Agrupamento de dados complexos para apoiar consultas por similaridade com tratamento de restrições

Jessica Andressa de Souza

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Jessica Andressa de Souza

Agrupamento de dados complexos para apoiar consultas por similaridade com tratamento de restrições

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutora em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Agma Juci Machado Traina

USP – São Carlos
Janeiro de 2019

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

d719a de Souza, Jessica Andressa
Agrupamento de dados complexos para apoiar
consultas por similaridade com tratamento de
restrições / Jessica Andressa de Souza; orientadora
Agma Juci Machado Traina. -- São Carlos, 2019.
102 p.

Tese (Doutorado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2019.

1. Métodos de Acesso Métrico. 2. Consultas por
Similaridade Restritas. 3. Detecção de Agrupamento
de Dados. I. Juci Machado Traina, Agma, orient. II.
Título.

Jessica Andressa de Souza

Clustering complex data for processing constrained similarity queries

Doctoral dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Profa. Dra. Agma Juci Machado Traina

USP – São Carlos
January 2019

AGRADECIMENTOS

À FAPESP (Processo Número 2013/21378-1), CAPES e CNPq pelo apoio financeiro à realização deste trabalho.

RESUMO

DE SOUZA, J. A. **Agrupamento de dados complexos para apoiar consultas por similaridade com tratamento de restrições**¹. 2019. 102 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Devido aos avanços tecnológicos ocorridos nos últimos anos, houve um aumento na quantidade e complexidade de dados gerados. Assim, aprofundou-se a necessidade do desenvolvimento de estratégias eficientes que permitam o armazenamento, a recuperação e a representação resumida desses tipos de dados complexos. Dentre as estratégias exploradas pelos pesquisadores da área para atender a esses propósitos estão os Métodos de Acesso. Esses métodos têm como objetivo indexar os dados de maneira eficaz para reduzir o tempo de consulta. Além disso, eles têm sido aplicados para apoiar o processamento de técnicas de Mineração de Dados, como a Detecção de Agrupamentos. Dentre os métodos de acesso, as estruturas de indexação métrica são construídas usando apenas o critério baseado na distância entre os elementos do conjunto de dados em questão, *i.e.* operações de similaridade sobre as características intrínsecas dos dados. Desse modo, nem sempre os resultados correspondem ao contexto desejado pelo usuário.

Este trabalho explorou o desenvolvimento de algoritmos que permitam aos métodos de acesso métrico processarem detecção de agrupamento de dados para auxiliar o processamento de consultas com maior carga semântica; visando contribuir no tratamento da questão da eficiência de abordagens que envolvam operações por similaridade (por exemplo, técnicas de mineração de dados e consultas por similaridade). Diante deste contexto, foram desenvolvidas três abordagens, a primeira apresenta o método `clusMAM` (*Unsupervised Clustering using Metric Access Methods*), o qual tem como objetivo apresentar um agrupamento dos dados com a aplicação de um Método de Acesso Métrico a partir de um conjunto resumido dos dados. A segunda abordagem apresenta a abordagem `CCkNN` (*Class-Constraint k-NN*) para lidar com o problema de restrições de múltiplas classes sobre o espaço de busca. Por fim, a terceira abordagem apresenta o método `CfQ` (*Clustering for Querying*) realizando a integração das técnicas `clusMAM` com `CCkNN`, empregando os pontos positivos de cada estratégia adotada pelos algoritmos. No geral, os experimentos realizados mostram que os métodos propostos contribuem de maneira efetiva na redução de medidas de similaridade requeridas durante um processamento de técnicas que são baseadas em computações de distância.

Palavras-chave: Métodos de Acesso Métrico, Consultas por Similaridade Restritas, Detecção de Agrupamento de Dados.

¹ Este trabalho conta com o apoio financeiro da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP Processo 2013/21378-1)

ABSTRACT

DE SOUZA, J. A. **Clustering complex data for processing constrained similarity queries**². 2019. 102 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Due to the technological advances over the last years, both the amount and variety of data available have been increased at a fast pace. Thus, this scenario has influenced the development of effective strategies for the processing, summarizing, as well as to provide fast and automatic understanding of such data. The Access Methods are strategies that have been explored by researchers in the area to aid these purposes. These methods aim to effectively index data to reduce the time required for processing similarity querying. In addition, they have been applied to aid the processing of Data Mining techniques, such as Clustering Detection. Among the access methods, the metric structures are constructed applying only the criterion based on the distance computation between the elements of the dataset, i.e. similarity operations on the intrinsic characteristics of the dataset. Thus, the results do not always correspond to the context desired by users.

This work explored the development of algorithms that allow metric access methods to process queries with a higher semantic load, aimed at contributing to the treatment of the quality question on the results of approaches that involve similarity operation (for example, data mining techniques and similarity queries). In this context, three approaches have been developed: the first approach presents the method `clusMAM` (Unsupervised Clustering using Metric Access Methods), which aims to display a clustering from a dataset with the application of a Metric Access Method from a summarized set. The second approach presents the `CCKNN` approach to dealing with the problem of multi-class constraints on the search space. Finally, the third proposal presents the method `CfQ` (Clustering for Querying) by integrating the techniques `clusMAM` with `CCKNN`, using the positive points of each strategy applied by the algorithms. In general, the experiments carried out showed that the proposed methods can contribute to an effective way of reducing similarity computations, which is required during a processing of techniques that are based on distance computations.

Keywords: Metric Access Methods, Constrained Similarity Queries, Clustering Detection.

² This research has been supported by FAPESP 2013/21378-1

LISTA DE ILUSTRAÇÕES

Figura 1 – Exemplificação da abordagem de amostragem com auxílio de estruturas de indexação para apoiar técnicas de detecção de agrupamento (conjunto de imagens de animais (FU <i>et al.</i> , 2018)).	28
Figura 2 – Exemplificação da representação de um dado complexo, tipo de domínio: imagens.	34
Figura 3 – Ilustração das formas geométricas produzidas com base nas métricas L_p . . .	36
Figura 4 – Representação da forma geométrica produzida pela função de distância <i>Mahalanobis</i>	36
Figura 5 – Representação de uma função de distância fictícia. Cada seta representa a distância entre as imagens, as distâncias são todas simétricas (setas duplas) e não negativas, atribuindo o valor 0 para imagens iguais. A distância entre as imagens x e y é sempre menor que a distância das imagens x e z somada com a distância das imagens z e y (desigualdade triangular).	38
Figura 6 – Imagem ilustrativa da organização dos elementos de dados em uma estrutura de métodos de acesso métrico. (A) apresenta uma indexação de 16 elementos com no máximo 03 elementos por nó em uma Slim-tree e em (B) exemplifica uma sobreposição de nós antes e depois da aplicação do algoritmo de pós-processamento <i>Slim-Down</i>	44
Figura 7 – Imagem ilustrativa de uma divisão de nós. (A) ilustra a inserção de um elemento em uma página com capacidade máxima causando <i>overflow</i> , logo, a página foi dividida e a página 2 foi gerada, onde um elemento de cada página foi promovido para subir um nível e juntos geram a página 3. (B) ilustra a diferença do processo resultante após uma divisão de nós com duas políticas diferentes para os mesmos elementos, com sobreposição e sem sobreposição entre os nós.	44
Figura 8 – Ilustração das consultas por abrangência e k -NN em um domínio bidimensional com função de distância Euclidiana. (A) apresenta um resultado de uma consulta por abrangência e (B) exemplifica um resultado de uma consulta aos k -vizinhos mais próximos.	45
Figura 9 – Esquematização das estratégias utilizadas dos algoritmos de agrupamento. Os detalhes específicos de cada estratégia podem ser adquiridos em (BERKHIN, 2006)	48

- Figura 10 – Número de cálculos de distância executados em cada algoritmo (os resultados do eixo-y estão em escala log). (a) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 32 dimensões, variando o número de grupos por conjunto. (b) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 64 dimensões, variando o número de grupos por conjunto. Para o conjunto sintético Sintciaccia100E10G64D, a estratégia executou quatro ordens de magnitude mais rapidamente do que o PAM-Slim. (c) gráfico referente aos conjuntos Dim-sets com 16 grupos cada, variando o número de dimensões. 61
- Figura 11 – Tempo de execução de cada algoritmo. Os gráficos (a) e (b) apresentam os resultados do eixo-y em escala log). (a) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 32 dimensões, variando o número de grupos por conjunto. (b) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 64 dimensões, variando o número de grupos por conjunto. (c) gráfico referente aos conjuntos Dim-sets com 16 grupos cada, variando o número de dimensões. c_{lus}MAM apresentou menor tempo de processamento, além disso, é possível observar que conforme o número de grupos aumenta, a estratégia ainda mantém seu tempo de processamento. 62
- Figura 12 – Qualidade resultante com relação ao índice de validação Silhueta Simplificada. É importante ressaltar que valores mais próximos de 1 apresentam uma estrutura apropriada de grupos. Nesse quesito c_{lus}MAM manteve equivalência aos resultados das outras estratégias. (a) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 32 dimensões, variando o número de grupos por conjunto. (b) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 64 dimensões, variando o número de grupos por conjunto. (c) gráfico referente aos conjuntos Dim-sets com 16 grupos cada, variando o número de dimensões. 63
- Figura 13 – Qualidade resultante com relação ao índice de validação Davies-Bouldin. Valores mínimos apresentados (mais próximos de 0) indicam um número potencial de grupos presentes nos dados. c_{lus}MAM manteve equivalência nos resultados. (a) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 32 dimensões, variando o número de grupos por conjunto. (b) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 64 dimensões, variando o número de grupos por conjunto. (c) gráfico referente aos conjuntos Dim-sets com 16 grupos cada, variando o número de dimensões. 63

Figura 14 – Conjunto de dados Iris com 150 elementos de 4 dimensões. O gráfico apresenta os valores sugeridos de n e, o índice de qualidade variando a capacidade do nó. <code>clusMAM</code> apresentou $n=2$ para 512 Bytes com melhor valor do índice de qualidade. É importante notar que esse conjunto contém 3 classes, no entanto, 2 são completamente sobrepostas. A estratégia sugere a presença de dois grupos bem separados nos dados, sendo 100 elementos em um grupo e 50 em outro.	64
Figura 15 – Conjunto de dados Wdbc com 569 elementos de 31 dimensões. <code>clusMAM</code> apresentou $n=2$ para todos os valores da capacidade de nó e com qualidade equivalente. Esse conjunto é composto de 2 classes a estratégia sugere a presença de 2 grupos bem separados na distribuição dos dados, sendo 558 elementos em um grupo e os demais (11) em outro grupo mais distante. . . .	64
Figura 16 – Exemplo no espaço bidimensional antes e após a execução da nova estratégia para o Sint-ciaccia com 20.000 elementos, 2 dimensões e 100 grupos. (a) Considerando um conjunto de dados bidimensional de 10 grupos e (b) após à aplicação do <code>clusMAM</code> com os rótulos dos grupos. Esse desempenho é mantido mesmo quando o número de dimensões e elementos são variados. .	65
Figura 17 – Ilustração do impacto sobre o espaço de busca, com a aplicação das restrições de classes sobre uma consulta k -NN tradicional, considerando o valor de $k=3$ com relação ao elemento s_q de $\mathbf{Q}_c=\{\mathcal{C}_1=\{\text{espécie:iris-virginica}\}\}$. (a) k -NN clássico e dados não rotulados, (b) k -NN e dados rotulados e (c) após à aplicação do algoritmo com restrição de classes sobre os dados rotulados. .	69
Figura 18 – Ilustração das combinações de classes dos diagnósticos de exames de mamografia. (a) $ \mathcal{C} =2$ dimensões de classes e para cada dimensão $\mathcal{C}_i=2$, (b) apresenta todas as combinações de classes sem sobreposições de classes da mesma dimensão e, para cada combinação um índice será construído.	70
Figura 19 – Exemplo considerando um espaço bidimensional referente aos conjuntos de dados sintéticos utilizados nos experimentos.	73
Figura 20 – Uma amostra do conjunto de imagens de mamografias.	73
Figura 21 – Tempo total das consultas CCKNN executadas sobre uma estrutura de indexação com todos os elementos do conjunto de dados. (a) Valores referentes ao conjunto S1E2D2DC2C para $ \mathbf{Q}_c =2$, (b) Valores referentes ao conjunto S1E2D3DC3C para $ \mathbf{Q}_c =3$ e (c) Valores referentes ao conjunto S1E2D3DC4C para $ \mathbf{Q}_c =3$. O eixo y (em escala $\log$) apresenta a média de 100 consultas para cada valor de k (no eixo x) utilizando diferentes conjuntos compostos de centros de consulta selecionados aleatoriamente.	76

- Figura 22 – Consultas restritas por classes sobre estruturas específicas (rotuladas), com relação às classes do elemento de consulta s_q . (a) Valores referentes ao conjunto S1E2D2DC2C para $|C|= 2$ e estruturas construídas com diferentes combinações de 2 dimensões de classes, (b) Valores referentes ao conjunto S1E2D3DC3C para $|C|= 3$ e estruturas construídas com diferentes combinações de 3 dimensões de classes, (c) Valores referentes ao conjunto S1E2D3DC4C para $|C|= 3$ e estruturas construídas com diferentes combinações de 3 dimensões de classes, mas com 4 classes diferentes por dimensão. 76
- Figura 24 – Consultas aos vizinhos mais próximos restritos por classes, sobre estruturas específicas (rotuladas). Os valores são referentes ao conjunto de dados de imagens de exames de mamografia (*Left CranioCaudal*) com 256 características e $|Q_c|= 2$. As estruturas de indexação foram construídas com diferentes combinações de classes considerando $|C|= 2$. O eixo-y (em escala *log*) apresenta os resultados com referência à média de 100 consultas **k**-NN restritas por classes para cada valor de **k** (no eixo-x) utilizando distintos conjuntos de centros de consulta selecionados aleatoriamente. 76
- Figura 23 – Consultas restritas por classes sobre estruturas específicas (rotuladas), com relação às classes do elemento de consulta s_q . (a) Valores referentes ao conjunto S1E2D3DC4C para $|Q_c|= 1$ e estruturas construídas com diferentes combinações considerando 1 dimensão de classe, (b) Valores referentes ao conjunto S1E2D3DC2C para $|Q_c|= 2$ e estruturas construídas com diferentes combinações considerando 1 dimensão de classe, (c) Valores referentes ao conjunto S1E2D3DC3C para $|Q_c|= 3$ e estruturas construídas com diferentes combinações considerando 1 dimensão de classe 77
- Figura 25 – Ilustração do processamento da abordagem CfQ. (A) Dado um conjunto de dados complexos (*e.g.* um conjunto de imagens), onde suas características são extraídas por um algoritmo de extração de características. (B) Todos os elementos são indexados sobre um método de acesso métrico. É importante destacar que o parâmetro tamanho da página (nó) influencia os resultados do método (ver Seção 4.4). (C) O processamento da amostragem do conjunto é executado sobre o nível $l - 1$, nessa fase todos os elementos representativos são selecionados a partir da estrutura construída. (D) A detecção de agrupamento é executada sobre a amostra selecionada do conjunto. Após essa fase, cada elemento do conjunto será atribuído ao elemento medóide mais próximo. (E) Índices de validade são aplicados sobre o agrupamento resultante para analisar a homogeneidade e separação dos elementos. (F) Consultas por similaridade serão executadas sobre uma lista ordenada de grupos, no qual o elemento s_q é o mais próximo dos elementos medóides de cada grupo. 83

Figura 26 – (a) Número de cálculos de distância executados pelos algoritmos clusMAM e PAM-Slim (os resultados do eixo-y estão em escala log). clusMAM apresentou o mesmo comportamento para os 04 conjuntos sintéticos aplicados (Rand100_10d_2dc_2c_5n, Rand100_10d_2dc_2c_10n, Rand100_10d_2dc_2c_15n e Rand100_10d_2dc_2c_20n). Além disso, observando os resultados sumarizados em (b) e (c) o algoritmo apresentou agrupamento apropriado com relação aos índices de qualidade.	85
Figura 27 – Número de cálculos de distância computados para retornar os k elementos mais próximos a partir de um elemento de consulta s_q . Resultados referentes aos conjuntos de dados sintéticos, variando o número de elementos entre 100.000, 200.000 e 300.000, com relação à execução de consultas k -NN clássicas, isto é, sem empregar restrições de classes.	87
Figura 28 – Número de cálculos de distância computados para retornar os k elementos mais próximos a partir de um elemento de consulta s_q . Resultados referentes aos conjuntos de dados sintéticos, variando o número de elementos entre 100.000, 200.000 e 300.000, com relação à execução de consultas k -NN clássicas, isto é, sem empregar restrições de classes.	87
Figura 29 – Número de cálculos de distância executados pelos conjuntos sintéticos Rand100_10d_2dc_2c_30n e Rand200_10d_3dc_2c_30n variando o número de dimensões de classes ($ C $). A abordagem CfQ-clusMAM apresentou comportamento similar aos apresentados anteriormente, ou seja, essa abordagem é capaz de ainda apresentar menor número de cálculos de distância, mesmo aumentando os valores de dimensões de classes.	87
Figura 30 – Número de cálculos de distância executados pelos conjuntos sintéticos Rand200_10d_3dc_2c_30n e Rand100_10d_3dc_2c_40n variando o número de grupos (n) de cada conjunto.	88
Figura 31 – Ilustração do impacto de agrupar para consultar sobre o espaço de busca, com a aplicação das restrições de classes sobre uma consulta k -NN tradicional, considerando o valor de k = 3 com relação ao elemento s_q de $Q_c=\{\mathcal{C}_1=\{\text{espécie:iris-virginica}\}\}$	89

LISTA DE ALGORITMOS

Algoritmo 1 – clusMAM	59
Algoritmo 2 – CCkNN	71
Algoritmo 3 – CfQ	82

LISTA DE TABELAS

Tabela 1 – Comparação entre as estratégias considerando: se os algoritmos sugerem um número de grupos (n), a complexidade de tempo sobre as amostras ($\mathcal{M}$) dos conjuntos de dados e parâmetro(s) de entrada. Os detalhes sobre esses algoritmos são abordados no Capítulo 4 e no trabalho apresentado em (SOUZA; CAZZOLATO; TRAINA, 2016)	30
Tabela 2 – Descrição dos conjuntos de dados apresentados nesse capítulo	60
Tabela 3 – Descrição dos conjuntos sintéticos e reais utilizados nos experimentos	85
Tabela 4 – Comparação entre os algoritmos <i>clusMAM</i> e PAM-Slim com relação aos conjuntos de dados reais considerados nesse capítulo. É importante notar que o número de grupos (n) apresentado nessa tabela para cada conjunto foi sugerido pelo <i>clusMAM</i>	88

LISTA DE SÍMBOLOS

$\mathbf{R}$ — conjunto resposta de uma consulta por similaridade

$d()$ — função de distância

$\mathbf{k}$ — o número de vizinhos em uma consulta NN (*nearest neighbors*)

s_q — um objeto de consulta (ou centro de consulta)

s_i, s_j — objetos de $\mathbf{S}$

$\mathbf{S}$ — conjunto dos elementos de domínio $\mathbb{D}$

$\mathfrak{M}$ — subconjunto (amostra) de $\mathbf{S}$

$\mathbb{D}$ — domínio dos elementos

$\mathbb{G}$ — conjunto de grupos

g_i — objeto de $\mathbb{G}$

$\mathbf{M}$ — conjunto de medóides

m_i — objetos de $\mathbf{M}$

n — número de grupos

$\mathbf{T}$ — conjunto de árvores

t_i — objeto de $\mathbf{T}$

$\mathfrak{C}$ — conjunto de dimensões de classes do conjunto de dados

$\mathcal{C}_i$ — conjunto de classes de uma dimensão de classe $\in \mathfrak{C}$

$\mathbf{Q}_c$ — conjunto de restrições de classes referente a s_q

c — número de classes do conjunto de dados

$\mathbf{C}_t$ — conjunto de classes de um objeto t_i

SUMÁRIO

1	INTRODUÇÃO	25
1.1	Contextualização	25
1.2	Motivação	26
1.3	Definição do Problema	28
1.4	Objetivos e Contribuições	29
1.5	Organização do Documento	31
2	REPRESENTAÇÃO E RECUPERAÇÃO DO DOMÍNIO DE DADOS	33
2.1	Considerações Iniciais	33
2.2	Extração de Características	34
2.3	Análise de Similaridade: Funções de Distância	34
2.4	Métodos de Acesso	37
2.5	Métodos de Acesso Métrico	37
2.6	Métodos de Acesso Métrico Estáticos	38
2.7	Métodos de Acesso Dinâmicos	40
2.8	Consultas por Similaridades	45
2.9	Considerações Finais	45
3	MÉTODOS DE ACESSO E DETECÇÃO DE AGRUPAMENTO DE DADOS	47
3.1	Considerações Iniciais	47
3.2	Estratégias de Otimização	49
3.3	Medidas de Avaliação de Agrupamento	50
3.4	Considerações Finais	52
4	ABORDAGEM PARA AGRUPAMENTO PARTICIONAL: <i>clusMAM</i> .	55
4.1	Considerações Iniciais	55
4.2	Trabalhos Correlatos	57
4.3	Método Proposto	57
4.4	Avaliações Experimentais	59
4.4.1	<i>Resultados Obtidos</i>	61
4.5	Considerações Finais	65

5	CONSULTAS POR SIMILARIDADE RESTRITAS POR CLASSES:	
	CCKNN	67
5.1	Considerações Iniciais	67
5.2	Definição do Problema	68
5.3	Método Proposto	70
5.4	Avaliações Experimentais	72
5.4.1	<i>Descrição dos conjuntos de dados</i>	72
5.4.2	<i>Resultados Obtidos</i>	74
5.5	Considerações Finais	77
6	CONSULTA POR SIMILARIDADE RESTRITA APOIADA POR TÉCNICA DE AGRUPAMENTO: CfQ	79
6.1	Considerações Iniciais	79
6.2	Trabalhos Correlatos	80
6.3	Abordagem Proposta	81
6.4	Avaliações Experimentais	84
6.5	Resultados Obtidos	84
6.5.1	<i>Considerações Finais</i>	88
7	CONCLUSÃO	91
7.1	Considerações Finais	91
7.2	Contribuições	92
7.3	Trabalhos Futuros	93
7.4	Publicações no Período	93
	REFERÊNCIAS	95

INTRODUÇÃO

1.1 Contextualização

Com o passar do tempo, devido aos avanços da tecnologia de armazenamento, houve um aumento expressivo não só na quantidade dos dados, mas também na sua complexidade. Então, o desenvolvimento de novas técnicas para manipulá-los de maneira eficiente tornou-se um objetivo importante para os pesquisadores da área ([SINAEPOURFARD *et al.*, 2016](#)). A complexidade está relacionada aos dados não estruturados, os quais necessitam de uma transformação para deixá-los adequados para a extração de padrões, em que suas características são extraídas e submetidas ao processo de descoberta de conhecimento, a qual direciona o suporte à tomada de decisão. Esses dados podem ser de natureza multimídia, como conjuntos de imagens, vídeos e áudios, ou uma coleção de documentos para ser manipulada, entre outros.

Dentre as técnicas de mineração de dados apresentadas na literatura encontra-se a detecção de agrupamento de dados, que pode ser realizada como aprendizado de máquina não supervisionado ([JAIN, 2010](#)). A detecção de agrupamento tem se destacado em diversas aplicações de análise de dados, como: exploração de dados científicos, recuperação de informação, mineração de textos, segmentação de imagens, aplicações em banco de dados espaciais, *marketing*, auxílio ao diagnóstico médico, biologia computacional, entre outras ([KALRA; LAL; QAMAR, 2018](#)).

A detecção de agrupamento tem sido aplicada, tanto como ferramenta isolada quanto como uma ferramenta de pré-processamento, com três objetivos principais ([JAIN; MURTY; FLYNN, 1988](#)):

1. Para descobrir a estrutura subjacente presente no conjunto de dados obtendo uma visão geral dos dados, gerando hipóteses, detectando anomalias, e identificando características proeminentes;

2. Para obter uma classificação natural dos dados direcionando análises futuras sobre determinados conjuntos de agrupamentos. Por exemplo, com o propósito de identificar o grau de similaridade entre formas de organismos (relacionamento filogenético);
3. Para conseguir uma representação resumida dos dados, ou seja, para organizar e sumarizar os dados por meio de protótipos de agrupamento. Os dados pré-processados podem então ser utilizados por técnicas como a de classificação, que utilizam os agrupamentos detectados como ponto de partida para a execução de seus processos de análise.

Embora o desenvolvimento desse tipo de técnica já tenha sido bastante explorado na literatura, ainda existem questões em aberto, e uma dessas está relacionada a *escalabilidade* e *qualidade* dos agrupamentos resultantes. O grande desafio é que o problema de executar eficientemente grandes conjuntos de dados não é uma tarefa fácil e, os resultados obtidos nem sempre correspondem ao particionamento mais adequado para os dados. Dessa maneira, esse processo pode requerer uma estratégia de otimização que capture a essência do conjunto de dados. Essa estratégia deve representar o conjunto de dados por meio de uma amostragem com um número adequado de elementos, de tal maneira que esses elementos sejam discriminatórios para resultar em particionamentos com qualidade apropriada. Baseando-se nessas necessidades, diversas pesquisas estão sendo realizadas com o objetivo de desenvolver abordagens eficientes e eficazes, que auxiliam outras técnicas que possam utilizar os agrupamentos detectados como ponto de partida, como exemplo, as consultas por similaridades.

