UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
FACULDADE DE MEDICINA DE RIBEIRÃO PRETO
INSTITUTO DE QUÍMICA DE SÃO CARLOS

SAMUEL ZANFERDINI OLIVA

Métodos para melhorar a semântica em buscas por similaridade, diversidade e sumarização de dados baseados no conceito da caminhada do turista

São Carlos
2019

UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
FACULDADE DE MEDICINA DE RIBEIRÃO PRETO
INSTITUTO DE QUÍMICA DE SÃO CARLOS

SAMUEL ZANFERDINI OLIVA

Methods to improve the semantics of similarity search, diversity and data summarization based on the tourist walk concept

São Carlos
2019

SAMUEL ZANFERDINI OLIVA

Methods to improve the semantics of similarity search, diversity and data summarization based on the tourist walk concept

VERSÃO CORRIGIDA

Doctoral thesis presented to the Interunit Graduate Program in Bioengineering - School of Engineering of São Carlos / Faculty of Medicine of Ribeirão Preto / Institute of Chemistry of São Carlos of the University of São Paulo, as a requirement for obtaining the Degree of Doctor in Science.

Concentration area: Bioengineering

Advisor: Prof. Dr. Joaquim Cezar Felipe

São Carlos

2019

SAMUEL ZANFERDINI OLIVA

Métodos para melhorar a semântica em buscas por similaridade, diversidade e sumarização de dados baseados no conceito da caminhada do turista

VERSÃO CORRIGIDA

Tese apresentada ao Programa de Pós-Graduação Interunidades em Bioengenharia da Escola de Engenharia de São Carlos – Faculdade de Medicina de Ribeirão Preto e Instituto de Química de São Carlos da Universidade de São Paulo, como requisito para a obtenção do Título de Doutor em Ciências.

Área de concentração: Bioengenharia

Orientador: Prof. Dr. Joaquim Cezar Felipe

São Carlos

2019

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da EESC/USP com os dados inseridos pelo(a) autor(a).

Oliva, Samuel Zanferdini
O48m Métodos para melhorar a semântica em buscas por similaridade, diversidade e summarização de dados baseados no conceito da caminhada do turista / Samuel Zanferdini Oliva; orientador Joaquim Cezar Felipe. São Carlos, 2019.

Tese (Doutorado) - Programa de Pós-Graduação Interunidades em Bioengenharia e Área de Concentração em Bioengenharia -- Escola de Engenharia de São Carlos; Faculdade de Medicina de Ribeirão Preto; Instituto de Química de São Carlos, da Universidade de São Paulo, 2019.

1. Data retrieval. 2. Content-based image retrieval. 3. Tourist walk. 4. Similarity query. 5. Query result diversification. 6. Data sampling. 7. Data summarization. I. Título.

Eduardo Graziosi Silva - CRB - 8/8907

FOLHA DE JULGAMENTO

Candidato(a): Samuel Zanferdini Oliva

TÍTULO: " Métodos para melhorar a semântica em buscas por similaridade, diversidade e sumarização de dados baseados no conceito da caminhada do turista"

Data da defesa: 13/11/2019

Comissão Julgadora	Assinatura	Resultado
Prof(a). Dr(a). Joaquim Cezar Felipe FFCLRP/USP		Não votante
Prof(a). Dr(a). Renato Bueno UFSCar		aprovado
Prof(a). Dr(a). Marcela Xavier Ribeiro UFSCar	Marcela Xavier Ribeiro	aprovado
Prof(a). Dr(a). Luiz Otávio Murta Junior FFCLRP/USP		aprovado
Prof(a). Dr(a). Marilde Terezinha Prado Santos UFSCar	Marilde Santos	Aprovado
Prof(a). Dr(a). Maria Cristiane Barbosa Galvão FMRP/USP		aprovado

DEDICATORY

To my parents and wife by the immeasurable support, patience, and care.

ACKNOWLEDGEMENTS

I would like to express my gratitude to Prof. Dr. Joaquim Cezar Felipe, my thesis advisor, who has collaborated very much with my scientific growth and provided invaluable and golden advice during conversations and discussions.

For Capes, I owe thanks for the continuous financial support, which has allowed the development and achievement of this project.

For the University of São Paulo and the Interunit Graduate Program in Bioengineering for hosting me during my Master and Doctorate period. I express also my gratitude to the professors that contributed with knowledge and experience.

Lastly, I thank God for all the protection and good physical and mental health that provided me to face the hard work and challenges of this thesis.

ABSTRACT

OLIVA, SZ. Methods to improve the semantics of similarity search, diversity and data summarization based on the tourist walk concept. 2019. 106 f. Thesis – Interunit Pos-graduation Program in Bioengineering of the School of Engineering of São Carlos - Faculty of Medicine of Ribeirão Preto and Institute of Chemistry of São Carlos of the University of São Paulo, São Carlos, 2019.

Due to the large increase in the amount of data that has occurred recently, several approaches seeking efficiency to deal with data storage and retrieval have been proposed in the literature, including those that study query by similarity and those that consider the diversification of query results. Similarly, different methods have been proposed in order to perform summarization to select representative samples of databases. In this work, methods for similarity, query diversification and data summarization are proposed, implemented and evaluated. This development has as reference the tourist walk heuristic, which consists of a walker going through a set of points within a multidimensional space. Hence, three approaches are proposed: the first consists of the SimWalk method to perform similarity searches; the second proposal corresponds to the DivWalk method whose purpose is to construct result sets from diversified search elements; the third presents the SummarizationWalk method for database summarization, considering the volume and amount of elements of data clusterings. The approaches were developed and evaluated with artificial and real databases. In the conducted experiments, SimWalk presented higher accuracy when compared to traditional similarity retrieval methods. DivWalk showed greater variances in the results, demonstrating that this method produces a better data distribution in the databases. SummarizationWalk presented better results in the retrieval of subsets, considering the metrics of selected elements in relation to volume and amount of elements by database clusters. The studies presented here show that the proposed methods for similarity search, query diversification and data summarization represent an optimization with respect to the state-of-the-art, thus consisting of contributions to the area of data retrieval.

Keywords: Data retrieval. Content-based image retrieval. Tourist walk. Similarity query. Query result diversification. Data sampling. Data summarization.

RESUMO

OLIVA, SZ. Métodos para melhorar a semântica em buscas por similaridade, diversidade e summarização de dados baseados no conceito da caminhada do turista. 2019. 106 f. Tese – Programa de Pós-Graduação Interunidades em Bioengenharia da Escola de Engenharia de São Carlos – Faculdade de Medicina de Ribeirão Preto e Instituto de Química de São Carlos da Universidade de São Paulo, São Carlos, 2019.

Devido ao grande aumento da quantidade e variedade de dados ocorrido recentemente, diversas abordagens buscando a eficiência para lidar com o armazenamento e a recuperação de dados têm sido propostas na literatura, dentre elas as que estudam recuperação baseada em similaridade e as que consideram a diversificação de resultados. Do mesmo modo, diferentes métodos têm sido propostos a fim de realizar a sumarização de dados, com a finalidade de selecionar amostras representativas das bases. Neste trabalho, métodos para buscas por similaridade, diversificação de consultas e sumarização de dados são propostos, implementados e avaliados. Esse desenvolvimento tem como referência a heurística da caminhada turista, a qual consiste de um caminhante percorrendo um conjunto de pontos dentro de um espaço multidimensional. Assim, são propostas três abordagens: a primeira consiste do método SimWalk, para realizar buscas por similaridade; a segunda proposta corresponde ao método DivWalk, cuja finalidade é construir conjuntos resultantes de buscas considerando elementos diversificados; a terceira apresenta o método SummarizationWalk, para realizar a sumarização de bases de dados, considerando o volume e a quantidade de elementos por agrupamentos de elementos de dados. As abordagens foram desenvolvidas e testadas com bases de dados artificiais e reais. Nos experimentos conduzidos, o SimWalk apresentou maior precisão, quando comparado com os métodos tradicionais de recuperação por similaridade. O DivWalk apresentou maiores variâncias nos resultados, demonstrando que este método produz uma melhor distribuição dos dados nas bases. O SummarizationWalk apresentou melhores resultados na recuperação de subconjuntos, considerando as métricas de elementos selecionados em relação ao volume e à quantidade de elementos por agrupamentos das bases. Os estudos aqui apresentados mostram que os métodos propostos para buscas por similaridade, diversificação de consultas e sumarização de dados representam uma otimização em relação ao estado da arte, consistindo, assim, de contribuições para a área de recuperação de dados.

Palavras-chave: Recuperação da informação. Recuperação de imagens baseada em conteúdo. Caminhada do turista. Buscas por similaridade. Diversificação de resultados de consultas. Amostragem de dados. Sumarização de dados.

CONTENTS

Chapter 1 - INTRODUCTION

In this section, we introduce the main subject regarding the doctorate project. So, we organized this section in subsections, namely: contextualization, motivation, and objectives. Regarding the subject of this project, which is the development of new methods for similarity query, diversity and summarization related to the tourist walk concept, in this chapter, we first contextualize the topics addressed in the study: tourist walk, similarity search, query diversification and summarization. After that, we point out our main motivations for developing this project. Finally, we explain the main and specific objectives that we intended to reach.

1.1. Contextualization

Random walk techniques have been widely studied, generating important results in different knowledge fields and applying to a vast range of applications, such as physics (HARDIMAN, 2013; STANLEY; BULDYREV, 2001), mathematics (NEWMAN, 2005; PONS, 2006), image segmentation (GRADY, 2006), stochastic process theory (TABRIZI, 2013), bioinformatics (MACROPOL, 2009), among others. The random walk can be defined as a stochastic process that describes a trajectory based on a sequence of random steps performed on a mathematical metric space. In graphs, for instance, the random walk can be seen as a walker starting from a given vertex, selecting one of its adjacent vertices randomly and moving to it, and successively moving between vertices and proceeding with the same logic until complete the predefined amount of steps.

Accordingly, the study of deterministic walks has also drawn the attention of the scientific community (BUNIMOVICH, 2004; FREUND; GRASSBERGER, 1992; KINOUCHI; MARTINEZ; LIMA; LOURENCO *et al.*, 2002; LIMA; MARTINEZ; KINOUCHI, 2001). The process implicated on the deterministic walks differs from the random walk with respect to the predefined rule for the selection of the trajectory to be taken by the walker. Whereas in the random walk the steps are taken following an equal probability, a deterministic walk follows a specific walker movement rule.

Tourist walk is a deterministic partially self-repulsive walk. Lima et al. (LIMA; MARTINEZ; KINOUCHI, 2001) define the tourist walk as a walker, also known as a tourist,

which wishes to visit cities distributed on a map of d dimensions. The tourist starts from a given city of that map and moves to the nearest city that still has not been visited in the last μ steps to perform his walking. The μ value is called memory and represents the time step (or the number of steps, or the number of data points visited) required to repeating data points already visited (TERCARIOL; MARTINEZ, 2005). In tourist walk, the resulting trajectory contains a non-periodic initial part of t steps, also known as transient, and ends in a stable cycle period of p steps, called attractor, where the same data points are visited repeatedly in the same order (LIMA; MARTINEZ; KINOUCHI, 2001; STANLEY; BULDYREV, 2001).

Tourist walk algorithms has increasingly attracted the attention of researches and are already applied on various computational approaches, for instance, image analysis (BACKES; BRUNO; CAMPITELI; MARTINEZ, 2006; BACKES; GONCALVES; MARTINEZ; BRUNO, 2010; CAMPITELI; MARTINEZ; BRUNO, 2006), pattern recognition (CAMPITELI; BATISTA; KINOUCHI; MARTINEZ, 2006), and classification based on high level pattern (SILVA; ZHAO, 2015). However, in the similarity retrieval context, as well as in the summarization context, this technique is still unexplored.

Thus, in this work we use the tourist walk concept as a starting point to create methods for similarity search, diversity search and summarization, and analyze the ability of the proposed methods to capture structural and semantical information.

Similarity search allows performing queries on a database checking which instances are more similar to a reference one, using a set of features extracted from them. The retrieving of the most similar objects can be done using two main well-known approaches: the *k-nearest neighbor query* (k-NNq) and the *range query* (Rq). k-NNq enables one to find, given an initial query object q and an integer number *k*, the *k* objects more similar to q. Rq allows, through an initial query object q and a distance range of size r, to retrieve all objects that are within the distance r from q. Through similarity search, content-based data retrieval systems allow processing queries on a database taking into account similarity criteria amongst instances.

Furthermore, it is worth stressing that large data sets can hold information with a high degree of similarity amongst their data objects and, consequently, similarity methods can retrieve redundant objects when a query is performed using a data object that is within a large and homogenous database. So, an alternative way to deal with this redundancy can be the inclusion of a diversity factor in the query results returned by similarity searches allowing, this way, to obtain a more diversified result subset.

Result set diversification stems from information retrieval studies (AGRAWAL, 2009; DOU; HU; CHEN; SONG *et al.*, 2011) and has stimulated research on various applications such as web search (BORODIN; LEE; YE, 2012; CAPANNINI, 2011; GOLLAPUDI; SHARMA, 2009) and recommendation systems (YU, C.; LAKSHMANAN, L. V.; AMER-

YAHIA, S., 2009; ZHANG; HURLEY, 2008; ZIEGLER; MCNEE; KONSTAN; LAUSEN, 2005). In database and information retrieval systems, for example, the use of a diversity factor can make these applications capable of ranking objects concerning their relevant and diverse characteristics simultaneously. In other words, the objects returned as the application's result set must be as relevant as possible regarding the query object and, at the same time, must also be as diverse as possible. Nonetheless, whereas returning relevant results to a query is relatively simple and there are already several studies in the database and information retrieval fields, the diversity is still a more challenging issue to deal with (CARTERETTE, 2009). In the present study, we intend to contribute both in the context of similarity search and on query result diversification.

Data summarization methodologies have been demonstrated to be a relevant and useful approach to support data analysis of large databases. It has been studied in a wide range of domains such as healthcare, text analysis, network traffic monitoring, and so on (AHMED, 2019a; b; AHMED; MAHMOOD; MAHER, 2014; ELFAYOUMY; THOPPIL, 2014). Data summarization can be described as the process of selecting concise and representative information of an original dataset.

Summarization approaches can be divided into different taxonomies and the two major categories are: summarization for structured data and for unstructured data. The first refers to the information that is organized in rows and columns within a matrix or file. The second is related to the information that doesn't have a pre-defined model (AHMED, 2019a).

Therefore, data summarization can be used to downsize a very bulky database, by generating a subset of it. This summarization represents a significant way to mitigate queries and analysis that could be intricate and time-consuming. In order to represent the original database in applications such as, for example, similarity search, this subset has to preserve the main features of it, such as class distribution, data density and space occupation. In the present study, we propose a new data summarization approach with this intention.

1.2. Motivation

Methods for content-based data retrieval have several relevant applications. We can find significant example in the context of medical imaging retrieval (AKGUL; RUBIN; NAPEL; BEAULIEU *et al.*, 2011; KUMAR; KIM; CAI; FULHAM *et al.*, 2013; MÜLLER; MICHOUX; BANDON; GEISSBUHLER, 2004) which presents a recognized relevance in aiding diagnosis, since it allows computer systems to provide to the medical expert access to images that, associated with other medical records, contain a whole chain of clinical,

therapeutic, and epidemiological information, in addition to the possibility of analyzing and visualizing these images.

However, in spite of the numerous efforts expended by the scientific community, there is still an inherent difficult regarding the process of automatic analysis, known as *semantic gap*, which consists of the discrepancy between the result returned by a computer system and the result expected by the user (TRAINA; TRAINA JR; CIFERRI; RIBEIRO *et al.*, 2009). This happens due to the complexity inherent in the analysis performed by the human brain, which takes into account the overlapping of multiple semantic aspects related to the information related to the objects.

Several studies have been conducted aiming at approximate the result of the computational analysis to the human analysis. Most of them focus on proposing or optimizing feature extractors (BALAN; TRAINA; RIBEIRO; MARQUES *et al.*, 2012; FELIPE; OLIOTI; TRAINA; RIBEIRO *et al.*, 2005; KUMAR; ONG; RANGANATH; ONG *et al.*, 2006), or on evaluating and modifying distance functions (FELIPE; TRAINA; TRAINA, 2009; RUBNER; TOMASI, 2013; SANTINI; JAIN, 1999), or on developing new techniques of relevance feedback (BUGATTI; TRAINA; TRAINA, 2011; DE AZEVEDO-MARQUES; ROSA; TRAINA; TRAINA *et al.*, 2008). Notwithstanding, the content-based retrieval methods in large majority determine the result set based strictly on evaluations of distances between candidates objects concerning the query object.

We hypothesize that it is possible to improve the semantics, i.e. to increase the precision of the retrieval process, by applying a method that takes into account inter-relations amongst the objects of the database. For this, we propose a new retrieval method (Method 1) that chooses the next object by considering its distance to all objects that are already part of the result set, instead of just taking the initial query object into account. This method represents an evolution of the tourist walk application. Tourist walk provides a resulting set of similar objects by choosing as the next object the one closer to the last. Silva and Zhao (SILVA; ZHAO, 2015), for instance, applied the tourist walk on complex networks for the classification of database objects, demonstrating that the results were more semantic, leading to a higher classification accuracy in non-trivial situations.

Furthermore, we consider that the proposed Method 1 can be a hybrid search method holding properties of both range and k-nearest neighbor similarity methods, since the modifications of its parameters allow to control the coverage of the final result sets.

Another important issue in this context is the large amount of instances present in the databases handled by the users in numerous general applications. These databases usually present high density, due to the occurrence of very similar or repetitive instances. Considering image databases, for example, the representation based on feature vectors makes that different images can present very similar vectors, leading to the overlapping of

instances in the feature space. The large volume of these databases makes the query process time-consuming, even with the use of appropriate indexing structures.

In various situations, it is undesirable that the retrieved result set contains amounts of very similar objects, since this impoverishes the analysis process that can be conducted by the expert. In order to make the result set more representative, it is desired to have a certain degree of diversity between the objects, though within the limits of a predefined similarity. Diversity based queries have several applications, such as recommendation systems (YU, C.; LAKSHMANAN, L.; AMER-YAHIA, S., 2009; ZHANG; HURLEY, 2008; ZIEGLER; MCNEE; KONSTAN; LAUSEN, 2005), sponsored search advertisement (FEUERSTEIN; HEIBER; MARTINEZ-VIADEMONTE; BAEZA-YATES, 2007), structured databases (DEMIDOVA; FANKHAUSER; ZHOU; NEJDL, 2010; FRATERNALI; MARTINENGHI; TAGLIASACCHI, 2012; LIU; SUN; CHEN, 2009; VEE; SRIVASTAVA; SHANMUGASUNDARAM; BHAT *et al.*, 2008), web searches (BORODIN; LEE; YE, 2012; CAPANNINI; NARDINI; PEREGO; SILVESTRI, 2011; GOLLAPUDI; SHARMA, 2009; VIEIRA; RAZENTE; BARIONI; HADJIELEFTHERIOU *et al.*, 2011), information retrieval (AGRAWAL; GOLLAPUDI; HALVERSON; IEONG, 2009; DOU; HU; CHEN; SONG *et al.*, 2011), and similarity searches (SANTOS; OLIVEIRA; FERREIRA; CORDEIRO *et al.*, 2013; SKOPAL; DOHNAL; BATKO; ZEZULA, 2009).

In many cases, the similarity and diversity combination becomes desirable, into a process called diversification, and there are several studies regarding the development of algorithms in order to promote the diversification of queries (BORODIN; LEE; YE, 2012; CAPANNINI; NARDINI; PEREGO; SILVESTRI, 2011; DEMIDOVA; FANKHAUSER; ZHOU; NEJDL, 2010; FRATERNALI; MARTINENGHI; TAGLIASACCHI, 2012; GOLLAPUDI; SHARMA, 2009; LIU; SUN; CHEN, 2009; VEE; SRIVASTAVA; SHANMUGASUNDARAM; BHAT *et al.*, 2008; VIEIRA, 2011). However, the majority of them explore only aspects related to time performance and do not consider the suitability of the set of objects as representatives of the broader search subspace. The existing methods set up the result set by adding the closest and the farthest objects to the query object and it can be seen as an important drawback when considering semantic aspects. Thus, in this work, we propose a new method (Method 2) to combine similarity and diversity, which generates a result set where the objects are uniformly distributed in the search subspace.

Still considering the problem of time-consuming and poor representativeness of result sets from large and dense databases, another way to mitigate this issue is to use some data summarization technique. The purpose is to have a sample of the database, which is somehow representative of it, maintains its most relevant characteristics and thus can be used in place of it when performing content-based queries. The data summarization process can generate a more compact and representative database, which we called *sample base*,

that would keep the characteristics of the original database and could replace it in query processes, making these queries more effective in terms of diversification of the result set and more efficient regarding the performance. This *sample base* would work as a representative *atlas* of the original database for the queries performed by experts.

Existing sampling techniques are based on randomness and do not allow the user to set up parameters that will guarantee some features relevant to the context, such as preserving the original classes of data, or preserving local data densities, or preserving the occupation of the feature space. Thus, we propose a new sampling method (Method 3) that uses clustering and diversification to address these relevant features.

As far as we know, there are not studies in the literature on the applicability and effectiveness of the tourist walk concept being used for content-based data retrieval, as well as its application for result set diversification and for database sampling. Our perspective is that the methods proposed in this work can contribute to improve semantics in the processes of similarity search, diversity search and data sampling.

1.3. Objectives

This work consists of proposing new methods for similarity and diversity search and for data sampling, aiming at improving the semantics intrinsic to these environments. On the whole, the proposed methods are based on the idea of considering not only the query object for constructing the result sets, but all the objects previously selected.

There are three specific application contexts, and for each one we developed a specific method. The objectives of each method are:

Method 1: perform content-based searches whose result sets present gains on precision and thus reducing the semantic gap, measured through the match of the classes of the objects present in the result set with the class of the query object.

Method 2: perform content-based searches based on similarity and diversity, providing the user with the control of the balance between similarity and diversity and generating a well-distributed result set.

Method 3: generate a database sampling that reproduces key characteristics of the original database, such as keeping the original distribution of data classes, keeping local data density and keeping the volume of the feature space for each class. This allows to perform searches and analysis on the sample base, reaching results that reflect or even improve the results that would be generate on the original database, with relevant gains of performance and reducing the semantic gap.

The study was carried out through the implementation of algorithms and their respective evaluations, comparing them to baseline methods from the literature and using artificial and real databases available for research purposes.

1.4. Document Organization

This document text is organized considering the following structure:

- **Chapter 2:** presents the referential theory regarding the fundamental concepts related to content-based information retrieval, also discussing aspects of data multimedia retrieving, such as medical images. Moreover, we present similarity query methods concepts highlighting their main characteristics.
- **Chapter 3:** introduces the concepts and techniques related to the query result diversification and explains how diversity is inserted and balanced when constructing the result of a query.
- **Chapter 4:** presents the differences and similarities between data sampling and summarization and the main methods proposed in the literature to deal with these issues.
- **Chapter 5:** describes the main deterministic tourist walk techniques, and some proposed algorithms that use this approach. The chapter also emphasizes the features of this approach when applied to image classification and pattern recognition.
- **Chapter 6:** presents a novel method that aims to perform information and image retrieving based on the deterministic tourist walk approach. The strategy adopted by this proposed algorithm can contribute to reduce the semantic gap in similarity queries.
- **Chapter 7:** introduces a new method for diversifying query results, which is also based on the deterministic tourist walk approach. The premise of this work is that the proposed method can improve the quality of the results retrieving data elements that is more spread throughout the databases.
- **Chapter 8:** proposes a novel method for data sampling and summarization aiming to improve the performance when selecting a representative subset of databases.
- **Chapter 9:** presents the conclusions and possible future works.

Chapter 2 - CONTENT-BASED RETRIEVAL

2.1. Initial Considerations

In a content-based retrieval environment, the representation of conventional data and complex data takes into account a set of features (feature vector) that describe each object of the database.

For images, for example, it is required to extract a set of pre-defined representative and inherent features of this type of data. These features are used instead of the data by itself when similarity comparisons are performed (GHOSH; AGRAWAL; MOTWANI, 2018). This approach, for example, has been adopted in Content Based Image Retrieval (CBIR) systems, that usually use features based on color, texture or shape for image representation (ASERY; SUNKARIA; MARWAHA; SHARMA, 2018). It is important to highlight that each data domain has particular properties that are used in the data representation.

Information retrieval, in turn, deals with searching documents within collections and can also perform content-based retrieval, when these documents are represented by a set of inherent features.

2.2. Information Retrieval (IR)

The term Information Retrieval (IR) is related to the process of finding documents (words, phrases, etc.) of an unstructured data (for example, text) that satisfies an information request from within collections stored on computers or on the internet (MANNING; RAGHAVAN; SCHÜTZE, 2010).

The IR process consists of finding in a document collection (corpus), which of them correspond to the information need of the user. Hence, the IR system's user is interested in obtaining "information" about a subject and not just in retrieve data that satisfy a certain search expression.

An IR system must represent the content of a document collection and present it to the user in a way that allows him to rapidly select items, which attend total or partially his need of information, this defined through a search expression. The following steps can

represent an IR process: Documents (Collection), Representation, Query, and Information Need (ROBERTSON, 1977b). The Documents step refers to the information (object, texts, image, audio, etc.) that is stored in digital media. The Representation step describes each document through its content, thus, through the analysis of its content, concepts can be extracted and translated for an indexing language. The user specifies an Information Need, which is then parsed and transformed the same way as the documents. Then, the Query is processed to obtain the retrieved documents. In the Representation step, several models have been proposed to handle the documents when a query is performed. The two most used models in IR literature are the vector space model and the probabilistic models.

2.2.1. Vector Space Model

The vector space model is an algebraic model used for representing documents as vectors (SALTON, 1975). These vectors contain terms such as words and phrases, when words are used as terms, then every word in the vocabulary becomes an independent dimension in a very high dimensional vector space. Case a term is in the text, the text-vector gets a non-zero value along the dimension corresponding to the term.

In a query, the model measures the similarity between the query vector and the document vector in order to assign a score to a document. Usually, the cosine of the angle between two vectors is used as the numeric similarity, due to its property of defining 1 for identical vectors and 0 for orthogonal vectors.

Another way to measure the similarity is through the dot product between two vectors. Thus, if $\vec{D}$ is the document vector and $\vec{Q}$ is the query vector then the similarity of document $\vec{D}$ to query $\vec{Q}$ can be represented by:

$$Sim(\vec{D}, \vec{Q}) = \sum_{t_i \in Q,D} w_{t_{iQ}} \cdot w_{t_{iD}} \quad (1)$$

where $w_{t_{iQ}}$is the value of the ith object in the query vector $\vec{Q}$, and $w_{t_{iD}}$ is the ith object in the document vector $\vec{D}$.

2.2.2. Probabilistic Models

The probabilistic models define that documents in a collection should be ranked by decreasing probability of their relevance to a query, which is referenced in the literature as

probabilistic ranking principle (ROBERTSON, 1977a). Probabilistic IR models estimate the probability of relevance of documents for a query. This estimation is the most important part of the model and since many probabilistic models have been proposed, each one of them is based on a different probability estimation technique.

The common basis for these models is: the probability of relevance for document D is denoted by $P(R|D)$. Thus, the documents can be ranked by $\log\frac{P(R|D)}{P(\bar{R}|D)}$, where $P(\bar{R}|D)$ is the probability that the document is non-relevant.

