Pattern Recognition 74 (2018) 266–285

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Graph-based bag-of-words for classification

Fernanda B. Silva[a,1], Rafael de O. Werneck[a,*], Siome Goldenstein[a], Salvatore Tabbone[b], Ricardo da S. Torres[a]

[a] RECOD Lab, Institute of Computing (IC), University of Campinas (Unicamp) Av. Albert Einstein, 1251, Campinas 13083-852, SP, Brazil
[b] Université de Lorraine-LORIA UMR 7503 BP 239, 54506 Vandoeuvre-lès-Nancy, France

ARTICLE INFO

Article history:
Received 6 September 2016
Revised 25 July 2017
Accepted 8 September 2017
Available online 12 September 2017

Keywords:
Bag models
Graph matching
Graph-based retrieval
Graph-based classification

ABSTRACT

This paper introduces the *Bag of Graphs (BoG)*, a Bag-of-Words model that encodes in graphs the local structures of a digital object. We present a formal definition, introducing concepts and rules that make this model flexible and adaptable for different applications. We define two BoG-based methods – *Bag of Singleton Graphs (BoSG)* and *Bag of Visual Graphs (BoVG)*, which create vector representations for graphs and images, respectively. We evaluate the Bag of Singleton Graphs (BoSG) for graph classification on four datasets of the IAM repository, obtaining significant results in accuracy and execution time. The method Bag of Visual Graphs (BoVG) is evaluated for image classification on Caltech and ALOI datasets, and for remote sensing image classification on images of Monte Santo and Campinas datasets. This framework opens possibilities for retrieval, classification, and clustering tasks on large datasets that use graph-based representations impractical before due to the complexity of inexact graph matching.

1. Introduction

Huge volumes of digital data have been created due to advances in acquiring, storing, sharing, and managing technologies. In this scenario, the appropriate use of data depends on the development of effective and efficient classification and retrieval tools, which in turn require the design of discriminant representation models of objects that enable us to identify/encode their similarities.

Bag-based representations have been extensively used to compute the similarity among digital objects by characterizing the frequency of occurrence of object features, *Bag of Words (BoW)* being one of the first successful models to create a vector representation of textual documents based on the frequency of word occurrences [1]. The adaptation of BoW for image context [2] is called *Bag of Visual Words (BoVW)*, or *Bag of Features*. This approach represents an image as a collection of visual words, where each visual word refers to a relevant visual pattern. The image descriptor is created based only on the number of occurrences of some particular visual appearances within the image.

This BoW model is a simple and efficient form of representation that enables a fast computation of object similarities. However, recent studies [3–5] have investigated the use of spatial information to improve these representations. Including local structures into the object description process can improve the bag representation, and lead to improvements in several tasks, which are dependent on the identification of semantic similarities. In fact, semantic meaning is a subjective concept, and it is not easily mapped to many digital objects. Sometimes we can use as evidence of similarity between a pair of objects the presence of *similar patterns* within them. These patterns may be defined in terms of relationships among object components, like spatial proximity. Thus, the use of a representation that describes an object through its local structures can lead to effective solutions for the recognition and categorization of digital objects. In this sense, graphs are a flexible tool for modeling relationships, and they are particularly useful for representing local structures within a digital object. Additionally, the invariance of graphs to several geometric transformations allows the creation of robust representations.

The hypothesis we explore in this paper is that the combination of graphs with the BoW model can create a discriminant and efficient representation based on local structures of an object, leading to fast and accurate results in classification tasks. The rationale is that the two representations are complementary and can help each other overcome their individual deficiencies. Graphs can encode local structures into a BoW-based descriptor, which can improve bag

[*] Corresponding author.
E-mail addresses: fernanda.silva@students.ic.unicamp.br (F.B. Silva), rafael.werneck@ic.unicamp.br (R.O. Werneck), siome@ic.unicamp.br (S. Goldenstein), tabbone@loria.fr (S. Tabbone), rtorres@ic.unicamp.br (R.S. Torres).
[1] Present address: Microsoft ATL, Rio de Janeiro, Brazil

http://dx.doi.org/10.1016/j.patcog.2017.09.018
0031-3203/© 2017 Elsevier Ltd. All rights reserved.

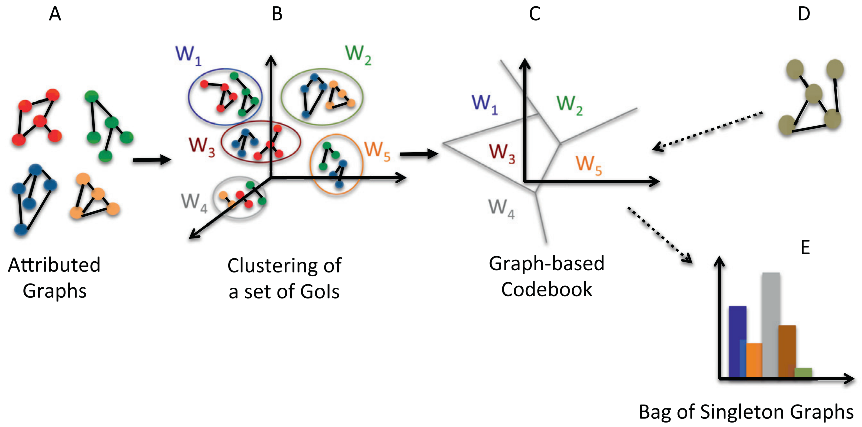

Fig. 1. Overview of the Bag of Singleton Graphs. We describe a set of graphs (A) in terms of vertex signatures, cluster them (B) to build the codebook (C), and count codeword occurrences within an input graph (D) to create the bag representation (E).

representations. At the same time, the use of BoW-based representations reduces the amount of time required by graph-based methods to compute the similarity between objects.

This paper introduces a novel object descriptor that combines bag and graph representations. We propose the *BoG*, a generic approach that creates a vector representation based on local structures defined by graph elements. This theoretical framework may be adapted to different contexts, and in this paper, we further describe two concrete realizations of the generic framework.

The first approach, called *BoSG*, generates a bag representation for objects that were previously modeled as graphs with attributes associated with their vertices and edges (illustrated in Fig. 1). The second approach, denominated *BoVG*, creates BoW-based descriptors using graphs to model the spatial relationships between the visual words found within an image. We also discuss the use of *BoVG* in the creation of a graph-based visual representation for remote sensing images that models the spatial relationships among their labeled regions. All case studies obtain accuracy rates comparable to other methods of the literature when evaluated on standard datasets [6–8].

This paper extends the works presented in [9,10]. Those papers present the use of the Bag-of-Graphs models concerning graph and image object classification problems. None of them, however, provides a comprehensive formal description of the model, which may guide researchers and developers in the creation of novel realizations and extensions. Another novelty of this journal paper refers to the introduction of a novel realization of the proposed model in the context of remote sensing image (RSI) representation and classification tasks. Finally, the experimental protocol was extended in order to include experiments with the ALOI dataset (in the case of image object classification) and model validation in RSI classification tasks. In summary, the main contribution of this work is the formal description of a generic graph model for digital object representation, with substantial practical demonstration through the instantiation, implementation, and validation of the theory in three real and distinct problems.

2. Related work

Graph is an abstract structure [11] that can be easily adapted, allowing its application in domains that range from biology to engineering [12,13]. Graphs can capture the relationships of an object's internal parts while being invariant to some transformations [14].

2.1. Graph representation

Some examples in image representation are the *graph of interest points* [15,16], *the graph of adjacent regions* [17,18], *the skeleton graph* [19–22], *the graph of primitives* [23,24], and *the graph of face fiducial points* [25].

In object recognition, *Spatial Relational Graphs (SRGs)* [24] describe symbols based on topological relationships of graphic primitives, while *Attributed Relational Graph (ARG)* can capture both topological and directional spatial relationships [23]. *Spatial Orientation Graph (SOG)* [26,27] describe the spatial positioning of objects within an image while *Skeleton Graphs* [28] and *Complete Graphs* [29] model the geometry of parts.

2.2. Graph matching

For most pattern recognition, indexing, and retrieval tasks, it is essential to compare similarities between data elements. So, when using graphs to represent objects, those tasks require the computation of the similarity between pairs of graphs, a complex problem usually addressed by graph-matching approaches, either exact or inexact.

Exact graph-matching algorithms determine if two graphs are isomorphic, a bijection between the elements of a pair of graphs. The complexity of exact graph matching has not yet been proven [30], but there are polynomial algorithms for solving the isomorphism problem of special types of graphs [12].

Inexact graph-matching algorithms provide a distance value that indicates graph dissimilarity. Different from the exact graph matching, the complexity of this problem has been proved to be NP-complete [30].

Graph-edit distance [31] is one of the most popular methods to perform inexact graph matching. Inspired by the traditional edit distance function that computes the similarity between two strings, the distance between a pair of graphs is defined as the minimum cost for converting one graph into another. This method is accurate, but it has an exponential time complexity [12]. Different Edit-Distance approaches propose sub-optimal edit cost with reduced computation time [32–34].

The use of traditional graph-matching methods to search and classify graphs on large datasets has severe limitations due to their high computational cost.

2.3. Graph embedding

The Vector Space Model (VSM) [35] is a well-known technique, commonly used in the context of text retrieval, that represents a document as a vector. The vector representation allows the computation of document similarity using different metrics, such as the cosine function and the Euclidean distance. Another advantage relies on the possibility of using indexing schemes to speedup search and classification tasks.

Therefore, an important research venue for handling large volumes of graphs relies on embedding them in the VSM. One of the first ideas to embed a graph relies on performing an eigen-decomposition of the adjacency or Laplacian matrix. Riesen and Bunke [36] proposed the use of graph kernels to map vectorial representations into dissimilarity spaces. Using the Graph-Edit Distance, a kernel calculates the distance between a sample graph to a prototype set, obtaining the distance to each graph in the prototype set. These distances are then used as the vectorial description of the sample graph. No special selection approach is used and all graphs from the training set are used as prototypes. Furthermore, traditional dimensionality reduction algorithms and data normalization procedures are used for creating the final vector representation. Riesen and Bunke [37] also proposed the use of Lipschitz mapping to graph embedding. This method describes a graph through n distances to predefined reference sets of graphs, which were defined by using a method based on the k-Medoids clustering. Hence, a graph is mapped to an n-dimensional real space by representing the edit distance of the graph to the n reference sets as a vector. In [38], the authors introduced an optimized dissimilarity space embedding. Their solution is based on a genetic-algorithm-based solution for estimating the distribution of dissimilarity values.

The methods proposed in [36–38] map the graphs into the vector space model, the same as the Bag of Graphs. However, they consider the distance to every graph of the training set, which is different from the Bag of Graphs as it considers the presence of local structures. The approach described in [36] also differs from the Bag-of-Graphs method as it selects all the training set as prototypes, while our method performs a clustering step to select the codewords.

2.4. Multi-graph classification

The method described in [39] addresses the object classification problem based on multiple graphs. Sets of graphs associated with positive, negative, and both positive and negative samples are combined with an efficient mining procedure to select discriminative candidate subgraphs. Later, subgraph features are selected to represent each bag using a binary encoding. The method proposed by Wu et al. [40] also addresses the multi-graph classification problem. The objective is to learn, based on a boosting mechanism, from sets of labeled bags of graphs. Subgraphs are selected to construct weak classifiers based on a dynamic weight adjustment approach at both bag and graph levels. One key aspect of the method relies on the generation of subgraph candidates based on their "informativeness." This approach is similar to the one employed in [39]. Also similar to what is proposed in [39], a bag constrained subgraph exploration step is used.

The approaches proposed in [41,42] investigate the use of bags of graphs in a positive and unlabeled multi-graph learning problem scenario. Similar to the above initiatives [39,40], a selection step is employed to determine the most informative subgraphs, which are later used to create a vector representation. Weights assigned to unlabeled bags are used to guide the identification of "reliable negative bags." Samples from positive and negative bags are then used to derive subgraph patterns, train classifiers, and update bag weights. A confidence weight value embedding approach is proposed to identify discriminative subgraph patterns to represent graphs in multi-graph bags for learning.

The approaches described in [43] and [44], in turn, address the problem of multi-graph-view for object classification. Each object is represented as bags of graphs collected from multiple graph views. The objective is to exploit complementary information provided by different views in order to create effective object classifiers. The method consists of three steps: an optimal subgraph exploration based on a multi-graph-view bag learning algorithm, a bag margin maximization procedure based on solving a linear programming problem, and update of bag and graph weights. These three steps are repeated until the algorithm converges.

Wu et al. [45] propose a dual embedding learning scheme based on both multiple instances and multiple graphs. The objective is to define classification models based on labeled bags containing both instances and graphs. Instance distributions are embed into the objective function so that the instance-based representation is consistent with selected subgraph features, while graph distributions are embed into the feature selection process. Based on the found optimal subgraphs and instance features, a concatenation strategy is employed to map bags into a new mixed feature space.

The concept of bag employed in [39–45] refers to a set of graphs. In our method, we used the term *bag* as it is employed in the information retrieval [1] and computer vision communities [2,46]: as a vector containing the distribution of frequency of occurrence of words (in our case, graph words defined in a vocabulary). The methods described in [39–42] consider that an object is represented based on multiple graphs and with each graph a vector representation is associated. These steps are similar to what we employ to create "attributed subgraphs." One key difference relies on the fact that, instead of performing classification based on the vectors associated with the multiple bags composed of multiple graphs, we use a single vector representation per object, i.e., one single bag per object. In our method, subgraph vectors are "combined" (in our case projected into a graph codebook) in order to generate the final vector representation. In this sense, our method is not targeted towards mapping the object classification task to a multi-graph classification problem. Our method performs object classification based on a single vector representation generated based on an offline-generated graph-based codebook. Finally, different from our work, the solutions presented in [39–42] are only targeted to binary classification problems.

2.5. BoW-based representations

Inspired by the VSM model, the BoW approach [1] represents a document through the distribution of frequency of occurrence of words. Since each document is represented as a collection of words, the vector representation is denominated a BoW. Barbu et al. [47] propose a *bag of symbols* to describe a graphical document. Hou et al. [48] propose the *Bag-of-Feature-Graphs*, an approach that describes a 3D-shape by a set of graphs. Karaman et al. [49] propose a BoW multi-layer approach (*Bag of Graph Words*) from the spatial distribution of interest points.

The Bag of Graph Words shares many ideas with our image classification example, but these two methods have different description, different codebook space, different quantization approaches, and different graph-matching mechanism. It is possible to describe the Bag of Graph Words as another concrete instantiation of the BoG theoretical framework described in this paper.

2.6. Bag of Visual Words

The BoW model applied to images [2] is known as BoVW, with applications in image classification and categorization [3,5,50], medical image screening [51], and image retrieval [52]. The bag

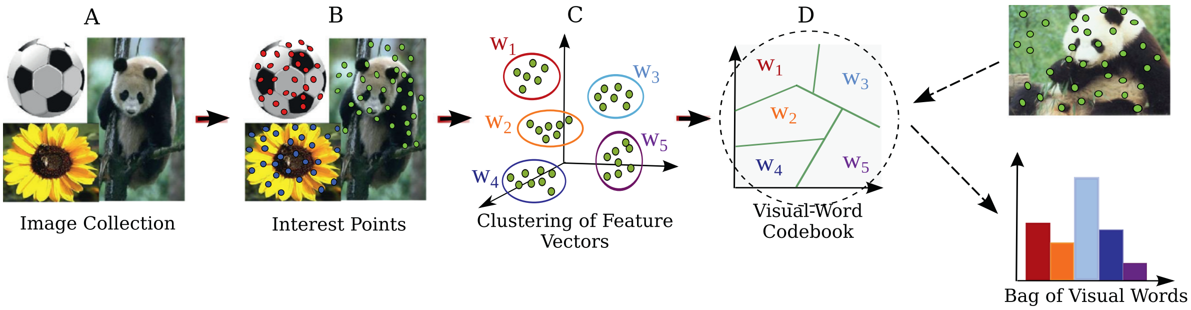

Fig. 2. Overview of the Bag of Visual Words. From a set of images (A), we detect interest points (B). Then, clustering the interest-point descriptors (C) can generate a codebook (D). Using this codebook, we compute the distribution of frequency of occurrence of visual words within an input image and create the corresponding BoVW descriptor.

model tends to be more discriminant than a global descriptor, and more general than a local descriptor.

