Pós-Graduação em Ciência da Computação

Rodrigo Cavalcanti de Araújo

Um modelo de agrupamento multi-view com ponderação simultânea de tabelas e variáveis

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife

2018

Rodrigo Cavalcanti de Araújo

Um modelo de agrupamento multi-view com ponderação simultânea de tabelas e variáveis

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Francisco de Assis Tenório de Carvalho

Recife
2018

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

A663m Araújo, Rodrigo Cavalcanti de
Um modelo de agrupamento multi-view com ponderação simultânea de tabelas e variáveis / Rodrigo Cavalcanti de Araújo. – 2018.
85 f.: il., fig., tab.

Orientador: Francisco de Assis Tenório de Carvalho.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2018.
Inclui referências e apêndices.

1. Inteligência computacional. 2. Agrupamento de dados. I. Carvalho, Francisco de Assis Tenório de (orientador). II. Título.

006.3 CDD (23. ed.) UFPE- MEI 2018-087

Rodrigo Cavalcanti de Araújo

Um Modelo de Agrupamento Multi-view com Ponderação Simultânea de Tabelas e Variáveis

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Aprovado em: 01/06/2018

BANCA EXAMINADORA

_____

Prof. Paulo Salgado Gomes de Mattos Neto
Centro de Informática / UFPE

_____

Prof. Marcelo Rodrigo Portela Ferreira
Departamento de Estatística / UFPB

_____

Prof. Francisco de Assis Tenório de Carvalho
Centro de Informática / UFPE
(Orientador)

Este trabalho é dedicado a minha família.

AGRADECIMENTOS

Agradeço a todos que me acompanharam e me motivaram durante a elaboração deste trabalho, aos professores e professoras do Centro de Informática que me ajudaram com o conhecimento necessário. Agradeço especialmente ao meu orientador professor Francisco de Assis Tenório de Carvalho pela oportunidade, compreensão e incentivo. Por fim, agradeço a toda minha família, principalmente a minha mãe, minha esposa e minha filha pelo apoio, motivação e paciência.

"É necessário dizer que não é a quantidade de informações, nem a sofisticação em Matemática que que podem dar sozinhas um conhecimento pertinente, mas sim a capacidade de colocar o conhecimento no contexto."

(Edgar Morin)

RESUMO

Modelos de agrupamento *multi-view* podem ser vistos como uma maneira de extrair informações de diferentes perspectivas dos dados para melhorar a precisão do particionamento resultante. No agrupamento de dados *muti-view*, é comum existir tabelas irrelevantes e, entre as relevantes, algumas podem ser mais ou menos importantes para contribuir na definição do particionamento final. Por esse motivo, a maior parte dos algoritmos existentes que trabalham com esse tipo de dado atribuem um peso a cada tabela com o objetivo de calcular as relevâncias destas visões no processo de agrupamento. No entanto, poucos algoritmos calculam, além do peso de relevância das visões, os pesos de relevância das variáveis dentro de cada visão com o objetivo de obter, também, a seleção automatizada desses atributos. Este trabalho propõe um algoritmo de agrupamento exclusivo do tipo *c-means* para dados *muti-view* que cálcula, de forma automática e simultânea, os pesos para tabelas e variáveis, de modo que as informações relevantes sejam selecionadas para definição da partição. Em comparação com trabalhos similares anteriores, uma vantagem do método proposto é que, além da necessidade de conhecer previamente o número de *clusters*, não há parâmetros adicionais que precisem ser ajustados. Para validação dos resultados, experimentos com conjuntos de dados de *benchmark* demonstram a utilidade do método proposto.

Palavras-chaves: Agrupamento de dados. Dados Multi-View. C-Means. Ponderação das visões. Ponderação das variáveis.

ABSTRACT

Multi-View Clustering models can be viewed as a way to extract information from different data representations to improve the clustering accuracy. In multi-view clustering, it is common to exist some irrelevant views and among the relevant ones, some may be more or less relevant than others to contribute in the final partition. This is why the most part of existing multi-view algorithms assign a weight to each view aiming to compute its relevance in the clustering process. However very few algorithms computes also the relevance weight of variables inside each view aiming to achieve automated feature selection. This work proposes a muti-view hard c-means clustering algorithm with automated computation of weights for both views and variables in such a way that the relevant views as well as the relevant variables in each view are selected for clustering. Compared to previous similar works, an advantage of the proposed method is that, apart the need to know previously the number of clusters, there are no additional parameters to tune. Experiments with benchmark data sets corroborate the usefulness of the proposed method.

Key-words: Clustering. Multi-view data. C-Means. View weighting. Feature weighting.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

MVKKM	*Multi-view Kernel K-Means.*
TW-K-Means	*Automated Two-Level Variable Weighting Clustering Algorithm for Multiview Data.*
WMCFS	*Weighted Multi-view Clustering with Feature Selection.*
WVVHCM	*Multi-View Hard C-Means With Automated Weighting Of Views And Variables.*

SUMÁRIO

1 INTRODUÇÃO

Agrupamento de dados é um problema recorrente em discussões de áreas como reconhecimento de padrões (BISHOP, 2006), bioinformática (HIGHAM; KALNA; KIBBLE, 2007), mineração de dados (KING, 2014), processamento de imagens (JOULIN; BACH; PONCE, 2010), entre outras. O objetivo principal do agrupamento de dados é dividir um conjunto de objetos em grupos fazendo com que os elementos dentro de um mesmo grupo possuam máxima semelhança e elementos que pertencem a grupos diferentes possuam mínima similaridade (JAIN, 2010).

