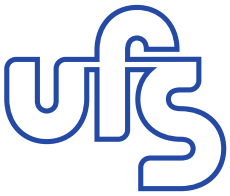


UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Metaheurísticas Inspiradas na Natureza Aceleradas por GPU para Agrupamento de Dados com Múltiplas Visões

Dissertação de Mestrado

Mariana Lira de Farias

São Cristóvão – Sergipe

2026

UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Mariana Lira de Farias

Metaheurísticas Inspiradas na Natureza Aceleradas por GPU para Agrupamento de Dados com Múltiplas Visões

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de mestre em Ciência da Computação.

Orientador(a): Renê Pereira de Gusmão

São Cristóvão – Sergipe

2026

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL
UNIVERSIDADE FEDERAL DE SERGIPE

Farias, Mariana Lira de
F224m Metaheurísticas inspiradas na natureza aceleradas por gpu para agrupamento de dados com múltiplas visões / Mariana Lira de Farias ; orientador Renê Pereira de Gusmão. - São Cristóvão, 2025.
77 f.; il.

Dissertação (mestrado em Ciência da Computação) – Universidade Federal de Sergipe, 2025.

1. Aprendizado do computador. 2. Otimização matemática. 3. Programação heurística. I. Gusmão, Renê Pereira de orient. II. Título.

CDU 004

Dedico este trabalho aos meus pais e ao meu irmão mais velho, que sob muito sol me fizeram chegar até aqui, na sombra.

Agradecimentos

Agradeço, primeiramente, a Deus, cuja presença iluminou cada passo desta jornada e fortaleceu meu caminho nos momentos de incerteza.

À minha família, que me ofereceu suporte incondicional ao longo de todo o percurso. Sem o amor, o incentivo e a confiança deles, esta caminhada não teria sido possível.

Ao meu noivo, por ser meu alicerce durante o último ano, oferecendo apoio, paciência e compreensão nos dias mais intensos deste processo.

Ao meu orientador, pelas valiosas sugestões, pelos direcionamentos ao longo do desenvolvimento desta pesquisa. Sua orientação foi essencial para o amadurecimento e conclusão deste trabalho.

Agradeço também a todos os professores, em especial do PROCC, da Universidade Federal de Sergipe, que contribuíram direta e indiretamente para a realização desta pesquisa. Cada disciplina cursada, cada conversa e apoio logístico foram fundamentais para que este trabalho se concretizasse.

“A persistência é o caminho do êxito.” (Charles Chaplin)

Resumo

O agrupamento multivisual surgiu como uma estrutura poderosa para integrar espaços de características heterogêneos em um processo de aprendizagem unificado. No entanto, a otimização de modelos multivisuais permanece computacionalmente exigente, especialmente quando se utilizam metaheurísticas baseadas em população em grandes conjuntos de dados. Este trabalho apresenta adaptações aceleradas em GPU de seis algoritmos meta-heurísticos — Particle Swarm Optimization (PSO), Cuckoo Search (CS), Firefly Algorithm (FA), Elephant Herding Optimization (EHO), Bees Algorithm (BA) e Black Hole (BH) — aplicados ao agrupamento multivisual. Cada algoritmo foi reformulado para operar sobre uma representação unificada de centróides e otimizado sob duas funções objetivo: uma versão completa derivada do TW-K-Means e uma versão simplificada ponderada por visão (VWOF). As implementações propostas exploram paralelismo em GPU e computação vetorizada para alcançar reduções substanciais no tempo de execução, mantendo a qualidade do agrupamento. Experimentos em sete bases de dados de referência demonstram melhorias consistentes em relação aos métodos tradicionais de agrupamento multivisual, tanto em acurácia (ARI, NMI, Pureza) quanto em eficiência computacional, com ganhos que variam entre algoritmos e alcançam picos superiores a 50× em cenários altamente paralelizáveis. Análises estatísticas confirmam a significância das diferenças observadas, reforçando o potencial da aceleração em GPU para o agrupamento multivisual baseado em metaheurísticas.

Palavras-chave: Agrupamento. Múltiplas visões dos dados. Otimização. Metaheurísticas.

Abstract

Multi-view clustering has emerged as a powerful framework for integrating heterogeneous feature spaces into a unified learning process. Nevertheless, the optimization of multi-view models remains computationally demanding, particularly when employing population-based metaheuristics on large datasets. This paper presents GPU-accelerated adaptations of six metaheuristic algorithms — Particle Swarm Optimization (PSO), Cuckoo Search (CS), Firefly Algorithm (FA), Elephant Herding Optimization (EHO), Bees Algorithm (BA), and Black Hole (BH) — applied to multi-view clustering. Each algorithm was reformulated to operate on a unified centroid representation and optimized under two objective functions: a full version derived from TW-K-Means and a simplified view-weighted version (VWOF). The proposed implementations exploit GPU parallelism and vectorized computation to achieve substantial reductions in runtime while maintaining clustering quality. Experiments on seven benchmark datasets demonstrate consistent improvements over traditional multi-view clustering methods in both clustering accuracy (ARI, NMI, Purity) and computational efficiency, achieving speedups of up to 50×. Statistical analyses confirm the significance of the observed differences, reinforcing the potential of GPU acceleration for metaheuristic-based multi-view clustering.

Keywords: Clustering. Multi-view data. Optimization. Meta-heuristics.

Lista de ilustrações

Lista de tabelas

Lista de algoritmos

Lista de abreviaturas e siglas

MVC	Agrupamento de dados de múltiplas visões
TWK	*Two-Weight K-Means*
VWOF	*View-Weighted Objective Function*
IMN	Informação Mútua Normalizada
DCOMP	Departamento de Computação
UFS	Universidade Federal de Sergipe
PSO	Particle Swarm Optimization
FA	*Firefly Algorithm*
CS	*Cuckoo Search*
BH	*Black Hole Algorithm*
EHO	*Elephant Herding Optimization*
BA	*Bees Algorithm*
TWK	Função objetivo *Two-Weight K-Means*
VWOF	Função objetivo *View-Weighted Objective Function*
KMeans	*K-Means Clustering*
EWK	*Entropy Weighting K-Means*
MVKKM	*Multi-View Multiple Kernel K-Means*
MVSC	*Multi-View Spectral Clustering*
ARI	*Adjusted Rand Index*
NMI	*Normalized Mutual Information*
GPU	*Graphics Processing Unit*
CPU	*Central Processing Unit*
CUDA	*Compute Unified Device Architecture*
NUMPY	*Numerical Python*
CUPY	*NumPy-like API para CUDA*

UNIVERSIDADE FEDERAL DE SERGIPE
PRÓ-REITORIA DE PÓS-GRADUAÇÃO E PESQUISA
COORDENAÇÃO DE PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Ata da Sessão Solene de Defesa da Dissertação do Curso de Mestrado em Ciência da Computação-UFS.
Candidata: MARIANA LIRA DE FARIAS

Em 19 dias do mês de dezembro do ano de dois mil e vinte cinco, com início às 10:00hs, realizou-se na Sala de Seminários do PROCC da Universidade Federal de Sergipe, na Cidade Universitária Prof. José Aloísio de Campos, a Sessão Pública de Defesa de Dissertação de Mestrado da candidata **Mariana Lira de Farias** que desenvolveu o trabalho intitulado: **"Metaheurísticas Inspiradas na Natureza Aceleradas por GPU para Agrupamento de Dados com Múltiplas Visões",** sob a orientação do Prof. Dr. **Renê Pereira de Gusmão.** A Sessão foi presidida pelo Prof. Dr. **Renê Pereira de Gusmão** (PROCC/UFS), que após a apresentação da dissertação passou a palavra aos outros membros da Banca Examinadora, o Dr. **Bruno Almeida Pimentel (UFAL)** e, em seguida, Dr. **André Britto de Carvalho** (PROCC/UFS). Após as discussões, a Banca Examinadora reuniu-se e considerou o mestrando (a) APROVADA "*(aprovado/reprovado)*". Atendidas as exigências da Instrução Normativa 05/2019/PROCC, do Regimento Interno do PROCC (Resolução 67/2014/CONEPE), e da Resolução nº 04/2021/CONEPE que regulamentam a Apresentação e Defesa de Dissertação, e nada mais havendo a tratar, a Banca Examinadora elaborou esta Ata que será assinada pelos seus membros e pelo mestrando.

Cidade Universitária "*Prof. José Aloísio de Campos",* 19 de dezembro de 2025.

gov.br Documento assinado digitalmente
RENE PEREIRA DE GUSMAO
Data: 19/12/2025 14:56:37-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Renê Pereira de Gusmão
(PROCC/UFS)
Presidente

gov.br Documento assinado digitalmente
BRUNO ALMEIDA PIMENTEL
Data: 19/12/2025 15:08:15-0300
Verifique em https://validar.iti.gov.br

Prof. Dr Bruno Almeida Pimentel
(UFAL)
Examinador Externo ao programa

gov.br Documento assinado digitalmente
ANDRE BRITTO DE CARVALHO
Data: 07/01/2026 11:30:01-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. André Britto de Carvalho
(PROCC/UFS)
Examinador Interno

gov.br Documento assinado digitalmente
MARIANA LIRA DE FARIAS
Data: 07/01/2026 14:30:41-0300
Verifique em https://validar.iti.gov.br

Mariana Lira de Farias
Candidata

Sumário

1

Introdução

A crescente disponibilidade de dados provenientes de múltiplas fontes é uma realidade (HAN; PEI; KAMBER, 2011). O cenário é resultado da informatização da nossa sociedade e do rápido desenvolvimento de tecnologias de coleta e extração de dados que traz, por sua vez, a necessidade de ferramentas e técnicas para extrair informações valiosas desses dados (YE et al., 2018). Nesse contexto, se destacam as técnicas de mineração de dados e reconhecimento de padrão, a exemplo da tarefa de agrupamento de dados.

De acordo com Chao et al. (2017), agrupamento é uma subárea da aprendizagem de máquina responsável por identificar subgrupos homogêneos - também chamados de *clusters* - e coesos dentro de um conjunto de dados. Os métodos tradicionais usam apenas uma visão, ou seja, uma única fonte específica dos dados. No entanto, uma mesma informação pode ser descrita por diversas perspectivas e fontes e, à medida que se ampliam as visões dos dados, surge a necessidade de estender a tarefa de agrupamento para além da análise univariada, integrando essas múltiplas visões.

Segundo Ye et al. (2018), *o multi-view clustering* (MVC) pode ser definido como a tarefa de agrupar dados da mesma classe, mas com múltiplas representações, oriundas várias fontes de informação. Chao, Sun e Bi (2021) afirma que a área está em crescente consolidação devido à habilidade de lidar com a complexidade dos dados contemporâneos, resultando em avanços significativos em diversas áreas do conhecimento, conforme indica.

No que diz respeito a categorização dos algoritmos de particionamento, esta baseia-se nos métodos de agrupamento tradicionais e é estabelecida de acordo com características específicas das técnicas e abordagens utilizadas. De cordo com Chao et al. (2017), com a aplicação do MVC, espera-se explorar dados redundantes e complementares para obter resultados mais robustos em diversos campos, como visão computacional, mídia social e bioinformática. Dentro desse contexto, segundo Nanda e Panda (2014), um dos elementos cruciais, independentemente do domínio de aplicação, é a busca por agrupamentos precisos, que envolvem a determinação do

número ideal de *clusters* e a maximização da similaridade entre os dados que compõem esses grupos.

Conforme mencionado por Hussain et al. (2019) e Minh et al. (2022), a aplicação de processos de otimização é uma abordagem eficiente para garantir resultados mais precisos. Um dos principais métodos de otimização são as metaheurísticas, que abrangem desde procedimentos simples de busca local até processos complexos de aprendizagem, permitindo explorar diversas soluções e encontrar a mais adequada para um problema específico, dentro das restrições e flexibilidades fornecidas. Em particular, metaheurísticas inspiradas na natureza são amplamente aplicadas para obter soluções satisfatórias para o problema de agrupamento em uma escala de tempo aceitável (NANDA; PANDA, 2014).

No entanto, mesmo com metaheurísticas, o agrupamento de dados continua sendo uma tarefa computacionalmente exigente em conjuntos de dados de grande escala ou alta dimensionalidade. Nesse contexto, a eficiência torna-se um fator essencial, e a natureza populacional das metaheurísticas as torna particularmente adequadas à execução paralela. A aceleração por GPU surge, assim, como uma abordagem natural para aumentar a escalabilidade e reduzir o tempo de execução, mantendo a eficácia dos métodos (VERONESE; KROHLING, 2018).

1.1 Justificativa

De acordo com Das, Abraham e Konar (2009), o agrupamento de dados tem atraído atenção significativa de pesquisadores que buscam propor algoritmos capazes de lidar com a complexidade crescente dos grandes volumes de dados do mundo real. Para José-García e Gómez-Flores (2016), dois desafios se destacam: determinar o número ideal de *clusters* e identificar corretamente todos os grupos presentes nos dados. A precisão desse processo é especialmente relevante quando aplicada a áreas sensíveis ou de alto impacto.

Cowgill, Harvey e Watson (1999) afirmam que o problema de agrupamento, na busca pela solução ótima, é classificado como NP-difícil, mesmo para conjuntos de dados de tamanho moderado. Quando os dados apresentam múltiplas visões, a integração dessas diferentes representações aumenta a complexidade do problema, exigindo métodos de otimização mais sofisticados. Nesse cenário, a utilização de técnicas de aceleração computacional, como o processamento paralelo em GPU, surge como uma alternativa promissora para lidar com o aumento do custo computacional, mantendo a precisão dos resultados (VERONESE; KROHLING, 2018).

Como indicam Hruschka et al. (2009), o uso de metaheurísticas tem se mostrado eficiente na otimização de tarefas de agrupamento. Entre essas abordagens, as metaheurísticas inspiradas na natureza se destacam pela flexibilidade e capacidade de adaptação, sendo aplicadas com sucesso a diversos problemas de engenharia e ciência de dados (MINH et al., 2022). Dessa forma, a combinação entre metaheurísticas e processamento paralelo em GPU constitui uma contribuição capaz de aumentar a escalabilidade e o desempenho do agrupamento multivisual.

1.2 Objetivo Geral

Este trabalho tem como objetivo geral investigar adaptações de metaheurísticas inspiradas na natureza no contexto de MVC e seu impacto. Para isso, os objetivos específicos abaixo serão seguidos.

1. Realizar uma Revisão Sistemática de Literatura para explorar bases relevantes, mapear e analisar estudos associados ao agrupamento de dados de múltiplas visões, acompanhando o avanço dessa área e contextualizando as áreas de aplicação.
2. Identificar a metodologia, algoritmo de particionamento, meta-heurísticas e índices de validação que serão utilizados nos experimentos.
3. Desenvolver e implementar versões paralelizadas em GPU das meta-heurísticas para MVC;
4. Executar experimentos comparativos com métodos tradicionais e acelerados, avaliando eficiência computacional e qualidade de agrupamento;
5. Analisar os resultados obtidos, destacando os ganhos de desempenho e as contribuições metodológicas das adaptações propostas.

1.3 Organização do Trabalho

Este trabalho está estruturado em seis capítulos, organizados de forma a conduzir o leitor desde os fundamentos teóricos até a análise experimental e as conclusões finais.

O Capítulo 1 — Introdução apresenta o contexto geral do problema de agrupamento de dados em múltiplas visões, destacando sua relevância em cenários de alta dimensionalidade e heterogeneidade. São expostas as motivações do estudo, os objetivos geral e específicos, bem como as contribuições desenvolvidas.

O Capítulo 2 — Fundamentação Teórica reúne os principais conceitos necessários para a compreensão deste trabalho, incluindo técnicas de agrupamento, funções objetivo, métricas de avaliação e princípios das metaheurísticas inspiradas na natureza.

O Capítulo 3 — Revisão Sistemática da Literatura descreve o protocolo adotado, abrangendo critérios de inclusão e exclusão, estratégias de busca e etapas de seleção dos estudos. São analisadas as abordagens existentes para metaheurísticas aplicadas ao agrupamento de múltiplas visões, destacando limitações, tendências e lacunas que motivaram as adaptações propostas nesta dissertação.

O Capítulo 4 — Metodologia detalha as adaptações realizadas nas metaheurísticas escolhidas e o desenho experimental. Também apresenta os conjuntos de dados utilizados, os

parâmetros experimentais, a arquitetura de execução com suporte a CPU e GPU, além dos procedimentos de avaliação estatística empregados.

O Capítulo 5 — Resultados apresenta, analisa e discute os resultados dos experimentos. Inicialmente, avalia-se o desempenho comparativo das duas funções objetivo. Em seguida, são examinados os resultados alcançados pelas metaheurísticas nas diferentes bases de dados, incluindo métricas, testes estatísticos e análise de desempenho computacional (CPU versus GPU).

Por fim, o Capítulo 6 — Considerações Finais sintetiza as principais conclusões obtidas, destacando as contribuições do estudo, as limitações identificadas e possíveis direções para trabalhos futuros, especialmente no contexto de otimização computacional e aprimoramento de algoritmos de agrupamento multivisual.

2

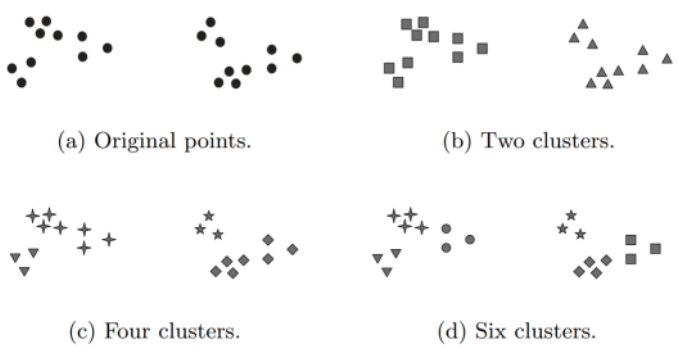
Fundamentação Teórica

Neste capítulo serão abordados os conceitos dos termos e áreas referenciados durante o trabalho.

2.1 Agrupamento de Dados

É uma técnica amplamente utilizada na análise exploratória de conjuntos de dados, visando identificar estruturas e padrões de similaridade entre os dados. Segundo Han, Pei e Kamber (2011), agrupamento é a tarefa de classificação que envolve a divisão de um conjunto de dados em subconjuntos em que os elementos dentro de cada subconjunto são semelhantes entre si, mas distintos dos elementos de outros subconjuntos. Reddy e Vinzamuri (2014) afirmam que a definição básica pode variar significativamente conforme o método de agrupamento utilizado, como um modelo gerativo probabilístico ou uma abordagem baseada em distância, e é influenciada pelo tipo específico de dados envolvidos. Na Figura 1 está ilustrada um exemplo de agrupamento.

Figura 1 – Exemplo de agrupamento de um conjunto de dados



Fonte: (STEINBACH; KUMAR; TAN, 2005)

De acordo com Steinbach, Kumar e Tan (2005), o agrupamento é comumente referido como uma classificação não supervisionada, uma vez que não são fornecidas informações prévias sobre a classificação dos dados, e o processo de agrupamento baseia-se nas características intrínsecas dos próprios dados.

Existem diferentes abordagens e técnicas para realizar o particionamento de dados. Cada abordagem possui suas próprias características e pressupostos subjacentes, e a escolha do método mais adequado depende das características dos dados e dos objetivos da análise. Para Han, Pei e Kamber (2011) é desafiador categorizar precisamente os métodos de agrupamento devido à sobreposição de características entre as categorias; no entanto, é útil organizar os métodos principais em categorias fundamentais, dado a diversidade de algoritmos presentes na literatura. Na Figura 2 é possível observar as principais categorias.

Figura 2 – Principais categorias de agrupamento

Método	Características gerais dos Métodos
Métodos de Particionamento	- Baseados em distância - Encontram clusters mutuamente exclusivos de forma esférica - Podem usar a média ou o medoide (etc.) para representar o centro do cluster - Eficazes para conjuntos de dados de pequeno a médio porte
Métodos Hierárquicos	- Agrupamento é uma decomposição hierárquica (ou seja, múltiplos níveis) - Não podem corrigir fusões ou divisões errôneas - Podem incorporar outras técnicas como microagrupamento ou considerar "vinculações" de objetos
Métodos Baseados em Densidade	- Podem encontrar clusters de formas arbitrárias - Clusters são regiões densas de objetos no espaço que são separadas por regiões de baixa densidade - Densidade do cluster: Cada ponto deve ter um número mínimo de pontos dentro de sua vizinhança" - Podem filtrar outliers
Métodos Baseados em Grade	- Usam uma estrutura de dados de grade multirresolução - Tempo de processamento rápido (tipicamente independente do número de objetos de dados, mas dependente do tamanho da grade)

Fonte: Adaptada de (HAN; PEI; KAMBER, 2011)

Independente da categoria dos algoritmos de particionamento, a função objetivo desempenha um papel fundamental. Ela é responsável por quantificar a qualidade dos grupos formados e pode ser classificada como monoobjetiva ou multiobjetiva, dependendo do contexto específico do problema. Segundo Mukhopadhyay, Maulik e Bandyopadhyay (2015), os algoritmos de agrupamento tradicionais geralmente utilizam uma função objetivo específica, otimizada por meio de técnicas clássicas ou metaheurísticas, essa abordagem é conhecida como agrupamento monoobjetivo. No entanto, em conjuntos de dados complexos, a utilização de apenas uma função objetivo pode ser inviável. Para contornar essa limitação, adota-se uma abordagem denominada agrupamento multiobjetivo, na qual múltiplas funções objetivo são otimizadas simultaneamente.

