International Journal on Semantic Web and Information Systems
Volume 18 • Issue 1

A Model of Semantic-Based Image Retrieval Using C-Tree and Neighbor Graph

Nguyen Thi Uyen Nhi, University of Science, Hue University, Vietnam & University of Economics, The University of Danang, Vietnam

Thanh Manh Le, University of Science, Hue University, Vietnam

Thanh The Van, HCMC University of Food Industry, Vietnam

ABSTRACT

The problems of image mining and semantic image retrieval play an important role in many areas of life. In this paper, a semantic-based image retrieval system is proposed that relies on the combination of C-Tree, which was built in the authors' previous work, and a neighbor graph (called Graph-CTree) to improve accuracy. The k-Nearest Neighbor (k-NN) algorithm is used to classify a set of similar images that are retrieved on Graph-CTree to create a set of visual words. An ontology framework for images is created semi-automatically. SPARQL query is automatically generated from visual words and retrieve on ontology for semantics image. The experiment was performed on image datasets, such as COREL, WANG, ImageCLEF, and Stanford Dogs, with precision values of 0.888473, 0.766473, 0.839814, and 0.826416, respectively. These results are compared with related works on the same image dataset, showing the effectiveness of the methods proposed here.

KEYWORDS

C-Tree, Graph-CTree, Image Retrieval, k-NN, Ontology, SBIR_grCT, Semantic-Based Image Retrieval, Similar Images, SPARQL

INTRODUCTION

The problems of image mining, image retrieval, and image semantics are becoming more popular among researchers. Identifying the desired image from a large and diverse image dataset is a challenging task. Therefore, it is essential to develop high-precision image retrieval systems for large datasets. However, the low-level content of images, including color, shape, and texture, cannot define the user's high-level semantics (Allani et al., 2017; Cevikalp et al., 2018). Therefore, it is essential to have a structure for mapping low-level features to high-level semantics based on ontology (Filali et al., 2017; Sarwara et al., 2013). Thus, high-level semantics of images can classify and retrieve high-level semantics of images based on SPARQL (Hogan, 2020).

Our ontology-based model is proposed to support two main features for semantic retrieval in image datasets: (1) searching for a set of images that are similar to a given image; and (2) mapping low-level features onto the high-level semantics of the image based on ontology. The result of using SPARQL query on ontology is the set of descriptive semantics of the dataset and the set of similar images according to the semantic of the input image. A balanced clustering tree structure, called

DOI: 10.4018/IJSWIS.295551

C-Tree, was published in Nhi et al. (2020) to automatically archive indexed images from low-level features of the image. C-Tree has many advantages, such as a multi-branch tree and clustered feature vectors, so it can store large amounts of data and retrieve images rapidly. However, image retrieval on C-Tree is conducted by finding the branch with the closest similarity measure to the input image. Therefore, it may not be possible to find all feature vectors that are similar to the input image because, when splitting the node, it may have split through another cluster or even another branch.

This paper proposes a combination model of C-Tree and a neighbor graph, Graph-CTree, to limit the omission of similar elements that have split into another branch and to improve the accuracy of image retrieval. For semantic-based image retrieval, a semi-automatic ontology framework is built upon the Resource Description Framework (RDF; Furche et al., 2006; Haase et al., 2004) for a multi-object ImageCLEF dataset. An input image is queried on Graph-CTree to find a similar set of images. From there, the k-Nearest Neighbor (k-NN) algorithm is performed on this set of images to classify and create a set of visual words. SPARQL query is generated based on these visual words and is queried on the ontology. The experiment was implemented on the COREL, WANG, ImageCLEF (IAPR TC-12 brenchmark), and Stanford Dogs datasets and compared with the previous results of related works to evaluate the efficiency of the proposals offered here.

The contributions of the paper include: (1) proposing a model combining C-Tree and Graph-CTree to improve the efficiency of image retrieval; (2) proposing a model for building a semi-automatic ontology framework; and (3) proposing a semantic-based image retrieval model combining machine learning method (Graph-CTree) and ontology.

RELATED WORK

There are many techniques of semantic-based image retrieval that have been implemented in many digital systems. The state-of-the-art techniques for reducing the ‘semantic gap’ can be broadly categorized into two main categories: (1) using machine learning methods to associate low-level features with visual concepts to describe image objects (Jui et al., 2017; Barz et al.,2020); (2) using ontology to define high-level semantics (Allani et al., 2017; Gonçalves et al., 2018; Bouchakwa et al.,2020). Many systems use one or a combination of different techniques to perform high-level semantic-based image retrieval.

Jiu Mingyuan and Hichem Sahbi (2017) used deep multi-layer networks based on nonlinear activation functions for image annotation (Jiu & Sahbi, 2017). The Support Vector Machine (SVM) technique was applied to layering images at the output layer to extract a semantic level according to visual information for similar pocket-based images from BoW (Bag-of-Words). Extensive experiments and analysis of the ImageCLEF and COREL5k datasets validated the effectiveness of the proposed method. The precision of image retrieval for each dataset was 0.597 and 0.8867, respectively.

Hakan Cevikalp et al. (2018) proposed a method for image retrieval using a class binary hierarchical tree and trained with a low-level feature-based transductive SVM (TSVM) technique to separate labeled and unlabeled data samples at each node of the binary hierarchy. The method was tested on the ImageCLEF, CIFAR100, and NUS-WIDE datasets with precision values of 0.4678, 0.7235, and 0.7621, respectively.

Barz Bjorn and Joachim Denzler (2020) proposed a method to integrate semantic relationships between classes. This approach has been mentioned as a classification into deep learning by deep neural networks. The hierarchy-based semantics preserve the semantic similarity of the classes in the common space of the images and classes. The proposed method has been implemented on the CIFAR-100, CUB-200, and Oxford Flowers-102 datasets, showing significant improvement in the semantic consistency of image retrieval results. At the same time, the precision of image retrieval for each dataset was 0.775, 0.68, and 0.711, respectively.

Jalila Filali et al. (2017) proposed to build ontology based on visual annotations of the ImageCLEF dataset. The image retrieval process is accomplished by integrating both low-level features and visual semantic similarity. The ontology is diversified by concepts and relationships extracted from BabelNet lexical resources. However, this proposal system has not combined low-level features with the semantic concept and semantic query of the image. The experiment has not been evaluated, so the effectiveness of the system in terms of accuracy has not been verified.

In 2017, Allani Olfa et al. (2017) proposed a pattern-based image retrieval system, SemVisIR, which combined semantic and visual features. They organized the image dataset in a graph of patterns that are automatically built for the different domains by clustering algorithms. SemVisIR modeled the visual aspects of images using graphs of regions and by assigning them to automatically built ontology modules for each domain. Their system was implemented and evaluated on ImageCLEF. The performance of this method was not strong compared to other methods (0.3513); the semantic image search process is based on automatic ontology, so the reliability is low and the precision is average.

Gonçalves Filipe Marcel et al. (2018) proposed a semantic approach that combines content-based image retrieval (CBIR), unsupervised learning, and ontology techniques. For the semantic model, a representative domain ontology was constructed for the properties and structures of plants. The proposed graph-based approach combined semantic information and low-level visual features and was evaluated on the Oxford Flowers dataset with 102 classifiers to demonstrate its effectiveness. The precision of image retrieval with ontology (0.8539) was higher than that of CBIR and unsupervised learning (0.8020). The efficiency of this method shows that image retrieval with the combination of low-level features and high-level semantics on ontology produces outstanding performance. However, this ontology is built on a small domain, has not yet implemented the definition of domain concepts, and has not clarified the relationship between value domains.

Bouchakwa Mariam et al. (2020) propose semantic improvement based on image tags, different semantic meanings are inferred from users' ambiguous queries based on ontology, then the original query is formatted by implementing a set of semantic rules on the ontology. The images on an image set are clustered by semantic content and filtered, re-ranking the image cluster retrieval results to sort by relevance.