1.2 Motivação

A representação resumida dos dados sintetiza o conjunto por meio de técnicas de mineração de dados, tal como detecção de agrupamento de dados, que os sintetiza, de acordo com as tarefas desejadas. Essa técnica aplica uma *função de distância*, cujo propósito é calcular a similaridade entre pares de elementos do conjunto. Desse modo, segundo (HAN; KAMBER; PEI, 2012), uma análise de grupos ou agrupamento é um processo de particionar o conjunto de dados em subconjuntos. Assim, os elementos do mesmo grupo apresentam maior grau de similaridade, com relação aos elementos de outros grupos.

As técnicas de agrupamentos podem ser *soft* (um elemento pode estar em vários grupos com diferentes valores de sobreposição) ou *hard* (um elemento deve pertencer somente a um grupo), dependendo do objetivo desejado. A questão é que selecionar/desenvolver uma técnica de agrupamento é uma tarefa desafiadora, pois existem muitos fatores que devem ser explorados em um conjunto de dados, tais como: análise exploratória dos conjuntos e distribuição dos dados, pois deve-se ter cautela na análise de conjuntos grandes, esparsos e com sobreposição, que podem não ser avaliados apropriadamente. As técnicas de agrupamento *hard* são amplamente utilizadas na literatura, mas as particionais *hard* são conhecidas por serem custosas para grandes

conjuntos de dados, pois apresentam alta complexidade de tempo. Por exemplo, a técnica PAM (*Partitioning Around Medoids*) (KAUFMAN; ROUSSEEUW, 1990), um dos primeiros algoritmos baseado na técnica *k-medoid* avalia o efeito de uma substituição entre um elemento medóide por um elemento não-medóide e realoca os demais elementos aos grupos candidatos. A abordagem adotada por esse algoritmo apresenta uma qualidade apropriada de agrupamento. No entanto, a desvantagem do uso desse tipo de abordagem está no fato de que as técnicas demandam alto custo computacional, apresentando complexidade de tempo quadrática sobre o número de elementos do conjunto de dados.

Nos últimos anos a utilização de técnicas de amostragem (*sampling*) (ALOISE; CONTARDO, 2018), (WANG *et al.*, 2008) para melhorar a eficiência dos algoritmos de agrupamento, tem recebido destaque na área de mineração de dados. O método CLARA (*Clustering LARge Applications*) (KAUFMAN; ROUSSEEUW, 1990) é baseado em amostras, *i.e.*, ele separa diversas amostras do conjunto de dados e aplica o PAM em cada amostra. Porém, esse algoritmo apresenta perda na qualidade resultante do agrupamento. Assim, para lidar com essa questão o algoritmo CLARANS (*Clustering Large Applications based upon RANdomized Search*) (NG; HAN, 1994) foi projetado com o intuito de melhorar a eficiência e a eficácia do PAM e do CLARA, para isto, o algoritmo escolhe elementos aleatoriamente no passo de avaliação de escolha do melhor medóide.

A utilização de técnicas de amostragem mostrou-se especialmente útil para os métodos que realizam várias iterações considerando diferentes inicializações, como os algoritmos CLARANS e PAM. Logo, variações desses algoritmos foram propostas em (ESTER; KRIEGEL; XU, 1995) e (BARIONI *et al.*, 2008), respectivamente. A principal diferença entre os trabalhos está na estrutura de indexação dos dados empregada para o desenvolvimento das estratégias de amostragem. A abordagem apresentada em (ESTER; KRIEGEL; XU, 1995) é semelhante à abordagem apresentada em (BARIONI *et al.*, 2008) utilizada para encontrar grupos relevantes e recuperar todos os elementos de um dado grupo a partir de uma estrutura de indexação, entretanto, embora simples, esse tipo de abordagem tem mostrado excelente desempenho não apenas para a tarefa de mineração de dados, mas também para apoiar outras tarefas que são baseadas em computações de distâncias, tal como, consultas por similaridade (CIACCIA; PATELLA, 2001). Dessa maneira, esse tipo de abordagem pode ser aplicada para acelerar as consultas por similaridade, com a organização dos dados por meio de protótipos de agrupamento. Assim, os dados pré-processados podem ser utilizados por técnicas como consultas por similaridade, que utilizam o agrupamento detectado como ponto de partida para execução de seus processos de análise.

1.3 Definição do Problema

A abordagem de amostragem com o auxílio de técnicas de indexação para apoiar a detecção de agrupamento e, então, serem utilizadas por técnicas que demandam um grande número de cálculos de similaridade (*e.g.*, consultas por similaridade), pode ser dividida em quatro passos (Figura 1): **(a)** indexação, **(b)** amostragem, **(c)** detecção de agrupamento e **(d)** aplicação do agrupamento resultante sobre consultas por similaridade. O passo de indexação constrói uma estrutura em árvore dos dados de acordo com os parâmetros selecionados, a etapa de amostragem seleciona uma amostra a partir da árvore resultante do passo anterior, e a etapa de detecção de agrupamento aplicará uma abordagem de agrupamento sobre a amostra selecionada anteriormente, assim, após esse processo, os grupos poderão ser utilizados para auxiliar outras técnicas. Nas abordagens de agrupamento clássicas, os algoritmos são aplicados sobre todo o conjunto de dados. Esse processo tem como objetivo melhorar a qualidade dos agrupamentos resultantes, no entanto, apresenta maior custo computacional.

Figura 1 – Exemplificação da abordagem de amostragem com auxílio de estruturas de indexação para apoiar técnicas de detecção de agrupamento (conjunto de imagens de animais (FU *et al.*, 2018)).

Fonte: Elaborada pela autora.

Trabalhos frequentemente encontrados na literatura defendem que a utilização de técnicas de pré-processamento, tal como detecção de agrupamento, para filtrar o conjunto de dados e, auxiliar o processamento de consultas por similaridade tem um impacto significativo sobre o desempenho desse processo (ZHANG *et al.*, 2017). Por esse motivo, estudos têm sido realizados com o propósito de desenvolver abordagens eficazes e eficientes para organização dos conjuntos de dados, de tal maneira que possam representá-los de forma sumarizada para auxiliar outras técnicas que demandam um grande número de cálculos de similaridade. De modo geral, o processamento realizado com uma busca por similaridade clássica, tem como objetivo executar o menor número de comparações de elementos, reduzir o espaço de memória utilizado, construir eficientemente a estrutura de indexação e apresentar menor tempo de resposta no processamento da consulta. No entanto, quando informações extrínsecas (*e.g.*, as classes dos elementos) são adicionadas no processo de busca por similaridade, isso pode causar um aumento no espaço

de busca para retornar os elementos mais próximos que satisfaçam o conjunto de restrições de classes. Diante desses argumentos, essa tese explorou as seguintes questões de pesquisa:

- Quais elementos serão considerados para compor um conjunto de amostragem dos dados?
- Qual é o efeito no espaço de busca quando informações extrínsecas (*e.g.*, classes) são adicionadas às consultas?
- Qual o impacto de aplicar uma técnica de detecção de agrupamento para auxiliar o processamento de consultas por similaridade tradicionais e com informações extrínsecas?

A motivação considerada neste trabalho é que os dados possam ser organizados por meio de protótipos de agrupamento, com o objetivo de auxiliar as consultas por similaridade. Pesquisas na área apresentam a importância no fato da eficiência e eficácia no processamento de consultas por similaridade (MILCHEVSKI; NEFFGEN; MICHEL, 2018), (ZHAO *et al.*, 2018), (ZHANG *et al.*, 2017). Portanto, o **desafio** é: como organizar e sumarizar o conjunto de dados para apoiar consultas por similaridade tradicionais e com informações extrínsecas, nesse caso o conjunto de classes de cada elemento.

1.4 Objetivos e Contribuições

Resumidamente, o principal problema explorado nesta tese de doutorado pode ser formulado pela seguinte hipótese:

A integração de detecção de agrupamento amplia a representação semântica e acelera o processo de consultas por similaridade.

O objetivo geral desta pesquisa foi demonstrar que a abordagem baseada em amostragem para detecção de agrupamento de dados e a aplicação do agrupamento resultante para apoiar consultas por similaridade possuem potenciais para melhorar a eficiência nesse processo. Assim, para alcançar esse objetivo, metas específicas foram estabelecidas:

- Desenvolvimento de algoritmos para detecção de agrupamentos com aplicação de estruturas de indexação métricas, tais que agreguem informação semântica a partir da distribuição dos dados.
- Integrar técnicas de agrupamento de dados para apoiar o processo de consultas por similaridade tradicionais e com informações extrínsecas.

As principais contribuições deste trabalho podem ser destacadas a seguir:

- A primeira contribuição denominada `clusMAM` busca explorar os conjuntos de dados com objetivo de encontrar grupos compactos e coesos (bem separados, quando existem) presentes nos conjuntos. Dessa maneira, o algoritmo pode auxiliar com uma sugestão inicial de um possível agrupamento. Assim, os dados poderão ser observados e analisados de maneira mais rápida, para uma análise preliminar de possíveis anomalias de interesse por trás dos dados em questão. Para isso, a estratégia adotada pelo algoritmo `clusMAM` utiliza todos os elementos *representantes* da estrutura de indexação resultante, isto é, a estratégia utiliza apenas parte do conjunto de dados para fazer seu processamento. Resumidamente, a Tabela 1 apresenta as descrições técnicas da nova estratégia com relação aos algoritmos clássicos da literatura.

Tabela 1 – Comparação entre as estratégias considerando: se os algoritmos sugerem um número de grupos (n), a complexidade de tempo sobre as amostras ($\mathcal{M}$) dos conjuntos de dados e parâmetro(s) de entrada. Os detalhes sobre esses algoritmos são abordados no Capítulo 4 e no trabalho apresentado em (SOUZA; CAZZOLATO; TRAINA, 2016)

Algoritmo	# Grupos	Complexidade	Parâmetro(s)
<i>clusMAM</i>	sim	$O(\mathcal{M} ^2)$	-
PAM-Slim	não	$O(n \cdot (\mathcal{M} - n)^2)$	n
CLARANS	não	$O(n \cdot (\mathcal{M} ^2))$	n

Fonte: Dados da pesquisa.

- DE SOUZA, J.A.; CAZZOLATO, M.T.; TRAINA, A.J.M. *ClusMAM: Fast and Effective Unsupervised Clustering of Large Complex Datasets using Metric Access Methods*. In: ACM/SIGAPP Symposium on Applied Computing (SAC), Italy, p. 986—991, DOI: <<http://dx.doi.org/10.1145/2851613.2851661>>, 2016.
- A segunda contribuição denominada CCKNN busca apresentar os k vizinhos mais próximos a partir de um elemento de consulta (s_q), os quais apresentam as mesmas classes contidas no conjunto de restrição de classes $\mathbf{Q}_c$. Essa abordagem executa consultas sobre múltiplos índices. Os índices são construídos e rotulados de acordo com as combinações de classes do conjunto de dados.
 - DE SOUZA, J.A.; TRAINA, A.J.M.; MICHEL, S. *Class-Constraint Similarity Queries*. In: ACM/SIGAPP Symposium on Applied Computing (SAC), France, p. 549—556 DOI: <<https://doi.org/10.1145/3167132.3167192>>, 2018.
- A terceira contribuição denominada CfQ busca explorar a integração de técnicas de agrupamento de dados para apoiar o processo de consultas por similaridade. Esta integração proporcionará ganho de eficiência para o algoritmo CCKNN, o qual executará sobre um número menor de elementos, ou seja, o processamento de consultas por similaridade será executado sobre os grupos do agrupamento resultante. Agrupamento obtido a partir de uma estratégia de agrupamento, tal como, o `clusMAM`.

1.5 Organização do Documento

O texto deste documento está organizado da seguinte maneira:

Capítulo 2: é apresentado o referencial teórico sobre conceitos fundamentais relacionados à recuperação de objetos por conteúdo, com discussão dos aspectos relacionados à recuperação de dados multimídia, como imagens. Além disso, são apresentados conceitos de Métodos de Acessos enfatizando as principais estruturas de indexação métricas e dinâmicas.

Capítulo 3: são descritas as principais técnicas de detecção de agrupamento de dados e, alguns algoritmos que utilizam essa abordagem. O capítulo também realça aspectos de otimização e medidas de validação de agrupamento.

Capítulo 4: é apresentado um novo método com o objetivo de demonstrar a utilização das estruturas de indexação para apoiar técnicas de mineração de dados, o algoritmo `clusMAM` foi desenvolvido. A estratégia adotada por este algoritmo poderá compartilhar dos benefícios adquiridos com as estratégias de construções de MAMs, para obter ganho de eficácia nos resultados.

Capítulo 5: é apresentada uma nova abordagem para realizar consultas restritas por classes (categorias). Esse propósito de pesquisa está relacionado ao estudo da eficácia nos resultados de operações que envolvam cálculos de similaridade, pois nem sempre os resultados apresentados são os esperados. Desse modo, uma nova abordagem `CCkNN` foi desenvolvida, visando atuar para o aprimoramento da eficácia e eficiência dessas operações. A ideia geral do `CCkNN` é executar consultas por similaridade restritas, na qual o resultado de um k -NN tradicional é restrito ao conjunto (Q_c) de tipos de classes do elemento de consulta (s_q).

Capítulo 6: é apresentada uma estratégia para integração dos algoritmos `clusMAM` e `CCkNN`.

Capítulo 7: são apresentadas as conclusões e os trabalhos futuros.

REPRESENTAÇÃO E RECUPERAÇÃO DO DOMÍNIO DE DADOS

2.1 Considerações Iniciais

A representação de dados tanto tradicionais quanto complexos consideram a coleção de características que descrevem cada elemento de um conjunto de dados, e que usualmente é armazenada em um vetor de características (*feature vector*). De acordo com o domínio de dados, os elementos podem ser representados tanto por um conjunto de atributos de domínios tradicionais (*e.g.*, posições geográficas, por meio da latitude e longitude) quanto por um tipo de dado mais complexo (como no caso de dados de natureza multimídia: imagens, vídeos e áudios). Para os dados de natureza multimídia, geralmente é necessário extrair um conjunto pré-definido de características representativas e inerentes a tal tipo de dados. Essas características são utilizadas no lugar dos dados propriamente ditos no momento da realização das comparações por similaridade ([GHOSH; AGRAWAL; MOTWANI, 2018](#)). Essa abordagem tem sido utilizada, por exemplo, na área de Recuperação de Imagens por Conteúdo (*Content Based Image Retrieval* - CBIR) que costuma usar características baseadas em cor, textura ou forma para uma representação das imagens. Detalhes e conceitos de CBIR podem ser encontrados em ([ASERY *et al.*, 2017](#)), ([BHAGYALAKSHMI; VIJAYACHAMUNDEESWAN, 2014](#)). É importante mencionar que cada domínio de dados possui propriedades específicas que são utilizadas na representação dos dados. Assim, os conceitos apresentados estão relacionados ao domínio de dados considerado neste trabalho, o domínio de imagens. Com isso, as Seções [2.2](#) e [2.3](#) apresentarão exemplos diretamente referidos a esse domínio.

2.2 Extração de Características

Um dos principais componentes dos sistemas de recuperação de imagens por conteúdo é o responsável pela extração de características, pois são as características extraídas das imagens que são utilizadas para a realização da indexação e da recuperação desses dados. Esse processo consiste no cálculo de representações numéricas que podem ser utilizadas para caracterizar um determinado dado. Para o domínio de imagens é comum a realização de extração de características a partir de dados brutos, ou seja, a partir dos *pixels* da imagem. A Figura 2 exemplifica¹ o processo da representação de um tipo de domínio de dados complexos, onde o vetor de características do elemento é representado como um ponto no espaço das características.

Figura 2 – Exemplificação da representação de um dado complexo, tipo de domínio: imagens.

Fonte: Elaborada pela autora.

A importância do extrator de características está relacionada ao fato que os elementos são comparados através dos vetores de características obtidos. Dessa maneira, um elemento pode ser representado por diferentes vetores obtidos de diferentes extratores. Isso afeta diretamente as consultas por similaridade que podem apresentar diferentes resultados, conforme, a qualidade das características extraídas. As comparações entre os vetores de características são realizadas através de funções de distância que serão abordadas a seguir.

2.3 Análise de Similaridade: Funções de Distância

As funções de distância fornecem as medidas que expressam a (dis)similaridade entre pares de elementos. Desse modo, quanto mais similares dois elementos forem entre si, menor o valor dessa função, assim, uma distância igual a zero indica similaridade total. Portanto, a escolha da função de distância é importante para as aplicações de busca por similaridade, mas uma seleção adequada dessas funções é frequentemente adquirida pela experiência e habilidade do especialista na área (GAN; MA; WU, 2007). As medidas de distância podem ser divididas em duas classes conforme o valor retornado (ZEZULA *et al.*, 2010).

¹ Imagens adquiridas a partir da url <<http://wang.ist.psu.edu/docs/related/>>

- Discreta: funções que retornam um conjunto pequeno de valores, como a função de distância L_{edit} para domínios não-vetorias;
- Contínua: funções cuja cardinalidade dos conjuntos de valores retornados é muito grande ou infinita, por exemplo, a função de distância Euclidiana (L_2)

Uma família de funções de distância bem conhecida é a família *Minkowski* ou L_p , porque seus casos individuais dependem do valor numérico p , a qual pode ser definida de acordo com a Equação 2.1:

$$L_p(s_a, s_b) = \sqrt[p]{\sum_{i=1}^n |s_{ai} - s_{bi}|^p} \quad (2.1)$$

onde n é o número de dimensões e a $L_p(s_a, s_b)$ é a distância do elemento s_a ao elemento s_b , que possuem dimensionalidade igual a n .

As três funções dessa família que são amplamente utilizadas em operações de comparações por similaridade são apresentadas na Figura 3, na qual são ilustradas as formas geométricas das regiões de abrangência de acordo com a métrica L_p para o espaço bidimensional. A sobreposição das figuras distingue os subespaços pelas diferentes funções de distância:

- Para $p = 1 (L_1)$, distância Manhattan (*City-Block*): apresentada pela forma geométrica losango e obtida pelo somatório do módulo das diferenças entre os elementos.
- Para $p = 2 (L_2)$, distância Euclidiana: é comumente utilizada na literatura e, todos os pontos que estão à mesma distância de r_q compõem um círculo em um espaço bidimensional, ou uma hiper-esfera em um espaço n -dimensional.
- Para $p \rightarrow \infty (L_\infty)$, distância Máxima (*Chebychev*): o conjunto de pontos equidistantes forma um quadrado.

Uma função de distância é uma medida de dissimilaridade que apresenta um conjunto de propriedades, e para que essa função seja uma *métrica* a propriedade de desigualdade triangular deve ser satisfeita. Nesse conjunto encontram-se as seguintes propriedades:

1. Simetria: $d(x, y) = d(y, x)$
2. Não-negatividade: $0 \leq d(x, y) < \infty$
3. Reflexividade: $d(x, y) = 0$ se $x = y$
4. Desigualdade triangular: $d(x, y) \leq d(x, z) + d(z, y)$, para quaisquer x, y e $z \in \mathbb{D}$.

Figura 3 – Ilustração das formas geométricas produzidas com base nas métricas L_p , a partir do ponto central (s_q) e um raio (r_q), considerando o espaço bidimensional.

Fonte: Elaborada pela autora.

A literatura apresenta outras funções de distância, como a de *Mahalanobis* que calcula a distância de um elemento a um grupo de pontos (*i.e.*, ao centro do grupo). Um exemplo geométrico dessa função pode ser encontrado na Figura 4 e generalizações em (EVERITT *et al.*, 2011). O trabalho de (BUGATTI; TRAINA; TRAINA JR., 2008) apresenta uma avaliação experimental de um conjunto de funções de distância quando aplicadas aos vetores de características adquiridos por diferentes extratores de dados complexos. O trabalho teve como propósito investigar as dependências entre uma função de distância e um extrator de características.

Figura 4 – Representação da forma geométrica produzida pela função de distância *Mahalanobis*, considerando o espaço bidimensional.

Fonte: Elaborada pela autora.

É importante ressaltar que o custo computacional para calcular as funções de distância é alto, sendo um objetivo importante para as estruturas de indexação métricas procurar minimizar o número de distância calculadas. Portanto, dentre as diversas funções de distância apresentadas na literatura, a melhor abordagem para o domínio de dados dependerá de uma análise exploratória dos dados.

2.4 Métodos de Acesso

A literatura da área apresenta dois tipos de modelos de espaços que podem ser aplicados a domínios de dados que necessitam de comparações por similaridade: o espaço multidimensional e o espaço métrico (BARTOLINI; CIACCIA; PATELLA, 2018), (SAMET, 2005). O primeiro modelo é conhecido como multidimensional e as características consideradas são numéricas e com o mesmo número de atributos. Para este tipo de modelo pode-se encontrar os Métodos de Acesso Espaciais (MAEs), também conhecidos como Métodos de Acesso Multidimensionais, por exemplo, a k-d-B-tree, a R-tree e suas variantes (BEGUM; SUPREETHI, 2018), (AHN; MAMOULIS; WONG, 2001). Uma das áreas de aplicação que são comumente utilizadas é a da geoprocessamento. Esses métodos também pode ser aplicados a outros tipos de dados não-estruturados, como os de natureza multimídia (BöHM; BERCHTOLD; KEIM, 2001). Entretanto, a eficiência no processamento de consultas dos MAEs degrada quando o número de características aumenta (*i.e.* as dimensões) (SELLIS; ROUSSOPOULOS; FALOUTSOS, 1987).

Para o segundo tipo encontra-se o modelo de espaço métrico, onde cada vetor de características pode ter um número variado de valores ou dimensão fixa. Para esse modelo foram desenvolvidos os Métodos de Acesso Métrico (MAMs) que lidam com dados de alta dimensionalidade e adimensionais (exemplo, lista de palavras, cadeia genética e outros). A abordagem utilizada por esses métodos assume que os elementos estão em um espaço métrico levando em consideração apenas as comparações por similaridade de suas características (veja Seção 2.2). Por este motivo este trabalho está relacionado as estruturas de indexação métricas, tendo em vista, que muitos algoritmos de Mineração de Dados são baseados em funções de distância. Então, o uso de MAMs para ganho de eficiência e eficácia desses algoritmos se apresenta como um objetivo importante. Portanto, as próximas seções referem-se aos MAMs com mais detalhes.

2.5 Métodos de Acesso Métrico

A otimização de operações que envolvam cálculos de similaridade é geralmente dada pela indexação dos dados em métodos de acesso especializados. Dentre esses métodos destacam-se os que consideram o modelo de espaço métrico por serem mais robustos para a realização de consultas baseadas em similaridade.

O espaço métrico é um par $\mathbb{M} = (\mathbb{D}, d())$, onde $\mathbb{D}$ é o domínio dos elementos e $d()$ uma função de distância (ou métrica). Como mencionado anteriormente os MAMs consideram apenas a distância relativa dos elementos para organizar e particionar o espaço de busca. As funções de distância aplicadas por esses métodos devem satisfazer às seguintes propriedades (CHÁVEZ *et al.*, 2001): simetria, não-negatividade e desigualdade triangular que são formalmente descritas na Seção 2.3. A Figura 5 ilustra as propriedades apresentadas da função $d()$, sobretudo nota-se que dessas três propriedades a mais importante para realização de consultas de maneira eficiente é a de desigualdade triangular, uma vez que ela pode ser utilizada para podar o espaço de busca

na estrutura.

Figura 5 – Representação de uma função de distância fictícia. Cada seta representa a distância entre as imagens, as distâncias são todas simétricas (setas duplas) e não negativas, atribuindo o valor 0 para imagens iguais. A distância entre as imagens x e y é sempre menor que a distância das imagens x e z somada com a distância das imagens z e y (desigualdade triangular).

Fonte: Elaborada pela autora.

Os objetivos dos métodos de acesso métrico consistem em reduzir o número de cálculos de distância e o número de acessos a disco na execução de operações de consultas (*i.e.*, operações que envolvam cálculos de similaridade). Dessa forma em quase quatro décadas, diversos trabalhos foram propostos para a criação de estruturas eficientes. As BK-trees (*Burkhard-Keller-trees* (BURKHARD; KELLER, 1973) são consideradas como o marco inicial dos métodos de acesso métrico. Seus autores propuseram três estratégias para particionar o espaço métrico hierarquicamente, ou seja, o desenvolvimento de alguns MAMs foram norteados por essa pesquisa. A ideia básica dessas estruturas consiste na escolha de um elemento arbitrário central e na aplicação de uma função de distância para dividir os demais elementos em vários subconjuntos, alguns exemplos podem ser encontrados nas Seções 2.6, 2.7.