Na literatura, algoritmos que realizam agrupamento de dados, ou *clustering*, podem ser divididos a partir das estruturas de agrupamento formadas para a solução do problema. Existem dois tipos de estruturas de agrupamento mais conhecidos, chamados de hierárquico e particional (KAUFMAN; ROUSSEEUW, 1990). Modelos hierárquicos resultam em uma estrutura classificatória hierárquica de agrupamentos - semelhante a um diagrama de árvore -, onde o nó raiz representa o grupo que contém todos os objetos e os nós finais, ou folhas, representam os grupos únitários, que possuem apenas um elemento. Modelos deste tipo ainda podem ser subdivididos em algoritmos aglomerativos ou divisivos, dependendo de como o diagrama de árvore é montado. Já os algoritmos classificados como particionais resultam em um agrupamento final com uma quantidade fixa de grupos (ou *clusters*). Para chegar nesse resultado, esses algoritmos constroem várias possibilidades de partições durante o seu processo de otimização e a partição ótima é escolhida a partir de um critério interno de seleção. Esse processo de escolha geralmente é feito através da otimização de uma função objetivo. Os métodos particionais podem ser subdivididos em *hard* (também chamados de algoritmos fixos, exclusivos ou rígidos) e *fuzzy* (também chamados de algoritmos difusos ou não exclusivos). Nos métodos do tipo *hard*, cada indivíduo é associado a somente um *cluster*, resultando em agrupamentos estrítos. Já nos algoritmos do tipo *fuzzy*, cada elemento pode pertencer a mais de um *cluster*, possuindo graus ou níveis de pertinência entre elementos e os grupos resultantes. O modelo apresentado neste trabalho pode ser classificado como um algoritmo particional do tipo fixo.

Hoje em dia, a maioria dos algoritmos de agrupamento de dados trabalham com elementos descritos por vários atributos que são reunidos em uma única visão ou tabela de dados. Porém, ao longo dos anos, com a crescente quantidade de informação que vem sendo armazenada e disponibilizada, a cada dia é mais comum notar a existência de diversas tabelas ou fontes de dados distintas que descrevem os mesmos objetos sobre diferentes visões com diferentes conjuntos de atributos. Dentro do contexto de *clustering*, esses conjuntos descritos por várias tabelas podem ser denominados conjuntos de dados *multi-view*.

Para tentar exemplificar melhor conjuntos de dados *multi-view*, podemos tomar como base um problema clássico da área de informática, onde os objetos são representados por páginas *web* que podem ser classificadas a partir dos seus conteúdos, como mostra a figura 1. Neste exemplo, imagens, textos e *hiperlinks* formam conjuntos de conteúdo que podem ser vistos como visões distintas. Essas visões podem gerar diferentes *clusters* a depender das perspectivas escolhidas e priorizadas durante o agrupamento.

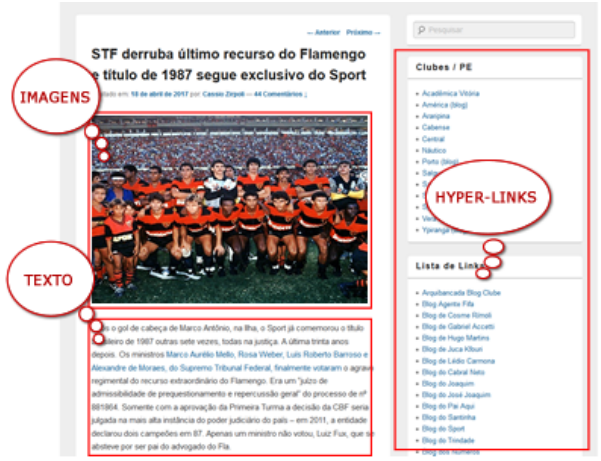


Figura 1 – Conjunto multi-view a partir de uma página web

Outro exemplo pode ser visto na área de biomedicina, onde diferentes tipos de informação podem ser obtidos dos pacientes, como imagens de ressonância magnética, dados de interação de proteínas, resultados de testes sanguíneos, dados genéticos, entre outros. Cada um desses conjuntos de dados pode ser tratado como uma visão distinta, podendo influenciar de forma diferente na classificação de grupos de pacientes.

É possível também obter visões distintas sabendo que informações que descrevem objetos podem ser extraídas levando em consideração diferentes pontos de vista. Por exemplo, a extração de atributos sobre cor e textura de um conjunto de imagens pode facilitar a classificação destes elementos. Com isso, cada perspectiva poderia ser tratada de uma forma mais individualizada durante o processo de agrupamento, ajudando na resolução de problemas de busca e classificação.

Na prática, a representação dos dados *multi-view* pode ser feita como um vetor (ou sequência) de matrizes, onde cada matriz equivale a uma visão, que comumente é a união de vários atributos que descrevem os elementos. Estes elementos são definidos pelas linhas das matrizes, enquanto os atributos são definidos pelas colunas. Desta forma, as diferentes matrizes (ou visões) possuem a mesma quantidade de linhas (que representam os elementos), mas podem ter diferentes quantidades de colunas (atributos dos elementos). Neste caso, o resultado da soma das quantidades das colunas pode ser considerada a dimensão do problema do agrupamento de dados. Uma forma visual de apresentar um conjunto de dados *multi-view* pode ser vista na figura 2, onde um conjunto de N elementos é

representado por V visões distintas. Todos os elementos são representados em todas as visões e cada visão pode ter uma dimensão diferente, ou seja, uma quantidade diferente de atributos.

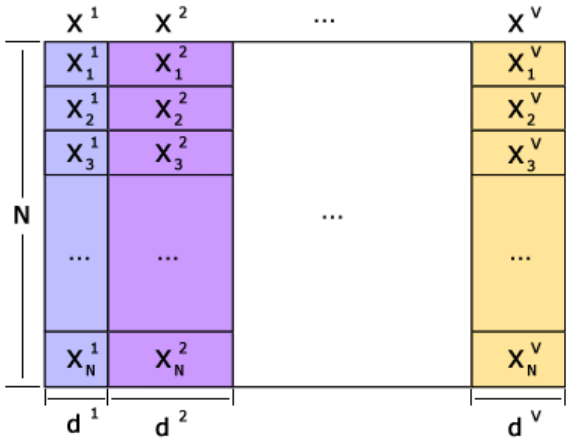


Figura 2 – Representação de um conjunto de dados multi-view

Atualmente já existem modelos de algoritmos que levam em consideração essas diferentes matrizes para chegar ao particionamento final. Esses modelos, que podem ser denominados de algoritmos de agrupamento de múltiplas visões, ou *multi-view clustering* (BICKEL; SCHEFFER, 2004), trabalham com o objetivo de agrupar os objetos levando em consideração a representação dos elementos pelas diferentes tabelas. Essas tabelas geralmente possuem vários atributos de diferentes tipos, que podem ser valores reais, binários, nominais (ou categóricos), ordinais, entre outros. Também é possível representar um atributo como uma combinação desses tipos.