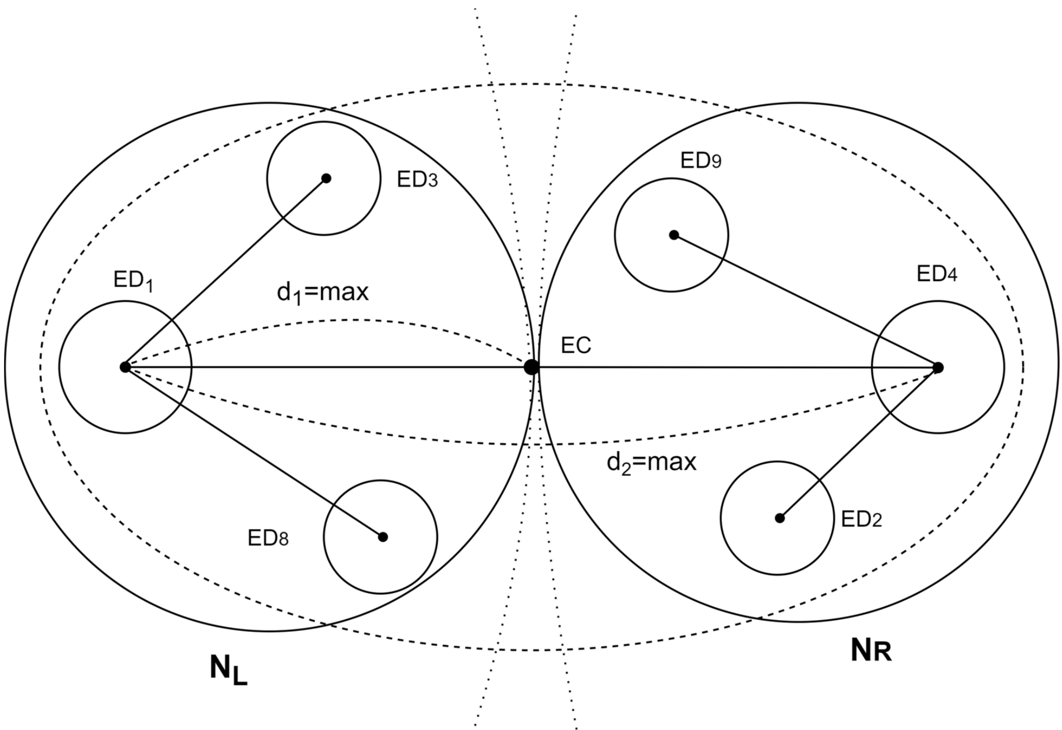
These recent approaches have focused on methods for mapping low-level features to semantic concepts by using supervised or unsupervised machine learning techniques; building data models such as graphs, trees, or deep learning networks to store low-level contents of images; building ontology to define the high-level concepts; and so on. Therefore, semantic-based image retrieval is a topic that has received much attention in recent scholarship.

MODEL COMBINING C-TREE AND THE NEIGHBOR GRAPH

Description of C-Tree

C-Tree (Nhi et al., 2020) consists of a root, a set of nodes I, and a set of leaves L. Nodes are connected through the path of the parent–child relationship. The leaves L, which are nodes without child nodes, contain element data ED so that $L = \{ED_1, ED_2, ..., ED_n \mid 1 \le n \le M\}$, in which M is the maximum number of elements in a leaf. The element data ED contain the following elements: the feature vector of an image f, the identifier of an image ID, the file containing the image caption $file$, and cla, which are classes of the image. Now, $ED = \left\langle f, ID, file, cla \right\rangle$. The nodes I have at least two child nodes, which contain center element EC so that $I = \{EC_1, EC_2, ..., EC_m \mid 1 \le m \le N\}$ in which N is the maximum number of elements in the inner node. Each element of a node links to its adjacent child node via its $link$ and each leaf has the same height. The center element EC contains the following components: f_c, or the center of feature vectors at a child node that has the path linking the $link$ to the EC, and $isNextLeaf$, which is the value used to check whether the next sub-cluster is a leaf or not. At this stage, the centroid element $EC = \left\langle f_c, isNextLeaf, link \right\rangle$. When a leaf or a node in C-Tree is full, the number of elements exceeds the threshold M, N, respectively, and the node performs splitting based on the K-Means algorithm. Figure 1 describes the process of splitting a node in C-Tree.

Figure 1. Splitting a node on the C-Tree

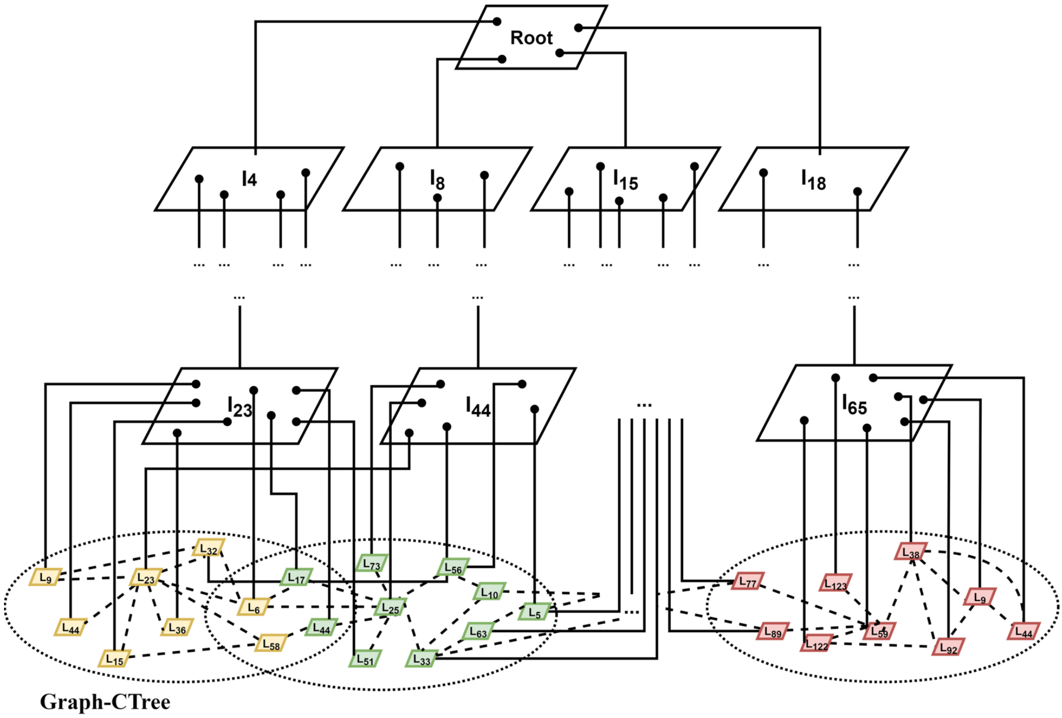


The node-splitting operation in C-Tree is executed as follows:

- Choose $k = 2$ and take the two elements furthest from each other according to the Euclidean measure. Define the node center as the mean value of the feature vectors of the node to be separated, and select an element ED_i (with a leaf) or EC_i (with a node) away from the center of the first node as the center of the first new cluster N_L. Next, select element ED_j (or EC_j), which is the farthest from ED_i (or EC_j), as the center of the second new cluster N_R.
- If it is the root, the system creates a new root. If it is a node or a leaf, the system creates a new parent node or adds two new center elements to the parent node.
- The elements in the original node are distributed into two new nodes according to the rule of selecting the closest node based on Euclidean distance.
- Update the center element at the parent node cluster and perform recursion to the root.\

If a leaf is full, this leaf is split into two leaves and creates two new elements at the parent node to link these two leaves. If this parent node is full, continue to split into two nodes and create two new elements at the next parent node. And so on, if the root node is full, split into two nodes and create a new root node. Thus, the tree grows in the root direction through node splitting. Any node can be

Figure 2. Structure of Graph-CTree



split in half when full. Therefore, with k=2, the C-Tree is developed into a balanced multi-branch tree, while reducing the complexity of node splitting.

Neighbor Graph

Graph-CTree is a neighbor-clustering graph created based on a node-splitting process when training a C-Tree. Each time the leaf is split, the neighboring levels of the newly split leaf nodes are marked. The clustering problem on the tree will become the graph clustering problem whose main task is to connect related clusters to avoid missing similar image data in the search process. Figure 2 describes the structure of the Graph-CTree neighbor-clustering graph.

The Graph-CTree neighbor-clustering graph is described as follows: Graph-CTree $G = \langle V, E \rangle$ is an undirected graph, including:

- The set of vertexes V, which are clusters of leaves in C-Tree;
- The set of edges $E \subseteq V \times V$, which are the links of a pair of leaves, formed in the relationship of neighboring levels; and
- Neighboring levels of the leaves cluster:
  o Neighboring *level 1*:

Let f_{c_i}, f_{c_j} be center feature vectors of the i^{th} and j^{th} leaves L_i, L_j, where $f_{c_i} = average\left\{f_k \left| f_k \in L_i, k = 1..\left|L_i\right|\right.\right\}, f_{c_j} = average\left\{f_l \left| f_l \in L_j, k = 1..\left|L_j\right|\right.\right\}$. If the Euclidean distance between the centers of the two leaves follows $d_E(f_{c_i}, f_{c_j}) < \theta$, then L_i, L_j are neighboring *level 1*, and θ is the threshold value.

o Neighboring *level 2*:

Let Cla_i, Cla_j be the representative classes of the i^{th} and j^{th} leaves L_i, L_j, respectively. The representative class is the class with the most frequency in the cluster. If $Cla_i \equiv Cla_j$, then L_i, L_j are neighboring *level 2*.

Every neighboring level of a leaf, after being specified, creates a file to store its neighbors by filename *Neighbor[i]Leaf[j].txt*. Thus, the accuracy of retrieval is enhanced.

