A similarity-based data warehousing environment for medical images

Computers in Biology and Medicine, Oxford, v. 66, p. 190-208, Nov. 2015
http://www.producao.usp.br/handle/BDPI/50729

Downloaded from: Biblioteca Digital da Produção Intelectual - BDPI, Universidade de São Paulo
A similarity-based data warehousing environment for medical images

Jefferson William Teixeira, Luana Peixoto Annibal, Joaquim Cezar Felipe, Ricardo Rodrigues Ciferri, Cristina Dutra de Aguiar Ciferri

1. Introduction

The decision-making process plays an important role in the medical field. Based on summarized and integrated data, specialists (e.g., hospital superintendents, board members, and medical staff leaders) are able to make more strategic and productive decisions concerning the clinical routine. For instance, the analysis of medical information can support efficient planning strategies against disease outbreaks (including the identification of contagious geographical areas or communities), the establishment of vaccination campaigns, the improvement in the vaccination delivery services, the development of educational campaigns, and the analysis of treatment effectiveness.

Data warehousing environments (DWEs) have emerged as a core component for the decision-making process, providing high quality informational data. Through the ETL (extract, transform, load) process, DWEs consolidate large amounts of data of interest from heterogeneous and distributed data sources into a specialized database, the data warehouse (DW). Besides being integrated, data in the DW are subject-oriented, multidimensional, historical and non-volatile [1]. Also, DWs are often implemented in relational databases through a star schema, which is composed of fact and dimension tables. Fact tables store numeric measures of interest while dimension tables contain attributes that contextualize these measures. Queries on such DWEs are called OLAP (on-line analytical processing), and they enable decision-making users to issue analytical queries against the DW without the need of accessing the original data sources [2].

Conventional DWS store only conventional data, such as data of numeric, alphanumeric, and date types. As a result, conventional DWEs only support the ETL process and the OLAP query processing based on conventional data. Conventional data types have a total order relation, and therefore can be sorted and searched using ordinary relational operators (i.e., <, ≤, >, ≥). However, decision-making users cannot use currently conventional DWEs to issue OLAP queries over complex multimedia data, such as images. This is related to the fact that, differently from conventional data, complex data do not have a total order relation, and therefore cannot be sorted and searched using ordinary relational operators.
In Example 1, we introduce a case study that illustrates different analyses that use conventional DWEs in the medical field. We also highlight an analysis that cannot be performed, as it requires the management of image data. This case study is used throughout the paper.

**Example 1.** Suppose a conventional DWE that integrates clinical data related to different modalities, such as breast, head, and knee related problems. These exams were collected over several years, and were carried on patients from different hospitals, belonging to different age groups. Based on the conventional data stored in the DW, specialists can perform the following OLAP analyses: simple trend analysis (e.g. “What is the incidence of breast cancer in 2011 in the southeast region of the US?”), comparative analysis (e.g. “What is the incidence of breast cancer in the last 3 years in the southeast region of the US?”), and multiple trend analysis (e.g. “What is the incidence of breast cancer in the last 3 years in the southeast region of the US, considering different age groups?”).

However, there is a range of queries that cannot be issued against this conventional DWE. For instance, specialists cannot use this application to evaluate the prevalence of certain types of pathology through OLAP queries like: “How many images, similar to a specific breast cancer image, occurred in patients from the southeast region of the US, aged between 30 and 40 years old, in the last 3 years?”. This kind of analysis involves comparisons over images, which are not supported by conventional DWEs.

Managing only conventional data in DWEs has two main drawbacks. First, there is a crescent volume of medical images produced day-by-day in clinics and hospitals, which have to be stored separately from the DW. Second, it is not feasible to use these images in the OLAP query processing to perform similarity search in a tandem with conventional data. This impairs the decision-making process, as specialists cannot base their decisions on important information that can be obtained from images.

The drawbacks motivate the challenge of expanding the storage capacity of DWEs (and their respective ETL and query processing) to support images. Regarding the OLAP query processing, a core challenge is to reduce the “semantic gap” in queries based on similarity search. The semantic gap refers to the difference between the result produced by a computational query and the result expected by specialists. In fact, different specialists may choose different aspects to determine the similarity among a set of images, according to the purpose of the ongoing medical task. That is, they may choose to analyze images according to different feature descriptors, which describe the intrinsic features taken from images (i.e. the images visual patterns) mostly regarding color, texture, and shape. For instance, a given specialist may choose to analyze the texture feature of images using the Haralick descriptors, while another specialist may choose to analyze the color feature of images using Colors Histograms. Thus, the precision of similarity comparisons depends on the task and on the specialist’s perception.

In this paper, we focus on these challenges. We propose imageDWE, a non-conventional DWE that enables the support of medical images in the data warehouse and the processing of OLAP similarity queries over these images. Using our proposal, specialists can perform a new range of interesting analyses, such as that described in Example 1.

We have designed imageDWE to provide major characteristics as follows:

- It defines how images should be stored in the DW. To this end, we extend the star schema design of conventional DWs to also encompass dimension tables specifically designed to store data related to images. Thus, our proposed imageDW is able to store conventional and image data together.
- It reduces the semantic gap in similarity queries. To comply with this goal, we introduce the concept of perceptual layer, which is an abstraction used to represent an image dataset according to a given feature descriptor (e.g. Color Histograms, Haralick) in order to enable similarity search. Each perceptual layer represents a particular specialist’ perception. All perceptual layers of interest are stored in the imageDW.
- It extends the ETL process to manage images. That is, we empower the conventional ETL process to also generate data for the perceptual layers and to store them in the imageDW accordingly.
- It extends the OLAP query process to support similarity queries over images. To comply with this goal, we integrate the conventional OLAP and the similarity search processes.
- It introduces an index technique, which encompasses the specification of an index and the definition of different processing strategies. The technique improves even more the extended OLAP processing of similarity queries over images provided by imageDWE.

We have presented a preliminary version of this study in [7]. Here, we extend that work by allowing the storage of several perceptual layers in the DW, and by extending the conventional ETL process and OLAP query processing to support these perceptual layers. This allows specialists to perform more complex analyses based on different aspects of images. Furthermore, we introduce a novel index technique to support the extended OLAP query processing. Moreover, we describe new performance tests that highlight the advantages of our proposal.

This paper is organized as follows. Section 2 describes the background needed to understand our proposal. Section 3 introduces the proposed imageDWE, which is described in terms of the star schema of the imageDW and the extended ETL and OLAP query processing capabilities. Section 4 introduces the index technique. Section 5 provides a set of guidelines for expanding imageDWE, Section 6 describes the performance evaluation of imageDWE, Section 7 surveys related work, and Section 8 concludes the paper.

## 2. Background

We detail the main issues related to similarity search over images in Section 2.1. Also, our work is based on two well-known concepts available in the literature: the Omni technique and the star-join Bitmap index, which are described in Sections 2.2 and 2.3, respectively.

### 2.1. Similarity search

To be computationally analyzed, images should be pre-processed using through feature extractors, which are responsible for generating feature vectors that describe their intrinsic characteristics. This process is detailed as follows. An image is represented as a two-dimensional \( m \times n \) matrix of pixels, where \( m \) and \( n \) are the image dimensions and the pixel have integer values that depend on the image type. For instance, the pixel values can be 0 or 1 in binary images, vary between 0 and 255 in grayscale images represented by 8 bits, and have three values in the range of 0–255 each in RGB color images.

A feature descriptor is characterized by [9]: (i) a feature extractor algorithm, which tracks down the images, processes their pixel values, produces numeric representations of them, and stores these values in feature vectors; and (ii) a distance function, which produces a similarity measure that is used to determine the
dissimilarity between two images based on their feature vectors [10,11].

The intrinsic characteristics of images are usually described by attributes regarding color, texture and shape. For instance, feature extractors could be implemented to calculate Color Histograms [6], the Haralick descriptors [5], and the Zernike moments [12]. Color histograms represent the distribution of colors (or levels of gray) in an image by calculating the frequency in which the color intensity of each pixel occurs. Regarding texture feature extractors, the Haralick descriptors use statistical approaches to determine co-occurrence matrices of images, such that each matrix represents the relation between the pixel's position and their values. Examples of Haralick descriptors include Uniformity, Variance, Entropy and Homogeneity. Finally, the Zernike moments are shape feature extractors based on complex polynomial functions, which form a complete orthogonal basis set (i.e. a set of functions for which the integral of the product of any pair of functions is zero).

Since images are complex data, their domains are usually represented in the metric space. Formally, a metric space is an ordered pair \((\mathcal{S}, d)\), where \(\mathcal{S}\) is the domain of data elements and \(d: \mathcal{S} \times \mathcal{S} \to \mathbb{R}^+\) is a distance function that, for any \(s_1, s_2, s_3 \in \mathcal{S}\), holds the following properties: (i) identity: \(d(s_1, s_1) = 0\); (ii) symmetry: \(d(s_1, s_2) = d(s_2, s_1)\); (iii) non-negativity: \(d(s_1, s_2) \geq 0\); and (iv) triangular inequality: \(d(s_1, s_3) \leq d(s_1, s_2) + d(s_2, s_3)\) [13]. There are several distance functions available in the literature, but the most widely used are those from the Minkowski family, called \(L_p\). Minkowski distance is typically used with \(p\) being 1 or 2, while \(L_1\) is known as the Manhattan distance, and \(L_2\) is known as the Euclidean distance. In case \(p\) reaches infinity, \(L_{\infty}\) is known as the Chebyshev distance.