Dentro do *multi-miew clustering* é comum encontrar três abordagens que podem ser utilizadas para solucionar problemas de agrupamento de dados:

- *Concatenação*: este método executa um tratamento prévio nas visões existentes, que pode ser desde um processo complexo de transformação de dados como simplesmente a concatenação dessas visões. O resultado obtido é uma única tabela que será usada para agrupar os indivíduos em uma partição final (TRIVEDI et al., 2010; YAMANISHI; VERT; KANEHISA, 2004).

- *Métodos distribuídos*: o objetivo deste método é agrupar as visões de forma independente. Esses agrupamentos podem ser construídos a partir de diferentes modelos. Os grupos resultantes fornecidos por cada visão são combinados, através de um método específico, para obter uma partição final (REZA et al., 2009; XIE; SUN, 2013).

- *Métodos centralizados*: são os métodos que, durante o processo de agrupamento de dados, levam em consideração todas as visões simultaneamente para realizar o particionamento dos elementos. Cada visão tem um peso de relevância específico que influenciará a formação dos grupos de indivíduos (TZORTZIS; LIKAS, 2009a).

No modelo centralizado, que foi o método mais estudado durante o trabalho de pesquisa realizado nesta dissertação, existem diferentes técnicas que levam em consideração as diferentes perspectivas dos elementos para realizar o agrupamento de dados *multi-view* de forma mais otimizada. Dentre elas, podemos citar técnicas que se baseiam na distância para cálculo dos centróides (CARVALHO; LECHEVALLIER; MELO, 2012; WANG; CHEN, 2016), em funções de kernel (TZORTZIS; LIKAS, 2012; SAEMAIL P. W. GALLAGHER; MALAVE, 2010) e em inteligência de enxames (WAHID; ANDREAE, 2015; JIANG F. QIU; WANG, 2016). Porém, geralmente essas técnicas consideram apenas os pesos de influência das diferentes visões durante o processo de particionamento. Alguns modelos mais recentes (CHEN X., 2013; XU; WANG; LAI, 2016), além de considerarem a influência das diferentes visões, também fazem uma seleção de quais atributos dentro dessas visões são mais pertinentes para o agrupamento dos dados. Entretanto, estes modelos mais recentes necessitam de parâmetros de entrada que só podem ser ajustados com um conhecimento prévio dos dados que serão agrupados. Esses parâmetros comumente servem para controle dos pesos das visões e dos atributos, fazendo com que as relevâncias calculadas automaticamente possam ser distribuídas de forma que o particionamento seja otimizado.

Este trabalho apresenta um algoritmo que, assim como as técnicas mais recentes de agrupamento centralizado de dados *multi-view*, realiza o particionamento de objetos levando em consideração as visões e os atributos destas visões durante o processo de agrupamento. O cálculo das relevâncias das visões e dos seus atributos é feito de forma automática e colaborativa, de forma que cada etapa realizada depende do resultado da etapa anterior. Diferente dos métodos existentes hoje em dia, o modelo proposto não necessita de nenhum conhecimento prévio dos dados ou ajuste de valor de parâmetros de entrada para que os resultados obtidos sejam competitivos. Desta forma, o único valor fornecido para o modelo proposto é a quantidade de *clusters* da partição.

1.1 Objetivo do Trabalho

O objetivo deste trabalho é apresentar um novo método particional do tipo *hard* que possui uma abordagem centralizada de agrupamento de dados *multi-view*. Neste novo modelo proposto, diferentes pesos são atribuídos às visões e às variáveis de cada visão durante o processo de agrupamento, de modo que as tabelas relevantes, bem como as variáveis relevantes em cada tabela, sejam selecionadas para geração de uma melhor partição. Podemos classificar o método proposto como um algoritmo de agrupamento exclusivo do tipo *c-means* para dados *multi-view* que computa automaticamente, e de forma colaborativa, os pesos das visões e os pesos das variáveis que irão influenciar na formação da partição final.

Alguns métodos em que as visões e as variáveis destas visões são levadas em consideração de forma automática durante o processo de agrupamento já foram apresentados

anteriormente. Porém, os algoritmos existentes desse tipo dependem de parâmetros de entrada que necessitam de um conhecimento prévio dos dados que serão agrupados. Até onde foi estudado neste trabalho, o algoritmo proposto nesta dissertação é o primeiro de seu tipo que, além da necessidade de saber previamente a quantidade de *clusters* da partição final, nenhum outro valor de parâmetro de entrada é necessário para seu funcionamento ideal.

1.1.1 Produção Bibliográfica

O modelo proposto neste trabalho foi submetido como artigo e aceito na *International Joint Conference on Neural Networks (IJCNN 2017)*, realizada entre os dias 14 e 19 de maio de 2017 em Anchorage, Alasca, Estados Unidos. O trabalho foi publicado nos anais da conferência pela editora da IEEE com DOI e difusão internacional. A mensagem de aceitação, a página *web* da IEEE com detalhes da publicação e a primeira página do artigo, extraída também da página *web* da IEEE, se encontram no apêndice.

1.2 Estrutura da Dissertação

Esta dissertação está organizada em 6 capítulos. Neste capítulo foi apresentada uma introdução ao agrupamento de dados e aos diferentes métodos existentes para solução desse problema. Além disso, foi explicada também a natureza dos dados *multi-view* e os tipos de métodos que trabalham com esses dados. O capítulo 2 apresenta um breve histórico sobre o agrupamento de dados *multi-view*, mostrando o surgimento de alguns modelos. Nele, são explicados os quatro algoritmos que foram selecionados para comparação de performance com o modelo proposto. São eles: o K-Means tradicional (MACQUEEN, 1967), o *Multi-view Kernel K-Means* (MVKKM) (TZORTZIS; LIKAS, 2012), o *Automated Two-Level Variable Weighting Clustering Algorithm for Multiview Data* (TW-K-Means) (CHEN X., 2013) e o *Weighted Multi-view Clustering with Feature Selection* (WMCFS) (XU; WANG; LAI, 2016). O capítulo 3 apresenta e detalha o modelo proposto, chamado de *Multi-View Hard C-Means With Automated Weighting Of Views And Variables* (WVVHCM). No capítulo 4, os experimentos realizados para validar esse novo modelo serão apresentados, sendo detalhados também a metodologia, os conjuntos de dados e as métricas utilizadas para avaliar os resultados. No capítulo 5, os resultados dos experimentos são apresentados e analisados, com um comparativo entre o algoritmo WVVHCM e os modelos relacionados. Por fim, o capítulo 6 apresenta a conclusão e trabalhos futuros.