2.6 Métodos de Acesso Métrico Estáticos

Os MAMs estáticos não permitem inserções e remoções após a criação da árvore (ou estrutura), pois constroem a estrutura de indexação utilizando todo o conjunto de dados disponível em uma única operação, como a BK-tree mencionada na Seção 2.5. A partir dessa estrutura outras bem conhecidas na literatura foram desenvolvidas, por exemplo, a FQ-tree (*Fixed Queries*

Tree) (BAEZA-YATES *et al.*, 1994) e sua variante FHFQ-tree (*Fixe-Height fixed Queries Tree*) (BAEZA-YATES, 1997), mais detalhes podem ser encontrados em (SAMET, 2005). No entanto, essas estruturas foram especialmente desenvolvidas para domínio de dados cuja função de distância utilizada fosse discreta (veja Seção 2.3). Para a classe de funções contínuas foram projetadas outras estruturas e algumas dessas estruturas são apresentadas a seguir.

A VP-tree (*Vantage Point Tree*) foi projetada para funções contínuas (YIANILOS, 1993), mas segundo os autores também pode lidar com as funções discretas sem muitas modificações. Essa estrutura utiliza o particionamento de bola (*ball partitioning*) e conforme o método de particionamento adotado conduz geração de uma árvore binária balanceada. Ela trabalha recursivamente com um método de particionamento que seleciona um elemento para ser o pivô a partir do conjunto de dados em questão. O particionamento é realizado através da média da distância do pivô aos demais elementos do conjunto. Dessa maneira, a qualidade da estrutura é dependente dos pivôs escolhidos em cada etapa de particionamento. Além disso, outros problemas podem ser gerados com o uso da VP-tree, isso pode ocorrer devido a sua restrição no particionamento de dois subconjuntos em cada etapa. Assim, as consultas podem ser prejudicadas, por causa do aumento na quantidade de nós para serem visitados devido à abrangência da região de cobertura. A partir dessas lacunas, variantes foram propostas com o objetivo de melhorar o desempenho na realização de consultas. Por exemplo, o trabalho realizado em (BOZKAYA; OZSOYOGLU, 1997) apresenta uma estrutura denominada MVP-tree (*Multi-Way Vantage Point Tree*) que busca reduzir as regiões de cobertura resultantes de um particionamento. A estratégia adotada por essa estrutura utiliza uma divisão de nós com um número maior do que 02 partições, entretanto, a geração de regiões pequenas também não garante bom desempenho na busca, pois aumenta a quantidade de ramos da árvore para serem visitados.

Diante deste contexto pode-se encontrar na literatura outras estruturas e variações que buscam atingir os objetivos mencionados na Seção 2.5 (ARORA *et al.*, 2018). Por exemplo, a GH-tree (*Generalized Hyperplane*) que trabalha com a estratégia de dividir o conjunto por meio de um hiperplano (UHLMANN, 1991). A DAHC-tree (*Divisive-Agglomerative Hierarchical Clustering*) (ALMEIDA *et al.*, 2010) aplica uma abordagem de agrupamento com o objetivo de respeitar a distribuição dos dados para evitar a separação de grupos naturais presentes nos dados. No entanto, a abordagem utilizada pela DAHC-tree gera estruturas desbalanceadas e apresenta dependência de parâmetros. Além dessas, outras estruturas podem ser encontradas, como as que trabalham com distância pré-computadas com o uso de uma matriz em vez de uma estrutura em árvore, por exemplo, a AESA (*Approximating and Eliminating Search Algorithm*) (RUIZ, 1986). Como já mencionado anteriormente essas estruturas não foram projetadas para suportar inserções e/ou remoções a posteriori, pois essas aplicações podem degradar a estrutura. Desse modo, em alguns casos pode ser necessária a geração de uma nova estrutura ou uma organização inteira da subárvore. Com isso, aumenta o custo computacional, além do mais, muitas aplicações necessitam de inserções e/ou remoções dinamicamente. Logo, a partir dessas lacunas os pesquisadores da área foram motivados no desenvolvimento de métodos que fossem capazes

de suportar de maneira eficiente inserções e remoções. Então, a próxima Seção apresentará exemplos de estruturas desenvolvidas para atingirem esses objetivos de inserção e remoção de dados sob demanda.

2.7 Métodos de Acesso Dinâmicos

Nesta Seção serão apresentadas algumas estruturas que podem alterar o tamanho das árvores resultantes dinamicamente. Dessa maneira, essas estruturas dinâmicas se tornam uma vantagem para as aplicações que necessitam de remoções e inserções frequentes (HANYF; SILKAN, 2018). As estruturas apresentadas nesta seção possuem essa vantagem além de semelhanças com as descritas na Seção anterior.

Elas buscam o mesmo objetivo, ou seja, aumentar a eficiência nas operações de buscas. O trabalho de (CIACCIA; PATELLA; ZEZULA, 1997) apresenta a primeira estrutura para lidar com organizações dinâmicas com um bom desempenho, chamada M-tree. Essa estrutura fornece estratégias de inserções e divisões de nó em *overflow* com o objetivo de minimizar o raio de cobertura de cada nó da árvore. Entretanto, essa estrutura não é determinística, isto é, os elementos inseridos em diferentes ordens resultarão em árvores distintas. A partir do desenvolvimento da M-tree uma variedade de estruturas e variantes foram desenvolvidas, alguns exemplos e uma visão geral sobre MAMs podem ser encontrados em (ZEZULA *et al.*, 2010).

A literatura apresenta uma variedade de MAMs (BEGUM; SUPREETHI, 2018), por exemplo, a DSC (*Dynamic Set of Clusters*) e a DLC (*Dynamic List of Clusters*) são apresentadas em (NAVARRO; REYES, 2016), a NOBH-tree (*Non-Overlapping Balls and Hyperplanes*) (POLA; TRAINA; TRAINA, 2014), a EGNA-tree (NAVARRO; URIBE-PAREDES, 2011), a DBM-tree (*Density-Based Metric*) (VIEIRA *et al.*, 2010), a Onion-tree (CARÉLO *et al.*, 2009), a CM-tree (*Clustered Metric*) (ARONOVICH; SPIEGLER, 2007), a PM-tree (*Pivoting M-tree*) (SKOPAL; POKORNÝ; SNASEL, 2004) e a Slim-tree (TRAINA JR. *et al.*, 2002). A PM-tree é uma variante da M-tree que combina uma abordagem baseada em pivô na estrutura original. A Slim-tree é uma extensão da M-tree que foi desenvolvida com objetivo de reduzir a sobreposição entre os nós. No entanto, além do desenvolvimento de novas estruturas a literatura apresenta estratégias para melhorar o desempenho delas (CHEN *et al.*, 2017), (SKOPAL; LOKOČ, 2009), (TRAINA-JR *et al.*, 2007). O estudo de (CHEN *et al.*, 2017) apresenta um levantamento bibliográfico do impacto de estratégias para seleção de pivôs sobre técnicas de indexação baseadas em pivô. Já o trabalho de (SKOPAL; LOKOČ, 2009) encontram-se duas técnicas para a realização de inserções dinâmicas aplicadas à estrutura de indexação M-tree. Essas técnicas utilizam a estratégia de reinserções forçadas para evitar que nós folhas sejam divididos no momento em que estiverem prestes a atingirem a capacidade máxima. Em (TRAINA-JR *et al.*, 2007) pode-se encontrar a *Omni-family*, que por análise experimental mostra ganho de desempenho quando aplicada sobre as estruturas existentes, outros exemplos de estratégias são apresentados nesta seção.

Dentre os métodos apresentados aqui, deve-se observar que a Slim-tree foi desenvolvida com o objetivo de minimizar a sobreposição entre os nós, que é um dos fatores principais com relação à eficiência da estrutura para recuperação da informação. Quanto maior a sobreposição entre nós da árvore, mais degenerada ela será e poderá ser menos eficiente do que uma busca sequencial. A Slim-tree provê mecanismos para mensurar o grau de sobreposição entre os nós da árvore (*Fat-Factor*) e para reorganizar os elementos na árvore (*Slim-down*), mais detalhes dessas medidas podem ser encontrados na próxima Seção. Por meio da utilização desses conceitos, a Slim-tree permite a realização de consultas por similaridade de uma maneira eficiente e superior a M-tree. Ela busca minimizar tanto o número de cálculos de distância quanto o de acessos a disco. Embora inicialmente, qualquer MAM dinâmico possa ser utilizado na realização do trabalho apresentado, aqui utilizou-se o MAM Slim-tree como bancada de experimentação. Assim, esse MAM será descrito em mais detalhes a seguir.

Slim-tree

A Slim-tree é uma estrutura de dados em modelo de árvore dinâmica e balanceada que cresce a partir das folhas em direção à raiz (*bottom-up*). Assim como outros MAMs (*e.g.*, M-tree), ela agrupa os elementos de um conjunto de dados em páginas de tamanho fixo, sendo que cada página corresponde a um nó da árvore. A Slim-tree armazena todos os elementos nas folhas organizando-os hierarquicamente na árvore. Essa hierarquia é construída a partir da seleção de elementos, denominados elementos representantes, que definem centros de regiões no espaço de dados. Cada região possui um raio de cobertura, desse modo, apenas os elementos que forem cobertos pelo raio de cobertura de uma determinada região podem ser associados ao nó do elemento representante em questão.

Então como outras estruturas *bottom-up* (*e.g.*, B-tree), a construção da Slim-tree é realizada da seguinte maneira: para cada novo elemento a ser inserido, o algoritmo de inserção percorre a árvore a partir da raiz para encontrar um nó folha cujo raio de cobertura possa abranger o novo elemento. Se mais de um nó se qualificar, o algoritmo de inserção utiliza uma política de escolha para selecionar a subárvore mais apropriada para armazenar o novo elemento (*ChooseSubtree*). Caso nenhum nó se qualifique, seleciona-se o nó que possui o representante mais próximo do novo elemento. Esse processo é aplicado recursivamente para todos os níveis da árvore até chegar a um nó folha, onde os novos elementos são realmente inseridos. À medida que os elementos são inseridos na árvore e os nós atingem suas capacidades máximas de armazenamento, novos nós são criados e os elementos são redistribuídos entre os nós (antigos e novos) por meio da utilização de uma política de divisão de nós.

Para a construção da Slim-tree é necessário definir quais políticas de escolha de subárvore e de divisão de nós devem ser utilizadas. As opções de políticas para a escolha de subárvore disponíveis na Slim-tree são descritas a seguir.

1. **Aleatório** (*Random*): seleciona aleatoriamente um dos nós qualificados;
2. **Ocupação mínima** (*MinDist*): escolhe o nó cujo representante esteja o mais próximo possível do novo elemento, isto é, o nó que resulte na menor distância entre o novo elemento e o seu representante;
3. **Ocupação mínima** (*MinOccup*): seleciona entre os nós qualificados aquele que possui a menor taxa de ocupação. Essa opção tende a gerar árvores com uma maior taxa de ocupação dos nós o que resulta um menor número de acessos a disco nas consultas. Assim, essa é a política padrão de construção para a Slim-tree.

Para a redistribuição de elementos entre nós:

1. **Aleatório** (*Random*): os dois novos representantes dos nós são selecionados aleatoriamente e os elementos existentes são distribuídos entre eles;
2. **Mínimo dos maiores raios** (*MinMax*): todas as combinações de pares de elementos possíveis são consideradas como potenciais representantes. Para cada par, atribui-se todos os elementos a um dos representantes. O par que minimizar o raio de cobertura é escolhido;
3. **Minimal Spanning Tree** (MST) essa estratégia baseia-se na construção de uma MST dos elementos (KRUSKAL, 1956). Após a construção da MST, a aresta mais longa é removida e o elemento mais central de cada um dos dois agrupamentos resultantes é selecionado como representante do nó. Essa opção gera Slim-trees tão eficientes quanto as geradas utilizando a opção *MinMax* em uma fração do tempo. Assim, esse é o método padrão para construção da Slim-tree.

Um problema comum a todos os MAMs que utilizam o particionamento de dados em regiões em “bola” está relacionado à sobreposição dos nós. À medida que essa sobreposição aumenta, a eficiência das estruturas diminui, uma vez que todos os nós cobertos por uma região de consulta têm que ser processados durante uma operação de busca. Estudos apresentam a importância do tratamento dessa questão. Por exemplo, (SOUZA; RAZENTE; BARIONI, 2014) apresenta novas estratégias de divisão de nós que resultam em menor número de cálculos de distância com grau de sobreposição equivalente quando comparadas com as estratégias clássicas para o MAM Slim-tree. Além disso, os autores apresentam uma análise experimental de desempenho, no momento em que diferentes políticas de construção são utilizadas para apoiar técnicas de agrupamento de dados. Já, a EGNAT apresentada inicialmente na Seção 2.7 realiza a indexação dos dados usando hiperplanos em vez do particionamento em bola, dessa maneira não apresenta regiões sobrepostas. Entretanto, quando comparada com a M-tree, a EGNAT resulta um número maior de acessos a disco na realização de consultas, por outro lado, o número de cálculos de distância é menor. Para tratar essa questão, a Slim-tree possui medidas para verificação do

grau de sobreposição de uma árvore resultante e técnica pós-processamento para redução da sobreposição, apresentadas na próxima Seção.

Fat-Factor e Slim-Down

Com o objetivo de verificar a qualidade das árvores resultantes, os autores da Slim-tree propuseram duas medidas para análise de elementos em regiões sobrepostas (*i.e.*, sobreposição de nós):

- **O Fat-factor absoluto:** mensura a quantidade de elementos dentro de regiões sobrepostas geradas por nós no mesmo nível da árvore. O cálculo dessa medida é obtido pela Equação 2.2:

$$fat(T) = \frac{I_c - H \cdot N}{N} \frac{1}{M - H} \quad (2.2)$$

onde, T é uma Slim-tree, H denota a altura da árvore, M o número de nós, N o número total de elementos e I_c o número de acessos necessários para responder a uma consulta pontual na estrutura. O *Fat-factor* absoluto apresentará um valor entre 0 e 1, sendo 0 para árvores sem elementos nas regiões sobrepostas e 1 para árvores em que todos os nós se sobrepõem, ou seja, todos os elementos são cobertos por mais de uma região.

- **O Fat-factor relativo:** compara duas árvores construídas com o mesmo conjunto de dados, mas resultantes de diferentes parâmetros. Essa medida não considera a altura e o número (total) de nós da árvore. A abordagem adotada no cálculo desta medida considera a menor altura e o menor número de nós. A medida pode ser obtida a partir da Equação 2.3:

$$rfat(T) = \frac{I_c - H_{min} \cdot N}{N} \frac{1}{M_{min} - H_{min}} \quad (2.3)$$

onde, T é uma Slim-tree, $H_{min} = \lceil \log_c N \rceil$ encontra a altura mínima da árvore, $M_{min} = \sum_{i=1}^{H_{min}} \lceil N/C^i \rceil$ encontra o número mínimo de nós e C é a capacidade dos nós. O valor do *Fat-factor* relativo será maior ou igual a 0.

É importante ressaltar que o fator absoluto mensura a qualidade de uma árvore considerando apenas a sobreposição. Já o fator relativo considera a sobreposição e a ocupação dos nós, sendo uma medida para comparar duas árvores resultantes de diferentes parâmetros (*e.g.*, política de divisão de nós), mas construídas sobre o mesmo conjunto de dados. A partir do uso dessas medidas de sobreposição de nós os autores desenvolveram um algoritmo de pós-processamento para reorganização da estrutura, denominado *Slim-Down*. Ele tem como objetivo reduzir o raio de cobertura, isto é, reduzir o número de elementos em regiões sobrepostas em um mesmo nível da árvore. A Figura 6 (A) exemplifica uma indexação de 16 elementos ($s_1 \dots s_{16}$) com capacidade

de 03 elementos por nó. Os círculos pontilhados representam os nós folhas, os círculos sólidos representam os nós índices e os pontos pretos são os elementos representantes de cada nó. É importante notar que as intersecções das regiões podem aumentar dependendo dos parâmetros selecionados, nesse caso uma política de divisão de nós em *overflow* (veja Figura 6) pode aumentar ou diminuir a quantidade de elementos em regiões sobrepostas. Então para minimizar os raios de cobertura e melhorar a eficiência da estrutura (durante as operações de busca) aplica-se o algoritmo de pós-processamento *Slim-Down* ilustrado na Figura 7 (B), mas esse processo deve ser executado apenas se o valor da medida de qualidade da árvore não for desejável, isso é para evitar tempo de processamento desnecessário. No exemplo ilustrado na Figura 7 (B) apenas um elemento encontra-se sobreposto, nesse caso para calcular o *Fat-factor* absoluto antes da aplicação do *slim-down* os valores são $I_c = 11$, $H = 2$, $N = 5$ e $M = 3$, resultando um $fat = 0,2$

Figura 6 – Imagem ilustrativa da organização dos elementos de dados em uma estrutura de métodos de acesso métrico. (A) apresenta uma indexação de 16 elementos com no máximo 03 elementos por nó em uma Slim-tree e em (B) exemplifica uma sobreposição de nós antes e depois da aplicação do algoritmo de pós-processamento *Slim-Down*.

Fonte: Adaptada de Barioni *et al.* (2008).

Figura 7 – Imagem ilustrativa de uma divisão de nós. (A) ilustra a inserção de um elemento em uma página com capacidade máxima causando *overflow*, logo, a página foi dividida e a página 2 foi gerada, onde um elemento de cada página foi promovido para subir um nível e juntos geram a página 3. (B) ilustra a diferença do processo resultante após uma divisão de nós com duas políticas diferentes para os mesmos elementos, com sobreposição e sem sobreposição entre os nós.

Fonte: Elaborada pela autora.

2.8 Consultas por Similaridades

A indexação em um espaço métrico tem o objetivo de suportar eficientemente consultas por similaridade cujo propósito é recuperar elementos da base de dados que sejam similares ao elemento fornecido como exemplo para a consulta. A literatura apresenta dois tipos elementares de consulta que comparam um elemento de referência com aqueles armazenados em uma coleção de elementos (ZEZULA *et al.*, 2010): **Consulta por abrangência** (*Range query*) e **Consulta aos k -vizinhos mais próximos** (*k-Nearest Neighbor query*, k -NN). A seguir é apresentada uma breve descrição desses tipos de consulta.

- *Range query* (R_q): dado um elemento de consulta $s_q \in \mathbf{S}$ e um raio r como restrição de distância, a consulta retornará todos os elementos que estiverem dentro do raio de abrangência r de s_q . A Figura 8 (A) ilustra uma resposta desse tipo de consulta.
- *k-Nearest Neighbor query* (k -NN): dado um elemento de consulta $s_q \in \mathbf{S}$ e um k inteiro ≥ 1 , o algoritmo k -NN selecionará os k elementos indexados que tenham a menor distância do elemento de consulta s_q , de acordo com a função de distância $d()$. A Figura 8 (B) exemplifica uma resposta para esse tipo de consulta para $k=5$.

Figura 8 – Ilustração das consultas por abrangência e k -NN em um domínio bidimensional com função de distância Euclidiana. (A) apresenta um resultado de uma consulta por abrangência e (B) exemplifica um resultado de uma consulta aos k -vizinhos mais próximos.

Fonte: Elaborada pela autora.

2.9 Considerações Finais

Este capítulo apresentou conceitos fundamentais para a representação e a manipulação de imagens. Nele pode ser encontrada a ideia geral de extração das características para esse domínio

de dados. Também abordou como são realizadas as operações que envolvam comparações por similaridade, através das principais funções de distância consideradas neste contexto. Também apresentou conceitos fundamentais a respeito de métodos de acesso métrico, assim, nas Seções 2.6 e 2.7 foram apresentadas algumas vantagens e desvantagens dos principais métodos mencionados na literatura. Resumidamente, os MAMs buscam indexar com eficiência os elementos dos conjuntos de dados para reduzir o número de cálculos de distância e de acessos a disco nas operações de consultas por similaridade. Desse modo, organizar os elementos em uma estrutura da melhor maneira possível é uma tarefa difícil. Essa dificuldade se deve a vários fatores, por exemplo, a forma da distribuição dos elementos no espaço, o tamanho do conjunto de dados, a dimensionalidade, entre outros. No entanto, respeitar a distribuição dos elementos do conjunto não é um dos objetivos das estruturas de indexação métrica, principalmente por causa do tamanho fixo de página. Assim, no momento da construção das estruturas, possíveis grupos naturais presentes no conjunto podem ser separados no processo de distribuição dos elementos nas páginas (ilustração apresentada na Figura 7). É importante destacar que, o fato de ter que violar alguns critérios para satisfazer outros está relacionado diretamente às estratégias desenvolvidas por cada abordagem de indexação.

De modo geral, este capítulo mostrou que o foco das otimizações de MAMs fica centrado na eficiência no processamento de consultas por similaridade. Portanto, os MAMs não tem como tarefa principal agrupar os elementos, pois um dos parâmetros desses métodos é o tamanho fixo do nó com o objetivo de obter eficiência nas operações de consultas por similaridade. Por exemplo, reduzir a quantidade de sobreposição de nós e procurar manter a estrutura balanceada ou tratar desbalanceamento (quando permitido) para que não degrade a eficiência da estrutura. De modo geral, se as estruturas forem agrupar os elementos respeitando a distribuição desses elementos no espaço, ela pode degenerar nas operações de buscas, pois a estrutura resultante pode ser completamente desbalanceada. Logo, pode-se perder eficiência nas operações de buscas, porque geralmente os grupos são desbalanceados. No entanto, as estruturas por particionamento em bola buscam particionar os dados e, para cada partição um elemento representante é selecionado. Dessa maneira, indiretamente, os elementos são organizados em diversas partições, mas com tamanhos equivalentes. Com isso, essas estruturas se apresentam apropriadas para auxiliar tarefas de mineração de dados, tal como, a detecção de agrupamento de dados. Para esse contexto, detalhes serão abordados no próximo Capítulo.

MÉTODOS DE ACESSO E DETECÇÃO DE AGRUPAMENTO DE DADOS

3.1 Considerações Iniciais

Segundo Han e Kamber ([HAN; KAMBER; PEI, 2012](#)) o processo de detecção de agrupamento de dados pode ser definido como: *"O processo de divisão de elementos em classes (ou grupos) de maneira que, os elementos dentro de uma mesma classe apresentem alta similaridade entre si, e ao mesmo tempo, baixa similaridade em relação aos elementos das outras classes"*. A medida de similaridade é obtida a partir das características intrínsecas que descrevem os elementos de um conjunto de dados, normalmente, por meio da aplicação de uma função de distância (conceitos abordados na Seção [2.3](#)).

As estratégias de algoritmos de detecção de agrupamento apresentadas na literatura ([FAHAD et al., 2014](#)), ([AGGARWAL; REDDY, 2013](#)), ([KRIEGEL; KRÖGER; ZIMEK, 2009](#)), ([BERKHIN, 2006](#)) (veja Figura [9](#)) foram desenvolvidas para uma grande variedade de aplicações. Esta seção abordará alguns dos algoritmos que utilizam a estratégia particional para agrupar os dados, a saber ([KAUFMAN; ROUSSEEUV, 2009](#)) PAM (*Partitioning Around Medoids*), CLARA (*Clustering LARge Applications*) e CLARANS (*Clustering Large Applications based upon RANdomized Search*). O motivo da escolha por esses algoritmos partiu das vantagens e desvantagens apresentadas em cada método, com relação, a qualidade apropriada dos agrupamentos resultantes e, o custo computacional requerido. A ideia básica dos algoritmos particionais consiste em tentar encontrar o melhor conjunto de n partições ($n \leq$ número de elementos) dos dados, por meio da construção de um único nível, que divide os dados em n grupos. Cada grupo pode ser representado de duas maneiras:

- Centróide: é constituído pelo valor médio dos elementos que compõem o agrupamento, exemplo, o algoritmo *k-means* ([HARTIGAN; WONG, 1979](#));

- Medóide: é formado pelo elemento que esteja localizado o mais próximo possível do centro do agrupamento, exemplo, o algoritmo PAM (KAUFMAN; ROUSSEUW, 1990).

Figura 9 – Esquematização das estratégias utilizadas dos algoritmos de agrupamento. Os detalhes específicos de cada estratégia podem ser adquiridos em (BERKHIN, 2006)

Fonte: Elaborada pela autora.

Centróides são usados quando os dados são representados segundo o modelo de espaço multidimensional (ou vetorial), e medóides são usados com o modelo de espaço métrico. Um exemplo de algoritmo que utiliza a abordagem de particionamento baseada em centróide é o método *k-means* (*k*-médias) e seus variantes e dentre os algoritmos que utilizam a abordagem baseada em medóide os três algoritmos mais conhecidos são (KAUFMAN; ROUSSEUW, 2009):

- PAM: um dos primeiros algoritmos baseado na abordagem *k*-medóide. Ele avalia o efeito de uma substituição entre um elemento medóide por um elemento não-medóide e realoca os demais elementos aos grupos candidatos;
- CLARA: este algoritmo é baseado em amostras, isto é, ele separa diversas amostras do conjunto de dados e aplica o PAM em cada amostra;
- CLARANS: este algoritmo foi projetado com o intuito de melhorar a eficiência e a eficácia do PAM e do CLARA, para isto, o algoritmo escolhe elementos aleatoriamente no passo de avaliação de escolha do melhor medóide.

