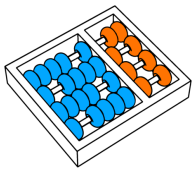
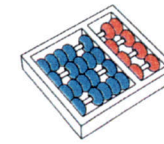


Fernanda Brandão Silva

"Bag of Graphs: Definition, Implementation, and Validation in Classification Tasks"

"*Sacola de Grafos: Definição, Implementação e Validação em Tarefas de Classificação*"

CAMPINAS
2014



University of Campinas
Institute of Computing

Universidade Estadual de Campinas
Instituto de Computação

Fernanda Brandão Silva

"Bag of Graphs: Definition, Implementation, and Validation in Classification Tasks"

Supervisor:
Orientador(a): Prof. Dr. Ricardo da Silva Torres

"Sacola de Grafos: Definição, Implementação e Validação em Tarefas de Classificação"

MSc Dissertation presented to the Post Graduate Program of the Institute of Computing of the University of Campinas to obtain a Master degree in Computer Science.

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Computação da Universidade Estadual de Campinas para obtenção do título de Mestre em Ciência da Computação.

THIS VOLUME CORRESPONDS TO THE FINAL VERSION OF THE DISSERTATION DEFENDED BY FERNANDA BRANDÃO SILVA, UNDER THE SUPERVISION OF PROF. DR. RICARDO DA SILVA TORRES.

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA DISSERTAÇÃO DEFENDIDA POR FERNANDA BRANDÃO SILVA, SOB ORIENTAÇÃO DE PROF. DR. RICARDO DA SILVA TORRES.

Ricardo Torres

Supervisor's signature / *Assinatura do Orientador(a)*

CAMPINAS
2014

ERRATA

Onde se lê: "...título de Mestre em Ciência da Computação."
Leia-se: "...título de Mestra em Ciência da Computação."

Prof. Paulo Lício de Geus
Coord. De Pós-Graduação
Instituto de Computação - Unicamp
Matrícula 10.326-8

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Maria Fabiana Bezerra Muller - CRB 8/6162

Si38b

Silva, Fernanda Brandão, 1988-
Bag of graphs : definition, implementation, and validation in classification tasks / Fernanda Brandão Silva. – Campinas, SP : [s.n.], 2014.

Orientador: Ricardo da Silva Torres.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Visão por computador. 2. Sistema de recuperação da informação. 3. Reconhecimento de padrões. 4. Classificação. I. Torres, Ricardo da Silva,1977-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

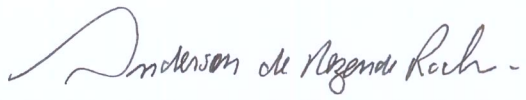
Informações para Biblioteca Digital

Título em outro idioma: Sacola de grafos : definição, implementação e validação em tarefas de classificação
Palavras-chave em inglês:
Computer vision
Information retrieval system
Pattern recognition
Classification
Área de concentração: Ciência da Computação
Titulação: Mestra em Ciência da Computação
Banca examinadora:
Ricardo da Silva Torres [Orientador]
Anderson de Rezende Rocha
Luciano da Fontoura Costa
Data de defesa: 16-05-2014
Programa de Pós-Graduação: Ciência da Computação

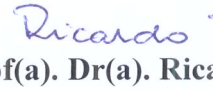
TERMO DE APROVAÇÃO

Defesa de Dissertação de Mestrado em Ciência da Computação, apresentada pelo(a) Mestrando(a) **Fernanda Brandão Silva**, aprovado(a) em **16 de maio de 2014**, pela Banca examinadora composta pelos Professores Doutores:

Prof(a). Dr(a). Luciano da Fontoura Costa
Titular



Prof(a). Dr(a). Anderson de Rezende Rocha
Titular



Prof(a). Dr(a). Ricardo da Silva Torres
Presidente

Institute of Computing /*Instituto de Computação*
University of Campinas /*Universidade Estadual de Campinas*

Bag of Graphs: Definition, Implementation, and Validation in Classification Tasks

Fernanda Brandão Silva[1]

May 16, 2014

Examiner Board/*Banca Examinadora*:

- Prof. Dr. Ricardo da Silva Torres (Supervisor/*Orientador*)
- Prof. Dr. Anderson de Rezende Rocha
  Institute of Computing - UNICAMP
- Prof. Dr. Luciano da Fontoura Costa
  São Carlos Institute of Physics - University of São Paulo
- Prof. Dr. David Menotti Gomes (Substitute/*Suplente*)
  Institute of Computing - UNICAMP
- Dr. Sandra Eliza Fontes de Avila (Substitute/*Suplente*)
  School of Electrical and Computer Engineering - UNICAMP

[1]Financial support: CNPq scholarship (grant 139135/2012-0) 2012–2013, and FAPESP scholarship (grants 2012/16172-2 and 2013/11378-4) 2013–2014.

Abstract

Nowadays, there is a strong interest for solutions that allow the implementation of effective and efficient retrieval and classification services associated with large volumes of data. In this context, several studies have been investigating the use of new techniques based on the comparison of local structures within objects in the implementation of classification and retrieval services. Local structures may be characterized by different types of relationships (e.g., spatial distribution) among object primitives, being commonly exploited in pattern recognition problems.

In this dissertation, we propose the *Bag of Graphs (BoG)*, a new approach based on the Bag-of-Words model that uses graphs for encoding local structures of a digital object. We present a formal definition of the proposed model, introducing concepts and rules that make this model flexible and adaptable for different applications. In the proposed approach, a digital object is represented by a graph that models the existing local structures. Using a pre-defined dictionary, the object is described by a vector representation with the frequency of occurrence of local patterns in the corresponding graph.

In this work, we present two BoG-based methods, the *Bag of Singleton Graphs (BoSG)* and the *Bag of Visual Graphs (BoVG)*, which create vector representations for graphs and images, respectively. Both methods are validated in classification tasks. We evaluate the Bag of Singleton Graphs (BoSG) for graph classification on four datasets of the IAM repository, obtaining significant results in terms of both accuracy and execution time. The method Bag of Visual Graphs (BoVG), which encodes the spatial distribution of visual words, is evaluated for image classification on the Caltech-101 and Caltech-256 datasets, achieving promising results with high accuracy scores.

Resumo

Atualmente, há uma alta demanda por soluções que possibilitem a implementação de serviços de recuperação e classificação eficazes e eficientes para grande volumes de dados. Nesse contexto, diversos estudos têm investigado o uso de novas técnicas baseadas na comparação de estruturas locais presentes em objetos na implementação de serviços de classificação e recuperação. Estruturas locais podem ser caracterizadas por diferentes tipos de relacionamentos (e.g., distribuição espacial) entre primitivas de objetos, sendo geralmente exploradas em problemas de reconhecimento de padrões.

Nessa dissertação de mestrado, propomos a *Sacola de Grafos*, uma nova abordagem baseada no modelo de *Sacola de Palavras Visuais*, que utiliza grafos para codificar estruturas locais de um objeto. Uma definição formal do modelo proposto é apresentada, assim como conceitos e regras que tornam este modelo flexível e ajustável a diferentes aplicações. Na abordagem proposta, um objeto é representado por um grafo que modela as estruturas locais existentes. Usando um dicionário pré-definido, o objeto pode ser descrito por uma representação vetorial com a frequência de ocorrência de padrões locais no grafo correspondente.

Neste trabalho, apresentamos dois métodos baseados no modelo proposto, a *Sacola de Grafos Triviais* e a *Sacola de Grafos Visuais*, que constroem representações vetoriais para imagens e grafos, respectivamente. Ambos os métodos são validados em tarefas de classificação. Nós avaliamos o método Sacola de Grafos Triviais para classificação de grafos em quatro bases do repositório IAM, obtendo resultados significativos em termos de acurácia e tempo de execução. O método Sacola de Grafos Visuais é avaliado para classificação de imagens nas bases Caltech-101 e Caltech-256, alcançando resultados promissores, com elevados valores de acurácia.

Acknowledgements

I would like to thank Prof. Ricardo Torres for the interest, encouragement, and support during my master work. I thank him for giving me this opportunity and for all the advices and teachings.

I would like to thank the persons that are always by my side. I thank my parents, José Wilson and Ilca Helena, for their dedication. The education, care, and support given by them over the years have been fundamental to my growth as a person and as a professional. I thank my sister and my brother, Priscilla and Fernando José, for the friendship, help, and advices. I also would like to thank my boyfriend, Gabriel, for the companionship, patience, and support. I wish to thank God for helping me reach this achievement in my life.

I thank Prof. Tabbone for welcoming me in his research lab, which has contributed a lot to enrich this work. I also thank direct and indirect collaborators of this project, professors, students, and researchers, who have contributed with their expertise and experience. All these collaborations were essential to my learning and to the improvement of this research work.

I wish to thank the colleagues from RECOD and other labs of the Institute of Computing - UNICAMP and LORIA - University of Lorraine for the friendship and talks.

Finally, I would like to thank FAPESP (grants 2012/16172-2 and 2013/11378-4), CNPq (grant 139135/2012-0), CAPES, and IEEE Signal Processing Society for the financial support, and AMD and Microsoft Research for the infrastructure provided to this research project.

"Learn from yesterday, live for today, hope for tomorrow. The important thing is to not stop questioning."
Albert Einstein

"O saber a gente aprende com os mestres e com os livros. A sabedoria se aprende com a vida e com os humildes."
Cora Coralina

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The number of applications from different areas of knowledge that handle large volumes of data is currently increasing. Nevertheless, the efficient use and analysis of data depends on the development of effective and efficient classification and retrieval tools. The challenge related to the development of such tools relies on the design of discriminant representation models that enable the identification of semantic similarities among particular instances.

Several digital objects do not possess a semantic meaning that can be easily identified from their content. Therefore, object recognition is a difficult task that should consider the proper characteristics of each object in order to distinguish its semantic content. An evidence of the similarity of a pair of objects is the identification 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.

Since graphs provide a flexible manner to model different types of relationships, they are useful for representing local structures within an object. Additionally, graphs are invariant to several geometric transformations, which allows the creation of robust representations. One limitation regarding the use of graph representation in large scale problems refer to the associated computational costs. The computation of graph similarity is usually an expensive task, requiring a high execution time to be accomplished [6].

In another research venue, bag-based representations have been proposed to effectively characterize the frequency of occurrence of object features. One example of successful representation is the *bag of words*. Originally, the Bag-of-Words model (BoW) creates a vector representation that describes a textual document based on the frequency of word occurrences. The adaptation of Bag of Words (BoW) for the image context [76], is called *Bag of Visual Words*, *Bag of Words* 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.

In this case, the image descriptor is created based only on the number of occurrences of some particular visual appearances within the image. Recently, different studies [33, 44] have investigated the use of spatial information in order to improve bag representations.

In summary, the BoW model proposes a simple and efficient form of representation that enables a fast computation of object similarities. Although this approach achieves good accuracy rates, the inclusion of the information about local structures into the object description process can contribute to improve the bag representation, which can lead to accurate results in classification and retrieval tasks.

We believe that both approaches, graphs and bags, are complementary, in the sense that one may contribute to each other to overcome their deficiencies. Graphs can be used for encoding local structures into a BoW-based descriptor, which would contribute to improve bag representations. At the same time, the use of BoW-based representations may contribute to reduce the amount of time required by graph-based methods to compute the similarity between objects. Therefore, our hypothesis is that by combining graphs with the BoW model, we can create a discriminant and efficient representation based on local structures of an object, which will lead to accurate results in classification tasks.

The goal of this research project is to create a novel object descriptor that combines bag and graph representations. In order to accomplish that, we propose the Bag of Graphs (BoG), a generic approach that creates a vector representation based on local structures defined by graph elements. Our method may be adapted to different contexts. In this work, we introduce two BoG-based approaches that are validated for the tasks of image and graph classification. The first approach, called Bag of Singleton Graphs (BoSG), generates a bag representation for objects that were previously modeled as graphs with attributes associated with their vertices and edges. The second one, denominated Bag of Visual Graphs (BoVG), creates BoW-based descriptors using graphs to model the spatial relationships between the visual words found within an image. Both approaches obtain good classification accuracy rates when evaluated on standard datasets [19, 27, 61], achieving comparable results to other methods of the literature.

The main contributions of this work are:

- the formal definition of a generic model for object representation;
- the definition and implementation of two different approaches based on the proposed model; and
- the validation of the proposed methods in classification tasks.

This dissertation is organized as follows. Chapter 2 introduces the main concepts and related work. Chapter 3 presents the formalization of the Bag of Graphs (BoG) model. Chapter 4 presents the description of two BoG-based approaches that we have proposed.

Chapter 5 describes the performed experiments for validating the proposed approaches and presents the main results obtained using different datasets. Finally, in Chapter 6, we discuss about the contributions and the future work of this research project.

Chapter 2

Background Concepts and Related Work

2.1 Graph Representation

Graphs are flexible structures that allow modeling different types of data. This characteristic has contributed to the development of graph-based applications on several domains, such as Chemistry, Biology, web, and image analysis [1].

A graph $G = (V, E)$ is composed of a set of vertices V and a set of edges E, where each edge of E represents a relationship between two vertices of V. Numerical or symbolic attributes may be associated with vertices and edges, which allows the definition of a graph in accordance with the characteristics of different applications.

The description of an object by means of the relationships of its components favors the use of graph representation. Let $G = (V, E)$ be a graph that represents an object O, each vertex of V can be associated with an object component and each edge of E may correspond to a particular relationship between a pair of object components.

Besides the flexibility for representing different types of objects, graphs are invariant to several transformations, such as rotation, translation, and mirroring [11]. Since these geometric transformations usually affect the application data, it is very important to adopt invariant structures, like graphs, in order to create object representations.

An image is an example of a digital object that is depicted as a graph in several works. Some of the graph representations proposed in the image context are: graph of interest points [63, 86], graph of adjacent regions [66], skeleton graphs [26, 67, 72, 73], graph of primitives [70, 88], and graph of face fiducial points [87].

2.2 Graph Matching

Graph matching algorithms enable solving complex pattern recognition problems. Thus, in several graph-based applications, it is necessary to compute the similarity of graphs. The computation of graph similarity is a highly complex problem that are usually addressed by using exact or inexact graph matching approaches.

The exact graph matching algorithms indicate if two graphs are isomorphic or not. The graph isomorphism is defined by a bijection between the elements of a pair of graphs. The complexity of exact graph matching has not yet been proven [6]. However, there are some polynomial algorithms for solving the isomorphism problem of special types of graphs [1].

Instead of only indicating whether a bijection can be defined between two graphs, the inexact graph matching approaches provide a distance value that indicates graph similarity. Different from the exact graph matching, the complexity of this problem has been proved to be NP-complete [6].

Graph Edit Distance [10] is one of the most popular methods to perform inexact graph matching. Inspired by the traditional edit distance function, which computes the similarity between two strings, this method defines the graph similarity based on some edit operations on vertices and edges. In this context, the distance between a pair of graphs corresponds to the minimum cost for converting a graph onto another one. This method provides accurate results, but it has an exponential time complexity [1]. In the literature, different Edit Distance approaches [23, 39, 62] have been proposed to compute a sub-optimal edit cost in order to reduce the computation time.

Another solution to the inexact graph matching problem relies on the use of kernel-based methods [24, 45, 84]. These methods project graphs onto a feature space, where pairs of graphs are compared by computing inner products. The effectiveness of kernel-based methods depends on the design of appropriate kernel functions, which is a complex task. Diffusion, convolution, and random walk are some examples of kernels that have been used for computing graph similarity [1].

Spectral methods [50, 64, 89] refer to a different kind of approach that uses the eigenvectors and eigenvalues of the adjacency matrix of a graph in order to compute graph similarity scores. In [49], feature vectors are extracted from the eigendecomposition of an adjacency matrix and then, embedded into an eigenvector space. In this method, a graph is described by a three-component vector in the pattern space. Another approach [79] builds a correspondence matrix with the eigenvalues obtained from the adjacency matrices of two graphs. Then, the Hungarian method [42] is applied to solve the correspondence of vertices, which allows the computation of the similarity between the corresponding pair of graphs. Since spectral methods use only the graph structure, one limitation of these

methods is that they usually can not handle vertex and edge attributes for performing graph matching.

Besides the techniques mentioned above, there are different approaches that have been proposed to solve the inexact graph matching problem, such as tree search, relaxation labelling, and artificial neural network [1, 15].

Recently, the amount of graph data is increasing quickly. However, the use of currently available methods to search and classify graphs on large datasets is very limited due to their high computational cost.

2.3 Encoding Spatial Relationships into Graphs

The flexibility afforded by graphs has motivated several studies in different contexts and applications. The use of graphs for representing spatial relationships is one of the topics that have been widely investigated in the literature.

Different graph representations have been proposed to describe spatial structures in the context of object recognition. Spatial Relational Graphs (SRGs) [88] describe symbols based on topological relationships (e.g., intersection, parallelism, and tangency) of their graphic primitives, while Attributed Relational Graphs (ARGs) have been used for modeling both topological and directional spatial relationships among graphic primitives [70]. In order to compute the spatial similarity of images, Spatial Orientation Graphs (SOGs) [28, 29] are used for describing the spatial positioning of objects within an image. In other works, skeleton graphs [32] and complete graphs [7] are employed to model the geometry defined by object parts.

Graphs have also been used along with BoW-based approaches [4, 34, 41]. Barbu et al. [4] propose a *bag of symbols* to describe a graphical document. Using a pre-defined codebook of connected components, Barbu et al. represent each document by a graph, which models the spatial structure of connected components. In their approach, frequent subgraphs are considered as graphic symbols and the BoW-based representation is obtained by counting the occurrences of frequent subgraphs within the input document.

Hou et al. [34] propose the Bag-of-Feature-Graphs (BoFG), an approach that describes a 3D-shape by a set of graphs. Each graph is represented by a matrix that describes the spatial relationships between geometric features considering their similarity to a particular word of a vocabulary. Thereby, in this approach, a shape is described by N matrices, where N corresponds to the codebook size. The similarity of a pair of 3D-shapes is then computed based on the eigenvalue similarity of their corresponding matrices.

In [41], Karaman et al. propose a multi-layer approach that exploits the spatial distribution of interest points to create BoW-based representations. In this method, called *Bag of Graph Words*, graphs are build upon a set of interest points and different layers

are defined based on the size of built graphs. A graph-based codebook is created for each layer and the final image representation indicates the frequency of graph-word occurrences on the input image.

Among the methods presented above, the Bag of Graph Words is the most similar method to our approach proposed for image classification. Besides the multi-layer component, the main difference between the Bag-of-Graph-Words method and the proposed Bag-of-Visual-Graphs method is the graph description. In Karaman's approach, the vertices are described by interest-point descriptors. In our approach, we create a visual codebook from a quantization of the feature space defined by interest-point descriptors. Then, this codebook is used for describing each vertex by a visual word. Additionally, different graph matching techniques are employed on both methods.

2.4 BoW-based Representations

The Vector Space Model (VSM) [68] is a well-known technique in the context of text retrieval that represents a document as a vector. Each feature of the document is represented by a vector dimension, whose value refers to the relevance of the feature in the document. The vector representation allows the computation of document similarity using different metrics, such as the cosine and the Euclidean distances.