For a given distance function, the distance becomes smaller as two images under comparison become more similar, thus enabling the execution of similarity search. An important type of similarity search over images is the range query, which is de
termined in terms of two representative images \(f_1\) and \(f_2\), for a range query in which \(s_q\) is the query image, \(r_q\) is the query radius and each \(d_i\) (\(i = 1\) or \(i = 2\)) is the distance between \(f_i\) and \(s_q\). A range query retrieves every element \(s_i \in \mathcal{S}\) that satisfies the condition \(d(s_i, s_q) \leq r_q\). An example is: “Select the images similar to a specific breast cancer image by up to five similarity units”.

2.2. The Omni technique

The Omni technique [14] is based on the selection of representative images (i.e. foci) from the dataset, which are strategically positioned in the metric space. The number of representative images is given by the intrinsic dimensionality of the dataset, while the positioning is given by the Hull of Foci algorithm. The main idea behind this algorithm is to choose images close to the dataset’s borders, which can be used to improve prunability in similarity search.

Fig. 1(a) depicts a visual representation of the metric space, where \(f_1\) and \(f_2\) are two representative images. Each representative image generates a ring in the metric space that surrounds the query image \(s_q\) given the query radius \(r_q\) and the distance between this representative image and \(s_q\). For instance, \(f_1\) generates the ring illustrated in dashed lines using as inferior border the value of \(d_1 - r_q\) and as superior border the value of \(d_1 + r_q\). The intersection of the rings produces a region called minimum-bounding-Omni-region (mbOr), which restricts the set of images that will be submitted to similarity comparison in respect to \(s_q\). The greater the number of representative images, the lower the number of images to be further analyzed. Fig. 1(b) shows the mbOr defined by three representative images. Clearly, the intersection of the three generated rings reduces the number of images to be compared.

The mbOr may contain false positive images. Thus, the refinement phase of the Omni technique calculates the distances between the query image and the images inside the mbOr, returning only the ones inside the region established by the query radius \(r_q\), which is illustrated by the dashed circle in Fig. 1. Note that this region is always inside the mbOr, ensuring that all images similar to the query image are always inside the mbOr.

2.3. The star-join Bitmap index

A basic Bitmap index [15] built on an attribute \(A\) consists of several bit-vectors, one for each value \(a\) of \(A\), where the \(i\)-th bit is 1 if the \(i\)-th row is equal to \(a\), otherwise, the \(i\)-th bit is 0. In DWEs, a Bitmap index can be constructed to index attributes of dimension tables indicating the set of tuples in a fact table that hold the corresponding attribute values. A Bitmap index with such design is called star-join Bitmap (SJB) index [16].

Fig. 2 depicts a fragment of data from a conventional DW star schema (Fig. 2(a)–(c)) and a SJB index defined over the attribute category of the dimension table Age (Fig. 2(d)). There are two bit-vectors composing the SJB index, one for each value of the attribute category of the dimension table Age. Also, the SJB index and the fact table have the same number of rows. As a result, a value of 1 in the first row of the index regarding the bit-vector child indicates that the first row of the fact table refers to the ageKey equal to 1.

The advantage of the SJB index is that the query processing of the conventional predicate can be performed quickly by the processing of bitwise logical operations (e.g. AND, OR, and XOR). Therefore, it is widely used to speed up OLAP queries issued over huge volumes of data. However, a SJB index constructed on high-cardinality attributes may lead to high costs of storage and performance losses caused by the extremely large and sparse generated bit-vectors. These problems can be surpassed by three different optimization techniques, named binning [17,18], compression [19,20] and encoding [21,22]. In this paper, we are especially
interested in the binning and encoding techniques. The first one creates an index over \( n \) predefined bins, covering multiple values of an attribute \( A \). This technique is mainly used for attributes with a large domain (e.g., floating point numbers). The second technique refers to the way in which the bit-vector is created. In its simplest form, the index is equality encoded, that is, a bit-vector refers to exactly one attribute value. However, we can reduce the size of the index by choosing different simple encoding techniques, such as binary, interval, range, or more complex ones, such as interval-equality and range-equality encoding.

3. The proposed imageDWE

In this paper, we introduce imageDWE, an image DWE aimed to perform OLAP similarity queries over medical images. The general architecture of imageDWE is depicted in Fig. 3. The data sources may contain conventional data (e.g., relational database with age data and text files for dates) and images (e.g., images repository). The extended ETL process is responsible for extracting conventional and image data of interest from these sources and storing them in the imageDW. It is also responsible for storing metadata resulting from the extraction and transformation processes in the
metadata repository. The imageDW is a star schema specially designed to store conventional and image data. In Fig. 3, it is composed of the dimension tables Age and Date, one or more dimension tables designed for medical image data, which are illustrated by the medical image pictures, and the fact table Exam. The extended OLAP query processing is responsible for efficiently processing OLAP similarity queries over images. We detail each component of imageDWE from Sections 3.1 to 3.3, focusing on the manipulation of images. Details regarding the manipulation of conventional data are out of the scope of this paper.

3.1. The imageDW

We designed the imageDW as a star schema composed of a fact table, conventional dimension tables and image dimension tables. We also designed the imageDW to support several perceptual layers. Before detailing the characteristics of the tables, we introduce the concept of perceptual layer.

A perceptual layer is an abstraction used to represent an image dataset according to a given feature descriptor in order to enable similarity search. In detail, a perceptual layer is an abstraction that represents images through their feature vectors, which were generated according to a specific feature descriptor (e.g. Color Histograms), and their similarity search data, which were generated according to a specific metric space. That is, each perceptual layer refers to a specific feature extractor that implements a given feature descriptor and, therefore, encompasses the generated feature vectors of all images to be stored in the imageDW according to this feature descriptor. Further, each perceptual layer refers to a specific metric space and, therefore, also encompasses the representative images of this metric space and the distances between each representative image and each image feature vector. Both representative images and distances are defined using the Omni technique. Note that any feature descriptor can be used to represent an image dataset to be stored in the imageDW, as long as this feature descriptor is implemented by a feature extractor that generates feature vectors and allows the specification of a distance function to determine the dissimilarity between two images based on the generated feature vectors.

We are now able to define the characteristics of each type of table that compose the imageDW, described as follows:

- The fact table is visually represented in the center of the star schema and contains numeric measures and foreign keys to the dimension tables where conventional and image data are stored. It preserves the relationship between each image and its respective conventional data.
- Conventional dimension tables are visually represented around the fact table, and store a primary key and several descriptive attributes.
- Image dimension tables are also visually represented around the fact table. They store intrinsic features of images and support the specification of different perceptual layers. The image dimension tables are composed of a primary image dimension table that stores a primary key and one or more feature vectors and, for each perceptual layer, a secondary image dimension table that contains a primary key and a set of attributes for similarity search. For a given image, these attributes represent the distances between the image and the representative images according to the Omni technique. Note that the image dimension tables store the representation of images through their intrinsic features, instead of storing the images themselves (i.e. matrices of pixels or images in the DICOM format).

The image dimensions are the new dimensions that we are proposing in this paper and must be used in DWEs to allow OLAP
similarity queries over images. The remaining dimensions represent conventional dimensions and can be changed according to the target DWE.

Furthermore, the imageDW allows the specification of one or more perceptual layer. In the most restrict scenario, i.e. there is only one perceptual layer, the imageDW is composed of a primary image dimension table containing a primary key and one feature vector and only one secondary dimension table containing a primary key and a set of attributes for similarity search. But, the greater the number of perceptual layers, the lower the semantic gap. That is, the definition of more than one perceptual layer extends the concept of multidimensionality to the perspective of the intrinsic features of images, allowing the reduction of the semantic gap in DWEs. As a result, specialists have more flexibility to choose the most appropriate feature descriptor, or a combination of different feature descriptors, to employ in their OLAP queries.

In Example 2, we illustrate the star schema of an imageDW that stores data related to the medical field using as a basis for the case study introduced in Example 1.

**Example 2.** The star schema depicted in Fig. 4 is composed of the fact table Exam, the conventional dimension tables Age, Patient, ExamDate, Hospital, and ExamDescription, the primary image dimension table FeatVec, and the secondary dimension tables ColorHistogram, Zernike, and Haralick. The primary keys of the tables are represented by the abbreviation PK while the foreign keys of the fact table are represented by the abbreviation FK.