2 MODELOS RELACIONADOS

Nos últimos anos, a pesquisa sobre agrupamento de conjuntos de dados descritos por múltiplas tabelas cresceu rapidamente, resultando no surgimento de diferentes modelos para solução desse problema. Alguns desses modelos serão apresentados brevemente neste capítulo. Também serão apresentados, com mais detalhes, quatro algoritmos que foram utilizados como comparativo de desempenho ao método proposto neste trabalho. Os algoritmos aqui descritos foram selecionados observando-se a relevância na evolução da seleção dos pesos das visões e das variáveis para solução do problema de agrupamento de dados.

A primeira abordagem que leva em consideração os pesos das visões e das variáveis ao mesmo tempo durante o agrupamento de dados foi proposta por DeSarbo et al. (1984). No entanto, o modelo computa automaticamente apenas os pesos das variáveis durante sua execução. Os pesos das visões são parâmetros de entrada dados pelo usuário, que necessita ter algum conhecimento prévio sobre os conjuntos de dados antes de executar o algoritmo.

O método apresentado em Carvalho, Lechevallier e Melo (2012) é capaz de agrupar objetos considerando diferentes visões que são representadas por matrizes de dissimilaridade (dados relacionais). Este método é capaz de levar em conta todas as visões, atribuindo-lhes pesos durante as etapas iterativas do algoritmo. O método também permite a consideração das visões de forma abrangente, onde cada visão terá o mesmo peso para todos os grupos, ou local, onde cada visão pode ter um peso diferente na formação de cada grupo. Porém, o método considera apenas os pesos das visões, não computando as relevâncias das variáveis que podem influenciar no agrupamento dos dados.

Os dois algoritmos propostos em Tzortzis e Likas (2012) ponderam as diferentes visões de atributos de acordo com as suas contribuições na partição resultante. Apesar desses algoritmos também considerarem a relevância de cada visão para agrupar os elementos, o desempenho dos modelos depende de um parâmetro de entrada que controla a distribuição dos pesos das tabelas que podem contribuir no processo de agrupamento. Não existe uma maneira simples de encontrar o valor de entrada desse parâmetro e nos experimentos do artigo mencionado o ajuste foi feito a partir do conhecimento prévio das classes dos elementos. No entando, no contexto de aprendizado não supervisionado, assume-se que os rótulos dos objetos não são conhecidos. Desta forma, a maneira utilizada pelos autores para ajustar o parâmetro só poderia ser realizada em um cenário de aprendizagem supervisionada. Um dos modelos apresentados pelos autores, denominado *Multi-view Kernel K-Means* (MVVKM), será um dos algoritmos utilizados para comparativo de performance com o modelo proposto neste trabalho.

O modelo apresentado em Wang, Nie e Huang (2013), apesar de não considerar as relevâncias das visões durante sua execução, propõe atribuir pesos a cada atributo de

cada visão localmente. Ou seja, um atributo específico de uma visão pode afetar de forma diferente a formação de cada *cluster*. Porém, como o método não considera que cada tabela também pode ter um peso de relevância independente que influencie a partição dos indivíduos, as visões que possuem muitos atributos com ruídos, por exemplo, podem prejudicar o agrupamento final dos dados.

Os modelos *Automated Two-Level Variable Weighting Clustering Algorithm for Multi-view Data* (TW-K-Means) e *Weighted Multi-view Clustering with Feature Selection* (WMCFS), que serão utilizados como comparativo de performance para o modelo proposto nesta dissertação, foram apresentados, respectivamente, por Chen X. (2013) e Xu, Wang e Lai (2016). Os autores propuseram algoritmos onde não apenas os pesos das visões são levados em consideração, mas a seleção das variáveis também é utilizada para agrupar os indivíduos. Nos dois casos, o cálculo das relevâncias das visões e das variáveis é feito de forma automática durante a execução do algoritmo. No entanto, os dois modelos precisam do conhecimento prévio de valores de dois parâmetros que devem ser ajustados de acordo com o conjunto de dados de entrada. Um dos parâmetros controla a dispersão dos pesos das visões e o outro controla a dispersão dos pesos das variáveis. Esses parâmetros não são fáceis de calcular, pois podem variar dentro de um amplo intervalo de valores e uma má escolha pode reduzir consideravelmente o desempenho do algoritmo. Além disso, assim como no modelo proposto por Tzortzis e Likas (2012), o ajuste dos valores desses parâmetros é feito quando a classificação dos indivíduos é previamente conhecida, como mostram os autores em seus respectivos artigos.

Três dos modelos mencionados, além do tradicional K-Means (MACQUEEN, 1967), foram selecionados como referência e serão descritos nas seções abaixo. A escolha foi feita com o objetivo de apresentar e comparar os resultados de diferentes abordagens para agrupar dados *multi-view*, além de avaliar a performance do algoritmo proposto neste trabalho. Desta forma, um dos modelos selecionados não leva em consideração a relevância dos atributos ou das visões durante o processo de agrupamento dos elementos. Outro modelo realiza o particionamento dos dados levando em consideração apenas os pesos das visões. Os dois modelos restantes consideram tanto a relevância das variáveis quanto a relevância das visões no processo de agrupamento, assim como o modelo proposto que será detalhado no próximo capítulo.

2.1 K-Means

O modelo K-Means (MACQUEEN, 1967) é um método bastante conhecido para agrupamento de dados. Sua ideia é dividir um conjunto de elementos em uma quantidade pré-determinada de *clusters*. Para isso, uma quantidade k de protótipos (ou centróides) ficam responsáveis por representar os k clusters do agrupamento. O centróide pode ser pensado como um ponto no centro de cada *cluster*, podendo possuir várias dimensões.

Cada dimensão representa um atributo do conjunto de dados. Entre todos os modelos utilizados nos experimentos deste trabalho, o K-Means é o único que funciona apenas com uma tabela de dados. Com isso, nos experimentos realizados para este modelo, os conjuntos de dados *multi-view* tiveram suas visões concatenadas, formando uma única tabela com todos os atributos.