2.3. Content Based Image Retrieval (CBIR)

CBIR is a process that allows performing queries on image databases considering the similarity factor. CBIR systems comprise a set of steps such as image preprocessing, feature extraction, similarity evaluation between images, and image indexing techniques. CBIR provides, as query result, an image set ordered by its similarity to the query image.

Images are complex data that to be computationally analyzed and compared regarding similarity requires finding a numerical representation for its content, considering its visual features such as color, shape and texture.

Image preprocessing includes computational techniques whereby we can perform modifications on an image in order to highlight features of interest in the image, to reduce noises, and other operations.

Image segmentation comprises the use of computational methods in order to separate regions or objects of interest from a particular image so that the highlighted regions can be further interpreted and evaluated for classification or pattern recognition techniques.

Feature extraction is performed through algorithms that allow extracting values, which represent visual aspects such as texture, color and shape of images. Through these features, images can be numerically represented. Thus, we can use feature vector to perform content-based image retrieval, data mining, as well as image indexing (FELIPE; OLIOTI; TRAINA; RIBEIRO *et al.*, 2005).

Color-based feature extraction can be done, for instance, through color histograms, whose mechanism allows performing a numerical mapping of the image from the quantization in certain levels of colors (PASS; ZABIH; MILLER, 1997). Furthermore, histograms and its variations can be used in similarity search (FELIPE; TRAINA; TRAINA JR, 2006; KO; LEE; BYUN, 2000).

Texture-based feature extraction can be elaborated using a range of existing approaches (FELIPE; TRAINA; TRAINA, 2003), among which we can mention techniques

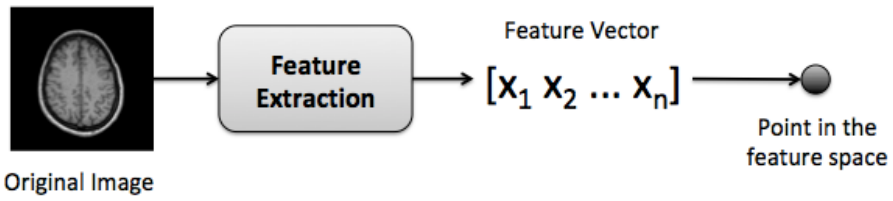
based on Gabor filters (MANJUNATH; MA, 1996) and on co-occurrence matrices (HARALICK; SHANMUGAM, 1973),

Shape-based feature extraction consists of the an approach that considers shapes and contours that can be found in images (FELIPE; RIBEIRO; SOUSA; TRAINA *et al.*, 2006). We can perform this contours extraction using methods such as the chain code or Fourier Transform (KHOTANZAD; HONG, 1990).

2.4. Feature Extraction

One of the main components of CBIR systems is responsible for the feature extraction, because features extracted from images are used to perform their retrieval. Generally, in the image domain, the feature extraction is performed over raw data, for example, over the image pixels.

Figure 1 - Illustration of the feature extraction process.



Source: Made by the author.

Good feature extractors are crucial to the success of the content-based process. Several feature vectors obtained by different extractors can represent an image. This is directly related to similarity queries because according to the extracted features quality the results can be different. Comparisons between feature vectors are performed through distance functions.

2.5. Distance Functions

Distance functions provide the measures that express the dissimilarity between pair of objects. Thus, when two objects are very similar to each other the value of this function is smaller; in other words, a distance equal to zero indicates total similarity. Thereby, the

distance function choice is very important for the similarity search applications, and an expert in the field must choose which one fit best depending on the context.

There is a family of distance functions known as Minkowski family or L_p, in which particular cases rely on the numerical value p, Eq. 2 can define this family.

$$L_p(s_a, s_b) = \sqrt[p]{\sum_{i=1}^{n} |s_{ai} - s_{bi}|^p} \quad (2)$$

where n is the number of dimensions and $L_p(s_a, s_b)$ is the distance between the object s_a and the object s_b, which have dimensionality equal to n (ZEZULA; AMATO; DOHNAL; BATKO, 2006).

Three functions of this family are widely used in similarity comparison operations, which are obtained with the modification of the p values: Manhattan or City-Block distance (L_1), Euclidean distance (L_2) and Chebychev distance (L_∞).

Metric distance functions are measures of dissimilarity that present a set of properties and should satisfy the following rules:

- Commutativity: $d(x, y) = d(y, x)$;
- Nonnegativity: $0 \leq d(x, y) < \infty$;
- Reflexivity: $d(x, y) = 0 \; se \; x = y$;
- Triangle inequality: $d(x, y) \leq d(x, z) + d(z, y)$,

where x, y and z are arbitrary data points.

2.6. Similarity Search

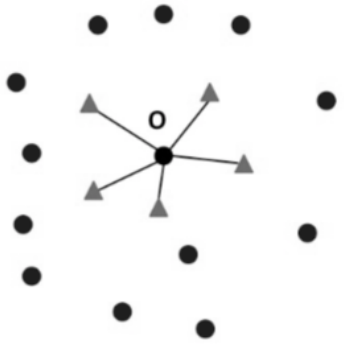
Content-based retrieval systems use the concept of similarity search for retrieving objects from a database based on their content. A similarity search allows performing queries on a database by checking which objects have certain features more similar to a search object. The task of retrieving the most similar objects can be accomplished using two techniques: k-nearest neighbor query and range query.

2.6.1. K-Nearest Neighbor Query (k-NNq)

The k-Nearest Neighbor query (k-NNq) consists of finding, given a query object and an integer number k, the k objects closest to the query one. Thus, the k objects more similar to the query object are retrieved, regardless of their distances. We can use Figure 2 to

illustrate a k-NNq. Figure 2 presents a query object (O) and an integer $k = 5$, defining as the retrieving parameter the five nearest objects to (O). The five points in triangular shape represent the objects retrieved by the query.

Figure 2 - Illustration of a k-nearest neighbor query.

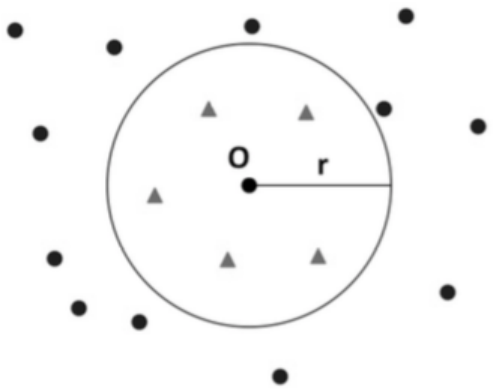


Source: Made by the author.

2.6.2. Range query (Rq)

The range query (Rq) consists of performing a search considering two parameters, a query object and a radius of size r, and through them retrieve all objects located on a distance equal or less than r from the query object, which can be seen as similar. We can use Figure 3 to illustrate Rq considering a query object (O), restricted by a radius distance r. The query returns the objects (represented by triangular shape points) which are within the radius r.

Figure 3 - Illustration of a range query.



Source: Made by the author.

2.7. Final Considerations

This chapter presented the basic concepts related to the content-based search. Here, we can find the overall idea of feature extraction for image and information domain. Furthermore, we also explained how operations are performed in order to compare these domains through distance functions. Besides, we introduced concepts regarding traditional similarity query methods and their main characteristics.

Chapter 3 - QUERY RESULT DIVERSIFICATION

3.1. Initial Considerations

In this chapter, we introduce definitions and settings of the query result diversification problem and two known methods such as Swap and MMR (Maximal Marginal Relevance), which were used to compare with our proposed method.

Firstly, we present preliminaries about the Result Diversification Problem, its fundamental definition with respect to similarity and diversity factors and its formal definition providing a mathematical notation.

Secondly, we explain one of the simplest methods to construct the result set, called Swap. We describe how its algorithm works and its main characteristics regarding the capability of constructing query results considering diversity.

Finally, we explicate about one of the earliest proposals to re-ranks elements, which include diversity in query results. This proposal is called MMR and we describe how it iteratively constructs the query result set selecting elements considering similarity and diversity.

3.2. Result Diversification Problem

The Result Diversification Problem can be described as a balancing between finding similar elements to the query, and also diverse elements in the result set. Mathematically, suppose S as a set of n elements $\{s_1, \ldots, s_n\}$, q as a query element and an integer size k that is less or equal than n. Thus, we can say that the similarity of each element $s_i \in S$ can be determined by the similarity function $\delta_{sim}(q, s_i)$, where a higher value indicates that the element s_i is more similar to the query q. Otherwise, the diversity between two elements $s_i, s_j \in S$ can be defined by the function $\delta_{div}(s_i, s_j)$.

In other words, the Result Diversification Problem can be defined as follows: given a set S and a query q, we may find the a result set $R \subseteq S$ of size $|R| = k$, where each element of R is similar to q regarding the similarity function δ_{sim} and also simultanenously diverse among other elements in R with respect to the diversity function δ_{div}. In this model, we can represent elements in S using the vector space model. Hence, the similarity and diversity functions can be calculated through a distance function, for example, the Euclidean distance.

Putting together these two functions, the Result Diversification Problem can be formally stated as follows: given a tradeoff λ, $0 \leq \lambda \leq 1$, between similarity and diversity, the k-similar diversification set R contains k elements in S, such that maximizes the objective function $\mathcal{F}$, as shown in Eq. 3:

$$\mathcal{F}(q, S') = (k-1)(1-\lambda) \cdot sim(q, S') + 2\lambda \cdot div(S') \quad (3)$$

Thus, on one hand, the similarity component of the objective function $\mathcal{F}$ measures the amount of "attractive forces" between q and k elements in S', where subset $S' \subseteq S$. Thereby, the function $sim(q, S')$ is the sum of similarity distances among the query element center and all elements of S', calculated by Eq. 4:

$$sim(q, S') = \sum_{i=1}^{k} \delta_{sim}(q, s_i), s_i \in S' \quad (4)$$

On the other hand, the diversity component of the objective function $\mathcal{F}$ measures the amount of "repulsive forces" among k elements in S', where subset $S' \subseteq S$. Hence, the function $div(S')$ is the sum of distances among all elements in S', calculated by Eq. 5:

$$div(S') = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \delta_{div}(s_i, s_j), s_i, s_j \in S' \quad (5)$$

Moreover, from this definition, we can emphasize two special cases for the parameter λ. Firstly, when $\lambda = 0$, the result set R is simplified to find the "k-similar set" and relies only on the query q. Secondly, when $\lambda = 1$, the result set R is reduced to find the "k-diverse set".

The query result diversification approaches to be compared with our proposed method are the Swap and the MMR and they work without extra information. These approaches are based on the max-sum dispersion problem (KUBY, 1988), which consists of maximizing the sum of distances among result set elements.

Both approaches have the similarity and diversity competing with each other, taking a parameter λ as input for one user control of preference. This parameter allows including diversity among elements based on a tradeoff objective function $\mathcal{F}$, thus, re-ranking the result of basic similarity algorithms, for example, a range query. Generally, these approaches take the initial result set returned by a similarity-based algorithm, which is called candidate set S, whose elements are as similar as possible to the query center element. Hereafter, a subset $R \subseteq S$ of size k is selected taking into account the objective function $\mathcal{F}$. In spite of these

approaches adopting the max-sum problem, other measures could be used, for instance, max-min, min-max and max-avg (CARBONELL; GOLDSTEIN, 1998; COYLE; SMYTH, 2004; GOLLAPUDI, 2009; YU, C.; LAKSHMANAN, L.; AMER-YAHIA, S., 2009).

Swap and MMR differs in function of strategy with respect to the generation of the result set. So, we can classify these two methods based on their traits that are exchanging and incremental, respectively. The incremental strategy starts the result set R empty and iteratively inserts into it elements selected from S that maximizes the objective function. The exchanging strategy, in its turn, selects an initial result set R and, then, the remaining elements in S are evaluated as candidates to replace an element from the current solution R (SANTOS; OLIVEIRA; FERREIRA; TRAINA *et al.*, 2013)

3.3. Swap

The Swap (YU, C.; LAKSHMANAN, L.; AMER-YAHIA, S., 2009) method is the simplest method to construct the result set R and as it was aforementioned is based on the exchanging strategy. This method is composed of two phases. Firstly, an initial result set R is created containing the top-k similar elements of S. After this, each remaining element in S, ordered by decreasing similarity values δ_{sim}, is tested to replace an element of the current solution R. Case the tested element improves $\mathcal{F}$, and then the replace operation that improves $\mathcal{F}$ is definitely applied to R. This process is performed until every element in the candidate set S is tested with respect to their similarity function δ_{sim}.

Nonetheless, this method has a drawback regarding the final result set. Since the candidate set S is tested with respect to their similarity function δ_{sim} order and does not consider diversity function values δ_{div} order, this may result in solutions that do not maximize $\mathcal{F}$ and, consequently, the result set may not be optimal.

Figure 4 presents the Swap algorithm explained in Vieira et al. (2011).

Figure 4 - Swap algorithm.

Input: candidate set S and result set size k
Output: result set $R \subseteq S, |R| = k$
1: $R \leftarrow \emptyset$
2: **while** $|R| < k$ **do**
3: $\quad s_s \leftarrow \text{argmax}_{s_i \in S}(\delta_{sim}(q, s_i))$
4: $\quad S \leftarrow S \setminus s_s$
5: $\quad R \leftarrow R \cup s_s$
6: **while** $|S| > 0$ **do**
7: $\quad s_s \leftarrow \text{argmax}_{s_i \in S}(\delta_{sim}(q, s_i))$
8: $\quad S \leftarrow S \setminus s_s$
9: $\quad R' \leftarrow R$
10: $\quad$ **for each** $s_j \in R$ **do**
11: $\quad\quad$ **if** $\mathcal{F}(q, \{R \setminus s_j\} \cup s_s) > \mathcal{F}(q, R')$ **then**
12: $\quad\quad\quad R' \leftarrow \{R \setminus s_j\} \cup s_s$
13: $\quad$ **if** $\mathcal{F}(q, R') > \mathcal{F}(q, R)$ **then**
14: $\quad\quad R \leftarrow R'$

Source: Vieira et al. (2011).

3.4. MMR

The Maximal Marginal Relevance (CARBONELL; GOLDSTEIN, 1998) is based on the incremental strategy, hence, it iteratively constructs the result set R by selecting the new element in S that maximizes the objective function $MMR(s_i)$, as described in Eq. 6:

$$MMR(s_i) = (1-\lambda)\delta_{sim}(s_i, s_q) + \frac{\lambda}{|R|}\sum_{s_j \in R}\delta_{div}(s_i, s_j) \quad (6)$$

MMR has also some drawbacks with respect to the result set R. Firstly, R always starts with the element with the highest δ_{sim} value in S, regardless of the λ value. Thereafter, the first selected element has a large influence in the quality of the final result set R, this because R is incrementally constructed by inserting a new element to previous results. Hence, according to the first selected element the final result set may have low quality in terms of $\mathcal{F}$.

Figure 5 presents the MMR algorithm explained in Vieira et al. (2011).

Figure 5 - MMR algorithm.

Input: candidate set S and result set size k
Output: result set $R \subseteq S, |R| = k$
1: $R \leftarrow \emptyset$
2: $s_s \leftarrow \text{argmax}_{s_i \in S}(mmr(s_i))$
3: $S \leftarrow S \setminus s_s$
4: $R \leftarrow s_s$
5: **while** $|R| < k$ **do**
6: $\quad s_s \leftarrow \text{argmax}_{s_i \in S}(mmr(s_i))$
7: $\quad S \leftarrow S \setminus s_s$
8: $\quad R \leftarrow R \cup s_s$

Source: Vieira et al. (2011).

3.5. Final Considerations

This chapter presented the fundamental concepts related to Result Diversification Problem. Here, we could see an overview of how diversity is included in a query result and how it can be controlled balancing the level of similarity and diversity.

Presented theoretical concepts regarding the max-sum dispersion problem, which is used by two well-known algorithms of the literature, Swap and MMR, and that work without extra information.

Furthermore, this chapter presented the Swap and MMR algorithms, their main characteristics and drawbacks and how they formally work regarding their parameters and the controlling with respect to the balancing of similarity and diversity.

Chapter 4 - DATA SAMPLING AND SUMMARIZATION

4.1. Initial Considerations

In this chapter, we present theories and methods related to the sampling and summarization process of datasets. These terms can confuse or even be overlapped depending on the context where they are applied.

The term sampling is quite generic and can be related to statistics, machine learning and for the reducing of large databases, this last one is within the concept of data summarization.

Briefly, sampling in statistics is a procedure that concerns with the gathering of a number of observations from a larger population (COCHRAN, 2007). This concept is also studied in the field of data summarization, which aims at reducing a large database selecting a representative subset of it. While in machine learning, sampling can be used to deal with the problem of imbalanced datasets.

Thereby, in this work, we firstly present preliminaries with respect to sampling for imbalanced dataset, what is the imbalance problem in machine learning, and the main methods that handle this issue.

After that, we discuss data summarization of large datasets, what are the challenges, issues and methods involved when scale a dataset down is desired. Thus, we present important concepts approached by the literature and the main methods proposed by the scientific community.

4.2. Sampling for Imbalanced Datasets

Imbalanced datasets are a special situation for classification problems where data points (instances) are not equally distributed among classes. The classes can be categorized into two defined groups - majority class (positive) that has most data instances and minority class (negative), which has the smallest number of data instances. The main challenge in imbalanced datasets is that minority classes are often very useful, however, traditional classifiers algorithms tend to ignore very small classes and have a bias towards the majority class. Several datasets in real applications comprehend imbalanced class distribution problem (MANI; ZHANG, 2003)

Sampling is a set of methods that deal with the imbalance problem by adding or removing instances from datasets. The process of removing data instances from the majority class is called Undersampling, while adding data instances to the minority class is called Oversampling. In both categories of methods, the objective is to reduce the level of imbalance to more balanced training set so that classifier algorithms can improve their results (SINGH; PUROHIT, 2015).

Besides that, the literature has differentiated data imbalance problem in two main categories: Binary class data imbalance and Multi class data imbalance (ORRIOLS-PUIG; BERNADÓ-MANSILLA, 2009). Binary class data imbalance consists of datasets that contains two classes, in which one class is represented by only a few numbers of instances. Multi-class data imbalance comprehends dataset that contains more than two classes. This type of dataset requires more complex methods and sometimes this is even divided into many binary class problems.

In this work, we focus on the undersampling category, more specifically summarization, which is the one that fits our proposed method. In this section we describe four known algorithms of undersampling to balance the class distribution on training data, and in the next section we describe summarization algorithms.

4.2.1. Random Undersampling

Is one of the simplest non-heuristic methods to perform undersampling, which aims to balance class distribution through the random removing of instances of the majority class. This method has a drawback with respect to the possibility of eliminating useful data that could be important for the classifier induction process (BATISTA; PRATI; MONARD, 2004).

4.2.2. Tomek Links

Tomek Links (TOMEK, 1976) is a heuristic method that can be used as an undersampling method or also as a data cleaning method. For the process of undersampling, it removes only instances that belong to the majority class.

This method detects the so-called Tomek's links, which exist if two instances are the nearest neighbor of each other. Thus, a link between two instances S_1 and S_2 of different is defined such data for any instance S_3:

$$d(S_1, S_2) < d(S_1, S_3) \; and \; d(S_1, S_2) < d(S_2, S_3) \tag{7}$$

where $d(.,.)$ is the distance between the two instances. Thus, if two examples form a Tomek link then either one of these instances is noise or both instances are borderline.

4.2.3. Condensed Nearest Neighbor Rule (CNN)

CNN (HART, 1968) selects a subset of instances that are able to correctly classify the original dataset using a one-nearest neighbor rule. Formally, a subset $S_1 \subseteq S$ is consistent with S if using a 1-nearest neighbor, S_1 correctly classifies the examples in S. The algorithm can be described as follows: the method starts with two blank datasets A and B. Initially, the first instance is allocated in dataset A, while the remaining of the instances are allocated in dataset B. Hence, the method goes through the set B, instance by instance, and classifies each instance using a 1-nearest neighbor rule. If an instance in B is misclassified, it is transferred from B to A. This process is performed until no instance is transferred from B to A.

4.2.4. One-sided Selection (OSS)

OSS (KUBAT; MATWIN, 1997) is an undersampling method resulting from the application of Tomek Links followed by the application of Condensed Nearest Neighbor (CNN). Firstly, Tomek links are applied in order to remove noisy and borderline majority class instances. Thereafter, CNN is applied to remove instances from the majority class that are distant from the decision border.

4.3. Sampling for Summarization

Summarization is a term in data analysis that describes the process of obtaining a concise and informative "portion" or interpretation out of a large database. Its definition or utility relies on the purpose of its utilization or domain, for instance, the usage for text analysis is different from the usage for network traffic monitoring.

The process of creating semantic content of the data aims at making the data intelligible for further data analysis. In this context, Summarization methods have demonstrated to be a useful and effective mechanism in the process of interpreting large amount of data. For example, in network traffic monitoring, the huge amount of data makes difficult to apply anomaly detection techniques due to computational cost (CHANDOLA; BANERJEE; KUMAR, 2009). Thus, a good summary of network traffic helps in the process

of getting insights from a large volume of network traffic and getting only the important parts of the data for the analysis, making it easier and faster for the human analysis and also with less computational time for anomaly detection techniques (AHMED; MAHMOOD; MAHER, 2014; MAHMOOD, 2008).

The paper of Ahmed (AHMED, 2019a) contributes with a comprehensive overview of data summarization methods. It provides a taxonomy for summarization approaches categorizing them into two major groups: structured and unstructured data.

4.3.1. Summarization of unstructured data

In the context of summarization, unstructured data refers to information that is typically in a text-heavy manner. In other words, the information doesn't have a pre-defined model, but it may contain dates and number. The summarization of this kind of data is known as text summarization and its process is composed of the sequence of steps (RADEV; HOVY; MCKEOWN, 2002), namely: extraction; abstraction; fusion; compression.

Since the objective of our proposed algorithm is not to deal with text summarization, we are not going to describe in detail the processes and algorithms of this kind of data. Nonetheless, for the sake of knowledge we can say briefly that different text summarization techniques have already been proposed (BAXENDALE, 1958; LUHN, 1958). Some of the most relevant techniques are based on: Naive-Bayes classifier (EDMUNDSON, 1969; KUPIEC; PEDERSEN; CHEN, 1999; LARSEN, 1999); decision tree (LIN, 1999; LIN; HOVY, 1997); hidden Markov model (CONROY; O'LEARY, 2001); artificial neural network (LIN, 2004; SVORE; VANDERWENDE; BURGES, 2007); natural language processing (BARZILAY; ELHADAD, 1999); similarity measure-based (MCKEOWN; KLANVAS; EVANS, 2005); and topic modeling (LEE; BELKASIM; ZHANG, 2013).

4.3.2. Summarization of structured data

Structured data consists of the information that has a minimal pre-defined model, for example, it is organized in a fixed field (rows and columns) within a matrix or file, and also includes data stored in spreadsheets or relational databases. For example, the network traffic data is typically registered in a structured data, which is composed of a number of rows and columns.

There are several methods proposed in the literature in order to summarize this kind of data (CAI, 1989; CHANDOLA; KUMAR, 2007; HAN; FU; HUANG; CAI *et al.*, 1994; HAN;

FU; WANG; CHIANG *et al.*, 1996; JAGADISH; MADAR; NG, 1999; POUZOLS; LOPEZ; BARROS, 2011; YAGER, 1982) and these methods can be categorized as: statistical, linguistic and machine learning. In this study, we focus on the main algorithms that are sampling-based or that use statistical functions such as L_1 and L_2 norms to represent the data.

4.3.2.1 *Balanced Iterative Reducing and Clustering (BIRCH)*

BIRCH (ZHANG; RAMAKRISHNAN; LIVNY, 1996) is a clustering algorithm that constructs a dynamic hierarchical tree structure in order to hold summary information. The tree hierarchically arranges the clusters that are at the leaf nodes; so this tree-based structure is called as Clustering Feature (CF-tree). After that, a clustering algorithm can be applied to the nodes of the tree for the resulting clusters.

Nevertheless, BIRCH has a drawback with respect to what clustering algorithm can be used, hierarchical algorithms, for example, cannot be used due to be based on distances between data objects, which are not suitable for compressed objects. Thus, BIRCH can only use partitioning algorithms, for instance *k-means* (MACQUEEN, 1967), when producing the clustering results.

4.3.2.2 *Modified k-means*

Clustering is a relevant mechanism in unsupervised learning that allows finding natural or intrinsic groups in data objects. Hence, this mechanism can also be used for data summarization.

Ha-Thuc et al. (HA-THUC; NGUYEN; SRINIVASAN, 2008) proposed a data summarization method that is based on a modified k-means algorithm. K-means is a centroid-based algorithm, thus, it clusters the dataset and in each cluster a centroid is defined. The modified algorithm version for summarization considers that the summary is the set clusters centroids.

A threshold is used to determine the number of clusters. The algorithm creates partitions of the dataset until the sum of squared error (SSE) is less than a given threshold.

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} d(c_i, x)^2 \quad (8)$$

Taking that into account, the algorithm begins by using k-means to find cluster centroids, once these are found the next steps are just executed if the SSE is greater than the given threshold, then, the existing cluster is split.

After that, a new centroid is introduced that is closer to the larger cluster centroid. This process is repeated until every cluster's SSE is smaller than the given threshold.

4.3.2.3 *Sampling methods*

Sampling is a relevant mechanism to deal with large databases. Shortly, a sample is a dataset's subset. Hence, sampling can be a good choice for database reduction considering that it has low cost and it has great efficiency. There are different ways of performing sampling. Cochran (COCHRAN, 2007) proposes some major categories of sampling:

- **Simple random sampling**

As the name implies, it is the simplest category of sampling and it consists of choosing samples at random, given the sample size, and where no data instance is included more than once in the result set.

- **Systematic sampling**

This kind of sampling, although random, it still uses a deterministic process of selecting data points, which is: a data instance is sampled from the dataset, beginning from a specific starting point until the end, considering equal intervals. The interval is calculated as rounded up $\left\lceil \frac{size\ of\ dataset}{size\ of\ sample} \right\rceil$.

For example: suppose a starting point is randomly selected in the 2rd position of the dataset and consider that the calculated interval was 3, then for a sample of size 4, the chosen instances for the sample are from the 2nd, 5th, 8th and 11th position of the dataset, respectively.

- **Stratified random sampling**

The dataset is divided in disjoint subsets, which are called *strata*. Then, on each *strata* is applied a simple random sampling in order to generate a stratified sample. In other

words, this method selects randomly a data point from each strata and creates a subset as a sample'.

- **Cluster random sampling**

This method splits the whole dataset in clusters. These clusters are randomly chosen considering a sampling rate, then, all data points of the selected clusters are chosen. For example: the method can divide the entire population (dataset) of a state into different cities (clusters). Then, the method selects a number of cities depending on the desired rate through a simple random sampling. Finally, from the selected cities (randomly selected) the method can select a number of subjects from each city also through simple random sampling.

4.4. Final Considerations

This chapter introduced the fundamental concepts of both sampling and summarization. Here, we explained their definitions, which have some overlapping characteristics and described the main methods encompassed in both processes.

Regarding the sampling of imbalanced datasets, we defined the problem of imbalanced classes and presented the main methods, which handle this situation, more specifically with respect to undersampling, such as: NearMiss; Tomek Links; Condensed Nearest Neighbor Rule; One Sided Selection; Edited Nearest Neighbor; and Neighborhood Cleaning Rule.