There are three perceptual layers, called ColorHistogram, Zernike, and Haralick. For each perceptual layer, there is a specific attribute in the primary image dimension table FeatVec that stores the respective feature vector and a correspondent secondary image dimension table that stores the set of attributes for similarity search. For instance, the attribute featVecColorHistogram of the table FeatVec stores the feature vector of an image according to the Color Histogram extractor. The attributes CHDistRep_1, CHDistRep_2 and CHDistRep_3 of the table ColorHistogram store the distances of the image to the first, second and third representative images, respectively. The quantity of attributes for similarity search depends on the Omni technique. For instance, the ColorHistogram and the Zernike perceptual layers have three representative images, while the Haralick perceptual layer has four representative images.

The conventional dimension tables store conventional data related to ages of patients (Age), patients (Patient), dates of exams (Date), hospitals (Hospital), and descriptions of exams (ExamDescription). For instance, conventional data related to an age of a patient are: his/her age (e.g. 73 years old), its classification interval (e.g. more than 60 years), and its category (e.g. elderly).

The fact table Exam associates each image to its conventional data related to hospitals, patients, exams, ages of patients, and dates. Also, this table is a factless fact table since it contains as numeric measure an artificial attribute that is always populated with the value of 1, i.e. the attribute amount. This allows the fact table to keep its nature of being an aggregative entity that supports SUM operations.

### 3.2. The extended ETL process

We developed the extended ETL process of imageDWE to also support the extraction and storage of features from images in addition to the tasks of extraction, transformation, and loading of conventional data offered by conventional DWEs. The management of an image in the extended ETL process is performed in three sequential tasks, as depicted in Fig. 5: extraction of features (Fig. 5(a)), definition of representative images (Fig. 5(b)), and calculation of distances (Fig. 5(c)).

The proposed three tasks are detailed as follows. In task (a), all the input images are processed by one or more feature extractors, each of them used to generate a given perceptual layer. A feature extractor may be any extractor already proposed in the literature, such as color histograms, the Haralick descriptors, and the Zernike moments. The outputs of this task are the feature vectors of the input images. That is, for each image, task (a) generates one feature vector for each perceptual layer. The generated feature vectors are inputs for the other two tasks, and are also stored in the primary image dimension table of the imageDWE.

In task (b), the representative images of each perceptual layer are determined using the Hull of Foci algorithm of the Omni technique. Note that this task can be executed only after all the images to be stored in the imageDWE have their feature vectors extracted by task (a). In task (b), the feature vectors of the representative images are stored in the metadata repository, together with the quantity of perceptual layers and additional metadata listed as follows. For each perceptual layer, we store the dimensionality of the feature vector and the diameter of the dataset. The diameter is used to determine the query radius, which is given in terms of the percentage of the dataset’s radius.

Finally, task (c) is responsible for calculating the distances between the image and each representative image of each perceptual layer, based on a distance function. A distance function may be any distance already proposed in the literature, such as the distances from the Minkowski family. For each perceptual layer, its corresponding calculated distances are stored as attributes for

![Fig. 5. The proposed extended ETL process.](image-url)
similarity search in a specific secondary image dimension table of the imageDW.

3.3. The extended OLAP query processing

OLAP similarity queries over images encompass conventional and similarity search predicates. While a conventional predicate refers to query conditions defined over the conventional attributes by using ordinary relational operators, a similarity search predicate refers to similarity conditions defined over the intrinsic features of images. Furthermore, the similarity search predicate supported by our proposed method also encompasses the specification of one or more perceptual layers. As a result, we developed the extended OLAP query processing of imageDW to carry out the conventional predicate and the similarity search predicate that deals with different perceptual layers.

The extended OLAP query processing is performed in up to six tasks, as depicted in Fig. 6: selection by conventional predicate (Fig. 6(a)), extraction of features (Fig. 6(b)), calculation of distances (Fig. 6(c)), filtering by distances of representative images (Fig. 6(d)), refinement by distances of selected images (Fig. 6(e)), and calculation of the answer (Fig. 6(f)).

The proposed six tasks are detailed as follows. In task (a), conventional data associated to the stored images are filtered according to the conventional predicate of the query. The goal of this task is to reduce the number of images for which the similarity search must be calculated, thus improving the query processing performance. If the conventional data do not satisfy the conventional query predicate, tasks (b), (c), (d), and (e) are not executed, and task (f) returns the value zero as the query answer. Otherwise, task (a) generates as output a set of selected images, i.e., those images whose conventional data satisfy the conventional query predicate.

The query processing continues with the extraction of the intrinsic features of the query image by task (b). For each perceptual layer specified in the query, this task employs a specific feature extractor, thus generating as output a set of feature vectors for the query image. Note that task (b) uses the same feature extractors used by the extended ETL process and the extraction of features from the query image is only executed for the perceptual layers involved in the query. Otherwise, the final query result would be meaningless.

Using the feature vectors of the query image and the feature vectors of the representative images recovered from the metadata repository, task (c) calculates the distances between the query image and each representative image of each perceptual layer. Similar to task (b), task (c) uses the same distance function employed in the extended ETL process. The resulting distances are the output of task (c).

In task (d), the selected images generated in task (a) are further filtered according to their similarity search predicates. To this end, task (d) uses the distances calculated in task (c) and determines the mbOr region of the Omni technique, selecting those images that are inside the mbOr. This task also reduces the number of images for which the similarity search must be calculated, improving even more the query processing performance. The similar images generated as output of task (d) satisfy the perceptual layers defined in the query. That is, this task analyzes the images through the intersection of the mbOr defined for each perceptual layer individually. If there are no selected images, task (e) is not executed, and task (f) returns the value zero as the query answer.

The selected images produced by task (d) are then processed by task (e), which compromises the refinement phase of the Omni technique. The generated output, i.e., similar images, are the selected images that are not false positives.

Finally, task (f) uses as input the similar images to calculate the query answer. That is, task (f) applies an aggregation function (e.g., SUM or AVG) to calculate the query answer.

To illustrate the proposed extended OLAP query processing, we introduce in Example 3 an OLAP similarity query over images defined by a specialist considering the imageDW described in Example 2.

Example 3. Consider the following query defined by a specialist. “How many images are similar to a given mammography image (i.e. the query image), according to the ColorHistogram and Zernike perceptual layers and within the query radius rQ, that occurred in patients from the New York state, aged between 30 and 40 years, in the years 1993 to 1994, and in the hospitals from the macro-region of the New York state?”

First, task (a) processes the conventional data of the images stored in the imageDW to evaluate the conventional predicate of the query. These images are filtered according to the year in which they were generated, to the macro-region where the hospitals were located, to the states where the patients were from and to their age on the exam dates. Such data is fetched from the conventional dimension tables Date, Hospital, Patient and Age, respectively. As a result of task (a), only images that occurred from 1993 to 1994 in hospitals from the macro-region of the New York...
state, considering patients from the New York state with ages between 30 and 40 years are selected to be further processed.

The query image is processed in task (b), which applies the ColorHistogram and the Zernike feature extractors, as specified in the specialist’s query. Two feature vectors are generated. They are then used together with the feature vectors of the representative images of the perceptual layers ColorHistogram and Zernike to calculate six distances in task (c). That is, task (c) calculates three distances for each perceptual layer, as both ColorHistogram and Zernike have three representative images.

Task (d) generates two mbOr regions, one for the Color-Histogram perceptual layer and the other for the Zernike perceptual layer. Both regions are defined based on three representative images, as illustrated in Fig. 1(b). Only the images that are present in the two regions are selected to be further processed by task (e). In this task, images that are false positives are eliminated.

Finally, task (f) counts the number of images resulting from task (e), and returns this number to the specialist.\(^\text{\textsuperscript{\textcircled{2}}}\)

4. The proposed index technique to support the extended OLAP query processing over images

In this section, we introduce a novel index technique aimed to improve the extended OLAP query processing provided by imageDWE. The proposed technique encompasses the specification of an index, called imageSJBindex, which data structure is described in Section 4.1 and its use in the extended OLAP query processing in Section 4.2. It also encompasses the definition of different processing strategies, which are detailed in Section 4.3.

4.1. Data structure of the imageSJBindex

The imageSJBindex is an SJB index that indexes the distances between the stored images and the representative images. All distances related to all perceptual layers are indexed in the same index structure. Conceptually, the imageSJBindex is a two-dimensional binary \(m \times n\) matrix where \(m\) is the number of bit-vectors such that there are one or more bit-vectors for each distance attribute related to each perceptual layer and \(n\) is the number of tuples in the fact table. Example 4 depicts how the imageSJBindex looks like conceptually for a sample imageDW.

**Example 4.** Fig. 7 depicts the imageSJBindex that indexes the ColorHistogram, Zernike, and Haralick perceptual layers of the star schema represented in Fig. 4 using the binning technique. For simplicity, we consider that the domain of all possible distance values is given by the interval \([0,8]\) and create three bins with the intervals \([0,2]\), \([3,5]\), and \([6,8]\). Also, we organize all perceptual layers using the same binning strategy, although different binning intervals \([0,2]\), \([3,5]\), and \([6,8]\). For instance, given that \(CHDistRep\_1\) and \(CHDistRep\_2\) are 1, we have the interval \([2-1, 2+1]=\{1, 3\}\).