The algorithm of splitting a leaf and creating Graph-CTree is as follows:

Algorithm **SGT**: Split the leaf on C-Tree and create Graph_CTree
Input: threshold θ, Leaf, Graph_CTree;
Output: Graph_CTree
Function SGT(θ, Leaf, Graph_CTree)
Begin
center = average{Leaf.ED_i.f, i=1..M};
left = arg-max{Euclidean(center, Leaf.ED_i.f), i=1..M};
right = arg-max{Euclidean(Leaf.ED_{left}.f, Leaf.ED_i.f), i=1..M};
EDLeft = Leaf.ED_{left}; EDRight = Leaf.ED_{right};
LeftLeaf = EDLeft; RightLeaf = EDRight;
Foreach ed **in** Leaf **do**
If (Euclidean(ed.f, EDLeft.f < Euclidean(ed.f, EDRight.f)) **then**
LeftLeaf= LeftLeaf È ed;
Else
RightLeaf = RightLeaf È ed;
EndIf
EndForeach
ECLeft = average{LeftLeaf.$ED_{i.}$f, i=1..|LeftLeaf|};
ECRight = average{RightLeaf.$ED_{i.}$f, i=1..|RightLeaf|};
Leaf.Parent = Leaf.Parent È {ECLeft, ECRight};
If (Leaf.Parent.count > N) **then**
SplitNode(Leaf.Parent);
EndIf
If (Euclidean(ECLeft, ECRight) < θ) **then**
Neighbor[1].LeftLeaf = Neighbor[1].LeftLeaf È {RightLeaf};
Neighbor[1].RightLeaf = Neighbor[1].RightLeaf È {LeftLeaf};
EndIf
LeftClass = arg-max{count(LeftLeaf.$ED_{i.}$cla), i=1..|LeftLeaf|};
RightClass = arg-max{count(RightLeaf.$ED_{i.}$cla), i=1..|RightLeaf|};
If (LeftClass = RightClass) **then**
Neighbor[2].LeftLeaf = Neighbor[2].LeftLeaf È {RightLeaf};
Neighbor[2].RightLeaf = Neighbor[2].RightLeaf È {LeftLeaf};
EndIf
If (LeftClass **is-a** RightClass) **then**
Neighbor[3].LeftLeaf = Neighbor[3].LeftLeaf È {RightLeaf};
EndIf
If (RightClass **is-a** LeftClass) **then**
Neighbor[3].RightLeaf = Neighbor[3].RightLeaf È {LeftLeaf};
EndIf
Graph_CTree = Graph_CTree È {Neighbor[1], Neighbor[2], Neighbor[3]};
Return Graph_CTree;

End.

Thus, the structure of Graph-CTree is an improvement over C-Tree for two main reasons:

- Graph-CTree creates a neighbor graph from leaves in the process of splitting nodes on C-Tree and limiting the omission of data similarity, thus improving precision for image retrieval; and
- Graph-CTree is a neighbor-clustering graph that at each vertex can retrieve similar images at neighboring vertices, meaning that the number of related clusters in a search is not large, so the image retrieval time is faster than for other conventional graphs.

BUILDING AN ONTOLOGY FRAMEWORK

Building a Semi-Automatic Ontology Framework for the Image Dataset

To enable semantic retrieval on images, an ontology framework for the dataset was built on the RDF. The RDF is used to name the images and to describe the attributes of the images and the relationships between them. The ontology is used to define the attributes and the relationships. The ontology framework is generated by extracting and annotating the images from the ImageCLEF dataset (Escalante et al., 2010). It is capable of scaling the domain, inheriting ontology systems for the individual image, and describing the relationships between the image and its classes. Each image is an individual item of one or more classes in the ontology. However, image data are enormous and are added regularly to improve and enrich the image ontology. Therefore, manually generated image ontology will take a lot of time and effort, which is not efficient for large image databases. Thus, a semi-automatic ontology framework is built, regularly updated, domain extensible, and inherits ontology systems. In this framework, data are preprocessed, defining steps are automatically generated, steps are created manually, and there is a data check before creating the ontology. Thus, this proposed ontology framework reduces time and effort but still has high reliability.

To construct a semi-automatic ontology framework, first, it is necessary to prepare sample data for classes, class hierarchy, and attributes, and to prepare literals for class and image instances and class definitions for the ImageCLEF background dataset. These sample data are automatically obtained from sources such as the WWW image dataset. Next, the data samples are edited and updated by experts before creating the automatic ontology using software built on the dotNET Framework 4.8 platform using the C# programming language. The RDF and the Web Ontology Language (OWL) libraries in C# are used to model the RDF/OWL data.

Figure 3. Proposes a model to create a semi-automatic ontology framework. The process for creating this includes the following steps:

Step 1: Define the image dataset to create a framework for ontology: inherit the list of classes from the ImageCLEF dataset to add and create a hierarchy for the classes (1);

Step 2: Obtain images from WWW, render the resource identifiers, and describe the images (3). Then, take the identifier of the image and the URL description (4) and add the input data (5) to the ontology learning process;

Step 3: Create sample data for the elements of the ontology framework (2), including:

- ◦ **Manual phase:** create a hierarchy of classes from generated classes, create properties for classes and relationships between classes, and combine this with the information about identifiers of the URI resource, resource identifiers, and so on, from WWW, such as Wordnet, BabelNet, Wikipedia, and Dbpedia;
- ◦ **Automatic phase:** create classes inherited from the image dataset; create literals for individuals and classes; create individuals according to each class of the image. Each image is an instance that belongs to one or more classes of the ontology. Use the k-NN algorithm to classify images (8,9) and automatically add matching classes of ontology (10).

Step 4: Based on data samples that have been updated, edit to create a framework for ontology based on the C# language (6).

The number of images is very large and grows very fast, so the classification, semantic extraction, and addition to the ontology must be automatic. After these steps, an ontology framework is stored by the Notation 3 syntax (N3) (Figure 4), that each image object links to a URI, so in the future, we can query on www by semantic web approach.

According to such work evaluation, our model estimates over 60% automatically because the number of images is very large and increases rapidly compared to the class names. For example, ImageCLEF has 267 classes, while the number of images is 20000. Therefore, automatic enrichment of image instances for ontology is an important contribution to reduce ontology construction time.