Outro tipo de categorização está relacionada ao pertencimento de cada objeto nos grupos,

os algoritmos podem ser classificados entre rígido e difuso. Conforme indicado por Steinbach, Kumar e Tan (2005), nos algoritmos baseados em métodos rígidos, cada objeto pertence a apenas um grupo, enquanto nos algoritmos com abordagem difusa, é atribuído a cada objeto um grau de pertencimento a todos os grupos formados.

Na perspectiva dos dados utilizadas, o primeiro ponto a se destacar é a ausência de dados em uma visão. Uma visão de dados refere-se a uma representação específica de uma informação em um conjunto de dados. Conforme ressaltado por Shao et al. (2016), é comum encontrar fontes de dados que apresentam ausência de instâncias devido à natureza das informações ou ao processo de coleta. Para enfrentar esse desafio, são empregados métodos de agrupamento capazes de lidar com a incompletude das visões. Dessa forma, os algoritmos podem ser classificados a partir métodos para dados completos e para dados incompletos. Além disso, é importante considerar a quantidade de visões utilizadas no método de particionamento. Segundo Ye et al. (2018), enquanto os algoritmos tradicionais se limitam a uma única visão, os algoritmos de agrupamento de dados de múltiplas visões exploram várias visualizações, ampliando assim a análise e compreensão dos dados.

Por fim, é importante destacar que o sucesso do agrupamento de dados não depende apenas da aplicação correta do algoritmo, mas também da interpretação dos resultados e da validação dos grupos obtidos.

2.1.1 Agrupamento de Dados de Múltiplas Visões

A MVC é uma abordagem avançada na análise de dados para lidar com conjuntos de dados complexos e heterogêneos. Segundo Yang e Wang (2018), essa abordagem é crucial, pois permite explorar características únicas de cada visão para fornecer uma descrição mais abrangente dos objetos de dados e insights mais profundos sobre o agrupamento interno. Por exemplo, no processamento de imagens, características têm propriedades distintas que, quando combinadas, enriquecem a análise e o entendimento dos dados.

A ideia central por trás do agrupamento de múltiplas visões é integrar as diferentes visualizações dos dados de maneira eficaz para obter agrupamentos mais robustos e precisos do que os métodos tradicionais que consideram apenas uma única visão dos dados.

As estratégias para agrupamento de dados com múltiplas visões podem ser divididas em três categorias distintas de acordo com o momento da combinação das visões: concatenação, distribuída ou centralizada. Segundo Cleuziou et al. (2009), na abordagem de concatenação, a combinação das visões ocorre antes do agrupamento, unificando-as em uma única visão. Na abordagem distribuída, cada visão é usada de forma independente para agrupar os dados, e uma partição final é gerada a partir dos agrupamentos obtidos. Na estratégia centralizada, todas as visões são utilizadas simultaneamente no processo de agrupamento, representando um desafio significativo. Os métodos desenvolvidos neste trabalho se enquadram na estratégia centralizada.

Em última análise, conforme indica Yang e Wang (2018), o MVC oferece uma abordagem poderosa para lidar com a complexidade dos dados do mundo real, permitindo a extração de informações mais ricas e significativas a partir de conjuntos de dados heterogêneos.

2.1.2 Índices de Validação

A validação dos resultados é uma etapa essencial no processo de agrupamento de dados. Conforme apontado por Halkidi, Batistakis e Vazirgiannis (2001), os índices de validação podem ser internos, os quais avaliam o resultado do agrupamento de um algoritmo utilizando apenas quantidades e características inerentes ao conjunto de dados, e externos, que compara o resultados com rótulos ou dados conhecidos previamente. A seguir, serão descritos índices de validação popularmente conhecidos.

2.1.2.1 Pureza

A Pureza é um índice externo de validação que mede o grau de correspondência entre os agrupamentos obtidos e as classes reais conhecidas. Ela avalia o quão homogêneos são os *clusters* formados, verificando se cada grupo contém majoritariamente elementos de uma mesma classe. Em outras palavras, indica a proporção de elementos corretamente agrupados de acordo com as classes de referência.

Sua fórmula é apresentada na Equação 2.1.

$$Purity = \frac{1}{n} \sum_{j=1}^{k} \max_{i} |C_j \cap L_i| \tag{2.1}$$

Fonte: Adaptado de Manning, Raghavan e Schütze (2008).

Na Equação 2.1, n representa o número total de objetos, k é o número de *clusters* gerados pelo algoritmo, C_j é o conjunto de elementos pertencentes ao *cluster* j, e L_i representa a classe i na partição de referência. O termo $|C_j \cap L_i|$ corresponde ao número de elementos comuns entre o *cluster* j e a classe i.

O valor da Pureza varia de 0 a 1, sendo que valores próximos de 1 indicam uma forte correspondência entre os *clusters* e as classes reais, ou seja, uma maior homogeneidade nos agrupamentos.

2.1.2.2 Medida F

Esse índice externo combina os conceitos de precisão e revocação, se trata da média harmônica entre esses dois valores. Ele fornece uma única métrica que leva em consideração tanto os falsos positivos quanto os falsos negativos, sendo útil quando há um desequilíbrio entre as classes. Sua fórmula é:

$$F = 2 \cdot \frac{\text{precisão}(i,j) \cdot \text{revocação}(i,j)}{\text{precisão}(i,j) + \text{revocação}(i,j)} \tag{2.2}$$

Fonte: Rendón et al. (2011)

As formulas de Precisão e Revocação são:

$$\text{Precisão(i,j)} = \frac{n_{ij}}{n_j} \tag{2.3}$$

Fonte: Rendón et al. (2011)

$$\text{Revocação(i,j)} = \frac{n_{ij}}{n_i} \tag{2.4}$$

Fonte: Rendón et al. (2011)

No cálculo desse coeficiente, n_{ij} é o número de objetos da classe i que estão no *cluster* j, enquanto n_j é o número de objetos no *cluster* j, e n_i é o número de objetos na classe i. Seu valor varia de 0 a 1, quanto mais próximo de 1, melhor o resultado.

2.1.2.3 Informação Mútua Normalizada

A Informação Mútua Normalizada (NMI) foi proposto por Kernighan e Lin (1970). É uma medida externa derivada do conceito de Informação Mútua que mede a similaridade entre dois agrupamentos. O NMI normaliza esta medida para garantir que o valor esteja entre 0 e 1, facilitando a comparação entre diferentes agrupamentos e partições de referência. Sua fórmula é 2.5 e

$$NMI(X,Y) = \frac{I(X,Y)}{\sqrt{H(X) + H(Y)}} \tag{2.5}$$

Fonte: Rendón et al. (2011)

$I(X,Y)$ denota a informação mútua entre X e Y, e $H(X)$ denota a entropia (sua fórmula 2.6) de X; X é o agrupamento que deve ser avaliado enquanto Y é o agrupamento conhecido previamente.

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log P(x_i) \tag{2.6}$$

Fonte: Rendón et al. (2011)

A Informação Mútua $I(X,Y)$ entre duas partições está definida na Equação 2.7. Onde $P(x,y)$ representa a probabilidade conjunta de um ponto pertencer ao *cluster* x na partição X e ao *cluster* y na partição Y. Enquanto isso, $P(x)$ e $P(y)$ são as probabilidades de um ponto estar no *cluster* x de X e no *cluster* y de Y.

$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)} \tag{2.7}$$

2.1.2.4 Índice Rand Ajustado (ARI)

Proposto por Das e Mannila (2000), o ARI é um índice externo que consiste em comparar o resultado final de um agrupamento com uma classificação prévia, para calcular a similiradidade considerando os *clusters* em que cada dado está ligado. Seu valor varia de -1 a 1, sendo 1 representando agrupamentos idênticos, 0 uma concordância aleatória, e -1 indica total discordância.

$$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}} \tag{2.8}$$

Fonte: Adaptado de Reddy e Vinzamuri (2014)

Observe a Equação 2.8, n_{ij} é o número de pares de elementos que estão no mesmo grupo nas duas partições, a_i é o número de elementos no grupo i na primeira partição, b_j é o número de elementos no grupo j na segunda partição, n é o número total de elementos, e $\binom{x}{2}$ representa o número de maneiras de escolher 2 elementos de um conjunto de x elementos.

2.2 Algoritmos de Agrupamento

Nesta seção são apresentados os algoritmos de agrupamento utilizados como métodos de referência neste estudo: o *Two-Weighted K-Means* (TW-K-Means), o *Entropy-Weighted K-Means* (EW-K-Means), *Spectral Clustering* (MVSC) e o *Multiple Kernel K-Means* (MKKM). Os dois primeiros derivam do K-Means tradicional, adaptados para cenários de múltiplas visualizações que incorporam mecanismos de ponderação adaptativa que ajustam automaticamente a relevância de atributos — e, no caso do TW-K-Means, também das visões — durante o processo de agrupamento. Já o agrupamento espectral adota uma abordagem baseada em grafos, utilizando propriedades espectrais da matriz Laplaciana para identificar estruturas complexas de dados e permitir a formação de *clusters* não lineares.

Esses algoritmos foram escolhidos por representarem abordagens complementares: o K-Means e suas extensões lidam diretamente com a estrutura vetorial dos dados, enquanto o método espectral atua sobre uma representação de similaridade, permitindo a detecção de fronteiras não lineares. Assim, eles servem como base comparativa para a avaliação das metaheurísticas propostas neste trabalho, cuja otimização será guiada pelas funções objetivo descritas em 2.3.

2.2.1 K-Means

O algoritmo K-Means foi introduzido por MacQueen et al. (1967) como uma proposta para particionar dados em grupos com base em sua similaridade. A ideia central consiste em minimizar a variância intra-*cluster* sobre uma partição $S = \{S_1, S_2, \ldots, S_k\}$ de um espaço $\mathbb{E}^N$, buscando configurações em que os elementos de cada grupo estejam próximos de seus respectivos centróides. A motivação original para o desenvolvimento do K-Means surgiu no contexto de problemas de estruturação ótima da informação, nos quais o objetivo era minimizar a perda esperada associada à representação dos dados por meio de mensagens reduzidas. Embora o algoritmo não garanta encontrar a partição globalmente ótima, MacQueen demonstrou que ele tende a produzir soluções com variância intra-*cluster* relativamente baixa, tornando-se computacionalmente viável e prático mesmo para grandes amostras.

Apesar de sua eficiência, o K-Means tradicional considera apenas uma única representação dos dados. Com o aumento da disponibilidade de bases multivisuais, diversas variantes do algoritmo foram desenvolvidas para incorporar estratégias específicas de tratamento de múltiplas fontes de informação. Duas dessas variantes, o TW-K-Means e o EW-K-Means, são descritas a seguir.

2.2.1.1 TW-K-Means

O algoritmo é uma variação desenvolvida por Chen et al. (2011) e projetada para lidar com problemas de agrupamento de dados em múltiplas visões. Enquanto o K-Means convencional atualiza os centroides com base na média de todos os pontos atribuídos a cada *cluster*, o TW-K-Means adota uma abordagem diferente. O algoritmo considera tanto os pontos atribuídos ao grupo quanto os pontos que estão mais próximos do centróide do *cluster* atual, o que pode levar a resultados mais robustos, especialmente em conjuntos de dados com distribuições assimétricas ou *clusters* de tamanhos variados.

No algoritmo TW-K-Means, são atribuídos pesos às visões e variáveis para avaliar a compacidade das visões e a importância das variáveis. Esses pesos são incorporados à função de distância utilizada para a formação dos *clusters*, e o algoritmo inclui etapas adicionais para calcular automaticamente esses pesos durante o processo de agrupamento iterativo do k-means. O pseucódigo pode ser observado no Algoritmo 1. É importante destacar que tanto pseucódigo quanto as principais equações do TW-K-Means apresentadas no presente estudo, foram extraídos de Chen et al. (2011).

$$P(U;Z;V;W) = \sum_{l=1}^{k}\sum_{i=1}^{n}\sum_{t=1}^{T}\sum_{j\in G_t} u_{i,l} w_t v_j d(x_{i,j}; z_{l,j}) + \eta \sum_{j=1}^{m} v_j \log(v_j) + \lambda \sum_{t=1}^{T} w_t \log(w_t) \tag{2.9}$$

Fonte: Adaptado de Chen et al. (2011)

Input: Número de clusters k, número máximo de iterações T, parâmetros de regularização α e β

Output: Matrizes de partição U, centróides Z, pesos por atributo V e pesos por visão W

Inicialização:
Selecionar aleatoriamente k centróides iniciais Z^0;
for $t = 1$ **to** T **do**
    Inicializar pesos de visão $w^t = \frac{1}{T}$;
    for *cada visão* $j \in G_t$ **do**
        Inicializar pesos de atributo $v^j = \frac{1}{|G_t|}$;
    end
end
$r \leftarrow 0$;
Iterações:
repeat
    Atualizar matriz de partição U^{r+1} utilizando a Eq. 2.10;
    Atualizar centróides Z^{r+1} utilizando a Eq. 2.11;
    Atualizar pesos de atributo V^{r+1} utilizando a Eq. 2.12;
    Atualizar pesos de visão W^{r+1} utilizando a Eq. 2.13;
    $r \leftarrow r + 1$;
until *a função objetivo (Eq. 2.9) convergir para um mínimo local*;

Algorithm 1: Algoritmo TW-K-Means

Fonte: Adaptado de Chen et al. (2011).

O processo de agrupamento para particionar o conjunto de dados $X = \{x_1, x_2, ..., x_n\}$ contendo n pontos em k *clusters*, é modelado como a minimização da função objetivo apresentada na Equação 2.9. Conforme do autor, a primeira parcela da equação representa a soma das dispersões dentro dos *clusters*, enquanto a segunda e a terceira parcelas correspondem a duas entropias de peso negativas. Para minimizá-la, são solucionados quatro problemas de minimização por meio das equações 2.10, 2.11, 2.12 e 2.13.

Na função objetivo e demais equações, U é uma matriz de partição de dimensões $n \times k$, onde seus elementos $u_{i,l}$ são binários, com $u_{i,l} = 1$ indicando que o objeto i pertence ao cluster l. O conjunto $Z = \{Z_1, Z_2, ..., Z_k\}$ representa os centróides, enquanto $W = \{w_1, w_2, ..., w_T\}$ e $V = \{v_1, v_2, ..., v_m\}$ são conjuntos de pesos para as visões T e variáveis individuais m, respectivamente. Os parâmetros $\lambda > 0$ e $\epsilon > 0$ são valores que especificam as tolerâncias nas medidas de distância ou dissimilaridade $d(x_{i,j}, z_{l,j})$, as quais quantificam a diferença entre o objeto i e o centro Z_l no contexto da j-ésima variável.

$$\begin{cases} u_{i,l} = 1, & \text{if } D_l \leq D_s \text{ for } 1 \leq s \leq k, \\ u_{i,s} = 0, & \text{for } s \neq l. \end{cases}$$
$$\text{Onde } D_s = \sum_{t=1}^{T} \sum_{j \in G_t} w_t v_j d(x_{i,j}, z_{s,j}) \tag{2.10}$$

Fonte: Adaptado de Chen et al. (2011)

Se a variável é numérica:
$$z_{l,j} = \frac{\sum_{i=1}^{n} u_{i,l} x_{i,j}}{\sum_{i=1}^{n} u_{i,l}}$$
Se a variável é categórica: (2.11)

$z_{l,j} = a_j^r$, Onde a_j^r é a moda dos valores variáveis

Fonte: Adaptado de Chen et al. (2011)

$$v_j = \frac{\exp\left(\frac{-E_j}{\eta}\right)}{\sum_{h \in G_t} \exp \frac{E_h}{\eta}}$$
$$\text{Onde } E_j = \sum_{k=1}^{l} \sum_{i=1}^{n} \hat{u}_{i,l} \hat{w}_t d(x_{i,j}) \tag{2.12}$$

Fonte: Adaptado de Chen et al. (2011)

$$w_t = \frac{\exp\left(\frac{-D_t}{\lambda}\right)}{\sum_{h=1}^{T} \exp\left(\frac{-D_h}{\lambda}\right)}$$
$$\text{Onde } \hat{D}_t = \sum_{l=1}^{k} \sum_{i=1}^{n} \sum_{j \in G_t} \hat{u}_{i,l} \hat{v}_j d(x_{i,j}, \hat{z}_{l,j}) \tag{2.13}$$

Fonte: Adaptado de Chen et al. (2011)

2.2.1.2 EW-K-Means

O algoritmo EW-K-Means é uma extensão proposta por Jing, Ng e Huang (2007). Sua principal característica é a atribuição de pesos às dimensões, refletindo sua importância relativa na formação de cada cluster. Esses pesos são ajustados iterativamente com base na dispersão intra-*cluster* e regularizados por um termo de entropia — a principal inovação do algoritmo — presente na função objetivo. Esse termo penaliza soluções em que poucos atributos dominam o agrupamento, incentivando uma participação mais equilibrada entre as dimensões relevantes.

A função objetivo do EW-K-Means é apresentada na Equação 2.14.

$$F(W, Z, \Theta) = \sum_{l=1}^{k} \sum_{j=1}^{n} \sum_{i=1}^{m} w_{lj}\, \theta_{li}\, (x_{ji} - z_{li})^2 + \lambda \sum_{i=1}^{m} \theta_{li} \log(\theta_{li}) \tag{2.14}$$

Fonte: Adaptado de Jing, Ng e Huang (2007)

Na Equação 2.14, w_{lj} indica a atribuição binária do objeto j ao *cluster* l (sendo 1 se o objeto pertence ao *cluster* e 0 caso contrário); z_{li} representa a coordenada do centróide do *cluster* l na dimensão i; θ_{li} é o peso associado à dimensão i no *cluster* l; e λ é o parâmetro de regularização que controla a influência do termo entrópico.

A atualização dos pesos das dimensões é realizada de forma iterativa, conforme a Equação 2.15, onde E_i representa a distorção média associada à dimensão i. Esse processo ajusta dinamicamente a importância de cada atributo de acordo com sua contribuição para a compactação dos *clusters*.

$$\theta_{li} = \frac{\exp\left(-\frac{E_i}{\lambda}\right)}{\sum_{h=1}^{m} \exp\left(-\frac{E_h}{\lambda}\right)}, \qquad E_i = \sum_{j=1}^{n} w_{lj} (x_{ji} - z_{li})^2 \tag{2.15}$$

Os centróides são atualizados de maneira semelhante ao K-Means convencional, como mostrado na Equação 2.16, calculando-se a média ponderada dos objetos atribuídos a cada *cluster*.

$$z_{li} = \frac{\sum_{j=1}^{n} w_{lj}\, x_{ji}}{\sum_{j=1}^{n} w_{lj}} \tag{2.16}$$

O processo iterativo do EW-K-Means está resumido no Algoritmo 2. Inicialmente, os centróides são definidos aleatoriamente, e os pesos dos atributos são uniformes. Em seguida, o algoritmo alterna entre a atribuição dos objetos, o recálculo dos centróides e a atualização dos pesos das dimensões, até que a função objetivo atinja convergência.

Input: Número de clusters k, parâmetro de regularização λ

Output: Centróides Z e pesos das dimensões Θ

Inicialização:

Selecionar aleatoriamente k centróides iniciais Z^0;

Inicializar pesos das dimensões como $\theta_{li}^0 = \frac{1}{m}$, para todo atributo i da visão l;

Iterações:

repeat

    Atribuir cada objeto ao cluster mais próximo considerando os pesos $\Theta^{(t)}$;

    Atualizar centróides $Z^{(t+1)}$ conforme Eq. 2.16;

    Calcular E_i e atualizar os pesos $\Theta^{(t+1)}$ conforme Eq. 2.15;

until *convergência da função objetivo* F;

Algorithm 2: Algoritmo EW-K-Means

Fonte: Adaptado de Jing, Ng e Huang (2007).

Em resumo, o EW-K-Means introduz uma regularização baseada em entropia que ajusta automaticamente a relevância dos atributos, resultando em agrupamentos mais equilibrados e robustos. Em comparação ao TW-K-Means, sua formulação elimina o componente de múltiplas visões, reduzindo significativamente a complexidade computacional sem comprometer a qualidade das partições obtidas.

2.2.2 Agrupamento Espectral (*Spectral Clustering*)

O agrupamento espectral é um método de particionamento que utiliza propriedades espectrais de grafos, prosposta por Ng, Jordan e Weiss (2002), para identificar estruturas de agrupamento em conjuntos de dados. Diferentemente do K-Means e as variantes citadas, que opera diretamente no espaço original das variáveis, o Spectral Clustering transforma os dados em um novo espaço de representação, derivado dos autovetores de uma matriz de similaridade. Essa matriz, também chamada de matriz de afinidade, captura a relação de proximidade entre os objetos, permitindo que o algoritmo detecte estruturas não esféricas e agrupamentos complexos.