First, improved task (d) performs the intersection among the bins indexed by the imageSJBindex illustrated in Fig. 7 and the intervals described in Fig. 8(a) and (b). The selected bins from the index are depicted in Fig. 8(c). For instance, consider the bins of CHDistRep\_1 from the ColorHistogram perceptual layer (i.e. \([0,2]\), \([3,5]\), \([6,8]\)) illustrated in Fig. 7 and the interval calculated for the query image regarding CHDistRep\_1 (i.e. \([1,3]\)) illustrated in Fig. 8(a). The retrieved bins from the imageSJBindex for CHDistRep\_1 are \([0,2]\) and \([3,5]\). Second, since two bins are selected for CHDistRep\_1 and CHDistRep\_2 of the ColorHistogram perceptual layer, improved

<table>
<thead>
<tr>
<th># tuple</th>
<th>Color Histogram</th>
<th>Zernike</th>
<th>Haralick</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CHDistRep_1</td>
<td>CHDistRep_2</td>
<td>CHDistRep_3</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

4.2. Improved task (d): improved task of filtering by distances of representative images

The use of the imageSJBindex modifies task (d) depicted in Fig. 6, i.e. the task of filtering by distances of representative images. As the imageSJBindex indexes the distances between the stored images and the representative images, there is no need to access the imageDW to perform task (d) as proposed in Section 3.3. Instead, improved task (d) filters the images by the similarity search predicate using the imageSJBindex.

Improved task (d) uses the imageSJBindex to select the images inside the mbOr through logical bitwise operations, described as follows. First, the task selects, for each distance of each perceptual layer, the bins of the imageSJBindex whose range of values intersect the intervals calculated for the query image. Second, for each distance of each perceptual layer with more than one selected bin, the task applies the OR operation over them. The result is one bit-vector for each distance attribute representing each tuple in the imageDW. Third, the task applies the AND operation over the resulting bit-vectors, thus performing the intersection of all mbOr related to the perceptual layers involved in the query. A given image is selected when its respective tuple in the resulting bit-vector is equal to 1. **Example 5** illustrates how the improved task (d) works for the query described in Example 3.

**Example 5.** Consider the similarity search predicate of the specialist’s query described in Example 3, i.e. “How many images are similar to a given mammography image (i.e. the query image), according to the ColorHistogram and Zernike perceptual layers and within the query radius \(r_q=1\)?”. For the sake of simplicity, we left the conventional predicate out of this example. Consider also that the task of calculation of distances (Fig. 6(c)) produced the distances depicted in Fig. 8(a) and (b) for the two perceptual layers specified in the query. In Fig. 8(a), we use three intervals \([1,3]\), \([4,6]\), and \([7,9]\), which represent the rings around each representative image of the ColorHistogram perceptual layer. In Fig. 8(b), we use three intervals \([0,2]\), \([3,5]\), and \([6,8]\), which represent the rings around each representative image of the Zernike perceptual layer. The intervals are calculated based on the value of the query radius. For instance, given that CHDistRep\_1=2 and \(r_q=1\), we have the interval \([2-1, 2+1]=\{1, 3\}\).
task (d) applies the OR operation over them, as depicted in Fig. 8 (d). In the sequence, for each row of the resulting bit-vector, the improved task (d) applies the AND operation, as illustrated in Fig. 8(e). In this example, the selected image is the one that corresponds to the second tuple of the fact table Exam. Note that we detail here only the task of filtering by distances of representative images. Therefore, this is not the final answer of the query.

**4.3. Processing strategies**

The proposed processing strategies aim to enable the use of the `imagesSJBindex` to process OLAP similarity queries over images. To comply with this goal, we define the processing strategies based on the following aspects: (i) optional indexing of conventional data and feature vectors using the `imagesSJBindex`, and (ii) different orders for analyzing conventional and similarity search predicates using the `imagesSJBindex`. Before describing the proposed processing strategies, we detail the concepts behind these aspects.

We define two different approaches to determine the order for analyzing conventional and similarity search predicates using the `imagesSJBindex`: split and combined. As detailed in Section 3.3, the original extended OLAP query processing first analyzes the conventional predicate in task (a) and then the similarity search predicate in task (e). On the other hand, the split approach first filters the images by the similarity search predicate using the `imagesSJBindex` and then filters the images by the conventional predicate applying task (a). As for the combined approach, it filters the conventional and the similarity search predicates together using the `imagesSJBindex`. Thus, the combined approach requires the conventional predicate to be also indexed by the `imagesSJBindex`. The combined approach was designed to efficiently process queries that have conventional and similarity search predicates while the split approach was designed to provide better support to queries that have only the similarity search predicate, although this approach is also able to process queries that have conventional and similarity search predicates.

Furthermore, we define two different approaches to determine whether the feature vectors should also be indexed by the `imagesSJBindex`: to index feature vectors and not to index feature vectors. The original extended OLAP query processing needs to access the `imageDW` to retrieve the feature vectors of the output images from task (d) to perform the refinement by distances of selected images in task (e). According to the to index feature vectors approach, the feature vectors of the stored images are also indexed by the `imagesSJBindex`. Therefore, in the query processing, the index can be used to retrieve the feature vectors without accessing the `imageDW`. In contrast, the not to index feature vectors approach does...
not index the feature vectors, so it needs to retrieve these data from the imageDW. The to index feature vectors approach was designed to be used only with feature vectors that have low dimensionality, due to the high costs in query processing related to dimensionality issues. The not to index feature vectors approach, on the other hand, focuses on feature vectors that have high dimensionality.

The aforementioned approaches are orthogonal. As a consequence, we propose four different processing strategies by coupling these approaches. We base their description on the tasks introduced in Section 3.3 and on the improved task (d) described in Section 4.2. Furthermore, we consider that there is an additional task (g), which is aimed to create the imagesJBindex. Task (g) should be performed after the imageDW has been populated and before the execution of the proposed processing strategies. The strategies are described as follows:

- **splitNotIndexFV**, coupling the split and not to index feature vectors approaches. This strategy filters the images first by the similarity search predicate and then by the conventional predicate. It considers that the imagesJBindex indexes only the distances between the stored images and the representative images. Fig. 9 depicts the processing order of the splitNotIndexFV strategy. As this strategy does not index the feature vectors of the stored images, they are recovered from the imageDW by task (e) during query processing.

- **splitIndexFV**, coupling the split and to index feature vectors approaches. This strategy filters the images first by the similarity search predicate and then by the conventional predicate. Here, in addition to the distances between the stored images and the representative images, the imagesJBindex also indexes the feature vectors of the stored images. Fig. 10 depicts the processing order of the splitNotIndexFV strategy. Note that task (e) does not need as inputs the feature vectors of the stored images, as they are already indexed by the imagesJBindex.

- **combinedNotIndexFV**, coupling the combined and not to index feature vectors approaches. This strategy filters the conventional and the similarity search predicates together. Note that in this strategy, in addition to the distances between the stored images and the representative images, the imagesJBindex also indexes conventional data. Fig. 11 depicts the processing order of the combinedNotIndexFV strategy. Similar to the splitNotIndexFV strategy, the combinedNotIndexFV strategy does not index the feature vectors of the stored images, which are recovered from the imageDW by task (e) during query processing.

- **combinedIndexFV**, coupling the combined and to index feature vectors approaches. This strategy filters the conventional and the similarity search predicates together. Here, the imagesJBindex indexes all types of data stored in the imageDW: the feature vectors of the stored images, conventional data and the distances between the stored images and the representative images. Fig. 12 depicts the processing order of the combinedIndexFV strategy. Similar to the splitIndexFV strategy, the combinedIndexFV strategy indexes the feature vectors of the stored images, which do not need to be recovered from the imageDW.

Figs. 9–12 visually highlight the differences among the proposed strategies. Also, Table 1 summarizes these differences. The
splitNotIndexFV strategy indexes only the similarity search predicates using the imageSJBindex. As a result, the imageDW is accessed to perform tasks (a) and (e), as depicted in Fig. 9. The splitIndexFV strategy retrieves the feature vectors together with the similarity search predicates using the imageSJBindex, allowing task (e) to be executed without accessing the imageDW, as shown in Fig. 10. The combinedNotIndexFV and the combinedIndexFV strategies, depicted respectively in Figs. 11 and 12, index conventional and similarity search predicates using the imageSJBindex, guaranteeing that all filtering processes to be executed by task (a) and improved task (d) without accessing the imageDW. While in the combinedNotIndexFV strategy the imageDW is still accessed to retrieve the feature vectors by task (e), this is not the case in the combinedIndexFV strategy.

5. Expanding the proposed imageDWE

In this section, we provide guidelines for expanding imageDWE considering two different scenarios. The first scenario regards the addition of new perceptual layers to imageDWE and is described in Section 5.1. The second scenario regards the coupling of imageDWE to an existing conventional DWE and is discussed in Section 5.2.