Figure 3. Model of creating the semi-automatic ontology framework

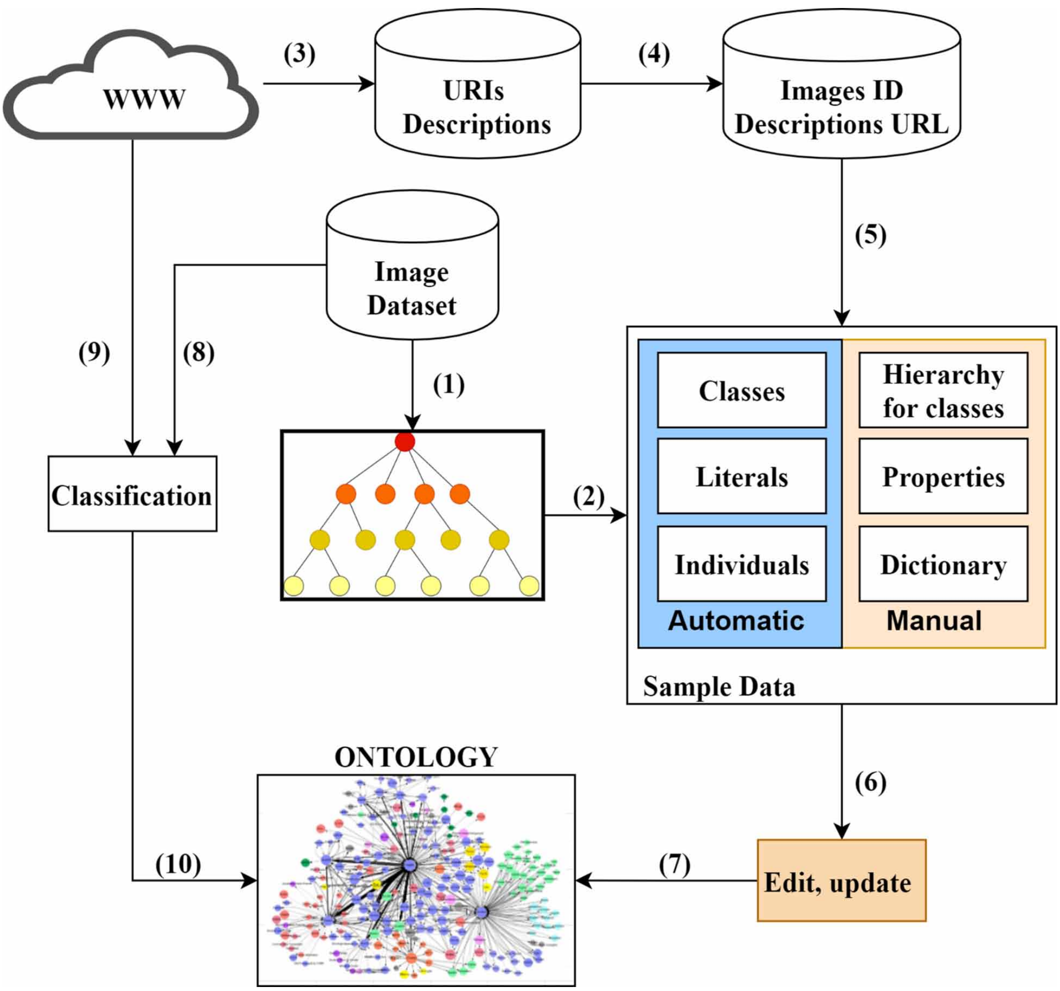


Figure 4. Ontology in N3 format

```
SBIR-Ontology.n3
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.
@prefix xml: <http://www.w3.org/XML/1998/namespace>.
@prefix sbir: <http://sbir-hcm.vn/>.

<http://sbir-hcm.vn/AEROSTATIC-BALLOON> a <http://www.w3.org/2002/07/owl#Class>;
                                        <http://www.w3.org/2000/01/rdf-schema#subClass
                                        Of> <http://sbir-hcm.vn/AIR-VEHICLES>.
<http://sbir-hcm.vn/AIRPLANE> a <http://www.w3.org/2002/07/owl#Class>;
                              <http://www.w3.org/2000/01/rdf-schema#subClassOf>
                              <http://sbir-hcm.vn/AIR-VEHICLES>.
<http://sbir-hcm.vn/AIR-VEHICLES> a <http://www.w3.org/2002/07/owl#Class>;
                                  <http://www.w3.org/2000/01/rdf-schema#subClassOf>
                                  <http://sbir-hcm.vn/VEHICLE>.
<http://sbir-hcm.vn/ANCENT-BUILDING> a <http://www.w3.org/2002/07/owl#Class>;
                                     <http://www.w3.org/2000/01/rdf-schema#subClassOf>
                                      <http://sbir-hcm.vn/EDIFICE>.
<http://sbir-hcm.vn/ANIMAL> a <http://www.w3.org/2002/07/owl#Class>.
<http://sbir-hcm.vn/anoDescription> a
<http://www.w3.org/2002/07/owl#AnnotationProperty>;
                                    <http://www.w3.org/2000/01/rdf-schema#range>
                                    <http://www.w3.org/2001/XMLSchema#string>.
```

The algorithms to build ontology are described as follows:
Algorithm **COC:** create ontology classes
Input: C = {class$_i$, i=1..N}, Ontology;
Output: Ontology;
Function COC (C, Ontology)
Begin
Foreach class **in** C **do**
Sub = “sbir:” + class;
Pre = “rdf:type”;
Obj = “owl:” + “Class”;
Ontology = Ontology È Triple(Sub, Pre, Obj);
EndForeach
Return Ontology;
End.
Algorithm **COCS:** create hierarchical classes in ontology
Input: superclass, subClass, Ontology;
Output: Ontology;
Function COCS (superclass, subClass, Ontology)
Begin
Foreach class **in** C **do**
Sub = “sbir:” + subClass;
Pre = “rdfs:subClassOf”;
Obj = “owl:” + superClass;
Ontology = Ontology È Triple(Sub, Pre, Obj);
EndForeach
Return Ontology;
End.

Algorithm **CIC:** create individual in ontology
Input: C = {$class_i$, i=1..N}, Individual, Ontology;
Output: Ontology;
Function COCS (C, Individual, Ontology)
Begin
Foreach class **in** C **do**
Sub = “sbir:” + Individual.ElementAt(i);
Pre = “rdf:type”;
Obj = “owl:NamedIndividual”;
Ontology = Ontology È Triple(Sub, Pre, Obj);
EndForeach
Return Ontology;
End.

Each classified vocabulary in the dictionary describes information, such as images representing vocabulary, meta-data, descriptions of vocabulary, and the concepts of that vocabulary. Therefore, the semi-automatic ontology framework is built from the ImageCLEF dataset, creating the foundation for additional sets of images and semantic-based image retrieval.

SPARQL Query on a Semi-Automatic Ontology Framework

SPARQL query is applied to query on N3 for semantic-based image retrieval on the ontology. An input query image can contain one or more objects. The system extracts low-level features of an image, then searches a combined model of a C-Tree and a neighbor graph (Graph-CTree) for a set of similar images. The k-NN algorithm is used to classify the image and store the classes representing the query image into a visual word vector. The algorithm to create a set of visual words is described as follows:

Algorithm **CVWV:** Create a set of visual word
Input: The set of similar images SI, threshold γ
Output: A set of visual word ***W***
Function CVWV(SI, γ)
Begin
L = ClassName.SI;
For (Li Î L) **do**
If freq(Li, L) ³ γ **then** W = W È Li; **EndIf**
EndFor
Return W;
End.

The SPARQL query is generated from a set of visual words to retrieve an ontology. The result of this query is a set of similar images and a set of semantic descriptions of images, including meta-data, URI identifiers, and semantic concepts of images mapped from a synonym dictionary of the ontology. The SPARQL query is generated from visual word vectors according to the following algorithm:

Algorithm **GSQ:** generate SPARQL queries
Input: A set of visual word ***W***
Output: SPARQL query
Function CSP(W)
Begin
n = W.count;
SPARQL = “SELECT DISTINCT ?Img WHERE{”;
For i=0 **to** n **do**
SPARQL += <subject>:W(i)+ “rdf:type”+<object>:+?Img+”UNION/AND”;
EndFor;

SPARQL += "}";
Return SPARQL;
End.

The input of the above algorithm is a set of visual words W, and the output is a SPARQL query to query on the ontology. The number of words is counted in a set of visual words and execute the SPARQL query according to those words. Figure 5 shows an example of the SPARQL query generated from a set of visual words (*ocean*, *cloud*) of an input image: 15372.jpg from the ImageCLEF dataset. The SPARQL query is created in the standard of the World Wide Web Consortium (W3C) and can be queried in two forms: UNION query or AND query.

Figure 5. An example of creating a SPARQL query from a visual word vector

Query Image

UNION Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX sbir: <http://sbir-hcm.vn/>
SELECT DISTINCT ?Subject
WHERE{{?Subject sbir:opOCEAN sbir:inOCEAN .}
          UNION
          {?Subject sbir:opCLOUD sbir:inCLOUD .}}
```

AND Query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX sbir: <http://sbir-hcm.vn/>
SELECT DISTINCT ?Subject
WHERE{{?Subject sbir:opOCEAN sbir:inOCEAN .}
          {?Subject sbir:opCLOUD sbir:inCLOUD .}}
```

SEMANTIC-BASED IMAGE RETRIEVAL

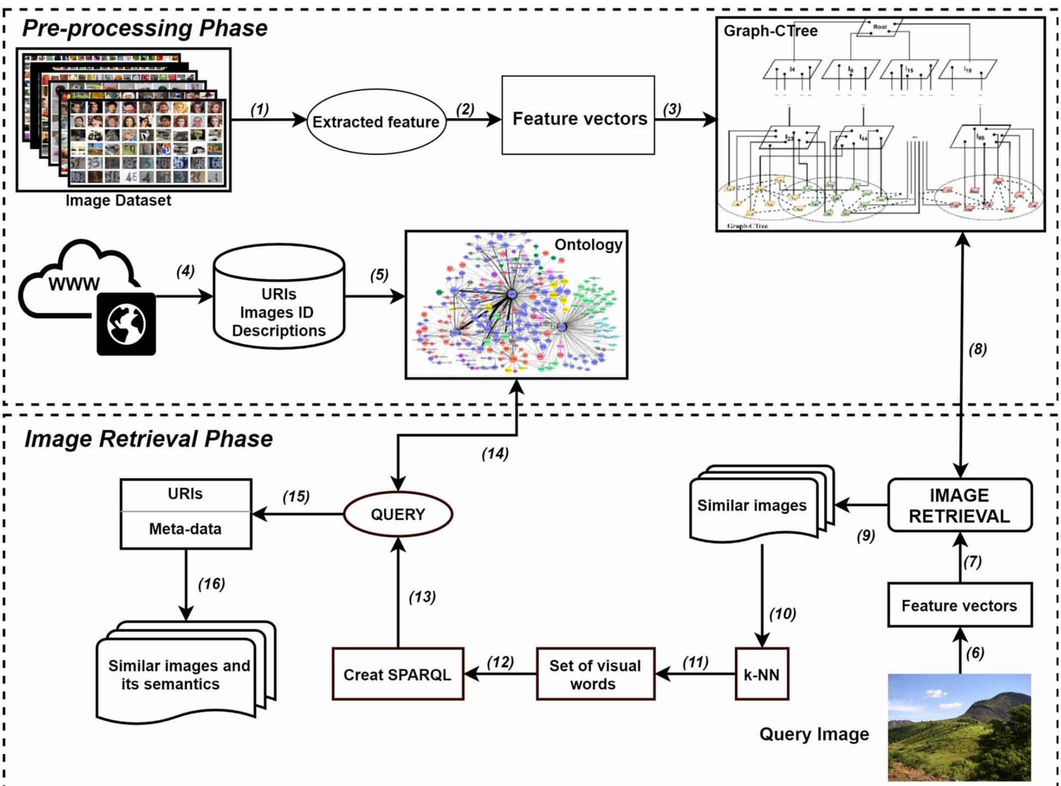
A Model of Semantic-Based Image Retrieval Based on Graph-CTree