In BoVW, there is a vocabulary of the main visual patterns of an image collection, sometimes also called *dictionary* or *codebook*, of *visual words*. Using a pre-defined visual codebook, the *bag of words* is created based on the occurrences of visual words within an image.

Regardless of the application domain, the process has the same sequence of steps: extraction, codebook, coding, and pooling; and differs from one application to another mostly in the definition of the vocabulary according to the intrinsic characteristics of each domain. Fig. 2 illustrates this process.

2.7. Encoding spatial relationships into BoW

Visual words do not possess semantic meaning, leading to the *semantic gap*. The semantic gap states that the visual similarity between a pair of images does not necessarily correspond to a semantic similarity. Thus, aggregating different types of information may help identifying correlations between the visual and the semantic contents of an image.

In Sivic et al. [2], Savarese et al. [53], and Hoàng et al. [4], the visual vocabulary is created based on the co-occurrence of groups of visual words. Sivic et al. [2] define a *doublet* as a pair of visual words that co-occur in a local area, and they propose to represent an image by the frequency of occurrence of *doublets*. Savarese et al. [53] define a codebook of correlograms of visual words, which is used for creating the bag representation. In Hoàng et al. [4], the spatial information is defined in terms of triangular structures. This approach, called Δ-TSR, computes the similarity of two images based on two aspects: the co-occurrence of visual word triplets and the geometric similarity of the corresponding triangles.

In Cao et al. [52], image regions are defined from linear and circular projections of interest points, and the image descriptor is created using a RankBoost algorithm that selects a combination of bags from different regions.

The Spatial Pyramids (SP) [3], one of the most famous BoW-based approaches, partitions hierarchically the image into cells. Each cell gets its own BoVW, and the final descriptor corresponds to the weighted concatenation of bags of the image cells.

The Word Spatial Arrangement (WSA) [5] divides the image into quadrants, considering each interest point as an origin for partitioning. Through these partitions, a histogram encapsulates the occurrence of visual words in each of the four relative positions.

In Sudderth et al. [54], a graphical model describes the visual appearances of interest points and their relative positions. In Niebles and Fei-fei [55], the BoW is combined with the *Constellation model* [56,57].

Bolovinou et al. [58] proposed the *Bag of Spatio-Visual Words* to represent the spatial information through correlograms of visual words. It defines a vocabulary of log-polar descriptors that encode the frequency of visual-word occurrences in regions of the image. In Liu and Caselles [59], strings corresponding to sequences of visual words within the neighborhood of interest-point describe the spatial arrangement of visual words. Zhou et al. [60] define vertical and horizontal regions on resolutions. Each image region is associated with a bag of visual words and the concatenation of the bags of all regions for an image resolution corresponds to an image descriptor.

3. Mathematical model

This section describes BoG's formal mathematical model, using some definitions or concepts introduced in [34,61,62].

3.1. Overview of the Bag-of-Graphs concepts

The BoG is a process that creates a vector representation from the local relationships within an object. Our model is defined by a composite function, denominated *bag extraction*, which combines *graph extraction*, *graph-of-interest detector* (GoI detector), *assignment*, *pooling*, and *feature extraction functions* for vertex, edge, and graph descriptors.

The *graph extraction* function extracts the intrinsic structure of a *digital object*. This structure's description is a graph that models the relationships among digital object elements (object components). The set of all components of an object is called power digital object.

A *GoI detector* function is then employed for detecting *graphs of interest* among all possible subgraphs (*power graph*) of the corresponding graph of an object, i.e., this function selects the subgraphs that represent relevant local structures within an object.

An *attributed graph* corresponds to a graph whose vertices and edges are described by features composed of simple and complex data types. The description of detected graphs is accomplished using three different types of descriptors: vertex descriptor, edge descriptor, and graph descriptor. A *vertex descriptor* comprises two functions: one that extracts features associated with vertices and a distance function that is used to compute the distance among different vertices given their features. The *edge descriptor* works similarly, except for the fact that it extracts features from edges. Finally, a *graph descriptor* combines both edge and vertex descriptors, allowing the computation of distances between graphs.

Using an *assignment* function, the object local structures are characterized in terms of the *words* of a *codebook*. These words correspond to the main patterns determined by *clustering* a set of graphs of interest extracted from a collection of objects. The final representation is created by a *pooling* function that summarizes the performed assignments.

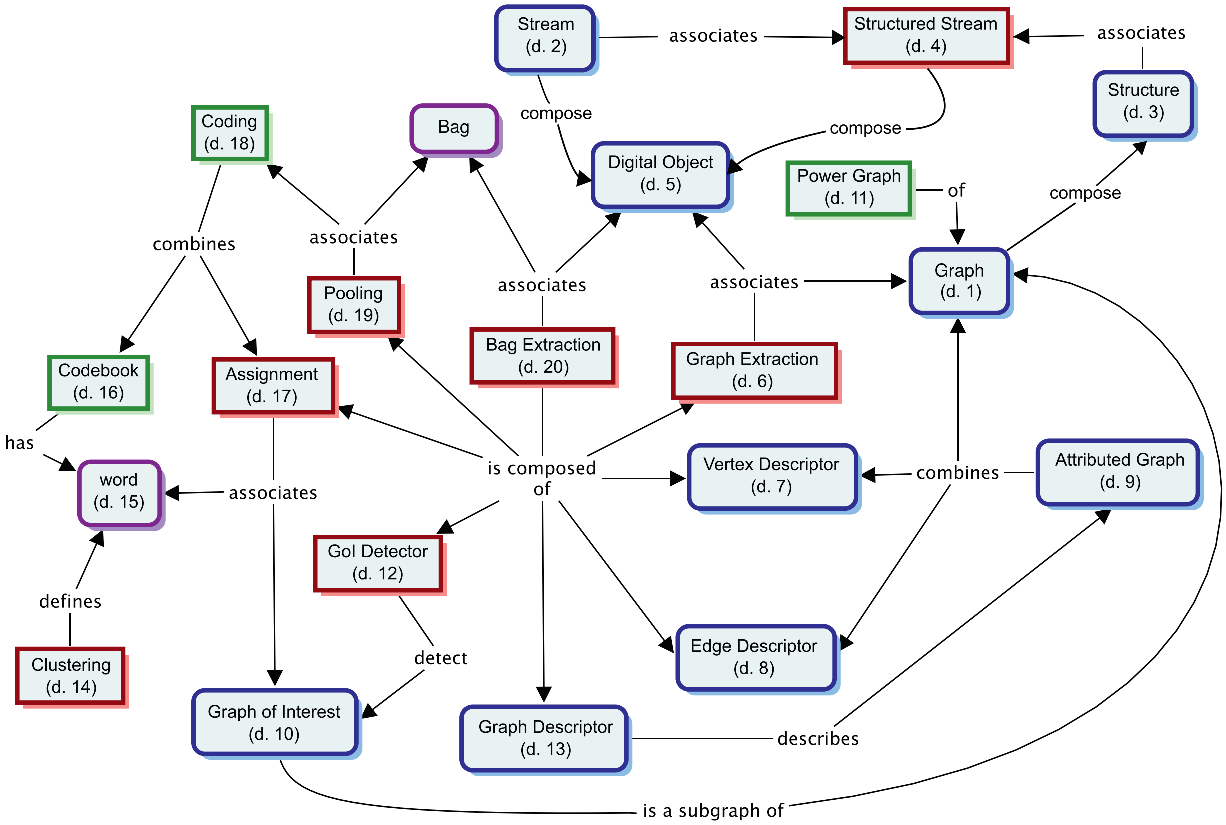

Fig. 3. Concept map of the Bag-of-Graphs model. The colors of the squares indicate the type of the concept: blue refers to the definition of particular tuples, red corresponds to function definitions, green refers to particular set definitions, and purple corresponds to specific representation elements. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3.2. Formal the Bag-of-Graphs model

In this section, we present the formal definition of the concepts related to the BoG model. Fig. 3 shows a map of the relationships between the concepts that will be introduced in this section.

Definition 1. A **graph** is a tuple $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of vertices, $\mathcal{E}$ is a set of edges. Each edge $e = (v_i, v_j)$ of $\mathcal{E}$ represents a link between the vertices v_i and v_j of $\mathcal{V}$.

The number of vertices, $|\mathcal{V}|$, is nominated *graph order* and the number of edges linked to a vertex is called *vertex degree*.

Definition 2. A **stream** is a sequence whose codomain is a nonempty set. A **sequence** is a function f whose domain is the set of natural numbers or some initial subset $1, 2, \ldots, n$ of the natural numbers and whose codomain is any set.

Definition 3. A **structure** is a tuple (G, L, F), where $G = (\mathcal{V}, \mathcal{E})$ is a directed graph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$, L is a set of label values, and F is a labeling function $F : (\mathcal{V}, \mathcal{E}) \rightarrow L$. As G is a **directed graph**, $\mathcal{E}$ is a set of edges (or arcs) where each edge is an ordered pair of distinct vertices (v_i, v_j), with $v_i, v_j \in \mathcal{V}$ and $v_i \neq v_j$.

Definition 4. Given a structure (G, L, F), $G = (\mathcal{V}, \mathcal{E})$ and a stream S, a **structured stream** is a function $\mathcal{V} \rightarrow (\mathbb{N} \times \mathbb{N})$ that associates each node $v_k \in \mathcal{V}$ with a pair of natural number (a, b), $a < b$ corresponding to a contiguous subsequence $[S_a, S_b]$ (segment) of the stream S.

Definition 5. A **digital object** is a tuple

$$DO = (h_{DO}, \mathcal{SM}, \mathcal{ST}, \mathcal{F}_{strStream}),$$

where h_{DO} is a set of universally unique handles (labels), $\mathcal{SM}$ is a set of streams, $\mathcal{ST}$ is a set of structural metadata specifications, i.e., a tuple composed of a graph, a set of literals and labels, and a set of functions that specifies the relationships among digital object components, and $\mathcal{F}_{strStreams}$ is a set of structured stream functions that associate a stream $s \in \mathcal{SM}$ with a structural metadata specification $m \in \mathcal{ST}$.

Definition 6. Let a graph $G = (\mathcal{V}, \mathcal{E})$, a **graph extraction**

$$\mathcal{P}(DO) \rightarrow (\mathcal{V} \bigcup \mathcal{E})$$

is a function that associates a digital object element of DO with a vertex of $\mathcal{V}$ or an edge of $\mathcal{E}$. The **power digital object**, denoted $\mathcal{P}(DO)$, is the set of all possible digital object elements of a given digital object DO. Fig. 4 illustrates an example of graph extraction function.

Definition 7. A **vertex descriptor** is a tuple

$$d_{\mathcal{V}} = (\epsilon_{\mathcal{V}}, \delta_{\mathcal{V}}),$$

where $\epsilon_{\mathcal{V}} : \mathcal{V} \rightarrow \mathcal{T}$ is a function that associates a vertex v of $\mathcal{V}$ with an element of $\mathcal{T}$, called a *vertex attribute*, and $\delta_{\mathcal{V}} : \mathcal{T} \times \mathcal{T} \rightarrow \mathbb{R}$ is a function that computes the similarity between a pair of vertices based on the distance, computed by a distance function of their corresponding attributes. Along with this manuscript, $\mathcal{T}$ is a set of node and edge attributes.

Definition 8. An **edge descriptor** is a tuple

$$d_{\mathcal{E}} = (\epsilon_{\mathcal{E}}, \delta_{\mathcal{E}}),$$

where $\epsilon_{\mathcal{E}} : \mathcal{E} \rightarrow \mathcal{T}$ is a function that associates an edge e of $\mathcal{E}$ with an element of $\mathcal{T}$, called an *edge attribute*, and $\delta_{\mathcal{E}} : \mathcal{T} \times \mathcal{T} \rightarrow \mathbb{R}$ is the similarity function between a pair of edges based on the distance of their attributes.

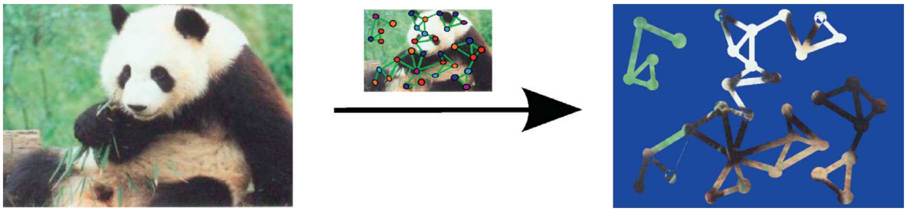

Fig. 4. Graph extraction function. A graph extraction function detects interest points on an image and builds the graph from their spatial relationships.

Definition 9. An **attributed graph** is a tuple

$\hat{G} = (G, \mathcal{D}_\mathcal{V}, \mathcal{D}_\mathcal{E})$,

where G is a graph, $\mathcal{D}_\mathcal{V}$ is a set of vertex descriptors, and $\mathcal{D}_\mathcal{E}$ is a set of edge descriptors.

Definition 10. A **graph of interest (GoI)** of a graph $G = (\mathcal{V}, \mathcal{E})$ is a subgraph $G' = (\mathcal{V}', \mathcal{E}')$ of G that satisfies a determined property P, such that $\mathcal{V}' \subset \mathcal{V}$ and $\mathcal{E}' \subset \mathcal{E}$.

Definition 11. The **power graph** of a graph G, denoted $\mathcal{P}(G)$, is the set of all possible subgraphs of G.

Definition 12. A **GoI detector** $\mathbb{D}$ is a characteristic function

$\mathbb{D} : \mathcal{P}(G) \rightarrow \{0, 1\}$

that determines if a subgraph of G is a GoI, i.e., verifies if a GoI satisfies a property P.

Definition 13. Let $\mathcal{G}$ be a set of attributed graphs and $\mathscr{T}$ be a set defined as the union of the domains of vertex and edge attributes, a **graph descriptor** is a tuple **(ϵ, δ)** where $\epsilon : \mathcal{G} \rightarrow \mathscr{T}$ is a function that associates an attributed graph with an element of $\mathscr{T}$ and $\delta : \mathscr{T} X \mathscr{T} \rightarrow \mathbb{R}$ is a function that computes the similarity between two attributed graphs. Both ϵ and δ are composite functions implemented based on vertex and edge descriptors of attributed graphs.

Definition 14. A **clustering** $\mathscr{C}$ of a set $\mathcal{S}$ is a partition on $\mathcal{S}$. The sets in $\mathscr{C}$ are the clusters.

Definition 15. Given a clustering $\mathscr{C}$, a **word** is an element $w \in \mathcal{T}$ (see Definition 7) that represents the prototype of an equivalent cluster defined by $\mathscr{C}$.

As example of Definition 15, the *centroids* of the clusters may be defined as words.

Definition 16. A **codebook**, or dictionary,

$\mathfrak{C} = \{w_1, w_2, \dots, w_{|\mathfrak{C}|}\}$

is a set of words representing each group defined by a clustering.

Definition 17. Let $\mathcal{G} = \{g_1, g_2, \dots, g_{|\mathcal{G}|}\}$ be a set of attributed graphs, and $\mathfrak{C} = \{w_1, w_2, \dots, w_{|\mathfrak{C}|}\}$ be a codebook. An **Assignment** is a function that defines an activation value for each pair (g_i, w_j), where $g_i \in \mathcal{G}$ and $w_j \in \mathfrak{C}$.