O algoritmo *k-means* e seus variantes são amplamente utilizados em várias aplicações (DUBEY; GUPTA; JAIN, 2018), (JAIN, 2010), (JAIN; MURTY; FLYNN, 1988). Este algoritmo é de fácil compreensão e apresenta complexidade de tempo de ordem $O(nkt)$ sobre o número de elementos (n), grupos (k) e iterações (t). Entretanto, o *k-means* apresenta algumas desvantagens, como: sensibilidade à presença de *outliers*, pois o algoritmo calcula a média entre os elementos de cada grupo para encontrar os seus centróides. Desse modo, ele é limitado ao tipo de atributo, porque para realizar o cálculo da média é necessário que os atributos sejam numéricos. Também possui dificuldades para encontrar grupos de formas não globulares, de diferentes tamanhos e diferentes densidades, por causa desses problemas o algoritmo pode dividir o que seria um grupo em vários. Outro problema é a inicialização aleatória, que pode obter centróides não representativos. No entanto, trabalhos recentes ainda buscam por otimizações que visam tratar os problemas apresentados pelo clássico *k-means*. Por exemplo, o trabalho de (TZORTZIS; LIKAS, 2014) apresenta uma estratégia que aborda o problema de inicialização. Por outro lado, os algoritmos baseados na abordagem *k-medóides* são mais robustos, pois consideram apenas a distância entre os elementos na fase de processamento. Entretanto, quando apresentam boa qualidade do agrupamento resultante o custo computacional é elevado, não sendo exequíveis em grandes conjuntos de dados, por exemplo, o PAM que apresenta ordem de complexidade quadrática. Diante do abordado, os algoritmos *k-medóides* que apresentam menor custo computacional são executados sobre uma amostra do conjunto de dados, dessa maneira apresentam perda de qualidade, como o CLARANS quando comparado com o PAM. Embora já tenha transcorrido 50 anos desde o desenvolvimento do primeiro algoritmo de detecção de agrupamento por particionamento, o *k-means* (JAIN, 2010), ainda existem pelo menos duas questões, que até então são exploradas pela comunidade da área: a escalabilidade dos algoritmos e a qualidade dos agrupamentos resultantes. A próxima Seção apresenta uma visão geral das estratégias que têm sido aplicadas para abordar essas questões.

3.2 Estratégias de Otimização

De modo geral, os algoritmos de agrupamentos enfrentam problemas de escalabilidade (com relação ao tempo de execução e à quantidade de memória disponível) à medida que o tamanho dos conjuntos de dados a serem analisados cresce. Com o aumento do volume de dados acumulados e a necessidade de que eles sejam analisados, esses fatores têm impulsionado a precisão do desenvolvimento de algoritmos escaláveis. Dessa maneira, vários pesquisadores concentraram seus esforços na tentativa de tornar o processo de detecção de agrupamentos exequível para grandes bases de dados (FAHAD *et al.*, 2014).

Uma estratégia amplamente utilizada para melhorar a eficiência dos algoritmos de agrupamento é a redução do número de elementos submetidos a esses algoritmos. Essa redução é obtida por meio de técnicas, tal como, amostragem (ALOISE; CONTARDO, 2018). Esse tipo de técnica mostrou-se útil para os métodos que realizam várias iterações considerando diferentes

inicializações, como o algoritmo PAM. Diante desse contexto, os trabalhos apresentados em (BARIONI *et al.*, 2008) e (ESTER; KRIEGEL; XU, 1995) propuseram variações dos algoritmos PAM e CLARANS, respectivamente. Os autores utilizaram estruturas de indexação para auxiliar na seleção de uma amostra do conjunto de dados e acelerar o processamento de detecção de agrupamento. A principal diferença dos trabalhos encontra-se na estrutura de indexação empregada, pois em (ESTER; KRIEGEL; XU, 1995) aplicou-se um método de acesso multidimensional R-tree e em (BARIONI *et al.*, 2008) os autores exploraram o uso de um método de acesso métrico para apoiar essa tarefa de Mineração de Dados.

A literatura apresenta outras abordagens de agrupamentos que aplicam estruturas de indexação para obterem melhora de eficiência e, em contrapartida eficácia. Por exemplo, (LAI *et al.*, 2012) apresenta uma análise experimental de várias abordagens para o processo de agrupamento em grandes conjuntos de dados, dentre as abordagens analisadas podem ser encontradas as estruturas de indexação: R-tree (mencionada na Seção 2.4) e SR-tree (GAEDE; GÜNTHER, 1998). O estudo de (HWANG *et al.*, 2004) apresenta uma nova abordagem para agrupamento baseado em densidade, com a utilização de uma estrutura de indexação multidimensional. A estratégia foi denominada, CF-tree (*clustering features*), na qual constrói uma árvore sobre dados sumarizados e, executa agrupamento utilizando a estrutura construída, ao invés do conjunto de dados original. De modo geral, o objetivo principal dessas estratégias é a redução da carga de processamento que os algoritmos requerem. No entanto, todos os métodos buscam por um agrupamento resultante apropriado, com relação ao domínio dos dados. Logo, é importante à aplicação de índices de avaliação de agrupamentos. Para esse propósito, pode-se encontrar várias medidas de avaliação da qualidade dos grupos (JAIN; MURTY; FLYNN, 1988), as quais serão apresentadas na próxima Seção.

3.3 Medidas de Avaliação de Agrupamento

De acordo com (JAIN; MURTY; FLYNN, 1988) a validação de agrupamentos (*cluster validation*) é considerada uma tarefa difícil dentro do processo de detecção de agrupamentos de dados, pois envolve uma cuidadosa análise exploratória dos dados. Essa tarefa verifica de maneira quantitativa a qualidade dos agrupamentos resultantes (HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2001). A verificação é obtida através da avaliação dos resultados dos índices de validade (*validity index*). Esses índices são divididos em três categorias de acordo com seus critérios de avaliação (GAN; MA; WU, 2007):

- **Critério Externo:** os índices dessa categoria utilizam informações externas da solução de um agrupamento esperado, essas informações não são consideradas no processo de detecção dos agrupamentos. Assim, eles avaliam o quanto um agrupamento obtido se aproxima da estrutura de grupos (partição ou hierarquia) conhecida *a priori* dos dados. Por exemplo, os índices mais conhecidos e utilizados são: *F-measure* (WITTEN *et al.*, 2016),

Jaccard Coefficient (XU; WUNSCH, 2008), *Rand Index Ajustado* (HUBERT; ARABIE, 1985), *Rand Index* (RAND, 1971), entre outros;

- **Critério Interno:** os índices dessa categoria não utilizam conhecimento externo para análise dos agrupamentos obtidos. A abordagem adotada por esses índices se baseia apenas nos próprios dados para avaliar o resultado. Por exemplo, os índices de compacidade e de separação dos grupos podem ser obtidos a partir de medidas baseadas puramente em distância (NGUYEN; RAYWARD-SMITH, 2008);
- **Critério Relativo:** a abordagem adotada pelos índices dessa categoria busca avaliar duas ou mais estruturas de agrupamentos obtidas por diferentes algoritmos ou pelo mesmo algoritmo, mas com diferentes parâmetros. O objetivo é determinar quais são as mais adequadas, isto é, qual estrutura é mais apropriada para um determinado conjunto de dados. Por exemplo, a família de índices *Dunn* (TRAUWAERT, 1988), *Silhouette* (ROUS-SEEUW, 1987), *Davies-Bouldin* (DAVIES; BOULDIN, 1979) e outros índices podem ser encontrados também em (VENDRAMIN; CAMPELLO; HRUSCHKA, 2010). Neste trabalho, os índices aplicados foram os critérios *Davies-Bouldin* e *Simplified Silhouette* (SS):
 - *Davies-Bouldin*: apresenta o número potencial de grupos. Se esse índice resultar um valor próximo de 0, isso indica a presença de grupos bem homogêneos e separados. O valor do índice é obtido pela equação:

$$DB = \frac{1}{k} \sum_{l=1}^k D_l \quad (3.1)$$

onde $D_l = \max_{l \neq m} (D_{l,m})$. O termo $D_{l,m}$ é a maior distância entre os grupos l -ésimo e m -ésimo, i.e. $D_{l,m} = (\bar{d}_l + \bar{d}_m) / d_{l,m}$, os valores de $\bar{d}_l$ e $\bar{d}_m$ são a distância média intragrupo para os grupos l -ésimo e m -ésimo e, $d_{l,m}$ é a distância intergrupo entre esses grupos.

- *Silhueta Simplificada*: apresenta a silhueta média de um elemento não medóide com relação ao elemento medóide do grupo vizinho mais próximo do elemento em questão. Se esse índice apresentar valores maiores do que 0 e próximos de 1, isso indica que um agrupamento apropriado foi encontrado. Diferente da silhueta média sobre todos os elementos:

$$Silhueta = \frac{1}{N} \sum_{i=1}^N s(i) \quad (3.2)$$

onde N é o número de elementos no conjunto de dados e $s(i)$ a silhueta do i -ésimo elemento, obtida pela equação $s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$. Os valores de $a(i)$ e $b(i)$ são calculados como a distância média (Euclidiana, Mahalanobis, etc) do i -ésimo elemento a

todos os demais elementos do grupo em questão, apresentando complexidade $O(N^2)$. Já, os valores de $a(i)$ e $b(i)$ da silhueta simplificada são calculados como a distância do i -ésimo elemento ao medóide do grupo em questão, apresentando a complexidade $O(N)$.

Diante do cenário colocado, pode-se considerar dois critérios para avaliar a qualidade dos resultados de processos de agrupamentos de dados (OMRAN; ENGELBRECHT; SALMAN, 2007). O primeiro critério é a solidez dos dados, ou seja, os elementos pertencentes ao mesmo grupo precisam ser similares entre si e diferentes dos elementos de outros grupos (NGUYEN; RAYWARD-SMITH, 2008). O segundo critério é a divisão dos dados, que verifica os agrupamentos resultantes com relação à separação dos grupos (HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2001). Portanto, determinar a qualidade dos agrupamentos é uma tarefa importante na área de Mineração de Dados, pois um conjunto de dados pode apresentar grupos naturais ou não. Mesmo que um conjunto não apresente grupos naturais, a tarefa de um algoritmo de agrupamento é agrupar os dados.

3.4 Considerações Finais

Este capítulo teve como objetivo mostrar conceitos fundamentais com relação aos problemas abordados na Seção 1.3, que ainda são amplamente estudados pelos pesquisadores da área. Dessa maneira, a Seção 3.2 apresentou estratégias aplicadas aos algoritmos de agrupamentos para lidar com a questão do tempo de processamento em grandes conjuntos de dados. Por exemplo, a utilização de uma amostra do conjunto de dados para acelerar os algoritmos baseados na abordagem k -medóides, ou seja, auxiliar o agrupamento com o emprego de Métodos de Acesso Métricos para selecionar uma amostra do conjunto de dados. Desse modo, os algoritmos são aplicados sobre uma parte dos conjuntos de dados, assim, a qualidade dos grupos resultantes é dependente da qualidade da amostra adquirida. Além disso, muitos algoritmos utilizam uma abordagem aleatória para selecionar uma amostra do conjunto de dados. Esse tipo de abordagem resulta em menor tempo de processamento, mas apresenta perda de qualidade. A perda de qualidade também está relacionada ao motivo das abordagens aleatórias não serem determinísticas. Deste modo, em vez de apresentar maior diversidade na seleção dos elementos, pode apresentar uma maior concentração sobre os dados de apenas uma categoria (classe) do conjunto para compor uma amostra. Assim, isso influenciaria negativamente sobre o uso do agrupamento resultante para aplicação de consultas por similaridade restritas por classes (abordagem apresentada no Capítulo 6).

De modo geral, selecionar uma amostra apropriada de maneira não supervisionada não é uma tarefa trivial, porque os grupos resultantes já perdem qualidade devido à execução ser sobre um subconjunto dos dados. Assim, não há garantia de que os representantes (medóides) escolhidos a partir da amostra são os mais adequados. Além do mais, não se sabe ao certo se o

subconjunto contém a quantidade de elementos proporcional ao número de elementos de cada categoria do conjunto de dados. Logo, para reduzir o tempo de processamento e apresentar menor perda de qualidade, alguns trabalhos discutem o uso de métodos de acesso para selecionar amostras dos conjuntos de dados. Como já discutido no Capítulo 2 os métodos de acesso dividem os dados em uma forma implícita de grupos, onde os elementos mais similares concentram-se na mesma página (nó) ou nós próximos. Por fim, com relação à qualidade dos agrupamentos resultantes são apresentadas medidas na Seção 3.3 que aplicam critérios para analisar a qualidade dos agrupamentos resultantes.

ABORDAGEM PARA AGRUPAMENTO PARTICIONAL: `clusMAM`

4.1 Considerações Iniciais

Amostragem (*sampling*) é uma estratégia comumente utilizada para melhorar a eficiência dos algoritmos de agrupamento através da redução do número de elementos submetidos a esses algoritmos, executando-os sobre uma amostra do conjunto de dados (ZHANG *et al.*, 2018), (WANG *et al.*, 2008). A utilização de técnicas de amostragem mostrou-se especialmente útil para os métodos que realizam várias iterações considerando diferentes inicializações como os algoritmos CLARANS (NG; HAN, 1994) e PAM (HAN; KAMBER; PEI, 2012). Exemplos de variações dos algoritmos CLARANS e PAM foram propostas em (ESTER; KRIEGEL; XU, 1995) e (BARIONI *et al.*, 2006). A principal diferença entre os trabalhos está na estrutura de indexação dos dados empregada para o desenvolvimento das estratégias de amostragem. O trabalho apresentado em (ESTER; KRIEGEL; XU, 1995) utiliza a R*-tree (BECKMANN *et al.*, 1990). Já o trabalho apresentado em (BARIONI *et al.*, 2006) emprega a SLIM-tree (TRAINA JR. *et al.*, 2002) no desenvolvimento do algoritmo PAM-Slim.

É importante ressaltar que a estratégia adotada pelo PAM-Slim é mais genérica, pois pode ser eficientemente aplicada para agrupar tanto conjuntos de dados multidimensionais quanto adimensionais. Além disso, esse último algoritmo também disponibiliza a possibilidade da incorporação de uma etapa adicional que leva em consideração a vizinhança dos elementos de dados no processo de detecção de agrupamentos (BARIONI *et al.*, 2008). Essa etapa adicional gera agrupamentos com qualidade superior apresentados em (BARIONI *et al.*, 2006). Com isso, a adição dessa estratégia ao algoritmo PAM-Slim provê outro recurso muito útil, que é o fato dele permitir que mais ou menos vizinhos sejam analisados de acordo com o tempo disponível. Uma propriedade interessante dessa etapa opcional é que seu custo é em função da quantidade de vizinhos que se solicita explorar.

Ainda que, as estratégias de otimização mencionadas anteriormente apresentem agrupamentos de boa qualidade, o objetivo principal dessas estratégias não é a melhora semântica do particionamento dos dados, mas sim a redução da carga de processamento que os algoritmos requerem. A principal vantagem de se utilizar uma estrutura de indexação para acelerar o processo de detecção de agrupamentos encontra-se na organização dos elementos sobre a estrutura em árvore construída (SOUZA; RAZENTE; BARIONI, 2014). Por definição, uma estrutura de indexação em árvore, tal como uma SLIM-tree divide o espaço métrico no qual estão os dados, indiretamente, em um número de grupos a partir do tamanho dos elementos e da capacidade de armazenamento do nó. Assim, o conjunto de representantes distorce o que seria um conjunto de medóides, pois o critério para se gerar os agrupamentos (os nós) não tem por prioridade a identificação dos agrupamentos. Além disso, o número de agrupamentos não é um parâmetro fornecido pelo usuário, o que faz com que os MAMs e, em particular a SLIM-tree, não possam realizar todo o processo de agrupamento dos dados. Por outro lado, os elementos representantes armazenados em algum nível da árvore, como nós índices, tendem a ser bons centros de grupos para compor o conjunto de amostras a ser usado em um algoritmo de agrupamento.

A suposição acima foi avaliada por experimentação prática que demonstrou resultados expressivos na redução do tempo de processamento dos algoritmos (BARIONI *et al.*, 2008). Durante a execução dos experimentos, verificou-se que a qualidade dos agrupamentos gerados é semelhante aos obtidos pelas técnicas de amostragem, mas, tal como aquelas, em várias ocasiões essa qualidade é inferior à dos agrupamentos obtidos por técnicas que não realizam amostragem (mas muito mais lentas e portanto, não escaláveis para grandes conjuntos de dados). Também foi possível verificar que diferentes configurações de construção da SLIM-tree influenciaram tanto o tempo gasto no processamento do algoritmo de agrupamento quanto a qualidade dos agrupamentos obtidos. Diante desse contexto, o **desafio** é: como implementar uma abordagem para detectar grupos automaticamente sobre estruturas de indexação de uma maneira eficaz e eficiente?

Como apresentado na Seção 3.2 trabalhos têm sido propostos para tratar essa questão de escalabilidade e qualidade das abordagens de agrupamentos de dados. Estes trabalhos diferem das estratégias tradicionais para realizar essa tarefa de mineração que pode ser executada sobre uma amostra do conjunto de dados. Desse modo, esse capítulo apresenta para essa etapa uma nova estratégia não supervisionada para apoiar na tomada de decisão. A técnica apresentada nesse capítulo foi publicada em:

DE SOUZA, J.A.; CAZZOLATO, M.T.; TRAINA, A.J.M. *ClusMAM: Fast and Effective Unsupervised Clustering of Large Complex Datasets using Metric Access Methods*. In: ACM/SIGAPP Symposium on Applied Computing (SAC), Italy, p. 986—991, DOI:<<http://dx.doi.org/10.1145/2851613.2851661>>, 2016.

4.2 Trabalhos Correlatos

As principais desvantagens do uso das técnicas apresentadas com a aplicação de estruturas de indexação é o fato que elas requerem como parâmetro de entrada um número de grupos presentes nos dados, assim, para explorar esse parâmetro algoritmos auxiliares podem ser aplicados para a execução desse processo. Um estudo importante que discutiu a utilização de estruturas de indexação para apoiar as tarefas de mineração de dados pode ser encontrado em (ESTER; KRIEGEL; XU, 1995). Os autores exploram a tarefa de identificação de classes com o apoio de técnicas de agrupamentos dos dados. Para essa tarefa, foram exploradas estratégias para focar em grupos relevantes e recuperar todos os elementos desse grupo com o auxílio de uma estrutura de dados multidimensional, R*-tree. Dessa maneira, focaram apenas em partes relevantes do conjunto de dados para descoberta de conhecimento. Essa analogia é muito relevante para o processamento de consultas por similaridade usando um sub-conjunto dos dados. Desde então, explorar o conjunto de dados em regiões relevantes e utilizar técnicas auxiliares para esse objetivo têm sido tema de pesquisas (HANYF; SILKAN, 2018), (KRASKA *et al.*, 2018), essa discussão será apresentada com mais detalhes no Capítulo 6. Diante das lacunas apresentadas com a aplicação de estruturas de indexação para auxiliar o processo de detecção de agrupamento, na próxima Seção é apresentado um novo método com as vantagens a seguir:

- *Não demanda parâmetro(s) de entrada:* clusMAM é capaz de sugerir automaticamente um número de grupos presente no conjunto de dados.
- *Eficiência:* A estratégia adotada pelo método é até quatro ordens de magnitude mais rápida do que os outros métodos do estado-da-arte.
- *Qualidade:* O método ganha ou empata dos algoritmos clássicos da literatura, a saber CLARANS e PAM-Slim, em relação à qualidade apropriada dos agrupamentos resultantes.

4.3 Método Proposto

Com o objetivo de demonstrar a utilização das estruturas de indexação para apoiar a detecção de agrupamento particional, o algoritmo clusMAM foi desenvolvido. A estratégia adotada por este algoritmo visa compartilhar dos benefícios adquiridos com as políticas de construções de MAMs, para obter ganho de eficiência no processamento de agrupamento particional. Esse tipo de estratégia têm como objetivo lidar com o problema de escalabilidade, através da redução de operações que envolvam cálculos de similaridade. O método clusMAM busca explorar os conjuntos de dados com objetivo de sugerir um número de grupos não sobrepostos presentes nos conjuntos. Dessa maneira, o algoritmo pode auxiliar com uma sugestão inicial de um possível agrupamento. Assim, os grupos poderão ser observados e analisados de maneira mais rápida, para uma análise preliminar. Para isso, a estratégia adotada pelo clusMAM utiliza todos os elementos representantes da estrutura de indexação resultante, isto é, a estratégia utiliza apenas

parte do conjunto de dados para fazer o processamento. Para análise experimental e discussão foram selecionados dois algoritmos de agrupamento apresentados na literatura, que são bastante utilizados e que também se beneficiam de amostras dos conjuntos de dados para tratar a questão de escalabilidade, a saber o CLARANS e PAM-Slim. Esses algoritmos apresentam grupos apropriados com menor número de cálculos de distância quando comparados com as estratégias que executam sobre o conjunto de dados inteiro. No entanto, as duas estratégias requerem um número de grupos (n) desejado *a priori* para serem executadas. De modo geral, o funcionamento do algoritmo CLARANS visa selecionar amostras aleatórias a partir do conjunto de dados fornecido e procura melhorar a qualidade dos elementos selecionados como medóides a cada iteração. O algoritmo PAM-Slim seleciona os elementos representantes da camada intermediária da estrutura de indexação (que tenha pelo menos n elementos) e aplica o PAM sobre esses elementos para seleção de medóides mais rapidamente. Com essa abordagem o PAM-Slim apresenta menor número de cálculos de distância do que o CLARANS, com qualidade dos agrupamentos resultantes equivalente.

A ideia geral do clusMAM é sugerir um número de grupos presentes no conjunto de dados. Para isso, a estratégia não utiliza outros algoritmos de agrupamento, como o PAM-Slim faz. A abordagem adotada pela estratégia parte da suposição que elementos do mesmo grupo estão no mesmo nó ou em um nó próximo. De modo geral, os resultados dos agrupamentos apresentam maior compatibilidade com a distribuição dos dados com relação a cada conjunto de dados fornecido, ou seja, um conjunto de dados pode conter c classes, mas nem sempre o mesmo número de grupos (n) apresentado por um determinado algoritmo e seus parâmetros (FÄRBER *et al.*, 2010). Em aplicações reais não sabemos antecipadamente uma forma apropriada de representar grupos para um determinado conjunto de dados, por causa da subjetividade. Assim, encontrar uma abordagem que se adeque melhor para realizar a tarefa de agrupamento é necessário uma análise exploratória dos dados, ou seja, o tipo de domínio do conjunto, a distribuição, o propósito do agrupamento ou uma análise dos resultados com abordagens diferentes (WANG *et al.*, 2018).

Nesse capítulo, tanto o CLARANS quanto o PAM-Slim utilizaram para execução os valores de grupos contidos em cada conjunto, como número desejado de n (número de grupos), além disso, esses algoritmos foram configurados com os parâmetros sugeridos na literatura. O método clusMAM sugeriu para esses conjuntos de dados o número exato de grupos com qualidade apropriada em apenas uma execução. Também como nota destaca-se que o interessante de realizar um análise exploratória com a estratégia clusMAM é que de acordo com a distribuição dos elementos e a capacidade do nó, o valor resultante será influenciado. Resumidamente, quanto mais próximos os elementos estiverem entre si (menor variância), então, a qualidade dos grupos será melhor com um tamanho de página menor. No entanto, se os elementos estiverem mais distantes entre si (maior variância), então, a exploração dos parâmetros para construção da Slim-tree é feita de modo invertido, ou seja, aumentando a capacidade do nó. Assim, pode-se observar a distribuição e os grupos resultantes de maneira mais rápida, ao invés de executar diretamente um algoritmo clássico de agrupamento e medidas de validação (variando o valor de n). A estratégia

adotada pelo `clusMAM` faz com que ele apresente esse comportamento, logo pode-se analisar os dados e verificar qual abordagem pode ser apropriada para a tarefa de agrupamento. O algoritmo não busca minimizar uma função objetivo, *i.e.* a distância intragrupo, ele verifica através dos representantes da estrutura um possível agrupamento para auxiliar na análise exploratória dos dados. Tendo em vista que os elementos serão primeiramente indexados para acelerar o processo de consultas por similaridade, então, um método que utilize os benefícios da estrutura resultante para uma exploração dos dados como o `clusMAM` é importante para acelerar esse processo.

É importante destacar que, os algoritmos apresentados na análise experimental são baseados na abordagem *hard*, cada elemento deve pertencer a apenas um grupo. A estratégia adotada pelo `clusMAM` é simples, mas efetiva e rápida, ela calcula a média e o desvio padrão da lista de distâncias mínimas construída. Essa lista contém as distâncias de todos os elementos representantes com relação ao elemento mais próximo contido na lista. O Algoritmo 1 apresenta os passos principais da estratégia proposta. O primeiro *loop* apresentado nas linhas 3 a 6 constrói uma lista de distâncias mínimas com todos os elementos representantes e o segundo *loop* apresentado nas linhas 10 a 13 constrói os grupos de acordo com o valor do limiar calculado na linha 9.