5.1. Adding new perceptual layers to imageDWE

A perceptual layer is an abstraction that groups image data according to the feature descriptor that generates them. Therefore, adding a new perceptual layer into imageDWE encompasses improving the design of the star schema, the functionalities of the extended ETL process and the OLAP query processing, and the support provided by the index technique.

The guidelines for adding a new perceptual layer are detailed as follows:

**Guideline 1.** The star schema of the imageDW described in Section 3.1 should be extended to store new attributes and a new dimension table.

Extend the star schema of the imageDW to include:

1. A new attribute in the primary image dimension table. For a given image, this attribute stores the feature vector of the image generated by the new feature descriptor. For instance, consider the star schema depicted in Fig. 4 and a new perceptual layer related to the Fourier descriptors [23]. The new attribute featVecFourier should be added to the primary image dimension table FeatVec.

2. A new secondary image dimension table, which contains a primary key and \( n \) attributes for similarity search. For a given image, these attributes represent the distances between the image and the representative images according to the Omni technique. Note that the value of \( n \) should be determined according to Guideline 2.2. For instance, the new secondary image dimension table Fourier should be created, which should be composed of the attributes foKey (PK), foDistRep_1, foDistRep_2, ..., foDistRep_n.

3. A new attribute in the fact table, which represents the foreign key to the new secondary image dimension table. For instance, the new attribute e_foKey should be added to the fact table Exam.

Fig. 11. Processing order for the combinedNotIndexFV strategy.

Fig. 12. Processing order for the combinedIndexFV strategy.
Guideline 2. The extended ETL process described in Section 3.2 should be improved to couple with the new perceptual layer.

Add new functionalities to the tasks of the extended ETL process:

1. Implement the new feature extractor in task (a). For each image obtained from the data source, use the new feature extractor to populate the new attribute created according to Guideline 1.1.
2. Determine the number \( n \) of representative images of the new perceptual layer by calculating the intrinsic dimensionality of the dataset. Also, use the Hull of Foci algorithm of the Omni technique to determine the \( n \) representative images in task (b). Note that \( n \) corresponds to the \( n \) attributes for similarity search created according to Guideline 1.2.
3. For a given image, use a distance function to calculate the distances between the image and each representative image of the new perceptual layer in task (c). Populate the attributes for similarity search created according to Guideline 1.2 with these calculated distances.
4. Update the metadata repository to add one quantity to the total of perceptual layers and to store the feature vectors of the representative images, the dimensionality of the feature vector, and the diameter of the dataset.

Guideline 3. The extended OLAP query processing described in Section 3.3 should recognize the new perceptual layer.

After performing the changes depicted in Guidelines 1 and 2, all information required to process a given OLAP similarity query that specifies the new perceptual layer is stored in \( \text{imageDWE} \). As a result, the extended OLAP query processing should

1. Use the new feature extractor implemented according to Guideline 2.1 to extract the feature vectors of the query image in task (b).
2. Answer the query considering all image data and metadata related to the new perceptual layer.

Guideline 4. The processing strategies described in Section 4.3 should recognize the new perceptual layer.

After performing the changes depicted in Guidelines 1–3, all information required to use the index technique should have been made and available in \( \text{imageDWE} \). As a result, the index technique should

1. Recreate the \( \text{imageSJBindex} \) to also index data from the new perceptual layer. For instance, the \( \text{imageSJBindex} \) depicted in Fig. 7 should have \( n \) additional columns since the new perceptual layer is represented by \( n \) representative images and the \( \text{imageSJBindex} \) of this example uses three bins with the intervals \([0,2] \), \([3,5] \), and \([6,8] \) to represent each representative image.
2. Answer the query considering the \( \text{imageSJBindex} \) created according to Guideline 4.1 and all image data and metadata related to the new perceptual layer.

5.2. Coupling imageDWE with an existing conventional DWE

Consider an existing conventional DWE implemented in a relational database. Consider also that the DW of this conventional DWE is designed according to the star schema, and contains a fact table and several conventional dimension tables modeled according to the target medical application. Furthermore, consider an existing repository that stores digital medical images from exams of several modalities. Coupling \( \text{imageDWE} \) with these existing components requires performing the actions described in Guidelines 1–4 introduced in Section 5.1 according to the order established by Algorithms 1 and 2. While Algorithm 1 refers to guidelines related to the ETL process, Algorithm 2 refers to guidelines related to the OLAP query processing.

Algorithm 1. CouplingETL.

1. define the perceptual layers to be added to \( \text{imageDWE} \);
2. for each perceptual layer do
   3. \( \text{// perform the actions described in Guidelines 1 and 2; } \)
   4. end
3. create the \( \text{imageSJBindex} \) to index all perceptual layers;

Algorithm 2. CouplingOLAP.

1. for each perceptual layer defined in a given OLAP similarity query issued against the \( \text{imageDWE} \) do
   2. \( \text{// perform the action described in Guideline 3.1; } \)
   3. end
4. if data is indexed by the \( \text{imageSJBindex} \) then
   5. \( \text{// follow Guideline 4.2 to perform one of the query processing strategies described in Section 4.3; } \)
6. else
7. \( \text{// follow Guideline 3.2 to perform the extended OLAP query processing described in Section 3.3; } \)
8. end

6. Performance evaluation

The proposed \( \text{imageDWE} \) was assessed through performance tests using as a basis a DWE that consolidates medical images of exams of several modalities, such as brain, breast, and chest related problems. The general goal of the performance evaluation was to demonstrate the feasibility and efficiency of \( \text{imageDWE} \) to store images in the \( \text{imageDW} \) and to process OLAP similarity queries over them.

Specifically regarding query processing, we analyzed the different strategies introduced in this paper in order to indicate which is the best strategy to process a given OLAP similarity query over images issued by a specialist according to the characteristics of the query. That is, in this paper, we introduced five query processing strategies, described as follows: (i) the extended OLAP

---

Table 1
Comparisons among the proposed strategies.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Indexed data</th>
<th>Access the imageDW</th>
<th>Processing order of tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Similarity search predicate</td>
<td>Conventional predicate</td>
<td>Feature vector</td>
</tr>
<tr>
<td>splitNotIndexFV</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>splitIndexFV</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>combinedNotIndexFV</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>combinedIndexFV</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

---
query processing, hereafter named as the basic strategy; (ii) the splitNotIndexFV strategy; (iii) the splitIndexFV strategy; (iv) the combinedNotIndexFV strategy; and the combinedIndexFV strategy. The first strategy was described in Section 3.3, and the last four strategies were detailed in Section 4.3. Depending on the characteristics of the OLAP similarity query over images, one processing strategy is able to provide the better performance results over the others.

Indeed, the performance results described in this section lead to some relevant points:

- The splitNotIndexFV and the splitIndexFV strategies were the most appropriated ones to process OLAP similarity queries over images that encompass only a similarity search predicate. While the splitNotIndexFV strategy should be used with feature vectors that have high dimensionality, the splitIndexFV should be used with feature vectors that have low dimensionality. These findings are described in Section 6.2.

- The combinedNotIndexFV and the combinedIndexFV strategies were the most appropriate ones to process OLAP similarity queries over images that encompass conventional and similarity search predicates, such that the conventional predicate is less selective. While the combinedNotIndexFV strategy should be used with feature vectors that have high dimensionality, the combinedIndexFV should be used with feature vectors that have low dimensionality. These findings are described in Section 6.3.

- The basic strategy was the most appropriate one to process OLAP similarity queries over images that encompass conventional and similarity search predicates such that the conventional predicate is more selective. These findings are described in Section 6.3.

The selectivity of a predicate refers to the ratio between the number of tuples from a table that matches the predicate and the total number of tuples of this table. A less selective predicate returns a smaller number of tuples. On the other hand, a more selective predicate returns a smaller number of tuples.

Section 6.1 describes the characteristics of the experiments and Sections 6.2 and 6.3 detail the performance results that support our findings. Furthermore, in order to provide an evaluation of our proposal from specialists, we conducted a study that investigated the feasibility of the proposed imageDWE from their point of view. Section 6.4 describes this study.

6.1. Experimental setup

The imageDWE was modeled according to the star schema introduced in Example 2. We designed our imageDWE with five perceptual layers, each one representing a specific feature extractor, detailed as follows: (i) Color Histogram; (ii) Haralick Variance; (iii) Haralick Entropy; (iv) Haralick Uniformity; and (v) Haralick Homogeneity. Table 2 describes the characteristics of each perceptual layer, regarding the dimensionality of the feature vector, the number of representative images and the diameter of the dataset. These data are stored in the metadata repository of the proposed imageDWE (Section 3.2).