With respect to the summarization of databases, we introduced the concepts and in which cases the reducing of large databases is desired, thus, providing example and the possible solution. And, we also presented the main methods proposed by the literature to deal with this kind of issue, such as: Balanced Iterative Reducing and Clustering; a modified k-means; and the sampling methods.

Chapter 5 - TOURIST WALK

5.1. Initial Considerations

The tourist walk (TW) is a deterministic approach based on the tracking of data points distributed throughout a metric space. Lima et al. (LIMA; MARTINEZ; KINOUCHI, 2001) defined the approach as a tourist (walker) whishing to visit N cities (data points) distributed randomly on a map of d dimensions. Thereunto, the tourist starts its walking in a given city of this map and moves according to a deterministic rule, namely: "go to the nearest city, which has not been visited in the last μ time steps". From the walking, we can extract some behavior properties according to the μ parameter value.

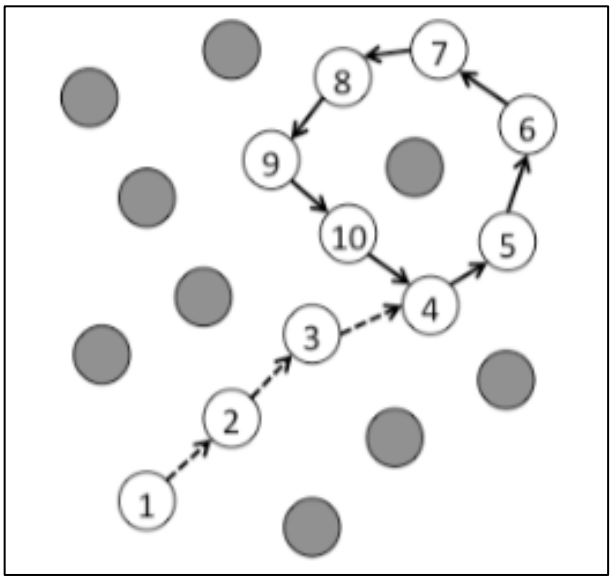
Firstly, one important property is when $\mu \geq 1$, which means that the action of self-avoiding is limited to the memory window $\tau = \mu - 1$. This represents a characteristic time step for the city to become attractive to the tourist again, also known as *refraction time*. From the tourist resulting trajectory, we can decompose it in two relevant parts: the transient part of size t (new cities are visited) and a final cycle period p (new cities are not visited anymore). The tourist steps can be performed considering a ranking of neighbors, which can be represented by means of a neighborhood table whose content consists of distances between cities (CAMPITELI; BATISTA; KINOUCHI; MARTINEZ, 2006).

Moreover, we can also consider other special properties when the memory μ expresses some specific values, such as:

- When we set the tourist memory $\mu = 0$ (the tourist is memoryless), so we deal with a trivial case, because the tourist has a memory of size null, thus, the tourist remains in the same city and each reference point represents a unique attractor. Hence, the resulting tourist trajectory has a transient t of size zero and a cycle period $p = 1$.
- When we set the tourist memory $\mu = 1$ (the tourist just remember the last visited city, that is his current city), so the tourist goes to his adjacent city until two mutually nearest neighbors are found, getting into a cycle period of two.

For the sake of clarity, we can exemplify a tourist walk considering the memory $\mu = 1$ and a table of distances between objects in a dataset. Thus, the trajectory results in a transient of size $t = 3$ and a cycle period p$= 7$ (Figure 6).

Figure 6 - Example of a tourist walk with memory size $\boldsymbol{\mu = 1}$. The arrows represent the walker's trajectory. The dashed arrows represent the transient (t=3) and the continuous arrows represent the cycle period (p=7)



Source: Made by the author.

5.2. Tourist Walk on Pattern Recognition

Regarding the tourist walk memory, Campiteli *et al.* (CAMPITELI; BATISTA; KINOUCHI; MARTINEZ, 2006) explored the complex behavior of the parameter μ value variation, focusing on the intermediate cases, when $1 < \mu < N - 1$. Taking into account the case $\mu = N - 1$, the tourist's trajectory tends to be totally auto-repulsive and the whole set with N data points comprehends an attractor. This particular case is known as the nearest neighbor construction heuristic.

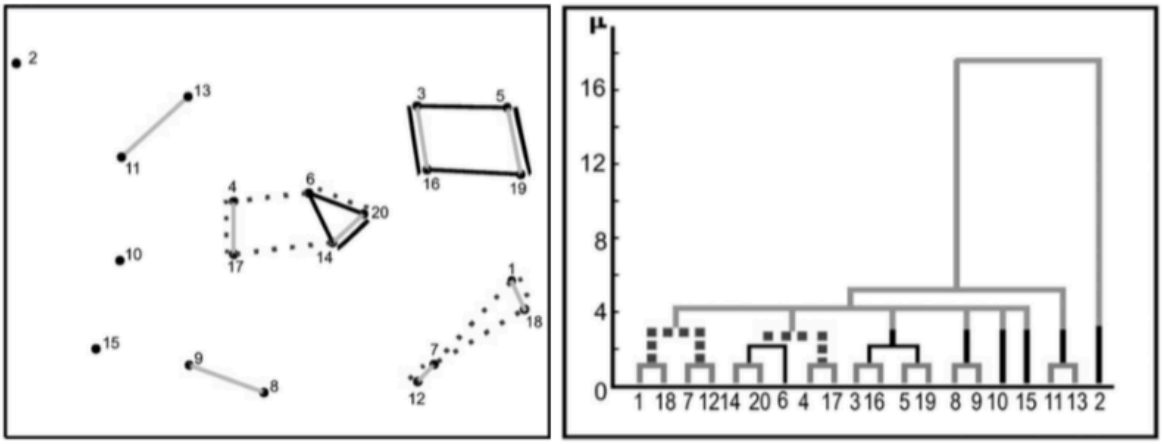
Thereby, the authors considered each attractor for a specific value of the parameter μ as a cluster. Thus, incrementing the value of μ leads to an increase of the auto-repulsive reaction of the tourist and the groups tend to merge. So, they have noticed that groups have at least $\mu + 1$ elements diverging from the pairing iteration of traditional hierarchy models.

Thus, we can represent the clustering process employing a tree, analogous to that used in hierarchy methods. The hierarchy levels represent the memory window rather than the similarity measure. For the sake of clarity, we can illustrate this algorithm with the following example: given a bidimensional map with 20 data points randomly distributed, we

can build a tree considering that data points belonging to the same attractor are seen as a group in each hierarchy level.

Figure 7 (a) shows a bidimensional map of 20 points randomly distributed and Figure 7 (b) shows the respective hierarchy tree.

Figure 7 - Constructing of a hierarchy tree through the tourist walk approach considering twenty data points randomly distributed on a bidimensional metric space.



(a) (b)

Source: (CAMPITELI; BATISTA; KINOUCHI; MARTINEZ, 2006)

We can notice in Figure 7 (a) twenty data points randomly distributed on a metric space, where lines represent the attractors formed by each value of memory considering the interval $\mu = [0, 3]$. For $\mu = 0$ each data point represents an attractor, thus, we have N clusters of single data points as the tree leaves. For $\mu = 1$, pairs of mutually nearest neighbours are the new attractors, represented by the clearer solid line. As the memory value increases, walks are longer and new attractors are obtained. Therefore, for $\mu = 2$ fresh attractors are formed, which are represented by the darker solid line. For $\mu = 3$ other fresh attractors are formed, these represented by the dotted line. In Figure 7 (b), we can see the matching tree with its respective forms and intensity colors of the drawn attractors.

The attractors are formed independently of the obtained results in the previous steps. Each fresh attractor can either contain a part of the previous attractors or a whole of them. Lastly, if overlaps occur, the clusters are combined, and this nesting process continues until all the set's data are contained in a single cluster.

The tourist walk capacity for automatically finding clusters that share statistics properties in heterogeneous dataset is of great value in the context of pattern recognition. The aggregated tree obtained by the method represents the nested structure of the data in an invariant form.

5.3. Tourist Walk on Images

Besides the aforementioned fields, the tourist walk was also applied to the context of image analysis. Backes *et al.* (BACKES; GONCALVES; MARTINEZ; BRUNO, 2010) present a study of deterministic walks that proposes a method of feature extraction from images. This approach allows exploring images in different scales, which is based on independent walkers starting from each pixel of an image.

The method considers a digital image of size $M_x \times M_y$ e $N = M_x \times M_y$, where each pixel (x, y) is associated with a gray level ranging from 0 to 255. Two pixels, (x_i, y_i) and (x_j, y_j) are considered neighbors when the geometric distance between them is less then a certain value, for instance, $d(i,j) < 2$, where $d(i,j)$ represents the euclidean distance:

$$d(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{9}$$

Since two pixels are considered geometric neighbors, their intensities difference modulo is defined as a "real distance" between them (BACKES; BRUNO; CAMPITELI; MARTINEZ, 2006). In digital images, the tourist walk attractors consist of a group of pixels, which compose a pathway where the tourist cannot scape. However, there are cases in which, depending on the pixels arranging used along the memory window, the tourist cannot find an attractor. In this case, the tourist walks until find a transient whose size is equal to the number of image pixels ($t = N$) and the resulting trajectory is considered as just containing the transient, with no cycle period ($p = 0$).

Another case that can be found on images is the existence of two or more direction options that are in agreement with the tourist walk rule. In this case, this problem can be solved through the selection of the first direction, amongst the tied options, when the neighbors are visited in the clockwise direction considering the neighbors matrix.

For each starting condition, the tourist can generate a different trajectory. We can observe, however, that different conditions can lead to the same attractor. Hence, taking into account all image's pixels as starting points, we can calculate the joint probability distribution of transient t for all cycle period p. Thereby, through the study of these distributions using statistical techniques, we can find a signature capable of discriminating the texture of an image (CAMPITELI; MARTINEZ; BRUNO, 2006).

5.4. Final Considerations

This chapter presented fundamental theories regarding deterministic tourist walk and also their capabilities in finding structures in different data domains. We presented different researches using this approach in order to detect patterns and to the classifications of images. Thus, from this, we have motivated the main part of our proposed approaches, which are based on the idea of the tourist walk that consists of tracking through data points and mapping their structures through the data distribution on a metric space.

Chapter 6 - METHOD 1: SIMILARITY WALK

6.1. Initial Considerations

Traditional methods of similarity query such as k-nearest neighbor (k-NNq) and range query (Rq) return a set of elements with the shortest distance from a query element (q), the first considering a predefined number of elements and the other considering a radius when composing the result set.

Similarity queries are widely used in information retrieval systems to retrieve objects based on their features. The features of a query object are compared with the features of objects stored in a database.

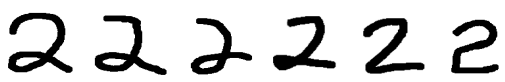
However, there is an issue in those systems, more specifically for content-based image retrieval (CBIR), regarding user satisfaction with the query result set. This intrinsic difficulty, known as the *semantic gap*, consists of the disparity between the result returned by the system and the result expected by the user (TRAINA; TRAINA JR; CIFERRI; RIBEIRO *et al.*, 2009). In this context, there are various studies toward approximating the results of computational analysis to human analysis. Most of them aims at optimizing or proposing features extractors (BALAN; TRAINA; RIBEIRO; MARQUES *et al.*, 2012; FELIPE; TRAINA; TRAINA JR, 2006; KUMAR; ONG; RANGANATH; ONG *et al.*, 2006), or evaluating and modifying distance functions (FELIPE; TRAINA; TRAINA, 2009; RUBNER; TOMASI, 2013; SANTINI; JAIN, 1999), or developing relevance feedback techniques (BUGATTI; TRAINA; TRAINA, 2011).

Notwithstanding, methods used in retrieval systems remain restricted to the direct assessment of distances between objects from the database to the query object. These methods do not consider the possibility of performing complex analyses that includes the interrelationship among elements of the database, as well as the employment of more dynamic comparison, which can take as a reference all the objects that are being retrieved on the query, instead of only the initial query one.

Hereupon, the study of tourist walk (TW) has drawn the attention of the scientific community regarding the capability of this technique to be employed in different types of applications, for example, image analysis, classification, clustering, and pattern recognition. Nevertheless, TW has not been explored to perform similarity retrieval, as an alternative to both Rq and k-NNq, allowing obtaining a result set of similar objects through the interconnection between elements that are being select throughout the query process.

Thereby, the hypothesis of the present study is that TW can provide more semantic results in some contexts of similarity retrieval. For instance, suppose a query where the user wants to retrieve images similar to an image displaying the handwritten number "2". In Figure 8, we can see a set of images of handwritten digits of the number two "2". Here, the query element is the first number "2", the second one is the most similar to the first element, and the third is the most similar to the second, and thereafter, until the last one is achieved. This last "2", despite of being also a "2", is not very similar to the first one. Thus k-NNq and Rq might not retrieve the last element, but TW can get it.

Figure 8 - Example of a set of handwritten numbers.



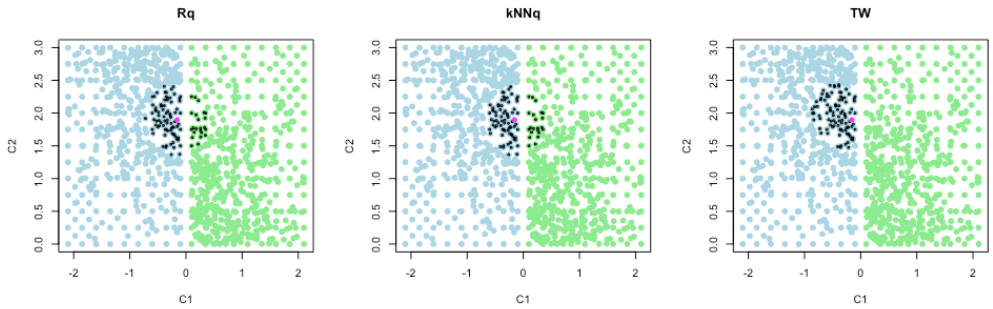
Source: Made by the author.

Another especial case, where TW can prevail over the traditional similarity queries is when the query element is located in the boundary of the class it belongs to. As an illustration, Figure 9 shows the Rq and k-NNq retrieving a bunch of elements from a class different from the query element's class, but TW, in contrast, retrieves mostly elements that belong to the same class of the query element.

As far as goes our research, there is no study in the literature that takes into account the usage of TW combined with distance functions in similarity retrieval environments, regarding its efficacy and efficiency, focusing on minimizing the semantic gap. Therefore, in the present study, we proposed a novel method based on the TW heuristic in order to perform content-based information retrieval.

Hence, we developed and investigated a method that is capable of retrieving elements considering their similarities, and the next object that will be part of the result set is the one that has the smallest sum of distances to all objects that are already in the result set. We have applied it to a group of artificial and real databases in order to evaluate its applicability.

Figure 9 - Examples of result sets of similarity queries having the query element located at the boundary of its class: range query (Rq), k-nearest neighbor query (k-NNq) and tourist walk (TW).



Source: Research data.

6.2. Literature Review

The similarity method proposed in this work is based on theoretical work that studies walks performed on data metric spaces and also based on similarity queries that can retrieve data or image objects that are similar to each other.

Shao et al. (SHAO; CUI; CHEN; LIU *et al.*, 2015) propose a two-stage random-walk sampling framework, called TSF, for the problem of top-k search. This problem consists of finding k vertices with the highest SimRank scores considering a given vertex v in a graph G. And SimRank (JEH; WIDOM, 2002) measures vertex-pair similarity according to the structure of graphs. Thus, the TSF maintains a set of random walks for each vertex, and an indexed tree approximately represents each walk. TSF estimates SimRank through these indexed walks in order to avoid redundant sampling. Furthermore, when the original graph is modified the indexed random walks can also be updated.

The study presented in (POLA; POLA; ELER, 2018) proposed a new technique to resolve the similarity search efficiently. This allows reducing distance calculations in similarity joins in order to achieve better performance. The study explores upper and lower bound properties of a metric to increase filtering of pairs of elements in similarity joins, this is done by using different positions of pivots, which are used to avoid unnecessary distance calculations considering metric property.

Another similarity measure method is presented in (LI; LV; HUANG, 2015), in which a proposed approach is based on probabilistic latent semantic analysis (PLSA) and Fisher kernel. This proposed method aims to exploit semantic information through topic inferring and label information.

Another method that assists in objects retrieval from image database is proposed in (LU; PENG; ZHU; WANG, 2013). In this study, the author suggests a concept of

comprehensive relevance (CR) that comprises high-level relevance measures integrated with low-level feature similarity, which is obtained using a Markov random field (MRF). Thus, the similarity between image pair is obtained through both low-level visual features and also the relations through other images in the database.

A recent study regarding image retrieval using similarity presented in (RANJAN; GUPTA; VENKATESH, 2019) proposes algorithms to retrieve based on maximum vote criterion and dictionary similarity measure. So, the authors propose a new measure called dictionary similarity measure that is used to find similarity between images. This measure is able to retrieve images with high computational efficiency and comparable accuracy with respect to other existing techniques. Other recent study that explores image retrieval based on similarity is proposed in (LIANG; SHI; WANG; MENG *et al.*, 2016), which proposes a similarity learning method that is able to maximize a top precision measure through parameter adjustment of the similarity function.

The first two studies are walk-based approaches but they focus on the process of ranking elements regarding their similarities. These walks are also different from our proposed approach because are random and not deterministic. The process of ranking elements gives no guarantee that element of others classes are not selected. The other studies use probability, Markov random field, dictionary, and learning measure similarity between data elements. But these studies don't mention the possibility of recovering data elements that have the same classification and how they can deal with it.

6.3. Proposed Approach

In this study, we propose a novel approach to perform similarity searches on databases, focusing on reducing the semantic gap. Hence, we have used the TW's trajectory, which is composed of visited data points distributed in a feature space and created two new different algorithms for the walk, modifying the rule of selecting the next elements to be visited.

We called our methods as SimWalk query (SWq) and numerated them as SWq1, SWq2 and SWq3. The difference between them resides in the rule for selecting the next element to compose the walk. Considering that the distances is not pre-computed, the time complexity of the methods SWq1, SWq2 and SWq3 in the worst case is O(*nk*), where *n* is the number of rows of the dataset and *k* is the size of the desired result set. Table 1 presents the rules for each one.

As SWq1 represents the original TW, its memory stores the last μ elements visited by the walker, who cannot revisit these elements when selecting the next element of the walk.

Thus, in SWq1 the memory is different from the walk, where the memory size is limited by a parameter k, while the walk size relies on the finding of an attractor.

Table 1: Description of the variations of the SimWalk method.

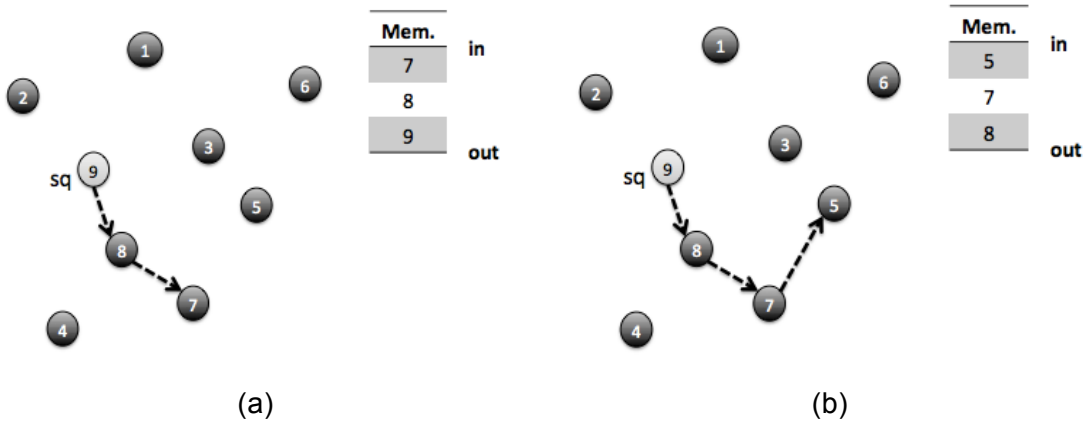
SWq1	Represents the original TW. The next element to be visited is the candidate that is closest to the last one in the walk.
SWq2	The next element to be visited is the candidate that has the minimum sum of distances for the first and for the last elements in the walk.
SWq3	The next element to be visited is the candidate that has the minimum sum of distances for all elements in the walk.

Source: Research data.

For the other two methods (SWq2 e SWq3), the walker's memory has the same size as the walk, because the walker cannot revisit any element. In other words, SWq2 and SWq3 do not rely on the finding of an attractor and their final walks are not composed of transient and/or cycle period because all selected elements will be part of the result set.

In order to illustrate how the SWq1 works, Figure 10 presents the schema for the selection of the next element in the SWq1 algorithm. In the example depicted in Figure 10 (a), elements nine (9), eight (8) and seven (7) are in the walker's memory. After that, to select the next element, the algorithm computes the distances from all candidates and chooses the one that has the shortest distance. In Figure 10 (b), element five (5) is selected as the next walk element. The SWq1 algorithm continues this process until a cycle period of p repeated elements is found.

Figure 10 - SWq1 schema.



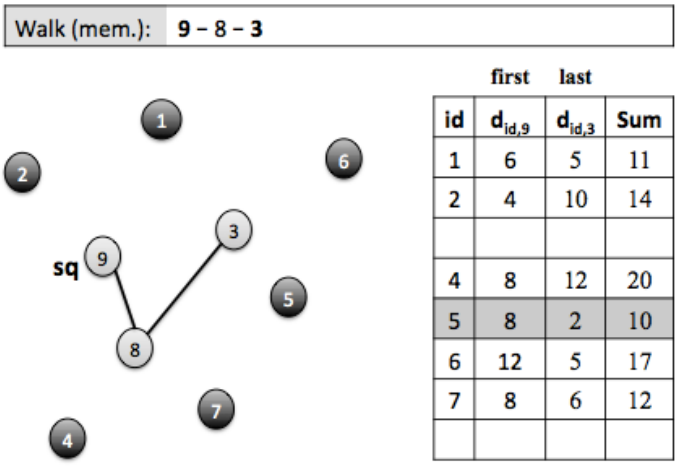
Source: Made by author.

In the same sense, Figure 11 presents the schema for the selection of the next element in the SWq2 algorithm. In the example depicted in Figure 11, elements nine (9),

eight (8) and three (3) are in the walk. In the SWq2 algorithm, the memory and the walk are the same thing, having the same size.

Thus, to select the next one, the algorithm calculates the distances from all the candidates to the first and the last elements in the walk, as shown in the table. In this example, element five (5) is the one that has the minimum sum of distances, thus it will be selected as the next walk element.

Figure 11 - SWq2 schema.



	first	last	
id	$d_{id,9}$	$d_{id,3}$	Sum
1	6	5	11
2	4	10	14
4	8	12	20
5	8	2	10
6	12	5	17
7	8	6	12

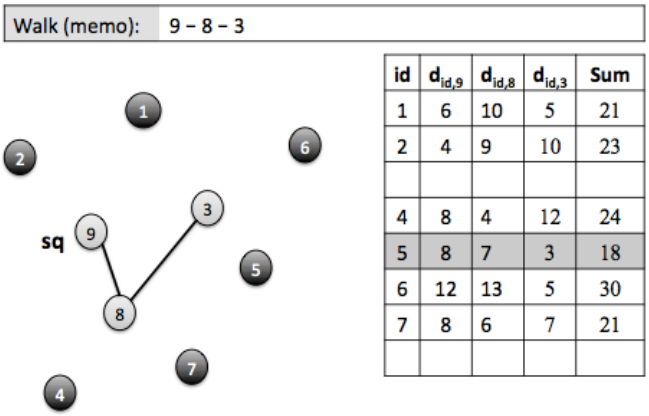
Source: Made by author.

Following the same context, Figure 12 shows the process for the selection of the next element in the SWq3 algorithm. In the example illustrated in Figure 12, elements nine (9), eight (8) and three (3) are in the walk (memory).

Hence, to pick up the next one, the algorithm computes the distances from all the candidates to the elements in the walk, as shown in the table. The element five (5) is the one that has the minimum sum of distances, so this will be the one selected as the next walk element. Coincidentally, in these above examples, which are very simple cases, element (5) is the one selected, but these results can change as the data set points distribution turns more complex.

In practice, when we observe real results, SWq1 provides a result set containing elements that follow a walk that goes far from the reference element. SWq2 generates an attraction force to the query element, so the result walk elements do not differentiate so much of it. SWq3, in its turn, maintains the result set element similar to each other.

Figure 12 - SWq3 schema.



id	$d_{id,9}$	$d_{id,8}$	$d_{id,3}$	Sum
1	6	10	5	21
2	4	9	10	23
4	8	4	12	24
5	8	7	3	18
6	12	13	5	30
7	8	6	7	21

Source: Made by author.

For the sake of clarity, Figure 13 shows the main function of the SWq2 and SWq3, which keeps the same logic in both algorithms. SWq1's main function is very different from this one because follows the rule of finding an attractor as the stopping criterion, which is part of the traditional TW rule. Besides, a traditional TW has other functions such as one to enqueue its memory and others to detect the attractor. In this study, we focus on describing SWq2 and SWq3, which are indeed our proposed algorithms.

Figure 13 - SimWalk algorithm main function

Algorithm 1

1: **procedure** SimWalk(S, k, q)
2: $W \leftarrow \{q\}$
3: $D \leftarrow$ *empty list of distance lists*
4: **while lengh**$(W) < k$
5: $s_n \leftarrow \text{findNext}(W)$
6: $W \leftarrow \{W, s_n\}$
7: **return**(W)

Source: Made by author

In Figure 13, we can notice that the function is responsible for construct a list W, which represents the walk. The parameters of SimWalk are the data set S, the result set k and the query element q. Firstly, the algorithm initializes the list W with the element q.

After this, a nested list of distance lists D is initialized as empty. S and D are global variables, so they keep their states in the other function. Inside the loop, the method searches for the next candidate element s_n and inserts it into W until its size reachs the value defined by k. In each loop step, the function *findNext* receives W as a parameter and returns the next candidate element. Depending on the SimWalk query setting, SWq2 or SWq3, the *findNext* function has a different implementation in order to define the next candidate

element. The Figure 14 and 15 present the findNext function of the SWq2 and the SWq3, respectively.

Firstly, in SWq2's findNext function, the algorithm begins storing the first and last element of the walk W in sf and s_l, respectively. The algorithm will select and store in a list L all distances from s_l to all elements of the dataset. After that, L is stored inside D, which is a list composed of distance lists. Each list has its position in D defined by the elements stored in W. Thus, the other positions in D store a null when don't have a list stored inside. Then, the algorithm creates a list Z of zeros in order to store the sums of distances, but it doesn't consider in the sum the elements stored in W.

After that, the algorithm checks if the first element (s_f) of W is equal to the last one (s_l), if it is equal then the algorithm replaces by *null* the elements defined by W in the list stored in the position $s_f$of D, and sum each element of D to each one of Z. Otherwise, if s_f is different from s_l then the algorithm gets the lists of the position s_f and s_l of D and replaces by $null$ in them the elements defined by W, and sum these both lists to each other and store the result of it in Z. Lastly, the algorithm selects the element in Z, which has the smallest sum of distances. Thereby, this element will be the one returned by the function.