Inspired by the VSM model, the Bag-of-Words (BoW) approach [3] has been proposed to represent a document by the distribution of frequency of occurrence of words. In this approach, words are considered as features and their relevance is based on their number of occurrences in the document. Since each document is represented as a collection of words, the vector representation is denominated a *bag of words*. The BoW model has been successfully adapted to different domains [12, 55, 65, 76].

2.4.1 Bag of Visual Words

In this section, we describe the BoW model in the image context [76], which can be also denominated Bag of Visual Words (BoVW). This model has been successfully used in image classification and categorization [9, 44, 59, 60], medical image screening [65], and image retrieval [12] tasks.

The BoVW propose to describe an image based on the global appearance of its local visual patterns. This approach has some advantages over the use of local [5, 48] and global descriptors [35, 77]: the final bag representation tends to be more discriminant than a global descriptor, and more general than a local descriptor. Comparing with local descriptors, the BoVW approach has also the advantage of creating a single representation to the image.

In the Bag-of-Visual-Words approach, the vocabulary is composed of *visual words*, which correspond to the main visual patterns of an image collection. Using a pre-defined visual codebook, the *bag of words* is created based on the distribution of frequency of occurrence of visual words within an image.

Regardless of the application domain, the process of creating a bag is very similar. It may only differ from one application to another with regard to the definition of the vocabulary, which will be adapted to the characteristics of each domain.

In general, the process of describing an object as a bag has the following steps.

1. **Extraction of local information**

   First, it is necessary to define an object in terms of local features. This local information will be used to build the dictionary and create the bags. For example, a text and an image are described by words and interest points, respectively. In the case of an image, the extraction of local information is usually accomplished through the use of interest point detectors, like Hessian Affine [52], and interest point descriptors, like SIFT [48].

2. **Creation of the codebook**

   After the definition of local information, the vocabulary will be created from the quantization of the feature space. The codebook will be composed of codewords, which represent the main characteristics that can be used to describe an object. In the Bag-of-Visual-Words model, the dictionary is usually created using a clustering method or a random selection of features [83].

3. **Coding**

   The next step consists in identifying the occurrences of the codewords in the object. In this step, there are two popular approaches to define how a local feature will be associated with the codewords: hard assignment, which assigns a local feature to the codeword that most resembles it, and soft assignment that uses a kernel function to determine the degree of association between local features and codewords [9, 82]. Other approaches for word assignments have been proposed in the literature [46, 82].

4. **Pooling**

   The last step summarizes the assignments in order to create the bag. By using a sum pooling, for example, the activation of each word of the vocabulary is given by the sum of all associations to it. By using an average pooling, in turn, each codeword activation is determined by the percentage of its assignments. Using a max pooling, the maximum value assigned to a codeword defines its activation [9].

Figure 2.1 illustrates the process for constructing the visual-word codebook and a bag of visual words.

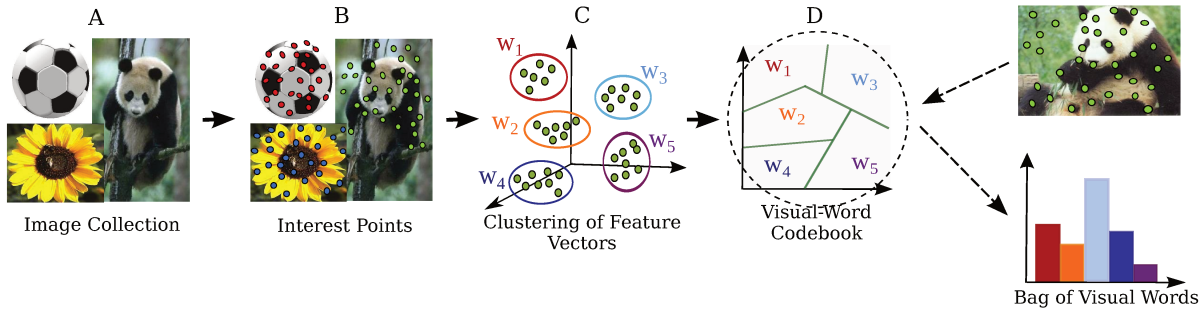


Figure 2.1: **Overview of the Bag of Visual Words.**
From a set of images (A), we detect interest points (B). Then, we apply a clustering method on the interest-point descriptors (C) to generate a codebook (D). Using this codebook, we compute the distribution of frequency of occurrence of visual words within an image and create the corresponding *Bag-of-Visual-Words* descriptor.

In Chapter 3, we present a formal definition of the steps described above. In [58], Penatti provides a more detailed description of the BoVW model and discusses its use in classification and retrieval tasks.

2.4.2 Encoding Spatial Relationships into BoW-based representations

Different from textual words, *visual words* do not possess a semantic meaning directly related to them. This characteristic leads to a problem known as *semantic gap*, which states that the visual similarity between a pair of images does not necessarily correspond to a semantic similarity. Thus, it is important to aggregate different types of information that may contribute to identify correlations between the visual and the semantic contents of an image.

Several works of the literature [12, 33, 44, 53, 71, 76] have proposed to include the spatial information into the BoW description process.

In [33, 71, 76], the visual vocabulary is created based on the co-occurrence of groups of visual words. Sivic et al. [76] 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 distribution of frequency of occurrence of *doublets*. Savarese et al. [71] define a codebook of correlograms of visual words, which is used for creating the bag representation. In [33], 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.

There are other approaches that compute a bag for different regions of the image. Then, the image descriptor is created from the combination of these bags. The Spatial-Bag-of-Features [12] is an example of this kind of approach. In [12], Cao et al. define image regions 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) [44] is one of the most famous BoW-based approaches. This method hierarchically partitions the image into cells. Each cell is described by a *bag of visual words*, and the final descriptor corresponds to the weighted concatenation of bags of the image cells. The Word Spatial Arrangement (WSA) [59, 60] divides the image into quadrants, considering each interest point as an origin for partitioning. Through these partitions, a histogram is constructed with the frequency of occurrence of the visual words in each of the four relative positions.

In the literature, the spatial information was also considered for some generative models. In [78], a graphical model is created with the information about the visual appearance of interest points and their relative positions. In [53], the BoW is combined with the Constellation model [20, 51]. In that way, the parts of an object is described with a *bag of words* and the object geometry is modelled through the spatial relationships of object parts.

Recently, several works have proposed to include the spatial information within the BoW model for scene and image classification [8, 47, 90]. In [8], Bolovinou et al. proposed to represent the spatial information through correlograms of visual words. This method, called *Bag of Spatio-Visual Words*, defines a vocabulary of log-polar descriptors, which encode the frequency of visual-word occurrences in particular regions of the image. Then, this vocabulary is used for creating the final BoW-based representation. In [47], the spatial arrangement of visual words are described by strings, which correspond to sequences of visual words within interest-point neighborhoods. In this work, the similarity of a pair of images is computed based on the difference of the strings whose corresponding interest points were assigned to the same visual word. In [90], Zhou et al. define vertical and horizontal regions on three different image 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. The image similarity is measured using a kernel that combines the similarity obtained from each level of resolution.

Chapter 3

Formalism

In this chapter, we present the formalization of the Bag-of-Graphs (BoG) model. First, in Section 3.1, we introduce the main concepts related to the model. In Section 3.2, we present the formal definition of the BoG concepts. Then, in Chapter 4, we present some approaches that illustrate the use of BoG model for different applications.

The content of this chapter may refer to some mathematical definitions or concepts introduced in [16, 21, 39]. Appendix A contains the basic concepts used in this chapter.

3.1 Overview of the Bag-of-Graphs concepts

The Bag of Graphs (BoG) corresponds to a description process that creates a vector representation based on the local relationships within an object. Our model is defined by a composite function, denominated *bag extraction*, which combines the following functions: graph extraction, graph-of-interest detector (GoI detector), assignment, pooling, and the feature extraction functions associated with vertex, edge, and graph descriptors.

The *graph extraction* function is used for extracting the intrinsic structure of a *digital object*. This structure is represented by 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, which means select 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 of *AllTypes* domain, which is composed of simple and complex datatypes. 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, called *bag*, is created by a *pooling* function that summarizes the performed assignments, which are represented by a set of vectors called *coding*.

3.2 Formalization of the Bag-of-Graphs model

In this section, we present the formal definition of the concepts related to the BoG model. Figure 3.1 shows a map of the relationships between the concepts that will be introduced in this section.

Definition 1. A **digital object** is a tuple $DO = (h_{DO}, \mathcal{SM}, \mathcal{ST}, \mathcal{F}_{strStream})$, such that

- h_{DO} is a set of universally unique handles (label).
- $\mathcal{SM}$ is a set of streams.
- $\mathcal{ST}$ is a set of structural metadata specifications.
- $\mathcal{F}_{strStreams}$ is a set of structuredStream functions that associate a stream $s \in \mathcal{SM}$ with a structural metadata specification $m \in \mathcal{ST}$.

A stream is a sequence of elements; a structural metadata specification is a structure, tuple composed of a graph (see Definition A.1 in the Appendix), a set of literals and labels, and a set of functions that specifies the relationships among digital object components; and a structuredStream is a function that associates a structure with a stream. These concepts were introduced in [21], where a more detailed discussion about digital library elements can be found.

Definition 2. Given a digital object $DO = (h_{DO}, \mathcal{SM}, \mathcal{ST}, \mathcal{F}_{strStream})$, a **digital object element** is a tuple $DOE = (\mathcal{SM}', \mathcal{ST}', \mathcal{F}'_{strStream})$ that respects the following constraints.

- $\mathcal{SM}' \subset \mathcal{SM}$
- $\mathcal{ST}' \subset \mathcal{ST}$
- $\mathcal{F}'_{strStream} \subseteq \mathcal{F}_{strStream}$

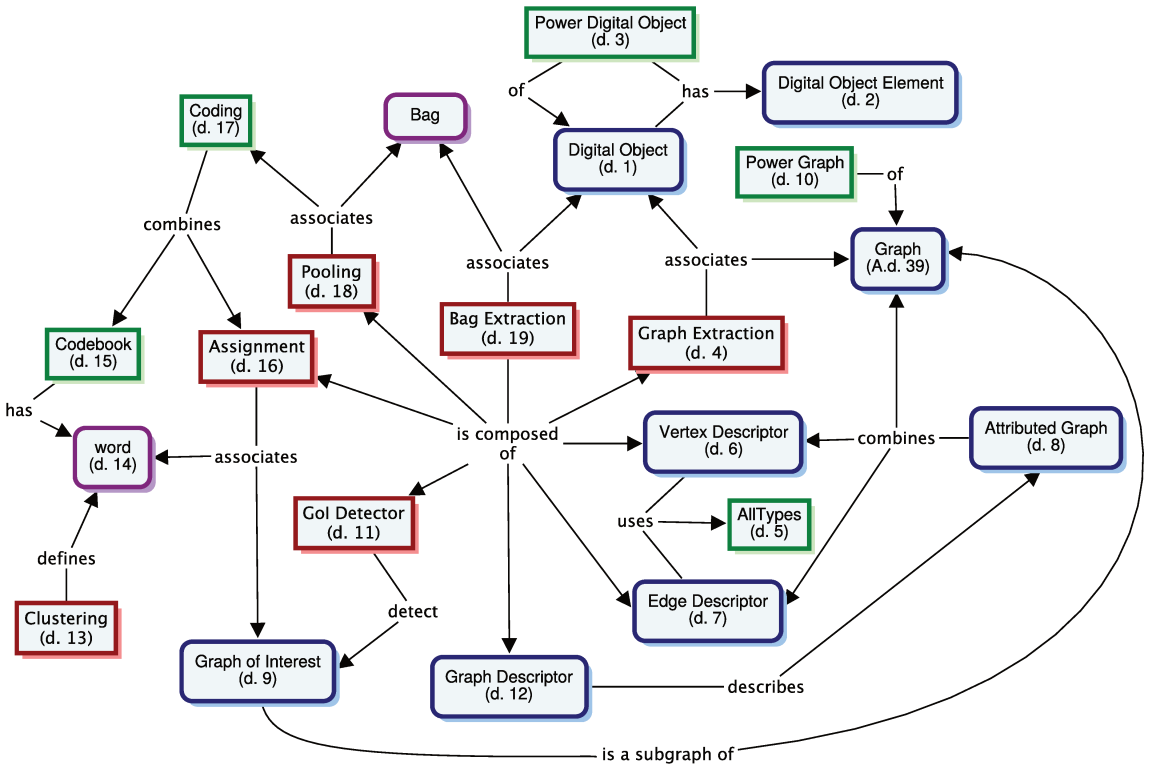


Figure 3.1: Concept map of the Bag-of-Graphs model. The colors of the squares indicate the type of the concept: blue refers to the definition of particular tuples, red corresponds to function definitions, green refers to particular set definitions, and purple corresponds to specific representation elements. Each concept refers to a definition of this section, denoted (d. #).

Definition 3. The **power digital object**, denoted $\mathcal{P}(DO)$, is the set of all possible digital object elements of a given digital object DO.

Definition 4. Let DO be a digital object and $G = (\mathcal{V}, \mathcal{E})$ be a graph, a **graph extraction** is a function $(\mathcal{V} \bigcup \mathcal{E}) \rightarrow \mathcal{P}(DO)$ that associates a vertex of $\mathcal{V}$ or an edge of $\mathcal{E}$ with a digital object element of DO.

Figures 3.2 and 3.3 illustrate examples of graph extraction functions for two different digital objects.

Definition 5. ***AllTypes*** is a set $\mathcal{T}$ defined as the union of the domains of simple and complex datatypes. Literals (e.g., numbers, strings, and date) are simple datatypes, and complex datatypes are created through the composition of different simple datatypes, like an array.

Definition 6. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph, a **vertex descriptor** is a tuple $d_v = (\epsilon, \delta)$, where:

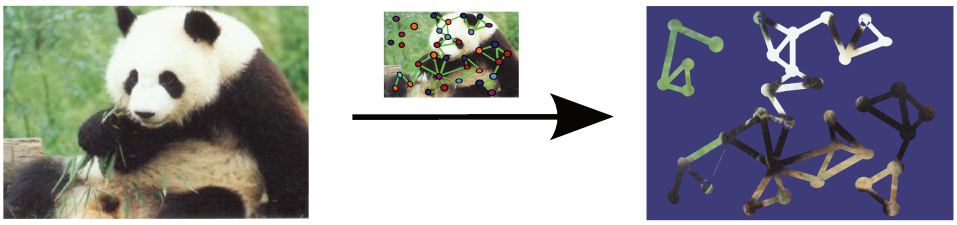


Figure 3.2: A graph extraction function that detects some interest points of an image and builds the corresponding graph based on the spatial relationships among these points.

```
<?xml version="1.0"?>
<gxl>
<graph id="metanol" edgeids="true" edgemode="undirected">
<node id="node1"><attr name="chem"><string>H</string></attr></node>
<node id="node2"><attr name="chem"><string>H</string></attr></node>
<node id="node3"><attr name="chem"><string>H</string></attr></node>
<node id="node4"><attr name="chem"><string>C</string></attr></node>
<node id="node5"><attr name="chem"><string>O</string></attr></node>
<node id="node6"><attr name="chem"><string>H</string></attr></node>
<edge from="node1" to="node4"><attr name="valence"><int>1</int></attr></edge>
<edge from="node2" to="node4"><attr name="valence"><int>1</int></attr></edge>
<edge from="node3" to="node4"><attr name="valence"><int>1</int></attr></edge>
<edge from="node4" to="node5"><attr name="valence"><int>1</int></attr></edge>
<edge from="node5" to="node6"><attr name="valence"><int>1</int></attr></edge>
</graph>
</gxl>
```

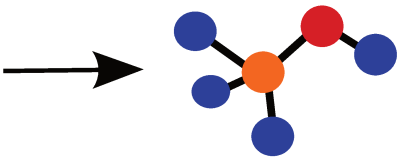


Figure 3.3: A graph extraction function that encodes the intrinsic structure of a molecule into a graph representation. In this example, the blue, red, and orange circles refer to hydrogen, carbon, and oxygen atoms, respectively.

- $\epsilon : \mathcal{V} \to \mathcal{T}$ is a function that associates a vertex v of $\mathcal{V}$ with an element of $\mathcal{T}$, called *vertex attribute*.
- $\delta : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$ is a function that computes the similarity between a pair of vertices based on the distance, computed by a distance function (Definition A.3), of their corresponding attributes.

Definition 7. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph, an **edge descriptor** is a tuple $d_v = (\epsilon, \delta)$, where:

- $\epsilon : \mathcal{E} \to \mathcal{T}$ is a function that associates an edge e of $\mathcal{E}$ with an element of $\mathcal{T}$, called an *edge attribute*.
- $\delta : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$ is a function that computes the similarity between a pair of edges based on the distance, computed by a distance function, of their corresponding attributes.

Definition 8. An **attributed graph** is a tuple $\hat{G} = (G, \mathcal{T}_D, \mathcal{D}_v, \mathcal{D}_e)$, where $G = (\mathcal{V}, \mathcal{E})$ is a graph, $\mathcal{D}_v$ is a set of vertex descriptors, $\mathcal{D}_e$ is a set of edge descriptors, and $\mathcal{T}_D$ is a subset of $\mathcal{T}$, defined as the union of the domains of vertex and edge attributes.

Definition 9. Given a graph G, a **graph of interest (GoI)** is a subgraph of G that satisfies a determined property P.

Definition 10. The **power graph** of a graph G, denoted $\mathcal{P}(G)$, is the set of all possible subgraphs of G.

Definition 11. Let G be a graph, $\mathcal{P}(G)$ be the power graph of G and $\mathcal{I}$ be a set of graphs of interest within G, a **graph of interest (GoI) detector** $\mathbb{D}$ is a characteristic function (Definition A.4) $\mathbf{1}_{\mathcal{I}} : \mathcal{P}(G) \rightarrow \{0, 1\}$ that indicates if a determined subgraph of G is a graph of interest.

Definition 12. Let $\mathcal{G}$ be a set of attributed graphs and $\mathscr{T}$ be a complex datatype domain, a **graph descriptor** is a tuple (ϵ, σ), where:

- $\epsilon : \mathcal{G} \rightarrow \mathscr{T}$ is a function that associates an attributed graph with an element of $\mathscr{T}$, obtained by means of the combination of vertex and edge attributes.
- $\sigma : \mathscr{T} X \mathscr{T} \rightarrow \mathbb{R}$ is a function that computes the similarity between two attributed graphs as a combination of the similarity values obtained from vertex and edge descriptors.

Definition 13. Let $\mathscr{T}$ be a subset of $\mathcal{T}$, a **clustering** $\mathscr{C}$ is an equivalence relation (Definition A.5) that defines a partition (Definition A.6) of $\mathscr{T}$ based on the similarity of its elements. Each subset of $\mathscr{T}$ defined under $\mathscr{C}$ is denominated **cluster**.

Mean Shift [14] and K-means [30] are some examples of algorithms that define a Clustering relation.

Definition 14. Given a clustering $\mathscr{C}$, a **word** is an element $e \in \mathcal{T}$ that represents the prototype of an equivalent class defined by $\mathscr{C}$.

For example, the *centroids* (Definition A.9) of the clusters may be defined as words. Figure 3.4 illustrates the words of each cluster defined by a clustering relation.