O método consiste em três etapas principais: (i) construção da matriz de similaridade S, cujos elementos s_{ij} representam o grau de afinidade entre os objetos x_i e x_j; (ii) cálculo do espectro da matriz Laplaciana normalizada L, obtida a partir de S; e (iii) aplicação do K-Means sobre os k autovetores associados aos menores autovalores de L, que formam uma representação vetorial reduzida dos dados.

A função Laplaciana pode ser expressa como:

$$L = D^{-1/2}(D - S)D^{-1/2} \tag{2.17}$$

onde D é a matriz diagonal de graus, com $D_{ii} = \sum_j s_{ij}$. A decomposição espectral de L permite projetar os dados em um espaço em que as fronteiras entre os agrupamentos se tornam mais lineares, tornando o K-Means mais eficaz na separação dos grupos.

O agrupamento espectral tem sido amplamente utilizado como método de referência em tarefas de *multi-view clustering*, pois permite capturar relações complexas entre as diferentes visões e identificar agrupamentos que não são detectáveis por métodos puramente baseados em distância euclidiana.

2.2.3 *Multi-View Kernel K-Means* (MVKKM)

O MVKKM é uma abordagem de agrupamento que combina múltiplas visões de dados por meio da integração de diferentes matrizes de kernel proposta por Liu et al. (2016), permitindo capturar relações não lineares e complementares entre as amostras. Dentre suas variantes, destaca-se a versão robusta proposta por Du et al. (2015), que incorpora uma penalização baseada na norma $\ell_{2,1}$, conferindo maior resistência a ruídos e visões irrelevantes.

O método busca minimizar a soma ponderada das distorções intra-*cluster* em todas as visões, definida por:

$$\min_{\mathbf{H},\mathbf{P}} \sum_{v=1}^{V} \left\| \mathbf{K}^{(v)} - \mathbf{H}\mathbf{H}^{\top} \right\|_{2,1} \quad \text{sujeito a} \quad \mathbf{H}^{\top}\mathbf{H} = \mathbf{I}, \tag{2.18}$$

em que $\mathbf{K}^{(v)}$ representa a matriz de kernel da v-ésima visão, e $\mathbf{H}$ é a matriz de rótulos binários que indica a associação das amostras aos *clusters*. A penalização pela norma $\ell_{2,1}$ promove a seleção automática de visões mais relevantes, reduzindo a influência de ruídos ou atributos redundantes.

A otimização é realizada de forma alternada, combinando atualização dos rótulos de agrupamento com reponderação dos kernels por visão. Essa estrutura permite ao MVKKM adaptar dinamicamente a contribuição de cada visão de acordo com sua consistência com a estrutura global dos dados.

2.3 Funções Objetivo

Segundo Nocedal e Wright (2006), a função objetivo é o elemento central de um processo de otimização, responsável por quantificar a qualidade de uma solução candidata.Em algoritmos de agrupamento, essa função atua como um critério de partição a ser minimizado, medindo a similaridade entre os objetos dentro de cada *cluster* (JAIN; MURTY; FLYNN, 1999).

Nesta seção são descritas as funções objetivo adotadas neste estudo, que orientam o processo de otimização das metaheurísticas. Foram utilizadas duas formulações principais: a função original do TW-K-Means, e uma versão simplificada denominada *View-Weighted*

Objective Function (VWOF), que pode ser interpretada como uma extensão direta da função objetivo do K-Means, adaptada para o contexto multivisual.

2.3.1 Função Objetivo do TW-K-Means

A função objetivo do TW-K-Means, apresentada na Equação 2.9, foi utilizada como referência neste estudo. Essa formulação estende o K-Means tradicional ao incluir pesos em dois níveis: pesos de visão (w_t) e pesos de atributo (v_j), atualizados iterativamente com base na compactação dos *clusters*. Os parâmetros η e λ controlam a regularização entrópica, evitando que apenas uma visão ou atributo domine o processo de agrupamento.

Essa função foi selecionada por capturar de forma eficiente a importância relativa de visões e atributos em contextos multivisuais, fornecendo uma base sólida de comparação para as metaheurísticas propostas.

2.3.2 View-Weighted Objective Function (VWOF)

A *View-Weighted Objective Function* (VWOF) é uma variação da função do K-Means para o cenário de múltiplas visões adotado em Zhang et al. (2024). Nessa formulação, cada visão recebe um peso proporcional à sua compacidade intra-*cluster*. Essa compacidade é obtida a partir da média das distâncias euclidianas quadráticas entre os objetos e seus respectivos centróides, considerando apenas os atributos pertencentes a cada visão. Os valores resultantes são então normalizados de forma inversa, atribuindo maior peso às visões que geram clusters mais coesos.

A função VWOF é expressa pela Equação 2.19.

$$J = \sum_{i=1}^{k} \sum_{x_j \in C_i} \sum_{t=1}^{m} w_t \, d_t(x_j, \mu_i) \tag{2.19}$$

Fonte: Função VWOF proposta neste estudo.

Na equação, C_i representa o conjunto de objetos pertencentes ao *cluster* i, e μ_i é o seu centróide. O termo $d_t(x_j, \mu_i)$ corresponde à distância euclidiana quadrática entre o objeto x_j e o centróide μ_i, considerando apenas os atributos da visão t. O peso w_t expressa a importância relativa de cada visão, sendo normalizado de modo que $\sum_{t=1}^{m} w_t = 1$.

2.3.3 Discussão Comparativa

As duas funções objetivo apresentam abordagens complementares em relação à ponderação das informações multivisuais. Enquanto a função do TW-K-Means realiza ponderação em dois níveis (visões e atributos), incluindo termos de regularização entrópica, a VWOF considera apenas as visões, simplificando o processo de otimização.

Em termos de custo computacional, ambas são dominadas pelo cálculo das distâncias entre objetos e centróides ($O(nkD)$), porém a função do TW-K-Means adiciona etapas de atualização de pesos e parâmetros de entropia, aumentando a complexidade. Já a VWOF apresenta menor sensibilidade a parâmetros e maior escalabilidade, sendo mais adequada para execução paralela em GPU.

A utilização de ambas as funções neste trabalho permite comparar o impacto de diferentes níveis de ponderação na qualidade do agrupamento e na eficiência das metaheurísticas aplicadas ao contexto multivisual.

2.4 Meta-heurísticas

Segundo Abdel-Basset, Abdel-Fatah e Sangaiah (2018), as metaheurísticas são estratégias de otimização computacional que buscam encontrar soluções sofisticadas para problemas de otimização em um tempo computacionalmente viável. De acordo com Hussain et al. (2019), se diferenciam dos métodos exatos, que garantem a obtenção da solução ótima, por oferecer abordagens aproximadas, porém eficientes com resultados simples e robustos, sendo assim mais preferíveis. Um algoritmo genérico para meta-heurísticas pode ser observado abaixo.

Input: Critério de parada e parâmetros específicos do algoritmo
Output: Melhor solução encontrada

```
1  Inicialização:
2  Gerar uma ou mais soluções iniciais da população;
3  Iterações:
4  while critério de parada não for satisfeito do
5      if condição favorece exploração then
6          Gerar nova(s) solução(ões) via operador de exploração (busca global);
7      else
8          Gerar nova(s) solução(ões) via operador de explotação (busca local);
9      end
10     Atualizar a melhor solução obtida até o momento;
11 end
```

Algorithm 3: Estrutura Algorítmica Abstrata para Metaheurísticas

Fonte: Adaptado de Abdel-Basset, Abdel-Fatah e Sangaiah (2018).

No contexto de algoritmos de particionamento, as metaheurísticas desempenham um papel crucial na obtenção de agrupamentos de alta qualidade em conjuntos de dados complexos e de grande escala. Segundo Abualigah et al. (2021), é possível encontrar na literatura diversas metaheurísticas para resolver problemas de agrupamento. De maneira geral, são aplicadas para

otimizar a função objetivo ajustando os parâmetros de acordo com a estrutura e a natureza dos dados.

Portanto, as metaheurísticas representam uma ferramenta poderosa para o problema de particionamento, oferecendo soluções eficazes e escaláveis. Dentre os algoritmos metaheurísticos existe uma categoria inspirada em processos e fenômenos naturais, são as metaheurísticas baseadas na natureza. Segundo Nanda e Panda (2014), as metaheurísticas inspiradas na natureza têm sido aplicadas para obter soluções satisfatórias para o problema de agrupamento. De acordo com Abdel-Basset, Abdel-Fatah e Sangaiah (2018) podem ser categorizadas em quatro grupos principais: baseadas em inteligência de enxame (SI), bioinspiradas (mas não baseadas em SI), baseadas em física/química e aquelas que não se enquadram nas categorias anteriores devido à diversidade de suas inspirações. A seguir, serão descritos alguns algoritmos mais populares.

2.4.1 *Particle Swarm Optimization* (PSO)

PSO foi inspirado pelo comportamento coletivo de animais. Segundo Abdel-Basset, Abdel-Fatah e Sangaiah (2018) e Castro e Tsuzuki (2007), é considerado um algoritmo de inteligência de enxame semi-evolutivo, onde cada solução candidata está explicitamente associada a um processo de busca, a uma velocidade e uma memória de suas melhores posições. Durante o processo de otimização, as partículas atualizam suas posições e velocidades com base em sua experiência individual (melhor posição pessoal, $pBest$) e na experiência do grupo (melhor posição global, $gBest$). As atualizações permitem que as partículas explorem o espaço de busca em busca de soluções ótimas. O ajuste da velocidade e da posição da partícula são baseadas nas seguintes equações respectivamente:

$$v_i^{t+1} = v_i^t + \alpha\varepsilon_1[pBest_i^t - x_i^t] + \alpha r_2\varepsilon_2(gBest - x_i^t) \tag{2.20}$$

Fonte: Adaptada de Abdel-Basset, Abdel-Fatah e Sangaiah (2018)

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2.21}$$

Fonte: Adaptada de Abdel-Basset, Abdel-Fatah e Sangaiah (2018)

Nas equações, i é o índice da partícula, t é um índice de tempo discreto, $pBest$ é a melhor posição pessoal, $gBest$) é a melhor posição global, α é uma constante positiva para o controle da velocidade da partícula, e por fim ε_1 e ε_2 são dois vetores aleatórios no intervalo [0,1]. O algortimo pode ser observado em Algoritmo 4.

Input: Função objetivo $f(x)$, tamanho do enxame N, número máximo de iterações T, parâmetros de controle α, vetores aleatórios ε_1 e ε_2

Output: Melhor solução global $gBest$

Inicialização:
Gerar aleatoriamente as posições iniciais das partículas $\{x_i^0\}_{i=1}^N$;
Gerar aleatoriamente as velocidades iniciais $\{v_i^0\}_{i=1}^N$;
Definir a melhor posição pessoal $pBest_i = x_i^0$ para cada partícula i;
Avaliar $f(x_i^0)$ e definir a melhor posição global inicial $gBest$ como o melhor entre os $pBest_i$;
Iterações:
for $t = 0$ **to** $T - 1$ **do**
    for $i = 1$ **to** N **do**
        Atualizar a velocidade da partícula i utilizando a Eq. 2.20;
        Atualizar a posição da partícula i utilizando a Eq. 2.21;
        Avaliar o valor da função objetivo $f(x_i^{t+1})$;
        if $f(x_i^{t+1}) < f(pBest_i)$ **then**
            Atualizar a melhor posição pessoal: $pBest_i \leftarrow x_i^{t+1}$;
        end
    end
    Atualizar a melhor posição global $gBest$ como o melhor entre os $pBest_i$;
end

Algorithm 4: Particle Swarm Optimization (PSO)

2.4.2 *Firefly* (FA)

FA, apresentado em Algoritmo 5 é uma metaheurística inspirada no comportamento de vaga-lumes na natureza. De acordo com Yang (2010), duas considerações são fundamentais: a variação da intensidade da luz e a atratividade. Em uma abordagem simplificada, a atratividade de um vaga-lume é determinada pelo seu brilho, que por sua vez, está relacionada à função objetivo. O brilho de um vaga-lume em uma determinada localização é representada por $I(x) = f(x)$. No entanto, a atratividade β varia de acordo com a percepção de outros vaga-lumes, sendo influenciada pela distância r_{ij} entre o vaga-lume i e vaga-lume j . Além disso, a intensidade da luz diminui com a distância da fonte e a luz é absorvida no meio, por isso a atratividade varia de acordo com o grau de absorção.

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha\varepsilon_i^t \tag{2.22}$$

Fonte: Adaptado de Yang (2010)

Input: Função objetivo $f(x)$, número de vaga-lumes n, parâmetro de absorção γ, número máximo de iterações T

Output: Melhor solução encontrada

Inicialização:
Gerar população inicial de vaga-lumes $\{x_i\}_{i=1}^{n}$;
Calcular a intensidade de luz $I_i = f(x_i)$ para cada vaga-lume;
Iterações:
for $t = 1$ **to** T **do**
    for $i = 1$ **to** n **do**
        for $j = 1$ **to** n **do**
            if $I_i < I_j$ **then**
                Mover o vaga-lume i em direção ao vaga-lume j conforme a atratividade;
            end
            Ajustar a atratividade $\beta = \beta_0 e^{-\gamma r_{ij}^2}$ de acordo com a distância r_{ij};
            Atualizar a solução x_i e recalcular a intensidade $I_i = f(x_i)$;
        end
    end
    Ordenar os vaga-lumes e registrar a melhor solução global;
end

Algorithm 5: Algoritmo Firefly (FA)

Fonte: Adaptado de Yang (2010).

2.4.3 *Cuckoo Search* (CS)

CS é baseado no parasitismo de algumas espécies de pássaro cuco, em que as fêmeas depositam seus ovos em ninhos de outras espécies de pássaros. De acordo com Yang (2010), o algortimo se baseia na competição entre ninhos, que representam as soluções candidatas, e na substituição gradual das piores soluções por novas soluções. Durante o processo de busca, os ninhos (representado por x) são atualizados por meio de operações de voo aleatórias e de substituição entre os ninhos. A exploração e a intensificação são equilibradas por meio de um parâmetro de controle, que influencia a taxa de abandono de ninhos e a intensidade da busca local. O pseudocódigo pode ser obsevado em Algoritmo 6.

No CS, a atualização das soluções candidatas (ou ninhos) ocorre por meio de um processo de voo L aleatório, onde os ninhos são ajustados com base em uma fórmula que combina a solução atual com um fator de aleatoriedade (gerado por $Levy(\lambda)$). A equação que descreve esse processo de atualização pode ser representada como:

$$x_i^{t+1} = x_i^t + \alpha \cdot L \cdot Levy(\lambda) \tag{2.23}$$

Fonte: Adaptado de Yang (2010)

Input: Função objetivo $f(x)$, número de ninhos n, taxa de abandono p_a, número máximo de iterações T

Output: Melhor solução encontrada

Inicialização:
Gerar população inicial de n ninhos hospedeiros $\{x_i\}_{i=1}^n$;
Avaliar a aptidão de cada ninho $F_i = f(x_i)$;
Iterações:
for $t = 1$ **to** T **do**
    Gerar um novo cuco por voo de Lévy: x_{new} e calcular sua aptidão F_{new};
    Escolher aleatoriamente um ninho j;
    if $F_{\text{new}} < F_j$ **then**
        Substituir o ninho j pela nova solução x_{new};
    end
    Abandonar uma fração p_a dos piores ninhos e substituí-los por novas soluções geradas aleatoriamente;
    Avaliar novamente a aptidão de todos os ninhos;
    Atualizar a melhor solução global;
end

Algorithm 6: Cuckoo Search (CS)

Fonte: Adaptado de Yang (2010).

2.4.4 *Black Hole (BH)*

O BH, apresentado no Algoritmo 7, é inspirado nos princípios físicos dos buracos negros no universo. Hatamlou (2013) indica que, nesse algoritmo, as soluções candidatas são representadas como estrelas em um espaço de busca, onde um buraco negro atrai essas estrelas em direção a uma região ótima. A intensidade da atração é determinada pela distância entre as estrelas e o buraco negro, assim como pela massa do buraco negro. Após a inicialização, os valores de aptidão da população são avaliados e o melhor candidato é selecionado para ser o buraco negro, enquanto o restante são as estrelas.

O buraco negro absorve as estrelas ao seu redor, movendo-as em direção ao melhor candidato a cada iteração, evitando a estagnação em mínimos locais e explorando eficientemente o espaço de busca em direção a soluções de alta qualidade. Quando estrelas são absorvidas pelo buraco negro, elas são substituídas por novas soluções, mantendo a diversidade da população e continuando a busca por soluções ótimas. A absorção de estrelas pelo buraco negro está

formulada na Equação 2.24, onde x_i(t) e x_i(t + 1) são as posições da i-ésima estrela nas iterações t e t+1,respectivamente. x_{BH} é a posição do buraco negro no espaço de busca. r é um número aleatório no intervalo [0, 1]. N é o número de estrelas (soluções candidatas).

$$x_i(t+1) = x_i(t) + r(x_{BH}(t) - x_i(t))i = 1, 2, ..., N \tag{2.24}$$

Fonte: Hatamlou (2013)

Input: Tamanho da população N, número máximo de iterações T
Output: Melhor solução obtida

Inicialização:
Gerar aleatoriamente uma população inicial de estrelas $\{x_i\}_{i=1}^{N}$;
Avaliar a aptidão de cada estrela $f(x_i)$;
Identificar a estrela com melhor aptidão e defini-la como o buraco negro x_{BH};
Iterações:
for $t = 1$ **to** T **do**
    for $i = 1$ **to** N **do**
        if $x_i \neq x_{\mathrm{BH}}$ **then**
            Mover a estrela x_i em direção ao buraco negro x_{BH};
        end
    end
    for $i = 1$ **to** N **do**
        if x_i *estiver dentro do horizonte de eventos do buraco negro* **then**
            Substituir x_i por uma nova estrela gerada aleatoriamente;
        end
    end
    Avaliar novamente a aptidão de cada estrela;
    Atualizar o buraco negro x_{BH} caso uma estrela com aptidão superior seja encontrada;
end

Algorithm 7: Algoritmo Black Hole (BH)

Fonte: Adaptado de Hatamlou (2013).

2.4.5 *Elephant Herding Optimization (EHO)*

Segundo Wang, Deb e Coelho (2015), o EHO é um algoritmo de otimização inspirado no comportamento dos elefantes na natureza. A estrutura geral do algoritmo é apresentada no Algoritmo 8 e 9. Ela modela dois comportamentos distintos: viver juntos em clãs sob a liderança de uma matriarca e a separação dos elefantes machos quando crescem. Esses comportamentos são traduzidos em dois operadores principais: o operador de atualização de movimentação no

clã e o operador de separação. No EHO, os elefantes em cada clã são atualizados com base em sua posição atual e na influência da matriarca por meio do operador de atualização do clã. O operador de separação visa aumentar a diversidade da população em fases posteriores da busca.

A fórmula de movimentação de um elefante j no clã ci que tem por matricarca $x_{best,ci}$ está descrita na Equação 2.25. Nessa equação, α representa o fator de influência da matriarca. Já a operação de separação é a Equação 2.26, onde x_{min} e x_{max} são os limites superior e inferior da posição do pior elefante do clã ci.

$$x_{new,ci,j} = x_{ci,j} + \alpha(x_{best,ci} - x_{ci,j})r \tag{2.25}$$

Fonte: Wang, Deb e Coelho (2015)

$$x_{worst,ci} = x_{min} + (x_{max} - x_{min})rand \tag{2.26}$$

Fonte: Wang, Deb e Coelho (2015)

Input: Número de clãs n_{Clan}, número de elefantes por clã n_{ci}
Output: Novas posições dos elefantes após movimentação

```
for ci = 1 to n_Clan do
    for j = 1 to n_ci do
        Atualizar a posição do elefante x_{ci,j} e gerar a nova posição x^{new}_{ci,j};
        if x^{new}_{ci,j} = x_{ci,j} then
            Recalcular a movimentação e atualizar x^{new}_{ci,j};
        end
    end
end
```

Algorithm 8: Elephant Herding Optimization (EHO) — Operação de Movimentação no Clã

Input: Número de clãs n_{Clan}
Output: Clãs atualizados após substituição dos piores elefantes

```
for ci = 1 to n_Clan do
    Identificar o pior elefante do clã ci;
    Substituir esse elefante por uma nova posição gerada aleatoriamente;
end
```

Algorithm 9: Elephant Herding Optimization (EHO) — Operação de Separação do Clã

Fonte: Adaptado de Wang, Deb e Coelho (2015).

2.4.6 Bees Algorithm (BA)

Proposto por Pham et al. (2006), o BA baseia-se no processo natural de forrageamento das abelhas, combinando estratégias de exploração global e busca local intensiva. O algoritmo

simula como abelhas procuram e exploram fontes de alimento, concentrando esforços nas regiões mais promissoras do espaço de busca.

No BA, as soluções são representadas por sítios de busca, enquanto as abelhas atuam como agentes que exploram esses sítios, avaliando sua qualidade e gerando variações locais. Após a avaliação da aptidão dos sítios, os melhores são classificados como elite e recebem um número maior de abelhas para exploração local, enquanto os sítios promissores recebem um número menor. As abelhas remanescentes, chamadas *scouts*, realizam uma busca global aleatória pelo espaço de soluções, descobrindo novas regiões de potencial.