Figure 14 - Function findNext of the SWq2

Algorithm 2

```
1:  procedure findNext(W)
3:      s_f ← W[0]
2:      s_l ← last(W)
3:      L ← empty list of distances from s_l to all
4:      if s_l ∉ {W\s_l} then
5:          for each s_j ∈ S do
6:              L ← {L, dist(s_l, s_j)}
7:          D[s_l] ← L
8:      Z ← list of sums initialized with zeros
9:      if s_f == s_l then
10:         Z ← Z + replace(D[s_f], W, null)
11:     else
12:         Z ← replace(D[s_f], W, null) + replace(D[s_l], W, null)
10:     s_i ← which.min(Z)
10:     return(s_i)
```

Source: Made by author

Secondly, SWq3's *findNext* function starts storing the last element of the walk W in s_l because the method will select and store in a list L all distances from s_l to all elements of the dataset. After that, L is stored inside D, which is a list composed of distance lists. Each list has its position in D defined by the elements that are stored in W, thus, the other positions in

D store a $null$ when don't have a list stored inside. Thereafter, the algorithm creates a list Z of zeros in order to store the sums of distances, but it doesn't consider in the sum the elements stored in W. Hence, the algorithm replaces in D the elements of W by null, and sums each element of D to each one of Z. Lastly, the algorithm selects the element in Z, which has the smallest sum of distances. Thereby, this element will be the one returned by the function.

Figure 15 - Function findNext of the SWq3

Algorithm 3

```
1: procedure findNext(W)
2:     s_l ← last(W)
3:     L ← empty list of distances from s_l to all
4:     if s_l ∉ {W\s_l} then
5:         for each s_j ∈ S do
6:             L ← {L, dist(s_l, s_j)}
7:         D[s_l] ← L
8:     Z ← list of sums filled with zeros
9:     for each w ∈ W
10:        Z ← Z + replace(D[w], W, null)
11:    s_i ← which.min(Z)
12:    return(s_i)
```

Source: Made by author

6.4. Materials and Methods

The proposed algorithms were implemented through the open-source software R[1], which provided us the language and the environment for statistical computing and graphics.

6.4.1. Datasets

In the experiments performed in order to evaluate the algorithms, we've used six elementary datasets proposed in (ULTSCH, 2003), which are grouped as the so-called Fundamental Clustering Problems Suite (FCPS)[2]. This offers a variety of datasets that explore clustering problems considering real world data situations. Twodiamonds dataset has as characteristic problem cluster borders defined by density. Lsun dataset has different variances and inter cluster distances. Engytime dataset has two classes defined by Gaussian mixture. Wingnut explores variations between density and distance. Tetra has four almost

[1] https://www.r-project.org
[2] https://www.uni-marburg.de/fb12/arbeitsgruppen/datenbionik/data

touching clusters. Another synthetic dataset we've used was the "two moons" that has intuitively separable clusters and well appropriated for classification problems.

A real world dataset that we've used was the Iris, which is widely used in the literature in studies involving pattern recognition techniques. This dataset originally has 3 classes ("setosa", "versicolor", and "virginica") 4 attributes ("sepal length", "sepal width", "petal length" and "petal width"), but we've used a version that has 2 attributes ("petal length" and "petal width") because it allows a better graphical visualization of the data distribution.

Another real world dataset we've used was one studied in (BALAN, 2007), which contains medical image features. For short, we called it MedImg dataset. This originally has 704 instances, 30 attributes and 8 classes ("angiogram MR", "axial pelvis', "axial head", "sagittal head", "coronal abdomen", "sagittal spine", "axial abdomen" and "coronal head"). But we've reduced the number of classes and attributes by selecting just 3 attributes (using a dimension reduction method called principal component analysis - PCA) and 4 classes ("axial pelvis", "angiogram MR", "sagittal head" and "coronal abdomen"), which consequently reduced the number of instances to 403. We've modified the MedImg dataset such that we could have a better visualization of the classes' data point distribution. Regarding the classes, we selected manually the ones that were better visually defined regarding their clusters so that we could easily choose as center query elements those located on the boundary between the classes. Table 2 describes the test datasets in terms of instances, attributes and classes.

Table 2: Description of the datasets.

Name	Instances	Attributes	Classes
Twodiamonds	800	2	2
Lsun	440	2	3
Engytime	409	2	2
Wingnut	1016	2	2
Tetra	400	3	4
Twomoons	449	2	2
Iris2d	150	2	3
MedImg	403	3	4

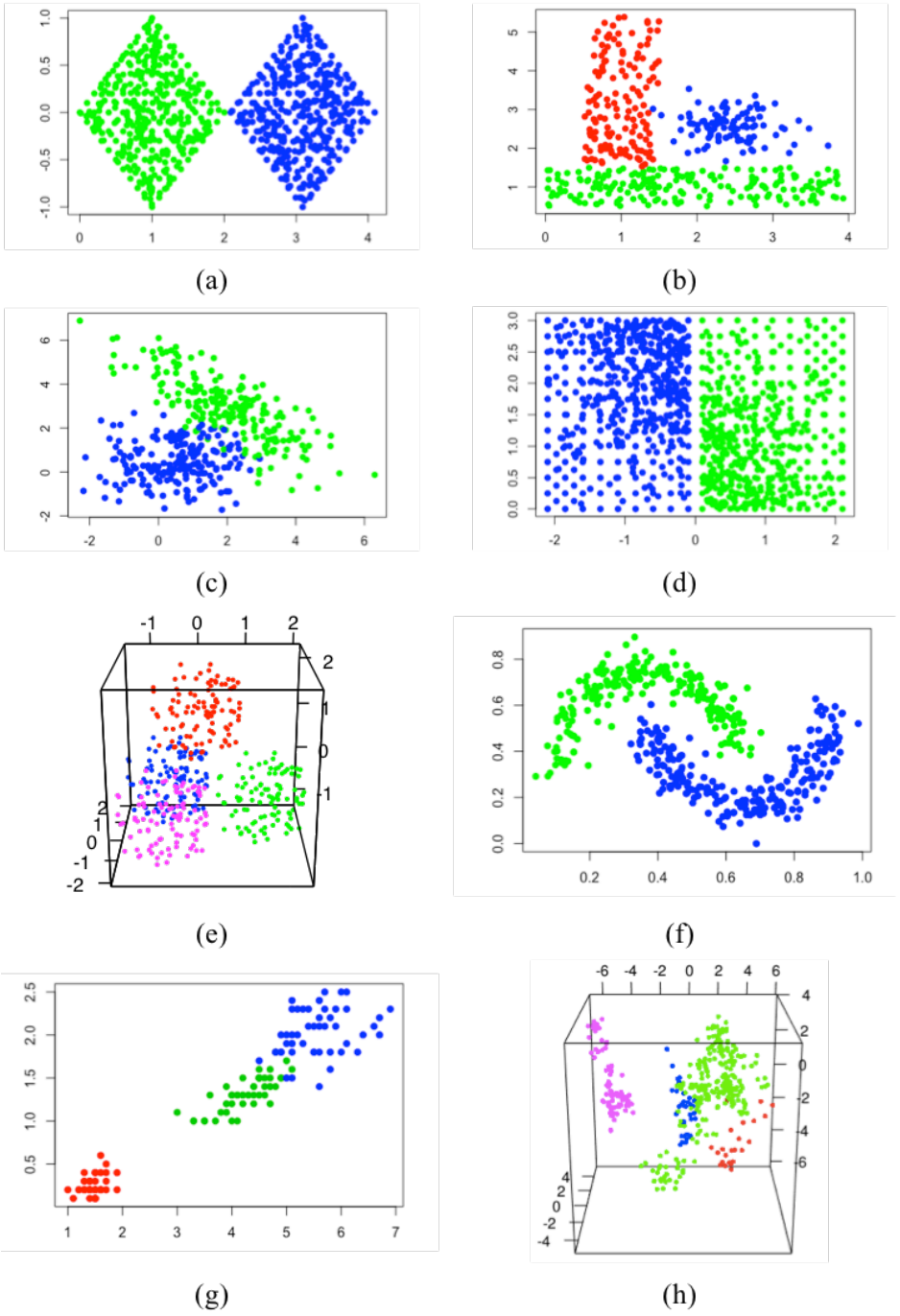
Source: Research data.

In Wingnut, Lsun and Twomoons datasets, we've applied some modifications in order to reduce the space that separates the classes, since a greater disjunction among classes tends to favor our methods over the traditional similarity searches. In Wingnut, we've reduced the distance between the two classes by 0.1. In Lsun, we've reduced the distances between the three classes by 0.5. In Twomoons, we've reduced (sampling randomly) from 14,977 to 449 (3% of the original) instances and decreased the distances between the two

classes by 0.1. We've also modified the Engytime dataset size (sampling randomly), from 4,096 to 409 (10% of the original) instances; due to the time processing that would delay our experiments.

Figure 16 shows the plotting of the datasets used in our experiments.

Figure 16 - Artificial and real datasets used in the experiments: (a) TwoDiamonds; (b) Lsun; (c) EngyTime; (d) Wingnut; (e) Tetra; (f) Twomoons; (g) Iris2d; (h) MedImg.



Source: Research data.

6.4.2. Similarity Search Evaluation

For the evaluation of the query algorithms performance, we have calculated the precision and recall of our proposed methods, considering the match of the class of each element from the dataset with the class of the query element.

This measure was adopted, because the match between the class of a result element and the class of the query one can be considered a measure of semantic accomplishment. Eq. 10 and 11 defines the precision and recall measures used in the evaluation process.

$$precision = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{retrieved\ documents\}|} \tag{10}$$

$$recall = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{relevant\ documents\}|} \tag{11}$$

Hence, we've evaluated the three SimWalk query approaches considering the datasets presented in this work. We also have compared our best SimWalk method with k-NNq. Thus, the results obtained by the methods were then evaluated through the precision x recall performance with respect to the result set size of each method. We've varied the input parameters of the searches so that they could return the same result set size. As initial objects queries, we've selected the ones situated at the frontiers of the dataset's classes, because it represents the critical situation to be considered. From Twodiamonds, we've selected 80 (out of 800) elements located closest to the border. From Lsun, we've selected 248 (out of 440) elements located closest to the classes' border. From Engytime, we've selected 310 (out of 409) elements closest to the border. From Wingnut, we've selected 172 (out of 1016) elements closest to the border. From Tetra, we've selected all elements. From Twomoons, we've selected 159 (out of 449) elements closest to the border. From Iris, we've selected 15 (out of 150) elements closest to the border between the classes "versicolor" and "virginica", which are overlapping classes. From MedImg, we've selected 40 (out of 403) elements closest to the border.

6.5. Experimental Results

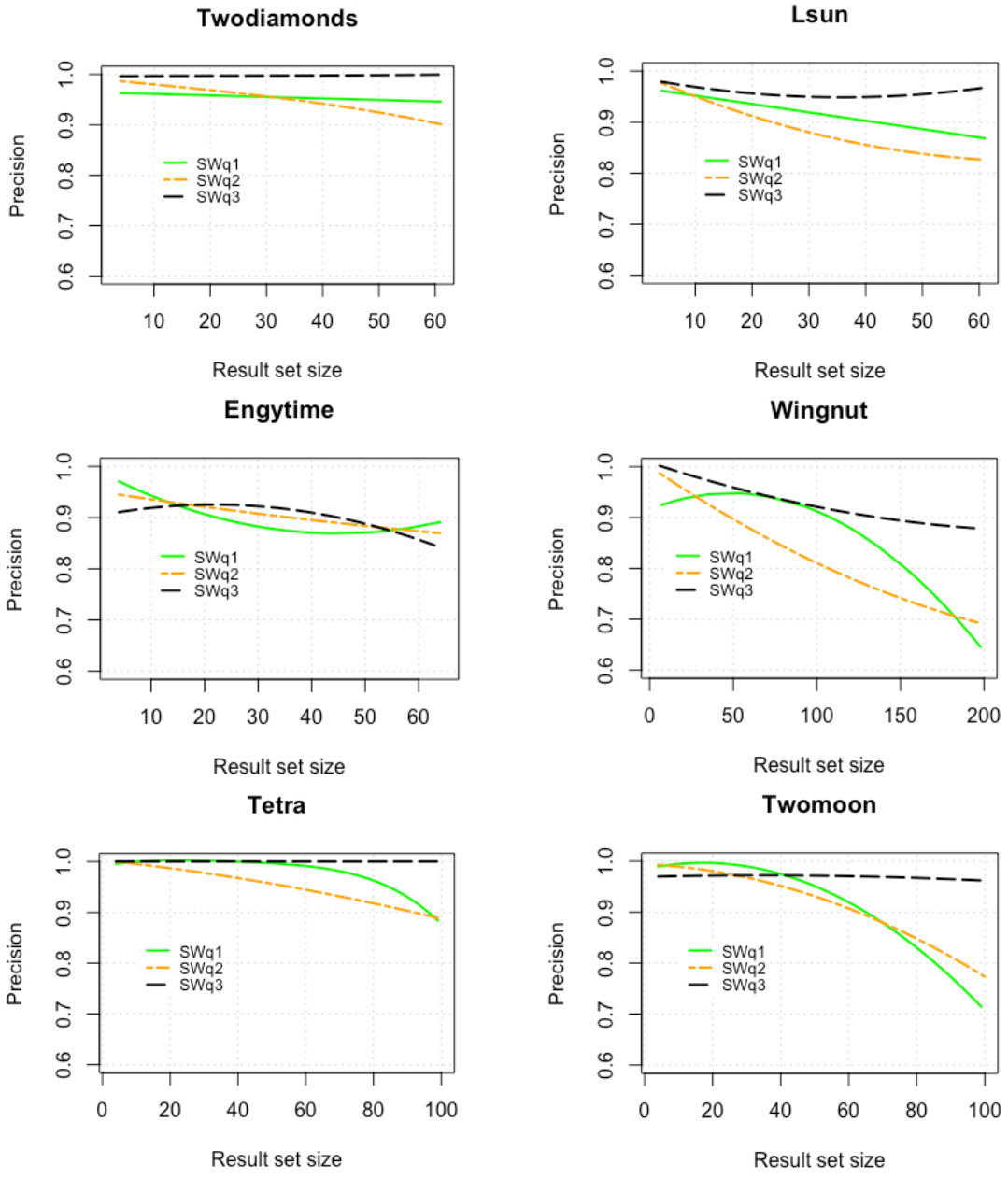
Firstly, in order to define which SimWalk configuration has the better performance we applied SWq1, SWq2 and SWq3 to artificial datasets and calculated the precision of them.

We have used the software R to calculate all precisions and to generate the respective charts.

Figure 17 presents the precision of each SWq variation considering the artificial datasets TwoDiamonds, Lsun, Engytime, Wingnut, Tetra and Twomoons.

As we can observe in Figure 17, SWq3 achieves better performances and SWq2 presents better performances than SWq1. Therefore, since SWq3 prevailed over the other proposed algorithms, we can say that SWq3 as the best among them. Thus, we chose it to compare with the traditional k-NNq.

Figure 17 - Evaluation of each SimWalk variation on the artificial datasets: TwoDiamonds; Lsun; Engytime; Wingnut; Tetra; Twomoons.



Source: Research data.

Thereby, we applied SWq3 algorithm and the traditional k-NNq to the artificial datasets and the real datasets Iris2d and MedImg. Figure 18 presents the results obtained by the queries and their respective precision x recall.

In Figure 18, we can notice that, according to the results presented, SimWalk query SWq3 presents better performance than the traditional k-NNq in almost all data sets, artificial and real world. Hence, we can assert that the result sets of SWq3 tends to stay inside the same class of the initial query element, different from k-NNq that returns the closest elements to the initial query element, but are not necessarily inside the same class.

According to Figure 18, in overall, for low values of recall, SWq3 presents results close to k-NNq. However, as the recall value increases SWq3 presents better performance results than k-NNq, since they tend to present a tendency to degrade their precision.

In Twodiamonds plot, SWq3, kNNq present a high precision when the recall is in low values. However, as the recall increases kNNq loses its precision significantly. On the other hand, SWq holds precision in high values. This is due to the dataset aspect, which has the borders defined by density and just two data points defining the boundary between the two classes.

In Lsun plot, SWq3 and kNNq also present a high precision when the recall is in low values, but kNNq still loses precision when the recall increases. In this case, Rq presents a decreasing of precision in intermediate values of recall, but it still maintains precisions higher than kNNq. This happens because the Lsun dataset has different variances in their clusters.

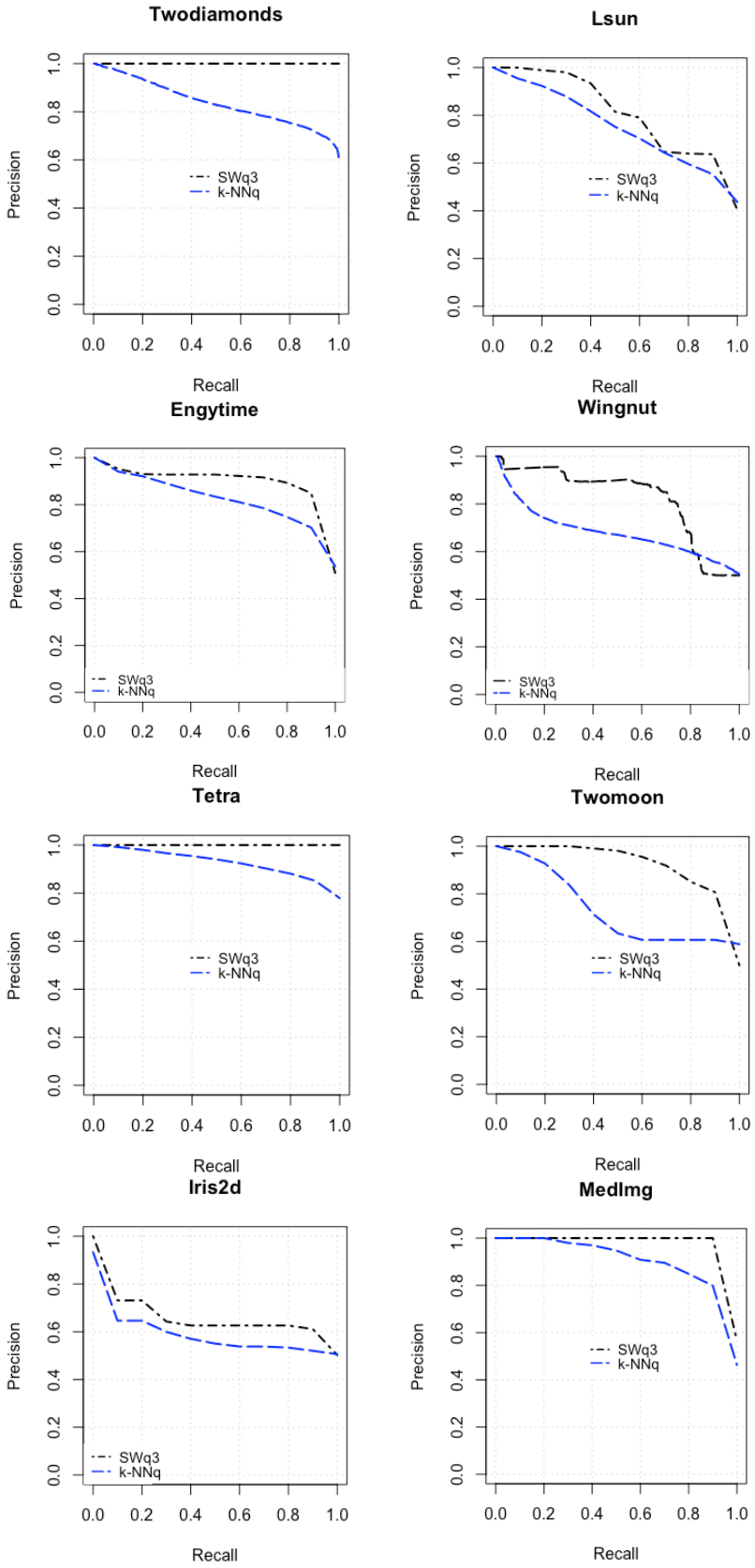
In Engytime plot, we can notice that SWq3 also keeps precisions higher than kNNq. This is because the Engytime dataset has its clusters defined by Gaussian mixture, where both classes have center points of high density and overlapping borders.

In Wingnut plot, we can observe a similar behavior to Twodiamonds and Lsun plots in terms of decreasing precision for both kNNq. On the other hand, SWq keeps its precision in high values. This is because Wingnut dataset has a certain distance between the classes and the border has reversed directions of densities.

In Tetra plot, all methods can keep high values of precision and don't present a distinguishable loss of it. However, kNNq slight decrease of precision when the recall increases. This is because the Tetra dataset has well defined and separable classes.

In Twomoon plot, we can notice again the same behavior as the Twodiamonds, Lsun and Wingnut, which is kNNq losing precision as the recall increases, while the SWq can keep its precision in high values. The Twomoon dataset has intuitively separable clusters.

Figure 18 - Evaluation of SWq3 algorithm, in comparison with the k-NNq.



Source: Research data.

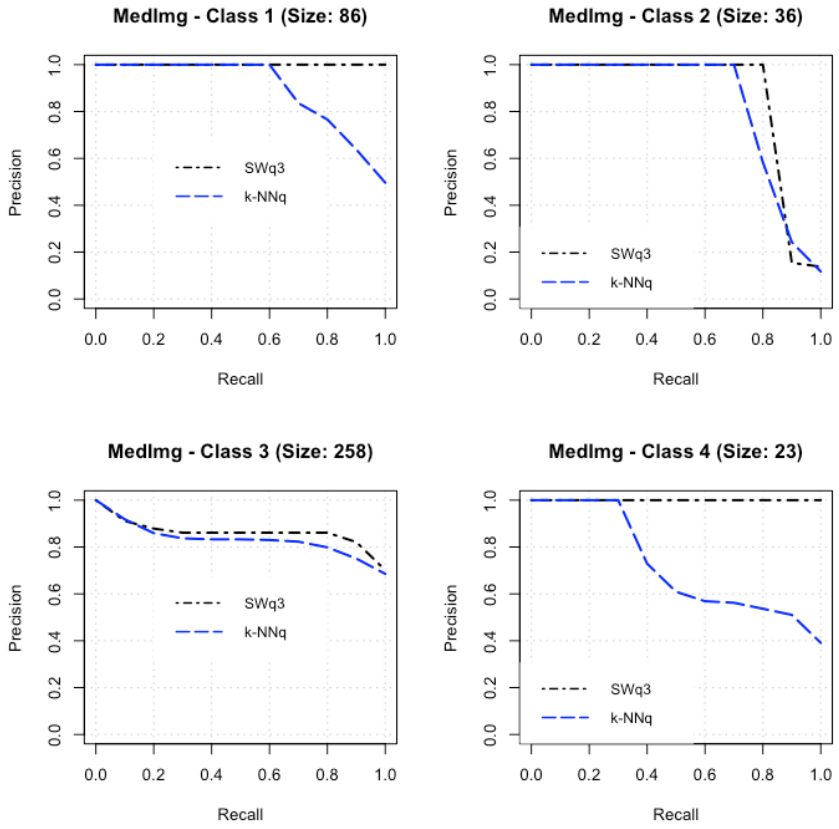
In Iris2d plot, we can see that the methods do not have such high precision values, but still SWq3 has higher values of precision than the kNNq. This is behavior is because Iris2d dataset has two classes well separated and two slightly overlapping classes.

In MedImg plot, we can also see similar behavior to Twodiamonds, Lsun, Wingnut and Twomoon. SWq3 again could keep high values of precision. This is because MedImg plot has two well-separated classes and two classes with some overlapping points.

In general, we can observe that SWq3 has high values of precision. Besides this, SWq3 has high values of precision when datasets have well separated classes even with high values of recall, this does not happen for the kNNq. And even when the datasets have overlapped classes SWq could keep high values of precisions outperforming kNNq in most cases.

Aiming at enriching our experiments, we also have evaluated the similarity methods applied to a real-world data set, the MedImg, considering each of its class separately, using the rule "class x others". Figure 19 displays the precision results obtained on MedImg data set taking into account each class size.

Figure 19 - Evaluation of SWq3 compared with k-NNq and Rq when applied to MedImg data set.



Source: Research data.

As we can notice in Figure 19, SWq3 shows in all cases better performance results than k-NNq. It can be noticed that the traditional methods tend to lose precision as the result set increases, indicating saturation of their performance; different from the proposed method that preserves its precision levels.

In MedImg's third class the performance trend lines are very close to each other. The class is the largest, the most scattered and the most mixed class of the dataset, and as we have selected the border objects between classes as the query elements, then the result of all queries are mixed enough to bring down the precision of all methods.

6.6. Final Considerations

In this chapter, we described a new method called SimWalk query, proposed in order to perform a similarity query, which considers all the elements of the dataset to evaluate the candidates and choose the next element. The goal of this approach is to reduce the semantic gap. Hence, we developed three approaches, represented by algorithms that vary the rule to select the next candidate element to compose the query result set.

The first algorithm, SWq1, is very similar to the tourist walk rule. However, the result sets returned by this are not very well suited to the context of similarity retrieval, since the result elements tend to set up a path that moves away from the query element. Trying to solve this issue, the second algorithm, SWq2, uses the smallest sum of distances to the reference element and to the last selected element to choose the next one. And the third algorithm, SWq3, in its turn, uses the smallest sum to all elements in the walk, yielding a less dispersed result set.

We have evaluated the algorithms using the precision and recall measures, considering the pertinence of each element of the result set to the same class of the query element, since the class pertinence is a common reference to access the semantic proximity of two elements. In order to perform the evaluation experiments, we have used a set of public datasets, along with a database of medical images.

Firstly, we have compared the precision between the three variations of the proposed algorithm. As result, the best precisions were reached by SWq3, followed by SWq2. Thereby, we have adopted SWq3 as our best algorithm to perform the next test sequence.

Subsequently, we have compared SWq3 with the traditional similarity methods, namely k-nearest neighbor query (k-NNq) and range query (Rq), using the same datasets experimented with the algorithms. The experiments show that SWq3 yields more precise results than k-NNq and Rq, in particular when the result set tends to increase in size. We can

also observe that the gain in the performance of our proposed method is larger for more disjoint datasets, as was already expected, due to the walk effectiveness in keeping the result in the same class when the query elements are at the borderline.

In most studied databases, the proposed algorithm's precision does not decrease when recall value increases. In contrast, k-NNq have a significant decrease of precision when the result set size is greater than a limit that ranges from 2% to 30% of the data set size, depending on the data set. Accordingly, when the result set size is greater than this limit k-NNq present a very low precision. Contrarily, SWq3 tends to maintain the same level of precision and this leads to a considerable gain.

Therefore, from these results, we can conclude that SimWalk method outperforms the traditional query methods in retrieving elements from the same class. Thus, we can assume that the proposed method represents a more effective choice when it is intended to reduce the semantic gap in similarity query environments.

Chapter 7 - METHOD 2: DIVERSITY WALK

7.1. Initial Considerations

Various databases and information retrieval systems have nowadays incorporated in query results a new capability that enables to return elements considering similarity and diversity features, i.e., the retrieved query elements should be as similar as possible to a query element, and, simultaneously, they should be as diverse as possible with each other.