Two widely used assignment functions are *hard* and *soft* assignments. Using a *hard assignment* function, each $g_i \in \mathcal{G}$ activates only one word of $\mathfrak{C}$. The assignment function is

$$f_{assign} : \mathcal{G} \times \mathfrak{C} \rightarrow \{0, 1\}; \quad (1)$$

$$f_{assign}(g_i, w_j) = \begin{cases} 1 & \text{if } w_j = \underset{w_k \in \mathfrak{C}}{argmax}\, \delta(\epsilon(g_i), w_k) \\ 0 & otherwise \end{cases}. \quad (2)$$

Using a *soft assignment*, each $g_i \in \mathcal{G}$ is assigned to multiple words of $\mathfrak{C}$ with different activation levels. The assignment function

$f_{assign} : \mathcal{G} \times \mathfrak{C} \rightarrow [0, 1]$

is usually computed using a kernel function such as the one from [63]

$$f_{assign}(g_i, w_j) = \frac{K_\sigma(\delta(\epsilon(g_i), w_j))}{\sum_{k=1}^{|\mathfrak{C}|} K_\sigma(\delta(\epsilon(g_i), w_k))}, \quad (3)$$

where K_σ is a normalized Gaussian kernel

$$K_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}, \quad (4)$$

and σ^2 is its variance.

Definition 18. Let $\mathcal{G} = \{g_1, g_2, \dots, g_{|\mathcal{G}|}\}$ be a set of attributed graphs, $\mathfrak{C} = \{w_1, w_2, \dots, w_{|\mathfrak{C}|}\}$ be a codebook, and f_{assign} an assignment function. A **coding** is

$C = \{c_1, c_2, \dots, c_{|\mathcal{G}|}\}$,

where c_i is a vector that $c_i[j] = f_{assign}(g_i, w_j)$, where $g_i \in \mathcal{G}$ and $w_j \in \mathfrak{C}$, $1 \le i \le |\mathcal{G}|$, $1 \le j \le |\mathfrak{C}|$.

Definition 19. Given a coding C, **pooling**

$C \rightarrow \mathbb{R}^N$

is a function that summarizes all word assignments, defined in a coding C, into a numerical vector.

Let $\mathcal{G}$ be a set of attributed graphs and $C = \{c_1, c_2, \dots, c_{|\mathcal{G}|}\}$ be the corresponding coding defined according to a codebook $\mathfrak{C}$. We may chose different pooling functions to create a vector representation of $\mathcal{G}$. Some examples of typical implementations of pooling functions:

Sum pooling is

$$f_{sumpool}(C) = \left\{ \vec{v} \,\middle|\, \left(\forall k \in \left[1, |\mathfrak{C}|\right]\right) \left[\vec{v}_k = \sum_{i=0}^{i<|\mathcal{G}|} c_i[k]\right] \right\}, \quad (5)$$

average pooling is

$$f_{avpool}(C) = \left\{ \vec{v} \,\middle|\, \left(\forall k \in \left[1, |\mathfrak{C}|\right]\right) \left[\vec{v}_k = \frac{1}{|\mathcal{G}|} \sum_{i=0}^{i<|\mathcal{G}|} c_i[k]\right] \right\}, \quad (6)$$

and max pooling is

$$f_{maxpool}(C) = \left\{ \vec{v} \,\middle|\, \left(\forall k \in \left[1, |\mathfrak{C}|\right]\right) \left[\vec{v}_k = \underset{0<i<|\mathcal{G}|}{max} (c_i[k])\right] \right\}. \quad (7)$$

where v_k is a component of v.

Definition 20. Bag extraction is

$\mathcal{O} \rightarrow \mathbb{R}^N$,

a function that associates a digital object from a collection $\mathcal{O}$ with a vector in R^N. Let DO be a digital object of $\mathcal{O}$ and $\mathfrak{C}$ be a codebook, the bag extraction is defined as the composition of a *graph extraction* function, *vertex and edge descriptors* that create a graph $\hat{G}$ to represent DO, A *graph descriptor* is used for describing each GoI from $\hat{G}$.

Fig. 5 illustrates the concept flow of the BoG model and Fig. 6 of the bag extraction function.

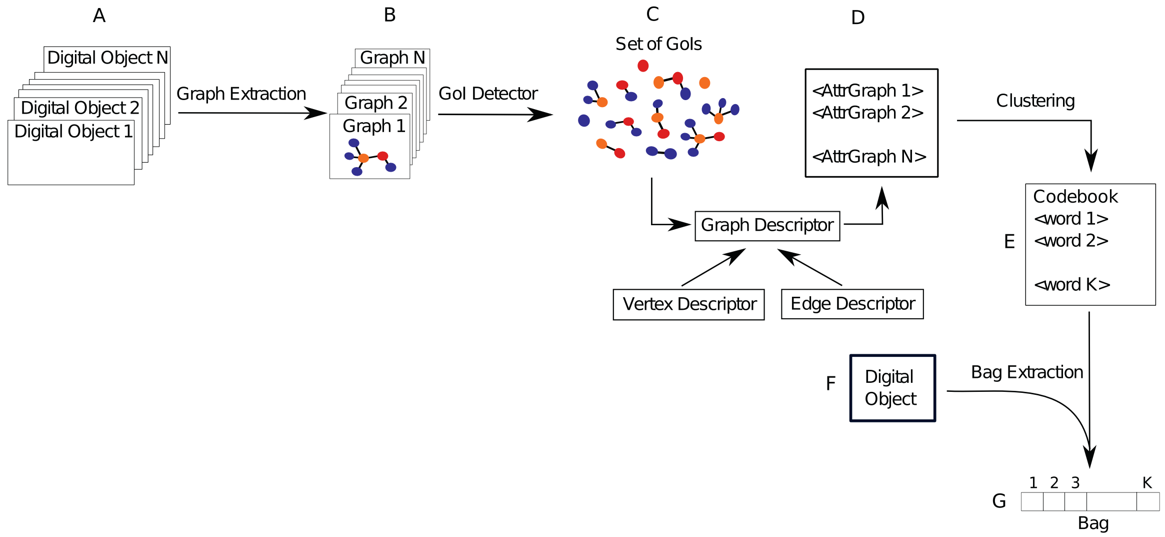

Fig. 5. Concept flow of the BoG model. From a set of digital objects (A), we extract graphs (B). Later, we detect graphs of interest (C) within these graphs and describe them (D). Next, graphs of interest are clustered, generating a codebook (E). Finally, an input digital object (F) is mapped to the codebook, generating a bag-of-graphs representation (G).

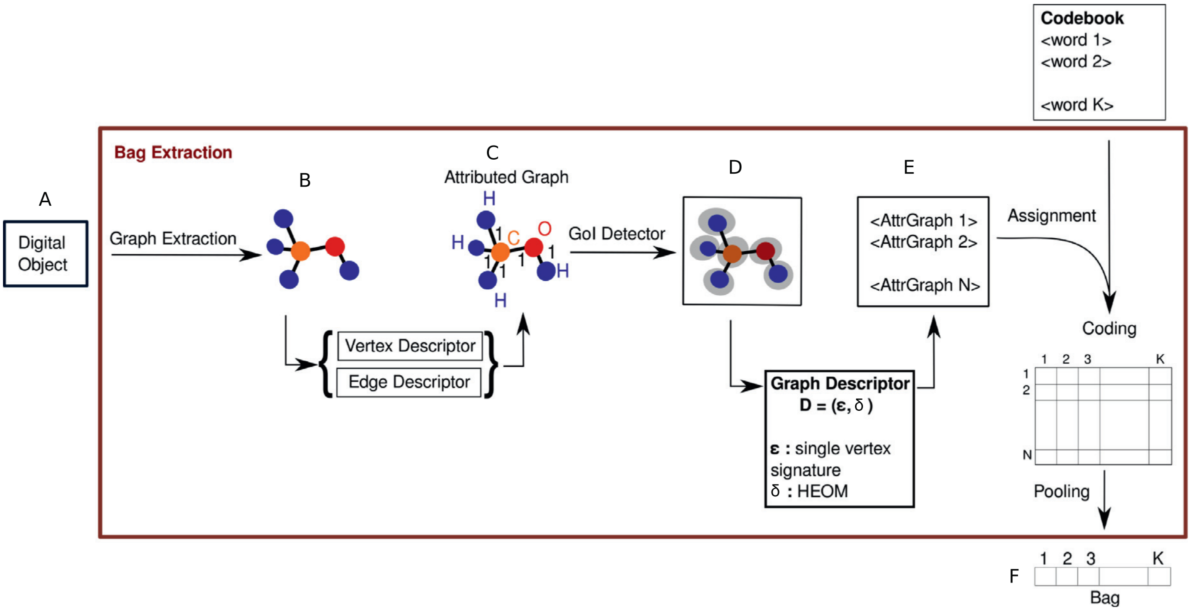

Fig. 6. Bag extraction function. Given an input digital object (A), we extract a graph (B) and describe it (C). Later, graphs of interests are determined (D) and characterized (E). Next, the attribute graph is mapped to the graph codebook and after coding and pooling procedures, a bag-of-graph representation is generated (F).

4. Use of the Bag-of-Graphs model

This section introduces two instantiations of the theory into real implementations of the BoG model.

4.1. Bag of Singleton Graphs

The Bag of Singleton Graphs (BoSG) is the first BoG approach to encode local patterns, describing objects already modeled as graphs. We use a molecule-description scenario to illustrate real realizations of the underlying concepts.

A molecule is a *digital object* that contains streams of atoms and chemical bounds in well defined spatial relation, the molecule's geometry. These spatial relationships justify the use of a graph framework.

From a collection of molecules, a graph extraction function, and vertex and edge descriptors, we find a set of attributed graphs $\mathcal{G}$. Each attributed graph from $\mathcal{G}$ is

$$\hat{G} = ((\mathcal{V}, \mathcal{E}), \{chem\}, \{valence\}),$$

where

chem is a vertex descriptor $(\epsilon_{chem}, \delta_{chem})$, the function $\epsilon_{chem} : \mathcal{V} \rightarrow \mathcal{A}$ associates a vertex of $\mathcal{V}$ with an atom symbol, and $\delta_{chem} : \mathcal{A} \times \mathcal{A} \rightarrow \mathbb{R}$ is the discrete distance function. $\mathcal{A}$ is a set of strings that identify atoms, such as "C", "H", and "O".

valence is an edge descriptor $(\epsilon_{valence}, \delta_{valence})$, where the function $\epsilon_{valence} : \mathcal{E} \rightarrow \mathbb{N}$ associates an edge of $\mathcal{E}$ with a number of valence, and the function $\delta_{valence} : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{R}$ computes the absolute difference between edge attributes.

The BoSG represents a graph as the frequency of occurrences of its local structures, the GoIs. A GoI detector $\mathbb{D}$ extracts a set of GoI from an attributed graph $\hat{G}$.

Definition 21. Let $\hat{G} = ((\mathcal{V}, \mathcal{E}), \mathcal{D}_v, \mathcal{D}_e)$ be an attributed graph and N be the vertex $v \in \mathcal{V}$ and its incident edges, a **vertex signature** of v is a sequence of elements of node and edge attributes associated with the components of N, composed of the vertex attributes of v, the vertex degree, and the attributes of the edges linked to v.

A GoI, identified by applying $\mathbb{D}$ on a graph

$\hat{G} = ((\mathcal{V}, \mathcal{E}), \{chem\}, \{valence\})$,

is a subgraph of $\hat{G}$ composed of a vertex $v \in \mathcal{V}$, the adjacent vertices of v on $\hat{G}$ and the edges of $\mathcal{E}$ that link v to another vertex of $\mathcal{V}$.

The set of GoIs $\mathbb{G}$ extracted from an attributed graph $\hat{G} \in \mathcal{G}$ represents the vertex neighborhoods of $\hat{G}$. Let $\mathscr{V}$ be a set of vertex signatures, a graph descriptor $\mathscr{D} = (\epsilon, \delta)$ is defined as follows:

- ϵ is a function $\mathbb{G} \to \mathscr{V}$ that associates an attributed graph $g \in \mathbb{G}$ with a single vertex signature $\mathscr{S} \in \mathscr{V}$. Since g represents the neighborhood of a vertex v, $\mathscr{S}$ corresponds to the vertex signature of v, which is composed of the vertex attributes of v, the vertex degree and the attributes of the edges linked to v.
  For each vertex signature $\mathscr{S}$, the edges are sorted by their attribute values. Let $L_i = [AE_{i,1}, AE_{i,2}, \ldots, AE_{i,D}]$ be the sequence of D edge attributes of the edge e_i. The edges are sorted in order of increasing values of AE_k. If two edges have the same value for an attribute AE_k, their order is determined by the values of the following attribute AE_{k+1} in L.
- $\delta : \mathscr{T} X \mathscr{T} \to \mathbb{R}$ is a function that computes the similarity between two vertex signatures obtained based on the similarity values obtained from vertex and edge descriptors. We use the Heterogeneous Euclidean-Overlap Metric (HEOM) [64] to implement δ.

The HEOM [64] is a heterogeneous distance function as it can handle linear and nominal attributes, using the *overlap* metric for nominal attributes and normalized Euclidean distance for linear attributes.

Given two heterogeneous vectors **x** and **y**

$$HEOM(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{k=1}^{N} \delta(\mathbf{x}_k, \mathbf{y}_k)^2} \tag{8}$$

$$\delta(\mathbf{x}_k, \mathbf{y}_k) = \begin{cases} \frac{|\mathbf{x}_k - \mathbf{y}_k|}{range_k} & \text{if } \mathbf{x}_k \text{ is linear attribute,} \\ \tau(\mathbf{x}_k, \mathbf{y}_k) & \text{if } \mathbf{x}_k \text{ is nominal attribute,} \\ 1 & \text{if } \mathbf{x}_k \text{ or } \mathbf{y}_k \text{ is missing} \end{cases} \tag{9}$$

$$\tau(\mathbf{x}_k, \mathbf{y}_k) = \begin{cases} 0 & \text{if } \mathbf{x}_k = \mathbf{y}_k, \\ 1 & \text{otherwise} \end{cases} \tag{10}$$

Let $g_i = ((\mathcal{V}, \mathcal{E}), \{\text{chem}\}, \{\text{valence}\})$ be the graph representing the neighborhood of a vertex $v_i \in \mathcal{V}$ and e_{ij} be an edge of $\mathcal{E}$, the functions ϵ and δ are

$$\epsilon(g_i) = \langle \epsilon_{chem}(v_i),\ degree_{v_i},\ \epsilon_{valence}(e_{i1}), \\ \epsilon_{valence}(e_{i2}), \ldots,\ \epsilon_{valence}(e_{in}) \rangle$$

and

$$\delta(\epsilon(g_1), \epsilon(g_2)) = ((\delta_{chem}(v_1, v_2))^2 + (S_e(\epsilon(g_1), \\ \epsilon(g_1)))^2 + (P_e(g_1, g_2))^2)^{\frac{1}{2}}, \tag{11}$$

where

$$S_e(\epsilon(g_1), \epsilon(g_2)) = \sum_{i=1}^{\min_{\{v_1,v_2\}}(degree)} \frac{(\delta_{valence}(e_{1i}, e_{2i}))^2}{max(\delta_{valence})} \tag{12}$$

$$P_e(g_1, g_2) = \sum_{i=\min_{\{v_1,v_2\}}(degree)}^{\max_{\{v_1,v_2\}}(degree)} 1 \tag{13}$$

Fig. 1 presents an overview of the main steps used to extract Bags of Singleton Graphs. Let $\mathscr{G}$ be the set of all GoIs (set labeled as A in the figure) extracted from attributed graphs in $\mathcal{G}$, defined as $\mathscr{G} = \bigcup_{i=0}^{i<|\mathcal{G}|} \mathbb{G}_i$, and $\mathbb{S}_G$ be the set of vertex signatures s_i associated with $\mathbb{G}_i \in \mathscr{G}$. s_i is computed based on the vertex and edge descriptors defined for each $\mathbb{G}_i$. We use a clustering relation on $\mathbb{S}_G$ to create a codebook $\mathfrak{C}$ (B and C in the figure), whose words correspond to vertex signatures that represent the main graph local structures within $\mathscr{G}$.