Definition 15. A **codebook** $\mathfrak{C} = \{w_1, w_2, ..., w_{|\mathfrak{C}|}\}$ is a set of words (dictionary).

Definition 16. Let $\mathcal{G} = \{g_1, g_2, \ldots, g_{|\mathcal{G}|}\}$ be a set of attributed graphs and $\mathfrak{C} = \{w_1, w_2, ..., w_{|\mathfrak{C}|}\}$ be a codebook. **Asssignment** is a function that defines an activation value for each pair (g_i, w_j), where $g_i \in \mathcal{G}$ and $w_j \in \mathfrak{C}$.

Let $\mathcal{D} = (\epsilon, \delta)$ be a graph descriptor, the hard and soft assignment functions can be defined as follows:

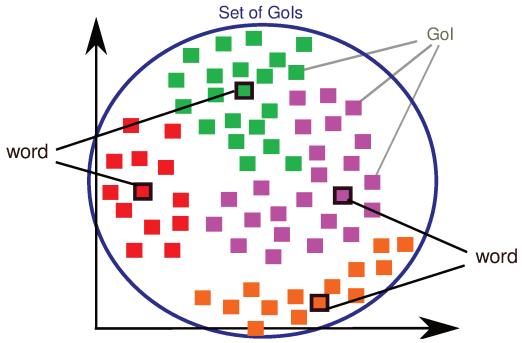


Figure 3.4: Words that describe the elements of a set $\mathcal{T}$ under a clustering relation.

- Using a *hard assignment* function, each $g_i \in \mathcal{G}$ activates only one word of $\mathfrak{C}$. The assignment function $f_{assign} : \mathcal{G} \times \mathfrak{C} \rightarrow \{0, 1\}$ is denoted as

$$f_{assign}(g_i, w_j) = \begin{cases} 1 & \text{if } w_j = \underset{w_k \in \mathfrak{C}}{argmin}\ \delta(\epsilon(g_i), w_k) \\ 0 & otherwise \end{cases} . \tag{3.1}$$

- 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.

  Germet et al. [82] propose the following assignment function.

$$f_{assign}(g_i, w_j) = \frac{K_\sigma(\delta(\epsilon(g_i), w_j))}{\sum_{k=1}^{|\mathfrak{C}|} K_\sigma(\delta(\epsilon(g_i), w_k))}. \tag{3.2}$$

  where K_σ is a normalized Gaussian kernel defined as

$$K_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \tag{3.3}$$

Definition 17. Let $\mathcal{G} = \{g_1, g_2, \ldots, g_{|\mathcal{G}|}\}$ be set of attributed graphs, $\mathfrak{C} = \{w_1, w_2, ..., w_{|\mathfrak{C}|}\}$ be a codebook, and f_{assign} be an assignment function. **Coding** is a set $C = \{c_1, c_2, \ldots, c_{|\mathcal{G}|}\}$, where c_i is a vector such that $c_i[j] = f_{assign}(g_i, w_j)$, $1 \leq i \leq |\mathcal{G}|$ and $1 \leq j \leq |\mathfrak{C}|$.

Definition 18. Given a set of coding $\mathcal{C}$, **pooling** is a function $\mathcal{C} \rightarrow \mathbb{R}^N$ that summarizes all word assignments, defined in a coding $C \in \mathcal{C}$, into a numerical vector representation.

Let $\mathcal{G}$ be a set of attributed graphs and $C = \{c_1, c_2, \ldots, c_{|\mathcal{G}|}\}$ be the corresponding coding defined according to a codebook $\mathfrak{C}$. Different pooling functions can be adopted to describe $\mathcal{G}$ with a vector representation.

- *Sum pooling*:

$$f_{pool}(C) = \left\{ \vec{v} \,\middle|\, \left(\forall k \in \left[1, |\mathfrak{C}|\right]\right) \left[\vec{v}_k = \sum_{i=0}^{i<|\mathcal{G}|} c_i[k] \right] \right\}. \tag{3.4}$$

- *Average pooling*:

$$f_{pool}(C) = \left\{ \vec{v} \,\middle|\, \left(\forall k \in \left[1, |\mathfrak{C}|\right]\right) \left[\vec{v}_k = \frac{1}{|\mathcal{G}|} \sum_{i=0}^{i<|\mathcal{G}|} c_i[k] \right] \right\}. \tag{3.5}$$

- *Max pooling*:

$$f_{pool}(C) = \left\{ \vec{v} \,\middle|\, \left(\forall k \in \left[1, |\mathfrak{C}|\right]\right) \left[\vec{v}_k = \underset{0<i<|\mathcal{G}|}{max} \left(c_i[k]\right) \right] \right\}. \tag{3.6}$$

Definition 19. Given a collection of digital objects $\mathcal{O}$, a **bag extraction** is a function $\mathcal{O} \rightarrow \mathbb{R}^{\mathrm{N}}$ that associates a digital object with a vector representation (bag). Let DO be a digital object of $\mathcal{O}$ and $\mathfrak{C}$ be a codebook, the bag extraction is defined as the composition of the following functions and concepts:

- Using a *graph extraction* function, the digital object DO is represented by a graph G. This function associates digital object elements of $\mathcal{P}(DO)$ with the vertices and edges of G.
- *Vertex and edge descriptors* associate attributes with edges and vertices of G, defining an attributed graph $\hat{G}$ to represent DO.
- Applying a *GoI detector* on $\hat{G}$, a set of graphs of interest $\mathfrak{G}$ is extracted. The GoIs of $\mathfrak{G}$ correspond to the relevant local structures of $\hat{G}$.
- A *graph descriptor* is used for describing each GoI detected from $\hat{G}$.
- An *assignment* function defines the association between the GoIs of $\mathfrak{G}$ and the words of $\mathfrak{C}$.
- A *pooling* function creates a vector representation (*bag*) by summarizing the word assignments. Thus, the final representation of DO is created based on the assignments of $\mathfrak{G}$ elements to the words of $\mathfrak{C}$.

Figure 3.5 illustrates the concepts related to the Bag-of-Graphs model. Figure 3.6, in turn, shows, in the red rectangle, the concepts embedded in the bag extraction function, which was one of the functions illustrated in Figure 3.5.

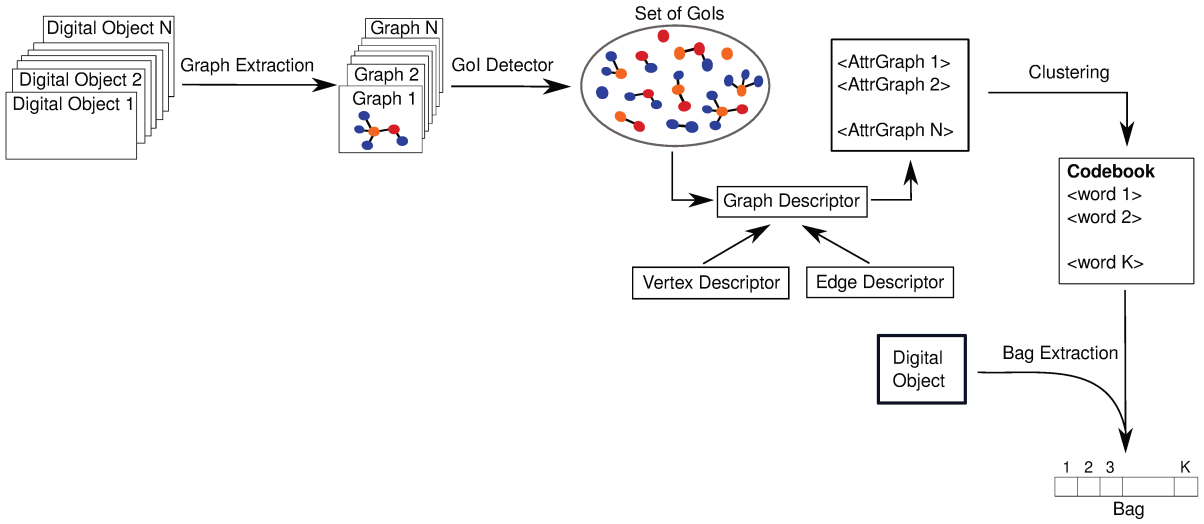


Figure 3.5: Concepts related to the Bag-of-Graphs Model (BoG).

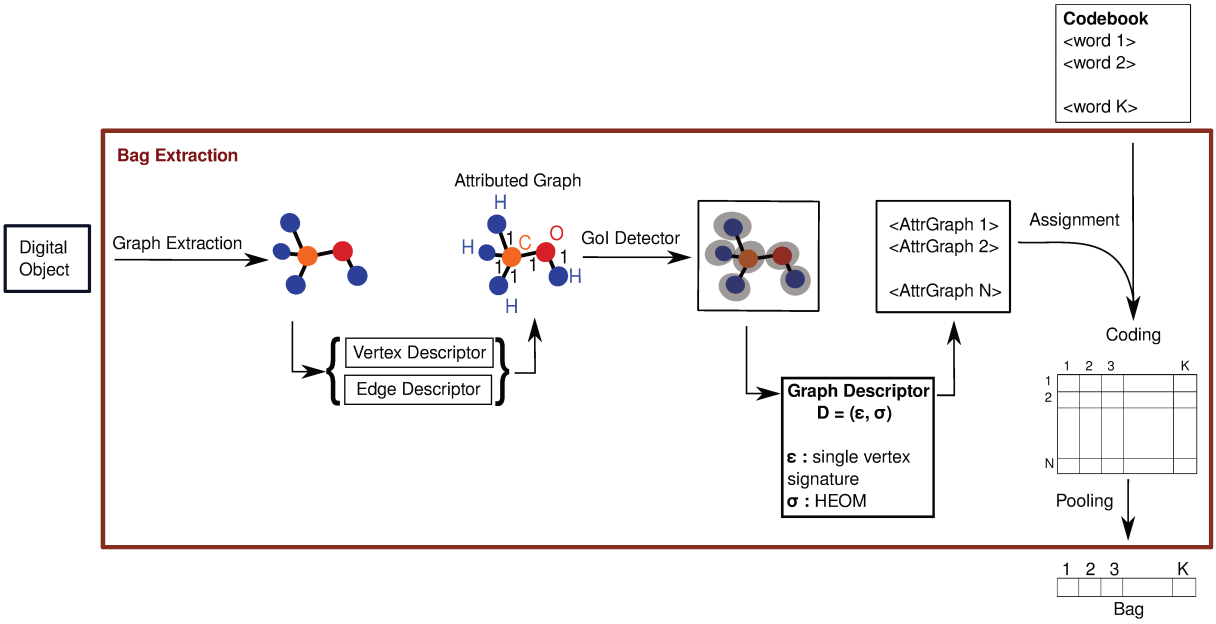


Figure 3.6: Concepts and definitions related to the bag extraction function.

Chapter 4

Approaches based on Bag-of-Graphs model

This chapter introduces two novel approaches for encoding local patterns by means of a *bag of graphs*.

4.1 Bag of Singleton Graphs

Graphs can model different types of objects, such as molecules, images, and architectural symbols [61]. In this section, we introduce the **Bag of Singleton Graphs (BoSG)**, a BoG-based approach that proposes a bag representation to describe objects that were previously modeled as graphs.

Figure 4.1 illustrates the sequence of steps used for creating a BoSG representation.

The scenario used for presenting the BoSG approach refers to the description of molecules. A molecule is a *digital object* that contains streams of atoms and chemical bounds, whose positioning in the space defines the molecule's geometry. The relevance of the spatial relationships between atoms justifies the use of graphs for representing this kind of digital object.

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

Each attributed graph $\hat{G} \in \mathcal{G}$ is a tuple $((\mathcal{V}, \mathcal{E}), \mathcal{T}, \{chem\}, \{valence\})$ such that

- $\mathcal{T} = \mathcal{A} \bigcup \mathbb{N}$, where $\mathcal{A}$ is a set of strings that identify atom symbols, such as "C", "H", "O", etc.
- *chem* is a vertex descriptor defined as $(\epsilon_{chem}, \sigma_{chem})$, where
  - $\epsilon_{chem} : \mathcal{V} \rightarrow \mathcal{A}$ is a function that associates a vertex of $\mathcal{V}$ with an atom symbol.

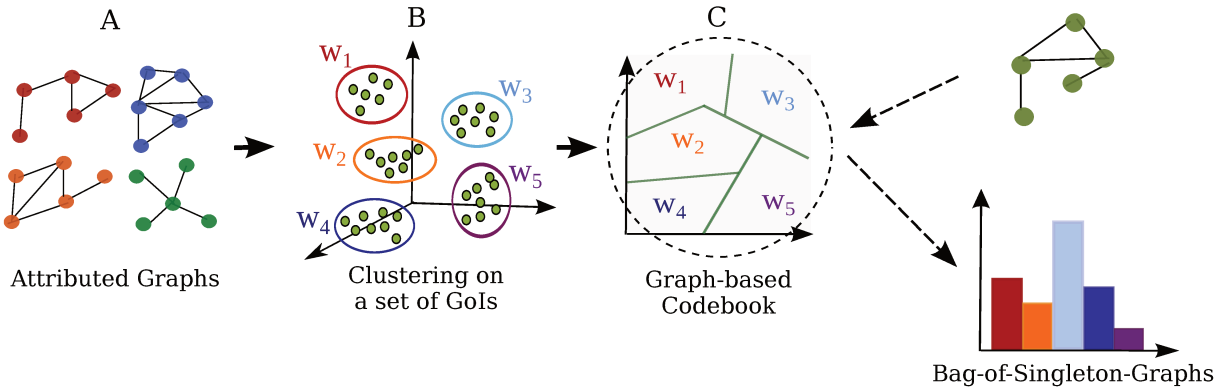


Figure 4.1: **Overview of the Bag of Singleton Graphs.**
First, we describe a set of graphs (A) in terms of vertex signatures. Then, we apply a clustering method (B) to build the codebook (C). Using this codebook, we count the codeword occurrences within each graph to create the corresponding bag representation.

  - $\sigma_{chem} : \mathcal{A} \times \mathcal{A} \to \mathbb{R}$ is the discrete distance function (see Equation A.3 in the Appendix).

- *valence* is an edge descriptor defined as $(\epsilon_{valence}, \sigma_{valence})$, where

  - $\epsilon_{valence} : \mathcal{E} \to \mathbb{N}$ is a function that associates an edge of $\mathcal{E}$ with a number of valence.

  - $\sigma_{valence} : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ is a function that computes the absolute difference between edge attributes.

Our approach proposes to represent a graph by the frequency of occurrence of its local structures, which correspond to graphs of interest (GoIs). In order to extract a set of GoIs from an attributed graph $\hat{G} \in \mathcal{G}$, we use a GoI detector $\mathbb{D}$ that identifies the minimum subgraphs comprising a vertex neighborhood.

Definition 20. Given a graph $G = (\mathcal{V}, \mathcal{E})$, the **neighborhood of a vertex** $v \in \mathcal{V}$ is composed of the vertex v and all edges e of $\mathcal{E}$ linked to it.

$$(\forall v \in \mathcal{V})\,[\, N(v) = \{v\} \bigcup E\,]\ , \text{where}$$

$$E = \{\, (v_i, v_j) \mid ((v_i, v_j) \in \mathcal{E}) \wedge\ ((v_i = v) \vee (v_j = v))\,\}$$

Definition 21. Let $\hat{G} = ((\mathcal{V}, \mathcal{E}), \mathcal{T}, \mathcal{D}_v, \mathcal{D}_e)$ be an attributed graph and N be the neighborhood of a vertex $v \in \mathcal{V}$, a **vertex signature** of v is a sequence of elements of $\mathcal{T}$ associated with the components of N.

Thus, a GoI, identified by applying $\mathbb{D}$ on a graph $\hat{G} = ((\mathcal{V}, \mathcal{E}), \mathcal{T}, \{chem\}, \{valence\})$, 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 $\mathscr{T}$ be a set of vertex signatures, a graph descriptor $\mathscr{D} = (\epsilon, \sigma)$ is defined as follows:

- ϵ is a function $\mathbb{G} \to \mathscr{T}$ that associates an attributed graph $g \in \mathbb{G}$ with a single vertex signature $\mathscr{S} \in \mathscr{T}$. 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.

  In each vertex signature $\mathscr{S}$, the edges are sorted by their attribute values. Let $L = [AE_1, AE_2, ..., AE_D]$ be the list of edge attributes, the edges of $\mathscr{S}$ are sorted in order of increasing values of AE_1. If two edges have the same value for an attribute AE_i, their order is determined by the values of the following attribute AE_{i+1} in L.

- $\sigma : \mathscr{T} X \mathscr{T} \to \mathbb{R}$ is a function that computes the similarity between two attributed graphs by applying the Heterogeneous Euclidean Overlap Metric (HEOM) [85] on the similarity values obtained from vertex and edge descriptors.

In the scenario proposed in this section, the functions ϵ and σ are defined as:

Let $g_i = ((\mathcal{V}, \mathcal{E}), \mathcal{T}, \{\text{chem}\}, \{\text{valence}\})$ be the representative graph of the neighborhood of a vertex $v_i \in \mathcal{V}$ and e_{ij} be an edge of $\mathcal{E}$.

$$\epsilon(g_i) =< \epsilon_{chem}(v_i),\ degree_{v_i},\ \epsilon_{valence}(e_{i1}),\ \epsilon_{valence}(e_{i2}), ...,\ \epsilon_{valence}(e_{in}) >$$

$$\sigma(\epsilon(g_1), \epsilon(g_2)) = \sqrt{(\sigma_{chem}(v_1, v_2))^2 + (S_e(\epsilon(g_1), \epsilon(g_1)))^2 + (P_e(g_1, g_2))^2}\,, \tag{4.1}$$

where

$$S_e(\epsilon(g_1), \epsilon(g_2)) = \sum_{i=1}^{\min\limits_{\{v_1,v_2\}}(degree)} \frac{(\sigma_{valence}(e_{1i}, e_{2i}))^2}{max(\sigma_{valence})} \tag{4.2}$$

$$P_e(g_1, g_2) = \sum_{i=\min\limits_{\{v_1,v_2\}}(degree)}^{\max\limits_{\{v_1,v_2\}}(degree)} 1 \tag{4.3}$$

Let $\mathscr{G}$ be the set of all GoIs extracted from $\mathcal{G}$, defined as $\mathscr{G} = \bigcup_{i=0}^{i<|\mathcal{G}|} \mathbb{G}_i$, and $\mathbb{S}_G$ be the set of vertex signatures associated with $\mathscr{G}$ under the function ϵ. We use a clustering

relation on $\mathbb{S}_G$ to create a codebook $\mathfrak{C}$, 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 with $\mathbb{D}$ from an attributed query graph Q and $\mathbb{S}_Q$ be the corresponding set of vertex signatures obtained with $\mathscr{D}$. Different coding and pooling functions can be employed to create a *bag of singleton graphs* that represents 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 different approaches from the literature. 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 BoSG does not depend on the number of vertices.

Besides, the methods based on the edit distance approach usually require the search of the optimal combination of parameters. The Bipartite Graph Matching [62] requires three parameters related to the cost of edit operations on vertices and edges, while, for example, BoSG requires at most one parameter, which is related to 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. In this sense, one 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 proposes the use of 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.

Figure 4.2 informally describes the steps for generating the proposed visual-graph codebook and the final image descriptor.