No K-Means, os protótipos são otimizados de forma iterativa durante o processo de agrupamento, que minimiza uma função objetivo. Sendo g_k o centróide do *cluster* k da partição P e e_i o i-ésimo elemento de um conjunto de dados que possui m atributos, a equação que representa a função objetivo do modelo K-Means pode ser descrita como mostrado abaixo, onde x_{ij} é o valor do m-ésimo atributo do i-ésimo elemento.

$$J_{k-means} = \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{j=1}^{m} (x_{ij} - g_{kj})^2 \tag{2.1}$$

É possível notar na equação acima que não existem pesos para atributos ou visões de atributos do conjunto de dados. Logo, todas as variáveis possuem a mesma relevância para definir a partição final do modelo. Os passos necessários para a execução do K-Means estão resumidos no algoritmo 1

Algorithm 1 K-Means

1: **Inicialização:**
2: Escolha aleatória dos protótipos dentre os elementos de entrada
3: **Iterar:**
4: Cálculo dos protótipos;
5: Atualização dos clusters;
6: **Até:**
7: Nenhum elemento mudar de *cluster*
8: **Saída:**
9: Protótipos;
10: Partição final;

Por não necessitar de qualquer outro cálculo e sua inicialização ter baixo custo computacional, a complexidade computacional total do K-Means é $\mathcal{O}(rKnm)$, com n representando a quantidade de elementos do conjunto de dados e r a quantidade de iterações necessárias para convergência do modelo.

2.2 Multi-view Kernel K-Means

O MVKKM (TZORTZIS; LIKAS, 2009a) é um modelo baseado em função de *Kernel* que calcula as relevâncias das visões durante o processo de agupamento. Por possuir uma performance bastante significativa no agrupamento de dados descritos por diversas tabelas, o MVKKM tem sido bastante utilizado como referência em comparação com outros

modelos (CALDERS et al., 2014; XU; WANG; LAI, 2016; LINTAS et al., 2017). Para o bom funcionamento do algoritmo, é necessário informar previamente um parâmetro de entrada, chamado de p pelos autores. Esse parâmetro irá controlar a dispersão dos pesos das relevâncias das visões.

Por ser baseado em função de *Kernel*, a ideia do algoritmo é mapear o conjunto de dados inicial, transformando o espaço original em um espaço de maior dimensão, chamado de espaço de características (ou *feature space*). O uso desse procedimento é motivado pelo teorema de Cover (HAYKIN, 1999). O teorema diz que dado um conjunto de dados que não é linearmente separável no espaço de entrada, esse espaço pode ser transformado em um espaço de características com maior dimensão no qual, com alta probabilidade, os dados serão linearmente separáveis. Para exemplificar, pode-se definir $\phi : X \rightarrow \mathcal{F}$ como um mapeamento não linear do conjunto de dados de entrada X em um *feature space* de maior dimensão $\mathcal{F}$. Logo, o produto interno $x_i^\intercal x_k$ do espaço original pode ser mapeado para $\phi(x_i)^\intercal \phi(x_k)$ no espaço de características. A ideia chave de um algoritmo de *kernel* é que o mapeamento não linear ϕ não precisa ser explicitamente especificado, pois cada *Mercer kernel* pode ser definido como $K(x_i, x_k) = \phi(x_i)^\intercal \phi(x_k)$. Essa operação, que usualmente é chamada de *kernel trick* (SCHOLKOPF; SMOLA; MULLER, 1998; FERREIRA; CARVALHO, 2014), possibilita o cálculo da distância Euclidiana no espaço $\mathcal{F}$ a partir de:

$$\begin{aligned} ||\phi(x_i) - \phi(x_k)||^2 &= (\phi(x_i) - \phi(x_k))^\intercal(\phi(x_i) - \phi(x_k)) \\ &= \phi(x_i)^\intercal\phi(x_i) - 2\phi(x_i)^\intercal\phi(x_k) + \phi(x_k)^\intercal\phi(x_k) \\ &= K(x_k, x_k) - 2K(x_k, x_i) + K(x_i, x_i) \end{aligned} \tag{2.2}$$

Com isso, o uso da função de *kernel* permite operar em um espaço de maior dimensão a partir do cálculo dos produtos internos entre as projeções dos elementos de entrada no espaço de características, fazendo com que não haja a necessidade da transformação desses elementos de entrada em coordenadas no espaço de maior dimensão. No caso do MVVKM, o *kernel* utilizado nos experimentos foi o Gaussiano, que pode mapear os dados de entrada em um espaço de dimensão infinita.

Nesse modelo, para realizar o particionamento dos N elementos de um conjunto de dados com V visões de atributos de entrada em M *clusters*, e simultaneamente explorar essas visões para obter matrizes de *Kernel* que satisfazem a equação 2.3, a função objetivo descrita em 2.4 deve ser minimizada.

$$K^{(V)} = \sum_{v=1}^{V} w_v^p K^{(v)}, w_v \geq 0, \sum_{v=1}^{V} w_v = 1, p \geq 1 \tag{2.3}$$

$$J_{MVKKM} = \sum_{v=1}^{V} w_v^p \sum_{i=1}^{N}\sum_{k=1}^{M} \delta_{ik} ||\phi^{(v)}(x_i^{(v)}) - m_k^{(v)}||^2 \tag{2.4}$$

Onde $m_k^{(v)}$ é o protótipo do *cluster* k para a visão v no *feature space*, que tem como equação:

$$m_k^{(v)} = \frac{\sum_{i=1}^{N} \delta_{ik} \phi^{(v)}(x_i^{(v)})}{\sum_{i=1}^{N} \delta_{ik}} \tag{2.5}$$

Como explicado anteriormente, com o uso do *kernel trick*, não é necessário encontrar os valores dos protótipos no espaço de maior dimensão para se calcular a distância Euclidiana da equação 2.4. Utilizando 2.2, Esse cálculo pode ser feito como mostrado em 2.6.

$$||\phi^{(v)}(x_i^{(v)}) - m_k^{(v)}||^2 = K_{ii}^{(v)} - \frac{2\sum_{j=1}^{N} \delta_{jk} K_{ij}^{(v)}}{\sum_{j=1}^{N} \delta_{jk}} + \frac{\sum_{j=1}^{N}\sum_{l=1}^{N} \delta_{jk}\delta_{lk} K_{jl}^{(v)}}{\sum_{j=1}^{N}\sum_{l=1}^{N} \delta_{jk}\delta_{lk}} \tag{2.6}$$