Let $\mathcal{Q}$ be a set of GoIs extracted from an attributed query graph Q (D in the figure) and $\mathbb{S}_Q$ be the corresponding set of vertex signatures obtained. Given the codebook $\mathfrak{C}$ defined previously, different coding and pooling functions can be used to create a *bag of singleton graphs* (E in the figure) that represent the digital object related to Q.

In the case of molecule representation, the number of vertices of the corresponding attributed graph is a relevant information that should be encoded into the graph representation. Therefore, the bags are generated using hard assignment and sum pooling functions.

The proposed method has some advantages over approaches based on complex graph matching procedures from the literature [30]. Since we represent graphs by feature vectors, simple distance functions, like the Euclidean distance, may be used for calculating the similarity of graphs. Therefore, our method is very fast for computing graph matching. In fact, different from traditional approaches, the complexity of the similarity between graphs does not depend on the number of vertices once the BoSG has already been calculated.

Besides, the methods based on the edit distance approach usually require the search of the optimal combination of parameters. The Bipartite Graph Matching [33] requires three parameters related to the cost of edit operations on vertices and edges, while, for example, BoSG requires one parameter: the codebook size.

4.2. Bag of Visual Graphs

In several applications, the semantics associated with the content of an image is perceived in terms of the spatial distribution of visual properties. A known limitation of the BoW model relies on its inability of encoding the spatial distribution of visual words within an image. In this section, we introduce the **Bag of Visual Graphs (BoVG)**, a BoG-based approach that uses graphs for encoding the spatial relationships among visual patterns into the image representation.

Our approach combines the spatial locations of interest points and their labels defined in terms of a traditional visual-word codebook. We also define a second vocabulary, the *visual-graph codebook*, which contains the main spatial relationships of visual words. In the following sections, we use the BoG model to create both visual dictionaries and the final image descriptor.

Fig. 7 summarizes the steps for generating both the visual-graph codebook, and the final image descriptor.

4.2.1. Visual-word codebook

This section describes how to create a traditional visual-word codebook. An interest-point detector, such as the *Hessian Affine*, is a graph extraction function that associates an image I with a graph $G = (\mathcal{V}, \mathcal{E})$, where a vertex $v \in \mathcal{V}$ corresponds to an interest point and $\mathcal{E}$ is an empty set.

We can use interest-point descriptors as the vertex descriptors that describe the visual content of an image. An image I would be represented by an attributed graph $\hat{G} = ((\mathcal{V}, \mathcal{E}), \{SIFT\}, \emptyset)$, where *SIFT* is a vertex descriptor defined as $(\epsilon_{sift}, \delta_{sift})$, such that $\epsilon_{sift} : \mathcal{V} \to \mathbb{N}^{\mathrm{N}}$ is a function that associates a vertex with a feature vector [65], and $\delta_{sift} : \mathbb{N}^{\mathrm{N}} \times \mathbb{N}^{\mathrm{N}} \to \mathbb{R}$ is a function that computes the

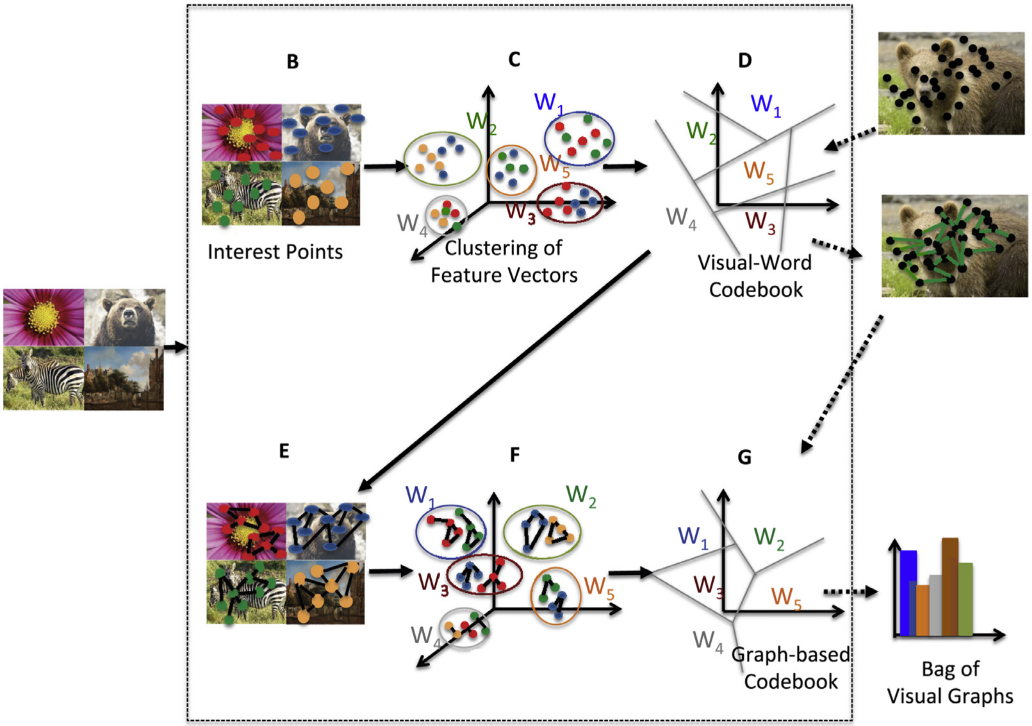

Fig. 7. Overview of the Bag of Visual Graphs. From an image collection (A), we detect all interest points (B). We cluster the descriptors of the interest points in feature space (C), and generate the *visual-word* codebook (D) from the prototypes of the clusters. Using this codebook and a Delaunay triangulation on the interest points of each image, we build a set of connected graphs (E) to represent the image, which encodes the spatial relationships of visual words. In a new clustering step (F), we select the words of the new vocabulary (G), the *visual graphs*. The *Bags of Visual Graphs* descriptor of an image uses the graph-based codebook (G) to compute a histogram, which counts the frequency of the *visual graphs* within the image.

similarity between two interest points using the Euclidean distance between their feature vectors.

Each interest point of an image corresponds to a relevant local information defined as a GoI. We use a GoI detector $\mathbb{D}$ that identifies the subgraphs of $\hat{G}$ composed of a single vertex as graphs of interest.

Let $\mathcal{G}$ be the set of attributed graphs extracted from a set of images $\mathcal{I}$ and $\mathbb{G}$ be the set of GoIs detected on $\mathcal{G}$ with $\mathbb{D}$. The attributed graphs of $\mathbb{G}$ are described with ϵ_{sift} function, generating a set of feature vectors $\mathcal{F}$ that characterizes $\mathbb{G}$. Clustering can use $\mathcal{F}$ to partition $\mathbb{G}$ with respect to the visual similarity of GoIs: from the clusters on $\mathcal{F}$, we create a codebook $\mathfrak{C}$ composed of *visual words*, which represent the main visual patterns within $\mathcal{I}$.

4.2.2. Encoding spatial relationships into BoVW

In this section, we present the process of generating the graph-based codebook and the proposed image representation, the *bag of visual graphs*.

In the proposed BoVG approach, the local structures of an image are defined in terms of visual patterns and their spatial locations. Thus, we apply a graph extraction function that associates an image I with a *weighted graph* $G = (\mathcal{V}, \mathcal{E}, \phi)$, where a vertex $v \in \mathcal{V}$ corresponds to an interest point, each edge $e \in \mathcal{E}$ encodes a spatial relationship between interest points, and ϕ is a function $\mathcal{E} \to \mathbb{R}$ that defines an edge weight based on the distance between connected vertices; the higher the distance, the higher the weight.

Let $\mathcal{P}(I)$ be the power digital object of I, we apply the graph extraction function $f_{delaunay} : \mathcal{P}(I) \to \mathcal{V} \bigcup \mathcal{E}$ that specifies that the vertices of V correspond to interest points, and that edges of $\mathcal{E}$ are defined by applying a Delaunay Triangulation on $\mathcal{V}$.

Similar to [66], edges are pruned based on their weights. Edges with low weights are removed because they encode relationships between close points, and are not useful to understand spatial arrangements of visual cues. Edges with high weights are also removed as they are associated with non-local structures. Usually, these constraints are defined empirically based on image dimensions, and relaxed in case there are not enough interest points.

We aim at an image representation that captures the spatial relationships of visual words. Thereby, to describe the image graphs extracted with $f_{delaunay}$, we describe vertices with visual words, and edges with texture-based signatures.

Defining vertex and edge descriptors, the image I is associated with an attributed graph $\hat{G} = (G, \{\text{VW}\}, \{\text{LBP}\})$,

- $G = (\mathcal{V}, \mathcal{E})$ is a graph defined under the graph extraction function.
- VW is a vertex descriptor defined as $(\epsilon_{vw}, \delta_{vw})$, where
  - Let $\mathfrak{C}$ be the visual codebook previously introduced, ϵ_{vw} is a composite function that combines an assignment and a labeling function. First, a hard assignment function is employed to associate each vertex $v \in \mathcal{V}$ with a visual word of $\mathfrak{C}$. Then, a labeling function associates v with the corresponding label of the assigned visual word.
  - $\delta_{vw} : L \times L \to \mathbb{R}$ is a function that determines the similarity between two vertices based on the labels assigned to vertices. The similarity value is computed through the use of the Discrete Distance function:

$$d(x, y) = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{if } x \neq y. \end{cases} \tag{14}$$

- LBP [67] is an edge descriptor defined as $(\epsilon_{lbp}, \delta_{lbp})$,
  - ϵ_{lbp} is a function $\mathcal{E} \to \mathbb{R}^{N}$ that associates each edge $e \in \mathcal{E}$ with a feature vector $\vec{fv}$. Let the local brightness variations be represented as binary patterns, $\vec{fv}$ represents the distribution of binary patterns within the region delimited by the connected vertices of e.

- $\delta_{lbp} : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}$ is a determines the similarity between two edges using the Manhattan Distance between their feature vectors.

Since images may be represented by attributed graphs, the BoG model can be used for generating bag representations that describe images based on the distribution of the spatial relationships of visual words.

Here, given an attributed graph $\hat{G}$ associated with a digital object $I \in \mathcal{I}$, the local structures of I are represented by graphs of interest detected with $\mathbb{D}_\triangle$, a GoI detector that identifies the connected subgraphs, whose vertices belong to a triangle defined under a Delaunay Triangulation. The detected GoIs on $\hat{G}$ are graphs with at most three vertices.

Let $\mathcal{S}$ be a set of vertex signatures (Definition 21). The set of graphs of interest $\mathbb{G}_\triangle$, obtained by applying $f_{delaunay}$ followed by $\mathbb{D}_\triangle$ on $\mathcal{I}$, is a graph descriptor $\mathscr{D} = (\epsilon, \delta)$,

- ϵ is a function $\mathbb{G}_\triangle \rightarrow \mathcal{S}^N$ that associates an attributed graph $g_i = ((\mathcal{V}_i, \mathcal{E}_i), \{\text{VW}\}, \{\text{LBP}\})$ with an array comprising its vertex signatures. In this section, a vertex signature is

  $$S(v_i) = \langle \epsilon_{VW}(v_i),\ degree_{v_i},\ \epsilon_{LBP}(e_{i1}),\\ \epsilon_{LBP}(e_{i2}), \ldots,\ \epsilon_{LBP}(e_{in}) \rangle,$$

  where $v_i \in \mathcal{V}_i$ and $e_{ij} \in \mathcal{E}_i$.
- $\delta : \mathcal{S}^N X \mathcal{S}^N \rightarrow \mathbb{R}$ computes the similarity between two graphs, as proposed by Jouili et al. [34],

  $$\delta(\epsilon(g_1), \epsilon(g_2)) = \frac{\bar{C}}{|C|} + ||g_1| - |g_2||, \tag{15}$$

  where $|g_i|$ is the order of graph g_i, $\bar{C}$ is the optimum graph matching cost and $|C|$ is a normalization constant that refers to the number of matching vertices.

The *Hungarian method* [68], a polynomial-time algorithm for the assignment problem, computes the optimum matching cost of a pair of graphs. Given a matrix M, where each element corresponds to the cost of assigning a job (column) to a worker (row), the Hungarian method finds the minimum cost for assigning jobs to workers in M.

In the proposed method, the Hungarian method is applied on two distance matrices C_1 and C_2. Each element of both matrices corresponds to the distance between a vertex of graph g_1 and a vertex of graph g_2, which is computed with a similarity function $\delta : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}$ (Eq. (16)) that computes the balanced sum of all similarity values obtained for each term of the vertex signatures. The sum is balanced so that all terms have the same weight (importance), so all similarity values are normalized in the range [0, 1].

Let $S(v_1), S(v_2) \in \mathcal{S}$,

$$\delta(S(v_1), S(v_2)) = \delta_{VW}(v_{i1}, v_{i2}) + S_e(S(v_1), S(v_2)) + \\ P_e(v_1, v_2), \tag{16}$$

where P_e is defined in Eq. (13), and

$$S_e(S(v_1), S(v_2)) = \sum_{i=1}^{\min_{\{v_1, v_2\}}(degree)} \frac{\delta_{LBP}(e_{1i}, e_{2i})}{|\epsilon_{lbp}(e_i)|} \tag{17}$$

The matrices C_1 and C_2 differ in how the distance between vertex signatures is computed. In C_1, the function δ considers that the sequence of edge attributes is defined with respect to counterclockwise direction of vertices. In C_2, the function δ considers that the sequence of edge attributes is defined using opposite directions on each graph. For the vertex signatures related to g_1, edges attributes are set respecting the counterclockwise direction of vertices, while edges attributes of g_2 are set in clockwise direction.

Given the matrices C_1 and C_2, the optimum matching cost is defined as

$$\bar{C} = \min(\bar{C_1},\ \bar{C_2}),$$

where $\bar{C_i}$ corresponds to the result of the Hungarian Method on matrix C_i, which handles reflection transformations.

Let $\mathscr{S}$ be the set of vertex signature arrays that describes $\mathbb{G}_\triangle$. We propose a second vocabulary, the *visual-graph codebook*, that quantizes, through a clustering relation on $\mathscr{S}$, the graph space defined by $\mathbb{G}_\triangle$. A word in this codebook, named as *visual graph*, refers to a group of similar visual words. We can use clustering methods [12,69,70] or a simple random selection to create the graph-based codebook.

Given a query image I_Q, a set of vector signature arrays $\mathscr{S}_I$ is extracted from I_Q by repeating the procedure for $\mathscr{S}$. Using different approaches of coding and pooling with the *visual-graph codebook*, a *bag of visual graphs* can be created to represent I_Q.

5. Validation

In this section, we validate and compare the proposed Bag of Graphs (Bog) to the existing literature. Due to their inherent differences, each approach has different validation requirements and analysis.

5.1. Bag of Singleton Graphs

The Bag of Singleton Graphs (BoSG), in Section 4.1, creates a bag representation based on the local structures of a graph. The method maps a feature vector for each graph and reduces the graph-matching problem to a feature-vector similarity. Different distance metrics, such as Euclidean, Manhattan, or Earth Mover's, can play this role with, each with distinct properties.

5.1.1. Experimental protocol

We estimated the size of the codebook using the Mean Shift unsupervised learning algorithm [71], and evaluated the method with two different codebook construction approaches, the *BoSG (random)* using a random selection, and *BoSG (Mean Shift)* using the Mean Shift algorithm itself for clustering. In all experiments, we used *hard assignment* and *sum pooling* to generate BoSG representations and the Euclidean distance to compare the feature vectors. We used the IAM repository protocols, which provides pre-defined training, validation, and test sets for the graph-classification problem.