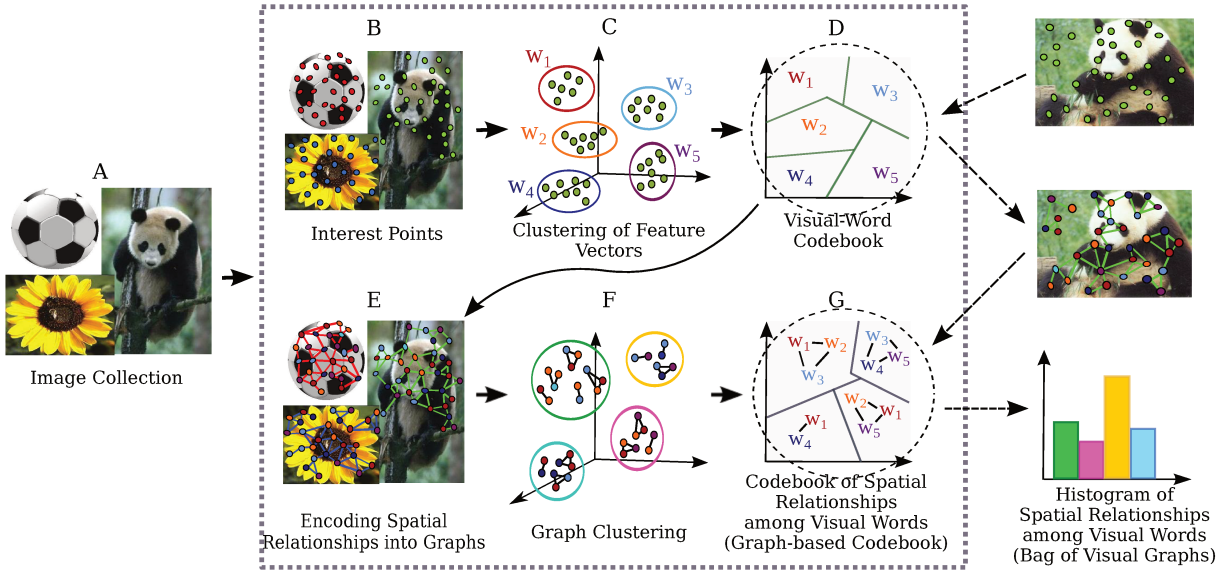


Figure 4.2: **Overview of the Bag of Visual Graphs.**
From an image collection (A), we detect all interest points (B). Then, we cluster the descriptors of the interest points in feature space (C), and generate the *visual-word* codebook (D) from the prototypes of the clusters. Using this codebook and a Delaunay triangulation on the interest points of each image, we build a set of connected graphs (E) to represent the image, which encode 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 process to generate 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.

4.2.1 Visual-Word Codebook

This section describes the process of creating a traditional visual-word codebook.

An interest-point detector, like Hessian Affine and Harris Laplace, 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.

In order to describe the visual content of an image, interest-point descriptors may be employed as vertex descriptors. In that case, an image I would be represented by an attributed graph $\hat{G} = ((\mathcal{V}, \mathcal{E}), \mathbb{R}^N, \{\text{SIFT}\}, \emptyset)$, where $SIFT$ is a vertex descriptor defined as $(\epsilon_{sift}, \sigma_{sift})$, such that

- $\epsilon_{sift} : \mathcal{V} \rightarrow \mathbb{N}^N$ is a function that associates a vertex with a feature vector [48].
- $\sigma_{sift} : \mathbb{N}^N \times \mathbb{N}^N \rightarrow \mathbb{R}$ is a function that computes the similarity between two interest points using the Euclidean distance (Equation A.1) between their corresponding feature vectors.

Each interest point of an image corresponds to a relevant local information, being defined as a graph of interest (GoI). Thus, 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 $\mathbb{G}$.

A clustering relation may be applied on $\mathcal{F}$ in order to partition $\mathbb{G}$ with respect to the visual similarity of GoIs. From the definition of 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* $G = (\mathcal{V}, \mathcal{E}, \phi)$ (Definition A.10), 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 $\mathcal{E} \rightarrow \mathbb{R}$ 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 $f_{delaunay} : \mathcal{P}(I) \rightarrow \mathcal{V} \bigcup \mathcal{E}$ that specifies the following rules:

- The vertices of V correspond to interest points, which can be detected with a sparse or dense sampling technique.
- The edges of $\mathcal{E}$ are defined by applying a Delaunay Triangulation on $\mathcal{V}$.

  Similar to the work proposed by Hashimoto [31], edges are pruned based on their weights. Edges with low weights are removed because they encode relationships between close points, and therefore they are not useful to define spatial arrangements of visual cues. Edges with high weights are also removed as they are associated with non-local structures. However, since we want that each image contains at least one spatial arrangement of visual patterns, these constraints are relaxed when an image does not have a sufficient number of interest points. The limits of edge size may be defined empirically based on the size of the images.

We propose to represent an image based on the spatial relationships of visual words. Thereby, in order to describe the image graphs extracted with $f_{delaunay}$, we propose to 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 $\hat{G} = (G, \mathcal{T}, \{\text{VW}\}, \{\text{LBP}\})$, such that

- $G = (\mathcal{V}, \mathcal{E})$ is a graph defined under the graph extraction function.
- $\mathcal{T} = \mathcal{L} \bigcup \mathbb{R}^{\text{N}}$, where $\mathcal{L}$ is a set of labels.
- VW is a vertex descriptor defined as $(\epsilon_{vw}, \sigma_{vw})$, where
  - Let $\mathfrak{C}$ be the visual codebook previously introduced, ϵ_{vw} is a composite function that combines an assignment and a labelling 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 labelling function associates v with the corresponding label of the assigned visual word.
  - $\sigma_{vw} : \mathcal{L} \rightarrow \mathbb{R}$ 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 (Equation A.3).
- LBP [54] is an edge descriptor defined as $(\epsilon_{lbp}\sigma_{lbp})$, where
  - ϵ_{lbp} is a function $\mathcal{E} \rightarrow \mathbb{R}^{\text{N}}$ 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.

    Figure 4.3 illustrates an example of a region described with ϵ_{lbp}. The dark-blue pixels represent two vertices $v_1, v_2 \in \mathcal{V}$ and the light-blue area represents the region of an edge $e \in \mathcal{E}$, being $e = (v_1, v_2)$. In order to create the LBP descriptor, we compute, for each window 3×3 in the blue area, the brightness transitions between the central pixel of the window and its adjacent pixels. The blue area is then described by a normalized histogram of size 10 that counts the occurrences of binary patterns. Non-uniform patterns correspond to windows with three or more brightness transitions, and they are associated with a single pattern located at the 9th position of the histogram. On the other hand, uniform patterns correspond to windows with two or less transitions, and they refer to nine different patterns, whose positions in the histogram correspond to the total number of positive brightness transitions.
  - $\sigma_{lbp} : \mathbb{R}^{\text{N}} \times \mathbb{R}^{\text{N}} \rightarrow \mathbb{R}$ is a function that determines the similarity between two edges by computing the Manhattan Distance (Equation A.2) between their corresponding feature vectors.

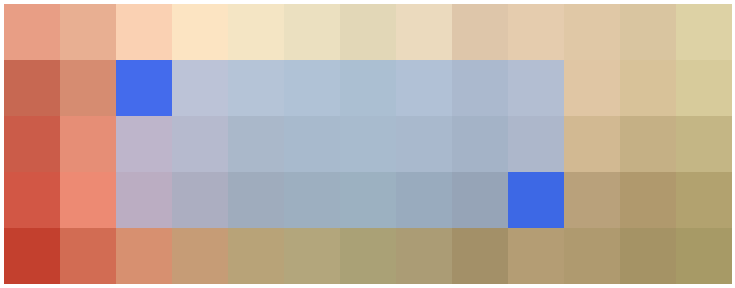


Figure 4.3: Example of region described with ϵ_{lbp}.

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.

In this context, 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. Therefore, the detected GoIs on $\hat{G}$ correspond to 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 described with a graph descriptor $\mathscr{D} = (\epsilon, \sigma)$, where:

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

$$S(v_i) =< \epsilon_{VW}(v_i),\ degree_{v_i},\ \epsilon_{LBP}(e_{i1}),\ \epsilon_{LBP}(e_{i2}), ...,\ \epsilon_{LBP}(e_{in}) >,$$

  where $v_i \in \mathcal{V}_i$ and $e_{ij} \in \mathcal{E}_i$.

- $\sigma : \mathcal{S}^N X \mathcal{S}^N \rightarrow \mathbb{R}$ is a function that computes the similarity between two graphs using the Equation 4.4, proposed by Jouili et al. [39].

$$\sigma(\epsilon(g_1), \epsilon(g_2)) = \frac{\bar{C}}{|C|} + ||g_1| - |g_2||, \tag{4.4}$$

  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 optimum matching cost of a pair of graphs is computed by applying the *Hungarian method* [43], an algorithm that solves the assignment problem in polynomial

time. 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 $\delta : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}$ (Equation 4.5) that computes the balanced sum of all similarity values obtained for each term of the vertex signatures. The sum is balanced in the sense that all terms have the same weight (importance), and for that, all similarity values are normalized in the range $[0, 1]$.

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

$$\delta(S(v_1), S(v_2)) = \sigma_{VW}(v_{i1}, v_{i2}) + S_e(S(v_1), S(v_2)) + P_e(v_1, v_2), \tag{4.5}$$

where P_e is defined in Equation 4.3, and

$$S_e(S(v_1), S(v_2)) = \sum_{i=1}^{\substack{min \\ \{v_1,v_2\}}(degree)} \frac{\sigma_{LBP}(e_{1i}, e_{2i})}{|\epsilon_{lbp}(e_i)|} \tag{4.6}$$

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, for all vertex signatures, 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,the edges attributes are set respecting the counterclockwise direction of vertices, while the edges attributes of g_2 are set with respect to the 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 applied on matrix C_i. The use of the Hungarian Method on both matrices aims at handling reflection transformations.

Let $\mathscr{S}$ be the set of vertex signature arrays that describes $\mathbb{G}_{\triangle}$. We propose to create 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 spatial arrangements of visual words. In order to create the

graph-based codebook, clustering methods [1, 38, 40] or a simple random selection can be employed for defining groups of graphs.

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

Chapter 5

Validation

5.1 Introduction

In this chapter, we present the experiments used for validating the proposed method BoG. This validation is accomplished by evaluating the BoG-based approaches introduced in Chapter 4. Sections 5.2 and 5.3 describe the experiments related to the use of Bag of Singleton Graphs (BoSG) and Bag of Visual Graphs (BoVG) approaches, respectively.

5.2 Bag of Singleton Graphs

The Bag of Singleton Graphs (BoSG), presented in Section 4.1, creates a bag representation based on the local structures of a graph. Since each graph is represented as a feature vector, the graph matching problem is reduced to the problem of computing the similarity between feature vectors. In that case, different distance functions can be used for this task, such as Euclidean distance, Manhattan distance, or Earth Mover's distance.

5.2.1 Experimental Protocol

In the experiments reported in this section, the BoSG representations were generated using *hard assignment* and *sum pooling*.

In the IAM repository, it is provided each dataset with pre-defined sets to graph classification. We used the training, validation, and test sets available in the datasets to perform the experiments.

Concerning the codebook, we used the Mean Shift algorithm [14] to define the vocabulary size. The optimum codebook size is a difficult parameter to be determined in BoW-based approaches. Therefore, the use of an unsupervised clustering method, like Mean Shift, has as objective to simplify the process of building the dictionary.

Nevertheless, we evaluated our approach with two different codebooks. The terms *BoSG (Mean Shift)* and *BoSG (random)* correspond to different versions of BoSG approach that create the codebook using the Mean Shift [57] and a simple random selection, respectively. Both approaches generate dictionaries of the same size, which is determined by the Mean Shift Algorithm.

In order to perform graph matching with our approach, we compute the similarity between the BoSG representations using the Euclidean distance.

5.2.2 Datasets

We used four online available graph datasets from IAM Repository[1] [61]: GREC, Mutagenicity, AIDS, and Letter (LOW). These datasets contain attributed graphs that represent different types of objects, such as letters, molecules, and symbols.

Table 5.1 summarizes some characteristics of these datasets, such as number of vertices, number of classes, and number of graphs. Afterwards, we present a detailed description of each dataset.

Table 5.1: 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

GREC

GREC is a dataset composed of graphs that represent symbols from architectural plans or electronic diagrams. This dataset has twenty two classes of symbols.

Let $G = (\mathcal{V}, \mathcal{E})$ be a graph that represents a symbol of the GREC dataset, each vertex of $\mathcal{V}$ corresponds to an interest point on the graphical symbol, and each edge of $\mathcal{E}$ corresponds to a line segment that link two vertices of $\mathcal{V}$. Some attributes are associated with the vertices and edges of G in order to describe the corresponding symbol.

The vertex attributes are the coordinates of the corresponding point and a label that identifies the corresponding point type (intersection, corner, circle, or an end-point).

[1] http://www.iam.unibe.ch/fki/databases/iam-graph-database (As of April 2014).

The edge attributes are the number of lines that link the corresponding pair of vertices, the label that identifies the type of line segment linking the vertices (arc or line), and the angle of the drawing line.

Figure 5.1 illustrates some samples from the GREC dataset.

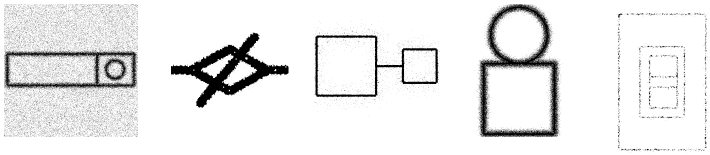


Figure 5.1: Symbol examples from the GREC dataset [61].

AIDS

The AIDS dataset is composed of graphs that belong to two classes that represent active or inactive molecules against HIV. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph that represents a sample of AIDS dataset, each atom of the molecule is associated with a vertex of $\mathcal{V}$ and each covalent bound is associated with an edge of $\mathcal{E}$.

The vertex attributes are the identification number, and the label of the chemical symbol associated with the corresponding atom, and the charge of the corresponding atom and its position on a two dimensional space. The edge attribute is the valence of the corresponding covalent bound.

Figure 5.2 illustrates some samples from the AIDS dataset.

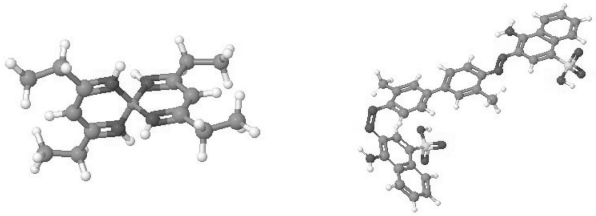


Figure 5.2: Molecule examples from the AIDS dataset [61].

Mutagenicity

The Mutagenicity dataset is composed of graphs that represent molecules that have or not the mutagenicity property. This dataset is divided into two classes: the *mutagen* and the *non-mutagen* molecules.

A molecule is composed of atoms and covalent bounds. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph that represents a sample of Mutagenicity dataset, each atom is associated with a vertex

of $\mathcal{V}$ and each covalent bound is associated with an edge of $\mathcal{E}$. The vertex attribute is the label of the chemical symbol associated with the corresponding atom and the edge attribute is the valence of the corresponding covalent bound.

Letter

The Letter dataset is composed of graphs that represent distorted letters. This dataset has fifteen classes, being each one associated with a capital letter.

The drawing of a letter corresponds to a set of line segments in a two-dimensional space. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph that represents a sample of Letter dataset, each segment line corresponds to an edge of $\mathcal{E}$ and its end-points correspond to vertices of $\mathcal{V}$. The end-point coordinates are the attributes associated with vertices and no attribute is associated with the edges of G.

Figure 5.3 illustrates some samples from the Letter dataset.

Figure 5.3: Letter examples from the Letter dataset [61].

The IAM repository has this dataset on three different levels of distortion: low, medium, and high. In the experiments presented on this chapter, we use the Letter dataset with the low level of distortion.

5.2.3 Baselines

The BoSG approach allows to obtain a distance value that indicates the degree of similarity between a pair of graphs. Therefore, in order to validate the proposed approach on the task of graph matching, it is important to compare it with standard methods for inexact graph matching from the literature.

The graph edit distance is a very popular alternative for performing inexact graph matching. Different approaches have been proposed to compute the edit distance between two graphs. Among the existing methods, we chose two approaches as baselines for our method: the Bipartite Graph Matching, proposed by Riesen et al. [62] and the Attributed Graph Matching, proposed by Jouili et al. [39]. Both approaches use the Hungarian method, which is a polynomial solution for the assignment problem.

In [62], Riesen et al. propose the recursively use of the Hungarian method for determining the minimum cost of vertex and edge assignments. Given two graphs $G_1 = (\mathcal{V}_1, \mathcal{E}_1)$ and $G_2 = (\mathcal{V}_2, \mathcal{E}_2)$, a distance matrix C is built with the costs of editing vertices from $\mathcal{V}_1$ to $\mathcal{V}_2$. For each element c of C, a distance matrix M is built with the edition costs related

to the edge assignments between the corresponding vertices of c. In this approach, the Hungarian Method is first applied on edge matrices, like M. The minimum cost obtained for these matrices is then incremented with the value of the corresponding element of C. After that, the Hungarian method is applied on C in order to compute the minimum total cost of graph matching.

Instead of computing separately the costs of vertex and edge operations, Jouili et al. [39] propose the use of a vector representation, called *node signature*, which gathers vertex and edge attributes. In the Attributed Graph Matching approach, the assignment costs of matrix C are obtained with a distance function that computes the similarity between *node signatures*. In that case, the minimum assignment cost is calculated by applying the Hungarian method on C.

A relevant difference among the methods used in the experiments is that Riesen's approach uses a specific implementation for each dataset, while Jouili's approach and our method use a unique implementation for all datasets.

In Riesen's approach, the computation of the substitution edition cost depends on the graph attributes. In the Letter dataset, the cost for substituting two vertices corresponds to the Euclidean distance between their coordinates, and the substitution of edges does not cause any additional cost. In the GREC dataset, the cost of vertex substitution corresponds to the Euclidean distance between point coordinates when both vertices are of the same type, and it assumes the double value of node insertion/deletion when the vertices are of different types. For this dataset, the edge substitution has a zero cost if both edges are of the same type, otherwise it has the double cost of edge insertion/deletion. In the AIDS dataset, the edge substitution does not cause any additional cost, and the vertex substitution assumes the double cost of vertex insertion/deletion when the vertices do not correspond to the same chemical symbol, otherwise it has a zero cost. The computation of substitution costs for the Mutagenicity dataset is the same one of the AIDS dataset, except for the fact that, when vertices refer to different chemical symbols, the vertex substitution assumes a cost proportional to the distance between their corresponding strings.

5.2.4 Research Questions

In this section, we aim to investigate if the use of a bag representation is an efficient and effective alternative for graph matching. The goal of the experiments is to evaluate our method considering different aspects: accuracy, performance, and learning capacity.

The BoSG approach allows to perform graph classification and graph retrieval in two stages. First, in an offline phase, all graphs of a database are described as feature vectors. Then, the similarity between a query graph and the graphs of the database is calculated online. The use of data structures to index the feature vectors would contribute to even

improve the performance of the method.