This is also known as Result Diversification Problem and can be formally defined as a *tradeoff* between finding similar elements to a query, and diverse elements in the result set (VIEIRA; RAZENTE; BARIONI; HADJIELEFTHERIOU *et al.*, 2011).

Generally, in query diversification applications, a process of two steps generates the final result set. First, a candidate set S is created with elements similar to the query center element. After this, a result set R, subset of S, is generated considering elements similar to the query center element and, at the same time, as diverse as possible to other elements in the result set R. In order to control the preference between similarity and diversity the user can choose a tradeoff parameter. For instance, suppose that a candidate set S has a large amount of redundant element, an increment in the tradeoff value introduces more diverse elements to the result set.

Thereby, motivated by the tourist walk we propose an approach that allows diversifying query results. This approach has the capability of returning elements more spread on the dataset, considering objects that are near to the query element center, far from it and also in intermediate regions of the dataset. The method controls the diversity of a query through two tradeoff parameters whose values range from zero to one. Thus, the user can balance how much diverse the query result must be.

In this chapter, we present our proposed approach for diversifying query results and also an investigation over the possible improvements provided. Through this approach, we expect that query results become more semantic in terms of diversity, retrieving objects that are in different positions of the data space. For the evaluation of the proposed approach, we measured its performance and scalability when it is applied to artificial and real-world datasets and compared it with the other two known approaches of the literature.

7.2. Literature Review

Query results diversification has been studied extensively in the literature and several approaches have been proposed. A survey of query result diversification (ZHENG; WANG; QI; LI *et al.*, 2017) provides a wide categorization of these proposed approaches.

In this work, we focus on the content-based diversification problem (DROSOU, 2012; ZIEGLER; MCNEE; KONSTAN; LAUSEN, 2005). A wide range of algorithms has been proposed to deal with this problem and, mostly, can be classified into two main groups: algorithms based on interchange operations and greedy algorithms.

Methods based on interchange operations focus on the increasing of results quality exchanging some element in the result set with a better one. This exchanging is performed through the maximizing of an objective function F, whose calculation is composed of the combination of both similarity and diversity.

Swap (YU, C.; LAKSHMANAN, L.; AMER-YAHIA, S., 2009) is an example of a method that generates a subset R from a candidate set S through the maximization of the objective function $\mathcal{F}$, this algorithm swaps (exchanges) continually elements that contribute least to diversity with the next most relevant element considering all elements in S. BSwap (YU, C.; LAKSHMANAN, L.; AMER-YAHIA, S., 2009) is another method with similar characteristics to the Swap method; the difference is that in each iteration, the sum of diversity is increased, thus, elements in the result set are as dissimilar as possible from each other.

Approaches based on greedy algorithms still aims at maximizing an object function, however, the result set is constructed by selecting each "optimal" element from a candidate set following some criterion.

Kan *et al.* (KHAN; DROSOU; SHARAF, 2013) propose an approach that deals with the problem of diversifying the results of multiple queries, which was called DoS (diversification of multiple search results). The DoS leverages the natural overlap in search results along with the concurrent diversification of those overlapping results.

In Borodin et al. (BORODIN; LEE; YE, 2012), two simple algorithms are proposed, both are a generalization of the max sum diversification. One of the algorithms is greedy and does not try to optimize an objective function; instead, it tries to optimize a closely related potential function. The other algorithm is a local search algorithm for an arbitrary matroid constraint.

In Veira *et al.* (VIEIRA; RAZENTE; BARIONI; HADJIELEFTHERIOU *et al.*, 2011), a set of algorithms for query result diversification is well presented and described and also two novel algorithms are proposed. One is a greedy marginal contribution (GMC) method, which calculates a maximal marginal contribution (MMC) using three components. The first

component is a similarity function, the second is a diversity function between elements, and the last calculates the diversity between elements in the result set R. This method constructs the result set R by picking the element with the highest MMC value. The other method of Veira *et al.* (VIEIRA; RAZENTE; BARIONI; HADJIELEFTHERIOU *et al.*, 2011) is also a greedy algorithm, named greedy randomized with neighborhood expansion (GNE). This method is a combination of both greedy and swapping approaches and uses the greedy randomized adaptive search procedure (GRASP) approach. The method first selects a subset R according to a greedy randomized ranking function that calculates the MMC, after this, the method improves R by swapping between the most diverse element in the candidate set and some element in R.

Another greedy algorithm for query result diversification is the clustering-based method (VAN LEUKEN; GARCIA; OLIVARES; VAN ZWOL, 2009), which is composed of two steps. Firstly, a clustering medoid algorithm is applied and a number of clusters are obtained according to a dissimilarity function. Then, in the second step, the algorithm selects an element from each cluster to be part of the result set R.

Most of these approaches diversify query result considering the maximization of an objective function. These have two drawbacks regarding the efficiency and the performance. The maximization of this function requires that all distance between elements are summed and tested or that all distances are pre-computed, thus, this increase the complexity of these methods. Besides, these methods are not capable of getting elements in intermediate regions because they tend to select the most similar elements or the most diverse ones.

7.3. Proposed Approach

The majority of the existing approaches for diversity explore only aspects related to time performance and do not consider that, for semantic purposes, it is important that the set of objects are representatives of the broader search subspace. The existing methods set up the result set merely by adding the nearest and the farthest objects to the query one. Thus, in this work, we propose a new method to combine similarity and diversity, which generates a result set where the objects are ideally distributed in the search subspace in a uniform way.

The proposed approach, which we named as Diversity Walk (DivWalk), is based on a walk to select the elements that will be part of the result set. DivWalk adopts a specific movement rule using a memory that we call itinerary, whose structure works as a queue that is filled with elements visited at each time step. Thus, when the itinerary is completely filled, it represents that the walk is over and the result set is complete. The max sum of distances

makes the tourist to perform a walk at the marginal elements of the map. So, in this case, this trajectory represents the most diverse objects of the dataset.

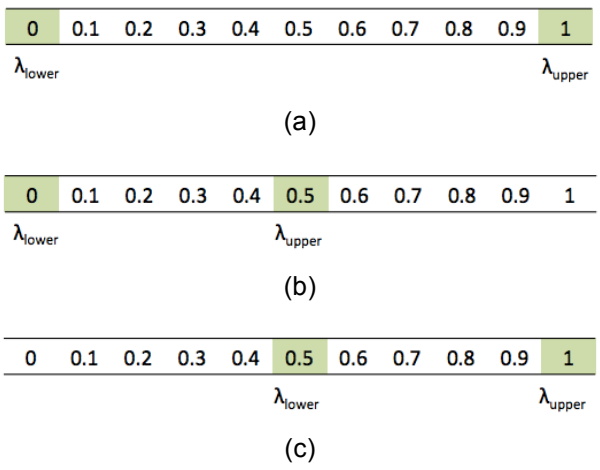
Notwithstanding, in order to allow the controlling of how much diverse or similar to the query object the result must be, we have included two tradeoff parameters: λ_{lower} and λ_{upper}. These parameters values are within the range [0:1]; when the value is 0 then the obtained result set is more similar to the query object, otherwise, if that value is 1, the result set contains objects more dissimilar from each other.

The tradeoff parameter λ works differently in our method because it acts over the sum list that the method keeps for defining, for instance, the max sum. Hence, through this parameter it is possible to choose from the list the object that has the minimum sum of distance (tradeoff value near to 0), or to select the object that has the max sum of distance (tradeoff value near to 1), or even to pick up the object that have intermediate sums defined through values within the range [0:1].

As we have two tradeoff parameters, the first one is used to control the lower value of tradeoff, we called it λ_{lower} , and the other is used to control the upper value of the tradeoff, we called it λ_{upper}. It is important to emphasize that the λ_{lower} value must be lower than or equal to the λ_{upper} value, because it defines the range of the tradeoff used to return objects more similar, more diverse, or even more spread in a dataset.

For illustration purposes, Figure 20 shows some examples of how the tradeoff parameter lambda can be configured in the method. When the tradeoff parameters are configured with $\lambda_{lower} = 0$ and $\lambda_{upper} = 1$ (Figure 20 (a)) the result set contains objects that are dispersed on the original dataset.

Figure 20 - Examples of tradeoff parameter values λ_{lower} and λ_{upper}.



Source: made by author.

Another situation is when the tradeoff parameters are configured with $\lambda_{lower} = 0$ and $\lambda_{upper} = 0.5$ (Figure 20 (b)), the result set comprehends objects that are dispersed on the dataset but also tend to be similar to the search object.

On the other hand, when we have the tradeoff parameters configured as $\lambda_{lower} = 0.5$ and $\lambda_{lower} = 1$ (Figure 20 (c)), the result set includes objects that are dispersed on the dataset but also tends to be more dissimilar to each other.

To generate a well-dispersed result set, DivWalk initiates the tradeoff parameter with λ_{lower} and, in each step of the walk, changes the value of the tradeoff parameter by an increment λ_{inc} $(\lambda_{upper} - \lambda_{lower})/k$, where k is the size of the desired result set, achieving λ_{upper} at the end. Thereby, the method considers different degrees of distance sum values defined by the parameter range when selecting the next object to be visited. This functionality represents a substantial advantage of our method over the others.

For the sake of clarity, Figure 21 shows an example of DivWalk being applied to an artificial dataset generated randomly. In this example, the parameters of the method were $\lambda_{lower} = 0$, $\lambda_{upper} = 1$, $k = 5$ and $q = 9$ (a query object also randomly chosen).

In Figure 21, we can see the two starting steps of the DivWalk, which allow understanding of how the method works. In the first step, Figure 21 (a), the method assembles the distance sum table, which is composed of distances from what is in the itinerary to other objects in the dataset. In this step, for instance, we have just one object in the itinerary (with id 9).

Thus, the sum list comprises the distance from this object to other ones in the dataset. After the sum list is assembled, the method calculates the median sum through the Eq. 12 and Eq. 13:

$$Sum = \sum_{i=1}^{|S|} d(id, s_i), s_i \notin Iti \tag{12}$$

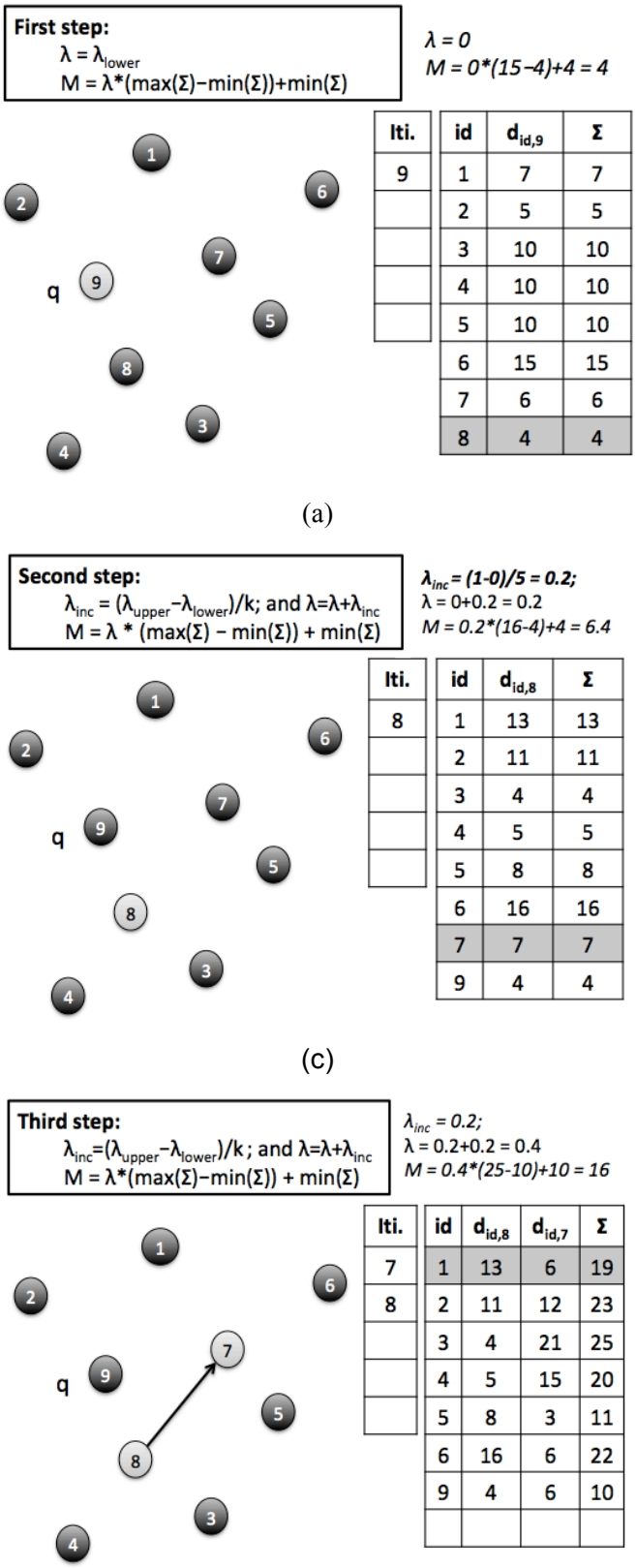
where $|S|$ is the dataset size, $d(\cdot, \cdot)$ s the distance between two objects, s_i is an object of the dataset and Iti represents the itinerary subset, and Sum is a set of distance sums.

$$M = \lambda \cdot (max(Sum) - min(Sum)) + min(Sum) \tag{13}$$

where λ is the tradeoff parameter, which in the first step corresponds to the λ_{lower}, and in the subsequent steps is defined as $\lambda = \lambda + \lambda_{inc}$, where λ_{inc} is the increment determined by $\lambda_{inc} = (\lambda_{upper} - \lambda_{lower}) / k$.

For example, in Figure 21 (b), we can observe the tradeoff increment λ_{inc} calculation, which in this case resulted in $\lambda_{inc} = 0.2$. Thus, in the next steps, λ is incremented by this value until the end of the method processing. This can be noticed in Figure 21 (c), which represents the third step of the method, where λ is incremented by 0.2.

Figure 21 - Example of DivWalk applied to an artificial dataset.



Source: made by author.

Thereby, the method selects, in each step, as the next object to be visited the one which holds the Sum value closer to the M value, as we can notice it in Figure 21 (a), (b) and (c). For example: in (a), the M value result in 4 that is exactly the same value of the "object 8" Sum; in (b), the M value result in 6.4 that is near to the value of the "object 7" Sum; in (c), the M value result in 16 that is near to the value of the "object 1" Sum.

Finally, the method stops its process when the itinerary subset is completely filled with objects, and these objects contained in the itinerary correspond to the subset that the method returns as a result.

Another important detail is that the method, right after the first step, removes the object q from the itinerary so that it can make the result set more diversified when both λ_{lower} and λ_{upper} are higher. Otherwise, if the object q were kept, the result set would always contain the start query object and the result set wouldn't be so diversified. Still, even though the object q is removed this does not hinder it to be included in the result set in subsequent steps when similarity is desired.

Figures 22 and 23 present the algorithm DivWalk, the first shows the main function of the algorithm and the second explains how the algorithm selects the next element of the walk to be visited. Firstly, DivWalk has as input parameters the subset S, that represents the candidate set to be diversified, k is the number of elements to be returned by the DivWalk, q is the query element center, and λ_{lower} and λ_{upper} are the tradeoff parameters.

Figure 22 - Main function of the DivWalk.

Algorithm 4

```
1:  procedure DivWalk(S, k, q, λ_lower, λ_upper)
2:      Iti ← {q}
3:      λ ← λ_lower
4:      s_n ← findNext(S, Iti, λ)
5:      Iti ← enqueue(Iti, s_n, k)
6:      Iti ← {Iti\q}
7:      λ_inc ← (λ_upper − λ_lower)/k
8:      while |Iti| < k
9:              λ ← λ + λ_inc
10:             s_n ← findNext(S, Iti, λ)
11:             Iti ← enqueue(Iti, s_n, k)
12:     return(Iti)
```

Source: Made by the author.

The main function of the algorithm initializes the Iti with the query element center q, and the tradeoff is initialized with the λ_{lower} value. After that, the next element is selected by

the function findNext, and this element is queued in Iti by the function enqueue, this function acts as a queue data structure. Then, q is removed from Iti and the tradeoff increment λ_{inc} is calculated. Afterwards, inside the loop, λ is incremented by λ_{inc}, the next element $\mathrm{s_n}$ is selected by the function findNext, then the $\mathrm{s_n}$ is queued into Iti. This is done until Iti reach the size k.

The findNext is the function responsible for selecting the next element to be part of the itinerary. It begins creating an empty list of sums, which will store the sums of distances from all elements in the itinerary to all elements in the dataset S. Thus, the nested loop calculates these distances and store inside the Sum list. It is important to notice that some distances are not calculated, and then a $null$ is stored instead ($\mathrm{agg} \leftarrow \mathrm{null}$). This occurs when the element $\mathrm{s_i}$ is in the itinerary. So, the Sum list will be a sparse list with sums and $null$ values stored.

Thereafter, the median M is calculated using the Sum list min and max value. But right before it, λ was readjusted in findNext function (Figure 23, line 9) in order to better balance between similarity and diversity. Once the result is obtained, the algorithm calculates which is the near element that has the sum value near to the median value.

Figure 23 - Function findNext of the DivWalk.

Algorithm 5

```
1:  procedure findNext(S, Iti, λ)
2:      Sum ← {}
3:      for each s_i ∈ S do
4:          agg ← null
5:          for each s_j ∈ Iti do
6:              if s_i ∉ Iti then
7:                  agg ← agg + dist(s_j, s_i)
8:          Sum ← {Sum, agg}
9:      λ ← 0.8 * λ^2 + 0.2 * λ
10:     M ← λ * (max(Sum) − min(Sum)) + min(Sum)
11:     near ← which.min(abs(Sum − M))
12:     return(near)
```

Source: Made by the author.

7.4. Materials and Methods

The DivWalk algorithm and its functions were also created through the software R. We have used three datasets in order to evaluate our methods, two artificial and a real-world one.

7.4.1. Datasets

In the experiments we have conducted, we have used two elementary datasets proposed in (ULTSCH, 2003), which are also part of the Fundamental Clustering Problem Suite(FCPS). The first one is originally called Target, but we have modified it for better evaluate within the context of our purpose, we dubbed this modified version as Circle dataset. The other one is called Twodiamonds, which we have used in its original version.

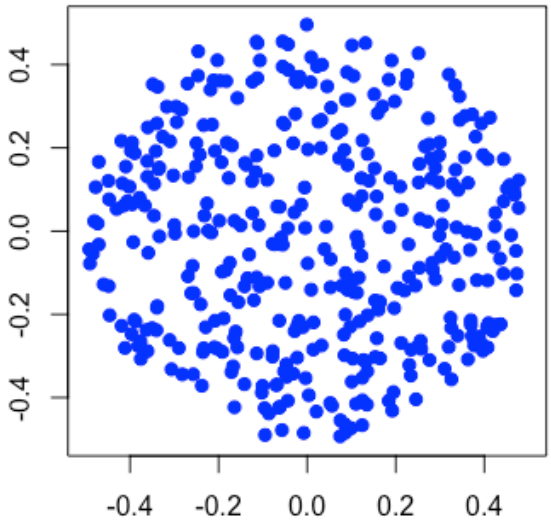
We have also used a real world dataset studied in (BALAN, 2007), which contains medical image features; for short, we called it MedImg dataset. As was explained in the past chapter, this dataset was modified for the purpose of a better visualization of the data distribution. Table 3 shows the datasets used in the evaluation considering the modifications for the purpose of better visualization. Although all datasets contain classes, these were not used in the experiments. Figure 24 presents the Circle dataset.

Table 3: Description of the modified versions of the dataset.

Name	Instances	Attributes
Wingnut	1016	2
Circle	395	2
MedImg	403	2

Source: Research data.

Figure 24 - Artificial dataset used in the experiments..



Source: Research data.

7.4.2. Diversity Evaluation Metric

The evaluation of our algorithm was performed considering the visualization of data scatter plots and we have calculated the variance of the distribution of the distances from the elements of the result sets to the reference element, in order to evaluate the elements distribution. In addition, we have accounted for the time processing to measure the efficiency and scalability of the algorithm.

The variance allows measuring the spread between data points in a data set. This measures how far each data point in the set is from the mean and is calculated by taking the differences between data point in the set and the mean, the square of the differences is calculated in order to make them positive and the sum of the squares is divided by the data point values in the set. The square root of the variance is the standard deviation (σ). The Eq. 14 defines the variance equation:

$$\sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n} \tag{14}$$

where x_i is the i^{th} data point , $\bar{x}$ is the mean of all data points, and n is the number of data points.

7.5. Experimental Results

In this section, we compared the proposed DivWalk algorithm, with two other algorithms, Swap and MMR, with respect to two aspects: 1) their capacity to uniformly distribute the result set elements; 2) their computational efficiency. And we used the variance as a way to measure if the obtained results set present data points more distributed over the feature space.

Although they are based on different kinds of approaches, we have used measurements that compare equally the capacity of each algorithm in their respective domain of problems.

7.5.1. Distribution

First, DivWalk was applied to the Circle dataset such that we could see the data distribution through data scatter plots. The dataset dispersion has a circle-shaped

configuration, which allowed us to evaluate how the algorithms behave when we want to return a more dispersed result set.

In order to compare the algorithms equally, our algorithm's tradeoff parameters (λ_{lower} and λ_{upper}) were both configured with the same value, for example, when Swap and MMR's tradeoff values are $\lambda = 0.3$, DivWalk's tradeoff values are $\lambda_{\text{lower}} = 0.3$ and $\lambda_{\text{lower}} = 0.3$. Figure 25 presents the data scatter plots of the algorithms results when applied to the Circle dataset.

We can see in plots from (a) to (f) that Swap and MMR return objects closer to the search one (center) when λ is 0.3, more distant, when λ is 0.9, but they are not so distributed when λ is 0.6, instead, they return some objects near to the center and include some very distant objects, but do not consider in-between objects.

On the other hand, DivWalk returns a more distributed result set, when λ is 0.6, thus, it is capable of considering objects between the center and the margins of the data points. The same occurs for $\lambda = 0.3$, in which DivWalk returns points similar to the center query object but not so agglomerated on it.

This is to show that, unlike DivWalk, the correlated algorithms are not linear in the point distribution of the result set, when we vary λ. This is one of the key advantages of DivWalk.

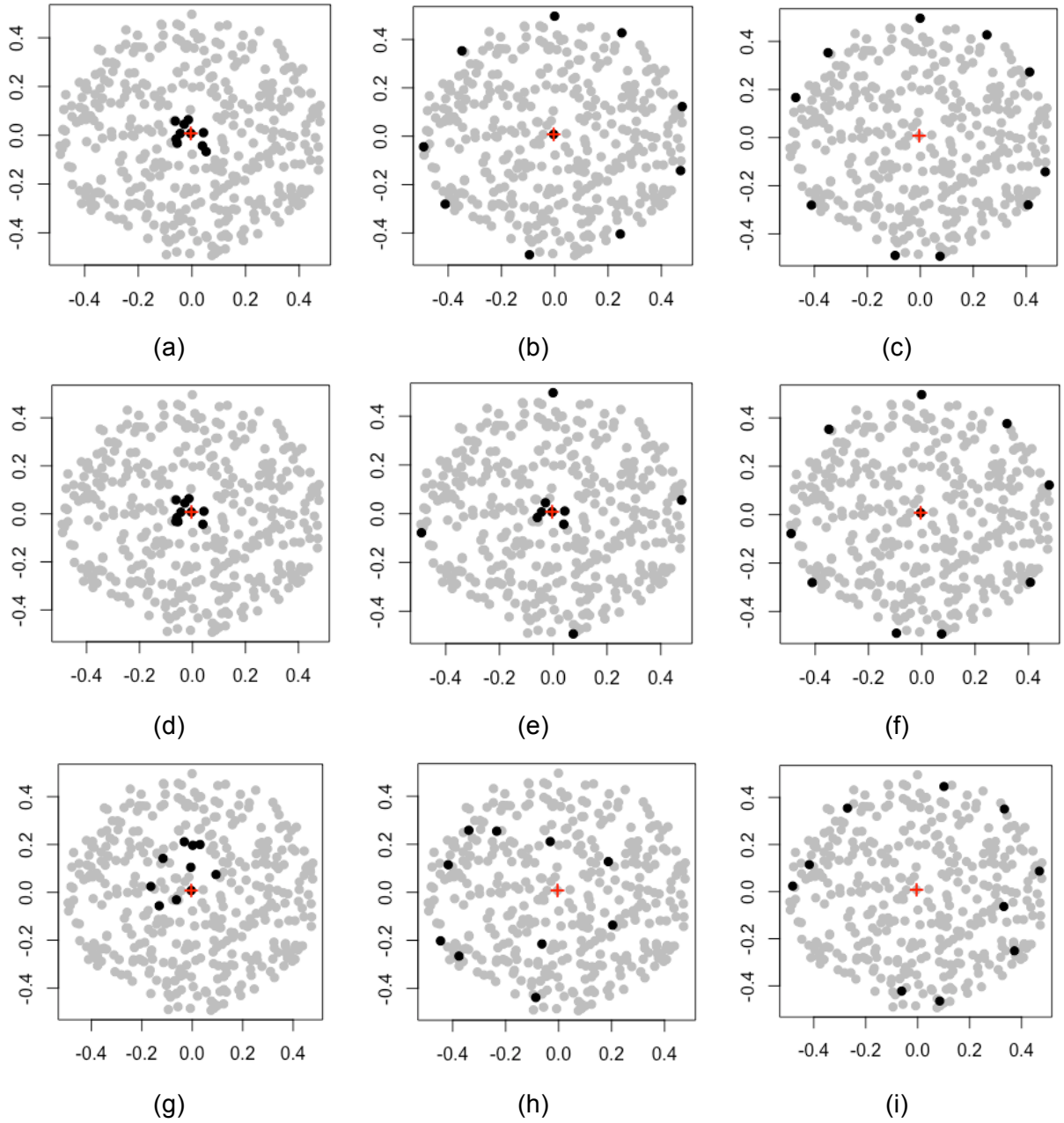
Furthermore, we have also applied the DivWalk algorithm to the Wingnut dataset also for the purpose of observing the data distribution through data scatter plots. Likewise, in the Circle dataset, for all algorithms, the same tradeoff parameter values were used.

Figure 26 presents the data scatter plots of the algorithms when applied to the Wingnut dataset.

In Figure 26, we can notice that the algorithms applied to Wingnut keep the same behavior with respects to their data distribution when the tradeoff parameter increases.