Algoritmo 1 – `clusMAM`

```

1: procedimento clusMAM(dados dos elementos do nível  $l - 1$  da estrutura resultante Slim-tree)
2:   construa uma lista de todos os elementos representantes
3:   para todo elemento  $s_i \in$  lista de elementos representantes faça
4:     encontre o elemento mais próximo
5:     construa uma lista de distância mínima
6:   fim para
7:   computar a soma das distâncias contidas na lista de distância mínima
8:   computar a média, a variância e o desvio padrão da lista de distância mínima
9:   limiar  $\leftarrow$  média + desvio padrão
10:  para todo elemento  $\in$  lista de distância mínima faça
11:    se valor da distância  $\geq$  limiar então construir grupo
12:  fim se
13:  fim para
14:  selecionar o elemento medóide para cada grupo (os elementos mais próximos do centro do grupo)
15:  atribuir os demais elementos do conjunto de dados ao elemento medóide mais próximo
16:  return o número sugerido de grupos ( $n$ ) e um agrupamento dos dados
17: fim procedimento

```

4.4 Avaliações Experimentais

Os algoritmos foram avaliados empregando conjuntos de dados sintéticos e reais. Ao todo foram 15 conjuntos de dados sendo 13 sintéticos e 02 reais, a função de distância considerada foi a L_2 (Euclidiana). A Tabela 2 apresenta uma breve descrição de cada conjunto

utilizado, na qual pode-se encontrar o nome do conjunto, a descrição dos dados, o número de elementos, o número de grupos e o número de dimensões de cada conjunto. Observe que os conjuntos em todo o capítulo serão mencionados através da seguinte representação: nome do conjunto ‘+’ o número de elementos ‘+’ o número de grupos ‘+’ o número de dimensões. Então, o conjunto sintético Dim-sets com 1024 elementos, 16 grupos e 128 dimensões se lê da seguinte maneira: *Dimsets1024E16G128D*. Para análise experimental foram considerados três parâmetros, o número de cálculos de distância, o tempo de execução e a qualidade dos grupos resultantes.

Tabela 2 – Descrição dos conjuntos de dados apresentados nesse capítulo

Conjuntos de Dados	Descrição do Conjunto	#Elementos	#Grupos	#Dimensões
Sint-ciaccia (CIACCIA; PATELLA; ZEZULA, 1997) apresentado nas Figuras (a) e (b) de 10, 11, 12 e 13	gaussianas com valor de variância $\sigma^2 = 0.01$ foram geradas sinteticamente variando o número de grupos e dimensões.	100.000	5, 10, 15 e 20	32 e 64
Dim-sets (FRÄNTI; VIRMAJOKI; HAUTAMÄKI, 2006) apresentado nas Figuras (c) de 10, 11, 12 e 13	conjuntos sintéticos variando o número de dimensões.	1024	16	128, 256, 512 e 1024
Iris (FISHER, 2018) apresentado na Figura 14	conjunto real refere-se ao tipo de planta íris e uma classe é linearmente separável a partir das outras duas.	150	3 classes, mas 2 grupos linearmente separados	4
Wdbc (WOLBERG; NICK; MANGASARIAN, 2018) apresentado na Figura 15	<i>Breast Cancer Wisconsin (Diagnostic)</i> : características computadas a partir de imagens digitalizadas de um tipo de biópsia <i>fine needle aspirate</i> (FNA)	569	2 classes, diagnóstico: benigno (B) ou maligno (M)	31

4.4.1 Resultados Obtidos

Para análise de eficiência do clusMAM as Figuras 10 e 11 apresentam os gráficos obtidos com relação às distâncias computadas e tempo de execução dos algoritmos. A Figura 10 apresenta os gráficos com relação ao número de cálculos de distância computados por cada algoritmo para os conjuntos sintéticos Sint-ciaccia e Dim-sets. Os números de grupos (n) para os conjuntos sintéticos Sint-ciaccia variam de 5, 10, 15 e 20. clusMAM apresentou menor número de computações de distância com diferentes números de grupos e dimensões (Figura 10 (a) e (b)). É importante notar que o tamanho de página influencia na execução do algoritmo, ou seja, sua qualidade é influenciada pelos parâmetros selecionados para construção da estrutura. De modo geral, o tamanho de página deve ser compatível com o número de dimensões (e.g. $64\text{dim} * 128\text{Bytes} = 8192\text{Bytes}$). Então para a realização dos experimentos, inicialmente os valores foram configurados de acordo com o número de dimensões. Após a execução do método proposto, a qualidade dos resultados foi analisada, ou seja, uma análise exploratória dos grupos resultantes (análise com diferentes medidas de qualidade). Destaca-se que o PAM-Slim também é influenciado pela configuração dos parâmetros de construção da estrutura, no entanto, ele apresenta grupos apropriados, com a execução do algoritmo PAM sobre a amostra adquirida da estrutura resultante.

Figura 10 – Número de cálculos de distância executados em cada algoritmo (os resultados do eixo-y estão em escala log). (a) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 32 dimensões, variando o número de grupos por conjunto. (b) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 64 dimensões, variando o número de grupos por conjunto. Para o conjunto sintético Sintciaccia100E10G64D, a estratégia executou quatro ordens de magnitude mais rapidamente do que o PAM-Slim. (c) gráfico referente aos conjuntos Dim-sets com 16 grupos cada, variando o número de dimensões.

As Figuras 12 e 13 apresentam os resultados de qualidade dos agrupamentos obtidos de cada algoritmo, com relação aos índices de validade (VENDRAMIN; CAMPELLO; HRUSCHKA, 2010): Silhueta Simplificada (SS) e Davies-Bouldin (DB). De modo geral, esses índices analisam a compacidade e a separação dos grupos resultantes. O SS verifica a compacidade do agrupamento resultante e um valor acima de 0 e próximo a 1 significa que uma estrutura forte de

Figura 11 – Tempo de execução de cada algoritmo. Os gráficos (a) e (b) apresentam os resultados do eixo-y em escala log). (a) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 32 dimensões, variando o número de grupos por conjunto. (b) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 64 dimensões, variando o número de grupos por conjunto. (c) gráfico referente aos conjuntos Dim-sets com 16 grupos cada, variando o número de dimensões. clusMAM apresentou menor tempo de processamento, além disso, é possível observar que conforme o número de grupos aumenta, a estratégia ainda mantém seu tempo de processamento.

agrupamento foi encontrada, ou seja, os elementos estão em grupos apropriados. O DB verifica a separação e coesão dos grupos resultantes, valores próximos de 0 indicam que o número de grupos apresentados (n) é um número favorável, então, quanto menor o valor de DB, maior a tendência dos grupos serem apropriados.

Observando a Figura 12 (a) todos os algoritmos apresentaram valores próximos de 1 para o índice SS e próximos de 0 para o índice DB. O clusMAM foi executado apenas uma vez e os valores de grupos sugeridos foram os n presentes em cada conjunto. Como mencionado anteriormente, a capacidade do nó influencia no resultado, então, a configuração inicial para construção da estrutura deve ser com relação ao número de dimensões de cada conjunto (e.g. para 64 dimensões a capacidade de página foi 8192 Bytes). Além disso, é importante realizar uma análise exploratória dos dados caso os grupos apresentados não sejam apropriados com relação às medidas de qualidade. Para isso, como um passo opcional sugere-se a realização de outras execuções variando a capacidade de página (ou nó).

A importância de uma análise exploratória aos dados está relacionada à subjetividade com relação aos grupos resultantes, *i.e.* o que pode ser considerado um agrupamento apropriado para um determinado conjunto de dados. Além disso, é necessário analisar prontamente se existe alguma estrutura de grupos presente nos dados. As Figuras 14 e 15 apresentam como o clusMAM pode auxiliar na fase de exploração inicial dos dados. A Figura 14 apresenta os resultados da execução com o conjunto Iris amplamente utilizado na literatura, o qual possui 3 classes, sendo que uma classe é sobreposta por outra. Esse conjunto apresenta 2 grupos bem separados. A Figura 15 apresenta os resultados de execuções realizadas com o conjunto Wdbc (apresentado na Tabela 2) também disponível para análise e utilizado na literatura da área, esse possui 2 classes e

Figura 12 – Qualidade resultante com relação ao índice de validação Silhueta Simplificada. É importante ressaltar que valores mais próximos de 1 apresentam uma estrutura apropriada de grupos. Nesse quesito *clusMAM* manteve equivalência aos resultados das outras estratégias. (a) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 32 dimensões, variando o número de grupos por conjunto. (b) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 64 dimensões, variando o número de grupos por conjunto. (c) gráfico referente aos conjuntos Dim-sets com 16 grupos cada, variando o número de dimensões.

Fonte: Dados da pesquisa.

Figura 13 – Qualidade resultante com relação ao índice de validação Davies-Bouldin. Valores mínimos apresentados (mais próximos de 0) indicam um número potencial de grupos presentes nos dados. *clusMAM* manteve equivalência nos resultados. (a) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 32 dimensões, variando o número de grupos por conjunto. (b) gráfico referente aos conjuntos sintéticos Sint-ciaccia com 64 dimensões, variando o número de grupos por conjunto. (c) gráfico referente aos conjuntos Dim-sets com 16 grupos cada, variando o número de dimensões.

Fonte: Dados da pesquisa.

apresenta a presença de 2 grupos bem separados nos dados.

Figura 14 – Conjunto de dados Iris com 150 elementos de 4 dimensões. O gráfico apresenta os valores sugeridos de n e o índice de qualidade variando a capacidade do nó. clusMAM apresentou $n=2$ para 512 Bytes com melhor valor do índice de qualidade. É importante notar que esse conjunto contém 3 classes, no entanto, 2 são completamente sobrepostas. A estratégia sugere a presença de dois grupos bem separados nos dados, sendo 100 elementos em um grupo e 50 em outro.

Fonte: Dados da pesquisa.

Figura 15 – Conjunto de dados Wdbc com 569 elementos de 31 dimensões. clusMAM apresentou $n=2$ para todos os valores da capacidade de nó e com qualidade equivalente. Esse conjunto é composto de 2 classes a estratégia sugere a presença de 2 grupos bem separados na distribuição dos dados, sendo 558 elementos em um grupo e os demais (11) em outro grupo mais distante.

Fonte: Dados da pesquisa.

De modo geral, a nova estratégia busca pela presença de grupos compactos e separados nos conjuntos de dados. Desse modo, sua abordagem pode auxiliar na fase de análise exploratória dos dados apresentando uma sugestão de agrupamento. clusMAM também pode auxiliar outras estratégias através da avaliação de seus resultados, ou seja, é possível verificar qual abordagem será mais apropriada para o conjunto em questão. Além disso, o valor de grupos retornado (n) pode ser submetido ao processo de agrupamento de outros algoritmos que necessitam desse parâmetro n , além de poder analisar os índices de qualidade mais rapidamente sem a necessidade de exaustivas variações de n . A Figura 16 apresenta um exemplo de resultado da estratégia adotada pelo clusMAM. A Figura 16 (a) apresenta os dados antes da execução e na Figura 16

(b) após execução com os rótulos dos grupos. O conjunto de dados utilizado neste exemplo é um Sint-ciaccia com 20.000 elementos, 2 dimensões e 100 grupos (gaussianas) de variância 0.001. A capacidade dos nós para esse conjunto foi de 512 Bytes, para essa configuração a estratégia retornou como sugestão de n o valor 95, isso é, por causa da presença de alguns grupos sobrepostos. No entanto, seus grupos retornados podem ser analisados individualmente e corroborar na tomada de decisão. É importante notar que o `clusMAM` apresenta processamento eficiente mesmo quando aumenta a quantidade de grupos, dimensões e elementos contidos nos conjuntos de dados.

Figura 16 – Exemplo no espaço bidimensional antes e após a execução da nova estratégia para o Sint-ciaccia com 20.000 elementos, 2 dimensões e 100 grupos. (a) Considerando um conjunto de dados bidimensional de 10 grupos e (b) após à aplicação do `clusMAM` com os rótulos dos grupos. Esse desempenho é mantido mesmo quando o número de dimensões e elementos são variados.

Fonte: Dados da pesquisa.

4.5 Considerações Finais

Neste capítulo foi apresentado `clusMAM`, uma nova estratégia que emprega o método de acesso métrico Slim-tree para acelerar o processo de detecção de agrupamentos de dados. No geral, algoritmos que utilizam uma abordagem relacional requerem alto custo computacional. A nova estratégia não aplica nenhum algoritmo auxiliar de detecção de agrupamento para ajudá-la no processo. Além disso, com a utilização de todos os elementos representativos de uma estrutura métrica resultante, é possível realizar uma análise exploratória nos conjuntos de dados mais rapidamente, desse modo, esse método permite uma exploração dos elementos que se desviam a partir de um padrão. Para isso, a estratégia executa seu processo com menor tempo de processamento quando comparada com os métodos que visam a mesma funcionalidade.

CONSULTAS POR SIMILARIDADE RESTRITAS POR CLASSES: CCKNN

5.1 Considerações Iniciais

Um algoritmo clássico de consulta aos **k**-vizinhos mais próximos (**k**-NN, *k-nearest neighbor*) retorna os **k** elementos com a menor distância a partir de um elemento de consulta (s_q) (ZEZULA *et al.*, 2010). No entanto, a utilização de estratégias para restringir o conjunto resultante a partir de consultas por similaridade clássicas são exploradas por pesquisadores da área (MILCHEVSKI; NEFFGEN; MICHEL, 2018), (FERHATOSMANOGLU *et al.*, 2001). A principal diferença entre os trabalhos são as restrições aplicadas sobre o espaço de busca. Por exemplo, o estudo apresentado em (MILCHEVSKI; NEFFGEN; MICHEL, 2018) explora o processamento de consultas **k**-NN com restrições de classes aplicadas a dados espaciais (ou multidimensionais apresentados na Seção 2.4), onde os elementos resultantes devem pertencer a um tipo específico, ou seja, os elementos do conjunto resposta desse tipo de consulta deve pertencer ao mesmo conjunto de classes do elemento de consulta (s_q). Entretanto, nesse trabalho os autores não variam o número de dimensões do conjunto de restrições Q_c do elemento (s_q), além disso, eles exploram apenas com estruturas multidimensionais. Já, o trabalho apresentado em (FERHATOSMANOGLU *et al.*, 2001) introduz uma série de métodos que aplicam limitações de regiões (*range constraints*) sobre consultas aos **k**-vizinhos mais próximos. Os autores desenvolveram métodos para avaliar esse tipo de consulta com diferentes propriedades e vantagens. No entanto, esses métodos desenvolvidos não aplicam informações externas (características extrínsecas dos dados) como restrições, tal como as classes dos elementos.

Apesar das consultas aos **k**-NN e por abrangência (*Range query*) já terem sido amplamente discutidas na literatura da área, no entanto, ainda existem questões em aberto sobre as implicações no emprego de restrições sobre consultas por similaridades. Isso é devido a versatilidade das aplicações que podem se beneficiar com esse tipo de consulta. Por exemplo,

considere que *Maria* esteja visitando São Paulo, enquanto ela explora os lugares turísticos, *Maria* gostaria de almoçar em um restaurante italiano. Entretanto, ela está viajando em modo econômico, então, o restaurante deve ser acessível. Portanto, para auxiliar *Maria*, uma aplicação precisaria encontrar os k mais próximos restaurantes italianos, que apresentam boa avaliação e, não sejam custosos. Assim, é apresentada nesse capítulo uma nova abordagem denominada CCk-NN (*Class-Constraint k-NN*). Esse método explora o problema de executar consultas aos k -NN, de modo que, o conjunto resultante deveria adicionalmente ser restrito a um tipo específico de classes. À aplicação CCk-NN não é restrita a dados geoespaciais, mas esse tipo de dados é um dos mais ilustrativos. Diante do exposto, a abordagem apresentada aqui aplica características extrínsecas (*e.g.* classes dos elementos) sobre a fase de construção da estrutura de indexação para executar consultas restritas por classes, com relação a um conjunto de classes específico ($\mathbf{Q}_c$), a partir de um elemento de consulta (s_q). Essa abordagem foi publicada em:

DE SOUZA, J.A.; TRAINA, A.J.M.; MICHEL, S. *Class-Constraint Similarity Queries*. In: ACM/SIGAPP Symposium on Applied Computing (SAC), France, p. 549—556 DOI: <https://doi.org/10.1145/3167132.3167192>, 2018.

5.2 Definição do Problema

A abordagem apresentada nesse capítulo visa explorar a seguinte questão de pesquisa: “Quais métodos de indexação podem ser considerados para avaliar consultas restritas por classes?”. Mais especificamente, assume-se como entrada: um conjunto $\mathbf{S}$ de elementos e uma medida de dissimilaridade expressada como uma função de distância métrica $d(s_i, s_j)$, onde s_i e s_j são elementos de um conjunto de dados $\mathbf{S}$. De modo geral, cada elemento $s \in \mathbf{S}$ está associado a um conjunto de classes. Cada classe possui uma forma $\langle \text{atributo} : \text{valor} \rangle$, por exemplo, classes podem ser combinadas $\langle \text{produto} : \text{notebook} \rangle, \langle \text{qualidade} : \text{boa} \rangle$, no qual este exemplo descreve *notebooks bons*. Para uma classe $\langle a : v \rangle$, o atributo “a” refere-se a uma dimensão de classe $\mathcal{C}$ do conjunto de dados e, “v” a uma classe dessa dimensão $\in \mathcal{C}_i$. Neste trabalho, assume-se que um elemento pertence apenas a uma classe do conjunto de classes $\mathcal{C}_i$ para cada dimensão de $\mathcal{C}$. Assim, não existe sobreposições entre as classes de uma dimensão (*e.g.* no caso anterior o elemento é um produto com uma qualidade). Portanto, o objetivo é computar os k -NN elementos a partir de um elemento de consulta (s_q) que apresente todas as classes do conjunto de restrições de classes $\mathbf{Q}_c$.

Na Figura 17 é ilustrado o problema geral dessa pesquisa, isto é, o impacto sobre o espaço de busca que precisa ser investigado para responder uma consulta aos k -NN restrita por classes. Assume-se que uma estrutura de indexação (ver Seção 2.5) é construída sobre o conjunto de dados apresentado na Figura 17. A consulta ilustrada nesse exemplo deseja os 3 elementos mais próximos com relação ao conjunto de restrições $\mathbf{Q}_c = \{\mathcal{C}_1 = \{\text{espécie:iris-virginica}\}\}$. Se 99% dos elementos são da espécie iris-virginica, intuitivamente apenas $(1/0.99) \times k \approx 1.01 \times k$

elementos mais próximos precisam ser recuperados para encontrar os k iris-virginica. No entanto, se ao contrário disso, apenas 1% dos elementos são iris-virginica, assim uma média de $100 \times k$ elementos precisam ser recuperados/computados e verificados para cada membro do conjunto de classes.

Figura 17 – Ilustração do impacto sobre o espaço de busca, com a aplicação das restrições de classes sobre uma consulta k -NN tradicional, considerando o valor de $k=3$ com relação ao elemento s_q de $\mathbf{Q}_c=\{\mathcal{C}_1=\{\text{espécie:iris-virginica}\}\}$. (a) k -NN clássico e dados não rotulados, (b) k -NN e dados rotulados e (c) após à aplicação do algoritmo com restrição de classes sobre os dados rotulados.

Fonte: Elaborada pela autora.

Agora, assume-se um conjunto de dados composto com exames de mamografia¹, o qual contém $|\mathcal{C}|=2$ dimensões de classes. O conjunto da primeira dimensão de classe apresenta a lesão encontrada na imagem $\mathcal{C}_1=\{\text{benigna}\}$ ou $\{\text{maligna}\}$, já a segunda dimensão apresenta o tipo de lesão $\mathcal{C}_2=\{\text{calcificação}\}$ ou $\{\text{massa}\}$. A ideia óbvia seria construir uma estrutura de indexação para cada classe. Por exemplo, considere apenas a dimensão de classe referente a lesão ($\mathcal{C}_1$), se cada elemento pertence apenas uma classe, então, não existiria nenhum *overhead* de armazenamento. Desse modo, se $|\mathbf{Q}_c|=1$ (conjunto de restrições de classes contendo uma dimensão de classe), apenas um índice precisa ser acessado. No entanto, no geral $|\mathbf{Q}_c| \geq 1$. Assim, um elemento pode ter múltiplas classes (ou seja, um tipo de classe por dimensão de classe), desse modo, haverá redundância sobre os índices, no qual um elemento pode ser indexado em mais de uma estrutura de indexação. Logo, assume-se que cada elemento pertence apenas uma classe para cada conjunto de dimensão de classe $\mathcal{C}_i$. Portanto, “*Quantos índices serão construídos?*”, considerando $|\mathcal{C}|=2$ dimensões de classes com $|\mathcal{C}_i|=2$ classes em cada dimensão, assim, para responder essa questão, temos: $\mathbf{T} = [(2+1).(2+1)] - 1$ (ilustração apresentada na Figura 18). Para isso, o número de índices com combinações distintas de classes, pode ser adquirido pela Equação 5.1:

$$1 \leq |\mathbf{T}| \leq \left[\prod_{i=1}^{|\mathcal{C}|} |\mathcal{C}_i| + 1 \right] - 1 \quad (5.1)$$

¹ <http://marathon.csee.usf.edu/Mammography/Database.html>

onde o termo -1 refere-se ao conjunto vazio ($\{\}$) de combinações. Desse modo, para executar uma consulta considere a seguinte questão: “*O quão redundante essas combinações de classes são?*”, dado um elemento com $|\mathcal{C}|$ dimensões de classes. Portanto, está claro que todos elementos que compartilham a mesma classe de uma dimensão, esses também estarão contidos nas classes que compartilham $|\mathcal{C}| - 1$ dimensão. Por exemplo, observe a Figura 18 (b) todos os elementos contidos na combinação (1) bg, também estarão nas estruturas com combinações (5) bg+ms e (6) bg+cl.

Figura 18 – Ilustração das combinações de classes dos diagnósticos de exames de mamografia. (a) $|\mathcal{C}| = 2$ dimensões de classes e para cada dimensão $\mathcal{C}_i = 2$, (b) apresenta todas as combinações de classes sem sobreposições de classes da mesma dimensão e, para cada combinação um índice será construído.

(a) Dimensão de classe 01: lesão	(b) Combinações distintas de classes (9-1 indexes)
benigno → bg	(1) bg (5) bg + ms
maligno → mg	(2) mg (6) bg + cl
Dimensão de classe 02: tipo de lesão	(3) ms (7) mg + ms
massa → ms	(4) cl (8) mg + cl
calcificação → cl	

Fonte: Elaborada pela autora.

Diante das questões mencionadas na Seção anterior, esse capítulo foi guiado pela suposição que é possível aplicar restrições de classes sobre um **k**-NN tradicional e, ainda obter um tempo de processamento apropriado com aplicação de múltiplas estruturas. Para isso, essa pesquisa apresenta uma nova abordagem para apoiar consultas restritas por classes, a qual apresenta as seguintes vantagens:

- *Eficiência*: múltiplos índices são construídos para a execução das consultas restritas por classes, considerando uma hierarquia de classes;
- *Qualidade*: o conjunto resultante é restrito a um conjunto $\mathbf{Q}_c$ a partir de um elemento de consulta s_q . Nesse contexto, pode-se esboçar algumas observações a partir da ilustração apresentada na Figura 17, onde apresenta uma consulta **k**-NN clássica para $\mathbf{k} = 3$ sobre um conjunto de (a) dados rotulados e (b) não rotulados. Em (c) apresenta o resultado de uma consulta **k**-NN restrita por classes, considerando uma amostra em espaço bidimensional do conjunto de dados Iris (FISHER, 2018) amplamente utilizado na literatura.

5.3 Método Proposto

Com o objetivo de aplicar restrições de classes sobre uma consulta aos **k**-NN, o método CCKNN foi desenvolvido. A estratégia adotada pelo método busca reduzir o número de cálculos

de distância durante o processamento das consultas restritas por classes. O algoritmo 2 apresenta os passos principais do método proposto. Na linha 1, onde são apresentados os parâmetros de entrada: k o número de vizinhos, s_q um elemento de consulta, Q_c conjunto de restrições de classes referente ao elemento de consulta s_q e T conjunto de árvores; destaca-se que o parâmetro Q_c influencia na escolha das estruturas resultantes para execução das consultas restritas. No geral, o conjunto resposta de uma consulta restrita apresentará os elementos mais próximos com relação ao elemento de consulta s_q , mas que sejam das mesmas classes do conjunto de restrições Q_c . A abordagem apresentada aqui procura por uma estrutura apropriada $\in T$ (conjunto de árvores) para a execução das consultas restritas por classe. Assim, CCKNN pode evitar consultas sobre um alto número de elementos, dessa maneira, reduz o número de cálculos de distância no processamento dessa tarefa. O *loop* (nas linhas 4—8) busca por uma árvore apropriada, *i.e.*, com as mesmas dimensões de classes do conjunto Q_c .