Since the computation of similarities between feature vectors has a very low complexity, we expect that our method outperforms the baseline approaches in terms of execution time. However, it is important to preserve the precision of the results. Therefore, the first questions related to these experiments are

- *How fast is BoSG compared to the baseline methods?*
- *Are the accuracy results of BoSG comparable to other methods?*

By achieving good accuracy rates and reduced execution time, the proposed approach could be considered a promising alternative to perform graph retrieval in large datasets, where the use of popular graph methods becomes unfeasible due to their high-computational cost.

In a BoW-based approach, the size of the codebook is always an important element that should be evaluated in order to generate discriminant bag representations. In the experiments of this section, we apply the Mean Shift algorithm to generate the codebook. This clustering algorithm simplifies the search for the optimal size of the codebook, since we do not need to directly specify the number of clusters. However, it has a parameter related to the kernel bandwidth, which impacts the final number of clusters. Therefore, another research question is

- *What is the impact caused by the codebook size on BoSG's performance?*

Some studies in the literature [69, 83] have shown that the use of a random selection instead of a clustering algorithm does not affect the quality of the codebook. In this context, we aim to evaluate BoSG with a random codebook and compare its results with the ones obtained with a Mean-Shift codebook. In these experiments, we use codebooks of the same size, which corresponds to the number of clusters generated by the Mean Shift algorithm.

Concerning the learning capacity of our method, another interesting question that should be evaluated is:

- *Does BoSG learn faster than other methods?*

Different from the baseline methods, BoSG describes a graph as a feature vector. Creating vector representations, different classification methods may be used for solving graph matching tasks. Thus, in this section, we also intend to answer the following question:

- *Is it possible to improve the accuracy results of BoSG by using different classifiers?*

In order to investigate the questions presented above, the proposed representation was evaluated for the task of graph classification from five different perspectives:

- Comparison with baselines in terms of classification accuracy;
- Comparison with baselines in terms of execution time;
- Evaluation of the impact of the codebook size;
- Evaluation of the impact of the training set size;
- Evaluation of the representation when combined with different classifiers.

5.2.5 Evaluation Measures

The experiments of this section consist in computing a distance matrix with the dissimilarity values between graphs from training and test sets.

In order to compare the performance of the proposed approach over the baseline approaches, we used some metrics to evaluate the efficacy and efficiency of each method.

Effectiveness Evaluation

In this section, the effectiveness of a method is measured in terms of accuracy. A K-Nearest Neighbor (KNN) algorithm is applied to classify 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 Riesen's approach, the validation sets are used for finding the best parameters for each dataset.

Efficiency Evaluation

In the experiments performed in this section, 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 measured the execution times in seconds and we executed all methods five times for each dataset. Table 5.6 in Section 5.2.6 presents the average times and standard deviations related to the performance of each method on the four datasets described in Section 5.2.2.

5.2.6 Results

Classification Accuracy

Tables 5.2, 5.3, 5.4, and 5.5 present the accuracy results obtained for each dataset using the KNN classifier with the parameter K assuming values of one, three, and five.

For the *BoSG (random)* approach, we repeated five times the process of creating the codebook and the bags aiming to evaluate the invariance of the representation to different seeds. The results of Tables 5.2 to 5.5 refer to the average accuracy with the standard deviation.

In the case of Mutagenicity dataset, a very large number of node signatures are generated from the training set. Thereby, in order to create the Mean-Shift codebook, it was used a subset of the training signatures.

Table 5.2: Results for the GREC dataset.

	BoSG (Mean Shift)	BoSG (random)	Riesen [62]	Jouili [39]
$K = 1$	0.934	0.969 ± 0.007	**0.983**	0.981
$K = 3$	0.896	0.947 ± 0.007	**0.983**	0.975
$K = 5$	0.860	0.92 ± 0.01	**0.985**	0.960

Table 5.3: Results for the Mutagenicity dataset.

	BoSG (Mean Shift)	BoSG (random)	Riesen [62]	Jouili [39]
$K = 1$	0.690	0.672 ± 0.008	**0.695**	0.652
$K = 3$	0.703	0.681 ± 0.008	**0.720**	0.663
$K = 5$	0.713	0.69 ± 0.01	**0.719**	0.652

Table 5.4: Results for the AIDS dataset.

	BoSG (Mean Shift)	BoSG (random)	Riesen [62]	Jouili [39]
$K = 1$	0.989	0.977 ± 0.003	0.993	**0.995**
$K = 3$	0.991	0.970 ± 0.006	0.990	**0.997**
$K = 5$	0.985	0.959 ± 0.006	0.984	**0.996**

The results of Tables 5.2-5.5 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 edition costs and it requires a search for the optimal combination of

Table 5.5: Results for the Letter dataset.

	BoSG (Mean Shift)	BoSG (random)	Riesen [62]	Jouili [39]
$K = 1$	0.945	0.89 ± 0.01	**0.989**	0.920
$K = 3$	0.948	0.89 ± 0.02	**0.991**	0.909
$K = 5$	0.949	0.88 ± 0.02	**0.993**	0.895

three parameters. In summary, 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 5.6 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 5.6 refer only to the time for computing the distances between graph bags. As it can be observed, BoSG has a much better performance when compared with all baselines in terms of execution time.

Table 5.6: Performance Results.

	GREC (s)	Mutagenicity (s)	AIDS (s)	Letter (s)
BoSG	0.11 ± 0.02	2.9 ± 0.1	0.29 ± 0.06	0.384 ± 0.003
Riesen [62]	262 ± 4	65430 ± 2825	616 ± 21	101 ± 6
Jouili [39]	327 ± 19	16668 ± 124	1558 ± 37	773 ± 16

In order to show the relative improvement for the execution time, Table 5.7 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.

Table 5.7: Relative Performance Results.

	GREC (s)	Mutagenicity (s)	AIDS (s)	Letter (s)
BoSG	1	1	1	1
Riesen [62]	2382	22562	2124	263
Jouili [39]	2973	5748	5372	2013

Figure 5.4 summarizes the results of evaluated methods in terms of both their accuracy rates and execution time. In this chart, 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.

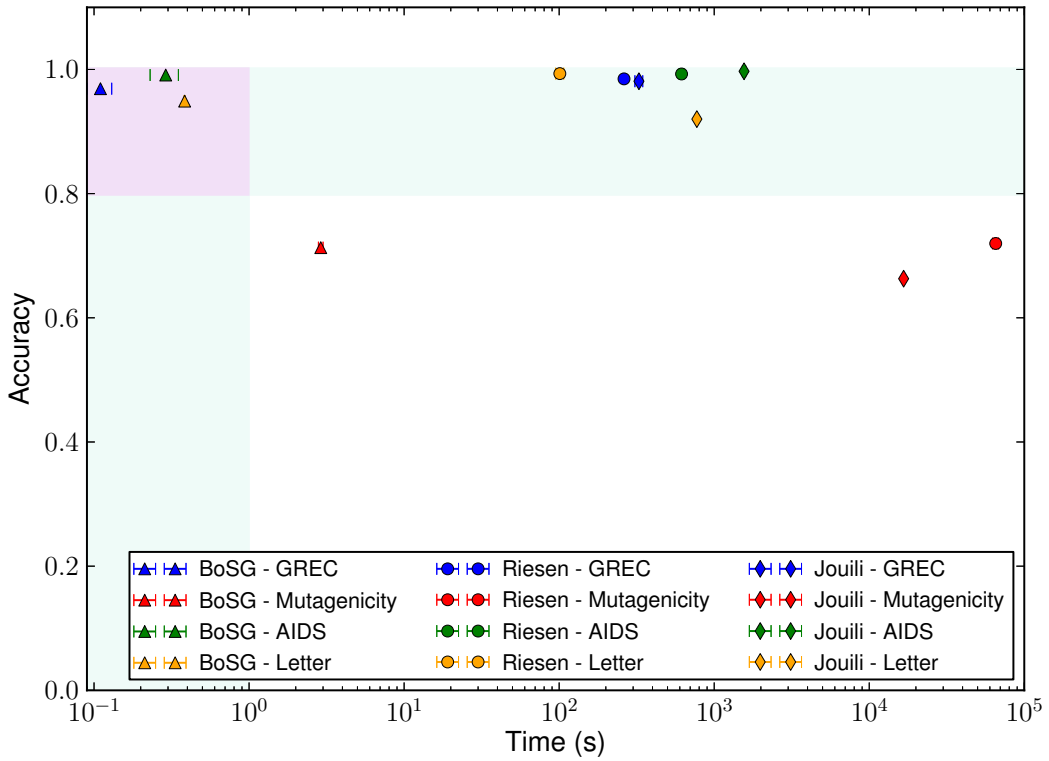


Figure 5.4: Accuracy rates with respect to the execution time for different methods and datasets. Each method is identified by a marker and each dataset is identified by a color. The light-blue bands indicate the areas of the highest accuracy rates or the lowest execution times, and the intersection of these bands (light-green square) indicates the area where the best results considering both accuracy and execution time are placed.

Table 5.8 shows the offline time spent for executing BoSG approach, which includes the time spent for generating the Mean-Shift codebook and the feature vectors that correspond to the bags. We do not show the time spent to generate the random codebook, since it is very small.

It is important to note that 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 5.8 correspond to the time required to generate the four codebooks using the same Mean Shift parameters used to obtain the results of Tables 5.2-5.5. In the experiments related to Tables 5.2, 5.3, 5.4 and 5.5, the used Mean Shift parameters were 0.05, 0.05, 0.3, and 0.01, respectively.

Table 5.8: Offline Time

	GREC (s)	Mutagenicity (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

Impact of the Codebook Size

In the Mean Shift algorithm [57], 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 reduction of the bandwidth, which contributes to increase the size of codebook. Figures 5.5 and 5.6 show, respectively, the variation of the codebook size and the performance of BoSG approach using the test sets for different values of the Mean Shift parameter.

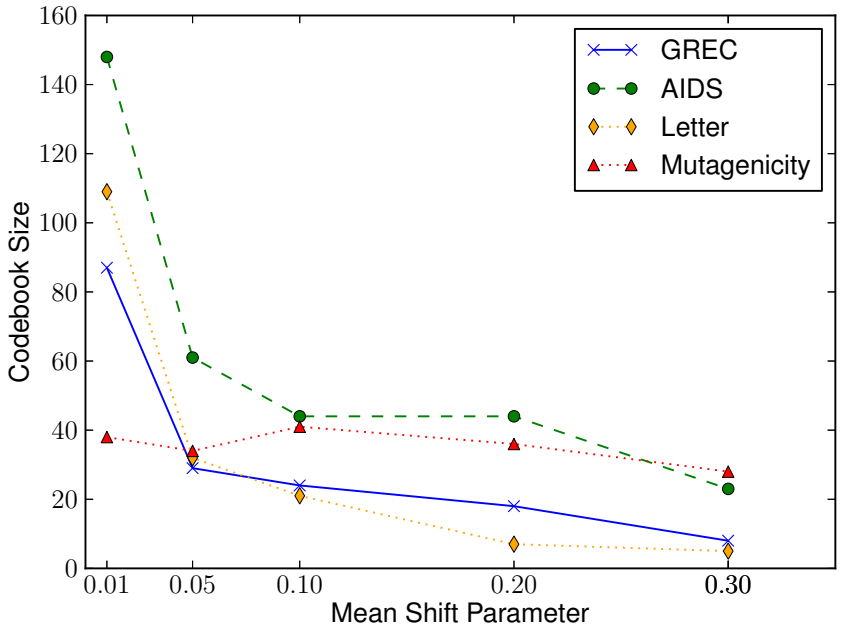


Figure 5.5: Size of the codebook using different values of the Mean Shift parameter.

Figure 5.5 indicates the impact of the Mean Shift parameter on the size of the code-

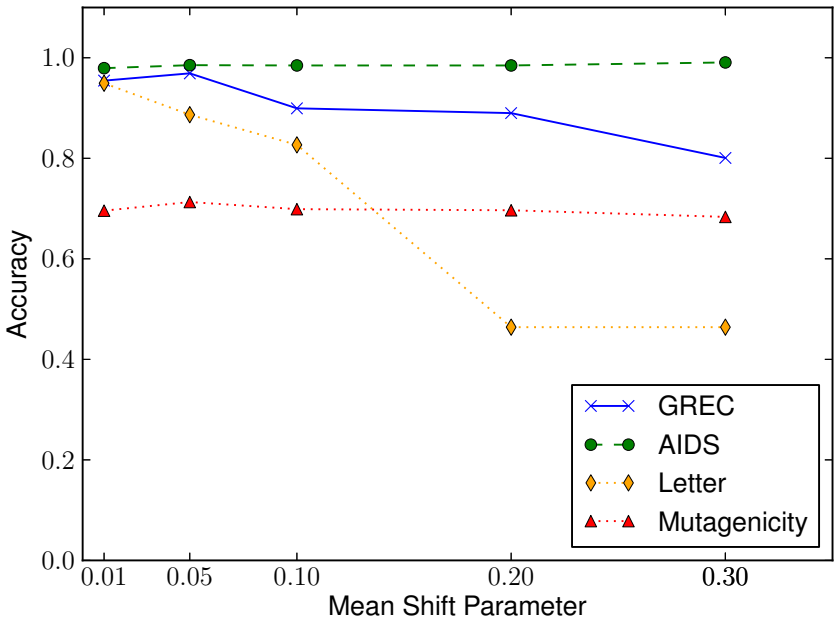


Figure 5.6: BoG performance using different values of the Mean Shift parameter.

book. 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. For this reason, the use of larger codebooks can improve the graph description and increase the classification results, as shown in Fig. 5.6. However, if the codebook is too large, the words are not good enough in terms of generality, which means that two similar patterns may be assigned to different words that could have been defined as a single word. Thereby, it is important to find out a trade-off that corresponds to the optimum size of codebook.

Impact of the Training Set Size

In this section, we also evaluated the performance of BoSG and the edit distance approaches using different sizes of training set. In this experiment, we built different training sets by selecting a percentage of graphs from each class of the original training sets.

Table 5.9 shows the different sizes of training set and dictionaries used in this experiment and Figures 5.7, 5.8, 5.9 and 5.10 show the best results obtained for each approach using the KNN classifier with K equals to one, three, and five.

It can be observed from Figures 5.7, 5.8, 5.9, and 5.10 that our results are similar to evaluated baselines.

Table 5.9: 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

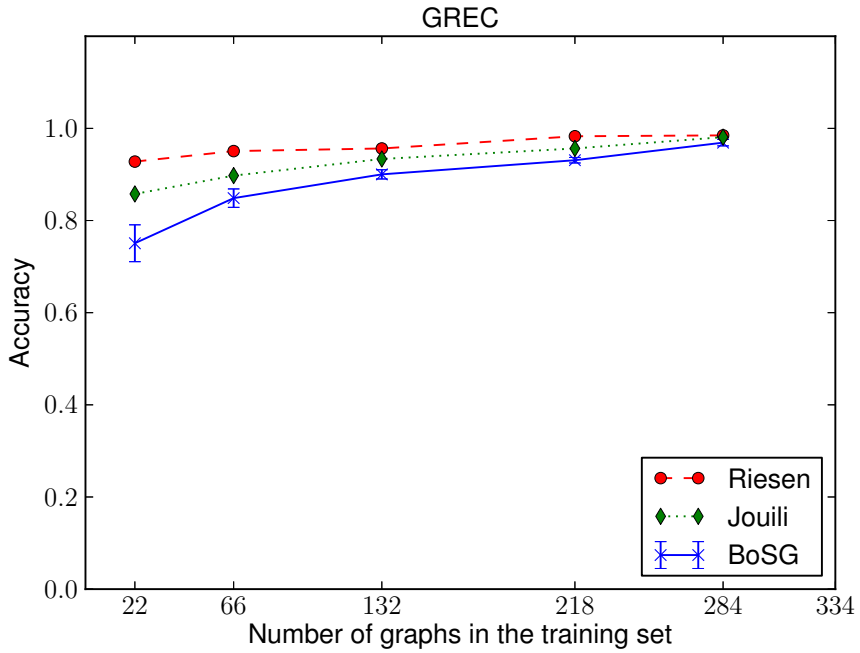


Figure 5.7: Results for the GREC dataset using different training set sizes.

The curves of Figures 5.8 and 5.10 show that our method learns faster than Jouili's approach in the case of Mutagenicity and Letter datasets. However, Jouili's approach has a better performance than ours in the GREC dataset, as shown in Figure 5.7.

In this experiment, Riesen's approach achieves the best scores in all datasets, but we reach a similar performance in the Mutagenicity and AIDS datasets.

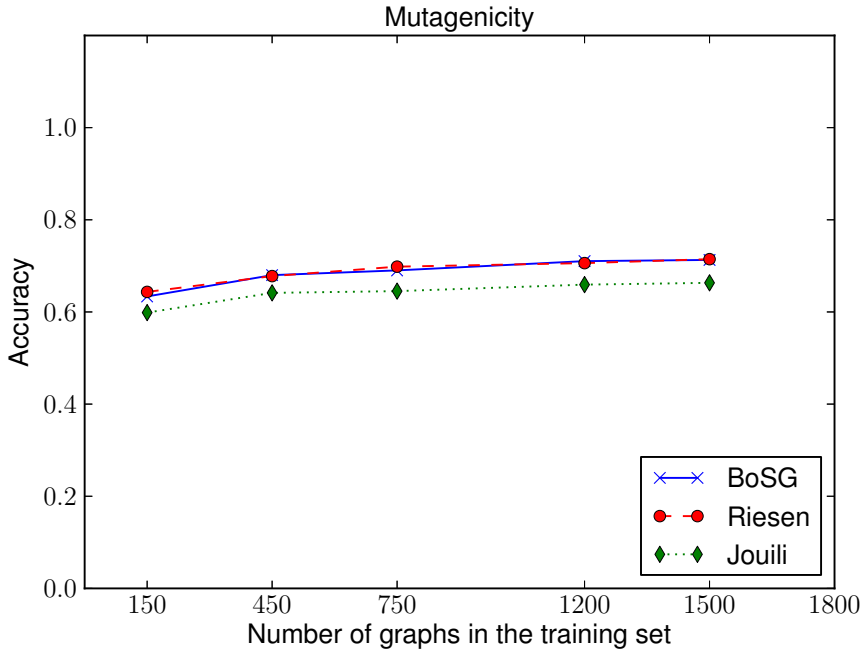


Figure 5.8: Results for the Mutagenicity dataset using different training set sizes.