We populated the conventional dimension table Hospital with real data from the Brazilian health care system available at www2.datasus.gov.br/datasus/index.php and the conventional dimension table Date with real data containing days from 1992 to 2012. Furthermore, we populated the image dimension tables and the attribute bodyPart of the conventional dimension table ExamDescription using real images from a Brazilian public hospital. We generated synthetic data to populate the remaining tables of the imageDWE, mainly due to privacy issues. The number of tuples stored in each table of the imageDWE was: Age: 121 tuples; Patient: 100,067 tuples; Date: 7677 tuples; Hospital: 645 tuples; ExamDescription: 1,409,848 tuples; Feature: 1,409,848 tuples; Color Histogram: 1,409,848 tuples; Haralick Variance: 1,409,848 tuples; Haralick Entropy: 1,409,848 tuples; Haralick Uniformity: 1,409,848 tuples; Haralick Homogeneity: 1,409,848 tuples; and Exam: 1,409,848 tuples. The total volume of the imageDWE was about 1.65 GB.

We created the imagesSBIndex using the interval encoding and the binning technique with 5 bins. We found experimentally using the FastBit software [24] that the interval encoding provided better performance results when compared to the interval-equality encoding and the binary encoding for indexing the current imageDWE. We also found experimentally that 5 bins is the best value among the values of 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50 bins for the current imageDWE.

In order to investigate the performance of imageDWE with regard to the similarity search processing over images, we defined four classes of queries that imposed distinct costs to the query processing. These classes are based on different numbers of perceptual layers and require the management of different sizes of feature vectors. They are detailed as follows:

- Class 1: queries of this class are defined in terms of only one perceptual layer that requires the management of feature vectors with low dimensionality.
- Class 2: queries of this class are defined in terms of two or more perceptual layers that require the management of feature vectors with low dimensionality.
- Class 3: queries of this class are defined in terms of only one perceptual layer that requires the management of feature vectors with high dimensionality.
- Class 4: queries of this class are defined in terms of two or more perceptual layers that require the management of feature vectors with low and high dimensionalities.

Regarding hardware and software, the experiments were performed on a computer with an Intel Core i7-4770 3.4 GHz processor, 32 GB of main memory, 2 TB hard disk, Linux CentOS 6.5 (64 bits), and PostgreSQL 8.3. We used the C++ programming language to implement the extended ETL algorithms and the environment for the query processing. We employed FastBit version 1.3.8 as the Bitmap software to implement the imagesSBIndex to process the conventional and the similarity search predicates, since it has proven to efficiently implement Bitmap indices, and it is an open source software as well.

In our tests, we analyzed the cost to process OLAP similarity queries over images. We collected as performance measure the elapsed time, which was recorded issuing each query five times and calculating the average time. All cache and buffers were flushed after finishing each complete query.

Table 2

<table>
<thead>
<tr>
<th>Perceptual layer</th>
<th>Dimensionality of the feature vector</th>
<th>Number of representative images</th>
<th>Diameter of the Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color Histogram</td>
<td>256</td>
<td>3</td>
<td>594,402.88</td>
</tr>
<tr>
<td>Haralick Variance</td>
<td>4</td>
<td>3</td>
<td>80.51</td>
</tr>
<tr>
<td>Haralick Entropy</td>
<td>4</td>
<td>3</td>
<td>4.02</td>
</tr>
<tr>
<td>Haralick Uniformity</td>
<td>4</td>
<td>3</td>
<td>1.98</td>
</tr>
<tr>
<td>Haralick Homogeneity</td>
<td>4</td>
<td>4</td>
<td>1.25</td>
</tr>
</tbody>
</table>
6.2. Analyzing the processing of the similarity search predicate

The goal of this test was to analyze the cost of processing OLAP similarity queries over images that encompass only the similarity search predicate and deal with one or more perceptual layers. Thus, we focused on the splitIndexFV and the splitNotIndexFV strategies. We also focused on the general-purpose basic strategy.

Using as a basis the classes of queries defined in Section 6.1, we issued the following queries against the imageDW:

- **Query Q1 (belonging to Class 1):** “How many images are similar to a brain query image, according to the Haralick Variance perceptual layer and within the query radius \( r_1 \)?”.
- **Query Q2 (belonging to Class 2):** “How many images are similar to a brain query image, according to the Haralick Variance, Haralick Entropy, Haralick Uniformity and Haralick Homogeneity perceptual layers and within the query radius \( r_2 \)?”.
- **Query Q3 (belonging to Class 3):** “How many images are similar to a brain query image, according to the Color Histogram perceptual layer and within the query radius \( r_3 \)?”.
- **Query Q4 (belonging to Class 4):** “How many images are similar to a brain query image, according to the Color Histogram, Haralick Variance, Haralick Entropy, Haralick Uniformity and Haralick Homogeneity perceptual layers and within the query radius \( r_4 \)?”.

Although we are analyzing in this section only the similarity search predicate, all queries search for medical images related to brain problems and conditions. Thus, we also indexed in the imageSJBindex the attribute bodyPart of the conventional dimension table ExamDescription.

For each query, we varied the query radius in order to select different numbers of similar images. In detail, we defined four distinct values of query radii, which selected 10, 1000, 10,000, and 100,000 similar images. These values of similar images correspond, respectively, to about 0.007%, 0.07%, 0.7%, and 7% of the images stored in the imageDW. We considered small and large numbers of similar images for completeness since specialists may issue ad hoc queries that depend on the ongoing medical task. On the one hand, one query may represent a very restrictive analysis, which requires the manipulation of only few images. On the other hand, another query may represent a very broad analysis, which requires the manipulation of many images. Recall that the similar images are not the query answer. That is, similar images are further processed by task (f) to apply an aggregation function in order to calculate the query answer (e.g. the SUM aggregation function to sum the number of similar images) and to return the query answer to the specialist.

Fig. 13 depicts the performance results of the basic, the splitIndexFV and the splitNotIndexFV strategies after issuing Queries Q1, Q2, Q3 and Q4. As expected, the splitIndexFV and the splitNotIndexFV strategies spent less time than the basic strategy to process the OLAP similarity queries over images. This is due the fact they index in the imageSJBindex the distances between the stored images and the representative images while the basic strategy performs several disk accesses to retrieve these data from the imageDW. The performance gain of the splitNotIndexFV strategy over the basic strategy varied from 22.51% to 77.74% for Query Q1, from 69.07% to 86.51% for Query Q2, from 53.37% to 80.50% for Query Q3, and from 62.84% to 81.43% for Query Q4. The performance gain of the splitIndexFV strategy over the basic strategy varied from 45.32% to 85.96% for Query Q1, from 62.80% to 88.07% for Query Q2, up to 63.82% for Query Q3, and from 6.69% to 57.54% for Query Q4. In general, the performance gain of the splitNotIndexFV and the splitIndexFV strategies over the basic strategy increased as the number of similar images also increased.

Specifically regarding the splitIndexFV and the splitNotIndexFV strategies, the performance results showed that in general the

![Graphs](image1.png)

Fig. 13. Performance results of the basic, the splitIndexFV and the splitNotIndexFV strategies for processing OLAP similarity queries that encompass only the similarity search predicate. (a) Results for Query Q1. (b) Results for Query Q2. (c) Results for Query Q3. (d) Results for Query Q4.
splitIndexFV strategy provided better performance results for Queries Q1 and Q2, which required the manipulation of feature vectors with low dimensionality (i.e. the feature vectors of the Haralick perceptual layers have only 4 dimensions). The performance gain of the splitIndexFV over the splitNotIndexFV strategy was up to 44.01% for Query Q1 and up to 12.58% for Query Q2. Specifically for Query Q2, the results showed that for more selective similarity search predicates, the splitNotIndexFV overcame the splitIndexFV (i.e. for 10, 100, and 1000 similar images, which represent 0.007%, 0.07%, and 0.7% of the images stored in the imageDW). For smaller number of similar images, the proposed index imageSJBindex did not improve query performance. Therefore, besides the dimensionality of the feature vectors, the selectivity also determined when it was more appropriate to use the splitNotIndexFV strategy or the splitIndexFV strategy to process Query Q2. On the other hand, the splitNotIndexFV strategy provided the best performance results for Queries Q3 and Q4, which required the manipulation of feature vectors with high dimensionality (i.e. the feature vectors of the Color Histogram perceptual layer have 256 dimensions). The performance gain of the splitNotIndexFV strategy over the splitIndexFV strategy varied from 46.11% to 53.58% for Query Q3 and from 49.42% to 60.17% for Query Q4. The results described here demonstrate that accessing the imageDW to recover high dimensional feature vectors is more efficient than indexing them in the imageSJBindex. Thus, we can conclude that imageSJBindex should index feature vectors only when they have low dimensionality.

6.3. Analyzing the processing of conventional and similarity search predicates

The goal of this test was to analyze the cost of processing OLAP queries over images that encompass conventional predicates and similarity search predicates that deal with one or more perceptual layers. We focused on the five strategies proposed in this paper (i.e. basic, splitIndexFV, splitNotIndexFV, combinedIndexFV and combinedNotIndexFV).