The system of semantic-based image retrieval based on the Graph-CTree model and ontology is called SBIR_grCT. The query system consists of two phases: (1) the pre-processing phase generates a neighbor-clustering graph of the leaf clusters while also building a semi-automatic ontology framework based on the RDF triple language and stores it in N3 format; (2) the query phase searches for a set of similar images on the Graph-CTree model, classifies the image with the k-NN algorithm to find visual word vectors, and creates the SPARQL query, which helps query on the ontology to render high-level semantics of the image. Figure 6 shows the architecture of the SBIR_grCT based on the Graph-CTree model with two specific phases as follows:

Phase One: Pre-processing phase:
Step 1: Have low-level features segmented and extracted from the image dataset (1);

Step 2: Create the data samples (2) containing the feature vectors and the image classes from the segmentation and extraction of the feature vector of the image;
Step 3: Create a model combining C-Tree and the neighbor graph (Graph-CTree) from the data samples (3); and
Step 4: Build a semi-automatic ontology framework from the WWW (4) and ImageCLEF datasets (5).
Phase Two: Image-retrieval phase:
Step 1: Have each input query image extracted into the feature vector (6);
Step 2: Perform the query (7) on Graph-CTree, first, based on the Euclidean measure to find the most suitable leaf on C-Tree, then, taking the set of neighboring nodes with this leaf on the neighbor graph (8)—the result is a set of similar images and an arrangement of a set of similar images by measure (9);
Step 3: Classify the similar image dataset based on a k-NN algorithm (10) to create a set of visual words (11);
Step 4: Automatically create the SPARQL query from a set of visual words (12) and execute the query on a framework for ontology (13);
Step 5: the result of the semantic-based image retrieval on this ontology (14) is the meta-data, URIs (15), the set of similar images, and its semantics (16).

Figure 6. Model of the semantic-based image retrieval system (SBIR_grCT)



The precision of image retrieval depends on the choice of low-level features. In this model (Figure 6), low-level features (color, texture, shape) are used to form a combined feature descriptor. Colors are extracted based on the dominant color descriptor of MPEG-7. Texture features are extracted based on contrast, the high-frequency filter, the Sobel filter, and the Gaussian filter. Shape features are extracted by relying on the Laplacian of Gaussian method. An 81-dimensional low-level feature vector is extracted for the image retrieval system in this paper.

At the same time, although the deep learning technique is being widely used in image recognition and classification, it requires a large dataset to complete the training steps with high computational complexity. Thus, in this paper, we classify a set of similar images (found on Graph-CTree) to generate a set of visual words. Because this image dataset has a small size, the use of a deep learning network in this step is not effective; therefore, we propose the k-NN algorithm, which has low computational complexity but still has high precision to classify the set of similar images.

Experiment

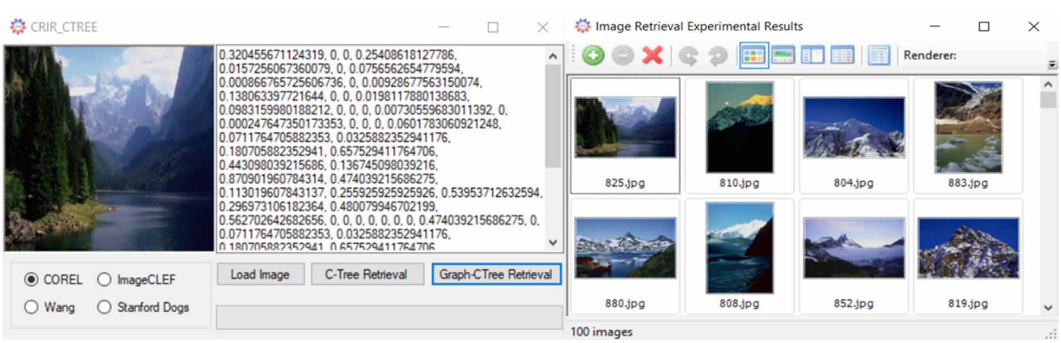
Experimental Environment

In the experiment, the SBIR_grCT system was built on the dotNET Framework 4.8 using the C# programming language. The SBIR_grCT query system was implemented on a computer with an Intel (R) CoreTM i7-8750H processor, 2.70GHz CPU, 8GB RAM, and Windows 10 Professional operating system. The image datasets used in the experiment were COREL, WANG, ImageCLEF, and Stanford Dogs, as described in Table 1. ImageCLEF is a multi-object image dataset used to build the ontology for semantic-based image retrieval and for the experiment with content-based image retrieval on Graph-CTree. In addition, single-object image datasets, such as COREL, WANG, and Stanford Dogs, were also tested to evaluate the accuracy of Graph-CTree.

Table 1. The image datasets

No.	Dataset	Number of images	Number of subjects	Size
1	COREL	1,000	10	30.3 MB
2	WANG	10,800	80	62.2 MB
3	ImageCLEF (IAPR TC-12 brenchmark)	20,000	276	1.6 B
4	Stanford Dogs	20,580	120	778.0 MB

Figure 7. The results of the content-based image retrieval of SBIR_grCT



Experimental Application

The image retrieval system SBIR_grCT performs two processes: (1) content-based image retrieval according to Graph-CTree, and (2) semantic-based image retrieval on ontology. For an input image, first, the SBIR_grCT system extracts the feature vector and searches a set of similar images on Graph-CTree.

Figure 8. The results of the semantic-based image retrieval of SBIR_grCT

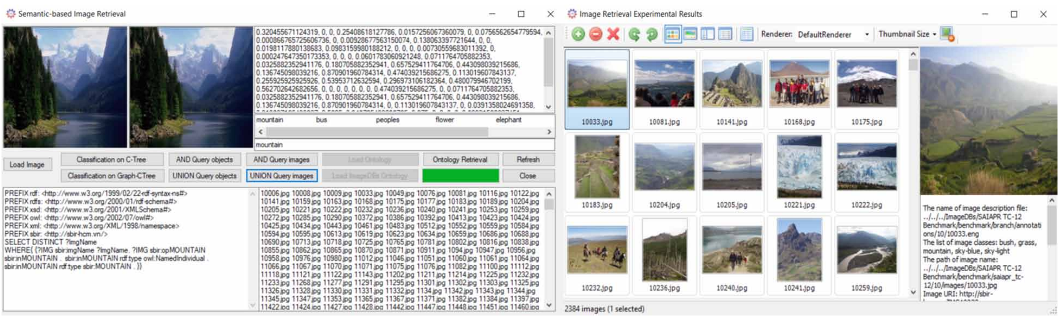


Figure 9. The semantic concept for class

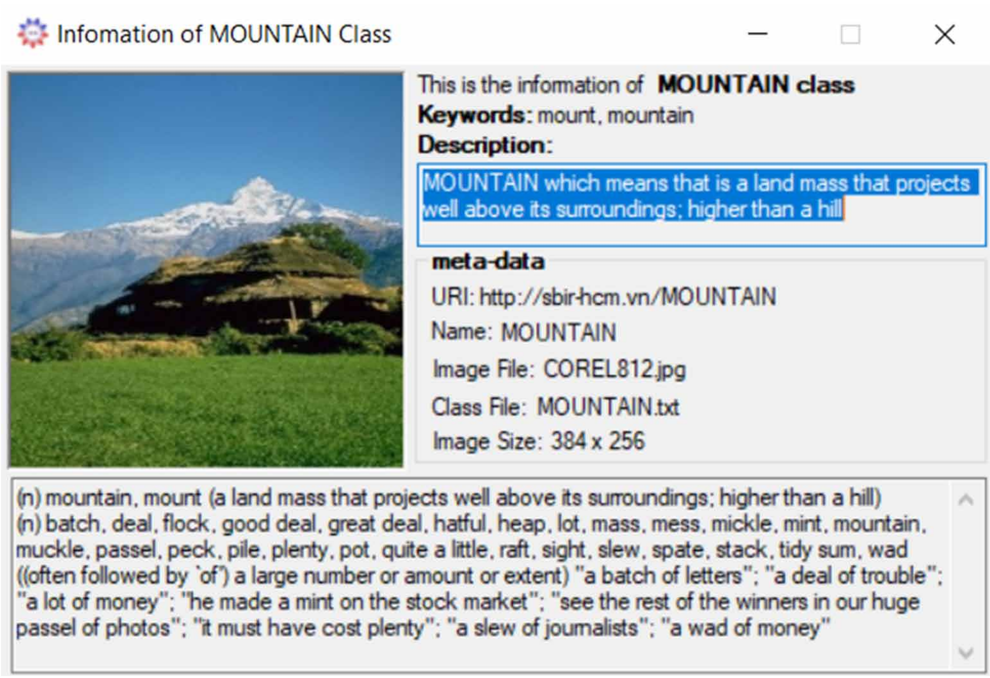


Figure 7 shows the result of finding a content-based image of SBIR_grCT. Then, from the set of content-based similar images, the SBIR_grCT performs image classification on Graph-CTree. The SPARQL query is generated to retrieve a semantic-based image on the ontology. A UNION or AND query is created depending on the goal of the user. The query result on the framework for ontology is a set of semantically similar images. Figure 8 displays the results of the semantic-based image retrieval and classification using the SBIR_grCT system. Each image in the set of similar images is described with the meta-data for the image annotation and the URIs of the image. At the same time, the classes derived through the classification process of the similar image dataset are mapped to the synonym dictionary of ontology to extract general concepts, equivalent keywords, the image representation of the classes, and its meta-data. Figure 9 is an example of this.