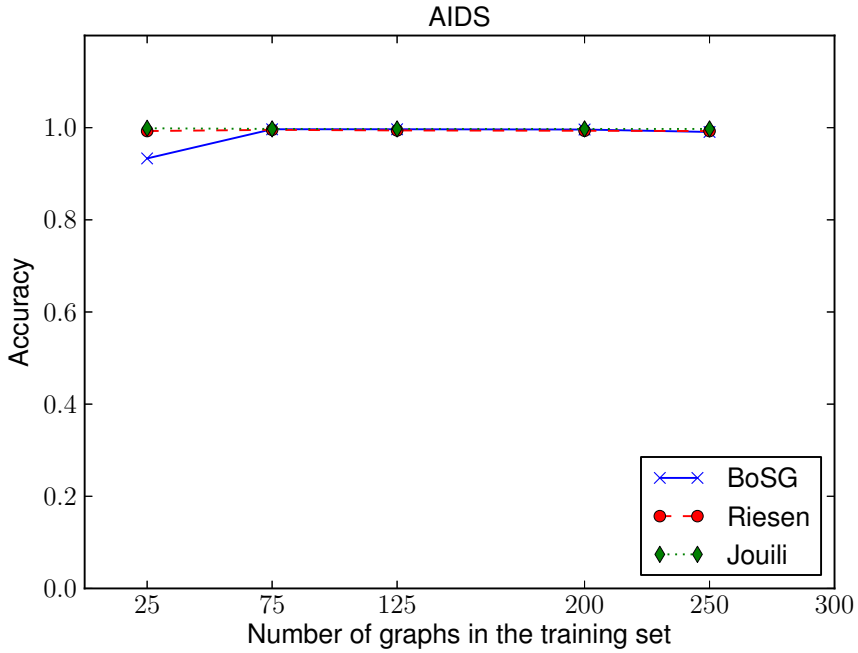


Figure 5.9: Results for the AIDS dataset using different training set sizes.

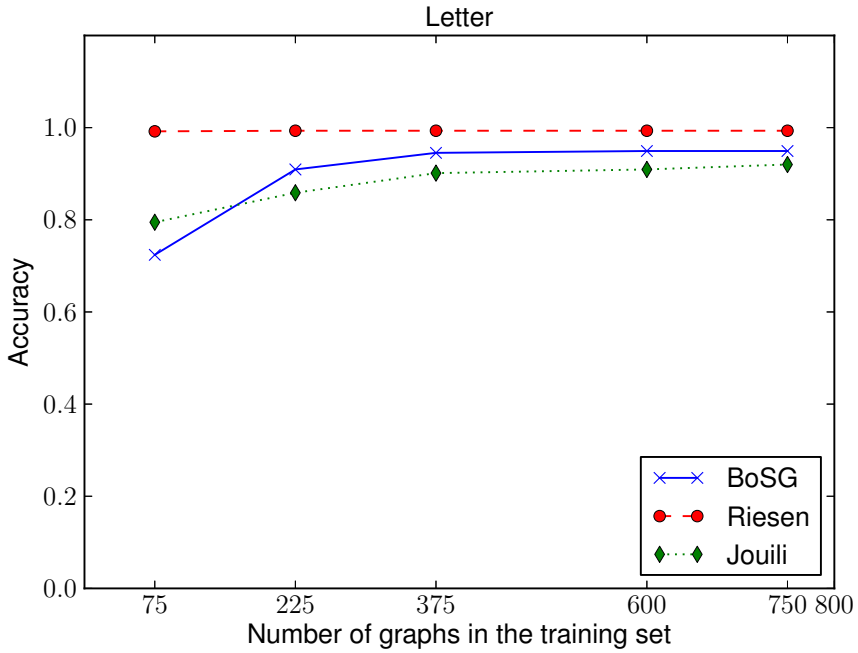


Figure 5.10: Results for the Letter dataset using different training set sizes.

Impact of Using Different Classifiers

Besides the fast performance, the representation of graphs as feature vectors allows the use of different classifiers. This flexibility can contribute to achieve higher accuracy rates. The results of Table 5.10 refer to the evaluation of our method using three classifiers: KNN, SVM [13], and OPF [56]. We compared our best result using KNN with the results obtained using Support Vector Machine (SVM) and Optimum-Path Forest (OPF). In this experiment, we used the validation sets to seek the best parameters for SVM and OPF.

Table 5.10: 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

It can be observed from Table 5.10 that, by using different classifiers, we improve the results of our method in three datasets and we achieve a tie in one dataset.

5.3 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 encode the spatial relationships of visual words and we use them to create a graph-based codebook. Then, an image is represented by a vector that describes the distribution of *visual graphs* within the image.

Since the computation cost for creating the visual codebook and classifying images are the same as those observed for the BoW approach, our method has an additional cost related to the creation of the graph-based codebook.

5.3.1 Experimental Protocol

In the experiments of this section, we employ SIFT [48] to describe interest points, which are detected using both sparse and dense techniques: a dense grid [80, 81] with sampling spacing of 6 pixels, and the *Hessian Affine* [52] and *Difference Of Gaussians* [48] keypoint detectors.

In the BoVG approach, the interest-point detection procedure affects the definition of graph edges. In the case of dense sampling, all edges have approximately the same weight (points are in a grid, separated by the same distance). In this case, enforcing constraints on edge weights would either eliminate all graph edges, or none of them. For this reason, we did not impose any edge constraint when using dense sampling for the interest points, and use all triangles of the Delaunay triangulation as connected graphs. In the case of sparse sampling, we defined the lower and upper size of edge as 10 and 150 pixels, respectively.

We used a simple random selection to generate both the visual-word and the graph-based codebooks. On all experiments, we created the graph-based codebook with the same size of the visual-word codebook. The size of the codebook impacts the size of the descriptor generated by each method. Let K be the size of the codebook, BoW and BoVG create bags of size K.

For all methods evaluated in the experiments, the bag representations were generated using *hard assignment* and *average pooling*.

5.3.2 Datasets

We used two online available image datasets: Caltech-101 [19] and Caltech-256 [27].

These datasets contain general objects from different categories and they are usually used for image classification. The detailed descriptions of both datasets are presented below.

Caltech-101

This dataset [19] 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 8,878 images that belong to the 101 object classes. Figure 5.11 illustrates some image examples of the Caltech-101 dataset.

Figure 5.11: Examples from the Caltech-101 dataset.

Caltech-256

This dataset [27] 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 30291 images that belong to the 256 object classes. Figure 5.12 illustrates some image examples of the Caltech-256 dataset.

Figure 5.12: Examples from the Caltech-256 dataset.

5.3.3 Baselines

In this section, we compare the proposed approach (BoVG) with the traditional Bag of Words (BoW) [76], the Spatial Pyramids (SP) method, proposed by Lazebnik et al. [44], and the Word Spatial Arrangement (WSA) method, proposed by Penatti et al. [59].

The BoW method was introduced in Section 2.4 and the SP and the WSA correspond to BoW-based approaches that improve the image representation by encoding the spatial relationships of visual words into the final descriptor.

Spatial Pyramids [44] is one of the most popular methods from the literature of Bag of Visual Words, achieving high accuracy rates on image classification. After hierarchically splitting the image into different regions, this method proposes to compute a BoW representation for each defined region. The final image descriptor is generated through the weighted concatenation of the histograms from all regions. One of the disadvantages of the SP approach is the high dimensionality of the generated feature vectors.

The WSA [59] is an approach that achieves good results using feature vectors with not too high dimensions. The WSA method proposes to account all spatial relative positioning between visual words and the interest points of an image. For that, each interest point is used as the origin center for partitioning the image into four quadrants. Each created partition defines new spatial arrangements of visual words. In the end, the final image descriptor contains the distribution of relative positions of visual words within the image.

Let K be the size of the codebook, the WSA uses a tuple of 4 values for each visual word, creating a feature vector of size $4K$. The SP bags, in turn, were generated using a 2-level Pyramid, which results in bags of size $21K$.

In the experiments of this section, we evaluated our approach and also some others that combine BoVG with a standard method of the literature. Table 5.11 presents the description of the BoVG-based approaches that were evaluated.

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

Table 5.11: Evaluated variations of the BoVG approach.

5.3.4 Research Questions

In this section, we evaluate the BoVG performance for the task of object classification. Our approach is compared, in terms of accuracy, with the baseline methods introduced in Section 5.3.3.

In a BoW-based approach, the size of the codebook and the interest point detection influence the quality of the generated representations. Therefore, the first questions that the experiments of this section aims to answer are:

- *What is the impact caused by the codebook size on BoVG's performance?*
- *What is the impact caused by the interest point detector on BoVG's performance?*
- *Are the accuracy results of BoVG comparable to other BoW-based methods?*

In order to better understand the overall results of each method, it is important to analyse the accuracy rates of each image category. This analysis may contribute to comprehend the behavior of each method in accordance with the object characteristics. Additionally, we investigate if the learning curve of our approach is similar to the other methods. Thereby, two other relevant questions that should be investigated are:

- *What are the object categories that BoVG performs better or worse than other methods?*
- *Does BoVG learn faster than other methods?*

The investigation of the questions presented above led to performing experiments under four different perspectives:

- Evaluation of the impact of the the codebook size and the interest-point detector;
- Evaluation of the impact of the training set size;
- Comparison with baselines in terms of classification accuracy;
- Evaluation of the classification accuracy rates per class;

5.3.5 Evaluation Measures

In this section, the effectiveness of a method is measured in terms of accuracy. For the classification procedure, we used an one-vs-all SVM [13, 36] 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%.

5.3.6 Results

Impact of the Codebook Size and the 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 approach (BoW) was used as reference and we used 30 samples per class for training the classifier.

Figures 5.13 and 5.14 show the mean accuracies of each method for different codebook sizes.

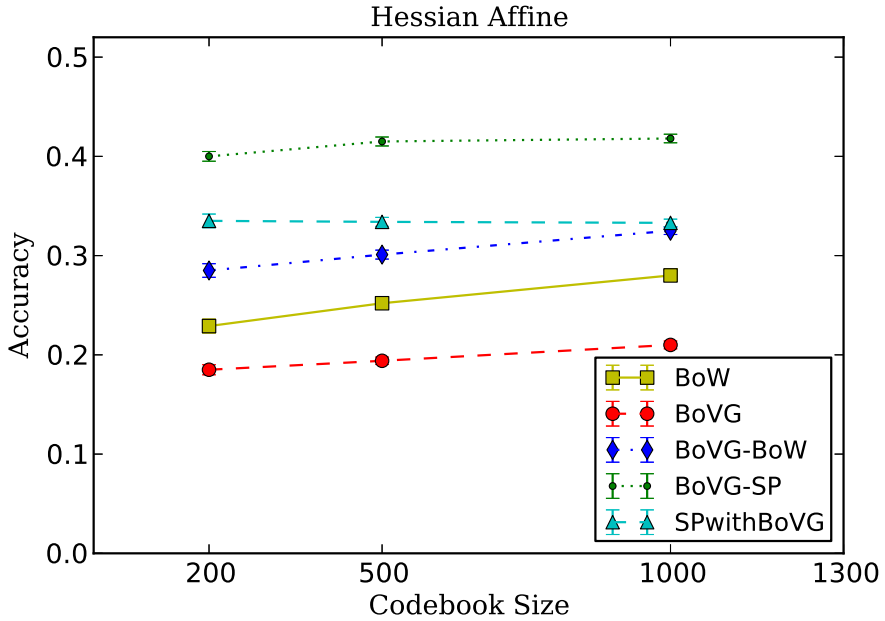


Figure 5.13: Performance of BoVG-based methods on Caltech-101 using Hessian Affine detector.

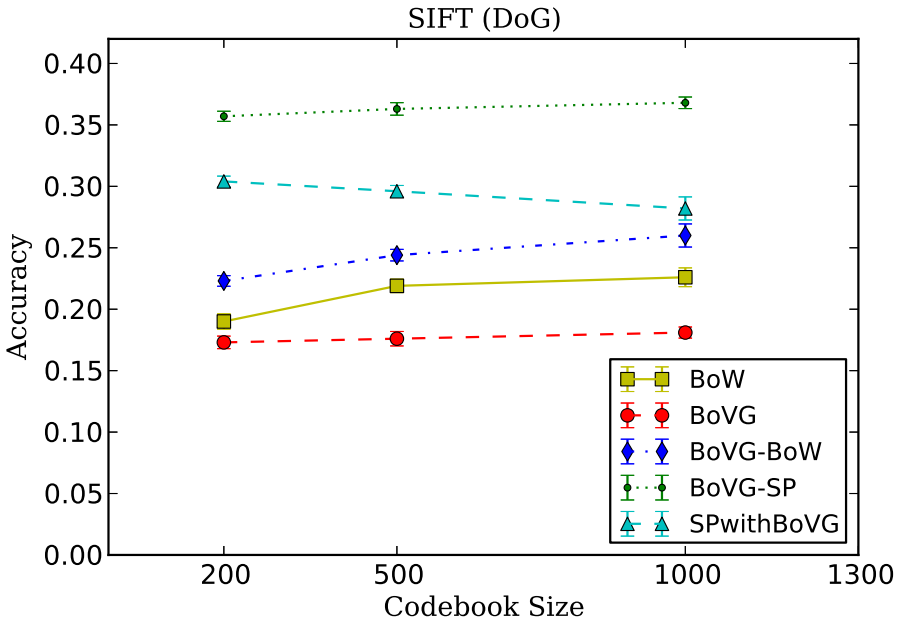


Figure 5.14: Performance of BoVG-based methods on Caltech-101 using SIFT detector.

The results in Figures 5.13 and 5.14 show that the distribution of visual-word arrangements contribute to improve the classification results of other approaches for all cases evaluated. The best results are observed for the combination of BoVG with Spatial Pyramids. It can also be observed that the size of the codebook has a greater impact on BoW. Thus, BoW obtained higher accuracy rates than BoVG, specially with large codebooks.

Impact of the Training Set Size

We also evaluated the impact of different training sizes on the overall performance. Figure 5.15 shows the results of BoVG, BoVG-SP, BoW, WSA, and SP using SIFT detector and a codebook of size 200 on Caltech-101. The curves in Figure 5.15 show that the accuracy rate increases with the number of samples for training. Since the results of all methods are equally improved, it indicates that the size of the training set does not favor one representation over another.

Classification Accuracy

Figures 5.16 and 5.17 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

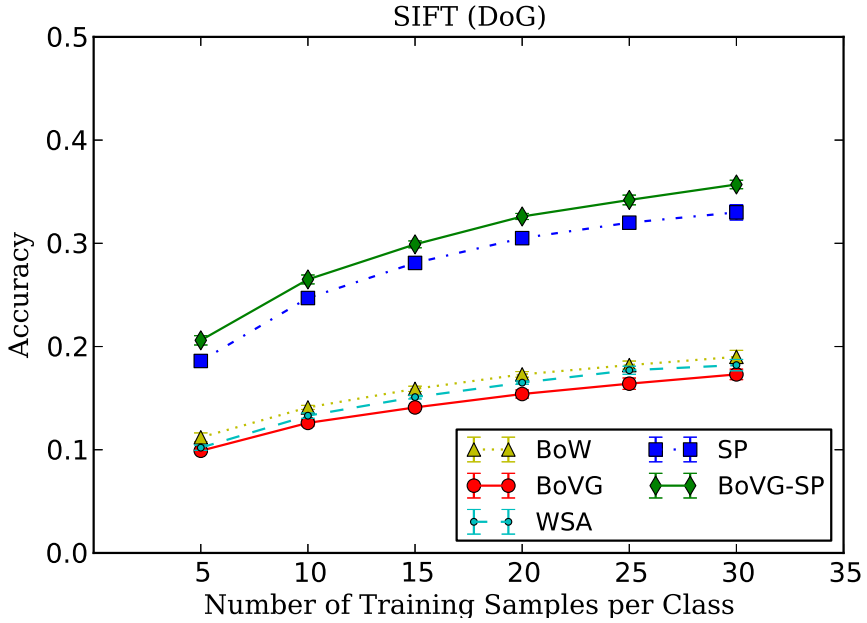


Figure 5.15: Classification results of BoW, BoVG, WSA, SP, and BoVG-SP on Caltech-101 for different training set sizes.

were evaluated on Caltech-101 and Caltech-256 using codebooks of size 200 and 1000, respectively. We used again a training set with 30 images per class.

The results in Figures 5.16 and 5.17 show that BoVG-SP has a competitive performance on both datasets. In the experiment on Caltech-101 (Figure 5.16), BoVG-SP yields the best classification accuracy rates in all cases, with a statistical tie with SP and BoW-SP for the dense-sampling case. In Caltech-256 (Figure 5.17), 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 both datasets, the use of dense sampling technique for interest-point detection provides the highest accuracy rates. The main difference between both techniques of interest-point detection is that sparse sampling consider only salient regions as points of interest, while dense sampling selects interest points independently from the visual content of the corresponding region. The superior performance of dense sampling can be explained by the fact that non-salient regions can be relevant for distinguishing some object classes, specially when the objects do not have enough salient regions to describe them.

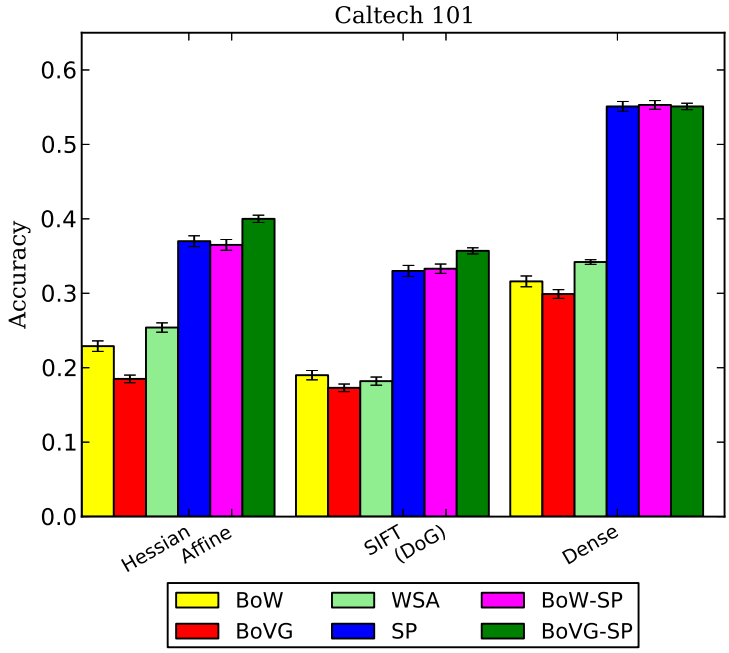


Figure 5.16: Classification results of Bow, BoVG, WSA, SP, and BoVG-SP on Caltech-101, using different interest-point detectors.

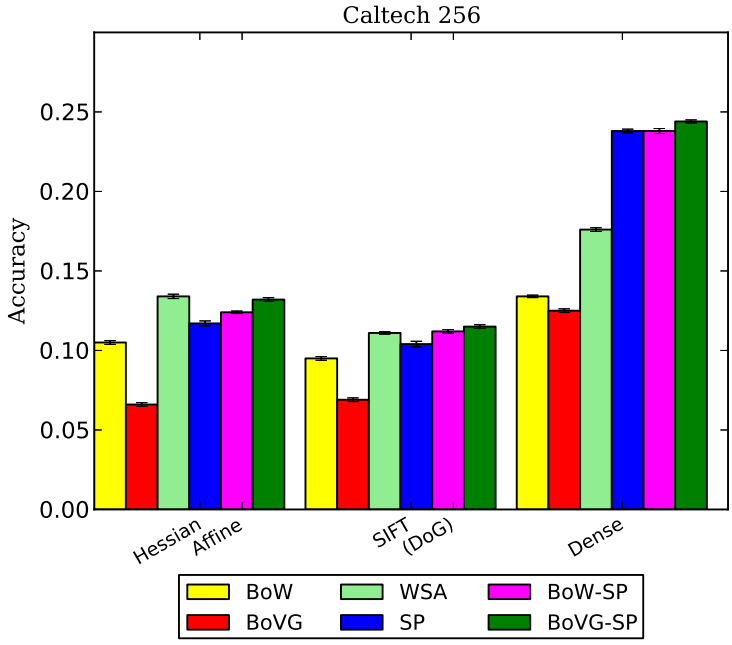


Figure 5.17: Classification results of Bow, BoVG, WSA, SP, and BoVG-SP on Caltech-256, using different interest-point detectors.