Using as basis the classes of queries defined in Section 6.1, we issued the following queries to the imageDW:

- Query Q1 (belonging to Class 1): “How many images are similar to a brain query image, according to the Haralick Variance perceptual layer and within the query radius $r_q$, and satisfy a given conventional predicate (i.e. C1, C2, C3, C4, C5 or C6 described in Table 3)?”
- Query Q2 (belonging to Class 2): “How many images are similar to a brain query image, according to the Haralick Variance, Haralick Entropy, Haralick Uniformity and Haralick Homogeneity perceptual layers and within the query radius $r_q$, and satisfy a given conventional predicate (i.e. C1, C2, C3, C4, C5 or C6 described in Table 3)?”
- Query Q3 (belonging to Class 3): “How many images are similar to a brain query image, according to the Color Histogram perceptual layer and within the query radius $r_q$, and satisfy a given conventional predicate (i.e. C1, C2, C3, C4, C5 or C6 described in Table 3)?”
- Query Q4 (belonging to Class 4): “How many images are similar to a brain query image, according to the Haralick Variance, Haralick Entropy, Haralick Uniformity and Haralick Homogeneity perceptual layers and within the query radius $r_q$, and satisfy a given conventional predicate (i.e. C1, C2, C3, C4, C5 or C6 described in Table 3)?”

In this test, we fixed the query radius to manipulate 100,000 similar images (i.e. about 7% of the images stored in the imageDW). In order to evaluate the impact of the query selectivity for the conventional predicate over the proposed strategies, we defined six conventional predicates, as detailed in Table 3. This table also depicts the selectivity of Queries Q1, Q2, Q3 and Q4 with regard to each conventional predicate, from the more selective to the less selective conventional predicate. We consider that configurations C1 and C2 are more selective because they select a small number of similar images (i.e. from zero to eleven images). On the other hand, the configurations C4, C5 and C6 are less selective because they select a large number of similar images (i.e. greater than 4,772 images). We classify the configuration C3 as an intermediate one regarding query selectivity. Furthermore, recall that all queries search for medical images related to brain conditions and diseases. Thus, we are also considering in the queries this condition defined over the attribute bodyPart of the dimension table ExamDescription.

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Query conventional predicate</th>
<th>Selectivity (number of similar images)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>city=“Santos” AND macroregion=“Baixada Santista” AND year BETWEEN 1990 AND 1995 AND age BETWEEN 0 AND 60 AND state=“SP”</td>
<td>1 0 0 0</td>
</tr>
<tr>
<td>C2</td>
<td>macroregion=“Baixada Santista” AND year BETWEEN 1990 AND 1995 AND reasonInvestigation=“Tumor” AND age BETWEEN 0 AND 60 AND state=“SP”</td>
<td>8 11 7 9</td>
</tr>
<tr>
<td>C3</td>
<td>year BETWEEN 1990 AND 1995 AND reasonInvestigation=“Tumor” AND age BETWEEN 0 AND 60 AND state=“SP”</td>
<td>906 931 3669 922</td>
</tr>
<tr>
<td>C4</td>
<td>reasonInvestigation=“Tumor” AND age BETWEEN 0 AND 60 AND state=“SP”</td>
<td>4810 4803 4931 4772</td>
</tr>
<tr>
<td>C5</td>
<td>age BETWEEN 0 AND 60 AND state=“SP”</td>
<td>19,141 19,327 19,805 19,125</td>
</tr>
<tr>
<td>C6</td>
<td>state=“SP”</td>
<td>44,185 44,658 44,605 44,511</td>
</tr>
</tbody>
</table>
Although this condition is not present in the predicates described in Table 3, it is indexed by the imageDWE.

Fig. 14 depicts the performance results of the basic, splitIndexFV, splitNotIndexFV, combinedIndexFV, and combinedNotIndexFV strategies for issuing Queries Q1, Q2, Q3, and Q4, varying the query’s conventional predicate. According to the results, the choice of the best strategy to process OLAP queries over images that encompass conventional and similarity search predicates is determined by two factors: the dimensionality of the feature vectors and the query selectivity (i.e., the number of similar images).

Differently from the results described in Section 6.2 for the processing of only the similarity search predicate, the results detailed here regarding the processing of conventional and similarity search predicates showed that the basic strategy generally overcame the other strategies when the query selectivity for the conventional predicate was more selective. That is, in general, the basic strategy was the best strategy for the configurations C1 and C2. In situations where the basic strategy was not the best, its performance results were very close to the best strategy.

When the query selectivity for the conventional predicate was less selective (i.e., the configurations C4, C5, and C6), the combinedIndexFV and the combinedNotIndexFV strategies overcame the basic strategy. Furthermore, the combinedIndexFV and the combinedNotIndexFV strategies spent less time than the splitIndexFV and the splitNotIndexFV strategies. Specifically regarding the combinedIndexFV and the combinedNotIndexFV strategies, the performance results showed that the combinedIndexFV strategy was the best strategy to process Queries Q1 and Q2, which have similarity search predicates querying against feature vectors with low dimensionality. These queries also manipulate the Color Histogram perceptual layer, whose feature vectors have 256 dimensions.

6.4. Analyzing the feasibility of imageDWE from the point of view of specialists

In order to provide an evaluation of our proposal from specialists, we conducted a study that investigated the feasibility of the proposed imageDWE from the point of view of the specialists. We applied a questionnaire to the specialists that evaluated the applicability of our proposal considering three aspects of interest in the medical decision-making process. These aspects are described as follows:

- A1: Assistance of imageDWE to associate medical images with phenotypic characteristics of patients.
- A2: Utility of imageDWE to aid in case studies.
- A3: Applicability of imageDWE to aid in epidemiological studies.

We also defined two high level OLAP similarity queries, which were defined as follows:

- Q1: Consider an image showed in the computer screen, which depicts a cervical biopsy with evidence of abnormality. The specialist may create filters to select a patient’s geographic region, a patient’s age range, a patient’s sex, and a period of time. For instance, the specialist may establish the following criteria: male patients, aged between 30 and 40 years, who lived in the north region in the last 3 years. As a result, imageDWE returns how many similar images exist in the DW according to these criteria.
- Q2: Consider an image showed in the computer screen, which depicts a thyroid biopsy containing a lump. The specialist may
create a criteria based on filters that select a patient's geographic region, a patient's age range, a patient's sex, and a period of time. Furthermore, the specialist can also use as an additional selection criterion a value for the degree of similarity (more similar or more dissimilar) among images. As a result, imageDWE returns how many similar images exist in the DW according to these criteria.

We interviewed 6 specialists (i.e. 3 radiologists and 3 pathologists) from the same Brazilian public hospital that provided the images used in the tests. They assigned scores from 1 (lower evaluation) to 7 (higher evaluation) to queries Q1 and Q2 according to each one of the three aspects. A higher score represents a higher applicability of imageDWE according to the specialists’ expectations. The obtained results are depicted in Table 4. With an average close to 6 out of 7 the specialists confirmed the feasibility of the proposed imageDWE in the medical decision-making process. Notably, the specialists see more potential in the analysis (A1) and support of studies (A3) than in the direct aid in cases.

Table 4
Feasibility of imageDWE from the point of view of specialists.

<table>
<thead>
<tr>
<th>Specialist</th>
<th>Q1 A1</th>
<th>Q1 A2</th>
<th>Q1 A3</th>
<th>Q2 A1</th>
<th>Q2 A2</th>
<th>Q2 A3</th>
</tr>
</thead>
<tbody>
<tr>
<td>radiologist 1</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>radiologist 2</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>radiologist 3</td>
<td>6</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>pathologist 1</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>pathologist 2</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>pathologist 3</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Average</td>
<td>5.8</td>
<td>5.2</td>
<td>6.0</td>
<td>5.8</td>
<td>5.5</td>
<td>6.0</td>
</tr>
</tbody>
</table>

7. Related work

Related work may be classified into two main areas of research. The first one refers to the use of medical images in DWEs, and is detailed in Section 7.1. The second area of research is related to extensions of the Bitmap index for processing similarity queries and is investigated in Section 7.2.

7.1. Use of medical images in DWEs

There are few proposals focusing on the use of multimedia data in the decision-making process that have been proposed in the literature (i.e. [25–28]), but they differ from our work on their purpose and applicability. Wong et al. [25] propose a database system based on the use of a DW and multimedia data, which enables clinical and scientific studies of temporal lobe-epilepsy patients through the analysis and processing of images. This work also allows the execution of queries over patient reports. However, different from our work, Wong et al. [25] neither describe how medical images are managed in the DW nor discuss how OLAP queries are processed. On the other hand, in this paper we focus on OLAP similarity queries over images that encompass conventional and similarity search predicates, and that enable the management of one or more perceptual layers. Furthermore, we propose different strategies for processing OLAP similarity queries.

Jin et al. [28] propose a visual data cube and a multidimensional OLAP tool of image collections, such as web images indexed by search engines and photos shared on social networks. The work aims to provide online answers to queries that require summarized statistic information from images, and to handle the semantic related to the visual characteristics of these images. On the other hand, we focus on the storage of medical images as intrinsic data of the DW, and on the execution of OLAP similarity queries over those images.