Para o cálculo dos pesos das visões, a seguinte equação deve ser utilizada:

$$w_v = \frac{1}{\sum_{v'=1}^{V} \left(\frac{D_v}{D_{v'}}\right)^{\frac{1}{p-1}}} \tag{2.7}$$

onde:

$$D_v = \sum_{i=1}^{N}\sum_{k=1}^{M} \delta_{ik} ||\phi^{(v)}(x_i^{(v)}) - m_k^{(v)}||^2 \tag{2.8}$$

Nas equações acima, $x_i^{(v)}$ é a representação do elemento i na visão v; $m_k^{(v)}$ é a representação do protótipo k na visão v; ϕ representa a transformação do conjunto de entrada em um espaço de maior dimensão; w_v é o peso da relevância da visão v; enquanto K^v é a matriz de *kernel* da visão v. A variável δ_i pode assumir os valores 1 ou 0, indicando, respectivamente, se um elemento i pertence ou não pertence a um determinado *cluster*. Durante o processo de particionamento, p deve ser mantido fixo e seu valor irá influenciar na esparsidade dos pesos das relevâncias das visões de atributos.

Para a execução do modelo MVKKM, o autor sugere que os pesos das visões sejam inicializados com o mesmo valor $(1/V)$ e que a solução inicial do modelo seja fornecida pela saída do *Global Kernel K-Means* (TZORTZIS; LIKAS, 2009b), uma extensão do algoritmo K-Means que identifica *clusters* não linearmente separáveis. Após a inicialização, o algoritmo alterna entre duas etapas. A primeira serve para atualizar os grupos de acordo com os pesos das visões - nesta etapa é aplicado o *Kernel* K-Means (SCHOLKOPF; SMOLA; MULLER, 1998), uma generalização do K-Means, para realizar a tarefa. A segunda etapa serve para atualizar os pesos das visões de acordo com os grupos. O autor não indica um critério de parada em seu artigo e neste trabalho as iterações foram feitas até que nenhum elemento mude de *cluster* entre duas iterações subsequentes. Os passos necessários para execução do MVKKM estão resumidos no algoritmo 2.

Em seu artigo, o autor ainda indica a convergência do modelo e apresenta a complexidade computacional de $\mathcal{O}(N^2(V+\tau)\tau' + N^3 M\tau)$, onde τ' e τ são, respectivamente, as iterações de MVKKM e do *Kernel* K-Means até a convergência dos algoritmos.

Algorithm 2 MVKKM

1: **Inicialização:**
2: Inicialização dos pesos das visões como $1/V$
3: Execução do *global kernel k-means algorithm*
4: **Iterar:**
5: Atualização dos clusters;
6: Cálculo dos pesos das visões;
7: **Até:**
8: Nenhum elemento mudar de *cluster*
9: **Saída:**
10: Vetor com os pesos das visões;
11: Partição final;

2.3 Automated Two-Level Variable Weighting Clustering Algorithm for Multiview Data

O TW-K-Means (CHEN X., 2013) é um dos algoritmos escolhidos que leva em consideração as relevâncias dos pesos das visões e dos pesos das variáveis de um conjunto de dados *multi-view* no particionamento dos elementos. Esses pesos são calculados de forma automática para cada conjunto de dados durante a execução do algoritmo.

Segundo os autores, o TW-K-Means pode ser definido como uma extensão do modelo K-Means, onde um processo semelhante é utilizado para realizar o agrupamento dos dados. Porém, neste novo processo, são adicionados dois novos passos em cada iteração para calcular os pesos das visões e das variáveis. Os pesos encontrados são incorporados no cálculo da distância na função objetivo para diferenciar os impactos das diferentes tabelas e atributos do conjunto de dados.

No TW-K-Means, os pesos das relevâncias das visões refletem a importância de cada visão no conjunto de dados completo, enquanto os pesos encontrados para as variáveis refletem a importância de cada variável dentro da visão que a possui. Neste modelo são necessários dois parâmetros de entrada nomeados de λ e η pelos autores. O objetivo desses parâmetros é controlar a distribuição dos pesos entre as visões (controladas pelo parâmetro λ) e as variáveis (controladas pelo parâmetro η) existentes.

Logo, sendo K a quantidade final de *clusters* de uma partição, para realizar o agrupamento de N elementos de um conjunto de dados com V visões de atributos de entrada, o modelo TW-K-Means minimiza a função objetivo definida abaixo para variáveis numéricas:

$$J_{TW-K-Means} = \sum_{l=1}^{K}\sum_{i=1}^{N}\sum_{t=1}^{V}\sum_{j\in V_t} u_{il} w_t f_j (x_{ij} - g_{lj})^2 + \eta \sum_{j=1}^{m} f_j \log f_j + \lambda \sum_{t=1}^{V} w_t \log w_t \quad (2.9)$$

Esta equação está sujeita às restrições:

- $\sum_{l=1}^{K} u_{il} = 1, u_{il} \in \{0,1\}, 1 \leq i \leq n$
- $\sum_{t=1}^{T} w_t = 1, 0 \leq w_t \leq 1$
- $\sum_{j \in V_t} f_j = 1, 0 \leq f_j \leq 1, 1 \leq t \leq V$

Para o cálculo dos pesos das visões a equação 2.10 deve ser utilizadas:

$$w_t = \frac{\exp\left(\frac{-D_t}{\lambda}\right)}{\sum_{h}^{V} \exp\left(\frac{-D_h}{\lambda}\right)} \tag{2.10}$$

Onde:

$$D_t = \sum_{l=1}^{K} \sum_{i=1}^{N} \sum_{j \in V_t} u_{il} f_j (x_{ij} - g_{lj})^2 \tag{2.11}$$

Já para o cálculo dos pesos das variáveis, a equações 2.12 deve ser utilizada.

$$f_j = \frac{\exp\left(\frac{-E_j}{\eta}\right)}{\sum_{h \in V_t} \exp\left(\frac{-E_h}{\eta}\right)} \tag{2.12}$$

Onde:

$$E_j = \sum_{l=1}^{K} \sum_{i=1}^{N} u_{il} w_t (x_{ij} - g_{lj})^2 \tag{2.13}$$

Nas equações acima, m indica a soma das quantidades de variáveis de todas as visões, ou seja, a quantidade total de variáveis do conjunto de dados; x_{ij} é o valor da variável j da visão V_t do elemento i; g_{lj} é o valor da variável j da visão V_t do protótipo (ou centróide) do *cluster* l; w_t representa o peso da relevância da visão t; enquanto f_j representa o peso da relevância da variável j na visão V_t. A variável u_{il} pode assumir os valores 1 ou 0, indicando, respectivamente, se o elemento i pertence ou não pertence ao *cluster* l. Durante o processo de agrupamento, λ e η devem ser mantidos fixos e seus valores devem ser previamente conhecidos.