Analysis of the Classification Accuracy per Class

Figures 5.18 and 5.19 show the results per class on Caltech-101 with the objective of evaluating the impact of different approaches in image categories. We used SIFT detector, a codebook of size 200 and a training set with 30 images per class to evaluate BoW, BoVG, WSA, SP, and BoVG-SP. The BoVG-SP was chosen over the other methods of Table 5.11, because it was superior to the others in the first experiment (see Figures 5.13 and 5.14). As it can be observed, BoVG-SP yields the best results for almost all classes.

In order to better analyse the results of Figure 5.18, we present results of three classes in which our method performs better than baselines, and vice versa. Figures 5.20, 5.21 present, respectively, a per-class comparison between BoVG and the baselines BoW and SP. Figures 5.22 and 5.23 present a per-class comparison between BoVG-SP and the baselines SP and WSA, respectively.

It can be observed from Figures 5.20 and 5.23 that BoW and WSA perform better for classes which contain a frequent occurrence of similar regions in the whole object, and from Figures 5.20 and 5.22, we can observe that SP performs better for object categories that have regions well defined visually and spatially.

The BoVG is worthwhile to classes whose objects have discriminant local regions without necessarily have a fixed relative position. In the BoVG-SP approach, the concatenation of SP bags with BoVG adds the multi-scale factor to our method. Thereby, BoVG-SP encodes two different types of spatial relationships: local structures and scale. This combination has obtained the best accuracy rates, improving BoVG and SP standalone results.

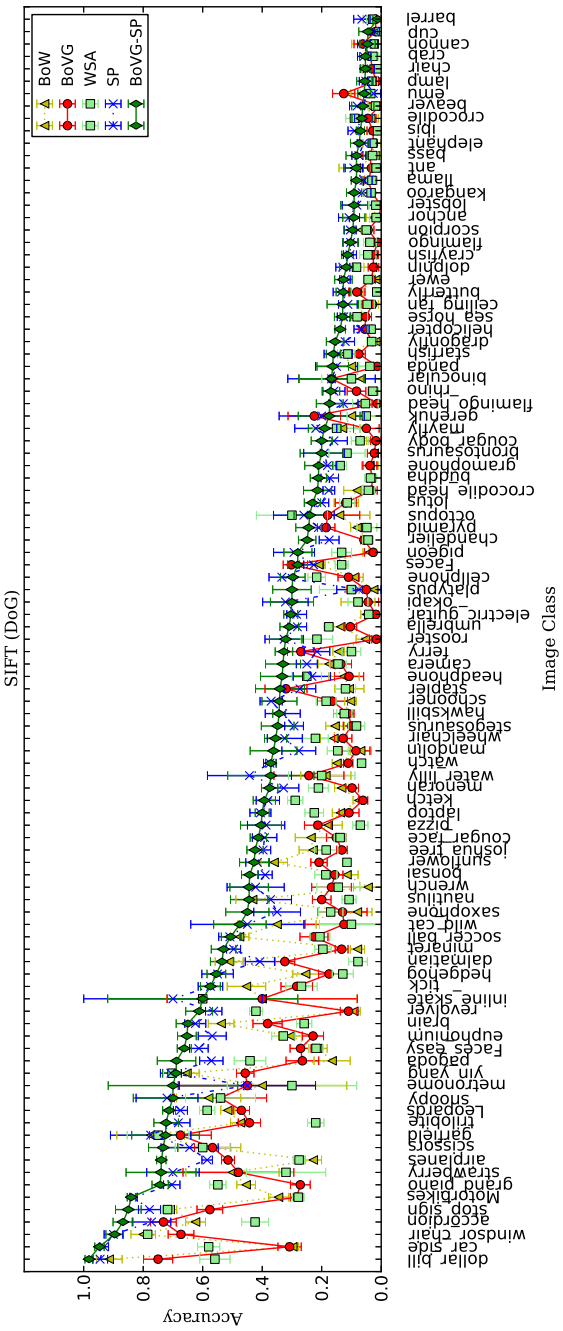


Figure 5.18: Accuracy rates per class on Caltech-101 with errors.

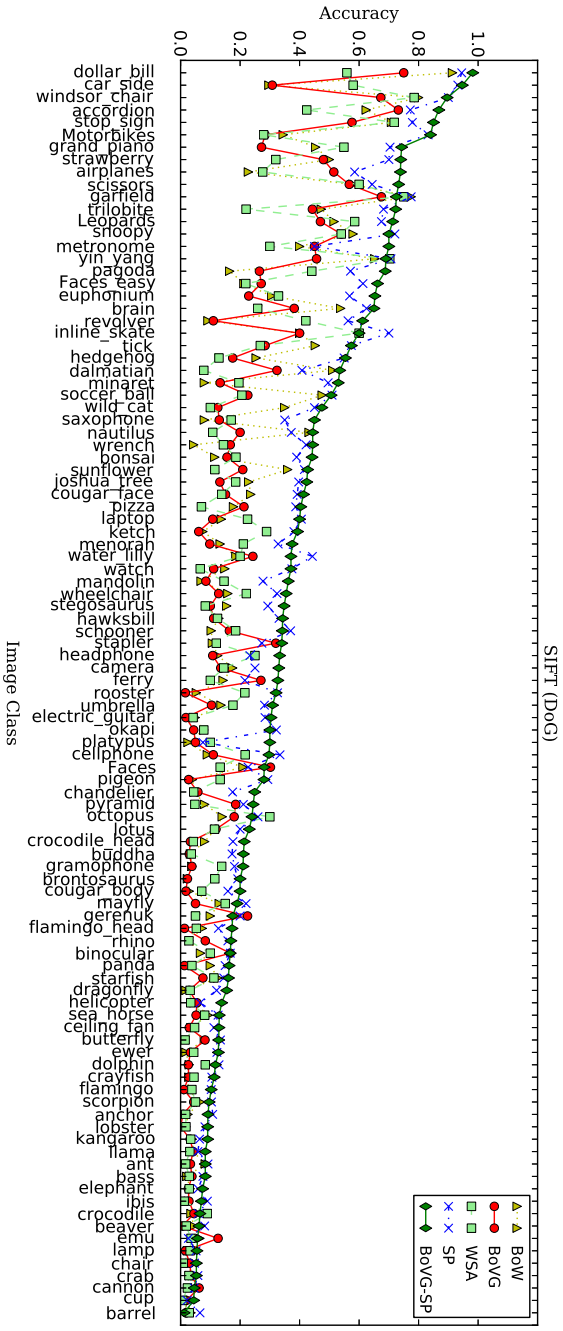


Figure 5.19: Accuracy rates per class on Caltech-101 without errors.

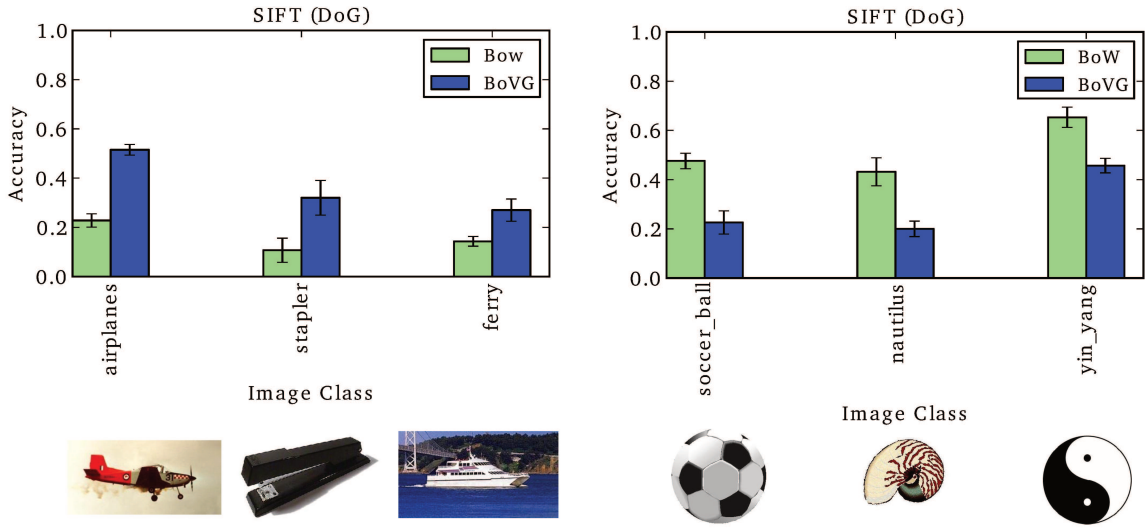


Figure 5.20: Analysis per class for comparing BoW and BoVG approaches.

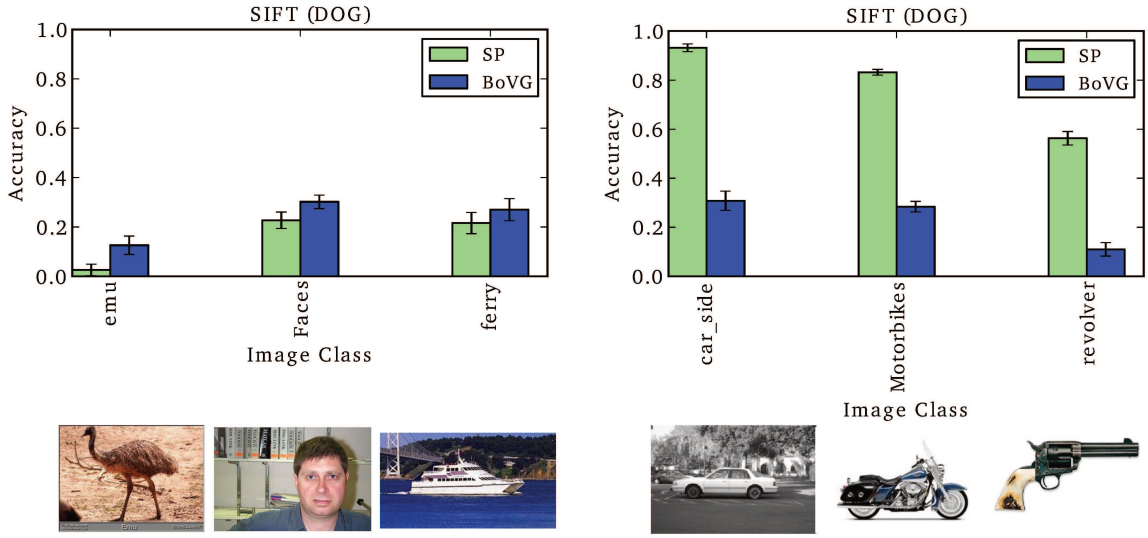


Figure 5.21: Analysis per class for comparing SP and BoVG approaches.

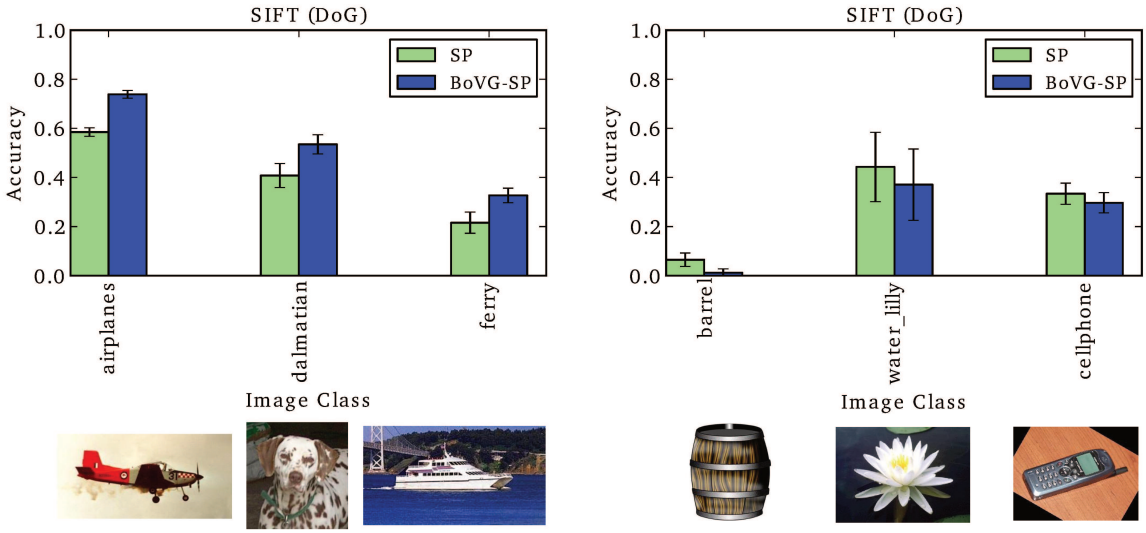


Figure 5.22: Analysis per class for comparing SP and BoVG-SP approaches.

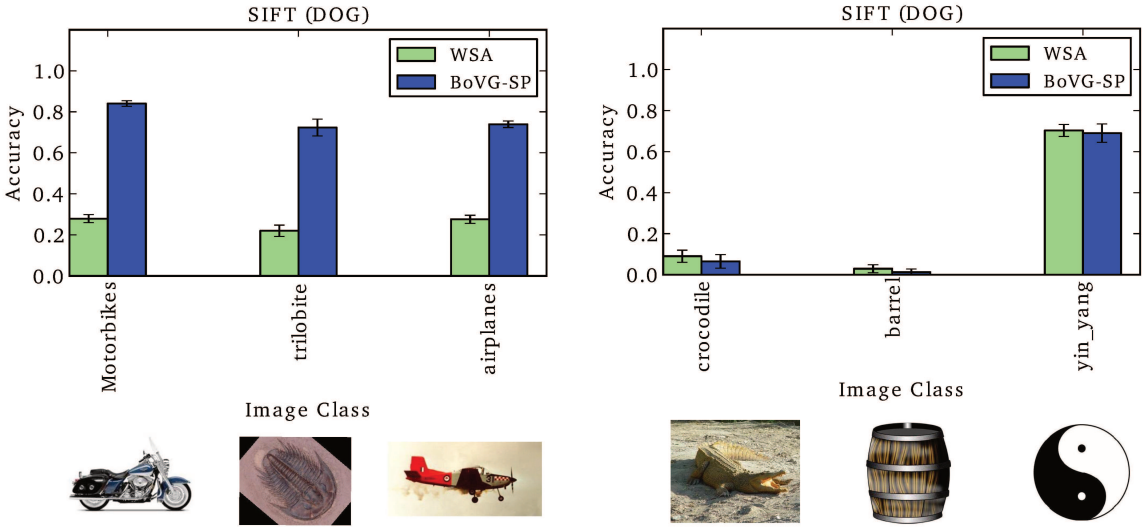


Figure 5.23: Analysis per class for comparing WSA and BoVG-SP approaches.

Chapter 6

Conclusions

6.1 Contributions

The hypothesis of this dissertation 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 vocabulary 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.

This work proposes a generic BoW-based approach, called Bag of Graphs (BoG), that uses a graph-based vocabulary to create object representations. We presented a formalization of the proposed method in terms of mathematical definitions, which enables the proper adaptation of the method to different applications. Additionally, two different approaches presented in this dissertation demonstrated how a graph-based vocabulary can be employed to describe images and some general objects, like graphical symbol and molecules.

We evaluated the performance of the proposed method in two tasks: graph classification and image classification. For both applications, the results show that our approaches achieve accurate results compared to standard methods of the literature.

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. In order to validate the proposed approach, we employed the Mean Shift algorithm to create the codebook, which has simplified the search for the optimal size of the codebook. In the experiments, the BoSG approach demonstrated to be an efficient alternative for performing graph matching. The use of feature vectors to represent graphs enables to perform graph retrieval in large databases, which was limited due to the high computational cost of the traditional graph-based approaches.

The baseline methods presented in Section 5.2.3 employ the Hungarian method to perform graph matching. Since the Hungarian method has a complexity $O(V^3)$, being V the number of vertices in the graphs, the performance of the baseline methods depends on the number of graphs and the size of these graphs. In the case of Riesen's approach, the Hungarian method is employed for both vertex and edge assignments. Thereby, its complexity also depends on the degree of vertices.

The main contribution of the BoSG approach is the proposal of a method that reduces the complexity of the inexact graph matching task, while keeping the same performance of classical approaches. Using this method, a search for similar graphs can be accomplished with a complexity $O(Nd)$, where N is the number of graphs in the training dataset and d is the size of the dictionary. Moreover, in the context of graph retrieval, it is possible to improve the overall performance of the method by using an indexing structure, such as Locality-Sensitive Hashing (LSH) [37] or K-Dimensional Tree (KDTree) [22].

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 we have 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 summary, we presented, in this work, a flexible model to create discriminant object representations. The experiments showed that the proposed BoG-based approaches are effective techniques to perform object recognition. Notwithstanding, the choice of appropriate functions and descriptors, that allow describing the relevant object characteristics, is essential to the proper performance of our method.

6.2 Future Work

In this section, we present possible future work related to this dissertation.

- *Evaluate the BoVG approach on different image datasets.*

  In this work, we evaluated the BoVG approach on Caltech-101 and Caltech-256, which are standard datasets to perform image classification. Notwithstanding, we believe that our method could achieve even better results if it was evaluated on a

dataset, where the spatial relationships of features is a determining factor. Therefore, a future work would be the investigation of more appropriate datasets [18, 25], which would better exploit the potential of our method.

- *Investigate the relation between the sizes of the two codebooks in the BoVG approach.*

  In the experiments for validating the BoVG approach, we have assigned the same size for both the visual-word and visual-graph codebooks. However, more exploration is necessary in order to confirm if this is a reasonable choice.

- *Investigate different applications for the proposed method.*

  There are different applications where the spatial information is a relevant characteristic, such as remote sensing [17], image classification [74], symbol spotting [4], and video retrieval [2]. Since BoG is a flexible method that can be adapted to different contexts, it would be interesting to evaluate our method on these applications.

- *Evaluate the performance of the proposed method in image retrieval and graph retrieval.*

  We validated the proposed methods BoVG and BoSG for classification tasks. Another future work concerns the evaluation of these methods in the retrieval context. For this application, we can use the experimental protocol and the datasets presented in [60].

- *Evaluate the performance of BoSG approach with an index structure.*

  The BoSG approach has demonstrated to be an efficient alternative for graph matching. Since this method represents a graph as a feature vector, it is possible to adopt indexing structures to speed up even more the search of graphs on large datasets. Thereby, an interesting experiment would be to evaluate the performance of BoSG approach with an indexing structure, like LSH [37] or KDTree [22], for graph retrieval on large datasets.

6.3 Publications

This section lists the articles published during the period of this Master's project. Two of these publications are directly related to the conducted research.