A geração de novas soluções em torno de um sítio s_i é realizada pela Equação 2.27, onde ngh define o raio de vizinhança e $\boldsymbol{\phi}$ é um vetor de perturbação aleatória uniforme. Para refinar a busca ao longo das iterações, o tamanho da vizinhança pode ser reduzido segundo a Equação 2.28.

$$x_{\text{new}} = s_i + \boldsymbol{\phi} \cdot \text{ngh}, \qquad \boldsymbol{\phi} \sim \mathcal{U}(-1, 1) \tag{2.27}$$

$$\text{ngh}^{(t+1)} = \rho \cdot \text{ngh}^{(t)}, \qquad 0 < \rho < 1 \tag{2.28}$$

As abelhas *scouts* executam a exploração global, realocando sítios aleatoriamente dentro dos limites do espaço de busca, conforme a Equação 2.29.

$$s_i = \mathbf{x}_{\min} + (\mathbf{x}_{\max} - \mathbf{x}_{\min}) \cdot \mathbf{u}, \qquad \mathbf{u} \sim \mathcal{U}(0, 1) \tag{2.29}$$

O ciclo de operação do BA é apresentado no Algoritmo 10.

Input: Limites do espaço de busca $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$, número total de sítios M, número de sítios de elite m_e, número de sítios promissores m_b, número de vizinhas para sítios de elite n_{re}, número de vizinhas para sítios promissores n_{rb}, parâmetro de vizinhança inicial ngh, número máximo de iterações T

Output: Melhor solução $\mathbf{x}^*$

Inicialização:
Gerar M sítios iniciais s_i aleatoriamente em $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$;
Avaliar a aptidão de cada sítio e identificar a melhor solução inicial $\mathbf{x}^*$;
Iterações:
for $t = 1$ **to** T **do**
    Ordenar os sítios pela aptidão e selecionar m_e sítios de elite e m_b sítios promissores;
    for *cada sítio de elite* s_i **do**
        Gerar n_{re} soluções locais na vizinhança de s_i conforme Eq. 2.27 e manter a melhor;
    end
    for *cada sítio promissor* s_i **do**
        Gerar n_{rb} soluções locais na vizinhança de s_i e manter a melhor;
    end
    for *cada sítio restante* **do**
        Realocar o sítio por exploração global conforme Eq. 2.29;
    end
    Atualizar $\mathbf{x}^*$ com a melhor solução corrente;
    Atualizar ngh conforme Eq. 2.28;
end

Algorithm 10: Bees Algorithm (BA)

Fonte: Adaptado de Pham et al. (2006).

Em síntese, o BA alterna entre intensificação, por meio da busca local conduzida pelas abelhas recrutadas nas regiões mais promissoras, e diversificação, realizada pelas abelhas *scout* que executam a exploração global. Essa dinâmica assegura um equilíbrio adaptativo entre exploração e intensificação do espaço de busca, servindo de base para as adaptações implementadas nesta pesquisa no contexto de agrupamento multivisual.

3
Revisão Sistemática

Neste capítulo, serão descritas as etapas de desenvolvimento da Revisão Sistemática, o objetivo e as questões de Pesquisa, a definição das strings de busca, seleção das bases de dados, os critérios de inclusão, exclusão e extração de dados. Por fim, serão apresentados os resultados da revisão, a partir da discussão das respostas das questões de pesquisa.

3.1 Objetivo e Questões de Pequisa

Levando em conta o crescimento do campo de estudo, a revisão sistemática tem por objetivo explorar bases relevantes, mapear e analisar estudos associados ao agrupamento de dados de múltiplas visões, acompanhando o avanço dessa área e contextualizando as áreas de aplicação. Para isso, foram elaboradas tais questões de pesquisa:

1. Quais são as principais abordagens utilizadas para lidar com o problema do agrupamento de dados com múltiplas visões??
2. Quais as principais áreas de aplicação ao se pesquisar sobre agrupamento de dados com múltiplas visões?
3. O que se pode afirmar sobre experimentos computacionais?
4. Quais os índices internos mais utilizados?
5. Quais os índices externos mais utilizados?

3.1.1 Estratégia de Busca

Inicialmente são definidas palavras-chaves para a construção da *string* de busca, que pode vir a ser adaptada de acordo com a base. Na Tabela 3.1.1 estão apresentadas as palavras,

enquanto na Tabela 3.1.1 a *string* de busca. Foram selecionadas quatro base eletrônicas: ACM, IEEE Xplore, Science Direct e Scopus. A seleção dos estudos foi realizada mediante os critérios de inclusão e exclusão estabelecidos. Para que um artigo fosse selecionado, deveria cumprir todos os critérios de inclusão. Um ponto a se destacar, é que a *string* de busca foi aplicada apenas nos títulos das publicações.

Os Critérios de Inclusão foram:

1. Trabalhos publicados entre os anos de 2012 e 2023;
2. Trabalhos que utilizam dados de múltiplas visões;
3. Trabalhos relacionados a tarefa de agrupamento;

Os Critérios de Exclusão foram:

1. Trabalhos em que não foi possível ter acesso ao texto completo;
2. Trabalho que são Revisão Sistemática ou Mapeamento Sistemático;
3. Trabalhos que não utilizaram técnicas de agrupamento de dados;
4. Trabalhos publicados com data inferior a 2013;

Tabela 1 – Palavras-Chave utilizadas na *string* de busca

Palavra chave	Sinônimo em Inglês
agrupamento	clustering
múltiplas visões	multi view, multi-view e multiview

Tabela 2 – *String* utilizada para realizar as buscas nas bases

(clustering) AND ((("multi view") OR (multi-view) OR (multiview))

3.1.2 Extração dos Dados

Para encontrar as respostas das questões de pesquisa, é necessário a extração de algumas informações dos artigos selecionados. Por isso, os dados escolhidos para extração são: país(es) das instituições dos pesquisadores, abordagem difusa ou rígida, método base, utilização de visão incompleta, representação dos dados, representativo dos grupos, função monoobjetiva ou multiobjetiva, estratégia de agrupamento, quantidade de base de dados, repetições dos testes, índice externo, índice interno, utilização de abordagem semi supervisionada.

3.1.3 Procedimento de Seleção de Estudos

O procedimento de seleção dos estudos foi dividido em três etapas. Na etapa inicial, compreendeu a aplicação da *string* de busca nas bases eletrônicas estabelecidas. A *string* foi adaptada de acordo com as particularidades de cada base, sendo do empregada exclusivamente nos títulos dos artigos científicos publicados entre os anos 2012 e 2023.

Na segunda etapa, foi realizada a retirada de estudos duplicados, leitura dos títulos e resumos dos artigos, além da aplicação dos critérios de inclusão e exclusão. Os estudos que apresentavam todos os critérios de inclusão foram pré-selecionados para a última fase. Por fim, ocorreu a leitura integral dos artigos, a extração dos dados e a resolução das questões de pesquisa. É relevante destacar que durante a fase 3 do estudo, cada artigo foi submetido à revisão por pelo menos dois avaliadores, visando assegurar a confiabilidade do processo de extração de dados. Na Tabela 3 ilustra a quantidade de artigos resultantes de cada etapa do processo de extração dos artigos desde a base até a análise.

Tabela 3 – Níveis de investigação.

Base	**Etapa 1**	**Etapa 2**	**Etapa 3**
IEEE	525	309	309
ACM Digital Library	213	54	54
Science Direct	175	145	145
Scopus	90	50	50
Total	1003	558	558

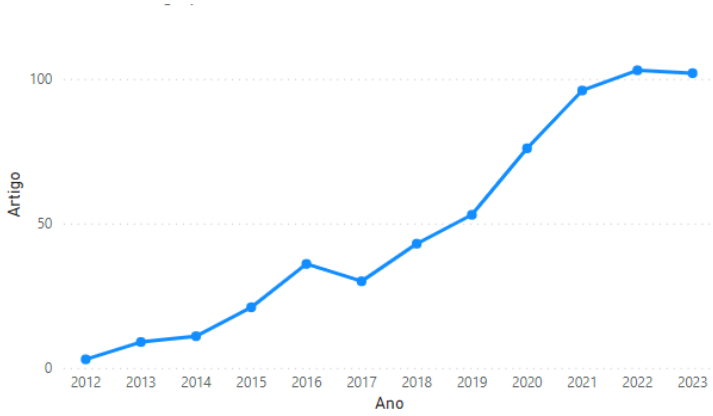
3.2 Resultados e Discussão

Nesta seção, estão apresentados os resultados obtidos na revisão sistemática, bem como as respostas as questões de pesquisa estabelecidas. Foram analisados cerca de 558 publicações e pode-se perceber que ao decorrer dos anos, houve uma crescente na quantidade de estudos voltados a agrupamento de múltiplas visões, conforme apresentado na figura 3.

O aumento de estudos sobre agrupamento de dados com múltiplas visões ao longo dos anos pode ser explicado por vários fatores. Primeiramente, conforme afirmado por Fu et al. (2020), os avanços tecnológicos têm proporcionado acesso a uma quantidade crescente de dados em diferentes formatos e modalidades. Além disso, o desenvolvimento de computadores mais robustos tem viabilizado a aplicação de técnicas cada vez mais complexas.

De maneira geral, pode-se perceber uma predominância da utilização da abordagem espectral, algoritmos não supervisionados, estratégia centralizada, além da a utilização de funções monoobjetivas e bases de dados com visões completas. A seguir estão detalhadas as questões de pesquisa.

Figura 3 – Artigos publicados por ano



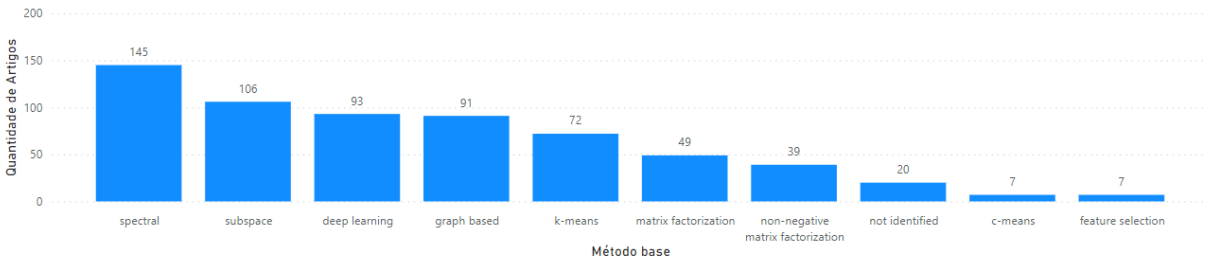
Fonte: O autor

3.2.1 Quais são as principais abordagens utilizadas para lidar com o problema do agrupamento de dados com múltiplas visões??

Na Figura 4 pode-se observar as principais abordagens utilizadas. A abordagem espectral foi a mais frequente, seguida pelas abordagens baseadas em grafos e subespaço. Além disso, a utilização de algoritmos de aprendizagem profunda foi prevalente. Embora não seja tradicionalmente classificada como uma abordagem clássica de agrupamento de múltiplas visões, a aprendizagem profunda oferece ferramentas e arquiteturas flexíveis que podem ser adaptadas à complexidade dos dados provenientes de diversas fontes.

Segundo Ye et al. (2018) o agrupamento espectral está intimamente ligado ao problema do corte mínimo em grafos. Inicialmente, realiza a redução de dimensionalidade no espaço de dados original usando o espectro da matriz de similaridade, seguido pelo agrupamento k-means para formar clusters distintos. Já Elhamifar e Vidal (2013) indica que o agrupamento de subespaços refere-se ao problema de separar dados de acordo com seus subespaços subjacentes. O agrupamento baseado em grafos destaca a representação e análise dos dados por meio de um grafo, que consiste em nós (ou vértices) e arestas que conectam esses nós.

Figura 4 – Abordagens utilizados nos artigos



Fonte: O autor

No contexto deste estudo, foi importante distinguir o agrupamento espectral de outros

métodos baseados em grafos, devido à significativa quantidade de artigos que adotam essa abordagem. Embora façam uso de conceitos da teoria espectral de grafos, esses métodos não necessariamente empregam a estrutura de grafos como parte fundamental do processo de agrupamento. Assim, aproximadamente 91 publicações utilizaram o agrupamento baseado em grafos como método principal.

Conforme destacado por Mingqiang, Hui e Qian (2012), os algoritmos de agrupamento baseado em grafos associam os conjuntos de dados a nós, que por sua vez, são tratados como vértices de um grafo, onde as arestas representam as distâncias entre esses nós. A abordagem tem sido utilizada ao longo de décadas para tarefas de agrupamento, apresentando a flexibilidade para incorporar diversos métodos da teoria de grafos.

A aplicação do algortimo k-means para contexto de múltiplas visões foi encontrado em 72 publicações. O algoritmo encontra os clusters particionando um conjunto de dados de modo a minimizar o erro quadrático. O algoritmo tradicional do K-means é voltado para apenas uma visão, mas conforme Jain (2010), foi estendido de várias maneiras diferentes, inclusive para dados de múltiplas visões.

Por fim, é importante ressaltar a presença representativa de metodologias de fatorização de matrizes e fatoração matricial não negativa, apesar de não está diretamente associada a uma única abordagem, pois são metodologias que podem ser utilizadas em contextos com diferentes técnicas.

3.2.2 Quais as principais áreas de aplicação ao se pesquisar sobre agrupamento de dados com múltiplas visões?

É possível perceber uma predominância, cerca de 495 publicações, de implementações com adaptações de abordagens tradicionais amplamente utilizadas, além da utilização do uso de bases de dados referência, ou seja, sem um contexto específico. Uma pequena parcela dos estudos aborda um domínio específico como sáude e bioinformática.

De acordo com Chao, Sun e Bi (2021), embora o MVC tenha sido aplicado em diversos domínios científicos, como visão computacional, processamento de linguagem natural, multimídia social, bioinformática e informática em saúde, ainda persistem alguns problemas em aberto que restringem seu progresso. Problemas como visões de dados incompletas e valores ausentes, além de problemas relacionados a inicialização e mínimos locais são exemplos.

Os resultados indicam que, embora o agrupamento de dados com múltiplas visões apresente um grande potencial de aplicação em diversas áreas, ainda há uma escassez de estudos em domínios específicos. Este fato ressalta a necessidade e as oportunidades para novas investigações por parte dos pesquisadores interessados neste campo.

3.2.3 O que se pode afirmar sobre experimentos computacionais?

O primeiro ponto a se destacar é em relação as categorias de algoritmos: rígido ou difuso. Cerca de 512 estudos aplicaram algoritmos classificados como rígido. Segundo Das, Abraham e Konar (2009), métodos rígidos atribuem cada objeto a apenas um cluster, em contrapartida, os métodos nebulosos atribuem a cada objeto um grau de pertencimento em cada cluster.

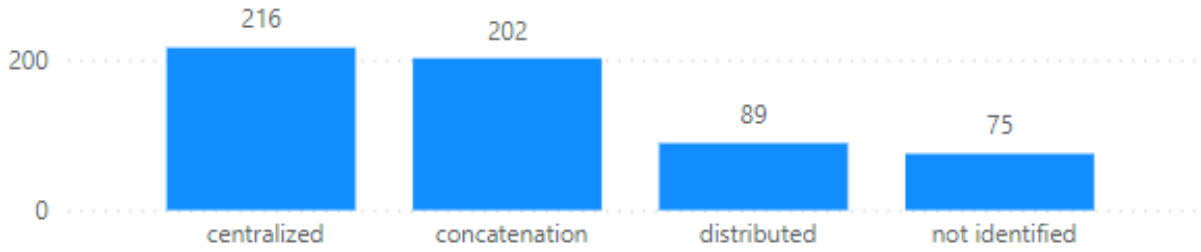
No que diz respeito ao tipo de aprendizagem dos algoritmos, no contexto de agrupamento, são a abordagem semisupervisionada e não supervisionada. Para Han, Pei e Kamber (2011) a aprendizagem semisupervisionada é uma classe que faz uso tanto de amostras rotulados e não rotulados durante o processo de aprendizado de um modelo, enquanto na não supervisionada utiliza dados não rotulados. Por ser a essência do agrupamento, abordagens não supervisonada foi utilizada em 541 publicações.

Em algoritmos de particionamento de dados, as funções objetivo são utilizadas para quantificar a qualidade dos agrupamentos formados. A escolha da função depende do contexto específico do problema, da natureza dos dados e do objetivo desejado para o agrupamento. Podem ser categorizadas como monoobjetivas ou multiobjetivas, dependendo do número de objetivos que buscam otimizar. Na revisão, cerca de 574 artigos utilizaram funções monoobjetivas.

As estratégias para agrupamento de dados com múltiplas visões podem ser divididas em três grupos: concatenação, distribuída ou centralizada. Segundo Cleuziou et al. (2009) as abordagens se distinguem pela etapa em que a integração das visões é realizada, seja ela antes, durante ou após o agrupamento. Entre os artigos analisados foi observado uma predominância da abordagem centralizada, conforme indica a Figura 5.

Segundo Shao et al. (2016), devido à natureza dos dados ou ao processo de coleta, é possível que algumas visualizações apresentem ausência de instâncias, o que requer a implementação de métodos de particionamento capazes de lidar com essa incompletude de visões. Aproximadamente 98 artigos utilizaram algoritmos de particionamento de dados adaptados para lidar com essa característica das visões. Este resultado sugere a existência de oportunidades para novos estudos focados no agrupamento de visões incompletas.

Figura 5 – Estratégias utilizadas nos artigos



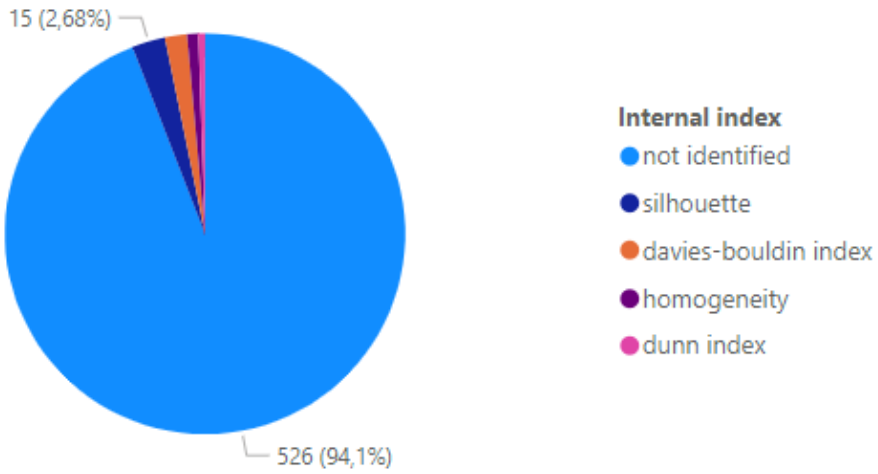
Fonte: O autor

3.2.4 Quais os índices internos mais utilizados?

Os índices internos referem-se as métricas para avaliação de resultados de algoritmos de agrupamento. São chamados de internos porque se baseiam apenas em informações dentro do conjunto de dados e não necessitam de informações adicionais. Os índices internos foram encontrados com menos frequência, se comparado aos externos. Os índices internos mais encontrados são: coeficiente de silhueta, índice Davies Bouldin (DB) e Homogeneidade, conforme indica a Figura 6.

Shahapure e Nicholas (2020) afirma que o coeficiente de silhueta é determinado mediante a consideração das distâncias médias intra-cluster e inter-cluster para cada ponto de dados, portanto, uma pontuação de silhueta com um valor próximo de 1 significa que o ponto de dados está no cluster correto. Já segundo Das, Abraham e Konar (2009), a métrica Davies-Bouldin (DB), que é a razão entre a soma da dispersão intra-cluster e a separação inter-cluster, revela que a menor pontuação desse índice indica uma partição ótima. Por fim, Homogeneidade indica o grau de semelhança dos registros que pertencentes a um cluster.

Figura 6 – Índices internos encontrados



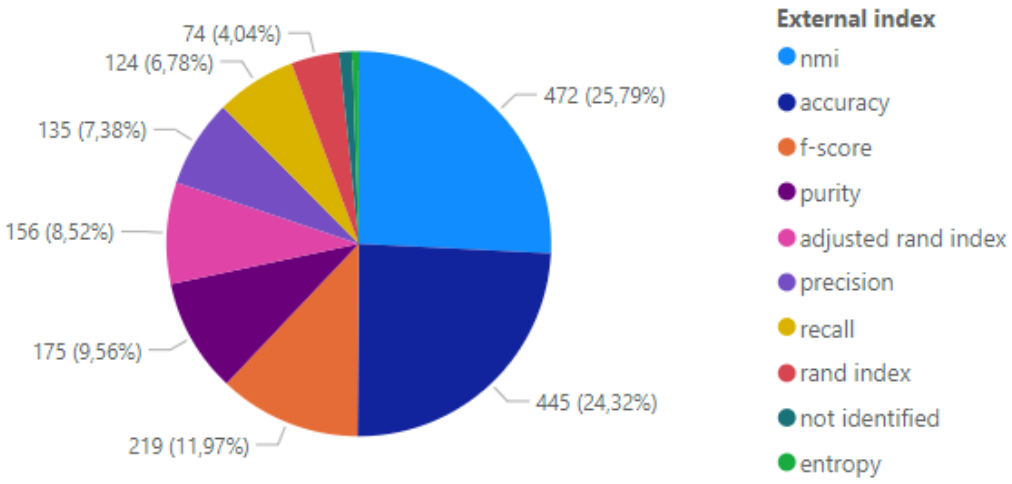
Fonte: O autor

3.2.5 Quais os índices externos mais utilizados?

Ao contrário dos índices internos, que se baseiam apenas nas informações disponíveis nos dados, os índices externos comparam os resultados do agrupamento com rótulos já conhecidos. Essa categoria de índice foi encontrada com maior frequência nos estudos analisados. O resultado pode ser explicado devido à capacidade dos índices externos de proporcionar uma validação mais precisa e relevante para aplicações práticas. Na Figura 7 estão apresentados índices externos, sendo os mais encontrados: informação mútua normalizada (NMI), acurácia, e medida f.