Experimental Results and Evaluation

To evaluate the SBIR_grCT system, factors such as precision, recall and F-measure, and query time (milliseconds) were used. The average query times and performance values for COREL, WANG, ImageCLEF, and StanfordDogs on C-Tree and Graph-CTree are presented in Tables 2–5.

Table 2. Performance of SBIR_grCT on the COREL dataset

Performance	Avg. precision	Avg. recall	Avg. F-measure	Avg. query time (ms)
C-Tree	0.677655	0.699885	0.688521	19.91437
Graph-CTree	0.888473	0.884555	0.886346	72.65352

Table 3. Performance of SBIR_grCT on the WANG dataset

Performance	Avg. precision	Avg. recall	Avg. F-measure	Avg. query time (ms)
C-Tree	0.607222	0.489184	0.545011	39.746901
Graph-CTree	0.7664731	0.667659	0.713353	176.8820

Table 4. Performance of SBIR_grCT on the ImageCLEF dataset

Performance	Avg. precision	Avg. recall	Avg. F-measure	Avg. query time (ms)
C-Tree	0.606217	0.409401	0.474718	44.08912
Graph-CTree	0.839814365	0.780735832	0.806435717	239.9458

Table 5. Performance of SBIR_grCT on the Stanford Dogs dataset

Performance	Avg. precision	Avg. recall	Avg. F-measure	Avg. query time (ms)
C-Tree	0.570385	0.37308	0.450394	48.30427
Graph-CTree	0.826416	0.513825	0.630859	261.8991

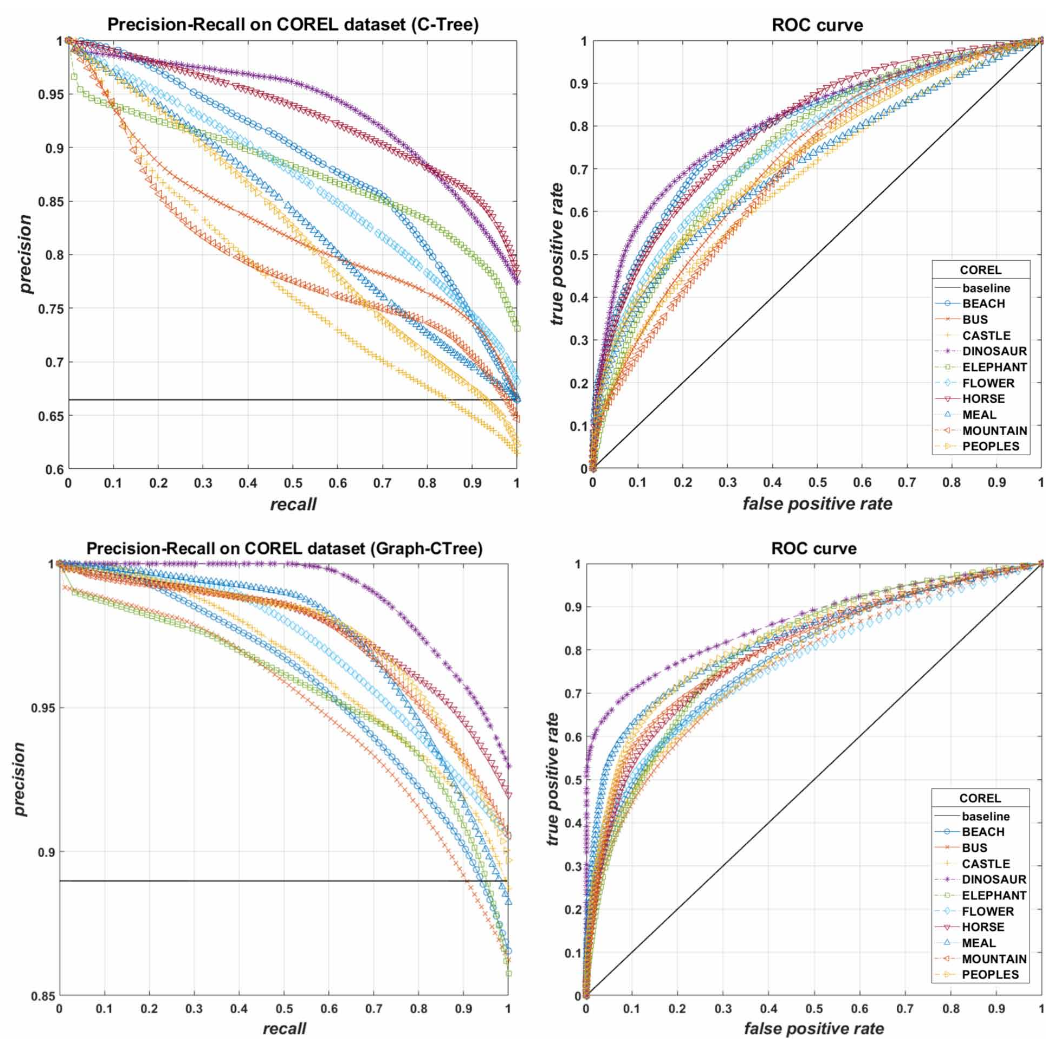
Tables 2–5 show that the precision of Graph-CTree is better than C-Tree due to the fact that searching the neighbor graph does not omit similar data because it can find the relevant leaf clusters.

However, the query time for Graph-CTree is higher than that for C-Tree since the neighbor graph requires a lot of memory to keep track of all clusters of neighboring nodes.

In a Precision-Recall curve graph, each curve is considered as an image folder of each image dataset. The larger area under the curve (AUC), the higher the precision. Meanwhile, in the Receiver Operating Characteristic (ROC) graph, a diagonal baseline divides the space of the ROC into two parts. The points above the baseline represent the correct classification results. The points below the baseline are the results of false classification. Points further above the baseline will give better classification results than those near the baseline. Based on the ROC curve, it can be determined whether a model is efficient or not. The curve closer to the coordinates (0, 1) on the graph (upper-left corner) represents the more efficient model.

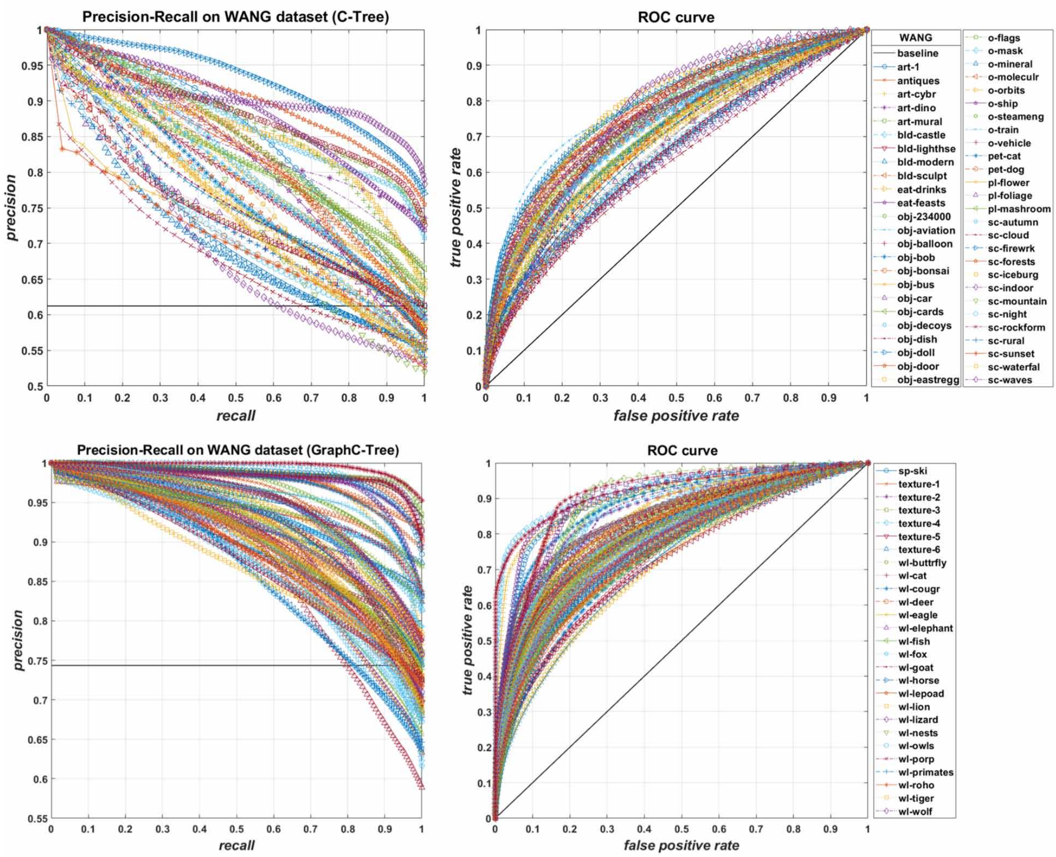
The Precision-Recall curve graphs and the ROC graphs of the image datasets (COREL, WANG, ImageCLEF, Stanford Dogs) are shown in Figures 10, 11, 12, and 13, respectively. The Precision-Recall curve graph indicates that the AUC of Graph-CTree is larger than that of C-Tree in all datasets, showing that the proposed improvements resulted in better precision. At the same time, all points

Figure 10. Precision-Recall curve and ROC curve on COREL dataset



on the ROC graphs are above the baseline, indicating that the image classification results are good. Observing the ROC graph, the points for Graph-CTree for the image sets are all farther above the baseline and closer to the coordinates (0, 1) than are those for C-Tree. Consequently, Graph-CTree's image classification results are better than those of C-Tree. Thus, the proposed model is efficient.