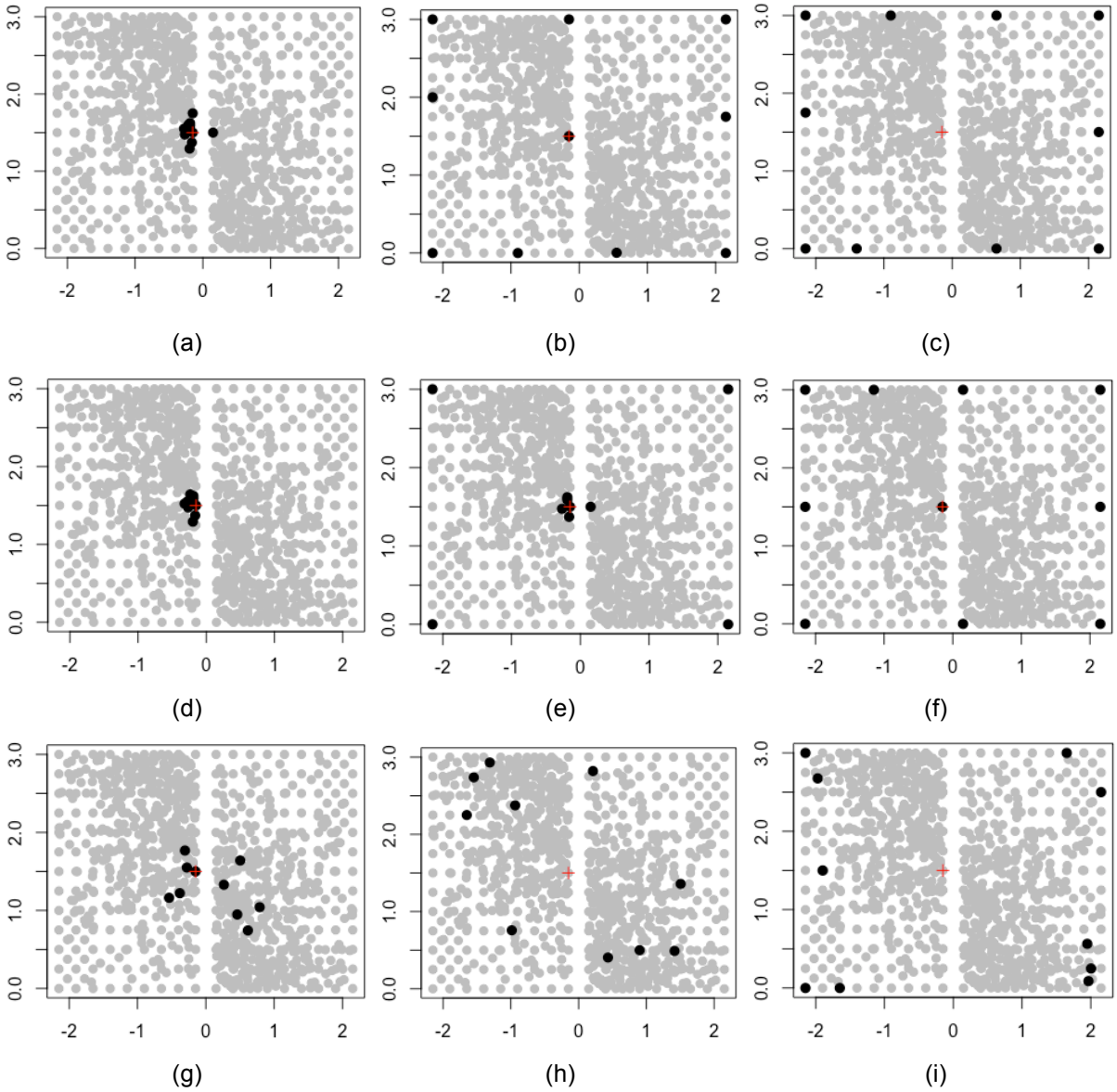
MMR and Swap diversify their query by selecting data near to the center object and far from it, not considering intermediate objects. And, unlike this, the DivWalk selects objects that are more distributed on the dataset.

Figure 25: Diversity algorithms experiment on Circle dataset. The result scatter plots of each algorithm are organized per line, and each line shows the algorithm varying the tradeoff parameter values of 0.3, 0.6 and 0.9, respectively. The first line – plots (a), (b), and (c) – shows the results of the Swap. The second line – plots (d), (e) and (f) – shows the results of the MMR. And the last line – plots (g), (h) and (i) – shows the results of the DivWalk.



Source: Research data.

Figure 26 – Diversity algorithms experiment on Wignut dataset. The result scatter plots of each algorithm are organized per line, and each line shows the algorithm varying the tradeoff parameter values of 0.3, 0.6 and 0.9, respectively. The first line – plots (a), (b), and (c) – shows the results of the Swap. The second line – plots (d), (e) and (f) – shows the results of the MMR. And the last line – plots (g), (h) and (i) – shows the results of the DivWalk.

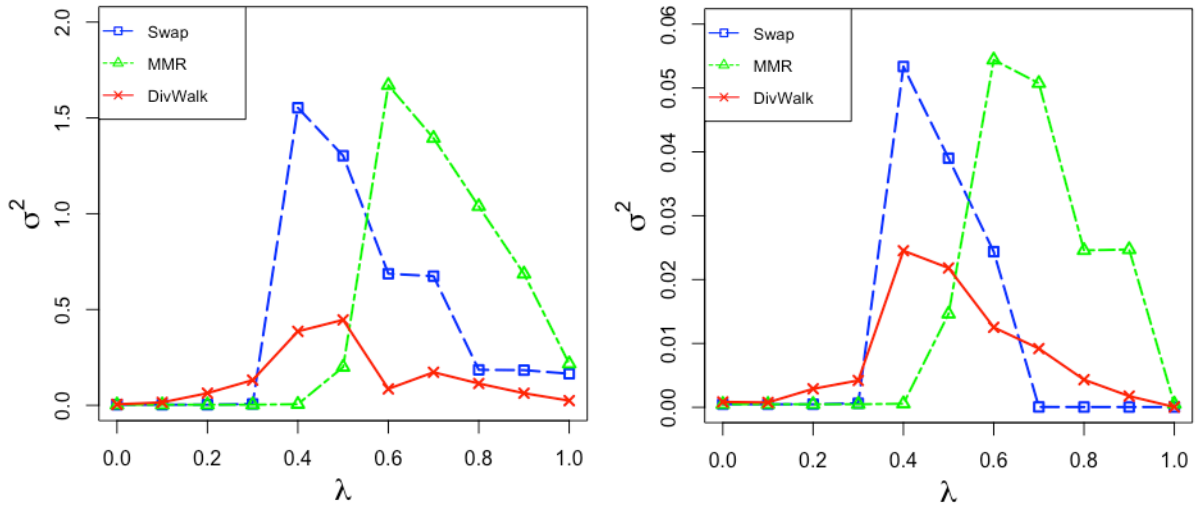


Source: Research data.

For the evaluation of our proposed algorithm, we have calculated the variances of the distances from the elements in the dataset to the query element, in each step of the walk, considering the variation of the tradeoff parameter values λ, which ranges from 0 to 1. These algorithms were applied to the Wingnut; Circle and MedImg datasets.

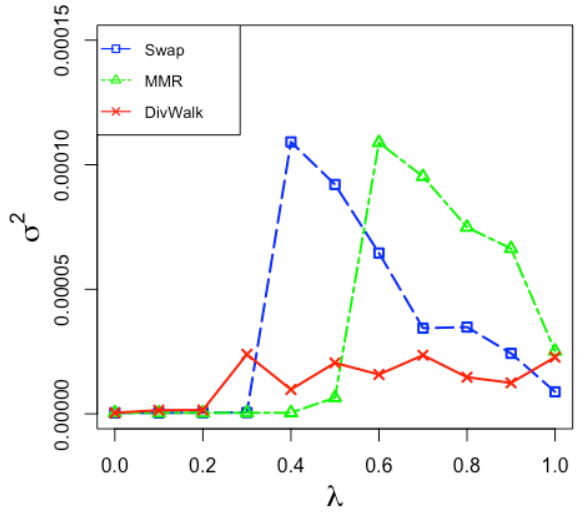
Figure 27 shows the variance plots of the algorithms Swap, MMR and DivWalk considering the variation of the λ values.

Figure 27: Variances of the algorithms Swap, MMR and DivWalk considering tradeoff parameter values (λ) in the interval [0:1] when applied to the datasets: (a) Wingnut; (b) Circle; and (c) MedImg.



(a) Wingnut

(b) Circle



(c) MedImg

Source: Research data.

We calculated the variances of the methods considering the distances of all result set objects to the query center object. In Figure 27, we can notice that DivWalk presents a better distribution regarding the variation of λ, because the distances don't change so much when the λ is changed. Thus, we can say that DivWalk keeps the distribution when more diversity is desired.

In other words, we can see that SWAP and MMR present low values of variance in regions of low values of lambda, with the abrupt increasing in regions of intermediate values of lambda and abrupt reduction when region of high lambda values are reached. This demonstrates in general what one observes in the results presented in Figure 25 and

26, that is, with low diversity, the returned elements have distance values to the query center element close to each other. With high diversity also in intermediate ranges, the high values of variance indicate that the result set have part of its values very close to the query center element and part of them very far. With the DivWalk, we can see that the variance variations are more regular, thus, this denotes that the objects distributions are homogeneously constructed, over all degrees of diversity.

7.5.2. Efficiency

The running time of the three algorithms was tested on MedImg dataset, which contains well-distributed data points and is large enough for our purpose. Beforehand, we can say that the Swap is the slowest algorithm due to its complexity that is $O(nk \log k)$, while MMR has a complexity of $O(nk)$ and DivWalk also has a complexity of $O(nk)$, this considering that the distance matrix is pre-computed.

Moreover, we have also evaluated the scalability of the algorithms regarding their running time while varying the query tradeoff parameters λ (from 0.1 to 0.9, incrementing by 0.1), the result set size k (from 5 to 20, incrementing by 5), and the candidate set size S (from 25 to 200, increment by the double). Figure 28 shows the results of the scalability evaluation.

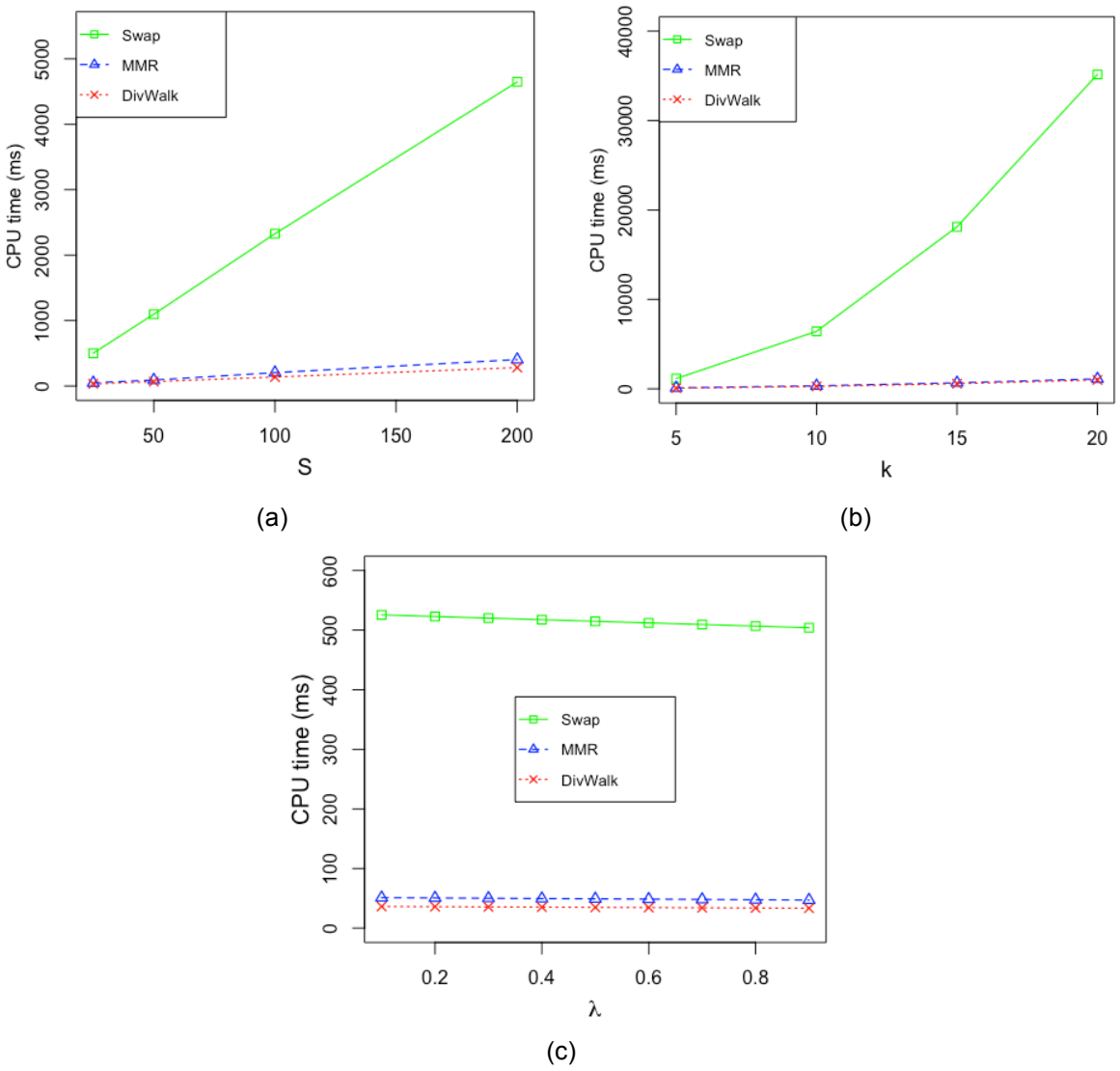
In Figure 28, we can notice that DivWalk and MMR are faster than the Swap regardless the variation of the parameters. Overall, DivWalk is slightly faster than MMR, except when the result set k is varied, in which they have almost the same running time.

In Figure 28 (a), the running time of all methods increases with S, since every element in S has to be checked. Swap had the highest running times since it performs several runs, which rely on k and S. MMR and DivWalk have had similar behavior in terms of increasing S, in the other hand, DivWalk was slightly faster.

In Figure 28 (b), we can notice that while the size of k is incremented the running times increase proportionally, as more iteration are performed over S in order to construct R. This is almost imperceptible for MMR and DivWalk because the running times increasing is very slight, but this can be easily noticed for Swap, which had significant running times increasing.

In Figure 28 (c), we can observe that the running time almost doesn't change when the parameter λ are varied, so we can say that this parameter doesn't interfere in the running time, but just is responsible for defining the selection of objects in the result set.

Figure 28: Scalability evaluation of the Swap, MMR and DivWalk algorithms regarding the variation of the candidate set size (S), the result set size (k) and the tradeoff parameter (λ) when applied to MedImg dataset.



Source: Research data.

7.6. Final Considerations

In this chapter, we have described our proposed method, called DivWalk, for diversifying query results. One of the advantages of the DivWalk is that users can set two parameters tradeoff that allows to control between finding the most relevant objects to the query and also finding the diverse objects in the result set. Moreover, DivWalk also allows finding objects that are more dispersed on the dataset, thus, improving the diversification of the query result set.

We described two known methods for query result diversification, which are called Swap and MMR. Both were used for comparison with the DivWalk in terms of performance and scalability.

DivWalk constructs the result set in an incremental way using a walker whose itinerary defines result set containing objects similar to the query center object or objects diverse from each other. A deterministic procedure is employed in order to choose, among different objects, the one to be included in the result set.

Regarding the experimental evaluation, we applied Swap, MMR and DivWalk to two artificial datasets and a real-world one. From that, DivWalk has demonstrated very good results with respect to the distribution of the result set data points. Hence, we can say that DivWalk has outperformed the two methods concerning the relevance and diversity between both in the result set, thus increasing the semantics of the query.

In order to measure the distribution of the result set data points, we have adopted the variance measurement and plotted the variances of the methods when the tradeoff parameter values are varied, this considering the MedImg dataset. The results have corroborated with the data scatter plots, confirming that DivWalk obtains a better data distribution even when more diversity is required.

Lastly, we have also evaluated our proposed method regarding its scalability, comparing the running time when the parameters are varied. In general, DivWalk has presented better result, except when the result set size is varied. Thus, in this case, DivWalk was very similar to MMR in terms of running time. Therefore, from these experiments, we can say that our method has a very good scalability, even when its parameters are varied. So we can conclude that the DivWalk is a very good alternative when we intend to diversify query results.

Chapter 8 - METHOD3: WALK FOR SUMMARIZATION

8.1. Initial Considerations

In this chapter, we present our proposed method to perform summarization on large datasets.

The proposed approach was named as SummarizationWalk. It relies on an agglomerative clustering algorithm in order to separate the dataset in clusters. The algorithm takes these clusters as input in order to perform the summarization of each dataset's class, using the proposed method 2 (DivWalk) to select elements from each class.

We evaluated the methods through the visualization of the data point's distribution and through two statistic metrics. For this, we have used one real world dataset and we created three artificial datasets with different aspects in order to test the capabilities of our proposed methods. Then, we chose two methods of the literature to compare with our proposed approach.

8.2. Literature Review

This section presents approaches and methods that deal with the topics of undersampling for dataset imbalance problem and data summarization. Some of the main proposed approaches were already discussed in Chapter 4, about imbalance problem and summarization. Here, we go through studies, applications and novel proposals with respect to these two topics.

8.2.1. Undersampling

In a recent study (DEVI; BISWAS; PURKAYASTHA, 2019), a novel approach is proposed to eliminate borderline, redundant and overlapping data instances in order to increase the classification accuracy of minority class instances and keeping the majority class accuracy in order to eliminate only a significant number of majority class instances. The approach adapts a one-class SVM-based anomaly detection to identify the cases of data

overlapping and also a modified Tomek-link undersampling is determined to handle both overlapped and imbalanced instances.

Another recent paper (ONAN, 2019) proposes a new technique to treat the problem of imbalanced datasets learning. This technique is based on a consensus clustering-based scheme and aims at reducing the number of instances in the majority class of an imbalanced dataset. The proposed scheme is defined by the combination of decisions of different clustering algorithms to get through their individual limitations so that more efficient clustering results can be obtained.

Guo and Wei (GUO; WEI, 2019) proposed a scheme composed of clustering and logistic regression processes in order to deal with imbalanced dataset. In this scheme, the clustering was used to split the majority class into clusters. In another recent study, Han et al. (HAN; HUANG; LI; JIA, 2019) present an approach focused on improving the learning process of imbalanced dataset considering the minority class. In the approach, instances of the minority class are separated into special groups of instances such as noisy, unstable, boundary, and stable based on the location information for the instances. This approach is applied on a medical diagnosis system.

Koziarski (KOZIARSKI, 2019) proposes a novel undersampling algorithm using the concept of mutual class potential focusing on preserving some of the performance gains provided by using the potential to lead the resampling, while at same the time it reduces the computational complexity. The proposed algorithm is referenced as Radial-Based Undersampling and is motivated on the notion of non-nearest neighbor based resampling, previously used in Radial-Based Oversampling, for the purpose of undersampling procedure.

A study in (VUTTIPITTAYAMONGKOL; ELYAN; PETROVSKI; JAYNE, 2018) proposes a new undersampling framework that mitigates the imbalance problem reducing the dominance of the majority class instances by removing it from the overlapping region. The authors called this framework Overlap-Based Undersampling method as OBU. The method explores a cluster algorithm to define which instances are in overlapped region. Then using OBU, overlapped negative instances can be removed.

The paper (GUO; DIAO; LIU, 2018) proposes a new method based on Rotation Forest ensemble learning, which is called Embedding Undersampling Rotation Forest (EURF), to handle class-imbalance problem. Beyond that, another paper explores two Self Organizing Maps that clusterize the rare and frequent instances in the original training dataset in order to mitigate the imbalance rate and to promote the correct detection of the rare instances (VANNUCCI; COLLA, 2018).

Another study (LIN; TSAI; HU; JHANG, 2017) presents two undersampling strategies, one that uses he k-means clustering method for undersampling in the class imbalance

domain problem, and the other several combinations of the clustering-based algorithm approach considering a set of different classification methods.

Besides that, another proposed method in the literature combine evolutionary undersampling with boosting to generate an ensemble classifier for breast cancer malignancy detection (KRAWCZYK; GALAR; JELEŃ; HERRERA, 2016).

8.2.2. Summarization

Regarding data summarization, in a recent study (KLEINDESSNER; AWASTHI; MORGENSTERN, 2019) it is proposed an algorithm that aims at output a small but representative subset of a large data set. This algorithm is based on a centroid-based clustering under a fairness constraint.

In a former study (CELIS; KESWANI; STRASZAK; DESHPANDE *et al.*, 2018), it is proposed a novel algorithm also based on fairness constraints, which incorporates fairness concerning sensitive attributes of data in sampling based on determinantal point process (DPP) for data summarization.

Another recent study (AHMED, 2019b) proposes a sampling based summarization algorithm that is able to create representative subset of large databases. This algorithm is applied on dataset inputs of anomaly detection algorithms in order to provide similar or the same performance as an anomaly detection applied on the original data.

In a other study (SHOU; LI, 2018), it is proposed a new technique for summarizing large datasets in order to assist local outlier detection. Furthermore, it is also proposed a new automatic parameter optimization approach and a method for parallel processing to accelerate the summary process.

Fernandes, Fanaee-T and Gama (FERNANDES; FANAEE-T; GAMA, 2018) propose a tensor method for real time structural pattern summarization of dynamic graphics, which is called tenClustS. This method consists of using tensor decomposition to simultaneously acquire the dynamics of dynamic networks and to reduce the dimensionality of the networks representation. Also in the idea of dynamic graph usage, Tsalouchidou et. al. (TSALOUCHIDOU; BONCHI; MORALES; BAEZA-YATES, 2018) propose two algorithms for summarizing dynamic large-scale graphs, one is based on cluster and the other uses micro-clusters concept to deal with the limitation of memory requirements. These studies are based on graph summarization, which is an approach that has been well studied in the literature for different purposes and applications, for example, summaries can be used for privacy and anonymity on networks (LEFEVRE; TERZI, 2010), can be used to create interpretable visualizations of the graph (NAVLAKHA; RASTOGI; SHRIVASTAVA, 2008), and also can be

used to store a compressed representation of the graph (RIONDATO; GARCÍA-SORIANO; BONCHI, 2017). Different approaches considering graph summarization has been also proposed - Shah et al (SHAH; KOUTRA; ZOU; GALLAGHER *et al.*, 2015) work on a proposal that approaches the problem of graph summarization as a compression problem and also extend it to dynamic graphs; Maserrat and Pei (MASERRAT; PEI, 2010) focus on neighbor queries; Hernández and Navarro (HERNÁNDEZ; NAVARRO, 2011) considers neighbor and community queries; Fan *et al.* (FAN; LI; WANG; WU, 2012) propose a summarization approach for reachability queries and another for graph patterns, and Toivonen et al. (TOIVONEN; ZHOU; HARTIKAINEN; HINKKA, 2011) propose an approach that creates a summary that maintain the distances between vertices.

8.3. Proposed Approach

In this section, we propose a new method for data summarization, which is based on clustering and diversification, considering the aspects presented in our proposed DivWalk method. The idea behind the method proposed here is to select a compact but meaningful representation of a large database. Thus, the method inputs a given dataset and transforms it to a smaller set of summaries focusing on retaining the maximum relevant information and features of the original dataset.

The proposed method for summarization makes use of some procedures defined in the DivWalk, but some parts were adapted in order to perform summarization instead of diversifying queries. Figure 29 presents the main procedure of the method.

The approach process can be divided in three parts: (1) the creation of the clusters; (2) the method that calculates the number of instances in which each class must retain in the summaries; (3) the algorithm that selects the instances that are dispersed on each cluster.

The main function of the algorithm is called SummarizationWalk that is responsible for calculate the amount of elements of each class to be selected given the value of ratio to be sampled. This function takes as input the ratio to be sampled and the lists of elements of each cluster. Thus, the algorithm requires that the dataset is already pre-classified by some clustering algorithm (part 1). And the function has also two parameter of control, one that allows to calculate the density of each cluster and the other that allows to adjust the samples when the sample value is too high and the class doesn't have enough elements to be sampled, thus, it can adjust the algorithm to get elements from another class.

The method has also two configuration options that allow calculating the amount of elements from each cluster: one proportional to the hyper volume of the cluster (keeping thus

the space ratio occupied by each cluster); the other proportional to the amount of elements of the cluster (keeping thus the cluster density).

Figure 29 presents the function SummarizationWalk of our proposed algorithm. The presented pseucode has functions that are native of the R platform that allows us to control the clusters as list and nested lists.

In Figure 29, we can notice that the first part of the algorithm receives through the parameter Clusters all the clusters that must be summarized; this parameter is a list of lists. Thus, each position of Clusters stores a list. Hence, the first algorithm's loop (lines 6 to 10) constructs a distance matrix for each one of these lists stored in Clusters. In each loop step, a matrix is constructed for the cluster stored in each Clusters' position. Then, the mean radius is calculated through the sum of all distances divided by the product of the matrix's dimensions and is stored inside the array $meanRadius$, and this together with the number of dimensions (dataset's features) are used to calculate the cluster's hyper volume. This is the first configuration option.

After the hyper volume is calculated for all clusters, the algorithm checks if the parameter flag ByHvm is set as true, if it isn't the algorithm gets the number of elements of each class and store inside the array hVm in the respective cluster hypervolume position (this is the second configuration option). Thereafter, we get the min value of the hVm ($minRadius$) and with this value we calculate the weight, defined by $weight = hVm/minValue$. Then, we calculate the amount of sample that must be summarized, determined by $Nam = weight * Nmr$.

So far, we can already use the array $Ncam$, that is the number of elements to be selected as sample from each class, and the Clusters nested list, that contains the list of clusters, to perform the summaries from each class (lines 36 to 41). For that, we apply the function walk, presented in Figure 30, in order to select the elements using the deterministic tourist walk heuristic. In case the amount of elements of being sampled exceed the amount of elements of some class, the algorithm adjust the sample (line 18 to 28) or the ratio to be sampled (line 30 to 35).

Figure 29 - Main function of the walk for summarization.

Algorithm 7

```
1:  procedure SummarizationWalk(Clusters, Ratio, ByHVm = T, AdjustSamples = F)
2:      clustersSize ← list with the size of each cluster
3:      nClusters ← number of clusters
4:      ndim ← dimension of the dataset
5:      N ← number of elements (rows) of the dataset
6:      for each cluter in Clusters do
7:          distMat ← distance matrix for each dataset's cluster
8:          meanRadius ← sum(distMat) / prod(dim(distMat))
9:          radiusClusters ← {radiusClusters, meanRadius}
10:         hVm ← {hVm, calculateHypervolume(meanRadius, ndim)}
11:     if ByHVm == F then
12:         hVm ← clustersSize
13:     minRadius ← min(hVm)
14:     weight ← hVm/minRadius
15:     Nam ← N * Ratio
16:     Nmr ← Nam / sum(weight)
17:     Ncam ← weight * Nmr
18:     if AdjustSamples == F then
19:         X = (Nam − sum(Ncam)/sum(weight))
20:         if X > 1 then
21:             for i in 1: nClusters do
22:                 if Ncam[i] <= Nc[i] then
23:                     Ncam[i]=Ncam[i]+X*weight[i]
24:         for i in 1: nClusters do
25:             if Ncam[i] >= Nc[i] then
26:                 Ncam[i] = clustersSize[i]
27:             else
28:                 Ncam[i] = Ncam[i]
29:     else
30:         for i in 1: nClusters do
31:             if Ncam[i] >= Nc[i] then
32:                 Ratio ← Nc[i] * sum(weight)/weight[i] * N
33:                 Nam ← N * Ratio
34:                 Nmr ← Nam / sum(weight)
35:                 Ncam ← weight * Nmr
36:     resultSet ← {}
37:     for i in 1: nClusters do
38:         k = round(Ncam[i])
39:         cluster ← Clusters[i]
40:         q ← centerOfMass(Clusters[i])
41:         resultSet ← {resultSet, walk(clusters, k, q)}
42:     return(resultSet)
```

Source: Research data.