Para a execução do modelo, o autor indica que inicialmente os pesos devem possuir os mesmos valores para cada visão e os mesmos valores para cada variável dentro de cada visão. Ou seja, atribuir $1/|V_t|$ para os pesos das variáveis em cada visão t e $1/V$ para os pesos das visões. Após essa inicialização, o algoritmo executa as seguintes etapas de forma iterativa: atualizar os *clusters*, atualizar os protótipos dos *clusters*, atualizar os pesos das variáveis e atualizar os pesos das visões. O autor indica que o critério de parada é quando a função objetivo atinge o mínimo local. Neste trabalho as iterações foram feitas até que nenhum elemento mude de *cluster* entre duas iterações subsequentes. Os passos necessários para execução do TW-K-Means estão resumidos no algoritmo 3.

Em seu artigo, o autor ainda indica a convergência do modelo e apresenta a complexidade computacional da função objetivo como sendo a mesma do algoritmo K-Means:

Algorithm 3 TW-K-Means

Inicialização:
    Inicialização dos pesos das visões como $1/V$
    Inicialização dos pesos das variáveis como $1/|V_t|$
    Escolha aleatória dos protótipos dentre os elementos de entrada
Iterar:
    Atualização dos clusters
    Cálculo dos protótipos dos clusters
    Cálculo dos pesos das variáveis
    Cálculo dos pesos das visões
Até:
    Nenhum elemento mudar de *cluster*
Saída:
    Vetor com os pesos das visões;
    Vetores com os pesos das variáveis;
    Protótipos;
    Partição final;

$\mathcal{O}(rKNm)$, onde r é a quantidade de iterações realizadas até a convergência do algoritmo. Porém, o autor não considera o custo computacional para os cálculos dos pesos das variáveis e das visões, que devem ser consideradas na complexidade total do algoritmo. Levando em consideração essas duas etapas, o modelo TW-K-Means tem uma complexidade total de $\mathcal{O}((m + V^2)rKN \log(m))$.

2.4 Weighted Multi-view Clustering with Feature Selection

Assim como o algoritmo apresentado na seção anterior, o modelo WMCFS calcula de forma automática os pesos das variáveis e os pesos das visões durante o agrupamento dos dados *multi-view*. Esses pesos influenciam na partição final encontrada pelo algoritmo com o objetivo de aprimorar os *clusters* resultantes.

Assim como todos os algoritmos apresentados que levam em consideração os pesos das visões e/ou pesos das variáveis, o WMCFS também necessita de parâmetros de entrada que devem ser informados pelo usuário antes do processo de particionamento dos dados. Os parâmetros nomeados pelos autores de p e β são responsáveis, respectivamente, pela esparsidade dos pesos das visões e das variáveis durante a execução do modelo, ou seja, controlam a quantidade de visões e variáveis que influenciarão de forma mais significativa o processo de agrupamento de dados. Assim como nos modelos anteriores, os autores não indicam uma maneira não supervisionada para escolha dos valores corretos desses parâmetros, que influenciam fortemente a performance dos algoritmos. Logo, é preciso ter um conhecimento prévio dos conjuntos de dados que serão agrupados antes da execução

do modelo.

Para agrupar um conjunto de elementos *multi-view* $E = \{e_1, ..., e_i, ...e_n\}$ com V visões de atributos de entrada em K *clusters*, o modelo WMCFS deve minimizar a função objetivo definida abaixo:

$$J_{WMCFS} = \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{l=1}^{V} (\pi_l)^p \sum_{j=1}^{|V_l|} \lambda_{lj} (x_{ilj} - g_{klj})^2 + \beta \sum_{l=1}^{s} \sum_{j=1}^{|V_l|} (\lambda_{lj})^2 \tag{2.14}$$

Essa equação está sujeita às restrições:

- $\sum_{l=1}^{V} \pi_l = 1, 0 \leq \pi_l \leq 1$
- $\sum_{j=1}^{|V_l|} \lambda_{lj} = 1, 0 \leq \lambda_{lj} \leq 1, 1 \leq l \leq V$

Para o cálculo dos pesos das visões a equação 2.15 deve ser utilizada:

$$\pi_l = \frac{1}{\sum_{l'=1}^{p_l} \left(\frac{D_l}{D_l'} \right)^{1/p-1}} \tag{2.15}$$

Onde:

$$D_l = \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{j=1}^{|V_l|} \lambda_{lj} (x_{ilj} - g_{klj})^2 \tag{2.16}$$

Já para o cálculo dos pesos das variáveis, a equação 2.17 deve ser utilizada.

$$\lambda_{lj} = \frac{1}{\sum_{j'=1}^{|V_l|} \frac{B_j^l}{B_{j'}^l}} \tag{2.17}$$

Onde:

$$B_j^l = \beta + (\pi_l)^p \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{l=1}^{V} (x_{ilj} - g_{klj})^2 \tag{2.18}$$

Nas equações acima, $|V_l|$ indica a quantidade de variáveis da visão l; x_{ilj} é o valor da variável j da visão l do elemento i; g_{klj} é o valor da variável j da visão l do protótipo (ou centróide) do *cluster* k; π_l representa o peso da relevância da visão l; enquanto λ_{lj} representa o peso da relevância da variável j da visão l. Durante o processo de agrupamento, p e β devem ser mantidos fixos e seus valores devem ser previamente conhecidos.

Para a execução do modelo, o autor indica que inicialmente os pesos devem possuir os mesmos valores para cada visão e os mesmos valores para cada variável dentro de cada visão. Ou seja, deve ser atribuído o valor $1/|V_l|$ para os pesos das variáveis em cada visão l e o valor $1/V$ para os pesos das visões. Após essa inicialização, o algoritmo executa as seguintes etapas de forma iterativa: atualizar os *clusters* e seus protótipos, atualizar os pesos das visões e atualizar os pesos das variáveis. O autor indica que o critério de parada acontece quando a função objetivo atinge a convergência ou quando a execução

atinge uma quantidade máxima de iterações. Neste trabalho, as iterações foram feitas até que nenhum elemento mude de *cluster* entre duas iterações subsequentes. Os passos necessários para execução do WMCFS estão resumidos no algoritmo 4.