Algoritmo 2 – CCKNN

```

1: procedimento CCKNN( $k, s_q, Q_c, T$ )
2:    $R \leftarrow \{\}$ 
3:   para todo  $t \in T$  faça
4:     se  $Q_c \subset C_t$  então
5:       Encontre os  $k$  elementos mais próximos a partir do elemento de consulta  $s_q$ 
6:        $R \leftarrow k$ 
7:     fim se
8:   fim para
9:   se  $R \neq \emptyset$  então
10:    return  $R$                                  $\triangleright$  os  $k$ -NN restritos pelo conjunto de classes  $Q_c$ 
11:   fim se
12: fim procedimento

```

Diante do contexto, para selecionar uma estratégia de indexação apropriada para o processamento de consultas restritas, dois conjuntos de experimentos foram analisados. Primeiramente, o número de cálculos de distância foi computado/analísado na fase de construção das estruturas de índice e durante o processamento das consultas. Além disso, como segundo parâmetro experimental, para explorar as estruturas de indexação construídas, o tamanho de memória é estimado em *megabytes* com relação a todas as estruturas geradas a partir das combinações de classes. Para isso, um índice invertido e um método de acesso métrico representativo foram aplicados. Essas estratégias de indexação podem ser consideradas esquemas representativos para o processamento de consultas por similaridade:

Índice invertido emprega *buckets* que armazenam elementos, os quais possuem uma combinação de classes e um conjunto de vetores de características que permite uma rápida recuperação de elementos individuais com relação às suas classes.

Método de acesso métrico Slim-tree uma estrutura de dados dinâmica e balanceada (detalhes podem ser encontrados na Seção 2.7). Aqui os experimentos foram executados sobre um algoritmo **k**-NN, no qual aplica uma fila de prioridade global com o uso de uma estrutura *heap* dinâmica, com o objetivo de melhorar o desempenho (HJALTASON; SAMET, 2000).

Indexação múltipla para combinações de classes com relação ao problema apresentado anteriormente na Seção 5.2, uma estrutura de indexação métrica será construída para cada combinação de classes e, um índice invertido será construído para cada conjunto de combinações das dimensões de classes. Por exemplo, dado um conjunto de classes com $|\mathcal{C}| = 2$ dimensões e $|\mathcal{C}_i| = 2$ classes por dimensão, o número de índices métricos para construir será 8. Entretanto, o número de índices invertidos será o número de dimensões de classes, neste caso, apenas 2 índices invertidos serão construídos. É importante mencionar que as combinações de classes foram construídas por um algoritmo recursivo garantindo a distinção entre as combinações.

5.4 Avaliações Experimentais

Para análise de eficiência os métodos de indexação mencionados na seção anterior foram avaliados, com relação a média de cálculos de distância e tempo de processamento das consultas. Os métodos de indexação considerados nessa fase experimental foram: busca sequencial, índice invertido e um método de acesso métrico, Slim-tree (neste capítulo as estruturas foram executadas em memória). Os resultados apresentam a média de 100 consultas **k**-NN restritas por classes, onde para cada valor de **k** é aplicado um conjunto distinto de centros de consultas selecionados aleatoriamente. Neste documento é apresentado os principais resultados, com relação à abordagem apresentada nesse capítulo. Portanto, é importante destacar que o tempo gasto para executar uma consulta é uma medida básica, porém importante quando avalia-se uma estrutura de indexação (*i.e.* aqui tempo está diretamente relacionado as computações de distâncias).

5.4.1 Descrição dos conjuntos de dados

Diferentes conjuntos de dados sintéticos com dimensões de classes diferentes foram aplicados para validação do método. Além disso, os resultados apresentados foram também avaliados com conjuntos de dados reais:

- **Dados sintéticos** foram gerados variando o número de dimensões de classes a partir de 2 até 4 com 2 ou 3 classes por dimensão. Os sintéticos serão referenciados neste capítulo desse texto, seguindo a regra de construção: S1E2D3DC2C para cada conjunto de dados com 1.000.000 de vetores de características compostos por 2 dimensões, com $|\mathcal{C}| = 3$

Figura 19 – Exemplo considerando um espaço bidimensional referente aos conjuntos de dados sintéticos utilizados nos experimentos.

Fonte: Dados da pesquisa.

dimensões de classes e para cada dimensão de classe $|\mathcal{C}_i| = 2$. Para isso, estes conjuntos foram gerados seguindo os parâmetros apresentados em (CIACCIA; PATELLA; ZEZULA, 1997), considerando uma variância de $\sigma^2 = -0.1$ para distribuição uniforme (Figura 19 (a)) e $\sigma^2 = 0.1$ para distribuição gaussiana (Figura 19 (b)).

- **Conjunto real** de imagens obtidas a partir de exames de mamografias (ver Figura 20), onde são categorizadas com relação a 4 tipos de visões de mamografia, esses tipos são: LCC (*Left CranioCaudal*), RCC (*Right CranioCaudal*), LMLO (*Left MedioLateral Oblique*) e RMLO (*Right MedioLateral Oblique*). Portanto, para a construção dos conjuntos reais utilizou-se o histograma normalizado, com 32 e 256 dimensões (NIXON; AGUADO, 2012). Desse modo, cada vetor de característica do primeiro conjunto é composto por 32 características (ou dimensões) e o segundo por 256 características. Esses dados contém 02 dimensões de classes e, cada dimensão apresenta 02 classes: lesão (benigno e maligno) e tipo de lesão (calcificação e massa). A abordagem apresentou o mesmo comportamento variando o número de elementos e dimensões. Informações mais detalhadas sobre esse conjunto de dados podem ser encontradas em (OLIVEIRA *et al.*, 2017).

Figura 20 – Uma amostra do conjunto de imagens de mamografias.

Fonte: Dados da pesquisa.

5.4.2 Resultados Obtidos

Os gráficos apresentados nessa seção mostram o comportamento das estruturas, com relação à análise de custo para executar consultas restritas por classes. A Figura 21 apresenta os resultados relacionados às consultas restritas por classes sobre uma estrutura de indexação com todos os elementos. A Figura 21 (a) apresenta os resultados com relação ao conjunto de dados S1E2D2CD2C com $|Q_c|=2$, (b) para o conjunto de dados S1E2D3DC3C com $|Q_c|=3$ e (c) para o conjunto de dados S1E2D3DC4C com $|Q_c|=3$. Pode-se observar que o método de acesso métrico apresentou aumento no tempo de processamento de até 63.06% na Figura 21 (b) para $k=90$ (eixo- x do gráfico) no uso de apenas uma estrutura. É importante notar que, o índice invertido apresenta os melhores valores quando o número de dimensões de classes aumenta. Este comportamento é explicado, pelo fato que, o número de elementos sobre os *buckets* de um índice invertido será menor. Assim, como esperado, quando o número de dimensões de classes $|Q_c|$ apresenta o mesmo valor de dimensões de classes do conjunto de dados, um método de acesso métrico apresenta maior custo para computar consultas restritas por classes do que um índice invertido, considerando todos os elementos indexados sobre uma única estrutura. Desse modo, essa característica guiou as próximas análises experimentais. Portanto, é importante destacar que a busca sequencial foi utilizada no primeiro conjunto de experimentos, como um *baseline*. Desse modo, a abordagem apresentada neste capítulo não foi aplicada sobre esse tipo de estrutura de indexação.

A Figura 22 apresenta os resultados obtidos, com relação à execução de consultas sobre uma estrutura rotulada, *i.e.* os elementos foram indexados de acordo com suas classes. Assim, a abordagem de indexação múltipla apresenta índices com elementos mais específicos às suas classes, onde são construídos a partir das combinações de classes. Com o propósito de realizar uma comparação justa, os conjuntos de restrições de classes (Q_c) dos elementos de consultas (s_q) foram selecionados com o mesmo número de dimensões de classes, considerado na fase de construção das estruturas de indexação. Por exemplo, se o Q_c de s_q apresenta $\{\mathcal{C}_1=\{\text{lesão:maligna}\}, \mathcal{C}_2=\{\text{tipo de lesão:benigna}\}\}$, as consultas restritas por classes (CCKNN) podem ser executadas sobre as estruturas com essas combinações de classes. Com o objetivo de lidar com esses problemas, foram analisados o comportamento das estruturas considerando a abordagem apresentada na Seção 5.3.

Observe que os gráficos apresentados nas Figuras 22 (a), (b) e (c) mostram o mesmo comportamento dos resultados apresentados nas Figuras 21 (a), (b) e (c). O método de acesso métrico foi uma das estruturas que apresentou melhor eficiência. Com isso, os gráficos apresentados na Figura 22 mostram a comparação entre o método de acesso métrico e índice invertido para avaliar os resultados referentes aos índices rotulados, com relação às combinações de classes. Esse conjunto de experimentos foi conduzido com objetivo de avaliar o impacto no processamento de pequena e grande quantidade de classes no conjunto de restrições (Q_c) do elemento de consulta s_q .

Particularmente, a respeito do comportamento do método de acesso métrico quando comparado ao índice invertido, é importante notar que, quando aumenta o número de dimensões de classes e o número de classes por dimensão, o tempo para executar consultas restritas por classes é reduzido quando aplicadas sobre um método de acesso métrico. Por exemplo, considere o conjunto de dados S1E2D3DC4C (veja Figura 22 (c)), o índice invertido apresentou um aumento no tempo de processamento de 92.62% para $k=50$ (com aproximadamente uma ordem de magnitude maior do que o método de acesso métrico), em contraste a Figura 21 (c) para $k=50$ o índice invertido demandou 18,13% do custo utilizado pelo método de acesso métrico. Assim, pode-se observar que, quando os elementos são indexados sobre apenas um método de acesso métrico e, deseja-se executar consultas restritas por classes, esse procedimento apresentará maior número de cálculos de distância, ou seja, maior tempo total gasto para computar consultas baseadas em distância. Para esse caso, o índice invertido apresenta resultados melhores se o número de dimensões de classes $|C|$ for igual o número de dimensões de classes do conjunto de dados. Entretanto, o número de $|C|$ pode ser diferente, dessa maneira, para executar restrições de classes sobre os *buckets* do índice invertido pode impactar negativamente tanto quanto executar esse tipo de consulta sobre um método de acesso métrico considerando todos os elementos e classes indexados em apenas uma estrutura.

Outro ponto importante, no geral, o método de acesso métrico foi uma das estruturas que apresentou os melhores resultados, com relação às execuções de consultas restritas por classes utilizando múltiplos índices com o mesmo número de dimensões de classes do conjunto de restrições Q_c (apresentados nas Figuras 22 e 23). Por exemplo, considere o conjunto S1E2D3DC3C (Figura 23 (c)) para $k=60$, quando comparado com o índice invertido, o custo é reduzido em até 98,91%. Assim, observando os resultados alcançados com todos os experimentos executados, é possível concluir que, a estratégia aplicada sobre um método de acesso métrico e índice invertido apresentaram o mesmo comportamento tanto para conjuntos sintéticos quanto para conjunto real (como apresentado na Figura 24). Nesse documento, foram apresentados os principais resultados obtidos, com relação aos parâmetros experimentais.

Figura 21 – Tempo total das consultas CCKNN executadas sobre uma estrutura de indexação com todos os elementos do conjunto de dados. (a) Valores referentes ao conjunto S1E2D2DC2C para $|Q_c| = 2$, (b) Valores referentes ao conjunto S1E2D3DC3C para $|Q_c| = 3$ e (c) Valores referentes ao conjunto S1E2D3DC4C para $|Q_c| = 3$. O eixo y (em escala $\log$) apresenta a média de 100 consultas para cada valor de k (no eixo x) utilizando diferentes conjuntos compostos de centros de consulta selecionados aleatoriamente.

Fonte: Dados da pesquisa.

Figura 22 – Consultas restritas por classes sobre estruturas específicas (rotuladas), com relação às classes do elemento de consulta s_q . (a) Valores referentes ao conjunto S1E2D2DC2C para $|C| = 2$ e estruturas construídas com diferentes combinações de 2 dimensões de classes, (b) Valores referentes ao conjunto S1E2D3DC3C para $|C| = 3$ e estruturas construídas com diferentes combinações de 3 dimensões de classes, (c) Valores referentes ao conjunto S1E2D3DC4C para $|C| = 3$ e estruturas construídas com diferentes combinações de 3 dimensões de classes, mas com 4 classes diferentes por dimensão.

Fonte: Dados da pesquisa.

Figura 24 – Consultas aos vizinhos mais próximos restritos por classes, sobre estruturas específicas (rotuladas). Os valores são referentes ao conjunto de dados de imagens de exames de mamografia (*Left CranioCaudal*) com 256 características e $|Q_c| = 2$. As estruturas de indexação foram construídas com diferentes combinações de classes considerando $|C| = 2$. O eixo-y (em escala $\log$) apresenta os resultados com referência à média de 100 consultas k -NN restritas por classes para cada valor de k (no eixo-x) utilizando distintos conjuntos de centros de consulta selecionados aleatoriamente.

Fonte: Dados da pesquisa.

Figura 23 – Consultas restritas por classes sobre estruturas específicas (rotuladas), com relação às classes do elemento de consulta s_q . (a) Valores referentes ao conjunto S1E2D3DC4C para $|Q_c|=1$ e estruturas construídas com diferentes combinações considerando 1 dimensão de classe, (b) Valores referentes ao conjunto S1E2D3DC2C para $|Q_c|=2$ e estruturas construídas com diferentes combinações considerando 1 dimensão de classe, (c) Valores referentes ao conjunto S1E2D3DC3C para $|Q_c|=3$ e estruturas construídas com diferentes combinações considerando 1 dimensão de classe

Fonte: Dados da pesquisa.

5.5 Considerações Finais

Neste capítulo foi apresentada a abordagem CCkNN para lidar com o problema de restrições de classes em combinação com o algoritmo k -NN. A combinação de restrições às características intrínsecas dos dados é uma atividade importante, pois muitas vezes apenas o uso de características intrínsecas dos dados não é suficiente para identificar padrões/características importantes nos conjuntos analisados. No entanto, combinando com informações extrínsecas (*e.g.* classes dos elementos) os resultados de custo para cada consulta por similaridade podem ser melhorados. Isso é devido ao fato que frequentemente a percepção do usuário considera a resposta em um dado contexto.

De modo geral, quando se pensa no suporte à eficiência de consultas restritas por classes sobre grandes conjuntos de dados, é importante fornecer não apenas estratégias para acelerar o tempo de construção (SOUZA; RAZENTE; BARIONI, 2013) das estruturas, mas também empregar estratégias que não degradem a eficácia das consultas por similaridade. As estruturas de indexação analisadas neste capítulo, com relação à indexação de conjuntos de dados rotulados, lidam com essa questão. Nesta pesquisa, avaliou-se estruturas de indexação de dados complexos sobre conjuntos de dados com múltiplos tipos de classes (≥ 1 dimensão de classes). Foram aplicados conceitos de análise combinatorial (ROBERTS; TESMAN, 2009) (tais como, combinações de dimensões de classes para criar subconjuntos distintos de classes) para a construção de múltiplas estruturas de indexação. As estruturas foram guiadas pela observação que se pode manter todas as classes de cada elemento sobre uma estrutura de indexação sem degradar a fase de construção para aplicação de consultas restritas por classes.

CONSULTA POR SIMILARIDADE RESTRITA APOIADA POR TÉCNICA DE AGRUPAMENTO: CfQ

6.1 Considerações Iniciais

De modo geral, a execução de consultas por similaridade em dados complexos têm seus próprios desafios e problemas. As operações de cálculos de similaridade são geralmente executadas sobre Métodos de Acesso Métrico. Esses métodos têm sido desenvolvidos para manipular eficientemente grandes conjuntos de dados complexos (detalhes apresentados na Seção 2.5). Neste cenário, abordagens têm sido propostas para melhorar a eficiência e a escalabilidade no processamento de consultas por similaridade (como apresentado no Capítulo 5). Além da eficiência no processo de recuperação, é importante destacar que a eficácia é um grande desafio em aplicações reais. Esse fato, é por causa da intenção e percepção do usuário na especificação da consulta e, na resposta, encontram-se em um dado contexto, que pode não atender ao esperado pelo usuário. Dessa maneira, como a organização dos dados é realizada nos Métodos de Acesso Métrico (MAMs), visando a agilização de consultas, apresentam uma boa oportunidade para explorar a informação adicional contida nos conjuntos de dados. Por exemplo, a aplicação de técnicas de Mineração de Dados (ex., detecção de agrupamento de dados) na fase de pós-processamento sobre estruturas de indexação (como apresentado no Capítulo 4).

A literatura apresenta abordagens que exploram as vantagens de informações adicionais sobre os conjunto de dados, por meio da exploração e consideração dos relacionamentos entre cada elemento do conjunto (ZHANG *et al.*, 2017). Desse modo, para abordar essas questões, esse capítulo apresenta a importância de ambas (i) a informação adicional, com relação o relacionamento entre os elementos; e (ii) a eficiência no processo de recuperação. Com o objetivo de apresentar maior flexibilidade ao usuário na definição de consultas sobre diferentes

regiões selecionadas para a busca (*e.g.* pesquisar dentro de um grupo específico). Aqui visou-se melhorar o desempenho de consultas por similaridade, reduzindo o número de operações de comparações e quantidade de dados acessados durante à execução de uma consulta. Assim, a abordagem apresentada aqui, foi guiada pela hipótese que elementos mais similares e/ou das mesmas classes estão no mesmo grupo ou em um grupo próximo, adquiridos a partir do agrupamento resultante de uma estrutura de indexação. Portanto, aplicando restrições de classes sobre uma consulta por similaridade dentro de um grupo específico, o conjunto resposta pode ainda apresentar os **k** elementos mais similares e, também, com um conjunto de classes similar. Desse modo, esse capítulo apresenta a abordagem CfQ (*Clustering for Querying*) para acelerar as consultas restritas por classes (problema abordado no Capítulo 5), a qual foi guiada pela seguinte questão: “Qual agrupamento deve-se considerar para melhorar o desempenho de consultas restritas por classes?”.

6.2 Trabalhos Correlatos

Reduzir o número de cálculos de distância e acesso a disco durante o processamento de consultas por similaridade é um processo crucial, logo, busca-se por mecanismos de poda no espaço de busca durante o processamento desse tipo de consulta. O estudo apresentado em (FERHATOSMANOGLU *et al.*, 2001) aborda que uma busca com restrições deveria minimizar o número de páginas acessadas e evitar cálculos de distância e/ou acessos a disco desnecessários. Diante do contexto, novas técnicas são exploradas pelos pesquisadores da área, para gerenciar eficientemente esses tipos de consultas. O estudo apresentado em (SOUZA; TRAINA; MICHEL, 2018) abordou um método baseado em múltiplos índices para processar consultas por similaridade restritas por classes, com relação as combinações distintas de classes. Este estudo foi guiado pela suposição, que é possível aplicar restrições de classes a um algoritmo **k**-NN tradicional e ainda obter um processamento de tempo apropriado. Entretanto, esta abordagem requer mais espaço em memória para manter todos os índices construídos de acordo com as combinações distintas de classes. De modo geral, o estudo explorou a execução de consultas restritas por múltiplas classes. Outro estudo relacionado com o trabalho anterior foi apresentando em (MILCHEVSKI; NEFFGEN; MICHEL, 2018), o qual analisou o impacto no processamento de consultas restritas por classes através da junção de duas estruturas de indexação, ao invés de aplicar indexação múltipla de um mesmo tipo de estrutura, os autores exploraram a construção de um índice invertido com um método de acesso multidimensional. Diante das questões abordadas anteriormente, na próxima Seção é apresentada uma nova abordagem com as seguintes contribuições:

1. *Redução do número de cálculos de distância*: a abordagem emprega estratégia de agrupamento dos dados sobre uma estrutura de indexação métrica, a qual executa consultas por

similaridade dentro do agrupamento resultante. Evidências experimentais mostram que essa abordagem apresenta um bom desempenho;

2. *Seleção de grupo(s) adequado(s)*: para o processamento de consultas sobre o agrupamento resultante;
3. *Seleção de grupos automaticamente*: para essa fase do processamento a abordagem emprega o algoritmo apresentado no Capítulo 5;
4. *Integração de detecção de agrupamento com consultas restritas por classes*: a abordagem utiliza indexação múltipla sobre o agrupamento, ao invés de aplicar sobre cada combinação de classes.

6.3 Abordagem Proposta

A abordagem CfQ executa consultas por similaridade a partir de um agrupamento. Dessa maneira, o estudo foi guiado pela suposição que é possível reduzir o número de computações de distância com a integração das estratégias apresentadas nos Capítulos 4 e 5. Essa integração permite explorar os pontos positivos das estratégias apresentadas anteriormente e analisar as lacunas abordadas para melhorar a eficiência e eficácia no processo de recuperação dos dados. A Figura 25 ilustra as principais fases da abordagem apresentada nesse capítulo. Para isso, inicia-se pela extração de características de cada elemento. Nessa fase, o extrator aplicado pode impactar positivamente ou negativamente, de acordo com o tipo de domínio de dados. Na segunda fase dessa abordagem seleciona-se todos os parâmetros para construção da estrutura de indexação (e.g. políticas de escolha de nós, detalhes apresentados em (SOUZA; RAZENTE; BARIONI, 2013)). A próxima fase seleciona o nível $l - 1$ de uma estrutura de indexação (nível com todos os elementos representativos). Mais adiante, uma lista de elementos representativos é construída, a qual é ordenada pela distância do elemento mais próximo com relação ao anterior sucessivamente. O objetivo é aplicar essa lista ordenada para dividir os elementos em grupos, de acordo com um limiar adquirido após a ordenação da lista. O valor do limiar é obtido pela soma da média com o desvio padrão da lista contruída. Após esse processo, o elemento mais central é selecionado em cada grupo para ser o medóide, por fim, cada elemento do conjunto de dados é atribuído ao elemento medóide mais próximo.

No geral, o agrupamento resultante apresenta maior grau de compatibilidade com a distribuição dos dados. Portanto, um conjunto de dados pode conter $|C|$ dimensões de classes, mas nem sempre o mesmo número de grupos n . Esse processo demanda uma análise exploratória dos conjuntos de dados. O algoritmo clusMAM automaticamente sugere um número de grupos (n). Na sexta fase são aplicadas medidas de validação para avaliar o agrupamento resultante, com relação a homogeneidade e separação dos elementos (veja Seção 3.3). Essa fase é importante, porque após a validação, os grupos serão ordenados para a execução de consultas. A ordenação é

obtida pela distância do elemento de consulta com relação ao medóide de cada grupo. Dessa maneira, destaca-se que serão construídas estruturas de indexação para cada grupo resultante, no entanto, as consultas serão executadas sobre os grupos selecionados com uma ordem de prioridade de distância entre o elemento de consulta (s_q) e o medóide de cada grupo.

O algoritmo 3 apresenta os passos principais da abordagem. O primeiro *loop* apresentado nas linhas 9 a 11 será executado apenas se o agrupamento resultante apresentar valores apropriados com relação aos índices de validade *Silhueta Simplificada* (medida com um intervalo de 0 a 1, quanto mais próximo de 1, significa que o agrupamento resultante é mais apropriado) e *Davies Bouldin* (medida que apresenta valores a partir de 0, quanto menor o valor ou mais próximo de zero, significa que o número de grupo (n) é o mais apropriado para o agrupamento resultante). Após a construção da lista ordenada de grupos (gerada a partir da seleção do medóide mais próximo do elemento de consulta s_q (linha 10)), o segundo *loop* (apresentado nas linhas 13 a 16) é responsável pela seleção da estrutura de indexação representante do grupo do medóide. Destaca-se que na linha 15 as consultas podem ser finalizadas antecipadamente, baseada na hipótese mencionada anteriormente, a qual supõe que elementos mais similares e/ou das mesmas classes estão no mesmo grupo ou em um grupo próximo. Entretanto, é importante notar que essa abordagem pode beneficiar tanto as consultas clássicas quanto as consultas restritas por classes (conforme apresentado na Seção 6.4).

Algoritmo 3 – CfQ

```

1: procedimento CfQ(dados dos elementos  $\in S$ )                                ▷ agrupar para consultar
2:   escolha os parâmetros para construção da estrutura
3:    $R \leftarrow \{\}$                                                          ▷ conjunto resposta
4:    $G \leftarrow \text{clusMAM}(\text{dados dos elementos para serem indexados})$       ▷ agrupamento
5:   calcule o índice de validade Silhueta Medóide para o agrupamento resultante
6:   calcule o índice de validade Davies Bouldin para o agrupamento resultante
7:   se Silhueta Medóide  $> 0,50$  and Davies Bouldin  $< 1,0$  então              ▷ limiares de qualidade
8:     construir múltiplas estruturas de indexação                            ▷ um índice para cada grupo
9:     para todo  $m \in M$  faça
10:       encontre o elemento medóide mais próximo de  $s_q$                     ▷ verificar conjunto de medóides
11:       escolha um índice  $t_i \in T$  para compor a lista ordenada de grupos
12:     fim para
13:     para todo  $t_i \in$  lista ordenada de grupos faça
14:       remova  $t_i \in$  lista ordenada de grupos
15:       execute consultas por similaridade sobre a estrutura selecionada    ▷ antecipar finalização
de consulta quando não houver elementos do conjunto resposta atual com menor distância, quando
comparados com as distâncias do conjunto resposta anterior
16:     fim para
17:   fim se
18:   se  $R \neq \emptyset$  então
19:     return  $R$ 
20:   fim se
21: fim procedimento

```

Figura 25 – Ilustração do processamento da abordagem CfQ. (A) Dado um conjunto de dados complexos (e.g. um conjunto de imagens), onde suas características são extraídas por um algoritmo de extração de características. (B) Todos os elementos são indexados sobre um método de acesso métrico. É importante destacar que o parâmetro tamanho da página (nó) influencia os resultados do método (ver Seção 4.4). (C) O processamento da amostragem do conjunto é executado sobre o nível $l - 1$, nessa fase todos os elementos representativos são selecionados a partir da estrutura construída. (D) A detecção de agrupamento é executada sobre a amostra selecionada do conjunto. Após essa fase, cada elemento do conjunto será atribuído ao elemento medóide mais próximo. (E) Índices de validade são aplicados sobre o agrupamento resultante para analisar a homogeneidade e separação dos elementos. (F) Consultas por similaridade serão executadas sobre uma lista ordenada de grupos, no qual o elemento s_q é o mais próximo dos elementos medóides de cada grupo.