Conforme explicado no capítulo 2, a NMI varia de 0 a 1, onde um valor mais próximo de 1 indica uma correspondência mais precisa entre os agrupamentos e as classes verdadeiras. Ela mensura a similaridade entre dois agrupamentos ao normalizar a pontuação de Informação Mútua (MI), levando em consideração a entropia das classes verdadeiras e dos agrupamentos preditos. Han, Pei e Kamber (2011) indica que a acurácia é porcentagem de acerto dos agrupamentos, no que diz respeito aos dados de cada cluster. A medida f varia de 0 a 1 e combina precisão e sensibilidade em uma única métrica, proporcionando uma visão equilibrada do desempenho do modelo, em que 1 indica a melhor concordância entre as partições.

Figura 7 – Índices externos encontrados



Fonte: O autor

3.3 Considerações Finais da Revisão Sistemática

Após a execução das etapas necessárias, aproximadamente 558 artigos foram revisados e avaliados para atender às questões de pesquisa propostas. Observaram-se tendências nas estratégias e abordagens adotadas, incluindo o uso predominante de estratégias centralizadas para o agrupamento, funções de otimização monoobjetivas e uma prevalência de índices de validação internos em comparação aos externos. Além disso, constatou-se uma predominância no uso de bases de dados com visões completas.

Para trabalhos futuros, incluem-se experimentos, como análises comparativas para verificar o desempenho dos algoritmos de um determinado método base, como agrupamento espectral e aprendizagem profunda. Além disso, investigações sobre o impacto de funções objetivas no agrupamento e outras aplicações podem ser exploradas. Tais direções podem contribuir significativamente para avançar nosso entendimento e aplicação de técnicas de agrupamento de dados de múltiplas visões em diversos contextos.

Por fim, os resultados da revisão sistemática foram utilizados para nortear a escolha dos algoritmos aplicados nos experimentos, bem como a escolha dos índices de validação e conjunto de dados utilizados.

4
Metodologia

Nesse capítulo está apresentada a metodologia que será aplicada no desenvolvimento deste estudo. O objetivo principal é analisar o impacto da aplicação de metaheurísticas inspiradas na natureza para o agrupamento de dados com múltiplas visões. Para alcançar este objetivo, uma série de experimentos comparativos foi conduzida a partir de duas funções objetivo distintas: a formulação original do TW-K-Means e a VWOF. Ambas as funções foram integradas ao framework das metaheurísticas detalhadas no Capítulo 2 para fins de comparação de desempenho, conforme será detalhado na seção 5.

As implementações foram desenvolvidas em Python 3.10, utilizando a biblioteca CuPy para operações paralelas baseadas em GPU. Todos os experimentos foram executados no ambiente Google Colab Pro+, com acesso a uma GPU NVIDIA Tesla T4 e 32 GB de memória RAM, garantindo padronização e eficiência computacional.

4.1 Adaptações das Metaheurísticas para múltiplas visões

As metaheurísticas consideradas neste estudo foram adaptadas para o problema de agrupamento de dados em múltiplas visões. Em todos os casos, cada indivíduo da população foi modelado como uma solução candidata composta pelos centróides de todos os *clusters* em todas as visões. Essas soluções são armazenadas em vetores achatados (*flattened arrays*), o que possibilita a execução uniforme das operações de otimização, independentemente do número de visões ou atributos de cada base de dados. O cálculo da aptidão de cada solução é determinado pelo valor da função objetivo, que combina medidas de compacidade intra-*cluster* com os pesos aprendidos dinamicamente.

Em todas as metaheurísticas, a avaliação da qualidade das soluções é realizada por meio da mesma função-objetivo, podendo operar em dois modos de execução: completo (TW-K-Means) e simplificado (VWOF). No modo completo, as distâncias são ponderadas por pesos de

visão e de atributo, atualizados a cada iteração por normalização *softmax* com piso mínimo e regularizados por termos de entropia. Já no modo simplificado, apenas os pesos de visão são mantidos, sendo ajustados dentro da própria função-objetivo com base no erro relativo de cada visão e normalizados em seguida. Essa formulação unificada permite que as metaheurísticas operem sobre o mesmo critério de otimização, diferenciando-se apenas pela estratégia de busca e atualização das soluções.

Na tentativa de alavancar o desempenho computacional e a escalabilidade, todos os algoritmos também foram implementados com paralelização em GPU. As operações de cálculo de distância, atualização de centróides, ajuste de pesos e avaliação da função objetivo foram vetorizadas e executadas em memória de dispositivo, explorando o modelo de paralelismo massivo do CUDA.

4.1.1 PSO

A adaptação do PSO, ilustrada no algoritmo 11, manteve o comportamento colaborativo do enxame, mas redefiniu a estrutura das partículas e os operadores de atualização para operar sobre o espaço vetorial que representa os centróides de todas as visões. Cada partícula foi modelada como uma solução candidata contendo, as coordenadas dos centróides de todos os *clusters* nas diferentes visões.

Nesta implementação, a posição e a velocidade da partícula controlam apenas os *centróides*; os *pesos* (por visão e, no modo completo, por atributo) são variáveis internas atualizadas a cada iteração por rotinas específicas, não fazendo parte do vetor de posição/velocidade. Essa representação unificada permite que o deslocamento das partículas considere simultaneamente a contribuição de cada visão na formação dos agrupamentos, refletindo a natureza integrada do problema multivisual.

As equações de atualização de velocidade e posição foram ajustadas para o novo formato vetorial. A velocidade de cada partícula representa a direção e a intensidade de deslocamento dos centróides multivisuais, sendo atualizada com base em três componentes: a inércia do movimento anterior, a atração em direção à melhor solução individual (componente cognitiva) e a atração para a melhor solução global (componente social). Emprega-se um agendamento decrescente do peso de inércia w ao longo das iterações, enquanto o *clamp* de velocidade não foi aplicado nesta implementação (podendo ser adotado como melhoria futura). Os estados $pBest$ e $gBest$ armazenam apenas os centróides das melhores soluções encontradas, enquanto os pesos são recalculados a cada avaliação. O processo é encerrado por critério de ausência de melhora do valor objetivo por um número pré-definido de iterações ou pelo limite máximo estabelecido.

Na implementação paralelizada para GPU, os vetores de posição, velocidade, pesos e valores de aptidão são mantidos integralmente em memória de dispositivo, evitando transferências desnecessárias entre CPU e GPU. As operações de cálculo de distâncias, atualização de centróides,

ajuste de pesos e avaliação da função-objetivo foram vetorizadas com a biblioteca CuPy, explorando o modelo de paralelismo massivo do CUDA. Fluxos assíncronos (*non-blocking streams*) são empregados para atualizar diferentes partículas em paralelo, eliminando laços explícitos e reduzindo significativamente o tempo de iteração.

Input: Dados X com V visões, número de clusters k, número máximo de iterações T, coeficientes c_1 e c_2, limites da inércia $w_{\min}$ e $w_{\max}$

Output: Melhor partícula encontrada e valor objetivo associado

Inicialização:
Inicializar n partículas com posições representando centróides multivisuais e velocidades correspondentes;
Inicializar pesos internos das visões (e pesos de atributos, no caso TW-K) por *softmax* com piso mínimo;
Avaliar cada partícula, definindo $pBest$ individuais e o $gBest$ global inicial;
Iterações:
for $t = 1$ **to** T **do**
    Atualizar peso de inércia:

$$w \leftarrow w_{\max} - \frac{(w_{\max} - w_{\min})\, t}{T}$$

    for *cada partícula p* **do**
        Atualizar velocidade:

$$v \leftarrow wv + c_1 r_1 (pBest - x) + c_2 r_2 (gBest - x)$$

        Atualizar posição:

$$x \leftarrow x + v$$

        Reatribuir objetos aos clusters e recalcular centróides por visão;
        Atualizar pesos internos das visões (e atributos, se aplicável) via *softmax* com piso mínimo;
        Avaliar a partícula; atualizar $pBest$ e $gBest$ se houver melhoria;
    end
    if *convergência detectada (ausência de melhoria por p iterações ou* $|f^t - f^{t-1}| < \varepsilon$*)* **then**
        break
    end
end
return $gBest$ *e a solução correspondente*;

Algorithm 11: PSO Adaptado para Agrupamento Multivisual

4.1.2 BH

A adaptação, apresentada no algoritmo 12, manteve o princípio de atração gravitacional em torno da melhor solução global, mas reformulou a representação das estrelas e o cálculo do horizonte de eventos. Cada estrela representa uma solução candidata que codifica todos os centróides dos *clusters* em todas as visões, permitindo que o movimento em direção ao buraco negro considere simultaneamente as contribuições de cada representação. Nesta implementação, a posição da estrela controla apenas os centróides, enquanto os pesos de visão e de atributo são variáveis internas atualizadas a cada iteração, de acordo com a função-objetivo utilizada.

A dinâmica de atração foi reescrita para operar diretamente sobre o vetor unificado de centróides. Em cada iteração, as estrelas são atraídas em direção ao buraco negro com uma taxa proporcional à distância vetorial no espaço multivisual. Quando uma estrela se aproxima demais — ou seja, quando sua distância cai abaixo do horizonte de eventos — ela é realocada aleatoriamente em uma nova posição do espaço de busca, garantindo diversidade populacional

e evitando convergência prematura. O horizonte de eventos é recalculado dinamicamente a cada iteração, sendo definido pela razão entre o valor da melhor função-objetivo e a soma das aptidões de todas as estrelas. Essa adaptação garante que o mecanismo gravitacional se ajuste automaticamente à dispersão e à qualidade média das soluções, promovendo um equilíbrio entre exploração e intensificação.

Na implementação paralelizada, todas as estrelas e centróides são mantidos em memória de dispositivo, e as operações de cálculo de distâncias, atualização de posições e avaliação da função-objetivo são realizadas por meio de operações matriciais vetorizadas em CuPy. O uso de fluxos assíncronos (*non-blocking CUDA streams*) permite que múltiplas estrelas sejam atualizadas em paralelo, eliminando laços explícitos e reduzindo a sobrecarga por iteração.

Input: Dados X com V visões, número de clusters k, número máximo de iterações T, modo $mode \in \{\text{VWOF}, \text{TWK}\}$, parâmetros λ e η
Output: Melhor estrela (black hole) e valor objetivo associado

Inicialização:
Gerar n estrelas com centróides multivisuais iniciais e pesos internos das visões $\mathbf{w}$ (e pesos de atributos $\boldsymbol{\alpha}$ no modo TWK);
Normalizar todos os pesos via *softmax* com piso mínimo;
for *cada estrela* **do**
    Atribuir objetos aos clusters utilizando distâncias ponderadas conforme o modo selecionado;
    Recalcular centróides por visão;
    Avaliar a função objetivo e atualizar a estrela black hole se necessário;
end
Calcular o horizonte de eventos:

$$R = \frac{f(\text{BH})}{\sum_{i=1}^{n} f(\text{estrela}_i)}$$

Iterações:
for $t = 1$ **to** T **do**
    for *cada estrela s tal que* $s \neq \text{BH}$ **do**
        Mover s em direção ao black hole no espaço de centróides, proporcionalmente à distância;
        Reatribuir objetos aos clusters e recalcular os centróides;
        Atualizar pesos internos $\mathbf{w}$ (e $\boldsymbol{\alpha}$, no modo TWK) por normalização *softmax*;
        Avaliar a estrela; atualizar o black hole se for encontrada uma solução melhor;
        if $\text{dist}(s, \text{BH}) < R$ **then**
            Reinicializar s aleatoriamente respeitando os limites de cada visão;
        end
    end
    Atualizar o horizonte de eventos R;
    Verificar critério de parada (ausência de melhoria ou $|f^t - f^{t-1}| < \varepsilon$);
end
return *black hole e solução correspondente*;

Algorithm 12: Black Hole Adaptado para Agrupamento Multivisual

4.1.3 BA

Diferentemente da formulação original, na qual os sítios representam as soluções candidatas e as abelhas apenas exploram suas vizinhanças, a adaptação redefiniu a estrutura das soluções e o mecanismo de exploração. Nesta adaptação, ilustrada no algoritmo 13, cada abelha corresponde diretamente a uma solução independente, armazenando em um único vetor os

centróides correspondentes aos *clusters* multivisuais. Nesta implementação, a posição da abelha controla apenas os centróides, enquanto os pesos de visão e de atributo são variáveis internas atualizadas a cada iteração, conforme a função-objetivo utilizada. Essa modificação elimina a necessidade de etapas de recrutamento e vizinhança, permitindo que cada indivíduo mantenha e atualize seus próprios centróides e pesos de forma autônoma.

O processo de busca foi ajustado para combinar exploração global e intensificação local de forma probabilística. Em cada iteração, as abelhas atualizam suas posições de duas maneiras: com 50% de chance realizam uma perturbação aleatória dos centróides e, com 50%, deslocam-se em direção à melhor solução conhecida (*best bee*). Essa estratégia promove o equilíbrio entre diversidade populacional e convergência, permitindo explorar regiões promissoras do espaço de soluções sem perder variabilidade.

Na implementação paralelizada, todas as abelhas e centróides são mantidos em memória de dispositivo, e as operações de cálculo de distâncias, atualização de centróides e avaliação da função-objetivo são totalmente vetorizadas em CuPy. O uso de *streams* assíncronos (*non-blocking*) permite atualização simultânea de todas as abelhas, eliminando laços explícitos e aproveitando melhor a capacidade de processamento da GPU.

Input: Dados X com V visões, número de clusters k, número máximo de iterações T, modo $mode \in \{\text{VWOF}, \text{TWK}\}$, probabilidade de intensificação ρ, escala inicial de ruído σ_0
Output: Melhor abelha encontrada e valor objetivo correspondente

Inicialização:
Gerar n abelhas com centróides multivisuais aleatórios e pesos internos das visões $\mathbf{w}$ (e dos atributos $\boldsymbol{\alpha}$, no modo TWK);
Normalizar pesos via *softmax* com piso mínimo;
for *cada abelha* **do**
    Atribuir objetos aos clusters conforme o modo selecionado;
    Recalcular centróides por visão;
    Avaliar a função objetivo da abelha;
    Atualizar a melhor solução global (*best bee*) se necessário;
end
Iterações:
for $t = 1$ **to** T **do**
    Ajustar a escala de ruído σ_t e o fator de intensificação γ_t (quando aplicável);
    for *cada abelha* b **do**
        if rand() $< \rho$ **then**
            Atualizar a posição da abelha por intensificação:

$$b \leftarrow b + \gamma_t \cdot (\text{best} - b)$$

        else
            Atualizar a posição da abelha por exploração aleatória:

$$b \leftarrow b + \mathcal{N}(0, \sigma_t^2)$$

        end
        Reatribuir objetos aos clusters e recalcular centróides por visão;
        Atualizar pesos internos $\mathbf{w}$ (e $\boldsymbol{\alpha}$, no modo TWK) via normalização *softmax*;
        Avaliar a abelha b e atualizar a melhor solução global se for encontrada melhoria;
    end
    Verificar critério de parada (ausência de melhoria ou $|f^t - f^{t-1}| < \varepsilon$);
end
return best *e solução correspondente*;

Algorithm 13: Bees Algorithm (BA) Adaptado para Agrupamento Multivisual

4.1.4 EHO

No contexto de agrupamento multivisual, o EHO foi adaptado para representar cada elefante como uma solução candidata composta pelos centróides de todos os *clusters* em todas as visões. A estrutura hierárquica original foi mantida, com a divisão da população em clãs e a designação de uma matriarca para cada grupo. Nesta implementação, a posição de cada elefante controla apenas os centróides, enquanto os pesos de visão e de atributo são variáveis internas atualizadas a cada iteração conforme a função-objetivo utilizada.

Os operadores originais foram ajustados para atuar diretamente sobre os vetores de centróides multivisuais, preservando a coerência entre visões. O movimento de cada elefante é calculado em relação à matriarca e ao centro médio de seu clã, ponderado por um parâmetro de influência que controla a intensidade de atração. O operador de separação é aplicado de forma probabilística, reinicializando parcialmente o pior elefante de cada clã para garantir diversidade

populacional e evitar estagnação.

A implementação paralelizada do EHO foi desenvolvida para execução em GPU, com cada clã processado de forma independente em blocos de threads. As etapas de cálculo de distâncias, atualização de centróides e avaliação da função-objetivo são vetorizadas e executadas em memória de dispositivo, utilizando operações matriciais em lote e difusão de dados para eliminar laços explícitos. Essa abordagem permite a atualização simultânea de todos os elefantes, garantindo escalabilidade e redução significativa do tempo de execução em bases multivisuais de alta dimensionalidade.

Input: Dados X com V visões, número de clusters k, número máximo de iterações T, modo $mode \in \{\text{VWOF}, \text{TWK}\}$, número de clãs C, taxa de atualização r_c, taxa de separação r_s, parâmetro de influência da matriarca μ

Output: Melhor elefante encontrado e valor objetivo correspondente

Inicialização:
Gerar n elefantes com centróides multivisuais aleatórios e pesos internos das visões $\mathbf{w}$ (e dos atributos $\boldsymbol{\alpha}$, no modo TWK);
Normalizar pesos via *softmax* com piso mínimo;
Particionar a população em C clãs aproximadamente balanceados;
for *cada elefante e* **do**
    Atribuir objetos aos clusters conforme o modo selecionado;
    Recalcular centróides por visão;
    Avaliar a função objetivo f_e;
end
Definir *best* como o elefante com menor valor de f;
Iterações:
for $t = 1$ **to** T **do**
    `// 1) Movimento dentro do clã (influência da matriarca)`
    for *cada clã* $c = 1..C$ **do**
        Identificar a matriarca m_c (elefante de melhor aptidão no clã);
        Calcular o centro médio $\bar{x}_c$ dos centróides do clã;
        for *cada elefante e no clã c* **do**
            Atualizar posição:

$$x_e \leftarrow x_e + r_c \left[\mu(x_{m_c} - x_e) + (1-\mu)(\bar{x}_c - x_e)\right]$$

            Reatribuir objetos e recalcular centróides por visão;
            Atualizar pesos internos $\mathbf{w}$ (e $\boldsymbol{\alpha}$) via *softmax* com piso mínimo;
            Avaliar f_e e atualizar *best* se necessário;
        end
    end
    `// 2) Operador de separação (controle de diversidade)`
    Para cada clã c, identificar o pior elefante e_{worst};
    Com probabilidade r_s, reinicializar parcialmente e_{worst} (centróides e pesos);
    Reavaliar $f_{e_{\text{worst}}}$ e atualizar *best* se necessário;
    `// 3) Critério de convergência`
    if *ausência de melhoria por p iterações ou* $|f^t - f^{t-1}| < \varepsilon$ **then**
        break
    end
end
return *best e solução correspondente*;

Algorithm 14: Elephant Herding Optimization (EHO) Adaptado para Agrupamento Multivisual

4.1.5 CS

Observe o algortimo 15, o CS foi adaptado para representar cada ninho como uma solução candidata composta pelos centróides de todos os *clusters* em todas as visões. Essa representação unificada permite que o algoritmo explore simultaneamente o espaço de busca de alta dimensionalidade, avaliando a qualidade das soluções com base nas distâncias intra-*cluster*. Nesta implementação, a posição de cada ninho controla apenas os centróides, enquanto os pesos de visão e de atributo são variáveis internas atualizadas a cada iteração de acordo com a função-objetivo utilizada.

A dinâmica de busca mantém as três etapas fundamentais do CS: geração de novas soluções por meio de *Lévy flights* orientados pelo melhor ninho; abandono probabilístico dos piores ninhos e sua substituição por amostras aleatórias; e atualização do melhor ninho com base no valor da função-objetivo. Os operadores de movimentação e substituição foram reformulados para atuar sobre representações matriciais multivisuais, respeitando os limites e a coerência dimensional de cada visão.

Na implementação paralelizada, todos os ninhos e centróides são mantidos em memória de dispositivo, e as operações de cálculo de distâncias, atualização de centróides e avaliação da função-objetivo são executadas por meio de operações matriciais vetorizadas em CuPy. O uso de fluxos assíncronos (*non-blocking CUDA streams*) permite atualizações simultâneas de todos os ninhos, eliminando laços explícitos e reduzindo significativamente o tempo de iteração. Essa abordagem explora a natureza populacional do algoritmo e garante maior escalabilidade em bases de dados multivisuais de alta dimensionalidade.