Figure 11. Precision-Recall curve and ROC curve on WANG dataset



To evaluate the precision and efficiency of the SBIR_grCT system, the obtained performance is compared with the results of related works using the same image datasets. Table 6 describes the results of the Mean Average Precision (MAP) comparison for the COREL dataset. The table also shows that our proposed method, when applied to COREL, gives better MAP values than other works. This proves that our proposal is effective for the COREL dataset.

Table 7 compares our proposed method with other methods using the WANG dataset. The results show that the MAP of our proposed method is higher than the alternatives. Therefore, it is effective for the WANG dataset.

Tables 8 and 9 show the results of the MAP comparison between our proposed method and other methods using the ImageCLEF and StanfordDogs datasets. The results show that our proposed method is effective for those datasets.

Figure 12. Precision-Recall curve and ROC curve on ImageCLEF dataset

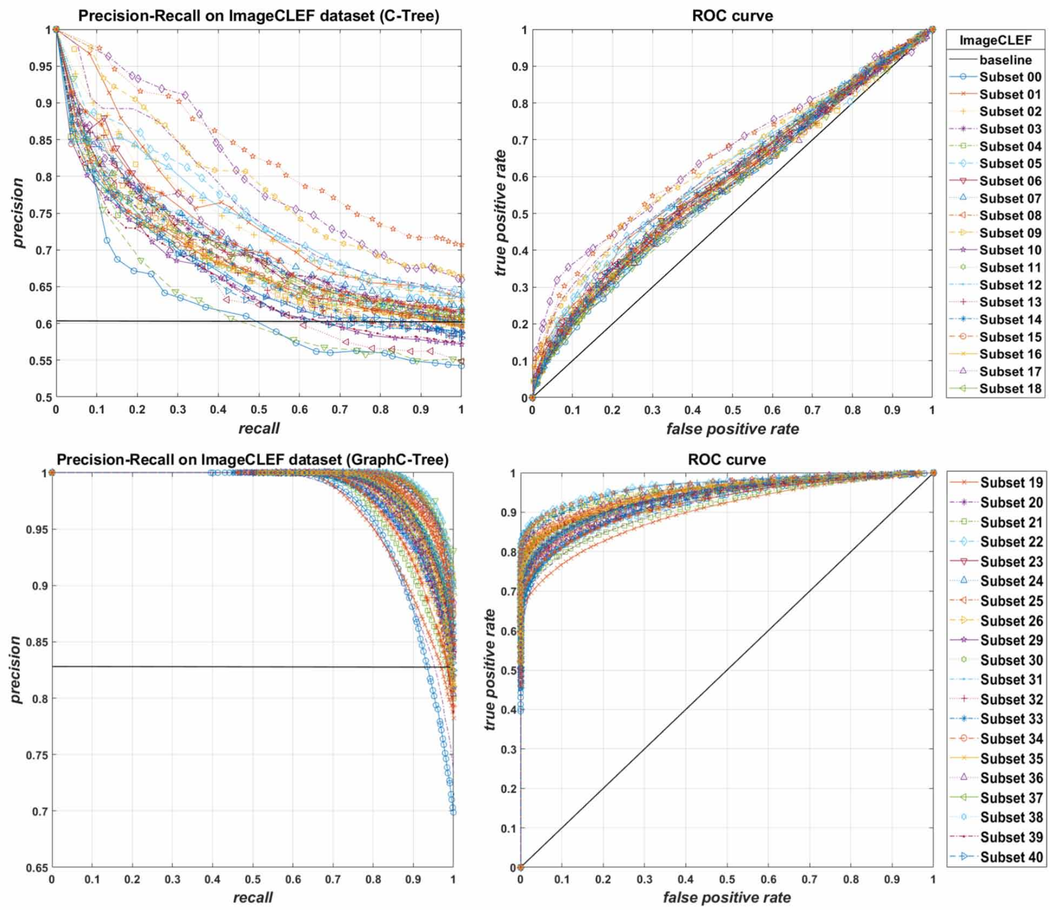


Figure 13. Precision-Recall curve and ROC curve on StanfordDogs dataset

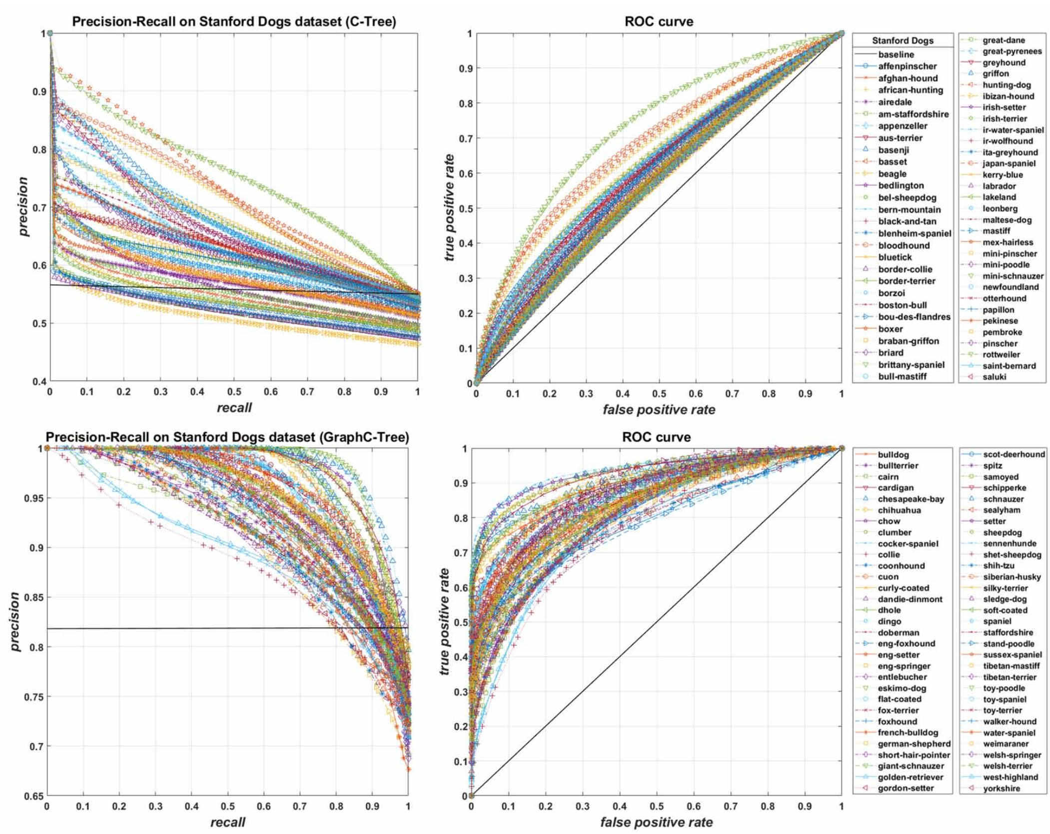


Table 6. Comparison of mean average precision (MAP) of methods on COREL dataset

Methods	Mean Average Precision (MAP)
Feature descriptions SURF and FREAK (Jabeen et al., 2018)	0.8251
Binary signatures cluster graph (Van & Le, 2018)	0.8262
Neuro-Fuzzy (Garg et al., 2019)	0.7340
FIF-IRS (Bella et al., 2019)	0.8330
Fuzzy c-means + SVM+SURF,ORB (Bibi R. et al., 2020)	0.8854
SIFT 8-dimensions with MPL, 2020 (Chhabra et al., 2020)	0.7711
Graph-CTree	**0.8885**

Table 7. Comparison of mean average precision (MAP) of methods on WANG dataset

Methods	Mean Average Precision (MAP)
CDH (Color Difference Histogram)+GLCM+ Zernike moment (Pavithra LK., 2019)	0.7534
Feature fusion method using BoVW (Vimina et al., 2019)	0.6920
The texture techniques GLCM (Dhingra et al., 2020)	0.7500
SIFT 8-dimensions with MPL, 2020 (Chhabra et al., 2020)	0.6320
Graph-CTree	**0.7665**

Table 8. Comparison of mean average precision (MAP) of methods on ImageCLEF dataset

Methods	Mean Average Precision (MAP)
Binary signatures cluster graph (Van & Le, 2018)	0.7945
BoVW and hybrid deep learning architecture HDLA (Arun KS. et al., 2018)	0.7430
Semantic Deep Cross-modal Hashing (SDCH) method (Yan et al., 2019)	0.8030
Deep semantic similarity adversarial hashing (DSSAH) 64 bit (Qiang et al., 2020)	0.5950
Graph-CTree	**0.8398**

Table 9. Comparison of mean average precision (MAP) of methods on Stanford Dogs dataset

Methods	Mean Average Precision (MAP)
GeM pooling with SCDA's feature (Wang et al., 2018)	0.7918
Semantics-preserving hashing method (Sun et al., 2020)	0.8220
Deep Saliency Hashing (DsaH) 48 bit (Jin et al., 2020)	0.408
Graph-CTree	**0.826416**

The data from the above tables indicate that the proposed method produces higher MAP values when compared with other methods that use the same image dataset. Thus, our proposed method is effective in solving the image-retrieval problem and semantic analysis of single- and multi-object image sets.