In Figure 30, we can notice that the procedure is very alike to the one presented in Figure 23. However, there are some slight differences with respect to how the tradeoff parameter is defined and incremented. Firstly, the tradeoff parameter is auto adjustable, thus, the user can't change it because it is initiated as 0 and it is automatically incremented in

each step. This increment is determined by the max allowed value for λ (1) divided by the size of the desired result set (k), which in the procedure is calculated as $\lambda = \lambda + 1/k$.

Figure 30 - Main function of the walk.

Algorithm 6

```
1:  procedure walk(S, k, q)
2:      Iti ← {q}
3:      λ ← 0
4:      s_n ← findNext(S, Iti, λ)
5:      Iti ← enqueue(Iti, s_n, k)
6:      Iti ← {Iti\q}
7:      while |Iti| < k
8:              λ ← λ + 1/k
9:              s_n ← findNext(S, Iti, λ)
10:             Iti ← enqueue(Iti, s_n, k)
11:     return(Iti)
```

Source: Research data.

Thereby, in each step the tradeoff parameter is incremented when selecting the next element to be visited. This next element is selected through the function $findNext(S,\ Iti,\ \lambda)$, which is the same function presented in Figure 24 and that we have already explained in Chapter 7, section 7.3. Another function that is the same is the $enqueue(Iti,\ s_n,\ k)$, which is in charge of queuing the elements inside the itinerary.

8.4. Materials and Methods

The proposed summarization algorithm and its functions were also implemented using the software R. We have used three datasets in order to evaluate our methods, two artificial and a real-world one.

In order to evaluate the algorithms regarding their capability of summarizing a dataset even when the dataset has different aspects of density and volume of each class, we create three datasets varying these characteristics. All of them have 2 dimensions, in order to allow the visualization.

The dataset Artificial1 contains 375 instances and 4 clusters with different areas, keeping the same density. To keep the density, the number of instances of each cluster was set proportionally to its area. One of the clusters has 25 instances, the other has 50, the other has 100 and the last one has 200. And the areas of the clusters are approximately 0.5,

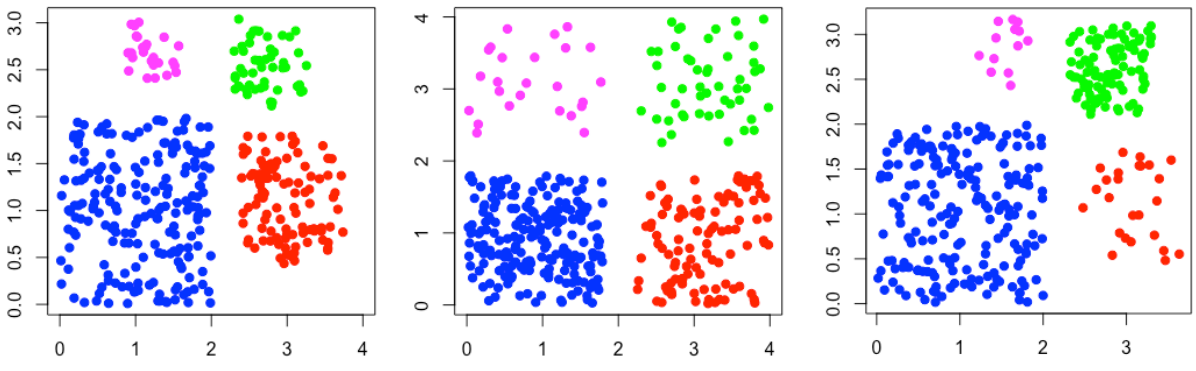
1, 2 and 4, respectively. With this dataset we expect to evaluate the algorithm when the size of the cluster is increased by the double, but keeps the same density.

The dataset Artificial2 also contains 375 instances and 4 clusters. The clusters' areas are the same and the number of instances of each cluster is the same as Artificial1. The areas of the clusters are 3.24. So, this dataset double the size and also the density among the clusters. With this dataset we expect to evaluate the algorithm considering the same cluster area and taking into account different but proportionally densities.

The dataset Artificial3 contains 338 instances and 4 clusters with different densities, and the clusters have different sizes but these are not proportional to each other. One of the clusters has 13 instances, the other has 25 instances, the other has 100 and the last one has 200. And the areas of the clusters are approximately 0.5, 1, 2 and 4, respectively. With this dataset we expect to evaluate the algorithm when the clusters are very different from each other.

The dataset Iris2d[3] contains 150 instances and three classes with different densities. It is a real world dataset known in the pattern recognition literature. In this dataset, one class is linearly separable from the other two; and the others are not linearly separable from each other. Figure 31 presents the artificial datasets that we created for the purpose of testing the summarization algorithms. Table 4 shows the datasets used in the evaluation considering the modifications for the purpose of a better visualization.

Figure 31 - Artificial datasets used in the experiments: (a) Artificial1; (b) Artificial2; and (c) Artificial3.



Source: Research data.

[3] https://archive.ics.uci.edu/ml/datasets/iris

Table 4: Description of the datasets.

Name	Instances	Attributes	Classes
Iris2d	150	2	3
Aritificial1	375	2	4
Artificial2	375	2	4
Artificial3	338	2	4

Source: Research data.

8.5. Experimental Results

In this section, we compared the proposed walk-based summarization algorithm that has two variations, one for sampling elements considering the hyper volume and the other that considers amount of objects per class. And compared these with two sampling methods of the literature.

One of the methods of the literature is the RandomSampling, which selects samples from the dataset at random. The other method is the SystematicSampling that selects samples from the dataset at regular intervals considering the size of the dataset. Both methods were already discussed in Chapter 4, Section 4.3. Figure 32 presents the results of the summarization methods applied on the iris2d dataset.

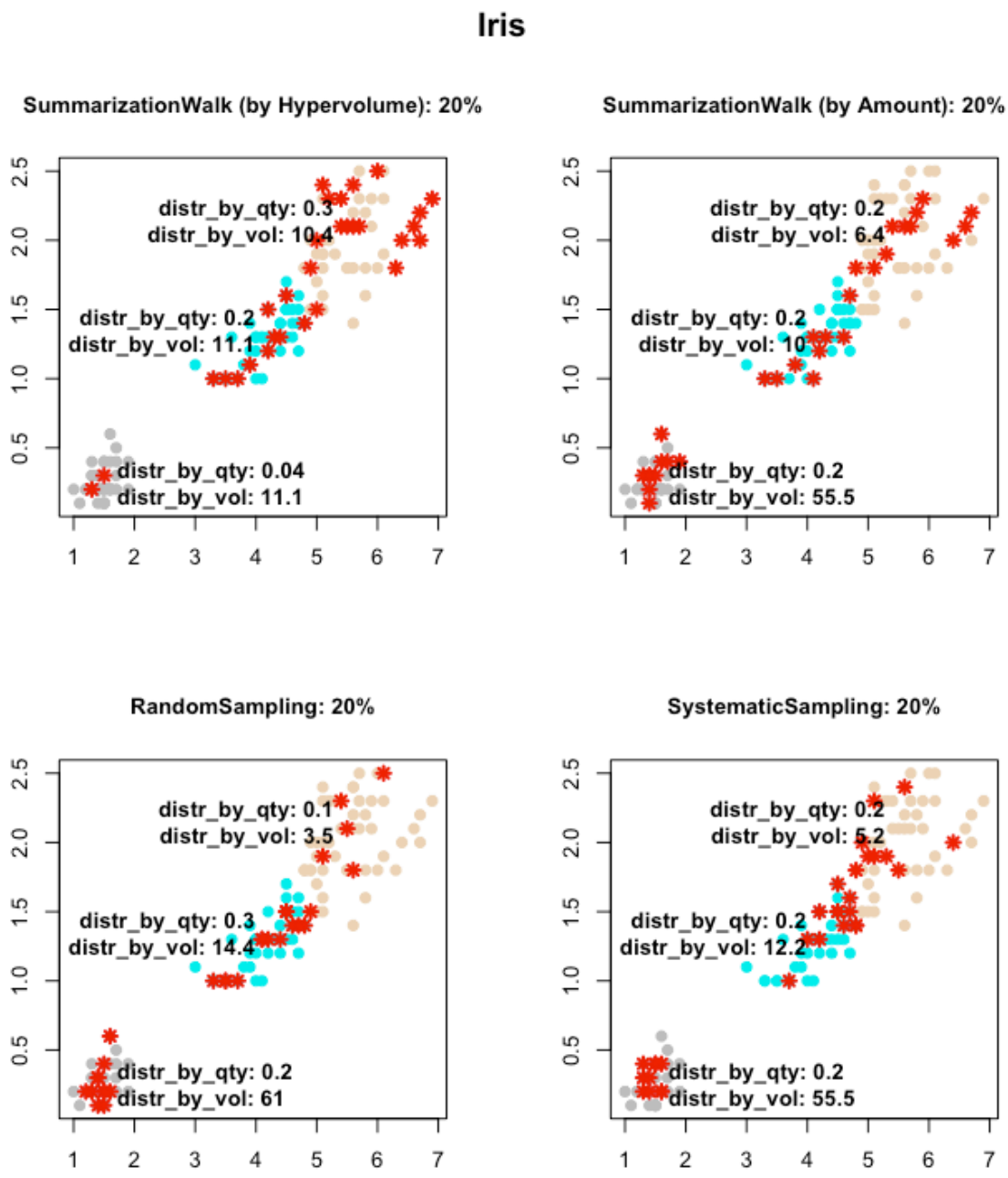
In order to evaluate the results returned by the methods, we defined and calculated two metrics defined by Eq. 13 and 14.

$$distr\_by\_qty = \frac{quantity\_returned\_class}{total\_amount\_class} \tag{13}$$

$$distr\_by\_vol = \frac{quantity\_returned\_class}{hypervolume\_class} \tag{14}$$

The first metric aims to show that the amount of objects returned of the class is proportional to the amount of objects of the class. The second metric aims to show that the amount of objects returned of the class is proportional to the volume occupied by the class.

Figure 32 - Evaluation of summarization methods on Iris2d, its clusters are represented by different colors. The red star points represent the results of the algorithm.



Source: Research data.

In Figure 32, we show the plots of each method applied to the Iris2d dataset that has three classes. These classes were separated through a hierarchical cluster technique based on centroids. For all methods we considered a sample of 20% of the original dataset. Each class of the dataset is differentiated by distinct colors and in each class there are two metrics that we described in Eq. 12 and 13.

In SummarizationWalk (by Hypervolume) plot, we can notice that the values of distr_by_vol of the 3 classes are very close, i.e., the results show a distribution of sampled elements according to the space occupied by each class. This is because the algorithm aims to sample elements considering the volume occupied by the class in the space.

In SummarizationWalk (by Amount) plot, we can see that the values of distr_by_qty of the 3 classes are equal, i.e., sampled elements are distributed in the classes and they keep the same amount of each class.

In the other hand, the plots and the metrics have shown that RandomSampling and SystematicSampling methods do not keep the quantity between the classes when select the elements, and we can observe this through the distr_by_vol that has quite different values for each class. This happens because both methods do not consider the construction of the result set by keeping distribution by class volume.

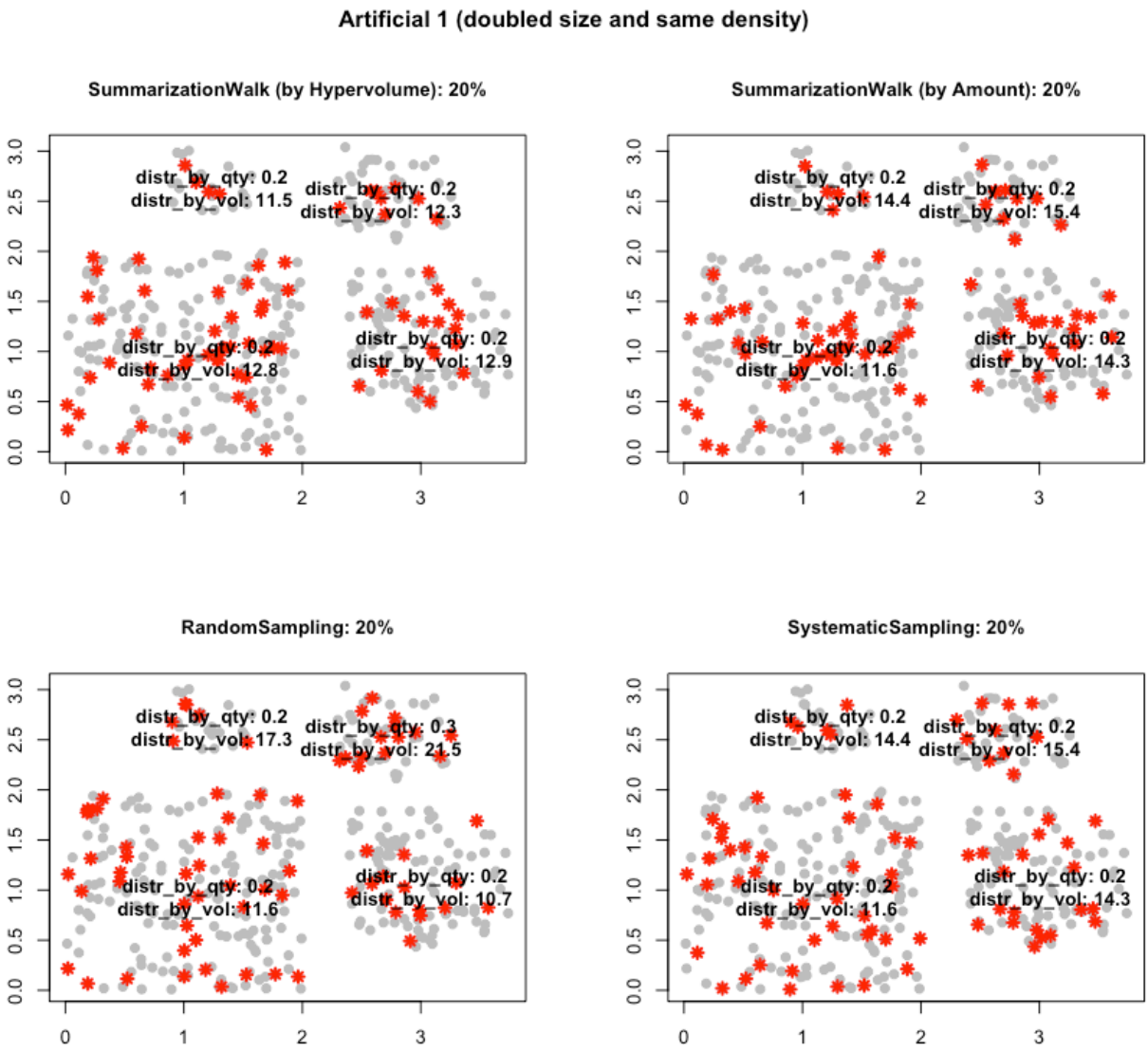
For the purpose of evaluating the methods considering dataset's aspects such as the double in sizes between classes and the no variation of density, we created a dataset distributing data points randomly but keeping the mentioned aspects. Figure 33 present the results of the methods applied to this artificial dataset, which we called Artificial 1.

In Figure 33, we can observe that Artificial 1 dataset has four classes. These classes were separated in four classes by the same hierarchical cluster technique that was aforementioned. For all methods we also considered a sample of 20% of the original dataset. In this case, we didn't defined colors for the classes because it can be easily noticed by the spaces between them. The statistical metrics printed in the plots follow the same order as it was defined for the Iris2d dataset.

In SummarizationWalk (by Hypervolume) plot, we can see that distr_by_vol for each class is very close to each other, thus, we can say that the algorithm selects the number of elements by class according the space occupied by the class, even when the number of elements is doubled.

In SummarizationWalk (by Amount) plot, as the classes keep the same distr_by_qty the algorithm results also keep the same amount of elements per class when selects the sample. For this case, RandomSampling and SystematicSampling methods also select the same amount of elements per class, but this is because of the dataset aspects.

Figure 33 - Evaluation of summarization methods on Artificial1.



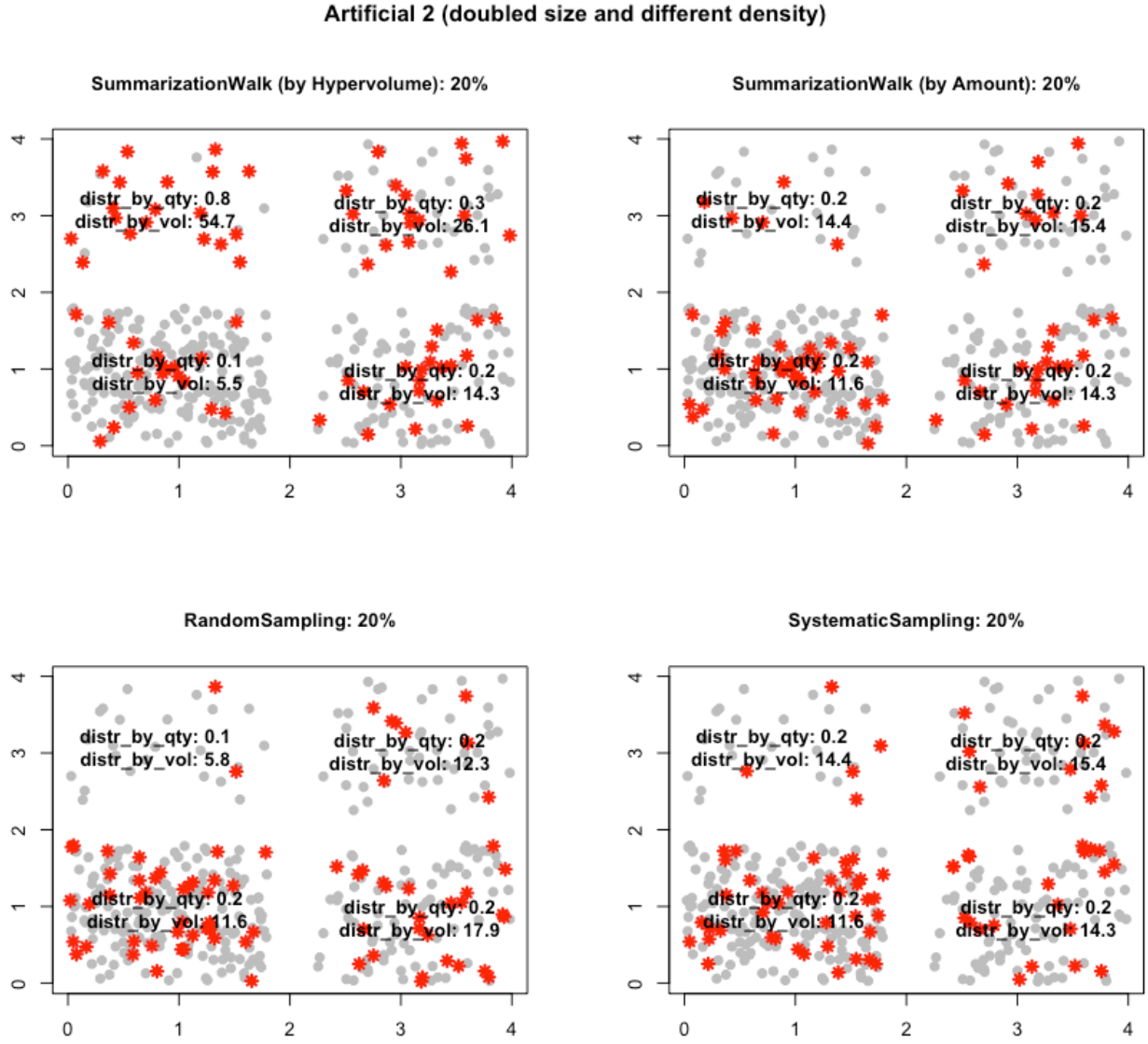
Source: Research data.

In order to evaluate the methods taking into account that the dataset has different densities and the double of size per class, we created an artificial dataset following these aspects but spreading the data points randomly by each class, we dubbed this dataset as Artificial 2.

Figure 34 shows the results of the methods for summarization applied to the Artificial 2 dataset.

In Figure 34, we can observe that Artificial 2 dataset has four classes. We also use the same hierarchical cluster technique to separate the classes, considering a sample of 20%, and the statistical metrics order is the same.

Figure 34 - Evaluation of summarization methods on Artificial 2.

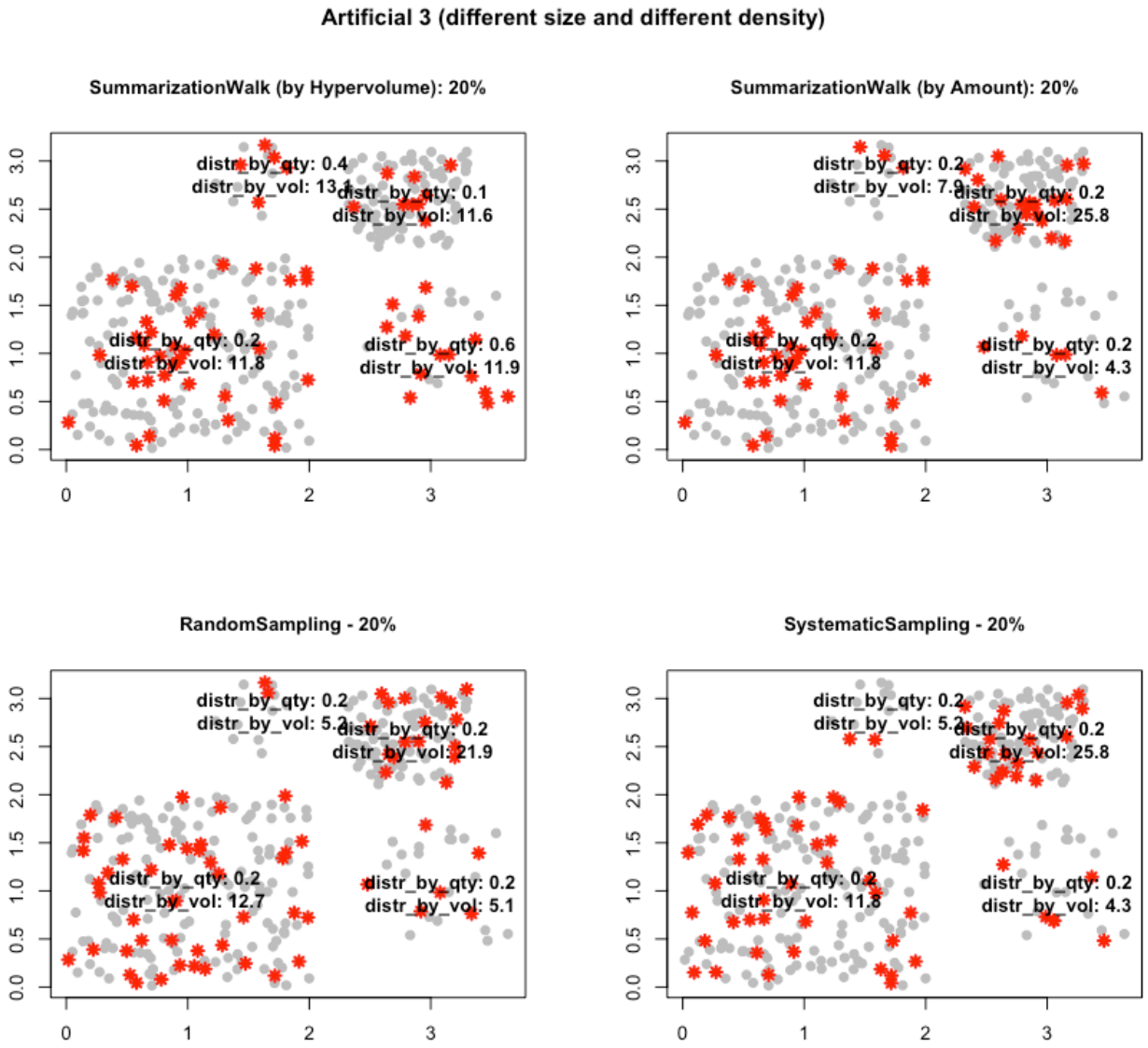


Source: Research data.

In SummarizationWalk (by Hypervolume), distr_by_vol shows values with significant differences among the classes. This is due to the dataset aspects, which has sizes doubling between the classes and different densities. On the other hand, SummarizationWalk (by Amount) keeps the value of distr_by_qty, which is related to the amount of elements per class. In this case, the SystematicSampling method has the same value of distr_by_qty. But RandomSampling still selects elements randomly and the amount of sample selected doesn't follow a specific standard.

Finally, we have created an artificial dataset that has different sizes and different densities, which we called as Artificial 3. Figure 35 presents the results returned by the methods for summarization when applied to the Artificial 3 dataset.

Figure 35 - Evaluation of summarization methods on Artificial3.



Source: Research data.

In Figure 35, we can notice that Artificial 3 dataset has also four classes, classified by a hierarchical clustering technique. The sample was also 20%, and the statistical metrics order is the same used in the last two artificial datasets.

The results of the method applied to the Artificial 3 dataset demonstrated what we have discussed in the other plots. In SummarizationWalk (by Hypervolume) plot, despite the different size and different densities between classes the method results keep distr_by_vol around the same value. Likewise, SummarizationWalk (by Amount) also selects elements respecting its rule that is keeping the distr_by_qty of each class. In this case, RandomSampling and SystematicSampling didn't succeed in maintain the volume of the sampled elements. However, both were able to maintain the values of distr_by_qty.

8.6. Final Considerations

In this chapter, we have presented our proposed approach for the creation of summaries from databases. This method is motivated by the process of the tourist walk that is capable of selecting elements in different positions of a dataset distribution. Hence, we developed an algorithm with two configurations for summarizing considering aspects of data points distributed in databases. One configuration aims to selecting data points that are distributed according to their space occupied in each class. The other setting focuses on selecting data points that are distributed according to the density of each class.

For comparison, we used two methods of sampling of the literature that are used to perform summarization, known as Random Sampling and Systematic Sampling. The first selects samples of a dataset at random and the second according to a random starting data point but with a fixed, periodic interval.

As regards the experimental evaluation, we applied the methods to a real world dataset and three artificial datasets constructed with aspects that test the algorithms with respect to the capability of selecting elements considering the distribution of the data points. The proposed approach presented positive results regarding the characteristics outlined by the datasets and we can say that it provides advantages over the other methods.

Chapter 9 - CONCLUSIONS

9.1. Considerations

To deal with databases with different aspects such as size, attributes, classifications, and redundancy, it is essential robust techniques that can:

- Retrieve information regarding similarity elements paying attention to reduce the semantic gap;
- Retrieve information considering not just the similarity of elements but also taking into account the diversity among elements of the dataset;
- An efficient approach to create a summary of a dataset in a proper way so that the semantic content between the elements in the dataset is preserved.

Nonetheless, reducing the semantic gap in a similarity search, including diversity in a search keeping its coherence and consistency, and representing large datasets using a semantic summary portion, is not a trivial task. These properties have been studied considering different aspects, which explore to treat the result set of a given technique in order to meet the users' expectations.

Thereby, the purpose of this work was to develop techniques to perform similarity searches considering the interrelationship between elements of a dataset, thus, improving the quality of the search by reducing the semantic gap. Besides that, to introduce the diversity factor in similarity search in order to enrich searches considering elements that are not too similar to a query center element. And also, to extract a representative subset of a dataset, in a summarized way, without losing the rich information of the original database. For this purpose, we developed, applied and tested new methods that use the tourist walk concept to obtain result sets that consider similarity, diversity and summarization.