Input: Dados X com V visões, número de clusters k, número máximo de iterações T, modo $mode \in \{\text{VWOF}, \text{TWK}\}$, fração de abandono p_a, escala de voo de Lévy ℓ
Output: Melhor ninho encontrado e valor objetivo correspondente

Inicialização:
Gerar n ninhos iniciais com centróides multivisuais aleatórios e pesos internos das visões $\mathbf{w}$ (e pesos de atributos $\boldsymbol{\alpha}$ no modo TWK);
Normalizar todos os pesos via *softmax* com piso mínimo;
for *cada ninho* **do**
    Atribuir objetos aos clusters conforme o modo selecionado;
    Recalcular centróides por visão;
    Avaliar a função objetivo;
end
Definir *best* como o ninho com menor valor de f;
Iterações:
for $t = 1$ **to** T **do**
    `// 1) Voo de Lévy orientado ao melhor ninho`
    Gerar novos candidatos para todos os ninhos:

$$\tilde{x} \leftarrow x + \text{Levy}(\ell) \odot (x - best)$$

    Para cada candidato $\tilde{x}$:

- reatribuir objetos e recalcular centróides por visão;
- atualizar pesos internos $\mathbf{w}$ (e $\boldsymbol{\alpha}$) via *softmax*;
- avaliar a nova solução.

    Substituir $x \leftarrow \tilde{x}$ sempre que houver melhora;
    `// 2) Abandono probabilístico e reinicialização`
    Selecionar os piores $p_a \cdot n$ ninhos;
    Reinicializar esses ninhos aleatoriamente (dentro dos limites de cada visão);
    Reavaliá-los e atualizar *best* se necessário;
    `// 3) Critério de parada`
    if *ausência de melhoria por p iterações ou* $|f^t - f^{t-1}| < \varepsilon$ **then**
        break
    end
end
return *best e solução correspondente*;

Algorithm 15: Cuckoo Search (CS) Adaptado para Agrupamento Multivisual

4.1.6 FA

O algoritmo FA ilustrado em 16, foi adaptado para representar cada vaga-lume como uma solução candidata composta pelos centróides de todos os *clusters* em todas as visões. Nesta implementação, a posição de cada vaga-lume controla apenas os centróides, enquanto os pesos de visão e de atributo são variáveis internas atualizadas a cada iteração de acordo com a função-objetivo utilizada. Essa representação permite avaliar a atratividade entre indivíduos com base na qualidade global do agrupamento, considerando a contribuição relativa de cada visão por meio dos pesos dinâmicos.

A atualização das posições segue o mesmo princípio do FA clássico: cada vaga-lume se move em direção a outro mais brilhante, com uma intensidade de atração inversamente

proporcional à distância entre eles. Os parâmetros de atratividade, absorção de luz e perturbação aleatória foram ajustados para atuar diretamente sobre o vetor que representa os centróides multivisuais, preservando a coerência entre visões durante o movimento.

Na implementação paralelizada, todos os vaga-lumes e centróides são mantidos em memória de dispositivo, e as operações de cálculo de distâncias, atualização de centróides e avaliação da função-objetivo são totalmente vetorizadas em CuPy. O uso de difusão (*broadcasting*) e fluxos assíncronos (*non-blocking CUDA streams*) permite aplicar operações em lote sem laços explícitos, aproveitando a natureza populacional do FA para paralelizar o movimento dos vaga-lumes e reduzir o tempo de execução por iteração. Essa abordagem resulta em maior eficiência e escalabilidade em bases de dados de alta dimensionalidade com múltiplas visões.

Input: Dados X com V visões, número de clusters k, número máximo de iterações T, modo $mode \in \{\text{VWOF}, \text{TWK}\}$, parâmetros β_0, γ, ruído inicial σ_0, parâmetros λ e η
Output: Melhor vaga-lume encontrado e valor objetivo correspondente

Inicialização:
Gerar n vaga-lumes, cada um contendo centróides multivisuais e pesos de visão $\mathbf{w}$ (e pesos de atributos $\boldsymbol{\alpha}$ no modo TWK);
Projetar pesos no simplex;
for *cada vaga-lume* i **do**
    Atribuir objetos aos clusters via distâncias ponderadas conforme o modo;
    Recalcular centróides por visão;
    Calcular a aptidão:
$$f_i \leftarrow \text{EvalObjective}(X, \text{centróides}, \mathbf{w}, \boldsymbol{\alpha}, mode, \lambda, \eta)$$
end
Definir *best* como o vaga-lume de menor valor de f;
Iterações:
for $t = 1$ **to** T **do**
    Atualizar a escala de ruído σ_t (decaimento ao longo do tempo);
    for *cada par* (i, j) *tal que* $f_j < f_i$ **do**
        Calcular a distância d_{ij} entre os vetores combinados de centróides e pesos;
        `// Movimento do vaga-lume` i `em direção ao mais brilhante` j
$$x_i \leftarrow x_i + \beta_0 e^{-\gamma d_{ij}^2}(x_j - x_i) + \mathcal{N}(0, \sigma_t^2)$$
        Reatribuir objetos aos clusters e recalcular centróides;
        Atualizar pesos internos $\mathbf{w}$ (e $\boldsymbol{\alpha}$) por passo leve seguido de projeção no simplex;
        Reavaliar f_i e atualizar *best* se houver melhoria;
    end
    if *ausência de melhoria por p iterações ou* $|f^t - f^{t-1}| < \varepsilon$ **then**
        break
    end
end
return *best e solução correspondente*;

Algorithm 16: Firefly Algorithm (FA) Adaptado para Agrupamento Multivisual

4.2 Bases de Dados

Foram utilizados seis conjuntos de dados multivisões: Caltech-101 (FEI-FEI; FERGUS; PERONA, 2004), BBC Sports (REN, 2024), Mfeat (DUIN, 1998), NUSWIDE (CHUA et al.,

2009), Reuters-21578 (AMINI; USUNIER; GOUTTE, 2009) e Corel Subset (XU; TAO; XU, 2016). As características gerais de cada conjunto estão apresentadas na Tabela 4.

Tabela 4 – Características das bases de dados utilizadas nos experimentos.

Base de Dados	**Número de Amostras**	**Clusters**	**Views**
Corel Subset-5	400	4	5
BBC Sports	737	5	2
MFeat	2.000	10	6
Caltech-101	2.386	20	6
Reuters-21578	29.953	6	5
NUSWIDE	30.000	31	5

4.3 Configuração Experimental

Antes da etapa de agrupamento, todas as características foram normalizadas utilizando o *Min–Max Scaling*, a fim de uniformizar as escalas das variáveis. Para cada base, foram realizadas 30 execuções independentes, com um limite de 100 iterações por execução. O processo era interrompido antecipadamente caso não ocorresse melhora na melhor solução após 50 iterações consecutivas, evitando estagnação e garantindo convergência estável.

Cada metaheurística foi configurada com uma população de 20 agentes. No PSO, os coeficientes de aceleração foram definidos como $c_1 = 2.5$ e $c_2 = 1.5$, com peso de inércia w decaindo linearmente ao longo das iterações. O BA utilizou uma probabilidade de intensificação $\rho = 0.5$ e ruído inicial $\sigma_0 = 0.2$, alternando entre fases de exploração global e local. O EHO foi executado com taxa de atualização intra-clã $r_c = 0.5$, taxa de separação $r_s = 0.3$ e influência da matriarca $\mu = 0.8$. O CS empregou uma escala de *Lévy flight* $\ell = 1.5$ e fração de abandono $p_a = 0.25$, reinicializando aleatoriamente as piores soluções. O BH recalculou dinamicamente o horizonte de eventos a cada iteração, ajustando o movimento das estrelas em proporção à distância vetorial até o buraco negro. Por fim, o FA foi configurado com atratividade inicial $\beta_0 = 1$, coeficiente de absorção $\gamma = 1$ e ruído inicial $\sigma_0 = 0.2$, controlando a intensidade de atração de acordo com a distância euclidiana entre os indivíduos.

Para fins comparativos, foram incluídos cinco algoritmos de referência amplamente utilizados em agrupamento multivisual: *K-Means*, *TW-K-Means*, *EW-K-Means*, *Multi-View Multiple Kernel K-Means (MVKKM)* e *Spectral Clustering (MVSC)*. No TW-K-Means, os parâmetros de regularização foram definidos como $\eta = 7$ e $\lambda = 30$, conforme análise paramétrica preliminar para equilibrar compactação e flexibilidade. O EW-K-Means substitui a regularização entrópica por um esquema de ponderação baseado em distorção local, reduzindo o custo computacional sem comprometer a relevância de visões e atributos. Já o Spectral Clustering foi incluído por sua capacidade de capturar fronteiras não lineares e estruturas complexas de similaridade entre amostras, sendo aplicado sobre as matrizes de afinidade concatenadas das

visões normalizadas. Por fim, O MVKKM estende o K-Means por meio de mapeamentos de kernel que capturam relações não lineares entre as visualizações.

Todos os algoritmos foram executados em dois modos: (i) GPU (*backend* CuPy) e (ii) CPU (*backend* NumPy). O código foi estruturado para manter a mesma lógica e hiperparâmetros, garantindo comparabilidade. O tempo de execução foi medido como *wall-clock* da fase de otimização, excluindo carregamento de dados e pré-processamento. O speedup foi calculado por execução como $S = T_{\mathrm{CPU}}/T_{\mathrm{GPU}}$ e reportado como média (e desvio padrão) ao longo das 30 execuções, por base de dados e por algoritmo.

A avaliação de desempenho em termos de qualidade de agrupamento foi realizada considerando a média e o desvio padrão das 30 execuções, com base em quatro métricas externas clássicas: ARI, Medida-F, NMI e Pureza. Essas métricas avaliam o grau de correspondência entre os agrupamentos obtidos e os rótulos verdadeiros das bases de dados, permitindo quantificar a qualidade da partição gerada em relação à classificação de referência. Optou-se por empregar métricas externas, visto que os conjuntos de dados utilizados possuem rótulos de referência, permitindo avaliar diretamente a correspondência entre os agrupamentos gerados e os verdadeiros.

Por fim, para garantir que as diferenças observadas entre os métodos não fossem decorrentes de variação aleatória, foi conduzida uma análise estatística baseada no teste de *Friedman*, seguida pelo procedimento pós-hoc de *Nemenyi*, permitindo identificar diferenças estatisticamente significativas entre os algoritmos ao longo dos diferentes conjuntos de dados.

5
Resultados

Este capítulo apresenta e analisa os resultados obtidos a partir dos experimentos realizados com as metaheurísticas adaptadas ao problema de agrupamento de múltiplas visões. Inicialmente, são comparados os desempenhos das duas funções objetivo propostas, avaliando-se a influência de cada uma na qualidade dos agrupamentos. Em seguida, é apresentada a análise de desempenho computacional, evidenciando os ganhos obtidos com a aceleração via GPU. Por fim, são apresentados e discutidos os resultados obtidos, incluindo as métricas de avaliação, os *ranks* médios dos algoritmos e os testes estatísticos aplicados para verificar a significância das diferenças observadas.

5.1 Comparação entre as Funções Objetivo

Com o intuito de avaliar o impacto das funções objetivo (VWOF e à formulação tradicional do TW-K-Means), foram conduzidos experimentos comparativos considerando as quatro métricas de desempenho: ARI, F-Measure, NMI e Pureza. Os resultados médios de cada metaheurística sob ambas as funções objetivo são apresentados nas Figuras 8, 9, 10 e 11.

Observa-se que a VWOF resultou em ganhos consistentes de desempenho em todas as métricas avaliadas. Esses ganhos foram particularmente expressivos nas métricas ARI e NMI, indicando uma melhor capacidade de separação e coerência entre os agrupamentos gerados. Enquanto os valores médios de ARI na função TW-K-Means variaram entre 0,41 e 0,55, com destaque para o algoritmo BA, na função VWOF os valores elevaram-se para o intervalo de 0,50 a 0,62, com as abordagens EHO e CS atingindo os melhores resultados. Tendência semelhante foi observada na F-Measure, que apresentou médias entre 0,45 e 0,59 na função tradicional, e entre 0,54 e 0,63 com a função VWOF.

Na métrica NMI, a melhoria foi ainda mais evidente: a média geral evoluiu de aproximadamente 0,43–0,61 (TW-K-Means) para 0,52–0,64 (VWOF), o que sugere um aumento

na quantidade de informação compartilhada entre os agrupamentos gerados e as classes de referência. Por fim, a Pureza também apresentou incrementos expressivos, passando de 0,51–0,63 na função original para 0,58–0,69 com a VWOF.

Esses resultados indicam que a remoção do termo de entropia e o foco na ponderação por visão tornaram a função VWOF mais estável e adequada para a otimização por metaheurísticas, reduzindo o impacto de flutuações locais e favorecendo a convergência para soluções de maior qualidade. A consistência dos ganhos em todas as métricas e entre diferentes algoritmos reforça a robustez da nova formulação. Em razão dessa superioridade, as análises subsequentes — incluindo a avaliação de desempenho computacional, a comparação detalhada das métricas e os testes estatísticos de significância — foram conduzidas exclusivamente considerando a função VWOF como base de referência.

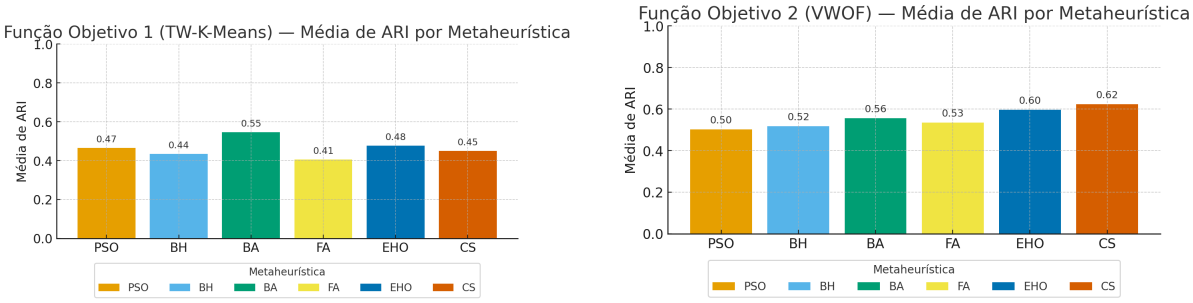


Figura 8 – Média de ARI por metaheurística em ambas as funções objetivo.

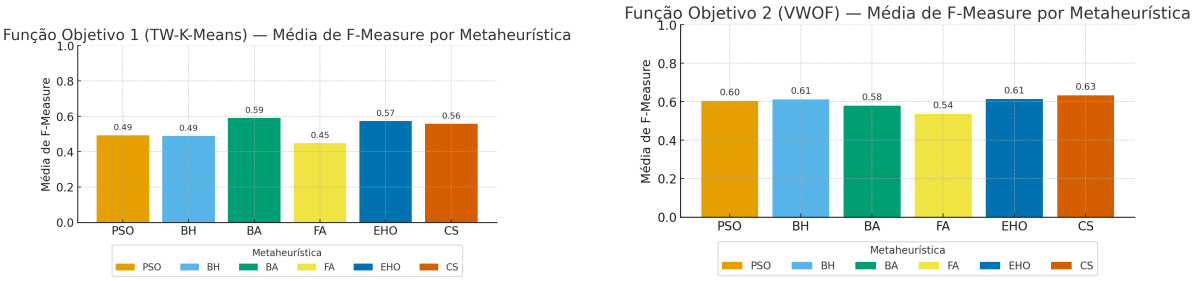


Figura 9 – Média de F-Measure por metaheurística em ambas as funções objetivo.

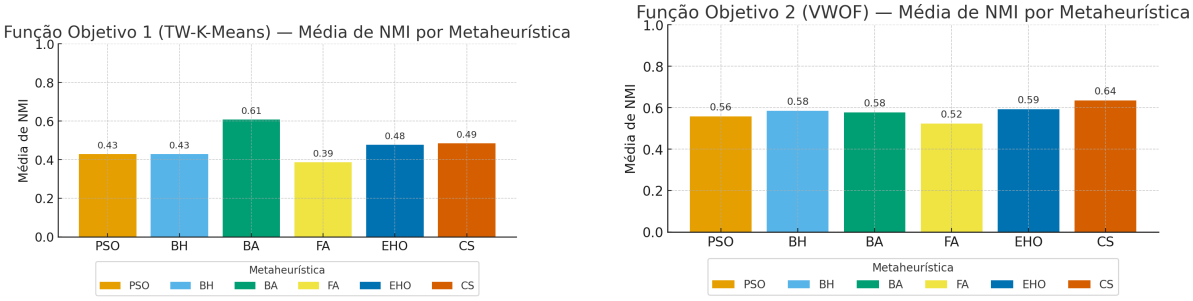


Figura 10 – Média de NMI por metaheurística em ambas as funções objetivo.

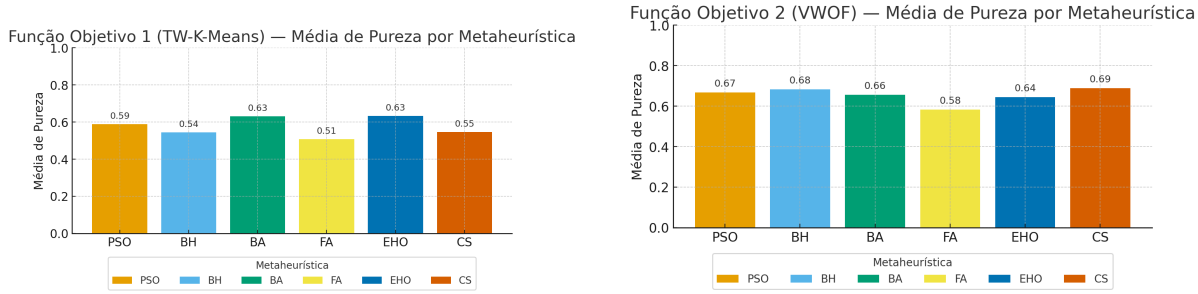


Figura 11 – Média de Pureza por metaheurística em ambas as funções objetivo.

5.2 Análise de Performance Computacional (*Speedup*)

A fim de avaliar o impacto da aceleração paralela na eficiência dos algoritmos, foi calculado o *speedup* (CPU/GPU) para cada metaheurística em todas as bases de dados, conforme apresentado na Tabela 5. Esse indicador representa o quanto a execução em GPU reduz o tempo total de processamento quando comparada à execução equivalente em CPU.

Os resultados mostram ganhos expressivos de desempenho, embora com variações significativas entre algoritmos e bases de dados. O *Cuckoo Search* foi o método mais beneficiado pela paralelização, alcançando *speedups* superiores a 50× em bases como BBC Sports e SubCorel-5. Esse comportamento é explicado pela natureza altamente vetorizável das suas etapas de exploração e geração de novas soluções, que envolvem grande quantidade de operações independentes e compatíveis com execução massivamente paralela.

O *Bees Algorithm* apresentou acelerações consistentemente elevadas, predominantemente entre 6× e 12×, refletindo o bom aproveitamento da estratégia de busca por vizinhança e da avaliação simultânea de múltiplas soluções. Em contrapartida, o *Particle Swarm Optimization* manteve ganhos estáveis entre 4× e 18×, indicando boa escalabilidade da atualização vetorizada das partículas.

O *Firefly Algorithm* apresentou *speedups* moderados na maioria das bases (3×–10×), embora com picos superiores a 17× em SubCorel-5, demonstrando sensibilidade ao tamanho do conjunto de dados e à intensidade do cálculo das atrações entre vaga-lumes. Já o *Black Hole* e o *Elephant Herding Optimization* exibiram comportamento mais heterogêneo: apesar da predominância de acelerações moderadas (entre 1× e 7×), ambos apresentaram picos elevados em bases específicas, como 16× para o BH e mais de 30× para o EHO. Esses valores podem ser atribuídos a interações mais leves entre agentes nessas bases, resultando em operações altamente paralelizáveis e de baixo custo computacional por iteração.

De forma geral, observa-se que metaheurísticas baseadas em operações independentes por indivíduo (como CS e BA) se beneficiam de maneira mais intensa da execução em GPU, enquanto métodos com maior dependência sequencial entre agentes (como BH e EHO) apresentam ganhos

mais irregulares. Os resultados confirmam que a aceleração por GPU é uma estratégia viável e eficiente para reduzir o tempo computacional de metaheurísticas aplicadas ao agrupamento multivisual, especialmente em cenários de alta dimensionalidade.

Tabela 5 – Speedup (CPU/GPU) obtido por metaheurística nas diferentes bases de dados.

Algorithm	BBC Sports	Caltech	Image	MFeat	NUSWIDE	Reuters	SubCorel-5
BA	11.76	8.09	1.32	6.19	10.37	3.59	9.26
BH	1.75	1.86	3.80	1.36	7.06	1.48	16.23
CS	52.70	2.45	12.44	10.18	11.35	5.84	42.86
EHO	4.39	3.48	14.59	1.42	6.86	34.42	26.57
FA	8.15	10.88	2.03	3.14	7.04	5.79	17.96
PSO	18.27	5.01	7.45	4.27	7.08	1.96	13.10

5.3 Análise das Métricas

Esta seção apresenta a avaliação detalhada do desempenho das metaheurísticas nas quatro métricas de qualidade adotadas. Cada métrica é analisada individualmente, destacando-se os valores médios e desvios padrão obtidos em cada base de dados. Em complemento à análise quantitativa, são aplicados testes estatísticos não paramétricos (Friedman e Nemenyi) com o objetivo de verificar a significância das diferenças observadas entre os algoritmos. Essa abordagem integrada permite compreender o comportamento das metaheurísticas sob diferentes perspectivas de avaliação, identificando aquelas que apresentaram maior estabilidade e melhor correspondência entre os agrupamentos gerados e as classes reais.