5.1.2. Datasets

We validate the BoSG on four online available graph datasets from the IAM Repository[2] [6]: GREC, Mutagenicity, AIDS, and Letter (LOW). These datasets contain attributed graphs that represent different types of objects, such as letters, molecules, and symbols. Table 1 summarizes some characteristics of these datasets.

5.1.3. Literature baseline

The BoSG provides a measure of similarity between graphs, and we compare it against two graph edit distances, from the literature: the Bipartite Graph Matching [33] and the Attributed Graph Matching [34]. Both approaches use the Hungarian method – a polynomial solution for the assignment problem. The Bipartite-Graph approach uses a specific implementation for each dataset, while the Attributed-Graph approach and our method use a unique implementation for all datasets.

[2] http://www.iam.unibe.ch/fki/databases/iam-graph-database (As of May 2017).

Table 1
Number of vertices and classes for each graph dataset and number of graphs in each classification set.

	GREC	Mutagenicity	AIDS	Letter
Mean number of vertices	11.5	30.3	15.7	4.7
Max number of vertices	25	417	95	8
Number of classes	22	2	2	15
Size of training set	284	1500	250	750
Size of validation set	286	500	250	750
Size of test set	527	2337	1500	750

Table 3
Average time spent by the proposed method and different baselines to construct a graph distance matrix.

	GREC (s)	Mutagen. (s)	AIDS (s)	Letter (s)
BoSG	**0.11 ± 0.02**	**2.9 ± 0.1**	**0.29 ± 0.06**	**0.384 ± 0.003**
Riesen	262 ± 4	65,430 ± 2825	616 ± 21	101 ± 6
Jouili	327 ± 19	16,668 ± 124	1558 ± 37	773 ± 16

Table 4
Relative time spent by baselines in relation to BoSG method.

	GREC (s)	Mutagen. (s)	AIDS (s)	Letter (s)
BoSG	1	1	1	1
Riesen	2382	22,562	2124	263
Jouili	2973	5748	5372	2013

Table 5
Offline time spent in each step of the BoSG approach for each dataset.

	GREC (s)	Mutagen. (s)	AIDS (s)	Letter (s)
Parser graphs	4.2 ± 0.9	10 ± 4	11.1 ± 0.8	2.1 ± 0.2
Codebook	596 ± 14	4753 ± 126	1308 ± 4	97 ± 4
Build bags	9 ± 6	22 ± 2	11 ± 6	9 ± 6
Total	605 ± 18	4795 ± 126	1330 ± 7	109 ± 10

5.1.4. Evaluation measures

The *effectiveness* of a method is measured in terms of accuracy. A K-Nearest Neighbor (KNN) algorithm classifies graphs of a test set. Then, the accuracy rate is computed in order to obtain the number of graphs correctly classified with (KNN). In the case of the Riesen's approach [33], the validation sets are used for finding the best parameters for each dataset.

Here, the *efficiency* of a method is related to the time spent for computing the graph distance matrix on an Intel Xeon CPU E5645 2.40GHz with 16GB of RAM. We ran each method five times for each dataset and measured the execution times in seconds.

5.1.5. Results and discussion

Classification accuracy. For the evaluation of *BoSG (random)* approach, we generated five codebooks and the bags to assess the invariance of the representation to different seeds. Tables 2(a) to (d) show the average and standard deviation of the accuracy. The Mutagenicity dataset has a very large number of node signatures in the training set, therefore, the Mean Shift codebook construction used only a subset of the training samples.

Tables 2(a)–(d) present the accuracy results for each dataset using the (KNN) classifier with the parameter $K = \{1, 3, 5\}$ and show that our method achieves comparable accuracy performance in relation to Riesen's and Jouili's approaches. Riesen's approach achieved the highest accuracy on three of four datasets, but it uses a specific implementation for the computation of the edit costs and it requires a search for the optimal combination of three parameters. Therefore, Riesen's approach obtains the best results, but it requires a large amount of time and effort for setting up the method for each dataset.

Execution time. Table 3 contains the average time spent by each algorithm to construct a graph distance matrix. Regarding our method, we considered the creation of bags as an offline phase. Therefore, the values of BoSG's row on Table 3 refer only to the time for computing the distances between graph bags, not including the time to generate these bags. As it can be observed, BoSG has a much better performance when compared with all baselines in terms of execution time.

In order to show the relative improvement for the execution time, Table 4 presents the relative times of Riesen's and Jouili's approaches in relation to BoSG method. For each pair method-dataset, this table shows how many times BoSG was faster than the corresponding method on a given dataset.

Fig. 8 summarizes the results of evaluated methods in terms of both their accuracy rates and execution time. The points correspondent to BoSG results are placed in the superior left corner, which highlights its efficiency. It yields high accuracy rates with very low computational costs.

Table 5 shows the offline time spent for running BoSG approach, including the time spent for generating the Mean-Shift codebook and the feature vectors that correspond to the bags. We do not show the random codebook creation time, as it is insignificant in comparison.

The offline time depends on the number of graphs in the dataset and the Mean Shift parameter. The values on the Codebook line of Table 5 correspond to the time required to generate the four codebooks (same Mean Shift parameters used to obtain the results of Tables 2(a)–(d), 0.05, 0.05, 0.3, and 0.01).

Impact of the codebook size. In the Mean Shift algorithm [72], the size of the codebook is influenced by the kernel bandwidth, which is determined based on the pairwise distances between training samples. The parameter used to specify the percentage of distances to be considered when calculating the bandwidth has a default value of 0.3. The reduction of this parameter value causes a

Table 2
Accuracy results of the proposed method and different baselines for each dataset, using the (KNN) classifier.

(a) GREC dataset

K	BoSG	BoSG (rand)	Riesen	Jouili
1	0.934	0.969 ± 0.007	**0.983**	0.981
3	0.896	0.947 ± 0.007	**0.983**	0.975
5	0.860	0.92 ± 0.01	**0.985**	0.960

(b) Mutagenicity dataset

K	BoSG	BoSG (rand)	Riesen	Jouili
1	0.690	0.672 ± 0.008	**0.695**	0.652
3	0.703	0.681 ± 0.008	**0.720**	0.663
5	0.713	0.69 ± 0.01	**0.719**	0.652

(c) AIDS dataset

K	BoSG	BoSG (rand)	Riesen	Jouili
1	0.989	0.977 ± 0.003	0.993	**0.995**
3	0.991	0.970 ± 0.006	0.990	**0.997**
5	0.985	0.959 ± 0.006	0.984	**0.996**

(d) Letter dataset

K	BoSG	BoSG (rand)	Riesen	Jouili
1	0.945	0.89 ± 0.01	**0.989**	0.920
3	0.948	0.89 ± 0.02	**0.991**	0.909
5	0.949	0.88 ± 0.02	**0.993**	0.895

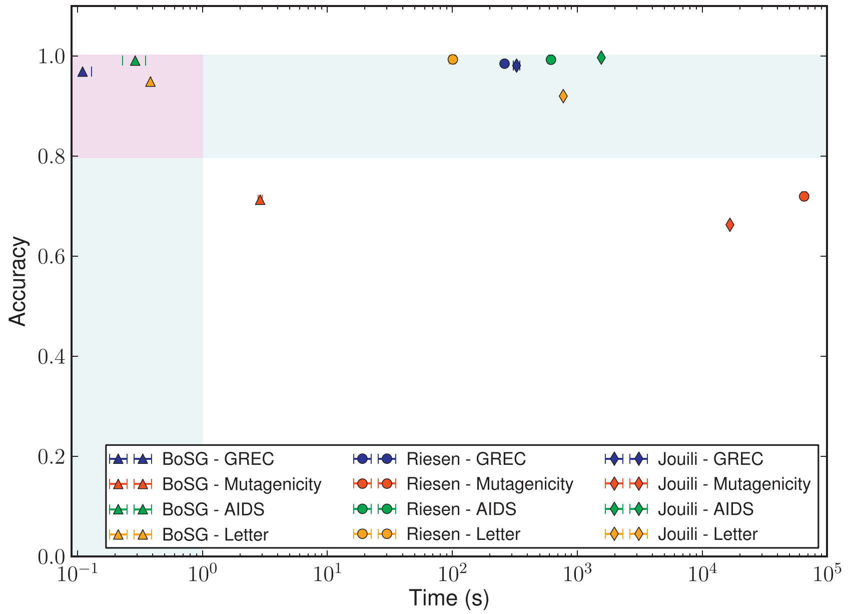

Fig. 8. Accuracy rates with respect to the execution time for different methods and datasets. Each method is identified by a marker and each dataset is identified by a color. The light-green bands indicate the areas of the highest accuracy rates or the lowest execution times, and the light-purple intersection of these bands indicates the area where the best results considering both accuracy and execution time are placed. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

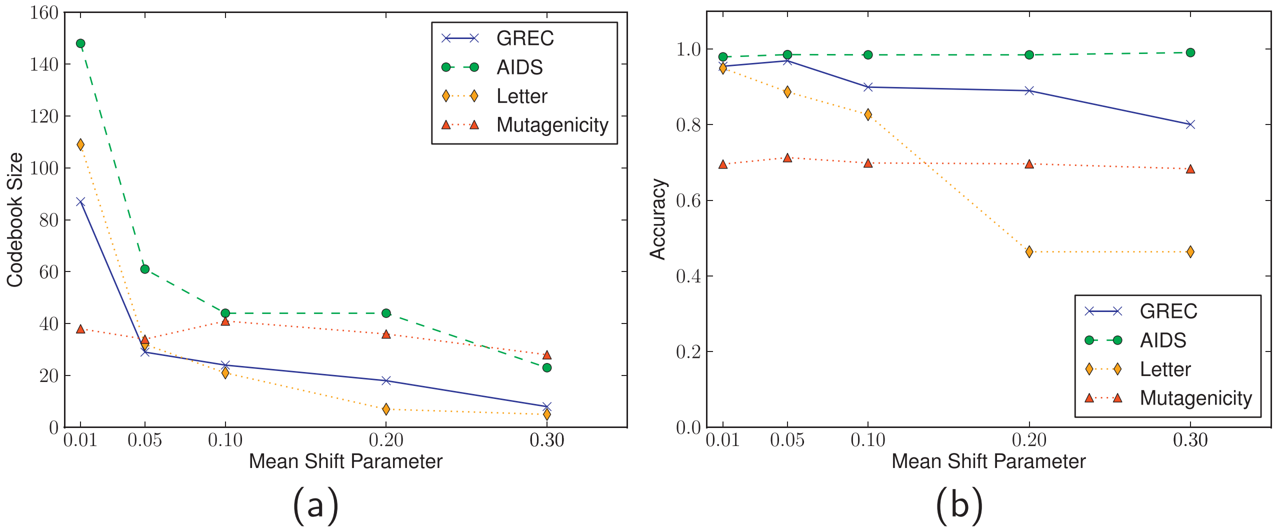

Fig. 9. Effect of kernel bandwidth on BoG. Study of the Mean Shift kernel bandwidth parameter on BoG regarding the codebook size (a) and its performance (b).

reduction of the bandwidth, which contributes to increase the size of codebook. Fig. 9(a) and (b) show the variation of the codebook size and the performance of BoSG approach using different values for the Mean Shift parameter.

When we use a low value, the kernel bandwidth is reduced, which contributes to create small clusters. In order to cover all data, the number of clusters tends to increase, resulting in larger codebooks.

The size of the codebook is directly related to the vocabulary diversity. The use of larger codebooks can improve the graph description and increase the classification results, as shown in Fig. 9(b). However, if the codebook is too large, the words become too specific: similar patterns that should correspond to the same word may be assigned to different words on the codebook. The optimum size is the right trade-off between diversity and generality.

Impact of the training set size. We evaluate the performance of BoSG and the edit distance approaches using different sizes of training set. In this experiment, we built different training sets selecting a percentage of graphs from each class of the original training sets.

Table 6 shows the different sizes of training set and dictionaries used in this experiment and Fig. 10(a) – (d) show the best results obtained for each approach using the (KNN) classifier with *K* equals to one, three, and five, and demonstrate that our results are similar to evaluated baselines. Fig. 10(b) and (d) show that our method yields better results than Jouili's approach in the case of Mutagenicity and Letter datasets. However, Jouili's approach has a better performance in the GREC dataset, as in Fig. 10(a). In this experiment, Riesen's is the best in all datasets, but we reach a similar performance in the Mutagenicity and AIDS datasets.

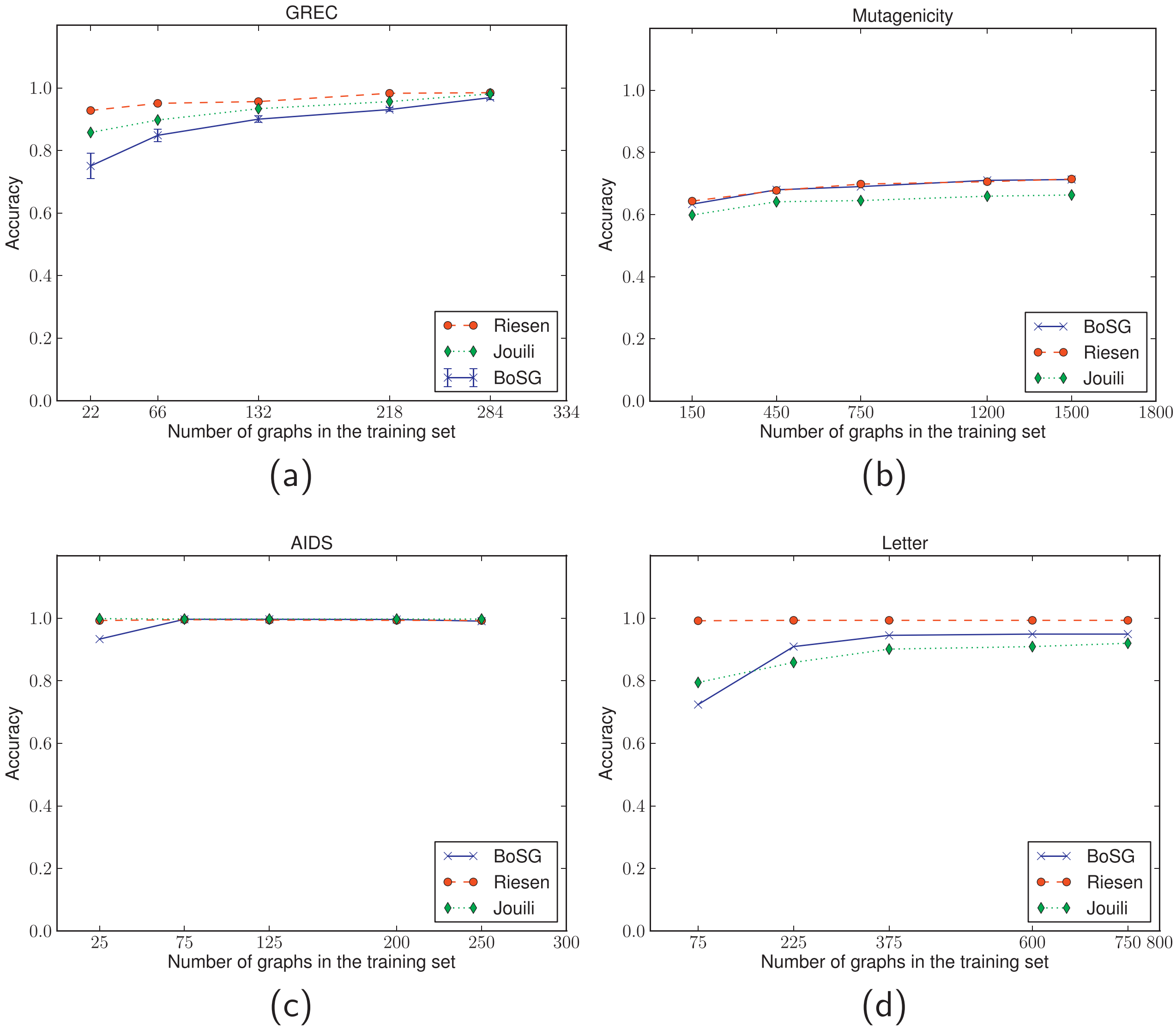

Fig. 10. Effect of training set size on BoSG. Study of the training set size on the performance of the BoSG on GREC (a), Mutagenicity (b), AIDS (c), and Letter (d) datasets.