6.4 Avaliações Experimentais

Para análise de eficiência a abordagem apresentada nesse capítulo foi avaliada com relação ao número de cálculos de distância necessários para executar cada algoritmo. A fase experimental foi dividida em dois conjuntos principais, com o objetivo de auxiliar na análise. Na primeira fase experimental, considerou-se a execução de consultas por similaridade clássicas, isto é, sem considerar as restrições de classes. Essas execuções foram realizadas com e sem filtro de agrupamento. Na segunda fase experimental foi computado o número de cálculos de distância com relação às consultas restritas por classes, nessa fase também analisou-se o desempenho com e sem filtro de agrupamento. Em todos os conjuntos de experimentos desse capítulo, considerou-se a média do tempo total de execução de 50 centros de consultas (selecionados aleatoriamente), para cada valor de k . O tamanho de página utilizado nos experimentos foi 4 *KBytes*.

Nove (09) conjuntos sintéticos foram empregados nos experimentos. Esses conjuntos foram construídos para permitir uma avaliação atenciosa dos algoritmos. O processo para geração desses conjuntos de dados é descrito em (CIACCIA; PATELLA; ZEZULA, 1997) e, a variância aplicada foi $\sigma^2 = 0.01$. Além desses dados sintéticos, dois (02) conjuntos de dados reais foram aplicados para analisar o comportamento dos algoritmos em situação real, a saber *Aloi*¹ e *Animals*². Os conjuntos reais são compostos de vetores de características com 24 dimensões adquiridas a partir do extrator de características Haralick. As descrições de cada conjunto de dados é apresentada na Tabela 3, cada conjunto é identificado com seu próprio nome, cada nome é seguido pela regra: o número de elementos do conjunto (#Elem), a dimensionalidade do conjunto (D), o número de dimensões de classes ($|C|$), o número de classes em cada dimensão ($|C_i|$), a função de distância utilizada ($d()$) e o número de grupos (n) usado para gerar os conjuntos de dados. Destaca-se que encontrar um número de grupos adequado em conjuntos reais não é uma tarefa trivial. Além disso, a aplicação de diferentes algoritmos com diferentes parâmetros, resultarão diferentes agrupamentos. Portanto, a coluna representada por “#grupos (n)” para os conjuntos reais encontra-se sem valor (como discutido no Capítulo 3 o objetivo de um algoritmo de agrupamento é encontrar grupos).

6.5 Resultados Obtidos

Nesta seção são apresentados os gráficos que mostram o desempenho das consultas por similaridades, quando executadas sobre um agrupamento. Destaca-se que os algoritmos de agrupamento selecionados para essa análise foram: *clusMAM* e *PAM-Slim*. Como apresentado no Capítulo 4, esses dois algoritmos apresentaram os melhores resultados com relação custo vs benefício no processo de detecção de agrupamento (Figura 26). A Figura 26 apresenta os resultados com relação aos conjuntos sintéticos compostos de 100.000 vetores de características

¹ <<http://aloi.science.uva.nl/>>

² <<https://cvml.ist.ac.at/AwA2/>>

Tabela 3 – Descrição dos conjuntos sintéticos e reais utilizados nos experimentos

Nome	#Elem	D	$ \mathcal{C} $	$ \mathcal{C}_i $	$d()$	#grupos (n)
Rand100_10d_2dc_2c_5n	100.000	10	2	2	L_2	5
Rand100_10d_2dc_2c_10n	100.000	10	2	2	L_2	10
Rand100_10d_2dc_2c_20n	100.000	10	2	2	L_2	20
Rand100_10d_2dc_2c_30n	100.000	10	2	2	L_2	30
Rand200_10d_3dc_2c_30n	200.000	10	3	2	L_2	30
Rand200_10d_3dc_2c_40n	200.000	10	3	2	L_2	40
Rand100_10d_2dc_2c_15n	100.000	10	2	2	L_2	15
Rand200_10d_2dc_2c_15n	200.000	10	2	2	L_2	15
Rand300_10d_2dc_2c_15n	300.000	10	2	2	L_2	15
Aloi_72_24d_1dc	72.000	24	01	1000	L_2	-
Animals_37_24d_2dc	37.000	24	02	$ \mathcal{C}_1 =50$ $ \mathcal{C}_2 =03$	L_2	-

Fonte: Dados da pesquisa.

com 10 dimensões cada (Rand100_10d_2dc_2c_5n, Rand100_10d_2dc_2c_10n, Rand100_10d_2dc_2c_15n e Rand100_10d_2dc_2c_20n). Diante dos resultados apresentados, os próximos conjuntos experimentais aplicaram esses dois algoritmos de detecção de agrupamento, como filtros para a execução de consultas por similaridade sem e com restrições de classes.

Figura 26 – (a) Número de cálculos de distância executados pelos algoritmos clusMAM e PAM-Slim (os resultados do eixo-y estão em escala log). clusMAM apresentou o mesmo comportamento para os 04 conjuntos sintéticos aplicados (Rand100_10d_2dc_2c_5n, Rand100_10d_2dc_2c_10n, Rand100_10d_2dc_2c_15n e Rand100_10d_2dc_2c_20n). Além disso, observando os resultados sumarizados em (b) e (c) o algoritmo apresentou agrupamento apropriado com relação aos índices de qualidade.

Fonte: Dados da pesquisa.

A Figura 27 apresenta os resultados referentes às computações de distância para o conjunto sintético Rand300_10d_2dc_2c_20n. Observando o gráfico é possível analisar que a aplicação de filtro, ou seja, o uso de técnicas de detecção de agrupamento podem corroborar na fase de consultas por similaridade. Logo, pode-se notar que a abordagem CfQ com o emprego do algoritmo clusMAM, apresentou a mesma ordem de grandeza de número de cálculos de distância, quando comparado com o PAM-Slim. Além disso, CfQ-clusMAM executou 58,98% menor número de cálculos de distância quando comparado com a execução sem filtro de agrupamento, para o

conjunto Rand200_10d_2dc_2c_15n e $k=10$. Esse comportamento, é pelo fato que a abordagem CfQ reduz o espaço de busca para execução dentro de um grupo ou mais grupo(s) específico(s), no qual pode-se garantir que os elementos mais próximos são os mais similares (até para elementos localizados na borda do grupo), pois a abordagem não aplica uma estratégia aleatória para seleção dos grupos. A partir das evidências experimentais, conclui-se que a abordagem CfQ pode auxiliar no problema CCkNN, com relação ao aumento no espaço de busca por similaridade restritas por classes. Assim, observando os gráficos apresentados na Figura 28 CfQ-clusMAM apresentou até 44,94% menor número de computações de distância com relação ao k -NN sem filtro com restrições de classes. Além disso, pode-se notar na Figura 28 (b) quando aumenta a cardinalidade do conjunto de dados, o número de cálculos de distância computados pelo algoritmo k -NN sem filtro com restrições de classes, aumenta uma ordem de grandeza comparado com os cálculos de distância computados pela abordagem CfQ-clusMAM. No entanto, esse evento não acontece quando se mantém a mesma cardinalidade de elementos nos conjuntos, mas aumenta o número de dimensões de classes do conjunto ($|C|$) (veja Figura 29). Esse comportamento é devido ao fato que o aumento de cálculos de distância, não está diretamente relacionado ao aumento da dimensionalidade de classes, mas sim a quantidade de elementos do conjunto e classes ($\mathcal{C}_i$) por dimensionalidade de classe. Isto é, a quantidade de classes por dimensionalidade de classes não apresenta balanceamento, logo, quanto maior o número de elementos no conjunto, maior o número de distâncias computadas para alcançar a restrição de classes desejada.

Observando a Figura 30 CfQ-clusMAM e CfQ-PAM-Slim apresentam o mesmo comportamento, mesmo quando se altera o número de grupos nos conjuntos de dados. No entanto, destaca-se que PAM-Slim requer maior computação de distâncias para realizar a tarefa de agrupamento (veja Figura 26). De modo geral, a abordagem CfQ apresentou menor número de cálculos de distância com relação aos parâmetros experimentais considerados nos conjuntos de dados sintéticos. Por exemplo, executou 34% menos computações de distâncias para o conjunto Rand200_10d_3dc_2c_30n quando comparada com k -NN sem filtro com restrições de classes (Figura 30 (a)). Já, para o conjunto Rand100_10d_2dc_2c_30n a abordagem executou 66,17% do número de cálculos de distância apresentado pelo k -NN sem filtro com restrições de classes, para $k=10$ (Figure 29 (a)).

A Tabela 4 apresenta uma comparação entre as abordagens *clusMAM* e PAM-Slim com relação aos conjuntos reais considerados nesse capítulo. Pode-se notar que no geral os algoritmos apresentaram comportamentos semelhantes para todos os conjuntos de dados apresentados na Tabela 3. Portanto, dado um elemento de consulta s_q , os k elementos mais próximos a s_q pertencem ao grupo, no qual s_q pertence e/ou aos grupos vizinhos mais próximos de s_q . Observa-se que a abordagem *clusMAM* possui valores menores de Silhueta Simplificada, mas realiza um número expressivamente menor de cálculos de distância do que a abordagem competitiva (como discutido no Capítulo 4).

Figura 27 – Número de cálculos de distância computados para retornar os k elementos mais próximos a partir de um elemento de consulta s_q . Resultados referentes aos conjuntos de dados sintéticos, variando o número de elementos entre 100.000, 200.000 e 300.000, com relação à execução de consultas k -NN clássicas, isto é, sem empregar restrições de classes.

Fonte: Dados da pesquisa.

Figura 28 – Número de cálculos de distância computados para retornar os k elementos mais próximos a partir de um elemento de consulta s_q . Resultados referentes aos conjuntos de dados sintéticos, variando o número de elementos entre 100.000, 200.000 e 300.000, com relação à execução de consultas k -NN clássicas, isto é, sem empregar restrições de classes.

Fonte: Dados da pesquisa.

Figura 29 – Número de cálculos de distância executados pelos conjuntos sintéticos Rand100_10d_2dc_2c_30n e Rand200_10d_3dc_2c_30n variando o número de dimensões de classes ($|C|$). A abordagem CfQ-clusMAM apresentou comportamento similar aos apresentados anteriormente, ou seja, essa abordagem é capaz de ainda apresentar menor número de cálculos de distância, mesmo aumentando os valores de dimensões de classes.

Fonte: Dados da pesquisa.

Figura 30 – Número de cálculos de distância executados pelos conjuntos sintéticos Rand200_10d_3dc_2c_30n e Rand100_10d_3dc_2c_40n variando o número de grupos (n) de cada conjunto.

Fonte: Dados da pesquisa.

Tabela 4 – Comparação entre os algoritmos *clusMAM* e PAM-Slim com relação aos conjuntos de dados reais considerados nesse capítulo. É importante notar que o número de grupos (n) apresentado nessa tabela para cada conjunto foi sugerido pelo *clusMAM*.

Algoritmo	n	Silhueta Simplificada	Conjunto	# Cálculos Distância
<i>clusMAM</i>	02	0.870	Animals_37_24d_2dc	1592795
PAM-Slim	02	0.891	Animals_37_24d_2dc	63800469
<i>clusMAM</i>	30	0.587	Aloi_72_24d_1dc	2504865
PAM-Slim	30	0.641	Aloi_72_24d_1dc	26926734361

Fonte: Dados da pesquisa.

6.5.1 Considerações Finais

Neste capítulo apresentamos o método CfQ, o qual combina as vantagens apresentadas pelos algoritmos *clusMAM* (Capítulo 4) e CCKNN (Capítulo 5). Por outro lado, CfQ particiona o conjunto de dados em grupos disjuntos selecionados a partir de uma amostra do conjunto de dados em questão. Além disso, o método constrói estruturas de indexação para cada grupo resultante do agrupamento realizado sobre a amostra do conjunto. O conjunto resposta deve ser o resultado da concatenação da lista resultante de cada grupo para garantir que realmente o conjunto resposta contenha os k elementos mais próximos. No entanto, a estratégia adotada pode finalizar antes de verificar todas as árvores representantes de cada grupo. Isso acontece pelo fato que as árvores são ordenadas, de acordo com o medóide do grupo, sobre uma lista de prioridades para execuções das consultas, com o objetivo de minimizar o número de computações de distância durante o processamento. Observe a Figura 31 uma amostra do conjunto de dados reais *Iris* (ilustrado na Figura 17 do Capítulo 5 e apresentado na Tabela 2 do Capítulo 4). Esse conjunto contém 03 classes, mas dois grupos linearmente separados. Imagine que o elemento de consulta (s_q) apresente $\mathbf{Q}_c = \{\mathcal{C}_1 = \{\text{espécie:iris-virginica}\}\}$, então, dado que o número de grupos $n=2$ e, o elemento de consulta encontra-se no grupo₂ (representado pela estrutura de índice $t_2 \in \mathbf{T}$). Pode-se intuitivamente notar que os k mais próximos que são do mesmo conjunto de classes

de s_q , pertencem ao grupo₂. Portanto, apenas os elementos do grupo₂ serão analisados, pois acontecerá uma poda no espaço de busca; quando o primeiro elemento mais próximo pertencente ao grupo₁ for retornado, ou seja, a lista resultante dos k elementos mais próximos restritos por classes do grupo₁, não será verificada/retornada, assim reduzindo o número de cálculos de distância.

Diante do contexto apresentado, pode-se notar que CfQ é uma ferramenta relevante para apoiar consultas por similaridade restritas por classes. Logo, dado um grupo selecionado a partir da distância do medóide ao elemento s_q (pela utilização de apenas medida de similaridade); pode-se apresentar um conjunto resposta com todos os elementos mais similares, mesmo se a abordagem empregar restrições de classes apenas dentro de um grupo.

Figura 31 – Ilustração do impacto de agrupar para consultar sobre o espaço de busca, com a aplicação das restrições de classes sobre uma consulta k -NN tradicional, considerando o valor de $k=3$ com relação ao elemento s_q de $Q_c=\{\mathcal{C}_1=\{\text{espécie:iris-virginica}\}\}$.

Fonte: Elaborada pela autora.

CONCLUSÃO

7.1 Considerações Finais

Para representar um conjunto de dados de maneira sumarizada, é necessário uma técnica robusta, que seja capaz de:

- Extrair informações representativas e relevantes para mantê-las a partir de um número adequado e, discriminatório de elementos;
- Uma estratégia eficiente e eficaz que compute esses dados de forma adequada para não perder o conteúdo semântico que existe dentro dos relacionamentos entre os elementos do conjunto.

No entanto, para o processamento de tarefas que exijam milhares de computações de medidas de similaridade, processar conjuntos de dados volumosos e representá-los através de uma porção resumida, para serem interpretados qualitativamente, não é uma tarefa trivial. Esse fato, vem sendo pesquisado sob diferentes aspectos, os quais exploram tratar o conjunto resposta de uma dada técnica às expectativas do usuário. Entretanto, atender à semântica esperada pelo usuário é uma tarefa desafiadora.

O objetivo final desse trabalho foi desenvolver técnicas para representar um conjunto de dados, em uma forma resumida sem perder a rica informação existente entre os relacionamentos dos elementos do conjunto. Além disso, reduzir de maneira eficaz as computações de medidas de similaridade que ocorrem durante o processamento de técnicas que são baseadas em funções de distância, tais como, detecção de agrupamento dos dados e consultas por similaridade. Para esse feito, algoritmos de detecção de agrupamento aplicados às consultas por similaridades foram empregados como motivação. As diversas contribuições propostas nesta pesquisa de doutorado permitiram o apoio às pesquisas na área, representando conjunto de dados complexos (*e.g.* imagens) de maneira resumida reduzindo a perda de informações semânticas, com a aplicação de

detecção de agrupamento de dados. Além disso, apoiar no estudo e desenvolvimento de técnicas que aplicam informações extrínsecas sobre consultas por similaridade.

7.2 Contribuições

Neste documento foram propostas novas abordagens para processar e representar dados complexos, motivadas pela metodologia de representação resumida dos dados. De modo geral, as principais contribuições do presente estudo para a área de Mineração de dados e Sistemas de Gerenciamento de Dados são descritas a seguir:

- No Capítulo 4 foi apresentada uma investigação da influência de Métodos de Acesso Métrico sobre técnicas de Mineração de Dados, como a de agrupamento de dados. A averiguação mais importante nessa fase experimental foi a influência da distribuição e do relacionamento entre os elementos, sobre maior impacto na qualidade final no processo de detecção de agrupamento.
- O Capítulo 5 apresenta o impacto com relação ao tratamento da questão de adicionar informações extrínsecas sobre consultas por similaridade. Foi apresentada uma nova abordagem para lidar com o problema de restringir por múltiplas classes, uma consulta aos k vizinhos mais próximos. Foi apresentada uma estratégia para aplicações de indexação múltiplas sobre combinações distintas de classe, a qual permite representar o conjunto de dados em pequenos subconjuntos, com o objetivo de reduzir o espaço de busca. Essa estratégia apresenta não só ganho semântico no conjunto resposta, mas também ganho de eficiência no processamento desse tipo de consulta.
- No Capítulo 6 uma nova abordagem de indexação múltipla é apresentada, visando melhorar a eficiência e eficácia no processamento de consultas por similaridade. A abordagem mostra a importância da integração de técnicas de agrupamento para apoiar o processamento de consultas restritas por classes. A ideia geral, é obter automaticamente um possível “agrupamento” dos dados, mas adquiridos a partir de uma amostra qualitativa do conjunto de dados. A amostra é obtida a partir de uma estrutura de indexação métrica, onde os elementos são distribuídos em nós (páginas de tamanho fixo) sobre uma estrutura hierárquica de árvore. Assim, considera-se que elementos pertencentes ao mesmo grupo compartilham o mesmo nó ou estão indexados em nós vizinhos. Com isso, aplicar o agrupamento resultante para otimizar consultas por similaridade que podem ser restritas por múltiplas classes, com o objetivo de podar um número maior de subárvores que serão visitadas, a qual também pode se beneficiar com uma finalização antecipada do processamento de consulta. Dado que uma consulta iniciará primeiramente no grupo (representado por uma estrutura de indexação), no qual o elemento de consulta pertence.

7.3 Trabalhos Futuros

Diante das contribuições mencionadas anteriormente nesta tese, pode-se apresentar algumas sugestões de tópicos de pesquisa para estender as técnicas apresentadas neste documento, visando abordar outras classes de problemas:

- Avaliar outras funções/heurísticas para cálculo do limiar de grupos, considerado na abordagem clusMAM e, o impacto dessas funções em outros domínios de dados;
- Avaliar o impacto na aplicação de estratégias para selecionar combinações de classes relevantes para a fase de construção de índices, bem como empregar um método para lidar com a redução do número de combinações de classes e, o impacto nas consultas por similaridade restritas;
- Investigar a aplicação e o impacto de múltiplas classes nas políticas de construção das estruturas;
- Avaliar a abordagem CfQ em outros domínios de dados;
- Avaliar o problema abordado pelo CCkNN, bem como estender essa abordagem para indexação híbrida com métodos de acesso métrico;
- Avaliar o impacto da integração sobre diferentes descritores de características e funções de distância.

7.4 Publicações no Período

As publicações de artigos científicos que refletem o desenvolvimento do projeto e as contribuições para o estado da arte foram realizadas durante todo o período de doutoramento. A seguir são apresentadas as publicações geradas até o presente momento:

Em andamento:

(2018) **DE SOUZA, J.A.**; Traina, A.J.M. *Clustering for processing similarity queries*. Submetido para : *Journal Data & Knowledge Engineering*¹

Publicados no período:

(2018) **DE SOUZA, J.A.**; Traina, A.J.M.; Michel, S. *Class-Constraint Similarity Queries*. In: ACM/SIGAPP Symposium on Applied Computing (SAC), France, p. 549—556, DOI: <<https://doi.org/10.1145/3167132.3167192>>

¹ <<https://www.journals.elsevier.com/data-and-knowledge-engineering>>

(2016) **DE SOUZA, J.A.**; Cazzolato, M.T; Traina, A.J.M. *ClusMAM: Fast and Effective Unsupervised Clustering of Large Complex Datasets using Metric Access Methods*. In: ACM/SIGAPP Symposium on Applied Computing (SAC), Italy, p.986—991, DOI:<<http://dx.doi.org/10.1145/2851613.2851661>>

Colaborações no período:

(2018) Nesso-Jr, M.R.; Cazzolato, M.T., Scabora, L.C.; Oliveira, P.H.; Spadon, G.; **DE SOUZA, J.A.**; Oliveira, W.D.; Chino, D.Y.T.; Rodrigues-Jr, J.F.; Traina, A.J.M.; Traina-Jr, C. *RAFIKI: Retrieval-Based Application for Imaging and Knowledge Investigation*. In: IEEE 31st International Symposium on Computer-Based Medical Systems, DOI: <[10.1109/CBMS.2018.00020](http://dx.doi.org/10.1109/CBMS.2018.00020)>

(2017) Cazzolato, M.T.; Avalhais, L.P.S.; Chino, D. Y. T.; Ramos, J.S.; **DE SOUZA, J.A.**; Rodrigues-Jr, J.F.; Traina, A.J.M. *FiSmo: A Compilation of Datasets from Emergency Situations for Fire and Smoke Analysis*. Simpósio Brasileiro de Banco de Dados. URL: <http://conteudo.icmc.usp.br/pessoas/junio/PublishedPapers/Cazzolato_et_al_SBBD2017.pdf>

(2016) Cazzolato, M.T.; Bedo, M.V.N.; Costa, A.F.; **DE SOUZA, J.A.**; Traina-Jr, C.; Rodrigues-Jr, J.F.; Traina, A.J.M. *Unveling Smoke in Social Images with the SmokeBlock Approach*. In Proceedings of the 31st Annual ACM Symposium on Applied Computing, DOI: <[10.1145/2851613.2851634](http://dx.doi.org/10.1145/2851613.2851634)>

(2014) **DE SOUZA, J.A.**; Razente, H.L.; Barioni, M.C.N. *Optimizing metric access methods for querying and mining complex data types*. Journal of the Brazilian Computer Society, pages: 17, DOI: <<https://doi.org/10.1186/s13173-014-0017-5>>

REFERÊNCIAS

AGGARWAL, C. C.; REDDY, C. K. **Data clustering: algorithms and applications**. [S.l.]: CRC press, 2013. Citado na página 47.

AHN, H.-K.; MAMOULIS, N.; WONG, H. M. A survey on multidimensional access methods. 2001. Citado na página 37.

ALMEIDA, J.; VALLE, E.; TORRES, R. d. S.; LEITE, N. J. Dabc-tree: An effective index for approximate search in high-dimensional metric spaces. **Journal of Information and Data Management**, v. 1, n. 3, p. 375, 2010. Citado na página 39.

ALOISE, D.; CONTARDO, C. A sampling-based exact algorithm for the solution of the minimax diameter clustering problem. **Journal of Global Optimization**, v. 71, n. 3, p. 613–630, Jul 2018. ISSN 1573-2916. Disponível em: <<https://doi.org/10.1007/s10898-018-0634-1>>. Citado nas páginas 27 e 49.

ARONOVICH, L.; SPIEGLER, I. Cm-tree: A dynamic clustered index for similarity search in metric databases. **Data Knowl. Eng.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 63, n. 3, p. 919–946, dez. 2007. ISSN 0169-023X. Disponível em: <<http://dx.doi.org/10.1016/j.datak.2007.06.001>>. Citado na página 40.

ARORA, A.; SINHA, S.; KUMAR, P.; BHATTACHARYA, A. Hd-index: Pushing the scalability-accuracy boundary for approximate knn search in high-dimensional spaces. **Proc. VLDB Endow.**, VLDB Endowment, v. 11, n. 8, p. 906–919, abr. 2018. ISSN 2150-8097. Disponível em: <<https://doi.org/10.14778/3204028.3204034>>. Citado na página 39.

ASERY, R.; MARWAHA, P.; SUNKARIA, R.; SHARMA, L. Image retrieval techniques using content-based local binary descriptors: A survey. p. 173–195, 07 2017. Citado na página 33.

