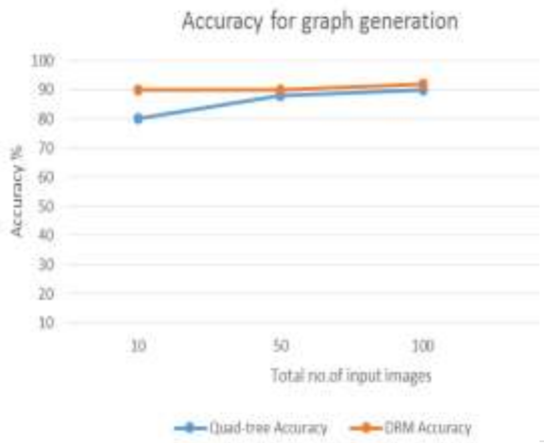
In table 1, results on different datasets are shown. The First 3, i.e. Testing (i.e. unknown) dataset I, II and III are tested for total number of images 10, 50 and 100 respectively. And last Training (i.e. COIL-100 and ETH-80) dataset is tested for about 7203 images.

For example, consider Testing dataset for 10 images and query image is purple color hibiscus flower as shown in figure 10, number of clusters is 2 as input. Then Accuracy and Precision values for it is 83% and 88% respectively in case of Quad-tree based classification. In case of DRM based classification, Accuracy and Precision values are 90% and 92% respectively.

Graph generated by DRM technique is very small in size as compared to quad tree in existing system. So lots of space and time get saved because of less number of patterns generated from graphs.

Table 1: Accuracy and precision values achieved by Quad tree and DRM algorithms in different datasets

No.	Description	No. of images	Quad-tree based classification		DRM based classification	
			Accuracy	Precision	Accuracy	Precision
1	Testing dataset I	10	80	85	90	95
2	Testing dataset II	50	88	93	90	98
3	Testing dataset III	100	90	95	92	99
4	Training dataset	7203	90	95	93	99



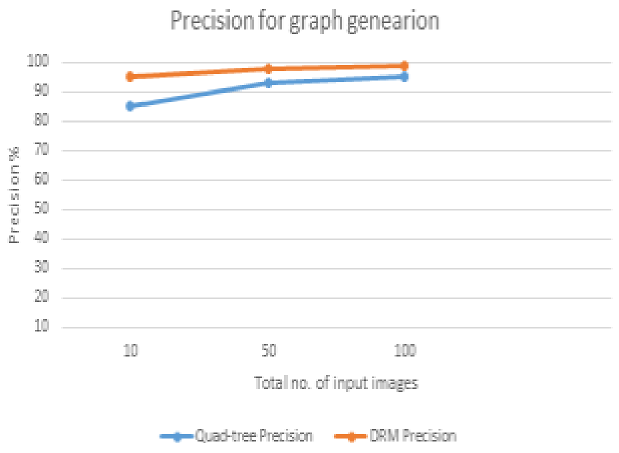


Fig 11: Accuracy for graph generation **Fig 12:** Precision for graph generation

5 CONCLUSION

The goal of this project is graph based image classification in which DRM technique is used for graph generation which improves the performance of the system. Graph generated by DRM technique is very small in size as compared to quad tree in existing system. So lots of space get saved and also time get reduced because of less number of patterns generated from graphs.

By using approximation in VEAM algorithm, more interesting patterns can be found for many applications, for example, when processing graph databases that have distortions (in terms of different topological, geometric or semantic variations) of similar structures in several objects. In several domains of science, distortion in data is one of the challenges for developing classifiers based on frequent patterns.

As future work, the current image segmentation technique can be fully automated in which over-segmentation of the data is done in automated fas'hion. The speed of combining the regions can be increased by using nearest neighbor graph. We can develop interactive DRM algorithm. We can increase the system efficiency.

6 REFERENCES

[1] L. Brun, W. Kropatsch, Contains and inside relationships within combinatorial pyramids, Pattern Recognition 39 (4) (2006) 108–128.

[2] Bo Peng, Lei Zhang1, *Member, IEEE* and David Zhang, *Fellow Member, IEEE,* Automatic Image Segmentation by Dynamic Region Merging, The Hong Kong Polytechnic University, Hong Kong,2007

[3] R.A. Finkel, J.L. Bentley, Quad trees: a data structure for retrieval on composite keys, Acta Informatica 4 (1974) 1–9.

[4] S.Zhang, J.Yang, RAM: randomized approximate graph mining, in: Proceedings of the 20th InternationalConference on Scientific and Statistical Database Management, Hong Kong, China, 2008, pp.187–203

[5] L.B. Holder, D.J.Cook, H.Bunke, Fuzzy substructure discovery, in: Proceedings of the Ninth International Workshop on Machine Learning, San Francisco, CA, USA, 1992, pp.218–223.

[6] Y.Song, S.Chen, Itemsets based graph mining algorithm and application in genetic regulatory networks, in: Proce”?edings of the IEEE International Conference on Granular Computing, Atlanta, GA, USA, 2006, pp.337–340.

[7] O. Papapetrou, E.Ioanno, D.Skoutas, Efficient discovery of frequent subgraph patterns in uncertain graph databases, in: Proceedings of the 14th International Conference on Extending Database Technology, vol.12, New York, USA, 2011, pp.355–366.

[8] Z. Zou, J.Li, H.Gao, S.Zhang, Mining frequent subgraph patterns from uncertain graph data, IEEE Transactions on Knowledge and Data Engineering 22 (9)(2010)1203–1218.

[9] M. Kuramochi, G.Karypis, GREW—a scalable frequent subgraph discovery algorithm, in: Proceedings of the 4th IEEE International Conference on Data Mining, 2004,pp.439–442.

[10] Y.Xiao, W.Wu, W.Wang, Z.He, Efficient algorithms for node disjoint subgraph homeomorphism determination, in: Proceedings of the 13th International Conference on Database Systems for Advanced Applications, New Delhi, India, 2008,pp.452–460.

[11] C. Chen, X.Yan, F.Zhu, J.Han, gApprox: mining frequent approximate patterns from a massive network, in: IEEE International Conference on Data Mining, 2007,pp.445–450.

[12] Y.Jia, J.Zhang, J.Huan, An efficient graph-mining method for complicated and noisy data with real-world applications, Knowledge Information Systems28 (2) (2011)423–447.

[13] N. Acosta-Mendoza, A.Gago-Alonso, J.E.Medina-Pagola, Frequent approximate subgraphs as features for graph-based image classification, Knowledge- Based Systems 27(March)(2012)381–392.

[14] N. Acosta-Mendoza, A.Gago-Alonso, J.E.Medina-Pagola, Representative Frequent Approximate Subgraph Mining in Multi-Graph Collections,Feb 2015.

[15] S. Nene, S. Nayar, H. Murase, Columbia Object Image Library (COIL-100), Technical Report, Department of Computer Science, Columbia University CUCS-006-96, 1996.

[16] B. Leibe, B. Schiele, Analyzing appearance and contour based methods for object categorization, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03), 2003, pp. 409–415.