7.2. Extensions of the Bitmap index for processing similarity queries

Although there are several approaches in the literature that focus on indices for DWEs and indices for images, to the best of our knowledge, there is no related work that focus on these two issues in the same set. Thus, we detail in this section approaches that are aimed to provide extensions to the Bitmap index for processing similarity queries (i.e. [29–31]), although they differ from our work on their purpose.

Jeong and Nang [29] introduce a Bitmap index based on representative dimensions. The feature vectors of images are mapped into bit-vectors where a bit equal to 1 indicates that a given feature is a representative dimension. Otherwise, the bit is set to 0. Given a query image, the dissimilarity is calculated by applying the XOR operation between the query and the bit-vectors of the stored images. The images whose resulting bit-vectors have low quantity of 1s are the most similar ones to the query image and, therefore, are the answer of the query. A first limitation of this related work is that it does not define a method for determining the number of dimensions that best represent an indexed image. Furthermore, this work focuses only on k-nearest neighbor queries and provides inaccurate answers to these queries.

The Hierarchical Bitmap Index [30] uses sets of two bits to represent the values of the feature vectors of an image to compare them to the feature values of other images. That is, the distance between the feature values of two images may be relatively high (“11”), low (“00”), or neither (“01”). This work also defines a hierarchy of intervals that enables the approximated calculation of the distances among the images based on the binary representation of the values of the feature vectors. One limitation of this approach is the fact that it indexes each dimension that represents an image using the same hierarchy of intervals. As a consequence, attributes that have a small domain may demand the creation of several unnecessary intervals.

The main idea of the grid Bitmap index [31] is to discretize the values of each dimension of the feature vector in several intervals where each interval represents a cluster containing a portion of data. Two points ranked in the same interval are considered similar in that dimension. The main limitation of this work is that it applies, for each dimension, the same mechanism to cluster data. However, in real-world applications, each dimension has a specific distribution of data, which may require the use of a specific clustering mechanism.

In addition to the particular limitations of the aforementioned related work, the indices described in [29,30] cannot be used to...
process OLAP similarity queries over images. In fact, these indices use bits to represent the dimensions of the feature vectors while a Bitmap index that improves the processing of OLAP similarity queries over images managed in a DW should represent the tuples of the fact table that reference the dimension tables. Furthermore, no related work focuses on the intrinsic characteristics of DWEs, such as the multidimensionality of data. On the other hand, we introduce in this paper the imageSJBindex, which is an index structure specially organized to index images and conventional data in a DWE. As described in Section 6, the use of the proposed imageSJBindex improves the performance of OLAP similarity queries over images.

8. Conclusions and future work

In this paper, we proposed imageDWE, a non-conventional data warehousing environment that enables the storage of intrinsic features of medical images in the data warehouse and supports OLAP similarity queries over them. The major characteristics of imageDWE are described as follows. It introduces the concept of perceptual layer, which contains data related to the intrinsic features of images according to a given feature descriptor. It also proposes the imageDW, which is an extended data warehouse with dimension tables specifically designed to support perceptual layers. Additionally, it empowers the conventional ETL process to also generate data of the perceptual layers and to store them in the imageDW. It also extends the conventional OLAP query processing capabilities so to support queries composed of a conventional predicate and a similarity search predicate that encompasses the specification of one or more perceptual layers. Furthermore, it proposes an index technique that improves the query processing performance.

The index technique encompasses the specification of the novel imageSJBindex, a star-join Bitmap index that indexes data related to the perceptual layers and optionally enables the indexing of conventional data and feature vectors of images. The technique also focuses on the definition of different strategies to process OLAP similarity queries over images in addition to the basic strategy provided by the extended OLAP query processing. The proposed strategies define different orders for processing conventional and similarity search predicates, and determine which data should be indexed by the imageSJBindex.

The proposed imageDWE was validated through performance tests carried out over a data warehouse environment that consolidated medical images from exams of several modalities, such as brain, breast, and chest related conditions and diseases. We analyzed the different strategies that we introduced in this paper in order to indicate which was the best one to process a given OLAP similarity query over images issued by a specialist. The performance results demonstrated the feasibility of our proposed imageDWE to manage images. They also demonstrated the efficiency of our proposed strategies to process OLAP similarity queries. In fact, depending on some characteristics of the query, such as selectivity, conventional and similarity search predicates involved, and dimensionality of the feature vectors, different processing strategies were able to provide better performance results over the others.

The proposed imageDWE may be extended to focus on supporting a wider range of functionalities and expanding its capability. Related research issues are described as follows:

- The definition of a taxonomy for classifying different types of OLAP similarity queries over images that can be issued against the imageDW. From the perspective of data warehousing environments, in this paper we focused on slice and dice OLAP queries. Other types of queries that should be considered are drill-down, roll-up and drill-across OLAP queries.
- The proposal of the cube operator for the imageDW, that is, how to deal with hierarchies of images and how to efficiently compute the cube.
- The extension of imageDWE to support fuzzy logic, providing a fuzzy image data warehousing environment. From the perspective of the specialists, this environment should also support other ranges of interesting queries such as “How many images are similar to a given mammography image that contains a tumor with no clear boundaries according to one or more appropriate perceptual layers?”
- The coupling of imageDWE with a hospital’s conventional data warehousing environment and a hospital’s picture archiving and communication system (PACS). As a consequence, imageDWE should be able to integrate the hospital’s day-by-day conventional data with medical clinical images retrieved from the PACS.
- The development of a tool used by specialists to interact with the proposed imageDWE. For instance, the tool could offer a graphical interface through which the specialists should be able to choose the feature descriptors to be applied in a given query by using as basis their high level characteristics, such as color, texture, and shape, instead of explicitly indicating the use of Color Histograms, Haralick descriptors and Zernike moments, respectively. Also, the tool could add to the query answer the clinical relevance of the feature descriptors used to represent the image dataset and to issue the query (see some comparisons about the clinical relevance of feature descriptors in [32,33]). Furthermore, recall that imageDWE does not return images: it applies an aggregation function over the similar images manipulated by the processing strategies to calculate the query answer. Thus, another interesting functionalities that could be provided by the tool are: (i) to show the similar images that were used to calculate the query answer, and (ii) to apply relevance feedback techniques over these similar images to take the preferences of the specialists into account (e.g. [34,35]). That is, we have successfully developed a technical infrastructure and will look into integration and usability issues next.

Conflict of interest statement

None declared.

Acknowledgments

We thank FAPESP, CAPES, CNPq and FINEP for the financial support. The first and last authors are funded by FAPESP, grants 2012/14469-8 and 2011/23904-7, respectively.

References


Jefferson William Teixeira received the B.Sc. degree in Computer Science from the University of São Paulo, Brazil, in 2011. In 2015, he received the M.Sc. degree in Computer Science from the University of São Paulo, Brazil. He is currently a Software Developer at Daitan Group in the area of telecom systems. His research interests are data warehousing, indexing, image databases and biomedical databases.

Luana Peixoto Annibali received the B.Sc. degree in Biomedical Informatics from the University of São Paulo, Brazil, in 2008. In 2011, she received the M.Sc. degree in Computer Science from the Federal University of São Carlos, Brazil. She is currently a Software Developer at Acesso Digital Company in the area of information technologies and services. Her research interests are data warehousing, image databases and biomedical databases.

Joaquim Cezar Felipe received the B.Sc. degree in Mechanical Engineering from the University of São Paulo, Brazil, in 1986. In 2000, he received the M.Sc. degree in Computer Science from the Federal University of São Carlos, Brazil. He obtained his Ph.D. degree in Computer Science from the University of São Paulo, Brazil, in 2005. He is currently an Assistant Professor at the Department of Computing and Mathematics at the University of São Paulo, Campus of Ribeirão Preto, Brazil. His research interests include biological databases, biomedical databases, medical image processing and analysis, and computer-aided diagnosis.

Ricardo Rodrigues Ciferri received the B.Sc. degree in Computer Science from the Federal University of São Carlos, Brazil, in 1992. In 1995, he received the M.Sc. degree in Computer Science from the State University of Campinas, Brazil. He obtained his Ph.D. degree in 2002 in Computer Science from the Federal University of Pernambuco, Brazil. He is currently an Associate Professor at the Department of Computer Science at the Federal University of São Carlos, Campus of São Carlos, Brazil. His research interests are data warehousing, spatial databases, geographical information systems, biological databases, biomedical databases, data integration, cloud databases and big data.

Cristina Dutra de Aguiar Ciferri received the B.Sc. degree in Computer Science from the Federal University of São Carlos, Brazil, in 1992. In 1995, she received the M.Sc. degree in Computer Science from the State University of Campinas, Brazil. She obtained her Ph.D. degree in 2002 in Computer Science from the Federal University of Pernambuco, Brazil. She is currently an Assistant Professor at the Department of Computer Science at the University of São Paulo, Campus of São Carlos, Brazil. Her research interests are data warehousing, spatial databases, biomedical databases, data provenance, data integration, cloud databases and big data.