Table 6
Different sizes of codebook used by BoSG approach.

		10%	30%	50%	80%	100%
GREC	Size of training set	22	66	132	218	284
	Size of Codebook	24	18	21	28	29
Mutagenicity	Size of training set	150	450	750	1200	1500
	Size of codebook	26	31	32	33	34
AIDS	Size of training set	25	75	125	200	250
	Size of codebook	11	14	19	22	23
Letter	Size of training set	75	225	375	600	750
	Size of codebook	122	101	102	104	109

Table 7
BoSG results using different classifiers.

	GREC	Mutagenicity	AIDS	Letter
KNN	0.969 ± 0.007	0.713	**0.991**	0.949
SVM	0.972 ± 0.008	**0.745**	**0.991**	**0.965**
OPF	**0.986** ± **0.004**	0.661	0.987	0.946

Impact of the classifier algorithm. The representation of graphs as feature vectors allows the use of different classifiers, and provides flexibility in tuning for higher accuracy rates. Table 7 compares our method using three classifiers: KNN, Support Vector Machine (SVM) [73], and Optimum-Path Forest (OPF) [74], and shows that the choice of the right classifier can improve the results. We used the validation sets to seek the best parameters for SVM and OPF. Using SVM, the obtained results are similar or better when compared with the other approaches.

Table 8, in turn, compared the results of BoG with SVM classifier with traditional methods proposed in the literature. The results observed for BoG are superior or comparable to the ones observed for methods based on Kernel Embedding [36], Lipschitz Embedding [37], and optimized dissimilarity space embedding (ODSEv1, ODSEv2) [38].

Table 8
Accuracy performance of the proposed BoG approach and two baselines based on Kernel Embedding [36], Lipschitz Embedding [37], and optimized dissimilarity space embedding (ODSEv1, ODSEv2) [38].

	Letter	GREC	AIDS	Mutagenicity
BoG_{SVM}	96.5	97.2	99.1	74.5
Kernel embedding [36]	92.7	92.9	98.2	**75.9**
Lipschitz embedding [37]	**99.3**	96.8	98.3	71.9
ODSEv1 [38]	98.6	96.2	**99.6**	73.4
ODSEv2 [38]	99.0	**97.9**	**99.6**	72.0

5.2. Bag of Visual Graphs

Section 4.2 introduced a graph-based approach to encode the distribution of visual-word arrangements, the *Bag of Visual Graphs (BoVG)*. The proposed approach combines the spatial locations of interest points and their labels defined in terms of the traditional visual codebook to define a set of connected graphs. This set of connected graphs encodes the spatial relationships of visual words and we use them to create a graph-based codebook. An image is represented by a vector of the distribution of *visual graphs*.

The computational costs for the visual codebook creation and image classification are the same as those observed for the BoW approach, but our method has an additional cost: the graph-based codebook creation.

The validation of the BoVG representation considers two different applications: image object classification and remote sensing image classification. In the first application scenario, the BoVG representation is computed as described in Section 4.2. For the latter application, we use the Border/Interior Classification (BIC) [75] descriptor in the characterization of vertices and edges. BIC is a color descriptor whose computation classifies pixels as belonging to a border or interior region. A border pixel is defined based on if its quantized color differs from any of its neighbors and as an interior pixel, otherwise. The feature vector generated is computed by combining the color histograms associated with interior and border pixels. The use of BIC is motivated by its recent good results when characterizing remote sensing image regions [76,77]. The first BoVG method implemented, named BoVG-BIC$_{LBP}$, uses BIC as local descriptor and LBP as edge descriptor. The second method, named BoVG-SIFT$_{BIC64}$, uses SIFT as local descriptor and BIC as edge descriptor. These variations demonstrate the flexibility of the method in encoding different region description approaches.

5.2.1. Image object classification

This section presents conducted experiments to validate the BoVG representation in image object classification problems.

Experimental protocol. In this experiments, we use SIFT [65] local descriptor, with sparse keypoint detection (*Hessian Affine* [78] and *Difference Of Gaussians* [65] keypoint detectors), and with a dense 6-pixel space sampling grid [79,80]. Interest-point selection impacts the definition of graph edges.

We did not impose edge-constraints when using dense sampling, it would either drop or keep all edges. In this case, we use all triangles of the Delaunay triangulation as connected graphs. In the case of sparse sampling, we used 10 and 150 pixels as edge weight constraints.

A simple random selection generates both visual-word and graph-based codebooks. On all experiments, we created the graph-based codebook with the same size of the visual-word codebook. Here, all bag representations were generated using *hard assignment* and *average pooling*.

Datasets. We used three online available image datasets: Caltech-101 [7], Caltech-256 [8], and ALOI [81]. These datasets contain general objects from different categories and they are usually used for image classification.

Caltech-101 [7] contains images from 101 object classes and a background category. The images of this dataset represent general objects, respecting a left-right alignment. The object classes do not have the same number of images, each of which may have from 31 to 800 images. For the experiments of this section, we did not use the background category. We used a total of 8878 images that belong to the 101 object classes.

Caltech-256 [8] contains images from 256 different object classes and a background clutter category. The image classes are not balanced, each of which may have from 80 to 827 images. The images of this dataset represent general objects that do not respect any alignment rule. Additionally, Caltech-256 contains images of widely different sizes. For the experiments of this section, we did not use the clutter category. We used a total of 30,291 images that belong to the 256 object classes.

Table 9
Evaluated variations of the BoVG approach.

Method	Description
BoVG	BoVG model
BoVG-BoW	BoW and BoVG feature vectors concatenated
BoVG-SP	BoVG and SP feature vectors concatenated
SPwithBoVG	SP using BoVG in each image region

The ALOI dataset [81] has 1000 classes and 108 samples for each class (108,000 in total). Samples contain objects under different viewing angle and illumination angles and colors.

Baselines. In this section, we compare the proposed approach (BoVG) with the traditional BoW [2], the SP method [3], and the WSA method [5].

Spatial Pyramids [3] is one of the most popular methods from the literature of Bag of Visual Words, achieving high accuracy rates on image classification. The WSA [82] achieves good results using feature vectors with smaller dimensions. Table 9 describes the evaluated BoVG-based approaches.

Evaluation measures. We measure the effectiveness of a method in terms of accuracy. For the classification procedure, we used an one-vs-all SVM [73,83] with kernel RBF and default parameters. The training and test sets were randomly separated, using the same number of samples per class for training and the rest for test. Each experiment was executed 10 times and the mean accuracy was computed with a confidence interval of 95%.

Results and discussion

Impact of codebook size and interest-point detector. The first experiment evaluates the performance of BoVG and some variations of this method on Caltech-101 using different sizes of codebook (200, 500, and 1000) and different interest-point detectors (Hessian Affine and SIFT). For this experiment, the traditional BoW was used as reference and we used 30 samples per class for training the classifier.

Fig. 11(a) and (b) compare mean accuracies of each method for different codebook sizes. These results show that the distribution of visual-word arrangements contribute to improve the classification results of other approaches for all cases evaluated, and the combination of BoVG with Spatial Pyramids hold the best results. The size of the codebook has a greater impact on BoW, which obtained higher accuracy rates than BoVG on large codebooks.

From now on, with regard the combination of BoVG and SP, we use only the variation of BoVG-SP in our experiments as it yields better results than SP with BoVG.

Impact of the training set size. Fig. 12 shows the effect of training size on the overall performance. The figure compares BoVG, BoVG-SP, BoW, WSA, and SP using SIFT detector and a 200-word codebook on Caltech-101 and shows that the accuracy increases with the training size. Since the results of all methods are equally improved, this experiment indicates that the size of the training set does not favor one representation over another.

Classification accuracy. Fig. 13(a) –(c) show the results of a experiment whose objective is to compare BoVG and BoVG-SP with BoW, WSA, SP, and the concatenation of BoW with SP (BoW-SP), with regard to the use of different techniques for interest-point detection. These methods were evaluated on Caltech-101, Caltech-256, and ALOI using codebooks of size 200 and 1000, respectively. We used again a training set with 30 images per class.

The results in Fig. 13 show that BoVG-SP has a competitive performance on all datasets. In the experiment on Caltech-101 (Fig. 13(a)), BoVG-SP yields the best classification accuracy rates in

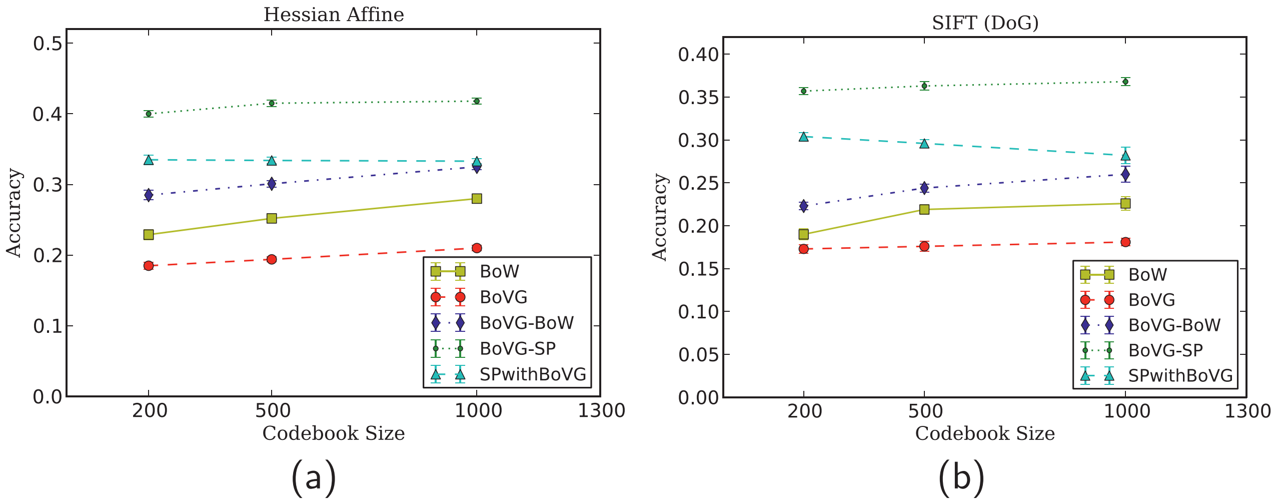

Fig. 11. Keypoint detector and codebook size on Caltech-101. Effect of keypoint detectors – Hessian Affine (a) and SIFT (b) – and codebook size on the performance of BoVG and variations on Caltech-101 dataset.

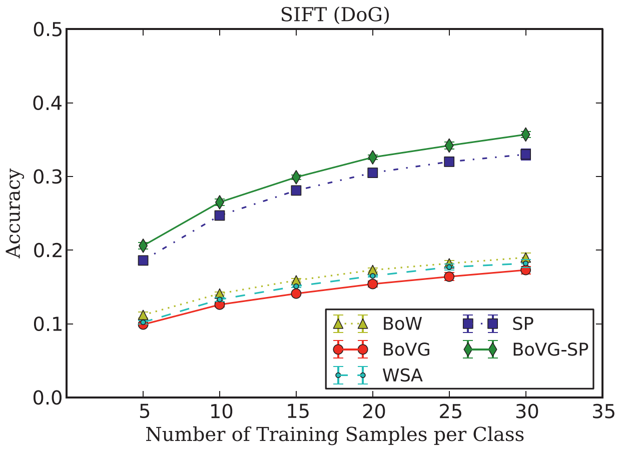

Fig. 12. Distinct training set size on Caltech-101. Caltech-101 classification results of BoW, BoVG, WSA, SP, and BoVG-SP for distinct training set sizes.

all cases, with a statistical tie with SP and BoW-SP for the dense-sampling case. In Caltech-256 (Fig. 13(b)), BoVG-SP achieves the highest accuracy rate using a dense sampling grid and SIFT detector. In the case of SIFT detector, BoVG-SP is statistically tied with BoW-SP. In ALOI (Fig. 13(c)), BoVG-SP achieves again the highest accuracy using a dense sampling grid and SIFT detector. In all datasets, dense interest-point sampling provides the highest accuracy rates, as some objects and/or properties do not have enough salient regions to be selected by sparse detectors, and dense sampling ensures their presence in the pooling.

5.2.2. Remote sensing image classification

This section presents conducted experiments to validate the BoVG representation in remote sensing image region classification problems.

Datasets. The first dataset used in this work was a composition of scenes of Monte Santo de Minas county, in the state of Minas Gerais, Brazil. These images were taken by a SPOT sensor in 2005, selecting the *red*, *infrared*, and *green* bands, with a total size of 1000 × 1000 pixels with spatial resolution of 2.5 m. This area comprises a coffee cultivation, and was divided into 3 region masks that comprehends the whole image.

The second dataset is an image of Campinas, in the state of São Paulo, Brazil. This image was taken by Quickbird satellite in 2003, comprising the three visible bands (*red*, *green*, and *blue*). This image size is 9079 × 9486 pixels, with 0.62 m of spatial resolution. Eight masks divide the entire image into eight labels (bare soil, building, forest, houses, mixed field, road and parking, sugar cane,

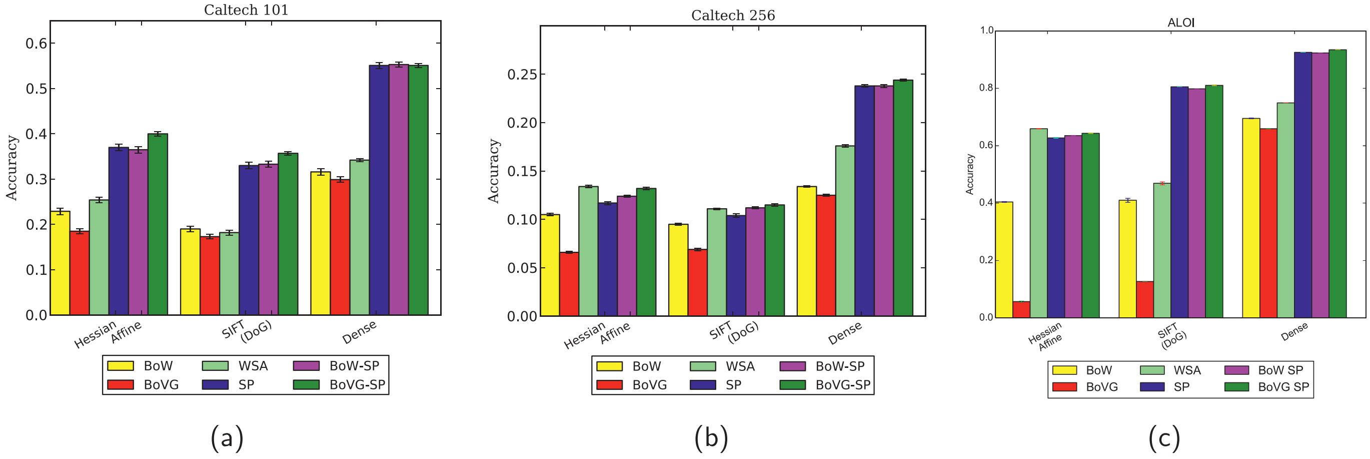

Fig. 13. Performance on datasets. Classification results of BoW, BoVG, WSA, SP, and BoVG-SP using different interest-point detectors on Caltech-101 (a), Caltech-256 (b), and ALOI (c) datasets.

Table 10
Comparison of global descriptors in the Monte Santo dataset.