- F. B. Silva, S. Goldenstein, S. Tabbone, and R. da S. Torres. Image classification based on Bag of Visual Graphs. In *Proceedings of 20th IEEE International Conference on Image Processing (ICIP)*, pages 4312–4316, 2013. doi: http://dx.doi.org/10.1109/ICIP.2013.6738888

- O. A. B. Penatti, F. B. Silva, E. Valle, V. Gouet-Brunet, and R. da S. Torres. Visual Word Spatial Arrangement for Image Retrieval and Classification. *Pattern Recognition*, 47(2):705 – 720, 2014. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/j.patcog.2013.08.012

- F. B. Silva, S. Tabbone, and R. da S. Torres. BoG: A new approach for graph matching. In *22nd International Conference on Pattern Recognition (ICPR)*, 2014 (To Appear)

Bibliography

[1] C. C. Aggarwal and H. Wang. *Managing and Mining Graph Data*, volume 40 of *Advances in Database Systems*. Springer, New York, NY, USA, 1st edition, 2010.

[2] F. S. P. Andrade, J. Almeida, H. Pedrini, and R. da S. Torres. Fusion of local and global descriptors for content-based image and video retrieval. In *17th Iberoamerican Congress on Pattern Recognition (CIARP)*, volume 7441, pages 845–853. Springer, 2012.

[3] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[4] E. Barbu, P. Héroux, S. Adam, and E. Trupin. Using Bags of Symbols for automatic indexing of graphical document image databases. In *Graphics Recognition. Ten Years Review and Future Perspectives*, volume 3926, pages 195–205. Springer Berlin Heidelberg, 2006.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer Vision Image Understanding*, 110(3):346–359, 2008. ISSN 1077-3142.

[6] E. Bengoetxea. *Inexact Graph Matching Using Estimation of Distribution Algorithms*. PhD thesis, Ecole Nationale Supérieure des Télécommunications, Paris, France, Dec 2002.

[7] M. Bergtholdt, J. Kappes, S. Schmidt, and C. Schnörr. A study of parts-based object class detection using complete graphs. *International Journal of Computer Vision*, 87 (1-2):93–117, 2010.

[8] A. Bolovinou, I. Pratikakis, and S. Perantonis. Bag of Spatio-Visual Words for context inference in scene classification. *Pattern Recognition*, 46(3):1039 – 1053, 2013.

[9] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2559–2566, 2010.

[10] H. Bunke. Inexact graph matching for structural pattern recognition. *Pattern Recognition Letters*, 1(4):245–253, 1983.

[11] H. Bunke. Recent developments in graph matching. In *Proceedings of 15th International Conference on Pattern Recognition, ICPR 2000*, number 2, pages 117–124, 2000.

[12] Y. Cao, C. Wang, Z. Li, L. Zhang, and L. Zhang. Spatial-Bag-of-Features. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3352–3359, 2010.

[13] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[14] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5): 603–619, 2002.

[15] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 18(3):265–298, 2004.

[16] R. da S. Torres and A. X. Falcão. Content-based image retrieval: Theory and applications. *Revista de Informática Teórica e Aplicada*, 13(2):161–185, 2006.

[17] J. Dos Santos, O. Penatti, R. da S. Torres, P.-H. Gosselin, S. Philipp-Foliguet, and A. Falcao. Remote sensing image representation based on hierarchical histogram propagation. In *Geoscience and Remote Sensing Symposium (IGARSS), 2013 IEEE International*, pages 2982–2985, 2013.

[18] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[19] L. Fei-fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[20] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 264–271, 2003.

[21] E. A. Fox, M. A. Gonçalves, and R. Shen. *Theoretical Foundations for Digital Libraries: The 5S (Societies, Scenarios, Spaces, Structures, Streams) Approach*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, San Francisco, CA, USA, 2012.

[22] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematics Software*, 3(3):209–226, 1977.

[23] X. Gao, B. Xiao, D. Tao, and X. Li. A survey of graph edit distance. *Pattern Analysis and Applications*, 13(1):113–129, 2010.

[24] T. Gärtner. A survey of kernels for structured data. *ACM SIGKDD Explorations Newsletter*, 5(1):49–58, 2003.

[25] J.-M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, Jan. 2005.

[26] W.-B. Goh. Strategies for shape matching using skeletons. *Computer Vision and Image Understanding*, 110(3):326–345, 2008.

[27] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[28] V. N. Gudivada and G. S. Jung. Spatial knowledge representation and retrieval in 3d image databases. In *Proceedings of the International Conference on Multimedia Computing and Systems (ICMCS)*, pages 90–97, 1995.

[29] V. N. Gudivada and V. V. Raghavan. Design and evaluation of algorithms for image retrieval by spatial similarity. *ACM Transactions on Information Systems (TOIS)*, 13(2):115–144, 1995.

[30] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979.

[31] M. Hashimoto. *Detecção de objetos por reconhecimento de grafos-chave*. PhD thesis, Universidade de São Paulo, São Paulo, Brasil, Fev 2012.

[32] L. He, C. Y. Han, and W. G. Wee. Object recognition and recovery by skeleton graph matching. In *Proceedings of the 2006 IEEE International Conference on Multimedia and Expo (ICME)*, pages 993–996, 2006.

[33] N. V. Hoíng, V. Gouet-Brunet, M. Rukoz, and M. Manouvrier. Embedding spatial information into image content description for scene retrieval. *Pattern Recognition*, 43(9):3013–3024, 2010.

[34] T. Hou, X. Hou, M. Zhong, and H. Qin. Bag-of-Feature-Graphs: A new paradigm for non-rigid shape retrieval. In *International Conference on Pattern Recognition (ICPR)*, pages 1513–1516, 2012.

[35] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, CVPR '97, pages 762–768. IEEE Computer Society, 1997. ISBN 0-8186-7822-4.

[36] T.-K. Huang, R. C. Weng, and C.-J. Lin. Generalized bradley-terry models and multi-class probability estimates. *Journal of Machine Learning Research*, 7:85–115, 2006.

[37] P. Indyk and R. Motwani. Approximate Nearest Neighbors: Towards removing the curse of dimensionality. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC)*, pages 604–613, 1998.

[38] S. Jouili and S. Tabbone. A hypergraph-based model for graph clustering: Application to image indexing. In *Proceedings of the 13th International Conference on Computer Analysis of Images and Patterns, CAIP 2009*, pages 360–368, 2009.

[39] S. Jouili, I. Mili, and S. Tabbone. Attributed graph matching using local descriptions. In *Advanced Concepts for Intelligent Vision Systems*, pages 89–99, 2009.

[40] S. Jouili, S. Tabbone, and V. Lacroix. Median graph shift: A new clustering algorithm for graph domain. In *Proceedings of the 20th International Conference on Pattern Recognition, ICPR 2010*, pages 950–953, Aug 2010.

[41] S. Karaman, J. Benois-Pineau, R. Mégret, and A. Bugeau. Multi-layer local graph words for object recognition. In *Proceedings of 18th International Conference on Advances in Multimedia Modeling (MMM)*, pages 29–39, 2012.

[42] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2(1–2):83–97, 1955.

[43] H. W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955.

[44] S. Lazebnik, C. Schmid, and J. Ponce. Beyond Bags of Features: Spatial Pyramid matching for recognizing natural scene categories. In *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2169–2178, 2006.

[45] J. Lebrun, P. H. Gosselin, and S. Philipp-Foliguet. Inexact graph matching based on kernels for object retrieval in image databases. *Image and Vision Computing*, 29 (11):716–729, 2011.

[46] L. Liu, L. Wang, and X. Liu. In defense of soft-assignment coding. In *Proceedings of the International Conference on Computer Vision, ICCV 2011*, pages 2486–2493, 2011. ISBN 978-1-4577-1101-5.

[47] Y. Liu and V. Caselles. Spatial string matching for image classification. In *Proceedings of the 20th International Conference on Pattern Recognition (ICPR)*, pages 2937–2940, 2010.

[48] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[49] B. Luo, R. C. Wilson, and E. R. Hancock. Spectral embedding of graphs. *Pattern Recognition*, 36(10):2213–2230, 2003.

[50] B. Luo, R. C. Wilson, and E. R. Hancock. Learning modes of structural variation in graphs. In *Proceedings of the 2003 International Conference on Image Processing (ICIP)*, volume 2, pages 37–40, 2003.

[51] M. W. M. Weber and P. Perona. Unsupervised learning of models for recognition. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 18–32, 2000.

[52] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1-2):43–72, 2005.

[53] J. C. Niebles and L. Fei-fei. A hierarchical model of shape and appearance for human action classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007.

[54] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[55] B. Pang, N. Zhao, D. Korkin, and C.-R. Shyu. Fast protein binding site comparisons using visual words representation. *Bioinformatics*, 28(10):1345–1352, 2012.

[56] J. Papa, A. Falcão, and 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`.

[57] 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, and E. Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[58] O. A. B. Penatti. *Image and video representations based on visual dictionaries*. PhD thesis, University of Campinas, Campinas-SP, Brasil, Nov 2012.

[59] O. A. B. Penatti, E. Valle, and R. da S. Torres. Encoding spatial arrangement of visual words. In *Proceedings of the 16th Iberoamerican Congress conference on Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications (CIARP)*, pages 240–247, 2011.

[60] O. A. B. Penatti, F. B. Silva, E. Valle, V. Gouet-Brunet, and R. da S. Torres. Visual Word Spatial Arrangement for Image Retrieval and Classification. *Pattern Recognition*, 47(2):705 – 720, 2014. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/j.patcog.2013.08.012.

[61] K. Riesen and H. Bunke. IAM graph database repository for graph based pattern recognition and machine learning. In *Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 287–297, 2008.

[62] K. Riesen, M. Neuhaus, and H. Bunke. Bipartite graph matching for computing the edit distance of graphs. In *Proceedings of the 6th IAPR International Conference on Graph-based Representations in Pattern Recognition (GbRPR)*, volume 4538, pages 1–12, 2007.

[63] A. Robles-Kelly and E. R. Hancock. Graph edit distance from spectral seriation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):365–378, 2005.

[64] A. Robles-Kelly and E. R. Hancock. A riemannian approach to graph embedding. *Pattern Recognition*, 40(3):1042–1056, 2007.

[65] A. Rocha, T. Carvalho, H. Jelinek, S. Goldenstein, and J. Wainer. Points of interest and visual dictionaries for automatic retinal lesion detection. *IEEE Transactions on Biomedical Engineering*, 59(8):2244 – 2253, 2012.

[66] A. Rosenfeld. Adjacency in digital pictures. *Information and Control*, 26(1):24–33, 1974.

[67] C. D. Ruberto. Recognition of shapes by attributed skeletal graphs. *Pattern Recognition*, 37(1):21–31, 2004.

[68] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[69] E. Santos, A. Lopes, E. Valle, J. de Almeida, and A. Araújo. Vocabulários visuais para recuperação de informação multimídia. In *Anais do XVI Simpósio Brasileiro em Sistemas Multimidia e Web*, volume 2, page 21–24, 2010 (in Portuguese).

[70] K. C. Santosh, B. Lamiroy, and L. Wendling. Symbol recognition using spatial relations. *Pattern Recognition Letters*, 33(3):331–341, 2012.

[71] S. Savarese, J. Winn, and A. Criminisi. Discriminative object class models of appearance and shape by correlatons. In *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2033–2040, 2006.

[72] T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26 (5):550–571, 2004.

[73] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1):13–32, 1999.

[74] F. B. Silva, S. Goldenstein, S. Tabbone, and R. da S. Torres. Image classification based on Bag of Visual Graphs. In *Proceedings of 20th IEEE International Conference on Image Processing (ICIP)*, pages 4312–4316, 2013. doi: http://dx.doi.org/10.1109/ICIP.2013.6738888.

[75] F. B. Silva, S. Tabbone, and R. da S. Torres. BoG: A new approach for graph matching. In *22nd International Conference on Pattern Recognition (ICPR)*, 2014 (To Appear).

[76] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 370–377, 2005.

[77] R. O. Stehling, M. A. Nascimento, and A. X. Falcão. A compact and efficient image retrieval approach based on border/interior pixel classification. In *Proceedings of the eleventh International Conference on Information and Knowledge Management*, CIKM '02, pages 102–109. ACM, 2002. ISBN 1-58113-492-4.

[78] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1331–1338, 2005.

[79] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988.

[80] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1582–1596, 2010.

[81] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Empowering visual categorization with the GPU. *IEEE Transactions on Multimedia*, 13(1):60–70, 2011.

[82] J. van Gemert, C. Veenman, A. Smeulders, and J.-M. Geusebroek. Visual word ambiguity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7): 1271–1283, 2010.

[83] V. Viitaniemi and J. Laaksonen. Experiments on selection of codebooks for local image feature histograms. In *Proceedings of 10th International Conference on Visual Information Systems (VISUAL)*, volume 5188, pages 126–137, 2008.

[84] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.

[85] D. R. Wilson and T. R. Martinez. Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research*, 6(1):1–34, 1997.

[86] R. C. Wilson, E. R. Hancock, and B. Luo. Pattern vectors from algebraic graph theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7): 1112–1124, 2005.

[87] L. Wiskott, J.-M. Fellous, N. Krüger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):775–779, 1997.

[88] X. Xiaogang, Z. Sun, B. Peng, X. Jin, and W. Liu. An online composite graphics recognition approach based on matching of spatial relation graphs. *International Journal on Document Analysis and Recognition (IJDAR)*, 7(1):44–55, 2004.

[89] L. Xu and I. King. A PCA approach for fast retrieval of structural patterns in attributed graphs. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 31(5):812–817, 2001.

[90] L. Zhou, Z. Zhou, and D. Hu. Scene classification using a multi-resolution Bag-of-Features model. *Pattern Recognition*, 46(1):424 – 433, 2013.

Appendix A

Basic Concepts

Definition A.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 A.2. The **Power Set** of a set $\mathcal{S}$ is the collection of all possible subsets of $\mathcal{S}$.

Definition A.3. Let x and y be two elements of a set $\mathcal{S}$, a **distance function** provides a numeric value that indicates how different these elements are. This type of function assumes a zero value when computing the distance between two equal elements. The lower the similarity between elements, the higher the distance value.

Some of the most popular metrics are listed below.

- Euclidean Distance

$$d(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2}. \tag{A.1}$$

- Manhattan Distance

$$d(\vec{x}, \vec{y}) = \sum_{i=1}^{N} |x_i - y_i|. \tag{A.2}$$

- Discrete Distance

$$d(x, y) = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{if } x \neq y \end{cases} . \tag{A.3}$$

Definition A.4. Given a set $\mathcal{S}$, a **characteristic function** $\mathbf{1}_{\mathcal{S}'} : \mathcal{S} \to \{0, 1\}$ indicates the elements of $\mathcal{S}$ that belong to a subset $\mathcal{S}'$. This function is defined as

$$S' \subset S$$

$$\mathbf{1}_{S'}(x) = \begin{cases} 1 & \text{if } x \in \mathcal{S}' \\ 0 & otherwise \end{cases} .$$

Definition A.5. Given a set $\mathcal{S}$, an **equivalence relation** $\overset{\mathscr{P}}{\sim}$ is a symmetric, transitive, and reflexive relation that defines a partition $\mathscr{P}$ of $\mathcal{S}$. A subset $\mathcal{S}' \in \mathscr{P}$ contains equivalent elements under $\overset{\mathscr{P}}{\sim}$ and it is called an equivalent class of $\mathcal{S}$ by $\overset{\mathscr{P}}{\sim}$.

Definition A.6. Given a set $\mathcal{S}$, a **partition** of $\mathcal{S}$ is a collection of disjoint subsets of $\mathcal{S}$.

$$\mathscr{P} \text{ is a partition of } \mathcal{S} \Leftrightarrow \begin{cases} \emptyset \notin \mathscr{P} \\ \bigcup_{\mathcal{A} \in \mathscr{P}} = \mathcal{S} \\ (\forall \mathcal{A}, \mathcal{B} \in \mathscr{P}) \, [\, \mathcal{A} \neq \mathcal{B} \rightarrow \mathcal{A} \bigcap \mathcal{B} = \emptyset \,] \end{cases}$$

Definition A.7. Given a function $f : \mathcal{A} \rightarrow \mathcal{B}$, **max** and **min** refer, respectively, to the maximum and minimum image values that a function f achieves with a particular domain.

$$\underset{\mathcal{A}}{max} \, (f) = f(y) \Leftrightarrow (\forall x \in \mathcal{A}) \, (\exists y \in \mathcal{A}) \, [\, f(y) \geq f(x) \,]$$

$$\underset{\mathcal{A}}{min} \, (f) = f(y) \Leftrightarrow (\forall x \in \mathcal{A}) \, (\exists y \in \mathcal{A}) \, [\, f(y) \leq f(x) \,]$$

Definition A.8. Given a function $f : \mathcal{A} \rightarrow \mathcal{B}$, **argmax** and **argmin** refers to the elements of the domain of f that achieve the maximum and minimum image values, respectively.

$$\underset{\mathcal{A}}{argmax} \, (f) = y \Leftrightarrow (\forall x \in \mathcal{A}) \, (\exists y \in \mathcal{A}) \, [\, f(y) \geq f(x) \,]$$

$$\underset{\mathcal{A}}{argmin} \, (f) = y \Leftrightarrow (\forall x \in \mathcal{A}) \, (\exists y \in \mathcal{A}) \, [\, f(y) \leq f(x) \,]$$

Definition A.9. A **centroid** refers to the element that represents the center of a space. Given a set $\mathcal{S}$ and a *distance function* $f : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}$, the **centroid** c of $\mathcal{S}$ is defined as:

$$\Big(\forall x \in \mathcal{S}\Big) \left[\sum_{z \in \mathcal{S}} f(c, z) \leq \sum_{z \in \mathcal{S}} f(x, z) \right]$$

Definition A.10. A **weigthed graph** is a tuple $G = (\mathcal{V}, \mathcal{E}, \phi)$, where $\mathcal{V}$ is a set of vertices, $\mathcal{E}$ is a set of edges that link two vertices of $\mathcal{V}$, and ϕ is a function $\mathcal{E} \rightarrow \mathbb{R}$ that associates each edge of $\mathcal{E}$ with a numerical value (weight).

Acronyms

ARG Attributed Relational Graph. 7

BoFG Bag-of-Feature-Graphs. 7

BoG Bag of Graphs. xi, 2, 13, 14, 21, 24, 28, 31, 59–61

BoSG Bag of Singleton Graphs. xi, xxiii, 2, 21, 24, 31, 32, 34–36, 38–43, 45, 59–61

BoVG Bag of Visual Graphs. xi, xxiii, 2, 24, 26, 31, 46, 48, 49, 54, 60, 61

BoVW Bag of Visual Words. 1, 8, 10, 48, 60

BoW Bag of Words. 1, 2, 7, 8, 10, 11, 24, 31, 36, 46, 48, 49, 54, 59

GoI graph of interest. 17, 19, 22–24, 26, 28

HEOM Heterogeneous Euclidean Overlap Metric. 23

KDTree K-Dimensional Tree. 60, 61

KNN K-Nearest Neighbor. 37, 38, 42, 45

LSH Locality-Sensitive Hashing. 60, 61

OPF Optimum-Path Forest. 45

SOG Spatial Orientation Graph. 7

SP Spatial Pyramids. 11, 48, 49, 54

SRG Spatial Relational Graph. 7

SVM Support Vector Machine. 45

VSM Vector Space Model. 8

WSA Word Spatial Arrangement. 11, 48, 54

Index