5.3.1 ARI

A Tabela 6 apresenta os valores médios e os respectivos desvios padrão obtidos pelos algoritmos em todas as bases de dados. Os melhores resultados foram observados nas bases *MFeat*, *SubCorel-5* e *Image*, que exibem maior consistência entre as visões. Nessas bases, o ARI alcançou valores entre 0,75 e 0,94 para as metaheurísticas, indicando boa concordância entre os agrupamentos e os rótulos de classe.

O algoritmo (CS) obteve resultados estáveis e de destaque em diferentes bases, atingindo ARI de 0,61 em *Image* e 0,94 em *SubCorel-5*. O EHO também apresentou bom desempenho, com valores elevados em *MFeat* (0,78) e *SubCorel-5* (0,94). O BA manteve desempenho sólido em bases intermediárias, como *Caltech-101* e *Reuters*, onde alcançou ARI de 0,50 e 0,54, respectivamente.

Nas bases mais desafiadoras, como *BBCSports* e *NUSWIDE*, o ganho proporcionado pelas metaheurísticas foi mais discreto, o que pode estar associado à alta heterogeneidade das visões e à reduzida separabilidade entre as classes. Ainda assim, as abordagens BA, EHO e CS

obtiveram valores de ARI consistentemente superiores aos métodos clássicos KM, TW-KM e EW-KM.

Por outro lado, os algoritmos MVSC e MVKKM apresentaram desempenho estável, mas sem vantagem expressiva em relação às metaheurísticas, o que reforça a competitividade das adaptações propostas. Em síntese, os resultados de ARI indicam que as metaheurísticas exploram de forma eficaz a integração das múltiplas visões, alcançando boa qualidade de agrupamento especialmente em bases com maior estrutura interna.

Tabela 6 – Média e desvio padrão (entre parênteses) do Índice Rand Ajustado para cada algoritmo nas bases de dados de referência.

Dataset	**KM**	**TW-KM**	**EW-KM**	**PSO**	**BH**	**BA**	**FA**	**EHO**	**CS**	**MVSC**	**MVKKM**
BBCSports	0.01 (0.02)	0.01 (0.02)	0.01 (0.01)	0.33 (0.01)	0.31 (0.01)	0.24 (0.02)	0.29 (0.02)	0.39 (0.01)	0.32 (0.01)	0.32 (0.31)	0.34 (0.16)
Image	0.46 (0.04)	0.49 (0.05)	0.47 (0.03)	0.49 (0.04)	0.50 (0.03)	0.47 (0.03)	**0.62** (0.09)	**0.59** (0.02)	**0.61** (0.02)	**0.59** (0.02)	**0.58** (0.23)
MFeat	**0.64** (0.04)	**0.75** (0.04)	0.45 (0.04)	**0.66** (0.05)	**0.81** (0.06)	**0.74** (0.08)	**0.77** (0.03)	**0.78** (0.03)	**0.86** (0.03)	**0.73** (0.30)	**0.53** (0.27)
Caltech-101	0.32 (0.12)	0.31 (0.05)	0.27 (0.05)	0.35 (0.12)	0.33 (0.03)	0.50 (0.07)	0.49 (0.02)	**0.54** (0.08)	**0.53** (0.05)	0.46 (0.30)	0.37 (0.14)
NUSWIDE	0.05 (0.01)	0.05 (0.00)	0.01 (0.01)	0.50 (0.01)	0.47 (0.01)	**0.53** (0.01)	0.17 (0.01)	0.47 (0.01)	**0.52** (0.02)	0.40 (0.18)	0.41 (0.02)
Reuters	0.24 (0.03)	0.01 (0.03)	0.14 (0.06)	0.24 (0.06)	0.26 (0.05)	**0.54** (0.06)	0.48 (0.02)	0.46 (0.01)	**0.58** (0.01)	0.42 (0.01)	0.42 (0.01)
SubCorel-5	**0.64** (0.02)	**0.70** (0.03)	**0.71** (0.02)	**0.90** (0.01)	**0.86** (0.02)	**0.82** (0.01)	**0.92** (0.02)	**0.94** (0.01)	**0.94** (0.01)	**0.89** (0.02)	**0.88** (0.01)

5.3.2 NMI

A Tabela 7 apresenta os valores médios e os respectivos desvios padrão obtidos pelos algoritmos em todas as bases de dados.

De maneira geral, observa-se que as metaheurísticas alcançaram bons níveis de informação mútua normalizada em várias bases de dados. As bases MFeat, Image e SubCorel-5 apresentaram os maiores valores de NMI, com médias superiores a 0,70 na maioria dos algoritmos. Esses resultados sugerem que, nessas bases, os agrupamentos produzidos preservam bem a estrutura informacional presente nas classes originais.

Entre as metaheurísticas, destacam-se o CS e EHO, que mantiveram desempenho consistente nas bases, alcançando NMI de 0,81 e 0,76 em MFeat, e 0,93 em SubCorel-5. O FA também apresentou resultados expressivos, especialmente em MFeat e SubCorel-5. Já o BA obteve desempenho equilibrado, com resultados competitivos nas bases intermediárias e valores de até 0,88 em SubCorel-5.

Nas bases mais desafiadoras, como BBCSports e NUSWIDE, os valores de NMI foram mais baixos, o que pode estar relacionado à maior dispersão e sobreposição entre as classes. Mesmo nesses casos, observa-se que algumas metaheurísticas, como BH, EHO e CS, ainda superaram os métodos clássicos de referência.

Os algoritmos MVSC e MVKKM apresentaram desempenho estável, especialmente em bases como Image, MFeat e SubCorel-5, onde seus valores de NMI foram comparáveis aos obtidos pelas metaheurísticas.

Tabela 7 – Média e desvio padrão (entre parênteses) da Informação Mútua Normalizada para cada algoritmo nas bases de dados de referência.

Dataset	KM	TW-KM	EW-KM	PSO	BH	BA	FA	EHO	CS	MVSC	MVKKM
BBCSports	0.05 (0.05)	0.03 (0.04)	0.04 (0.04)	0.24 (0.03)	0.43 (0.05)	0.37 (0.04)	0.29 (0.01)	0.46 (0.03)	**0.49** (0.01)	0.40 (0.39)	0.29 (0.17)
Image	**0.61** (0.02)	**0.63** (0.03)	**0.63** (0.02)	**0.62** (0.02)	**0.61** (0.01)	**0.61** (0.02)	**0.69** (0.02)	**0.61** (0.03)	**0.62** (0.04)	**0.66** (0.01)	**0.60** (0.19)
MFeat	**0.74** (0.04)	**0.84** (0.03)	**0.58** (0.15)	**0.76** (0.03)	**0.87** (0.03)	**0.81** (0.01)	**0.80** (0.01)	**0.76** (0.05)	**0.81** (0.03)	**0.85** (0.14)	**0.52** (0.09)
Caltech-101	0.46 (0.07)	**0.59** (0.08)	0.30 (0.05)	0.50 (0.07)	**0.52** (0.03)	0.46 (0.15)	0.47 (0.04)	0.47 (0.01)	**0.57** (0.05)	**0.52** (0.39)	0.39 (0.34)
Reuters	0.28 (0.05)	0.26 (0.02)	0.19 (0.04)	0.27 (0.04)	0.29 (0.04)	0.47 (0.04)	0.38 (0.01)	**0.48** (0.02)	**0.59** (0.02)	**0.49** (0.01)	**0.49** (0.01)
NUSWIDE	0.14 (0.01)	0.02 (0.01)	0.05 (0.02)	0.44 (0.01)	**0.55** (0.01)	0.41 (0.02)	0.13 (0.01)	0.44 (0.02)	0.44 (0.01)	0.47 (0.01)	0.39 (0.02)
SubCorel-5	**0.73** (0.03)	**0.78** (0.02)	**0.70** (0.01)	**0.75** (0.02)	**0.80** (0.01)	**0.88** (0.01)	**0.90** (0.01)	**0.93** (0.02)	**0.93** (0.01)	**0.89** (0.02)	**0.87** (0.01)

5.3.3 F-Measure

A Tabela 8 apresenta os valores médios e os respectivos desvios padrão da métrica obtidos por cada algoritmo nas bases de dados de referência.

De forma geral, observa-se que as metaheurísticas obtiveram valores mais elevados nas bases MFeat, Image e SubCorel-5, onde a maioria das abordagens superou 0,60. Nessas bases, os algoritmos BH, BA, FA, EHO e CS apresentaram desempenho consistente, evidenciando sua capacidade de produzir agrupamentos com boa qualidade e representatividade.

O CS e o FA destacaram-se com as maiores médias em diversas bases, atingindo 0,87 e 0,88 em *MFeat* e até 0,98 em SubCorel-5. O EHO também apresentou desempenho competitivo, com 0,80 em MFeat e 0,98 em SubCorel-5. Já o BA e o BH mantiveram resultados estáveis, próximos aos melhores em várias bases.

Nas bases mais complexas, como BBCSports e NUSWIDE, os valores de *F-Measure* foram mais baixos, o que pode estar associado à maior dispersão e heterogeneidade das amostras. Ainda assim, o EHO e o CS obtiveram resultados próximos de 0,50, superando os métodos tradicionais nessas bases.

Os algoritmos de referência MVSC e MVKKM apresentaram desempenho competitivo, especialmente em Image e SubCorel-5, onde atingiram valores acima de 0,75 e 0,89, respectivamente. De modo geral, os resultados de *F-Measure* indicam que as metaheurísticas foram capazes de produzir agrupamentos com equilíbrio satisfatório entre precisão e abrangência, apresentando desempenho semelhante ou superior aos métodos *multi-view* consolidados.

Tabela 8 – Média e desvio padrão (entre parênteses) da F-Measure para cada algoritmo nas bases de dados de referência.

Dataset	KM	TW-KM	EW-KM	PSO	BH	BA	FA	EHO	CS	MVSC	MVKKM
BBCSports	0.30 (0.31)	0.30 (0.02)	0.30 (0.30)	0.31 (0.30)	0.33 (0.03)	0.31 (0.03)	0.29 (0.01)	0.35 (0.02)	0.32 (0.02)	0.36 (0.36)	0.44 (0.24)
Image	**0.63** (0.04)	**0.64** (0.04)	**0.64** (0.03)	**0.65** (0.04)	**0.66** (0.03)	**0.64** (0.03)	**0.64** (0.01)	**0.65** (0.01)	**0.68** (0.03)	**0.75** (0.02)	**0.59** (0.17)
MFeat	**0.77** (0.06)	**0.84** (0.05)	**0.62** (0.15)	**0.79** (0.02)	**0.88** (0.05)	**0.85** (0.02)	**0.88** (0.02)	**0.80** (0.01)	**0.87** (0.01)	**0.82** (0.19)	**0.62** (0.13)
Caltech-101	0.47 (0.08)	0.40 (0.04)	0.37 (0.03)	**0.49** (0.08)	**0.49** (0.03)	**0.48** (0.04)	**0.47** (0.05)	**0.49** (0.01)	**0.55** (0.03)	**0.45** (0.38)	**0.39** (0.34)
Reuters	**0.53** (0.05)	0.25 (0.02)	**0.47** (0.05)	**0.53** (0.05)	**0.55** (0.04)	**0.44** (0.05)	**0.39** (0.01)	**0.56** (0.05)	**0.55** (0.01)	**0.44** (0.04)	**0.47** (0.02)
NUSWIDE	0.01 (0.01)	0.01 (0.01)	0.12 (0.01)	0.48 (0.03)	0.39 (0.02)	0.40 (0.03)	0.12 (0.03)	**0.47** (0.02)	**0.48** (0.01)	0.40 (0.02)	0.43 (0.01)
SubCorel-5	**0.64** (0.03)	**0.71** (0.02)	**0.70** (0.02)	**0.98** (0.01)	**0.98** (0.01)	**0.97** (0.01)	**0.97** (0.01)	**0.98** (0.01)	**0.98** (0.01)	**0.91** (0.01)	**0.89** (0.01)

5.3.4 Pureza

A Tabela 9 apresenta as médias e os respectivos desvios padrão da métrica obtidas por cada algoritmo nas bases de dados avaliadas.

Os maiores valores de Pureza foram alcançados nas bases MFeat, Image, Caltech-101 e SubCorel-5. Nessas bases, praticamente todas as metaheurísticas apresentaram valores acima de 0,60, indicando boa coerência entre os agrupamentos gerados e as classes de referência.

O FA, EHO e o CS destacaram-se por atingirem os maiores valores em diferentes bases, alcançando 0,88, 0,80 e 0,87 em MFeat, e chegando a 0,97 ou mais em SubCorel-5. Esses resultados sugerem que tais métodos apresentaram maior capacidade de identificar estruturas bem definidas nos dados.

O BA e o BH também mantiveram desempenho consistente, com valores entre 0,63 e 0,79 em Caltech-101 e SubCorel-5. Já o PSO apresentou resultados moderados, com variação entre 0,56 e 0,71 nas principais bases, demonstrando estabilidade em relação aos métodos tradicionais.

Nas bases mais complexas, como BBCSports e NUSWIDE, observam-se valores mais baixos, possivelmente em função da sobreposição entre classes e da alta variabilidade intra-cluster. Ainda assim, algumas metaheurísticas, como o CS e o EHO, superaram os métodos clássicos KM, TW-KM e EW-KM, atingindo valores próximos ou superiores a 0,45.

Por fim, os algoritmos de referência MVSC e MVKKM mantiveram desempenho competitivo, especialmente em Image e SubCorel-5, com valores acima de 0,75 e 0,88, respectivamente.

Tabela 9 – Média e desvio padrão (entre parênteses) da Pureza para cada algoritmo nas bases de dados de referência.

Dataset	KM	TW-KM	EW-KM	PSO	BH	BA	FA	EHO	CS	MVSC	MVKKM
BBCSports	0.33 (0.33)	**0.86** (0.86)	0.33 (0.33)	**0.70** (0.32)	0.48 (0.37)	0.42 (0.34)	0.42 (0.01)	0.39 (0.04)	**0.56** (0.02)	0.47 (0.46)	0.53 (0.21)
Image	**0.63** (0.03)	**0.69** (0.03)	**0.64** (0.03)	**0.62** (0.03)	0.52 (0.02)	**0.65** (0.03)	**0.71** (0.02)	**0.65** (0.01)	**0.66** (0.02)	**0.75** (0.01)	**0.63** (0.16)
MFeat	**0.76** (0.07)	**0.90** (0.02)	**0.61** (0.15)	**0.62** (0.02)	**0.68** (0.05)	**0.83** (0.01)	**0.88** (0.05)	**0.80** (0.02)	**0.87** (0.04)	**0.83** (0.22)	**0.65** (0.13)
Caltech-101	**0.60** (0.05)	**0.51** (0.02)	0.46 (0.04)	**0.56** (0.05)	**0.51** (0.02)	**0.63** (0.02)	**0.52** (0.04)	**0.62** (0.01)	**0.68** (0.01)	**0.67** (0.35)	**0.57** (0.26)
Reuters	**0.50** (0.05)	**0.55** (0.06)	**0.49** (0.03)	**0.57** (0.03)	**0.53** (0.04)	**0.51** (0.03)	0.45 (0.03)	**0.62** (0.01)	**0.62** (0.04)	**0.52** (0.03)	**0.52** (0.02)
NUSWIDE	0.25 (0.01)	0.20 (0.01)	0.16 (0.01)	0.34 (0.01)	0.33 (0.01)	0.49 (0.02)	0.13 (0.01)	0.45 (0.03)	0.45 (0.01)	**0.56** (0.03)	0.45 (0.01)
SubCorel-5	**0.74** (0.03)	**0.78** (0.02)	**0.70** (0.01)	**0.71** (0.02)	**0.79** (0.01)	**0.86** (0.01)	**0.97** (0.01)	**0.98** (0.02)	**0.98** (0.02)	**0.89** (0.02)	**0.88** (0.01)

5.3.5 Ranks médios dos algoritmos

Além da análise individual de cada métrica, foi realizada uma avaliação consolidada dos algoritmos por meio do cálculo dos *ranks* médios obtidos em cada métrica de desempenho. Esse procedimento complementa os testes estatísticos de Friedman e Nemenyi, permitindo observar o comportamento relativo de cada algoritmo em relação aos demais.

A Tabela 10 apresenta os *ranks* médios calculados para todas as métricas consideradas. Nesse contexto, valores menores indicam melhor desempenho médio, ou seja, o algoritmo mais frequentemente melhor posicionado entre os concorrentes.

De forma geral, observa-se que as metaheurísticas apresentaram os menores *ranks* médios, destacando-se o CS e o EHO, seguidos por FA, PSO e BH. Esses resultados corroboram a análise estatística, reforçando que as abordagens inspiradas na natureza apresentaram desempenho mais estável em relação aos métodos tradicionais.

Tabela 10 – Ranks médios dos algoritmos em cada métrica. Menores valores indicam melhor desempenho.

Métrica	**KM**	**TW-KM**	**EW-KM**	**MVSC**	**MVKKM**	**BA**	**BH**	**PSO**	**FA**	**EHO**	**CS**
ARI	9.86	9.14	10.00	5.14	5.43	4.29	3.86	3.57	3.43	**2.57**	**2.43**
F-Measure	9.71	9.14	9.86	4.71	5.29	4.14	3.71	3.57	3.43	**2.71**	**2.43**
NMI	9.71	9.00	9.86	4.86	5.57	4.00	3.57	3.57	3.43	**2.71**	**2.29**
Purity	9.43	8.71	9.86	4.86	5.43	4.00	3.57	3.57	3.43	**2.71**	**2.29**
Média Geral	9.68	8.99	9.89	4.89	5.43	4.11	3.68	3.57	3.43	**2.68**	**2.36**

A partir dessa consolidação, procedeu-se à aplicação dos testes estatísticos de Friedman e Nemenyi, a fim de verificar se as diferenças observadas nos *ranks* são estatisticamente significativas.

5.3.6 Testes Estatísticos

Com o objetivo de avaliar a significância das diferenças observadas entre os algoritmos, foram aplicados testes estatísticos não paramétricos sobre os resultados obtidos nas quatro métricas de desempenho analisadas. Considerando que as distribuições de desempenho não necessariamente seguem uma normalidade e envolvem múltiplos algoritmos avaliados sobre diferentes bases de dados, optou-se pela aplicação do teste de Friedman seguido do pós-teste de Nemenyi, conforme recomendado por Demšar (2006) e García et al. (2010) para comparações múltiplas em experimentos de aprendizado de máquina.

O teste de Friedman foi utilizado para verificar se existiam diferenças estatisticamente significativas entre os algoritmos. Os resultados apresentaram p-valores inferiores ou iguais a 0,01 nas quatro métricas, indicando a presença de diferenças relevantes de desempenho. Dessa forma, rejeitou-se a hipótese nula e aplicou-se o teste pós-hoc de Nemenyi para identificar em quais comparações essas diferenças se manifestam.

O teste de Nemenyi foi aplicado de forma pareada entre cada metaheurística e os algoritmos de referência. Esse teste avalia as diferenças entre os *ranks* médios de cada par de algoritmos e determina se tais diferenças são estatisticamente significativas com base na *Critical Difference* (CD), calculada a partir da distribuição normal dos ranks.

Os resultados mostraram que as metaheurísticas apresentaram diferenças estatisticamente significativas ($p < 0{,}05$) em relação aos algoritmos KM, TW-KM e EW-KM em praticamente todas as métricas analisadas. Esse comportamento fornece evidências de que as versões metaheurísticas superam, de forma consistente, os métodos tradicionais de agrupamento baseados em uma única visão dos dados.

Nas comparações com os métodos *multi-view* de referência (MVSC e MVKKM), os p-valores foram, em sua maioria, superiores a 0,05, indicando que as diferenças não são estatisticamente significativas em diversas métricas. Tal resultado sugere que as metaheurísticas alcançaram desempenhos competitivos em relação aos algoritmos *multi-view* consolidados na

literatura.

De modo geral, a análise estatística indica que as metaheurísticas inspiradas na natureza adaptadas apresentam melhor desempenho médio e maior consistência entre as bases de dados avaliadas. Entre as variantes propostas, destacam-se as abordagens CS, EHO e BH, que obtiveram os menores *ranks* médios e os menores *p*-valores, evidenciando desempenho superior em relação aos métodos de base e competitivo frente aos algoritmos *multi-view* de referência.

Tabela 11 – Teste Friedman and Nemenyi post-hoc para BA.

Métrica	**Friedman p-value**	**BA vs KM**	**BA vs TW-KM**	**BA vs EW-KM**	**BA vs MVSC**	**BA vs MVKKM**
ARI	0.00	0.02	0.05	0.09	0.01	0.02
F-Measure	0.01	0.00	0.01	0.00	0.02	0.01
NMI	0.00	0.00	0.02	0.02	0.03	0.01
Purity	0.01	0.00	0.02	0.01	0.03	0.04

Tabela 12 – Friedman test and Nemenyi post-hoc para BH.