Global descriptor	Norm. accuracy	Kappa
BIC	**94.20%**	**0.8987**
GCH	90.89%	0.8288
QCCH	89.16%	0.8255
Unser	56.37%	0.3758

Table 11
Effectiveness performance results for the proposed method and baselines in the Monte Santo dataset.

Descriptor	Normalized acc.	Global acc.	Kappa
Global BIC	94.20%	93.63%	0.8987
BoW-BIC	88.81%	87.74%	0.8046
BoW-SIFT	57.93%	63.12%	0.3924
BoVG-BIC$_{LBP}$	**95.14%**	**96.10%**	**0.9275**
BoVG-SIFT$_{BIC64}$	93.31%	93.56%	0.8966

Table 12
Effectiveness performance results for the proposed method and baselines in the Campinas dataset.

Descriptor	Normalized acc.	Global acc.	Kappa
Global BIC	80.41%	86.99%	0.8351
BoW-BIC	88.64%	90.15%	0.8762
BoW-SIFT	73.20%	83.39%	0.7901
BoVG-BIC$_{LBP}$	**87.71%**	**92.73%**	**0.9068**
BoVG-SIFT$_{BIC64}$	49.34%	63.11%	0.5318

and unclassified), but our work only uses seven of them, excluding the unclassified mask.

The next step is concerned with the annotation of image regions. In order to obtain the set of regions, we segment the image and associate each segmented region with a label. In this work, we use a superpixel algorithm to group pixels into perceptually meaningful regions. We selected the Simple Linear Iterative Clustering (SLIC) [84] because it is fast and it is memory efficient. The SLIC algorithm has only one parameter, which is the number of equally-sized superpixels wanted. We selected 300 superpixels for the Monte Santo dataset. To assign a label to each superpixel region, we looked up the intersection of each region with the masks of the classes, and if a mask intersected the region in more than 60% of its pixels, this region is labeled as the mask label. After labeling the regions, we cropped the superpixels into separated files, obtaining a total of 203 images. The same was made with the Campinas dataset, selecting approximately 900 superpixels in the SLIC algorithm. It was necessary to select this number of superpixels because the unclassified mask takes a large part of the Campinas remote sensing image, then most of the superpixels have a majority of pixels in this unclassified mask. By cropping the superpixels labeled with the selected classes, we obtained a total of 246 regions to classify.

Baselines. To compare our feature extraction with the literature, we selected two global color descriptors, two global texture descriptors, and the traditional Bag-of-Visual-Words approach (BoW) as baselines.

We chosen the Border/Interior pixel Classification (BIC) [75] and Global Color Histogram (GCH) [85] because GCH is one of the most popular descriptor and a constant baseline, and BIC achieved a better overall effectiveness in a web retrieval scenario [86] and RSI classification tasks [87]. The texture descriptors chosen in this work are Quantized Compound Change Histogram (QCCH) [88], because of the simplicity of its extraction algorithm and compact feature vector, and Unser [89], which has a compact feature vector and lower complexity [86].

We performed experiments to select the global descriptor baseline with the best performance describing the images of the Monte Santo dataset. Table 10 shows their normalized accuracy and kappa index. As we can see, the BIC descriptor had the best result, and we selected it for the following experiments.

The BoW approach was built according to the literature [90]. We used a dense sampling of 16 pixels with an overlapping of half of the region using either the BIC descriptor as the SIFT descriptor. The codebook has 200 words, selected through the K-Means clustering. Hard assignment and sum pooling were used.

Experimental protocol. The protocol selected in this work was a stratified k-fold cross-validation, which splits the dataset into k folds, but preserves as much as possible the class proportion of images among the folds. We used 5 folds, with one fold for testing, while the remain four are used for training. We made the classifications using a linear SVM from libSVM 3.17 with default parameters.

Evaluation measures. We present our results using the balanced average accuracy, which is the mean of the accuracy for each of the classes. To evaluate our results, we present the agreement between the classification and the ground-truth with Cohen's Kappa, and an statistical analysis with Student's t-test and Wilcoxon test. These tests were applied to confirm if our approach has a significantly difference from the other experimented methods.

Results and discussions. The objective of the performed experiments is to demonstrate that the proposed Bag of Visual Graphs yields effective results in remote sensing image classification tasks. The best results for the Monte Santo dataset are shown in Table 11. The results for the Campinas dataset are shown in Table 12.

Table 11 shows that the BoVG approach, either with the BIC descriptor or with the SIFT descriptor, has a similar accuracy to the global BIC. In fact, the accuracy of all BoVG surpasses by far the accuracy of the literature BoW approach. The use of a color descriptor in the BoVG-SIFT$_{BIC64}$ led to a great increase in the accuracy from the BoW-SIFT, confirming that the combination of texture and color is a good strategy for the description of RSIs. The results shown in Table 12 are consistent with the results presented in Table 11. The methods that describe the image using the BIC descriptor achieve better results than the methods using SIFT.

We performed statistical tests to compare the obtained results. The results are presented using the Students t-test for the normalized accuracy, in which we compare our best approach (BoVG-BIC$_{LBP}$) with the other methods. If the comparison is above the zero line, the BoVG-BIC$_{LBP}$ yields better results and is statistically different from its counterpart. If the comparison cross the zero line, we can not assure their difference. Fig. 14(a) shows the Students t-test analysis for the Monte Santo dataset. We can observe that our BoVG-BIC$_{LBP}$ is statistically different from the BoW approaches, yielding comparable results to the Global BIC method. The Wilcoxon test confirms the results obtained with the Student's t-test.

Fig. 14(b) shows the Student's t-test applied to the experiments performed on the Campinas dataset. Our method yields better or comparable results to the ones observed for the baselines, when normalized accuracy measure is considered. As it can be observed in Table 12, in all the other evaluation measures, our proposed method achieves the best overall scores.

6. Conclusions

The guiding principle of this article was that a discriminant and efficient representation based on local structures of an object could be created by combining graphs with the BoW model. Based on this hypothesis, we investigated how to generate a meaningful vo-

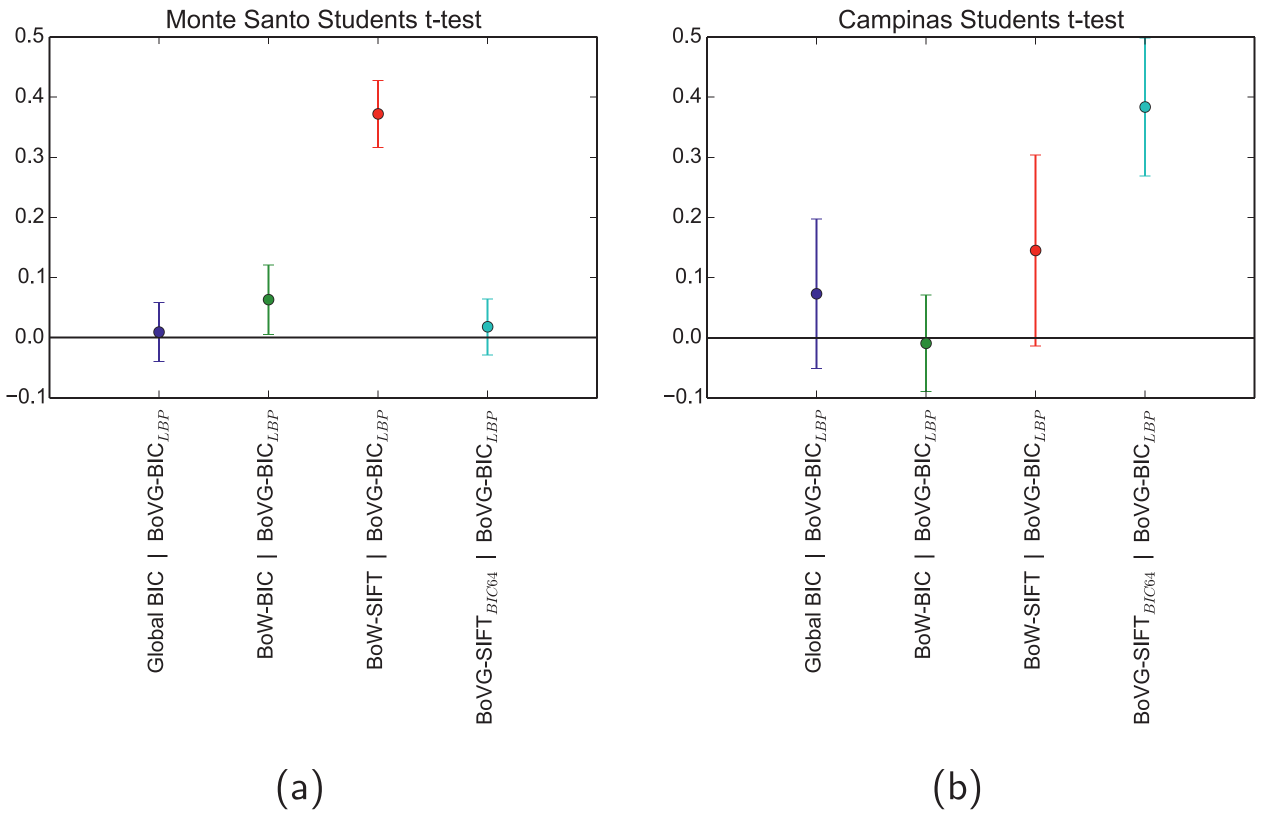

Fig. 14. Statistical analyses. Statistical analyses of the experiments in remote sensing image classification using Student's *t*-test on Monte Santo (a) and Campinas (b) datasets.

cabulary that describes the main local patterns of a set of objects. Using this vocabulary, an object would be represented by a feature vector that describes the occurrence of local patterns within this object.

We introduced a generic BoW-based approach, called Bag of Graphs (BoG), that uses a graph-based vocabulary to create object representations. We then describe two instances of the theory that show how a graph-based vocabulary can describe images and molecules.

For graph classification, we proposed the Bag of Singleton Graphs (BoSG), an approach that uses the BoG model to describe a graph with a vector representation based on graph local structures. The experiments show how the BoSG approach is an efficient alternative for performing graph matching. The use of feature vectors to represent graphs makes large database graph-retrieval possible, limited before due to the high computational cost of approaches that rely on graph-matching.

For image classification, we presented the Bag of Visual Graphs (BoVG), a new approach to incorporate the information about spatial relationships of visual words into the BoVW model. This approach uses graphs to represent the local distribution of visual words, and proposes the use of a graph-based vocabulary to generate image descriptors. As in all the BoW-derived methods, an important open question is how to find out the optimum codebook size. In the case of BoVG, this question is even harder, as there are two different codebooks interacting together. Experimental results show that BoVG improves the classification performance when combined with other approaches, such as the Spatial Pyramid method. Since our approach is a generic descriptor method, the BoVG is a promising alternative for image classification and retrieval.

In future work, we plan to investigate the relation between the sizes of the two codebooks in the BoVG approach and investigate different applications for the proposed method, such as symbol spotting [47] and video retrieval [91]. Finally, the performance of BoSG approach with an index structure could be evaluated. An interesting experiment would be to test the performance of BoSG approach with an indexing structure, like Locality-Sensitive Hashing (LSH) [92] or K-Dimensional Tree (KDTree) [93], for graph retrieval on large datasets.

Acknowledgments

We would like to thank FAPESP-Microsoft Virtual Institute (grants #2013/50169-1 and #2013/50155-0), FAPESP (grants #2016/18429-1, #2014/12236-1, #2012/16172-2, #2012/50468-6, and #2013/11378-4), CNPq (grants #139135/2012-0, #308882/2013-0, #141584/2016-5), CAPES (#1458623), and IEEE Signal Processing Society for the financial support, and AMD, Microsoft Research, and SAMSUNG for the infrastructure provided to this research project.

References

[1] R.A. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.
[2] J. Sivic, B.C. Russell, A.A. Efros, A. Zisserman, W.T. Freeman, Discovering objects and their location in images, in: IEEE Int. Conf. on Comp. Vision, 1, 2005, pp. 370–377.
[3] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: spatial pyramid matching for recognizing natural scene categories, in: IEEE Comp. Vision and Pat. Recog., vol. 2, IEEE, 2006, pp. 2169–2178.
[4] N.V. Hoàng, V. Gouet-Brunet, M. Rukoz, M. Manouvrier, Embedding spatial information into image content description for scene retrieval, Pat. Recognit. 43 (9) (2010) 3013–3024.
[5] O. Penatti, F. Silva, E. Valle, V. Gouet-Brunet, R. da Silva Torres, Visual word spatial arrangement for image retrieval and classification, Pat. Recognit. 47 (2) (2014) 705–720.
[6] K. Riesen, H. Bunke, Iam graph database repository for graph based pattern recognition and machine learning, in: Structural, Syntactic, and Statistical Pat. Recog., Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 287–297.