BAEZA-YATES, R. Searching: an algorithmic tour. **Encyclopedia of Computer Science and Technology**, v. 37, p. 331–359, 1997. Citado na página 39.

BAEZA-YATES, R.; CUNTO, W.; MANBER, U.; WU, S. Proximity matching using fixed-queries trees. In: CROCHEMORE, M.; GUSFIELD, D. (Ed.). **Combinatorial Pattern Matching**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994. p. 198–212. ISBN 978-3-540-48450-9. Citado na página 39.

BARIONI, M. C. N.; RAZENTE, H. L.; TRAINA, A. J. M.; TRAINA JR., C. An efficient approach to scale up k -medoid based algorithms in large databases. Simpósio Brasileiro de Banco de Dados (SBBD), p. 265–279, 2006. Disponível em: <<http://webdocs.cs.ualberta.ca/~mn/SBBD2006/19-barioni-efficient-approach-sbbd.pdf>>. Citado na página 55.

_____. Accelerating k -medoid-based algorithms through metric access methods. **J. Syst. Softw.**, Elsevier Science Inc., New York, NY, USA, v. 81, n. 3, p. 343–355, mar. 2008. ISSN 0164-1212. Disponível em: <<http://dx.doi.org/10.1016/j.jss.2007.06.019>>. Citado nas páginas 27, 44, 50, 55 e 56.

- BARTOLINI, I.; CIACCIA, P.; PATELLA, M. Multimedia, similarity, and preferences: Adding flexibility to your information needs. In: _____. **A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years**. Cham: Springer International Publishing, 2018. p. 127–141. ISBN 978-3-319-61893-7. Disponível em: <https://doi.org/10.1007/978-3-319-61893-7_8>. Citado na página 37.
- BECKMANN, N.; KRIEGEL, H.-P.; SCHNEIDER, R.; SEEGER, B. The r*-tree: An efficient and robust access method for points and rectangles. In: **Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1990. (SIGMOD '90), p. 322–331. ISBN 0-89791-365-5. Disponível em: <<http://doi.acm.org/10.1145/93597.98741>>. Citado na página 55.
- BEGUM, S. A. N.; SUPREETHI, K. A survey on spatial indexing. **Journal of Web Development and Web Designing**, v. 3, n. 1, 2018. Citado nas páginas 37 e 40.
- BERKHIN, P. A survey of clustering data mining techniques. In: _____. **Grouping Multidimensional Data: Recent Advances in Clustering**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 25–71. ISBN 978-3-540-28349-2. Disponível em: <https://doi.org/10.1007/3-540-28349-8_2>. Citado nas páginas 11, 47 e 48.
- BHAGYALAKSHMI, A.; VIJAYACHAMUNDEESWAN, V. A survey on content based image retrieval using various operators. In: **Proceedings of IEEE International Conference on Computer Communication and Systems ICCCS14**. [S.l.: s.n.], 2014. p. 018–023. Citado na página 33.
- BöHM, C.; BERCHTOLD, S.; KEIM, D. A. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 33, n. 3, p. 322–373, set. 2001. ISSN 0360-0300. Disponível em: <<http://doi.acm.org/10.1145/502807.502809>>. Citado na página 37.
- BOZKAYA, T.; OZSOYOGLU, M. Distance-based indexing for high-dimensional metric spaces. In: ACM. **ACM SIGMOD Record**. [S.l.], 1997. v. 26, n. 2, p. 357–368. Citado na página 39.
- BUGATTI, P. H.; TRAINA, A. J. M.; TRAINA JR., C. Assessing the best integration between distance-function and image-feature to answer similarity queries. In: **Proceedings of the 2008 ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2008. (SAC '08), p. 1225–1230. ISBN 978-1-59593-753-7. Disponível em: <<http://doi.acm.org/10.1145/1363686.1363969>>. Citado na página 36.
- BURKHARD, W. A.; KELLER, R. M. Some approaches to best-match file searching. **Commun. ACM**, ACM, New York, NY, USA, v. 16, n. 4, p. 230–236, abr. 1973. ISSN 0001-0782. Disponível em: <<http://doi.acm.org/10.1145/362003.362025>>. Citado na página 38.
- CARÉLO, C. C. M.; POLA, I. R. V.; CIFERRI, R. R.; TRAINA, A. J. M.; TRAINA-JR., C.; CIFERRI, C. D. de A. The onion-tree: Quick indexing of complex data in the main memory. In: GRUNDSPENKIS, J.; MORZY, T.; VOSEN, G. (Ed.). **Advances in Databases and Information Systems**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 235–252. ISBN 978-3-642-03973-7. Citado na página 40.
- CHÁVEZ, E.; NAVARRO, G.; BAEZA-YATES, R.; MARROQUÍN, J. L. Searching in metric spaces. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 33, n. 3, p. 273–321, set. 2001. ISSN 0360-0300. Disponível em: <<http://doi.acm.org/10.1145/502807.502808>>. Citado na página 37.

- CHEN, L.; GAO, Y.; ZHENG, B.; JENSEN, C. S.; YANG, H.; YANG, K. Pivot-based metric indexing. **Proceedings of the VLDB Endowment**, VLDB Endowment, v. 10, n. 10, p. 1058–1069, 2017. Citado na página 40.
- CIACCIA, P.; PATELLA, M. **Approximate similarity queries: a survey**. [S.l.], 2001. Citado na página 27.
- CIACCIA, P.; PATELLA, M.; ZEZULA, P. M-tree: An efficient access method for similarity search in metric spaces. In: **Proceedings of the 23rd International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. (VLDB '97), p. 426–435. ISBN 1-55860-470-7. Disponível em: <<http://dl.acm.org/citation.cfm?id=645923.671005>>. Citado nas páginas 40, 60, 73 e 84.
- DAVIES, D. L.; BOULDIN, D. W. A cluster separation measure. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, n. 2, p. 224–227, 1979. Citado na página 51.
- DUBEY, A. K.; GUPTA, U.; JAIN, S. Comparative study of k-means and fuzzy c-means algorithms on the breast cancer data. **International Journal on Advanced Science, Engineering and Information Technology**, v. 8, n. 1, p. 18–29, 2018. Citado na página 49.
- ESTER, M.; KRIEGEL, H.-P.; XU, X. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. In: EGENHOFER, M. J.; HERRING, J. R. (Ed.). **Advances in Spatial Databases**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995. p. 67–82. ISBN 978-3-540-49536-9. Citado nas páginas 27, 50, 55 e 57.
- EVERITT, B. S.; LANDAU, S.; LEESE, M.; STAHL, D. **Cluster Analysis**. 5. ed. [S.l.]: John Wiley and Sons, 2011. Citado na página 36.
- FAHAD, A.; ALSHATRI, N.; TARI, Z.; ALAMRI, A.; KHALIL, I.; ZOMAYA, A. Y.; FOUFOU, S.; BOURAS, A. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. **IEEE Transactions on Emerging Topics in Computing**, v. 2, n. 3, p. 267–279, Sept 2014. ISSN 2168-6750. Citado nas páginas 47 e 49.
- FERHATOSMANOGLU, H.; STANOI, I.; AGRAWAL, D.; ABBADI, A. E. Constrained nearest neighbor queries. In: **Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases**. London, UK, UK: Springer-Verlag, 2001. (SSTD '01), p. 257–278. ISBN 3-540-42301-X. Disponível em: <<http://dl.acm.org/citation.cfm?id=647227.719234>>. Citado nas páginas 67 e 80.
- FISHER, R. **Machine Learning Repository**. 2018. Disponível em: <<https://archive.ics.uci.edu/ml/datasets/iris>>. Citado nas páginas 60 e 70.
- FRÄNTI, P.; VIRMAJOKI, O.; HAUTAMÄKI, V. Fast agglomerative clustering using a k-nearest neighbor graph. **IEEE Trans. on Pattern Analysis and Machine Intelligence**, v. 28, n. 11, p. 1875–1881, 2006. Disponível em: <<http://cs.uef.fi/sipu/datasets/>>. Citado na página 60.
- FU, Z.; XIANG, T.; KODIROV, E.; GONG, S. Zero-shot learning on semantic class prototype graph. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 40, n. 8, p. 2009–2022, Aug 2018. ISSN 0162-8828. Citado nas páginas 11 e 28.

- FÄRBER, I.; GÜNNEMANN, S.; KRIEGEL, H. peter; KRÖGER, P.; MÜLLER, E.; SCHUBERT, E.; SEIDL, T.; ZIMEK, A. On using class-labels in evaluation of clusterings. 2010. Disponível em: <<http://www.dbs.ifi.lmu.de/%7Ezimek/publications/MultiClustAtKDD2010/Faerberetal.pdf>>. Citado na página 58.
- GAEDE, V.; GÜNTHER, O. Multidimensional access methods. **ACM Computing Surveys (CSUR)**, ACM, v. 30, n. 2, p. 170–231, 1998. Citado na página 50.
- GAN, G.; MA, C.; WU, J. **Data clustering: theory, algorithms, and applications**. [S.l.]: Siam, 2007. v. 20. Citado nas páginas 34 e 50.
- GHOSH, N.; AGRAWAL, S.; MOTWANI, M. A survey of feature extraction for content-based image retrieval system. In: TIWARI, B.; TIWARI, V.; DAS, K. C.; MISHRA, D. K.; BANSAL, J. C. (Ed.). **Proceedings of International Conference on Recent Advancement on Computer and Communication**. Singapore: Springer Singapore, 2018. p. 305–313. ISBN 978-981-10-8198-9. Citado na página 33.
- HALKIDI, M.; BATISTAKIS, Y.; VAZIRGIANNIS, M. On clustering validation techniques. **Journal of Intelligent Information Systems**, v. 17, n. 2, p. 107–145, Dec 2001. ISSN 1573-7675. Disponível em: <<https://doi.org/10.1023/A:1012801612483>>. Citado nas páginas 50 e 52.
- HAN, J.; KAMBER, M.; PEI, J. **Data mining: concepts and techniques**. [S.l.: s.n.], 2012. Citado nas páginas 26, 47 e 55.
- HANYF, Y.; SILKAN, H. A queries-based structure for similarity searching in static and dynamic metric spaces. **Journal of King Saud University - Computer and Information Sciences**, 2018. ISSN 1319-1578. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S131915781830137X>>. Citado nas páginas 40 e 57.
- HARTIGAN, J. A.; WONG, M. A. Algorithm as 136: A k-means clustering algorithm. **Journal of the Royal Statistical Society. Series C (Applied Statistics)**, JSTOR, v. 28, n. 1, p. 100–108, 1979. Citado na página 47.
- HJALTASON, G. R.; SAMET, H. Incremental similarity search in multimedia databases. Citeseer, 2000. Citado na página 72.
- HUBERT, L.; ARABIE, P. Comparing partitions. **Journal of classification**, Springer, v. 2, n. 1, p. 193–218, 1985. Citado na página 51.
- HWANG, J.-J.; WHANG, K.-Y.; MOON, Y.-S.; LEE, B.-S. A top-down approach for density-based clustering using multidimensional indexes. **Journal of Systems and Software**, v. 73, n. 1, p. 169 – 180, 2004. ISSN 0164-1212. Performance modeling and analysis of computer systems and networks. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0164121203002553>>. Citado na página 50.
- JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern Recognition Letters**, v. 31, n. 8, p. 651 – 666, 2010. ISSN 0167-8655. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR). Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0167865509002323>>. Citado nas páginas 25 e 49.
- JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. **Algorithms for Clustering Data**. [S.l.: s.n.], 1988. Citado nas páginas 25, 49 e 50.

KALRA, M.; LAL, N.; QAMAR, S. K-mean clustering algorithm approach for data mining of heterogeneous data. In: MISHRA, D. K.; NAYAK, M. K.; JOSHI, A. (Ed.). **Information and Communication Technology for Sustainable Development**. Singapore: Springer Singapore, 2018. p. 61–70. ISBN 978-981-10-3920-1. Citado na página 25.

KAUFMAN, L.; ROUSSEEUW, P. J. **Finding groups in data: An introduction to cluster analysis**. [S.l.: s.n.], 1990. Citado nas páginas 27 e 48.

_____. **Finding groups in data: an introduction to cluster analysis**. [S.l.]: John Wiley & Sons, 2009. v. 344. Citado nas páginas 47 e 48.

KRASKA, T.; BEUTEL, A.; CHI, E. H.; DEAN, J.; POLYZOTIS, N. The case for learned index structures. In: **Proceedings of the 2018 International Conference on Management of Data**. New York, NY, USA: ACM, 2018. (SIGMOD '18), p. 489–504. ISBN 978-1-4503-4703-7. Disponível em: <<http://doi.acm.org/10.1145/3183713.3196909>>. Citado na página 57.

KRIEGL, H.-P.; KRÖGER, P.; ZIMEK, A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. **ACM Trans. Knowl. Discov. Data**, ACM, New York, NY, USA, v. 3, n. 1, p. 1:1–1:58, mar. 2009. ISSN 1556-4681. Disponível em: <<http://doi.acm.org/10.1145/1497577.1497578>>. Citado na página 47.

KRUSKAL, J. B. On the shortest spanning subtree of a graph and the traveling salesman problem. **Proceedings of the American Mathematical society**, JSTOR, v. 7, n. 1, p. 48–50, 1956. Citado na página 42.

LAI, H. P.; VISANI, M.; BOUCHER, A.; OGIER, J.-M. An experimental comparison of clustering methods for content-based indexing of large image databases. **Pattern Analysis and Applications**, Springer, v. 15, n. 4, p. 345–366, 2012. Citado na página 50.

MILCHEVSKI, E.; NEFFGEN, F.; MICHEL, S. Processing class-constraint k-nn queries with misp. In: **Proceedings of the 21st International Workshop on the Web and Databases**. New York, NY, USA: ACM, 2018. (WebDB'18), p. 2:1–2:6. ISBN 978-1-4503-5648-0. Disponível em: <<http://doi.acm.org/10.1145/3201463.3201466>>. Citado nas páginas 29, 67 e 80.

NAVARRO, G.; REYES, N. New dynamic metric indices for secondary memory. **Inf. Syst.**, Elsevier Science Ltd., Oxford, UK, UK, v. 59, n. C, p. 48–78, jul. 2016. ISSN 0306-4379. Disponível em: <<http://dx.doi.org/10.1016/j.is.2016.03.009>>. Citado na página 40.

NAVARRO, G.; URIBE-PAREDES, R. Fully dynamic metric access methods based on hyper-plane partitioning. **Information Systems**, v. 36, n. 4, p. 734 – 747, 2011. ISSN 0306-4379. Selected Papers from the 2nd International Workshop on Similarity Search and Applications SISAP 2009. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0306437911000032>>. Citado na página 40.

NG, R. T.; HAN, J. Efficient and effective clustering methods for spatial data mining. In: **Proceedings of the 20th International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994. (VLDB '94), p. 144–155. ISBN 1-55860-153-8. Disponível em: <<http://dl.acm.org/citation.cfm?id=645920.672827>>. Citado nas páginas 27 e 55.

NGUYEN, Q. H.; RAYWARD-SMITH, V. J. Internal quality measures for clustering in metric spaces. **International Journal of Business Intelligence and Data Mining**, Inderscience Publishers, v. 3, n. 1, p. 4–29, 2008. Citado nas páginas 51 e 52.

- NIXON, M.; AGUADO, A. S. **Feature Extraction & Image Processing for Computer Vision, Third Edition**. 3rd. ed. Orlando, FL, USA: Academic Press, Inc., 2012. ISBN 0123965497, 9780123965493. Citado na página 73.
- OLIVEIRA, P. H.; SCABORA, L. C.; CAZZOLATO, M. T.; BEDO, M. V. N.; TRAINA, A. J. M.; TRAINA-JR., C. MAMMOSET: An Enhanced Dataset of Mammograms. In: **Proceedings of the Satellite Events of the 32nd Brazilian Symposium on Databases**. [S.l.]: SBC, 2017. p. 256–266. Citado na página 73.
- OMRAN, M. G.; ENGELBRECHT, A. P.; SALMAN, A. An overview of clustering methods. **Intelligent Data Analysis**, IOS Press, v. 11, n. 6, p. 583–605, 2007. Citado na página 52.
- POLA, I. R. V.; TRAINA, C.; TRAINA, A. J. M. The nobh-tree: Improving in-memory metric access methods by using metric hyperplanes with non-overlapping nodes. **Data and Knowledge Engineering**, v. 94, p. 65 – 88, 2014. ISSN 0169-023X. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0169023X14000822>>. Citado na página 40.
- RAND, W. M. Objective criteria for the evaluation of clustering methods. **Journal of the American Statistical association**, Taylor & Francis Group, v. 66, n. 336, p. 846–850, 1971. Citado na página 51.
- ROBERTS, F.; TESMAN, B. **Applied combinatorics**. [S.l.]: Chapman and Hall/CRC, 2009. Citado na página 77.
- ROUSSEEUW, P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. **Journal of computational and applied mathematics**, Elsevier, v. 20, p. 53–65, 1987. Citado na página 51.
- RUIZ, E. V. An algorithm for finding nearest neighbours in (approximately) constant average time. **Pattern Recognition Letters**, Elsevier, v. 4, n. 3, p. 145–157, 1986. Citado na página 39.
- SAMET, H. **Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling)**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005. ISBN 0123694469. Citado nas páginas 37 e 39.
- SASSI, M.; GRISSA, A. Clustering large data sets based on data compression technique and weighted quality measures. In: **2009 IEEE International Conference on Fuzzy Systems**. [S.l.: s.n.], 2009. p. 396–402. ISSN 1098-7584. Nenhuma citação no texto.
- SELLIS, T.; ROUSSOPOULOS, N.; FALOUTSOS, C. **The R+-Tree: A Dynamic Index for Multi-Dimensional Objects**. [S.l.], 1987. Citado na página 37.
- SINAEPOURFARD, A.; GARCIA, J.; MASIP-BRUIN, X.; MARÍN-TORDERA, E. A comprehensive scenario agnostic data lifecycle model for an efficient data complexity management. In: **2016 IEEE 12th International Conference on e-Science (e-Science)**. [S.l.: s.n.], 2016. p. 276–281. Citado na página 25.
- SKOPAL, T.; LOKOČ, J. New dynamic construction techniques for m-tree. **Journal of Discrete Algorithms**, Elsevier, v. 7, n. 1, p. 62–77, 2009. Citado na página 40.
- SKOPAL, T.; POKORNÝ, J.; SNASEL, V. Pm-tree: Pivoting metric tree for similarity search in multimedia databases. In: **ADBIS (Local Proceedings)**. [S.l.: s.n.], 2004. Citado na página 40.

- SOUZA, J. A. de; CAZZOLATO, M. T.; TRAINA, A. J. M. Clusmam: Fast and effective unsupervised clustering of large complex datasets using metric access methods. In: **Proceedings of the 31st Annual ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2016. (SAC '16), p. 986–991. ISBN 978-1-4503-3739-7. Disponível em: <http://doi.acm.org/10.1145/2851613.2851661>>. Citado nas páginas 19 e 30.
- SOUZA, J. A. de; RAZENTE, H. L.; BARIONI, M. C. N. Faster construction of ball-partitioning-based metric access methods. In: **Proceedings of the 28th Annual ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2013. (SAC '13), p. 8–12. ISBN 978-1-4503-1656-9. Disponível em: <http://doi.acm.org/10.1145/2480362.2480365>>. Citado nas páginas 77 e 81.
- _____. Optimizing metric access methods for querying and mining complex data types. **Journal of the Brazilian Computer Society**, Springer, v. 20, n. 1, p. 17, 2014. Citado nas páginas 42 e 56.
- SOUZA, J. A. de; TRAINA, A. J. M.; MICHEL, S. Class-constraint similarity queries. In: **Proceedings of the 33rd Annual ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2018. (SAC '18), p. 549–556. ISBN 978-1-4503-5191-1. Disponível em: <http://doi.acm.org/10.1145/3167132.3167192>>. Citado na página 80.
- TRAINA JR., C.; TRAINA, A.; FALOUTSOS, C.; SEEGER, B. Fast indexing and visualization of metric data sets using slim-trees. **IEEE Trans. on Knowl. and Data Eng.**, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 14, n. 2, p. 244–260, mar. 2002. ISSN 1041-4347. Disponível em: <http://dx.doi.org/10.1109/69.991715>>. Citado nas páginas 40 e 55.
- TRAINA-JR, C.; TRAINA, A. J.; VIEIRA, M. R.; FALOUTSOS, C. *et al.* The omni-family of all-purpose access methods: a simple and effective way to make similarity search more efficient. **The VLDB Journal—The International Journal on Very Large Data Bases**, Springer-Verlag New York, Inc., v. 16, n. 4, p. 483–505, 2007. Citado na página 40.
- TRAUWAERT, E. On the meaning of dunn's partition coefficient for fuzzy clusters. **Fuzzy sets and systems**, Elsevier, v. 25, n. 2, p. 217–242, 1988. Citado na página 51.
- TZORTZIS, G.; LIKAS, A. The minmax k-means clustering algorithm. **Pattern Recognition**, Elsevier, v. 47, n. 7, p. 2505–2516, 2014. Citado na página 49.
- UHLMANN, J. Implementing metric trees to satisfy general proximity/similarity queries. In: **Proc. Command and Control Symposium, Washington, DC**. [S.l.: s.n.], 1991. Citado na página 39.
- UTHAYAKUMAR, J.; VENGATTARAMAN, T.; DHAVACHELVAN, P. A survey on data compression techniques: From the perspective of data quality, coding schemes, data type and applications. **Journal of King Saud University - Computer and Information Sciences**, 2018. ISSN 1319-1578. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1319157818301101>>. Nenhuma citação no texto.
- VENDRAMIN, L.; CAMPELLO, R. J. G. B.; HRUSCHKA, E. R. Relative clustering validity criteria: A comparative overview. **Stat. Anal. Data Min.**, John Wiley & Sons, Inc., New York, NY, USA, v. 3, n. 4, p. 209–235, ago. 2010. ISSN 1932-1864. Disponível em: <http://dx.doi.org/10.1002/sam.v3:4>>. Citado nas páginas 51 e 61.

- VIEIRA, M. R.; JR, C. T.; CHINO, F. J.; TRAINA, A. J. Dbm-tree: A dynamic metric access method sensitive to local density data. **Journal of Information and Data Management**, v. 1, n. 1, p. 111, 2010. Citado na página 40.
- WANG, L.; BEZDEK, J. C.; LECKIE, C.; KOTAGIRI, R. Selective sampling for approximate clustering of very large data sets. **Int. J. Intell. Syst.**, John Wiley & Sons, Inc., New York, NY, USA, v. 23, n. 3, p. 313–331, mar. 2008. ISSN 0884-8173. Disponível em: <<http://dx.doi.org/10.1002/int.v23:3>>. Citado nas páginas 27 e 55.
- WANG, W.; YANG, C.; CHEN, H.; FENG, X. Unified discriminative and coherent semi-supervised subspace clustering. **IEEE Transactions on Image Processing**, v. 27, n. 5, p. 2461–2470, May 2018. ISSN 1057-7149. Citado na página 58.
- WITTEN, I. H.; FRANK, E.; HALL, M. A.; PAL, C. J. **Data Mining: Practical machine learning tools and techniques**. [S.l.]: Morgan Kaufmann, 2016. Citado na página 50.
- WOLBERG, W. H.; NICK, W.; MANGASARIAN, O. L. **Machine Learning Repository**. 2018. Disponível em: <[https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+\(Diagnostic\)](https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic))>. Citado na página 60.
- XU, R.; WUNSCH, D. **Clustering**. [S.l.]: John Wiley & Sons, 2008. v. 10. Citado na página 51.
- YIANILOS, P. N. Data structures and algorithms for nearest neighbor search in general metric spaces. In: **SODA**. [S.l.: s.n.], 1993. v. 93, n. 194, p. 311–321. Citado na página 39.
- ZEZULA, P.; AMATO, G.; DOHNAL, V.; BATKO, M. **Similarity Search: The Metric Space Approach**. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2010. ISBN 1441939725, 9781441939722. Citado nas páginas 34, 40, 45 e 67.
- ZHANG, Q.; YANG, L. T.; CHEN, Z.; LI, P. High-order possibilistic c-means algorithms based on tensor decompositions for big data in iot. **Information Fusion**, v. 39, p. 72 – 80, 2018. ISSN 1566-2535. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1566253517302245>>. Citado na página 55.
- ZHANG, Y.; LI, X.; WANG, J.; ZHANG, Y.; XING, C.; YUAN, X. An efficient framework for exact set similarity search using tree structure indexes. In: **2017 IEEE 33rd International Conference on Data Engineering (ICDE)**. [S.l.: s.n.], 2017. p. 759–770. Citado nas páginas 28, 29 e 79.
- ZHAO, X.; XIAO, C.; LIN, X.; ZHANG, W.; WANG, Y. Efficient structure similarity searches: A partition-based approach. **The VLDB Journal**, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 27, n. 1, p. 53–78, fev. 2018. ISSN 1066-8888. Disponível em: <<https://doi.org/10.1007/s00778-017-0487-0>>. Citado na página 29.