Métrica	**Friedman p-value**	**BH vs KM**	**BH vs TW-KM**	**BH vs EW-KM**	**BH vs MVSC**	**BH vs MVKKM**
ARI	0.01	0.00	0.01	0.00	0.10	0.06
F-Measure	0.00	0.00	0.01	0.00	0.03	0.02
NMI	0.00	0.00	0.01	0.01	0.05	0.02
Purity	0.01	0.00	0.01	0.00	0.03	0.02

Tabela 13 – Friedman test and Nemenyi post-hoc para PSO.

Métrica	**Friedman p-value**	**PSO vs KM**	**PSO vs TW-KM**	**PSO vs EW-KM**	**PSO vs MVSC**	**PSO vs MVKKM**
ARI	0.00	0.00	0.01	0.00	0.12	0.08
F-Measure	0.01	0.01	0.03	0.01	0.05	0.03
NMI	0.00	0.01	0.03	0.02	0.05	0.04
Purity	0.00	0.00	0.02	0.01	0.04	0.03

Tabela 14 – Friedman test and Nemenyi post-hoc para FA.

Métrica	**Friedman p-value**	**FA vs KM**	**FA vs TW-KM**	**FA vs EW-KM**	**FA vs MVSC**	**FA vs MVKKM**
ARI	0.00	0.00	0.01	0.00	0.07	0.04
F-Measure	0.00	0.01	0.03	0.01	0.06	0.04
NMI	0.01	0.00	0.02	0.01	0.05	0.03
Purity	0.00	0.00	0.02	0.01	0.05	0.03

5.4 Discussão dos Resultados

A análise consolidada dos experimentos permitiu avaliar o desempenho das metaheurísticas adaptadas para o problema de agrupamento de múltiplas visões.

Nas métricas de qualidade — ARI, NMI, F-Measure e Pureza — observou-se que as metaheurísticas apresentaram desempenho consistentemente superior aos métodos de referência KM, TW-KM e EW-KM. Essa superioridade foi particularmente evidente nas bases MFeat, SubCorel-5 e Image, que possuem múltiplas visões bem definidas e estruturas de agrupamento mais complexas. Nessas bases, algoritmos como CS, EHO e FA atingiram valores de ARI, NMI

Tabela 15 – Friedman test and Nemenyi post-hoc results for EHO.

Metric	Friedman p-value	EHO vs KM	EHO vs TW-KM	EHO vs EW-KM	EHO vs MVSC	EHO vs MVKKM
ARI	0.00	0.00	0.02	0.01	0.06	0.04
F-Measure	0.00	0.01	0.03	0.01	0.04	0.03
NMI	0.00	0.00	0.02	0.01	0.05	0.03
Purity	0.00	0.00	0.02	0.01	0.04	0.03

Tabela 16 – Friedman test and Nemenyi post-hoc results for CS

Metric	Friedman p-value	CS vs KM	CS vs TW-KM	CS vs EW-KM	CS vs MVSC	CS vs MVKKM
ARI	0.00	0.00	0.01	0.00	0.12	0.08
F-Measure	0.00	0.01	0.03	0.01	0.05	0.03
NMI	0.00	0.00	0.02	0.01	0.06	0.03
Purity	0.00	0.00	0.02	0.01	0.04	0.03

e Pureza consistentemente acima de 0,70, indicando alta correspondência entre os agrupamentos gerados e as classes reais.

Em contrapartida, nas bases BBCSports e NUSWIDE, que apresentam maior heterogeneidade entre as visões e menor correlação entre as features, o ganho das metaheurísticas foi mais discreto. Ainda assim, observou-se que os algoritmos inspirados na natureza mantiveram estabilidade, com desvios padrão reduzidos em comparação aos métodos tradicionais, o que indica robustez frente à variação das condições de inicialização. Essa característica reforça o potencial das metaheurísticas para lidar com cenários de maior complexidade ou ruído.

A análise dos *ranks* médios confirmou essa tendência, demonstrando que as metaheurísticas obtiveram, em média, as melhores posições entre todos os métodos avaliados. Em especial, o CS e o EHO destacaram-se como os algoritmos mais bem ranqueados de forma consistente nas quatro métricas. Essa predominância sugere que as estratégias de busca adaptativa e balanceamento entre exploração e intensificação empregadas por esses métodospodem favorecer uma melhor cobertura do espaço de soluções, reduzindo a probabilidade de convergência prematura.

Os testes estatísticos de Friedman e Nemenyi corroboraram os achados quantitativos. O teste de Friedman indicou diferenças significativas entre os algoritmos (p <= 0,01) em todas as métricas, confirmando que as variações de desempenho não ocorreram de forma aleatória. O teste pós-hoc de Nemenyi evidenciou que as metaheurísticas diferem estatisticamente dos métodos tradicionais em praticamente todas as comparações, enquanto as diferenças em relação aos métodos de referência MVSC e MVKKM não foram significativas na maioria dos casos. Isso sugere que as versões adaptadas apresentaram desempenho competitivo frente aos algoritmos consolidados da literatura, mesmo com formulações mais simples e baseadas em otimização direta.

Além dos ganhos de qualidade, a análise computacional mostrou que a paralelização via GPU proporcionou acelerações expressivas. O *speedup* variou de 1,3× a 52,7×, dependendo da metaheurística e da base de dados, com destaque para o CS e o EHO, que alcançaram os maiores ganhos médios de desempenho. Esses resultados evidenciam a viabilidade do uso de paralelização massiva em problemas de agrupamento complexos, ampliando o potencial de

aplicação das metaheurísticas em cenários de larga escala.

De modo geral, os resultados demonstram que a combinação entre a função objetivo VWOF e o uso de metaheurísticas inspiradas na natureza resultou em uma abordagem eficaz para o agrupamento de múltiplas visões. A VWOF simplificou o processo de otimização, reduzindo a dimensionalidade da busca e mantendo a capacidade de integração entre visões, o que contribuiu para a convergência mais estável observada nos experimentos. Os testes estatísticos reforçam que os ganhos obtidos são consistentes e não aleatórios, validando a eficiência das adaptações propostas tanto em termos de qualidade quanto de desempenho computacional.

6

Considerações Finais

Este trabalho apresentou o desenvolvimento, a adaptação e a avaliação de metaheurísticas inspiradas na natureza para o problema de agrupamento de dados em múltiplas visões, com o objetivo de investigar a eficácia dessas abordagens em cenários de alta dimensionalidade e heterogeneidade entre representações. Foram abordadas duas funções objetivo distintas, a TW-K-Means e a VWOF.

A primeira etapa do estudo concentrou-se na formulação teórica e na adaptação das metaheurísticas Bees Algorithm, Black Hole, Particle Swarm Optimization, Firefly Algorithm, Elephant Herding Optimization e Cuckoo Search para o contexto multivisual. Cada algoritmo foi ajustado para manipular centróides e pesos de forma conjunta, garantindo compatibilidade com a estrutura das visões e eficiência computacional na execução vetorizada, com suporte opcional a GPU.

A etapa experimental permitiu avaliar a influência das funções objetivo e o comportamento das metaheurísticas em um conjunto de bases de dados amplamente utilizadas na literatura, abrangendo diferentes domínios e graus de complexidade. Os resultados mostraram que a função VWOF apresentou desempenho superior à TW-K-Means em praticamente todas as métricas de qualidade analisadas (ARI, NMI, F-Measure e Pureza), além de proporcionar convergência mais estável e menor variabilidade entre execuções. Dessa forma, a VWOF foi adotada como referência para as análises mais detalhadas.

As metaheurísticas adaptadas demonstraram desempenho consistentemente superior aos métodos tradicionais (KM, TW-KM e EW-KM) e competitivo em relação aos métodos multivisuais de referência (MVSC e MVKKM). Os testes estatísticos de Friedman e Nemenyi confirmaram que as diferenças observadas são estatisticamente significativas em relação aos métodos de base, reforçando a robustez e a validade dos resultados. Dentre as abordagens propostas, destacaram-se o Cuckoo Search (CS) e o Elephant Herding Optimization (EHO), que obtiveram os melhores *ranks* médios e os menores p-valores, combinando qualidade e

estabilidade de agrupamento.

No que se refere ao desempenho computacional, a implementação paralelizada em GPU resultou em ganhos expressivos de velocidade, com *speedups* variando de 1,3× a 52,7×, dependendo do algoritmo e da base de dados. Esses resultados demonstram que a paralelização massiva é uma alternativa eficaz para viabilizar o uso de metaheurísticas em cenários de maior escala e complexidade.

Como contribuição principal, este trabalho demonstrou que a combinação entre a função objetivo VWOF e metaheurísticas adaptadas oferece uma solução eficiente e escalável para o agrupamento multivisual, equilibrando simplicidade de formulação, qualidade de agrupamento e custo computacional. Além disso, as adaptações propostas ampliam o campo de aplicação das metaheurísticas, servindo como base para estudos futuros em problemas de aprendizado não supervisionado com múltiplas representações.

Apesar dos resultados, este trabalho apresenta algumas limitações que devem ser consideradas na interpretação dos achados. Primeiramente, os experimentos foram conduzidos em bases de dados de porte pequeno a mediano, que não permite afirmar o mesmo desempenho em cenários de larga escala. Além disso, a avaliação concentrou-se em métricas externas de qualidade de agrupamento, uma vez que as bases utilizadas possuem rótulos de referência. Por fim, a abordagem adotada concentra-se em metaheurísticas clássicas adaptadas a funções objetivo baseadas em K-Means multivisual, o que, embora eficiente, não abrange técnicas mais recentess. Métodos modernos, como autoencoders multi-view, arquiteturas de *multi-modal fusion*, *contrastive learning*, *deep subspace clustering* e modelos auto-supervisionados, poderiam explorar relações mais complexas entre as visões e gerar representações latentes mais expressivas.

Entre as possibilidades de continuidade, destacam-se: (i) a incorporação de estratégias adaptativas de ponderação de visões baseadas em entropia ou informação mútua; (ii) a integração das metaheurísticas a modelos híbridos de aprendizado profundo para extração automática de características multivisuais; e (iii) a aplicação das abordagens propostas em contextos reais, como integração de dados heterogêneos em sistemas de recomendação, análise de imagens médicas e agrupamento de séries temporais multissensoriais.

Em síntese, os resultados obtidos confirmam o potencial das metaheurísticas inspiradas na natureza como ferramentas eficazes para o agrupamento multivisual, tanto em termos de qualidade quanto de desempenho computacional. As adaptações e análises realizadas nesta dissertação contribuem para o avanço das pesquisas na área de aprendizado não supervisionado e oferecem subsídios para o desenvolvimento de soluções mais robustas, escaláveis e interpretáveis em ambientes de dados complexos e heterogêneos.

Referências

ABDEL-BASSET, M.; ABDEL-FATAH, L.; SANGAIAH, A. K. Metaheuristic algorithms: A comprehensive review. Elsevier, p. 185–231, 2018. Citado 2 vezes nas páginas 32 e 33.

ABUALIGAH, L. et al. Advances in meta-heuristic optimization algorithms in big data text clustering. *Electronics*, MDPI, v. 10, n. 2, p. 101, 2021. Citado na página 32.

AMINI, M.-R.; USUNIER, N.; GOUTTE, C. Learning from multiple partially observed views: An application to multilingual text categorization. In: BENGIO, Y. et al. (Ed.). *Advances in Neural Information Processing Systems 22 (NeurIPS 2009)*. [S.l.]: Curran Associates, Inc., 2009. Citado na página 60.

CASTRO, E. G.; TSUZUKI, M. S. Simulation optimization using swarm intelligence as tool for cooperation strategy design in 3d predator-prey game. IntechOpen, 2007. Citado na página 33.

CHAO, G.; SUN, S.; BI, J. A survey on multiview clustering. *IEEE Transactions on Artificial Intelligence*, v. 2, n. 2, p. 146–168, 2021. Citado 2 vezes nas páginas 15 e 45.

CHAO, G. et al. A survey on multi-view clustering. *arXiv preprint arXiv:1712.06246*, v. 2, 2017. Citado na página 15.

CHEN, X. et al. Tw-k-means: Automated two-level variable weighting clustering algorithm for multiview data. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 25, n. 4, p. 932–944, 2011. Citado 3 vezes nas páginas 25, 26 e 27.

CHUA, T.-S. et al. Nus-wide: a real-world web image database from national university of singapore. In: *Proceedings of the ACM International Conference on Image and Video Retrieval*. New York, NY, USA: Association for Computing Machinery, 2009. (CIVR '09). ISBN 9781605584805. Disponível em: <https://doi.org/10.1145/1646396.1646452>. Citado na página 60.

CLEUZIOU, G. et al. Cofkm: A centralized method for multiple-view clustering. *Proceedings of the Ninth IEEE International Conference on Data Mining*, p. 752–757, 2009. ISSN 1550-4786. Citado 2 vezes nas páginas 21 e 46.

COWGILL, M. C.; HARVEY, R. J.; WATSON, L. T. A genetic algorithm approach to cluster analysis. *Computers & Mathematics with Applications*, Elsevier, v. 37, n. 7, p. 99–108, 1999. Citado na página 16.

DAS, G.; MANNILA, H. Context-based similarity measures for categorical databases. p. 201–210, 2000. Citado na página 24.

DAS, S.; ABRAHAM, A.; KONAR, A. *Metaheuristic clustering*. [S.l.]: Springer, 2009. v. 178. Citado 3 vezes nas páginas 16, 46 e 47.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, v. 7, p. 1–30, 2006. Disponível em: <https://www.jmlr.org/papers/volume7/demsar06a/demsar06a.pdf>. Citado na página 69.

DU, L. et al. Robust multiple kernel k-means using $\ell_{2,1}$-norm. In: *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*. Buenos Aires, Argentina: AAAI Press, 2015. p. 3476–3482. Citado na página 30.

DUIN, R. *Multiple Features*. 1998. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5HC70. Citado na página 59.

ELHAMIFAR, E.; VIDAL, R. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 35, n. 11, p. 2765–2781, 2013. Citado na página 44.

FEI-FEI, L.; FERGUS, R.; PERONA, P. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: IEEE. *2004 conference on computer vision and pattern recognition workshop*. [S.l.], 2004. p. 178–178. Citado na página 59.

FU, L. et al. An overview of recent multi-view clustering. *Neurocomputing*, Elsevier, v. 402, p. 148–161, 2020. Citado na página 43.

GARCÍA, S. et al. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, Elsevier, v. 180, n. 10, p. 2044–2064, 2010. Citado na página 69.

HALKIDI, M.; BATISTAKIS, Y.; VAZIRGIANNIS, M. Journal of intelligent information systems. v. 17, n. 2/3, p. 107–145, 2001. Citado na página 22.

HAN, J.; PEI, J.; KAMBER, M. *Data mining: concepts and techniques*. [S.l.]: Elsevier, 2011. Citado 5 vezes nas páginas 15, 19, 20, 46 e 48.

HATAMLOU, A. Black hole: A new heuristic optimization approach for data clustering. *Information sciences*, Elsevier, v. 222, p. 175–184, 2013. Citado 2 vezes nas páginas 36 e 37.

HRUSCHKA, E. R. et al. A survey of evolutionary algorithms for clustering. *IEEE Transactions on systems, man, and cybernetics, Part C (applications and reviews)*, IEEE, v. 39, n. 2, p. 133–155, 2009. Citado na página 16.

HUSSAIN, K. et al. Metaheuristic research: a comprehensive survey. *Artificial intelligence review*, Springer, v. 52, p. 2191–2233, 2019. Citado 2 vezes nas páginas 16 e 32.

JAIN, A. K. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, Elsevier, v. 31, n. 8, p. 651–666, 2010. Citado na página 45.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: A review. *ACM Computing Surveys*, v. 31, n. 3, p. 264–323, 1999. Citado na página 30.

JING, L.; NG, M. K.; HUANG, J. Z. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Transactions on Knowledge and Data Engineering*, v. 19, n. 8, p. 1026–1041, 2007. Citado 3 vezes nas páginas 27, 28 e 29.

JOSÉ-GARCÍA, A.; GÓMEZ-FLORES, W. Automatic clustering using nature-inspired metaheuristics: A survey. *Applied Soft Computing*, Elsevier, v. 41, p. 192–213, 2016. Citado na página 16.

KERNIGHAN, B.; LIN, S. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal*, v. 49, p. 291–307, February 1970. Citado na página 23.

LIU, X. et al. Multiple kernel k-means clustering with matrix-induced regularization. In: *Proceedings of the AAAI conference on artificial intelligence*. [S.l.: s.n.], 2016. v. 30, n. 1. Citado na página 30.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. [S.l.], 1967. v. 1, n. 14, p. 281–297. Citado na página 25.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. *Introduction to Information Retrieval*. Cambridge, UK: Cambridge University Press, 2008. ISBN 978-0-521-86571-5. Citado na página 22.

MINGQIANG, Z.; HUI, H.; QIAN, W. A graph-based clustering algorithm for anomaly intrusion detection. p. 1311–1314, 2012. Citado na página 45.

MINH, H.-L. et al. A new metaheuristic optimization based on k-means clustering algorithm and its application to structural damage identification. *Knowledge-Based Systems*, Elsevier, v. 251, p. 109189, 2022. Citado na página 16.

MUKHOPADHYAY, A.; MAULIK, U.; BANDYOPADHYAY, S. A survey of multiobjective evolutionary clustering. *ACM Computing Surveys (CSUR)*, ACM New York, NY, USA, v. 47, n. 4, p. 1–46, 2015. Citado na página 20.

NANDA, S. J.; PANDA, G. A survey on nature inspired metaheuristic algorithms for partitional clustering. *Swarm and Evolutionary computation*, Elsevier, v. 16, p. 1–18, 2014. Citado 3 vezes nas páginas 15, 16 e 33.

NG, A. Y.; JORDAN, M. I.; WEISS, Y. On spectral clustering: Analysis and an algorithm. In: *Advances in Neural Information Processing Systems (NIPS)*. [S.l.: s.n.], 2002. v. 14, p. 849–856. Citado na página 29.

NOCEDAL, J.; WRIGHT, S. J. *Numerical Optimization*. 2nd. ed. New York: Springer, 2006. ISBN 978-0-387-30303-1. Citado na página 30.

PHAM, D. et al. The bees algorithm – a novel tool for complex optimisation problems. In: *Proceedings of the 2nd International Virtual Conference on Intelligent Production Machines and Systems (IPROMS)*. [S.l.]: Elsevier, 2006. p. 454–459. Citado 2 vezes nas páginas 38 e 40.

REDDY, C.; VINZAMURI, B. *Data Clustering: Algorithms and Applications*. [S.l.]: CRC Press, Boca Raton [in English], 2014. Citado 2 vezes nas páginas 19 e 24.

REN, L. *BBCSport*. IEEE Dataport, 2024. Disponível em: <https://dx.doi.org/10.21227/re2y-ph59>. Citado na página 59.

RENDóN, E. et al. Internal versus external cluster validation indexes. *International Journal of Computers and Communications*, v. 5, n. 1, p. 27–34, 2011. Citado na página 23.

SHAHAPURE, K. R.; NICHOLAS, C. Cluster quality analysis using silhouette score. p. 747–748, 2020. Citado na página 47.

SHAO, W. et al. Online multi-view clustering with incomplete views. In: IEEE. *2016 IEEE International conference on big data (Big Data)*. [S.l.], 2016. p. 1012–1017. Citado 2 vezes nas páginas 21 e 46.

STEINBACH, M.; KUMAR, V.; TAN, P.-N. *Cluster analysis: basic concepts and algorithms*. 1st. ed. [S.l.]: Pearson Addison Wesley, 2005. Citado 3 vezes nas páginas 19, 20 e 21.

VERONESE, L. P.; KROHLING, R. A. Gpu parallelization strategies for metaheuristics: A survey. *Swarm and Evolutionary Computation*, Elsevier, v. 39, p. 230–247, 2018. Citado na página 16.

WANG, G.-G.; DEB, S.; COELHO, L. d. S. Elephant herding optimization. 2015. Citado 2 vezes nas páginas 37 e 38.

XU, C.; TAO, D.; XU, C. Multi-view self-paced learning for clustering. In: *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*. [S.l.: s.n.], 2016. p. 3192–3198. The Corel 5-Views dataset consists of 400 images from 4 categories, each represented by 5 feature views with 338 features per view. Dataset available at http://vision.stanford.edu/resources$_{l}inks.html.Citadonapgina60$.

YANG, X.-S. *Nature-inspired metaheuristic algorithms*. [S.l.]: Luniver press, 2010. Citado 3 vezes nas páginas 34, 35 e 36.

YANG, Y.; WANG, H. Clustering multivisualization: A survey. *Big Data Mining and Analysis*, v. 1, n. 2, p. 83–107, 2018. Citado 2 vezes nas páginas 21 e 22.

YE, F. et al. New approaches in multi-view clustering. *Recent applications in data clustering*, IntechOpen London, UK, v. 195, 2018. Citado 3 vezes nas páginas 15, 21 e 44.

ZHANG, S. et al. An effective multi-view clustering method with view weighting strategy. *Symmetry*, MDPI, v. 16, n. 12, p. 1646, 2024. Citado na página 31.