[7] L. Fei-fei, R. Fergus, P. Perona, Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories, Comp. Vis. Image Understand. 106 (1) (2007) 59–70.
[8] G. Griffin, A. Holub, P. Perona, Caltech-256 Object Category Dataset, Technical Report, 7694, California Institute of Technology, 2007.
[9] F.B. Silva, S. Goldenstein, S. Tabbone, R. da Silva Torres, Image classification based on bag of visual graphs, in: IEEE Int. Conf. on Image Proc., 2013, pp. 4312–4316.
[10] F.B. Silva, S. Tabbone, R. da Silva Torres, Bog: a new approach for graph matching, in: Int. Conf. on Pat. Recog., 2014, pp. 82–87.
[11] M. Vento, A long trip in the charming world of graphs for pattern recognition, Pat. Recog. 48 (2) (2015) 291–301.
[12] C.C. Aggarwal, H. Wang (Eds.), Managing and mining graph data, vol. 40. Advances in Database Systems, Springer, 2010.
[13] V. Carletti, P. Foggia, M. Vento, X. Jiang, Report on the first contest on graph matching algorithms for pattern search in biological databases, in: Graph-Based Rep. in Pat. Recog., 2015, pp. 178–187.
[14] H. Bunke, Recent developments in graph matching, in: Int. Conf. on Pat. Recog., vol. 2, 2000, pp. 117–124.
[15] A. Robles-Kelly, E.R. Hancock, Graph edit distance from spectral seriation, IEEE Trans. Pat. Anal. Mach. Intell. 27 (3) (2005) 365–378.
[16] R.C. Wilson, E.R. Hancock, B. Luo, Pattern vectors from algebraic graph theory, IEEE Trans. Pat. Anal. Mach. Intell. 27 (7) (2005) 1112–1124.
[17] A. Rosenfeld, Adjacency in digital pictures, Inf. Control 26 (1) (1974) 24–33.
[18] R. Raveaux, J. Burie, J. Ogier, Structured representations in a content based image retrieval context, J. Vis. Comm. Image Rep. 24 (8) (2013) 1252–1268.
[19] W.-B. Goh, Strategies for shape matching using skeletons, Comp. Vis. Image Understand. 110 (3) (2008) 326–345.
[20] C.D. Ruberto, Recognition of shapes by attributed skeletal graphs, Pat. Recognit. 37 (1) (2004) 21–31.
[21] T.B. Sebastian, P.N. Klein, B.B. Kimia, Recognition of shapes by editing their shock graphs, IEEE Trans. Pat. Anal. Mach. Intell. 26 (5) (2004) 550–571.
[22] K. Siddiqi, A. Shokoufandeh, S.J. Dickinson, S.W. Zucker, Shock graphs and shape matching, Int. J. Comp. Vis. 35 (1) (1999) 13–32.
[23] K.C. Santosh, B. Lamiroy, L. Wendling, Symbol recognition using spatial relations, Pat. Recognit. Lett. 33 (3) (2012) 331–341.
[24] X. Xiaogang, S. Zhengxing, P. Binbin, J. Xiangyu, L. Wenyin, An online composite graphics recognition approach based on matching of spatial relation graphs, Doc. Anal. Recognit. 7 (1) (2004) 44–55.
[25] L. Wiskott, J.-M. Fellous, N. Krüger, C. von der Malsburg, Face recognition by elastic bunch graph matching, IEEE Trans. Pat. Anal. Mach. Intell. 19 (7) (1997) 775–779.
[26] V.N. Gudivada, V.V. Raghavan, Design and evaluation of algorithms for image retrieval by spatial similarity, ACM Trans. Inf. Syst. 13 (2) (1995) 115–144.
[27] V.N. Gudivada, G.S. Jung, Spatial knowledge representation and retrieval in 3d image databases, in: Intl. Conf. on Multim. Comp. and Systems, 1995, pp. 90–97.
[28] L. He, C.Y. Han, W.G. Wee, Object recognition and recovery by skeleton graph matching, in: Intl. Conf. on Multim. and Expo, 2006, pp. 993–996.
[29] M. Bergtholdt, J. Kappes, S. Schmidt, C. Schnörr, A study of parts-based object class detection using complete graphs, Int. J. Comp. Vis. 87 (1) (2010) 93–117.
[30] E. Bengoetxea, Inexact Graph Matching Using Estimation of Distribution Algorithms, Ecole Nationale Supérieure des Télécommunications, Paris, France, 2002 Ph.D. thesis.
[31] H. Bunke, G. Allermann, Inexact graph matching for structural pattern recognition, Pat. Recognit. Lett. 1 (4) (1983) 245–253.
[32] X. Gao, B. Xiao, D. Tao, X. Li, A survey of graph edit distance, Pat. Anal. Appl. 13 (1) (2010) 113–129.
[33] K. Riesen, M. Neuhaus, H. Bunke, Bipartite graph matching for computing the edit distance of graphs, in: Graph-based Rep. in Pat. Recog., 2007, pp. 1–12.
[34] S. Jouili, I. Mili, S. Tabbone, Attributed graph matching using local descriptions, in: Adv. Conc. for Intell. Vision Systems, 2009, pp. 89–99.
[35] G. Salton, A. Wong, C.S. Yang, A vector space model for automatic indexing, Commun. ACM 18 (11) (1975) 613–620.
[36] K. Riesen, H. Bunke, Reducing the dimensionality of dissimilarity space embedding graph kernels, Eng. Appl. Artif. Intell. 22 (1) (2009) 48–56.
[37] K. Riesen, H. Bunke, Graph classification by means of lipschitz embedding, IEEE Trans. Syst. Man Cybern. 39 (6) (2009) 1472–1483.
[38] L. Livi, A. Rizzi, A. Sadeghian, Optimized dissimilarity space embedding for labeled graphs, Inf. Sci. 266 (2014) 47–64.
[39] J. Wu, X. Zhu, C. Zhang, P.S. Yu, Bag constrained structure pattern mining for multi-graph classification, IEEE Trans. Knowl. Data Eng. 26 (10) (2014) 2382–2396.
[40] J. Wu, S. Pan, X. Zhu, Z. Cai, Boosting for multi-graph classification, IEEE Trans. Cybern. 45 (3) (2015) 416–429.
[41] J. Wu, Z. Hong, S. Pan, X. Zhu, C. Zhang, Z. Cai, Multi-graph learning with positive and unlabeled bags, in: SIAM Int. Conf. on Data Mining, 2014, pp. 217–225.
[42] J. Wu, S. Pan, X. Zhu, C. Zhang, X. Wu, Positive and unlabeled multi-graph learning, IEEE Trans. Cybern. 47 (4) (2017) 818–829.
[43] J. Wu, Z. Hong, S. Pan, X. Zhu, Z. Cai, C. Zhang, Multi-graph-view subgraph mining for graph classification, Knowl. Inf. Syst. 48 (1) (2016) 29–54.
[44] J. Wu, S. Pan, X. Zhu, Z. Cai, C. Zhang, Multi-graph-view learning for complicated object classification, in: Int. Conf. on Artif. Intell., AAAI Press, 2015, pp. 3953–3959.
[45] J. Wu, X. Zhu, C. Zhang, Z. Cai, Multi-instance multi-graph dual embedding learning, in: Int. Conf. on Data Mining, 2013, pp. 827–836.
[46] J. Sivic, A. Zisserman, Video google: A text retrieval approach to object matching in videos, in: IEEE Int. Conf. on Comp. Vision, 2003, pp. 1470–1477.
[47] E. Barbu, P. Héroux, S. Adam, É. Trupin, Using bags of symbols for automatic indexing of graphical document image databases, in: Graphics Recognition. Ten Years Review and Future Perspectives, Springer Berlin Heidelberg, 2006, pp. 195–205.
[48] T. Hou, X. Hou, M. Zhong, H. Qin, Bag-of-Feature-Graphs: A new paradigm for non-rigid shape retrieval, in: Int. Conf. on Pat. Recog., 2012, pp. 1513–1516.
[49] S. Karaman, J. Benois-Pineau, R. Mégret, A. Bugeau, Multi-layer local graph words for object recognition, in: Adv. in Multim. Modeling, 2012, pp. 29–39.
[50] Y.-L. Boureau, F. Bach, Y. LeCun, J. Ponce, Learning mid-level features for recognition, in: IEEE Comp. Vision and Pat. Recog., 2010, pp. 2559–2566.
[51] A. Rocha, T. Carvalho, H. Jelinek, S. Goldenstein, J. Wainer, Points of interest and visual dictionaries for automatic retinal lesion detection, IEEE Trans. Biomed. Eng. 59 (8) (2012) 2244–2253.
[52] Y. Cao, C. Wang, Z. Li, L. Zhang, L. Zhang, Spatial-bag-of-features, in: IEEE Comp. Vision and Pat. Recog., 2010, pp. 3352–3359.
[53] S. Savarese, J. Winn, A. Criminisi, Discriminative object class models of appearance and shape by correlatons, in: IEEE Comp. Vision and Pat. Recog., 2, 2006, pp. 2033–2040.
[54] E.B. Sudderth, A. Torralba, W.T. Freeman, A.S. Willsky, Learning hierarchical models of scenes, objects, and parts, in: IEEE Int. Conf. on Comp. Vision, 2, 2005, pp. 1331–1338.
[55] J.C. Niebles, L. Fei-fei, A hierarchical model of shape and appearance for human action classification, in: IEEE Comp. Vision and Pat. Recog., 2007, pp. 1–8.
[56] M.W. M. Weber, P. Perona, Unsupervised learning of models for recognition, in: Europ. Conf. on Comp. Vision, 2000, pp. 18–32.
[57] R. Fergus, P. Perona, A. Zisserman, Object class recognition by unsupervised scale-invariant learning, in: IEEE Comp. Vision and Pat. Recog., vol. 2, 2003, pp. 264–271.
[58] A. Bolovinou, I. Pratikakis, S. Perantonis, Bag of spatio-visual words for context inference in scene classification, Pat. Recognit. 46 (3) (2013) 1039–1053.
[59] Y. Liu, V. Caselles, Spatial string matching for image classification, in: Int. Conf. on Pat. Recog., 2010, pp. 2937–2940.
[60] L. Zhou, Z. Zhou, D. Hu, Scene classification using a multi-resolution bag-of-features model, Pat. Recognit. 46 (1) (2013) 424–433.
[61] E.A. Fox, M.A. Gonçalves, R. Shen, Theoretical Foundations for Digital Libraries: The 5S (Societies, Scenarios, Spaces, Structures, Streams) Approach, Morgan & Claypool Publishers, 2012.
[62] R. da S. Torres, A.X. Falcão, Content-based image retrieval: theory and applications, Revista de Informática Teórica e Aplicada 13 (2) (2006) 161–185.
[63] J. van Gemert, C. Veenman, A. Smeulders, J.-M. Geusebroek, Visual word ambiguity, IEEE Trans. Pat. Anal. Mach. Intell. 32 (7) (2010) 1271–1283.
[64] D.R. Wilson, T.R. Martinez, Improved heterogeneous distance functions, J. Artif. Intell. Res. 6 (1) (1997) 1–34.
[65] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comp. Vis. 60 (2) (2004) 91–110.
[66] M. Hashimoto, Detecção de objetos por reconhecimento de grafos-chave, Universidade de São Paulo, São Paulo, Brasil, 2012 Ph.D. thesis.
[67] T. Ojala, M. Pietikäinen, T. Mäenpää, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Trans. Pat. Anal. Mach. Intell. 24 (7) (2002) 971–987.
[68] H. Kuhn, The hungarian method for the assignment problem, Nav. Res. Logist. Q. 2 (1–2) (1955) 83–97.
[69] S. Jouili, S. Tabbone, V. Lacroix, Median graph shift: A new clustering algorithm for graph domain, in: Int. Conf. on Pat. Recog., 2010, pp. 950–953.
[70] S. Jouili, S. Tabbone, A hypergraph-based model for graph clustering: Application to image indexing, in: Comp. Anal. of Images and Pat., 2009, pp. 360–368.
[71] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pat. Anal. Mach. Intell. 24 (5) (2002) 603–619.
[72] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[73] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Trans. Intel. Syst. Tech. 2 (3) (2011) 27:1–27:27.
[74] J. Papa, A. Falcão, C. Suzuki, LibOPF: a library for the design of optimum-path forest classifiers, 2009. Software version 2.0 available at http://www.ic.unicamp.br/~afalcao/LibOPF.
[75] R.O. Stehling, M.A. Nascimento, A.X. Falcão, A compact and efficient image retrieval approach based on border/interior pixel classification, in: ACM Int. Conf. on Inform. and Knowl. Manag., ACM, 2002, pp. 102–109.
[76] J.A. dos Santos, P.H. Gosselin, S. Philipp-Foliguet, R. da Silva Torres, A.X. Falcão, Multiscale classification of remote sensing images, IEEE Trans. Geosci. Remote Sensing 50 (10) (2012) 3764–3775.
[77] J. dos Santos, O. Penatti, P. Gosselin, A. Falcão, S. Philipp-Foliguet, R. Torres, Efficient and effective hierarchical feature propagation, IEEE J. Select Top. Appl. Earth Observ. aRemote Sensing 7 (12) (2014) 4632–4643.
[78] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L.V. Gool, A comparison of affine region detectors, Int. J. of Comp. Vision 65 (1) (2005) 43–72.
[79] K.E.A. van de Sande, T. Gevers, C.G.M. Snoek, Evaluating color descriptors for object and scene recognition, IEEE Trans. Pat. Anal. Mach. Intell. 32 (9) (2010) 1582–1596.

[80] K.E.A. van de Sande, T. Gevers, C.G.M. Snoek, Empowering visual categorization with the GPU, IEEE Trans. Multim. 13 (1) (2011) 60–70.
[81] J.-M. Geusebroek, G.J. Burghouts, A.W.M. Smeulders, The Amsterdam library of object images, Int. J. Comp. Vis. 61 (1) (2005) 103–112.
[82] O.A.B. Penatti, F.B. Silva, E. Valle, V. Gouet-Brunet, R.S. Torres, Visual word spatial arrangement for image retrieval and classification, Pattern Recognit. 47 (2) (2014) 705–720.
[83] T.-K. Huang, R.C. Weng, C.-J. Lin, Generalized bradley-terry models and multi–class probability estimates, J. Mach. Learn. Res. 7 (2006) 85–115.
[84] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, Slic superpixels compared to state-of-the-art superpixel methods, IEEE Trans. Pat. Anal. Mach. Intell. 34 (11) (2012) 2274–2282.
[85] M.J. Swain, D.H. Ballard, Color indexing, Int. J. Comp. Vis. 7 (1) (1991) 11–32.
[86] O.A.B. Penatti, E. Valle, R.d. S. Torres, Comparative study of global color and texture descriptors for web image retrieval, J. Visu. Comm. Image Rep. 23 (2) (2012) 359–380.
[87] J.A. dos Santos, O.A.B. Penatti, R. da Silva Torres, Evaluating the potential of texture and color descriptors for remote sensing image retrieval and classification, in: Conf. on Comp. Vision Theory and Appl., 2010, pp. 203–208.
[88] C.-B. Huang, Q. Liu, An orientation independent texture descriptor for image retrieval, in: Int. Conf. on Comm., Circuits and Systems, 2007, pp. 772–776.
[89] M. Unser, Sum and difference histograms for texture classification, IEEE Trans. Pat. Anal. Mach. Intell. 8 (1) (1986) 118–125.
[90] L. Chen, W. Yang, K. Xu, T. Xu, Evaluation of local features for scene classification using vhr satellite images, in: Joint Urban Remote Sensing Event, 2011, pp. 385–388.
[91] F.S.P. Andrade, J. Almeida, H. Pedrini, R. da S. Torres, Fusion of local and global descriptors for content-based image and video retrieval, in: Iberoamerican Cong. on Pat. Recog., Springer, 2012, pp. 845–853.
[92] P. Indyk, R. Motwani, Approximate nearest neighbors: towards removing the curse of dimensionality, in: ACM Symp. on Theory of Comp., 1998, pp. 604–613.
[93] J.H. Friedman, J.L. Bentley, R.A. Finkel, An algorithm for finding best matches in logarithmic expected time, ACM Trans. Math. Softw. 3 (3) (1977) 209–226.

Fernanda B. Silva received her engineering degree from Télécom Paris- Tech, France, in 2011, computer engineering degree from the University of Campinas, Brazil, in 2012, and M.Sc. degree in computer science from the University of Campinas in 2014. Currently, she works at Microsoft ATL Brazil.

Rafael de O. Werneck received the B.Sc. degree in computer science from the Universidade Federal de Juiz de Fora (UFJF), Juiz de Fora, Brazil, in 2011, the M.Sc. degree in computer science from the University of Campinas (Unicamp), Campinas, Brazil, in 2014, and is currently a Ph.D. student at the same university. His research interests include object representation, remote sensing, machine learning, and pattern recognition.

Siome Goldenstein is an Associate Professor at the Institute of Comput- ing, University of Campinas, Unicamp, Brazil and a senior IEEE member. He received a Ph.D. in Computer and Information Science from the University of Pennsylvania in 2002, a M.Sc. in Computer Science from Pontifical Catholic University of Rio de Janeiro in 1997, and an Electronic Engineering degree from the Federal University of Rio de Janeiro in 1995. His interests lie in the analysis of complex data for real problems, leveraging his background in im- age processing, computer vision, computer graphics, computer forensics, and machine learning. He is an Area Editor of two journals, Computer Vision and Image Understanding (CVIU) and Graphical Models (GMOD), has been in the program committee of multiple conferences and workshops, was the local or- ganizer of the 2007 IEEE Intl. Conference on Computer Vision in Brazil, and was co-chair of the 2007 IEEE Workshop on Computer Vision Applications for Developing Regions, and co-chair of the 2008 IEEE Workitorial of vision of the unseen.

Salvatore Tabbone is a professor in computer science at Université de Lorraine (France). He received his Ph.D. degree in computer science from Institut Polytechnique de Lorraine, France, in 1994. Since 2007, he heads the team QGAR at LORIA Laboratory and his research is related to image and graphics document processing and indexing, pattern recognition, image filtering and segmentation, content-based image retrieval. He has been and is the leader of several national and international projects funded by French and European institutes. He is author/co-author of more than 100 articles into refereed journal and conferences (see https://members.loria.fr/SATabbone/). He serves as PC members for several international and national conferences.

Prof. Dr. Ricardo da Silva Torres is Full Professor of computer science at the University of Campinas (UNICAMP). Dr. Torres received a B.Sc. in Computer Engineering from University of Campinas, Brazil, in 2000 and his Ph.D. degree in Computer Science at the same university in 2004. Dr. Torres is co-founder and member of the RECOD lab, where he has been developing multidisciplinary e-Science research involving Multimedia Analysis, Multime- dia Image Retrieval, Databases, Digital Libraries, and Geographic Information Systems. Dr. Torres is author/co-author of more than 100 articles in refereed journal and conferences and serves as PC member for several international and national conferences.