CONCLUSION

This paper proposes a model combining a C-Tree and a neighbor-clustering graph named Graph-CTree. This model was built on a semi-supervised learning technique that combines hierarchical clustering and partitional clustering methods to map low-level features of an image into high-level semantics. From then, a semantic-based image retrieval system, SBIR_grCT, was built based on the Graph-CTree and ontology. The experiment was performed on four image datasets—COREL, WANG, ImageCLEF, and StanfordDogs—with precision values of 0.888473, 0.766473, 0.839814, and 0.826416, respectively. To evaluate the proposed methods, the experimental results were compared with other methods on the same image dataset. The comparison results indicate that the SBIR_grCT system is effective and precise. In the future, a semi-automatic ontology will be built from WWW image collections; in doing so, the ontology framework will add and be enriched by other image sets. In conclusion, the structure and query algorithms have been improved on the C-Tree and Graph-CTree to enhance the efficiency of semantic-based image retrieval.

ACKNOWLEDGMENT

The authors would like to thank the Faculty of Information Technology, University of Science–Hue University, for their professional advice about this study. This work has been sponsored and funded by the Ho Chi Minh City University of Food Industry under Contract No. 147/HD-DCT.

REFERENCES

Allani, O., Zghal, H. B., Mellouli, N., & Akdag, H. (2017). Pattern graph-based image retrieval system combining semantic and visual features. *Multimedia Tools and Applications*, *76*(19), 20287–20316. doi:10.1007/s11042-017-4716-8

Arun, K. S., & Govindan, V. K. (2018). A hybrid deep learning architecture for latent topic-based image retrieval. *Data Science and Engineering*, *3*(2), 166–195. doi:10.1007/s41019-018-0063-7

Barz, B., & Denzler, J. (2020). Deep Learning on small datasets without pre-training using cosine loss. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)* (pp. 1371–1380). IEEE. doi:10.1109/WACV45572.2020.9093286

Bella, M. I. T., & Vasuki, A. (2019). An efficient image retrieval framework using fused information feature. *Computers & Electrical Engineering*, *75*, 46–60. doi:10.1016/j.compeleceng.2019.01.022

Bibi, R., Mehmood, Z., Yousaf, R. M., Saba, T., Sardaraz, M., & Rehman, A. (2020). Query-by-visual-search: Multimodal framework for content-based image retrieval. *Journal of Ambient Intelligence and Humanized Computing*, *11*(11), 5629–5648. doi:10.1007/s12652-020-01923-1

Bouchakwa, M., Ayadi, Y., & Amous, I. (2020). Multi-level diversification approach of semantic-based image retrieval results. *Progress in Artificial Intelligence*, *9*(1), 1–30. doi:10.1007/s13748-019-00195-x

Cevikalp, H., Elmas, M., & Ozkan, S. (2018). Large-scale image retrieval using transductive support vector machines. *Computer Vision and Image Understanding*, *173*, 2–12. doi:10.1016/j.cviu.2017.07.004

Chhabra, P., Garg, N. K., & Kumar, M. (2020). Content-based image retrieval system using ORB and SIFT features. *Neural Computing & Applications*, *32*(7), 2725–2733. doi:10.1007/s00521-018-3677-9

Dhingra, S., & Bansal, P. (2020). Experimental analogy of different texture feature extraction techniques in image retrieval systems. *Multimedia Tools and Applications*, *79*(37), 27391–27406. doi:10.1007/s11042-020-09317-3

Escalante, H. J., Hernández, C. A., Gonzalez, J. A., López-López, A., Montes, M., Morales, E. F., Sucar, L. E., & Grubinger, M. (2010). The segmented and annotated IAPR TC-12 benchmark. *Computer Vision and Image Understanding*, *114*(4), 419–428. doi:10.1016/j.cviu.2009.03.008

Filali, J., Zghal, H., & Martinet, J. (2017). Towards visual vocabulary and ontology-based image retrieval system. In *Proceedings of* the *International Conference on Agents and Artificial Intelligence* (pp. 560–565). SCITEPRESS.

Furche, T., Linse, B., Bry, F., Plexousakis, D., & Gottlob, G. (2006). *RDF querying: Language constructs and evaluation methods compared. In Reasoning web* (Vol. 4126). Lecture Notes in computer science. Springer.

Garg, M., Singh, H., & Malhotra, M. (2019). Fuzzy-NN approach with statistical features for description and classification of efficient image retrieval. *Modern Physics Letters A*, *34*(3), 1–24. doi:10.1142/S0217732319500226

Gonçalves, F. M. F., Guilherme, I. R., & Pedronette, D. C. G. (2018). Semantic guided interactive image retrieval for plant identification. *Expert Systems with Applications*, *91*, 12–26. doi:10.1016/j.eswa.2017.08.035

Haase, P., Broekstra, J., Eberhart, A., & Volz, R. (2004). A comparison of RDF query languages. *International Semantic Web Conference: Lecture notes in computer science*, vol. 3298 (pp. 502–517). Springer.

Hogan, A. (2020). *SPARQL query language. In The web of data*. Springer.

Jabeen, S., Mehmood, Z., Mahmood, T., Saba, T., Rehman, A., & Mahmood, M. T. (2018). An effective content-based image retrieval technique for image visuals representation based on the bag-of-visual-words model. *PLoS One*, *13*(4), e0194526. doi:10.1371/journal.pone.0194526 PMID:29694429

Jin, S., Yao, H., Sun, X., Zhou, S., Zhang, L., & Hua, X. (2020). Deep saliency hashing for fine-grained retrieval. *IEEE Transactions on Image Processing*, *29*, 5336–5351. doi:10.1109/TIP.2020.2971105 PMID:32191885

Jiu, M., & Sahbi, H. (2017). Nonlinear deep kernel learning for image annotation. *IEEE Transactions on Image Processing*, *26*(4), 1820–1832. doi:10.1109/TIP.2017.2666038 PMID:28186895

Nhi, N. T. U., Van, T. T., & Thanh, L. M. (2020). A self-balanced clustering tree for semantic-based image retrieval. *Journal of Computer Science and Cybernetics*, *36*(1), 49–67. doi:10.15625/1813-9663/36/1/14347

Pavithra, L. K., & Sharmila, T. S. (2019). An efficient seed points selection approach in dominant color descriptors (DCD). *Cluster Computing*, *22*(4), 1225–1240. doi:10.1007/s10586-019-02907-3

Qiang, H., Wan, Y., Xiang, L., & Meng, X. (2020). Deep semantic similarity adversarial hashing for cross-modal retrieval. *Neurocomputing*, *400*, 24–33. doi:10.1016/j.neucom.2020.03.032

Sarwara, S., Qayyum, Z. U., & Majeed, S. (2013). Ontology based image retrieval framework using qualitative semantic image descriptions. *Procedia Computer Science*, *22*, 285–294. doi:10.1016/j.procs.2013.09.105

Sun, H., Fan, Y., Shen, J., Liu, N., Liang, D., & Zhou, H. (2020). A Novel Semantics-Preserving Hashing for Fine-Grained Image Retrieval. *IEEE Access: Practical Innovations, Open Solutions*, *8*, 26199–26209. doi:10.1109/ACCESS.2020.2970223

Nguyen Thi Uyen Nhi graduated in Computer Science in 2008 from Volgograd State Technical University, Russia. In 2010, she received a Master's degree in Computer Science from Volgograd State Technical University, Russia. From 2017 to now, she is a PhD student in Computer Science at the University of Science, Hue, Vietnam. Her research interests include image processing, image retrieval.

Thanh Manh Le received a Ph.D. degree in computer science from Budapest University (ELTE), Hungary, in 1994. He became an associate professor at Hue University, Vietnam, in 2004. His research interests include databases, knowledge bases and logic programming.

Thanh The Van graduated in Mathematics & Computer Science in 2001 from University of Science - Vietnam National University HCMC. In 2008, he obtained a Master's degree in Computer Science from Vietnam National University HCMC. In 2016, he received a PhD degree in Computer Science from University of Science, Hue, Vietnam. His research interests include image processing, image mining, and image retrieval.