9.2. Contributions

In this work, we proposed new approaches to process, represent and retrieve information regarding the similarity, diversity and the representation of summarized data.

Briefly, the main contributions of this work to the area of content-based data retrieval and even indirectly for the area of data mining are the following:

- In Chapter 6, we presented a new method for similarity retrieval and investigated over the possibility of improve the semantics of content-based searches. The most important inquiry in the performed experiments was the improvement of result sets when the interrelationship among elements of the database is considered, taking as reference all the objects that are being retrieved on a query, rather than only the initial query object.
- In Chapter 7, we introduced a new approach to diversify query results, aiming to improve the distribution and efficiency in the process of constructing diversified result sets. The proposed method constructs the result set incrementally through the use of a walker that selects data objects when it travels through objects that are near to the query center and also far from it. The proposed method demonstrated satisfactory results, by outperforming related works, regarding the distribution of the result set objects.
- In Chapter 8, we presented a novel strategy that uses our diversity method aiming at implement a new way to create summaries of a database. To this end, we developed two algorithms that deal with two issues of datasets objects distribution. One considers the space occupied by the objects in each dataset class. The other considers the density of each class. Both methods have displayed good result with respect of selecting samples from databases considering their different aspects of distribution.

9.3. Future Works

Considering the aforementioned contributions of this work, we can present some suggestions of topics to be explored by researches in order to extend the approaches presented in this document in order to deal with other class of problems. One possible investigation would be to investigate other metrics when selecting the next object to be chosen by the algorithms in the method for similarity and in the method for diversity too. Another important future investigation could be to develop a new metric access method based in the similarity method proposed in this work, to support the queries in the databases by an index structure. Finally, the proposed methods could be compared to the traditional ones in a controlled environment, considering the similarity of medical images evaluated by specialists.

REFERENCES

AGRAWAL, R.; GOLLAPUDI, S.; HALVERSON, A. and IEONG, S. Diversifying search results. **Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM '09)**, 10, p. 5-14, 2009.

AHMED, M. Data summarization: a survey. **Knowledge and Information Systems**, 58, n. 2, p. 249-273, 2019a.

AHMED, M. Intelligent Big Data Summarization for Rare Anomaly Detection. **IEEE Access**, 2019b.

AHMED, M.; MAHMOOD, A. N.; MAHER, M. J. **A novel approach for network traffic summarization**. Springer. p. 51-60, 2014.

AKGUL, C. B.; RUBIN, D. L.; NAPEL, S.; BEAULIEU, C. F. *et al.* Content-Based Image Retrieval in Radiology: Current Status and Future Directions. **Journal of Digital Imaging**, 24, n. 2, p. 208-222, Apr 2011. Article.

ASERY, R.; SUNKARIA, R. K.; MARWAHA, P.; SHARMA, L. D. Image Retrieval Techniques Using Content-Based Local Binary Descriptors: A Survey. *In*: **Handbook of Research on Advanced Concepts in Real-Time Image and Video Processing**: IGI Global, 2018. p. 173-195.

BACKES, A. R.; BRUNO, O. M.; CAMPITELI, M. G.; MARTINEZ, A. S. Deterministic tourist walks as an image analysis methodology based. **Progress in Pattern Recogniton, Image Analysis and Applications, Proceedings**, 4225, p. 784-793, 2006.

BACKES, A. R.; GONCALVES, W. N.; MARTINEZ, A. S.; BRUNO, O. M. Texture analysis and classification using deterministic tourist walk. **Pattern Recognition**, 43, n. 3, p. 685-694, Mar 2010. Article.

BALAN, A. G.; TRAINA, A. J.; RIBEIRO, M. X.; MARQUES, P. M. *et al.* Smart histogram analysis applied to the skull-stripping problem in T1-weighted MRI. **Computers in Biology and Medicine**, 42, n. 5, p. 509-522, 2012.

BALAN, A. G. R. **Métodos adaptativos de segmentação aplicados à recuperação de imagens por contéudo**. 2007. -, Universidade de São Paulo.

BARZILAY, R.; ELHADAD, M. Using lexical chains for text summarization. **Advances in automatic text summarization**, p. 111-121, 1999.

BATISTA, G. E.; PRATI, R. C.; MONARD, M. C. A study of the behavior of several methods for balancing machine learning training data. **ACM SIGKDD explorations newsletter**, v. 6, n. 1, p. 20-29, 2004.

BAXENDALE, P. B. Machine-made index for technical literature—an experiment. **IBM Journal of research and development**, v. 2, n. 4, p. 354-361, 1958.

BORODIN, A.; LEE, H. C.; YE, Y. **Max-sum diversification, monotone submodular functions and dynamic updates**. ACM. p. 155-166, 2012,

BUGATTI, P. H.; TRAINA, A. J.; TRAINA, C. **Improving content-based retrieval of medical images through dynamic distance on relevance feedback**. IEEE, p. 1-6, 2011,

BUNIMOVICH, L. A. Deterministic walks in random environments. **Physica D-Nonlinear Phenomena**, v. 187, n. 1-4, p. 20-29, Jan 2004.

CAI, Y. **Attribute-oriented induction in relational databases**. Theses (School of Computing Science)/Simon Fraser University, 1989.

CAMPITELI, M. G.; BATISTA, P. D.; KINOUCHI, O.; MARTINEZ, A. S. Deterministic walks as an algorithm of pattern recognition. **Physical Review E**, v. 74, n. 2, Aug 2006.

CAMPITELI, M. G.; MARTINEZ, A. S.; BRUNO, O. M. An image analysis methodology based on deterministic tourist walks. **Advances in Artificial Intelligence - Iberamia-Sbia 2006, Proceedings**, p. 159-167, 2006.

CAPANNINI, G.; NARDINI, F. M.; PEREGO, R.; SILVESTRI, F. Efficient diversification of web search results. **Proceedings of the VLDB Endowment**, v. 4, n. 7, p. 451-459, 2011.

CARBONELL, J.; GOLDSTEIN, J., 1998, **The use of MMR, diversity-based reranking for reordering documents and producing summaries**. ACM. 335-336.

CARTERETTE, B. **An analysis of NP-completeness in novelty and diversity ranking**. Springer. p. 200-211, 2009.

CELIS, L. E.; KESWANI, V.; STRASZAK, D.; DESHPANDE, A. *et al.* Fair and diverse DPP-based data summarization. **arXiv preprint arXiv:1802.04023**, 2018.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. **ACM computing surveys (CSUR)**, v. 41, n. 3, p. 15, 2009.

CHANDOLA, V.; KUMAR, V. Summarization–compressing data into an informative representation. **Knowledge and Information Systems**, v. 12, n. 3, p. 355-378, 2007.

COCHRAN, W. G. **Sampling techniques**. John Wiley & Sons, 2007.

CONROY, J. M.; O'LEARY, D. P. **Text summarization via hidden markov models**. ACM. 406-407, 2001.

COYLE, M.; SMYTH, B. On the importance of being diverse: analysing similarity and diversity in web search, Intelligent information processing II. Springer-Verlag, London, UK 2004.

DE AZEVEDO-MARQUES, P. M.; ROSA, N. A.; TRAINA, A. J. M.; TRAINA, C. *et al.* Reducing the semantic gap in content-based image retrieval in mammography with relevance feedback and inclusion of expert knowledge. **International Journal of Computer Assisted Radiology and Surgery**, v. 3, n. 1, p. 123-130, 2008.

DEMIDOVA, E.; FANKHAUSER, P.; ZHOU, X.; NEJDL, W., 2010, **DivQ: diversification for keyword search over structured databases**. ACM. 331-338.

DEVI, D.; BISWAS, S. K.; PURKAYASTHA, B. Learning in presence of class imbalance and class overlapping by using one-class SVM and undersampling technique. **Connection Science**, v. 31, n. 2, p. 105-142, 2019.

DOU, Z.; HU, S.; CHEN, K.; SONG, R. *et al.*, 2011, **Multi-dimensional search result diversification**. ACM. p. 475-484.

DROSOU, M. A. P. E. DisC diversity: Result diversification based on dissimilarity and coverage. **Pvldb**, v. 6, n. 1, p. 13-24, 2012.

EDMUNDSON, H. P. New methods in automatic extracting. **Journal of the ACM (JACM)**, v. 16, n. 2, p. 264-285, 1969.

ELFAYOUMY, S.; THOPPIL, J. A survey of unstructured text summarization techniques. **The International Journal of Advanced Computer Science and Applications**, v. 5, n. 7, p. 149-154, 2014.

FAN, W.; LI, J.; WANG, X.; WU, Y. **Query preserving graph compression**. ACM. 157-168, 2012.

FELIPE, J. C.; OLIOTI, J. B.; TRAINA, A. J.; RIBEIRO, M. X. *et al.* **A low-cost approach for effective shape-based retrieval and classification of medical images**. IEEE. p. 6-pp, 2005.

FELIPE, J. C.; RIBEIRO, M. X.; SOUSA, E. P.; TRAINA, A. J. *et al.* **Effective shape-based retrieval and classification of mammograms**. ACM. p. 250-255, 2006.

FELIPE, J. C.; TRAINA, A. J.; TRAINA, C., **Retrieval by content of medical images using texture for tissue identification**. IEEE. 175-180, 2003.

FELIPE, J. C.; TRAINA, A. J.; TRAINA JR, C. **A new similarity measure for histograms applied to content-based retrieval of medical images**. ACM. p. 258-259, 2006.

FELIPE, J. C.; TRAINA, C.; TRAINA, A. J. M. A new family of distance functions for perceptual similarity retrieval of medical images. **Journal of digital imaging**, v. 22, n. 2, p. 183, 2009.

FERNANDES, S.; FANAEE-T, H.; GAMA, J. Dynamic graph summarization: a tensor decomposition approach. **Data Mining and Knowledge Discovery**, v. 32, n. 5, p. 1397-1420, 2018.

FEUERSTEIN, E.; HEIBER, P. A.; MARTINEZ-VIADEMONTE, J.; BAEZA-YATES, R. **New stochastic algorithms for scheduling ads in sponsored search**. IEEE. p. 22-31, 2007.

FRATERNALI, P.; MARTINENGHI, D.; TAGLIASACCHI, M. **Top-k bounded diversification**. ACM. p. 421-432, 2012.

FREUND, H.; GRASSBERGER, P. The red queens Walk. **Physica A**, v. 190, n. 3-4, p. 218-237, Dec 1992.

GHOSH, N.; AGRAWAL, S.; MOTWANI, M. **A survey of feature extraction for content-based image retrieval system**. Springer. p. 305-313, 2018.

GOLLAPUDI, S.; SHARMA, A. **An axiomatic approach for result diversification**. ACM. 381-390, 2009.

GOLLAPUDI, S. A. S. A. An Axiomatic Approach for Result Diversification. **Proceedings of WWW '09**, p. 381-390, 2009.

GRADY, L. Random walks for image segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 28, n. 11, p. 1768-1783, 2006.

GUO, H.; DIAO, X.; LIU, H. Embedding Undersampling Rotation Forest for Imbalanced Problem. **Computational Intelligence and Neuroscience**, 2018, 2018.

GUO, H.; WEI, T. Logistic regression for imbalanced learning based on clustering. **International Journal of Computational Science and Engineering**, v. 18, n. 1, p. 54-64, 2019.

HA-THUC, V.; NGUYEN, D.-C.; SRINIVASAN, P., 2008, **A quality-threshold data summarization algorithm**. IEEE. 240-246.

HAN, J.; FU, Y.; HUANG, Y.; CAI, Y. *et al.* **DBLearn: A system prototype for knowledge discovery in relational databases**. ACM. p. 516, 1994

HAN, J.; FU, Y.; WANG, W.; CHIANG, J. *et al.*, **DBMiner: A System for Mining Knowledge in Large Relational Databases**. p. 250-255, 1996.

HAN, W.; HUANG, Z.; LI, S.; JIA, Y. Distribution-sensitive unbalanced data oversampling method for medical diagnosis. **Journal of medical Systems**, v. 43, n. 2, p. 39, 2019.

HARALICK, R. M.; SHANMUGAM, K. Textural features for image classification. **IEEE Transactions on systems, man, and cybernetics**, v. 3, n. 6, p. 610-621, 1973.

HARDIMAN, S. J. A. K. L. Estimating Clustering Coefficients and Size of Social Networks via Random Walk. **World Wide WEb**, p. 1-11, 2013.

HART, P. The condensed nearest neighbor rule (Corresp.). **IEEE transactions on information theory**, v. 14, n. 3, p. 515-516, 1968.

HERNÁNDEZ, C.; NAVARRO, G. **Compression of web and social graphs supporting neighbor and community queries**, 2011.

JAGADISH, H.; MADAR, J.; NG, R. T. **Semantic compression and pattern extraction with fascicles**. p. 7-10, 1999.

JEH, G.; WIDOM, J. **SimRank: a measure of structural-context similarity**. ACM. p. 538-543, 2002.

KHAN, H. A.; DROSOU, M.; SHARAF, M. A. **Dos: an efficient scheme for the diversification of multiple search results**. ACM. p. 40, 2013.

KHOTANZAD, A.; HONG, Y. H. Invariant image recognition by Zernike moments. **IEEE Transactions on pattern analysis and machine intelligence**, v. 12, n. 5, p. 489-497, 1990.

KINOUCHI, O.; MARTINEZ, A. S.; LIMA, G. F.; LOURENCO, G. M. *et al.* Deterministic walks in random networks: an application to thesaurus graphs. **Physica a-Statistical Mechanics and Its Applications**, v. 315, n. 3-4, p. 665-676, Dec 2002. Article.

KLEINDESSNER, M.; AWASTHI, P.; MORGENSTERN, J. Fair k-center clustering for data summarization. **arXiv preprint arXiv:1901.08628**, 2019.

KO, B.; LEE, H.-S.; BYUN, H. **Image retrieval using flexible image subblocks**. ACM. p. 574-578, 2000.

KOZIARSKI, M. Radial-Based Undersampling for Imbalanced Data Classification. **arXiv preprint arXiv:1906.00452**, 2019.

KRAWCZYK, B.; GALAR, M.; JELEŃ, Ł.; HERRERA, F. Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy. **Applied Soft Computing**, v. 38, p. 714-726, 2016.

KUBAT, M.; MATWIN, S. **Addressing the curse of imbalanced training sets: one-sided selection**. Nashville, USA. p. 179-186, 1997.

KUBY, M. J. Programming models for facility dispersion: the p-dispersion and maxisum dispersion problems. **Mathematical and Computer Modelling**, v. 10, n. 10, p. 792, 1988.

KUMAR, A.; KIM, J.; CAI, W.; FULHAM, M. *et al.* Content-based medical image retrieval: a survey of applications to multidimensional and multimodality data. **Journal of digital imaging**, v. 26, n. 6, p. 1025-1039, 2013.

KUMAR, S.; ONG, S. H.; RANGANATH, S.; ONG, T. C. *et al.* A rule-based approach for robust clump splitting. **Pattern Recognition**, v. 39, n. 6, p. 1088-1098, 2006.

KUPIEC, J.; PEDERSEN, J.; CHEN, F. A trainable document summarizer. **Advances in Automatic Summarization**, p. 55-60, 1999.

LARSEN, B. A trainable summarizer with knowledge acquired from robust NLP techniques. **Advances in automatic text summarization**, p. 71, 1999.

LEE, S.; BELKASIM, S.; ZHANG, Y. **Multi-document text summarization using topic model and fuzzy logic**. Springer. p. 159-168, 2013.

LEFEVRE, K.; TERZI, E., **GraSS: Graph structure summarization**. SIAM. p. 454-465, 2010.

LI, X.; LV, Q.; HUANG, W. Learning similarity with probabilistic latent semantic analysis for image retrieval. **KSII Transactions on Internet and Information Systems (TIIS)**, v. 9, n. 4, p. 1424-1440, 2015.

LIANG, R.-Z.; SHI, L.; WANG, H.; MENG, J. *et al.* **Optimizing top precision performance measure of content-based image retrieval by learning similarity function**. IEEE. p. 2954-2958, 2016.

LIMA, G. F.; MARTINEZ, A. S.; KINOUCHI, O. Deterministic walks in random media. **Physical Review Letters**, v. 87, n. 1, Jul 2001.

LIN, C.-Y. **Training a selection function for extraction**. ACM. p. 55-62, 1999.

LIN, C.-Y. **Rouge: A package for automatic evaluation of summaries**. p. 74-81, 2004.

LIN, C.-Y.; HOVY, E. **Identifying topics by position**. p. 283-290, 1997.

LIN, W.-C.; TSAI, C.-F.; HU, Y.-H.; JHANG, J.-S. Clustering-based undersampling in class-imbalanced data. **Information Sciences**, v. 409, p. 17-26, 2017.

LIU, Z.; SUN, P.; CHEN, Y. Structured search result differentiation. **Proceedings of the VLDB Endowment**, v. 2, n. 1, p. 313-324, 2009.

LU, P.; PENG, X.; ZHU, X.; WANG, X. Finding More Relevance: Propagating Similarity on Markov Random Field for Image Retrieval. **arXiv preprint arXiv:1312.7085**, 2013.

LUHN, H. P. The automatic creation of literature abstracts. **IBM Journal of research and development**, v. 2, n. 2, p. 159-165, 1958.

MACQUEEN, J. **Some methods for classification and analysis of multivariate observations**. Oakland, CA, USA. p. 281-297, 1967.

MACROPOL, K. A. C. T. A. S. A. RRW: repeated random walks on genome-scale protein networks for local cluster discovery. **BMC Bioinformatics**, v. 10, n. 1, p. 283, 2009.

MAHMOOD, A. N. **Hierarchical clustering and summarization of network traffic data**. 2008.

MANI, I.; ZHANG, I. **kNN approach to unbalanced data distributions: a case study involving information extraction**, 2003.

MANJUNATH, B. S.; MA, W. Y. Texture features for browsing and retrieval of image data. **Ieee Transactions on Pattern Analysis and Machine Intelligence**, v. 18, n. 8, p. 837-842, Aug 1996.

MANNING, C.; RAGHAVAN, P.; SCHÜTZE, H. Introduction to information retrieval. **Natural Language Engineering**, v. 16, n. 1, p. 100-103, 2010.

MASERRAT, H.; PEI, J. **Neighbor query friendly compression of social networks**. ACM. p. 533-542, 2010.

MCKEOWN, K.; KLANVAS, J. L.; EVANS, D. K. Similarity-based multilingual multi-document summarization, 2005.

MÜLLER, H.; MICHOUX, N.; BANDON, D.; GEISSBUHLER, A. A review of content-based image retrieval systems in medical applications—clinical benefits and future directions. **International journal of medical informatics**, v. 73, n. 1, p. 1-23, 2004.

NAVLAKHA, S.; RASTOGI, R.; SHRIVASTAVA, N. **Graph summarization with bounded error**. ACM. p. 419-432, 2008.

NEWMAN, M. E. J. A measure of betweenness centrality based on random walks. **Social Networks**, v. 27, n. 1, p. 39--54, 2005.

ONAN, A. Consensus Clustering-Based Undersampling Approach to Imbalanced Learning. **Scientific Programming**, 2019.

ORRIOLS-PUIG, A.; BERNADÓ-MANSILLA, E. Evolutionary rule-based systems for imbalanced data sets. **Soft Computing**, v. 13, n. 3, p. 213, 2009.

PASS, G.; ZABIH, R.; MILLER, J. **Comparing images using color coherence vectors**. ACM. p 65-73, 1997.

POLA, I.; POLA, F.; ELER, D. Double Distance-Calculation-Pruning for Similarity Search. **Information**, v. 9, n. 5, p. 124, 2018.

PONS, P. A. L. M. Computing communities in large networks using random walks. **Journal of Graph Algorithms and Applications**, v. 10, n. 2, p. 191-218, 2006.

POUZOLS, F. M.; LOPEZ, D. R.; BARROS, A. B. Summarization and analysis of network traffic flow records. *In*: **Mining and control of network traffic by computational intelligence**: Springer, p. 147-189, 2011.

RADEV, D. R.; HOVY, E.; MCKEOWN, K. Introduction to the special issue on summarization. **Computational linguistics**, v. 28, n. 4, p. 399-408, 2002.

RANJAN, R.; GUPTA, S.; VENKATESH, K. Image retrieval using dictionary similarity measure. **Signal, Image and Video Processing**, v. 13, n. 2, p. 313-320, 2019.

RIONDATO, M.; GARCÍA-SORIANO, D.; BONCHI, F. Graph summarization with quality guarantees. **Data mining and knowledge discovery**, v. 31, n. 2, p. 314-349, 2017.

ROBERTSON, S. E. The probability ranking principle in IR. **Journal of documentation**, v. 33, n. 4, p. 294-304, 1977a.

ROBERTSON, S. E. Theories and models in information retrieval. **Journal of documentation**, 33, n. 2, p. 126-148, 1977b.

RUBNER, Y.; TOMASI, C. **Perceptual metrics for image database navigation**. Springer Science & Business Media, 2013.

SALTON, G. A vector space model for information retrieval. **Journal of the ASIS**, p. 613-620, 1975.

SANTINI, S.; JAIN, R. Similarity measures. **IEEE Transactions on pattern analysis and machine Intelligence**, v. 21, n. 9, p. 871-883, 1999.

SANTOS, L. F.; OLIVEIRA, W. D.; FERREIRA, M. R.; CORDEIRO, R. L. *et al.* Evaluating the diversification of similarity query results. **Journal of Information and Data Management**, v. 4, n. 3, p. 188, 2013.

SANTOS, L. F.; OLIVEIRA, W. D.; FERREIRA, M. R.; TRAINA, A. J. *et al.* **Parameter-free and domain-independent similarity search with diversity**. ACM. p. 1-12, 2013.

SHAH, N.; KOUTRA, D.; ZOU, T.; GALLAGHER, B. *et al.* **Timecrunch: Interpretable dynamic graph summarization**. ACM. p. 1055-1064, 2015.

SHAO, Y.; CUI, B.; CHEN, L.; LIU, M. *et al.* An efficient similarity search framework for SimRank over large dynamic graphs. **Proceedings of the VLDB Endowment**, v. 8, n. 8, p. 838-849, 2015.

SHOU, Z.; LI, S. Large dataset summarization with automatic parameter optimization and parallel processing for local outlier detection. **Concurrency and Computation: Practice and Experience**, v. 30, n. 23, p. e4466, 2018.

SILVA, T. C.; ZHAO, L. High-level pattern-based classification via tourist walks in networks. **Information Sciences**, v. 294, p. 109-126, Feb 2015.

SINGH, A.; PUROHIT, A. A survey on methods for solving data imbalance problem for classification. **International Journal of Computer Applications**, v. 127, n. 15, p. 0975-8887, 2015.

SKOPAL, T.; DOHNAL, V.; BATKO, M.; ZEZULA, P. **Distinct nearest neighbors queries for similarity search in very large multimedia databases**. ACM. p. 11-14, 2009.

STANLEY, H. E.; BULDYREV, S. V. Statistical physics: The salesman and the tourist. **Nature**, v. 413, n. 6854, p. 373-374, 2001.

SVORE, K.; VANDERWENDE, L.; BURGES, C. **Enhancing single-document summarization by combining RankNet and third-party sources**, 2007.

TABRIZI, S.A.; SHAKERY, A.; ASADPOUR, M.; ABBASI, M. *et al*. Personalized PageRank Clustering: A graph clustering algorithm based on random walks. **Physica A: Statistical Mechanics and its Applications**, v. 392, n. 22, p. 5772-5785, 2013.

TERCARIOL, C. A. S.; MARTINEZ, A. S. Analytical results for the statistical distribution related to a memoryless deterministic walk: Dimensionality effect and mean-field models. **Physical Review E**, v. 72, n. 2, p. 8, Aug 2005.

TOIVONEN, H.; ZHOU, F.; HARTIKAINEN, A.; HINKKA, A., **Compression of weighted graphs**. ACM. p. 965-973, 2011.

TOMEK, I. Two modifications of CNN. **IEEE Trans. Systems, Man and Cybernetics**, v. 6, p. 769-772, 1976.

TRAINA, A. J.; TRAINA JR, C.; CIFERRI, C. D.; RIBEIRO, M. X. *et al.* How to cope with the performance gap in content-based image retrieval systems. **International Journal of Healthcare Information Systems and Informatics (IJHISI)**, v. 4, n. 1, p. 47-67, 2009.

TSALOUCHIDOU, I.; BONCHI, F.; MORALES, G. D. F.; BAEZA-YATES, R. Scalable dynamic graph summarization. **IEEE Transactions on Knowledge and Data Engineering**, 2018.

ULTSCH, A. U*-matrix: a tool to visualize clusters in high dimensional data, 2003.

VAN LEUKEN, R. H.; GARCIA, L.; OLIVARES, X.; VAN ZWOL, R. **Visual diversification of image search results**. ACM. p. 341-350, 2009.

VANNUCCI, M.; COLLA, V. **Self–Organizing–Maps Based Undersampling for the Classification of Unbalanced Datasets**. IEEE. p. 1-6, 2018.

VEE, E.; SRIVASTAVA, U.; SHANMUGASUNDARAM, J.; BHAT, P. *et al.* **Efficient computation of diverse query results**. IEEE. p. 228-236, 2008.

VIEIRA, M. R.; RAZENTE, H. L.; BARIONI, M. C. N.; HADJIELEFTHERIOU, M. *et al.* On Query Result Diversification. *In*: **Ieee 27th International Conference on Data Engineering**. New York: IEEE, p. 1163-1174, 2011.

VIEIRA, M. R. A. R. H. L. A. B. M. C. N. A. H. M. A. S. D. A. T. C. A. T. V. J. On query result diversification. **Proceedings - International Conference on Data Engineering**, p. 1163-1174, 2011.

VUTTIPITTAYAMONGKOL, P.; ELYAN, E.; PETROVSKI, A.; JAYNE, C. **Overlap-Based Undersampling for Improving Imbalanced Data Classification**. Springer. p. 689-697, 2018.

YAGER, R. R. A new approach to the summarization of data. **Information Sciences**, v. 28, n. 1, p. 69-86, 1982.

YU, C.; LAKSHMANAN, L.; AMER-YAHIA, S. **It takes variety to make a world: diversification in recommender systems**. ACM. p. 368-378, 2009.

YU, C.; LAKSHMANAN, L. V.; AMER-YAHIA, S. **Recommendation diversification using explanations**. IEEE. 1299-1302, 2009.

ZEZULA, P.; AMATO, G.; DOHNAL, V.; BATKO, M. **Similarity search: the metric space approach**. Springer Science & Business Media, 2006.

ZHANG, M.; HURLEY, N. **Avoiding monotony: improving the diversity of recommendation lists**. ACM. p. 123-130, 2008.

ZHANG, T.; RAMAKRISHNAN, R.; LIVNY, M. **BIRCH: an efficient data clustering method for very large databases**. ACM. 103-114, 1996.

ZHENG, K.; WANG, H.; QI, Z.; LI, J. *et al.* A survey of query result diversification. **Knowledge and Information Systems**, v. 51, n. 1, p. 1-36, 2017.

ZIEGLER, C.-N.; MCNEE, S. M.; KONSTAN, J. A.; LAUSEN, G. **Improving recommendation lists through topic diversification**. ACM. 22-32, 2005.