Algorithm 4 WMCFS

Inicialização:
    Inicialização dos pesos das visões como $1/V$
    Inicialização dos pesos das variáveis como $1/|V_l|$
    Escolha aleatória dos protótipos dentre os elementos de entrada
Iterar:
    Atualização dos clusters
    Cálculo dos protótipos dos clusters
    Cálculo dos pesos das visões
    Cálculo dos pesos das variáveis
Até:
    Nenhum elemento mudar de *cluster*
Saída:
    Vetor com os pesos das visões;
    Vetores com os pesos das variáveis;
    Protótipos;
    Partição final;

Em seu artigo, o autor ainda indica a convergência do modelo, porém não apresenta a complexidade computacional. Assim como no modelo TW-K-Means, o WMCFS possui a complexidade computacional de $\mathcal{O}((m + V^2)rKn\log(m))$, onde n é a quantidade de elementos do conjunto de dados, m é a soma das quantidades de variáveis de todas as visões e r é a quantidade de iterações realizadas até a convergência do algoritmo.

Como foi visto neste capítulo, os modelos selecionados para comparação de performance com o modelo proposto possuem diferentes características. Um dos modelos não leva em consideração a relevância dos atributos ou das visões. Outro modelo realiza o agrupamento dos dados levando em consideração os pesos apenas das visões. Os outros dois modelos consideram tanto a relevância das variáveis quanto a relevância das visões no processo de agrupamento, se diferenciando em suas funções objetivo. As tabelas 2 e 3 apresentam, respectivamente, as funções objetivo e um resumo das diferentes características dos quatro modelos selecionados para comparação de performance com o modelo proposto que será detalhado no próximo capítulo.

Tabela 2 – Funções objetivo dos modelos relacionados

Modelo	**Função objetivo**
K-Means	$J = \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{j=1}^{m} (x_{ij} - g_{kj})^2$
MVKKM	$J = \sum_{v=1}^{V} w_v^p \sum_{i=1}^{N} \sum_{k=1}^{M} \delta_{ik} \|\|\phi^{(v)}(x_i^{(v)}) - m_k^{(v)}\|\|^2$
TW-K-Means	$J = \sum_{l=1}^{K} \sum_{i=1}^{N} \sum_{t=1}^{V} \sum_{j \in V_t} u_{il} w_t f_j (x_{ij} - g_{lj})^2 + \eta \sum_{j=1}^{m} f_j \log f_j + \lambda \sum_{t=1}^{V} w_t \log w_t$
WMCFS	$J = \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{l=1}^{V} (\pi_l)^p \sum_{j=1}^{\|V_l\|} \lambda_{lj} (x_{ilj} - g_{klj})^2 + \beta \sum_{l=1}^{s} \sum_{j=1}^{\|V_l\|} (\lambda_{lj})^2$

Tabela 3 – Resumo das características dos modelos relacionados

Modelo	**Parâmetros de entrada**	**Ponderação**
K-Means	- Quantidade de clusters	Nenhuma
MVKKM	- Quantidade de clusters - Parâmetro que controla os pesos das visões	Visões
TW-K-Means	- Quantidade de clusters - Parâmetro que controla os pesos das visões - Parâmetro que controla os pesos das variáveis	Visões e Variáveis
WMCFS	- Quantidade de clusters - Parâmetro que controla peso das visões - Parâmetro que controla os pesos das variáveis	Visões e Variáveis

3 MODELO PROPOSTO

Este capítulo irá descrever o modelo *Multi-View Hard C-Means With Automated Weighting Of Views And Variables* (WVVHCM), que é o algoritmo proposto neste trabalho e que tem por objetivo agrupar elementos que possuem informações descritas por múltiplas visões. Será mostrado que a partir de uma solução inicial, onde o conjunto de entrada possui seus dados descritos em diferentes tabelas, é possível calcular, de forma automática e colaborativa, durante o processo de agrupamento, os pesos das tabelas e os pesos das variáveis de cada tabela. Esses pesos influenciam na formação da partição final dos elementos, fazendo com que uma determinada variável ou visão tenha mais ou menos relevância para o algoritmo.

O funcionamento do modelo WVVHCM tem como base a otimização de uma função objetivo, que é feita de forma iterativa. Cada iteração possui quatro etapas que serão apresentadas com detalhes. Como resultado do modelo, os elementos de entrada são divididos em diferentes grupos, onde cada elemento pertence exclusivamente a um grupo. Além do agrupamento, o algoritmo gera como saída os protótipos (ou centróides) dos *clusters*, os vetores de pesos das visões e os vetores de pesos das variáveis das visões.

Para que seja possível explicar o WVVHCM, é preciso definir $E = \{e_1, \ldots, e_n\}$ como um conjunto de n indivíduos que são descritos a partir de s tabelas: $\mathbf{X}_1, \ldots, \mathbf{X}_l, \ldots, \mathbf{X}_s$. Com isso, cada indivíduo e_i é descrito por um vetor $\mathbf{x}_i = (\mathbf{x}_{i1}, \ldots, \mathbf{x}_{il}, \ldots, \mathbf{x}_{is})$; onde o componente $\mathbf{x}_{il} = (x_{il1}, \ldots, x_{ilj}, \ldots, x_{ilp_l})\,(l = 1, \ldots, s)$ é a descrição do elemento e_i na tabela $\mathbf{X}_l$ pelas variáveis $X_{il}, \ldots, X_{il}, \ldots, X_{il}$ com $x_{ilj} \in \mathtt{R}$ $(j = 1, \ldots, p_l)$. Logo, $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ é o conjunto de dados de n elementos. A tabela 4 abaixo mostra o conjunto de dados de entrada e seus atributos de forma mais visual:

Tabela 4 – Conjunto de entrada com múltiplas tabelas e variáveis

	Tabela $\mathbf{X}_1$	...	Tabela $\mathbf{X}_l$	...	Tabela $\mathbf{X}_s$
E	$X_{11}, \ldots, X_{1j}, \ldots, X_{1p_1}$	...	$X_{l1}, \ldots, X_{lj}, \ldots, X_{lp_l}$	...	$X_{s1}, \ldots, X_{sj}, \ldots, X_{sp_s}$
e_1	$x_{111} \ldots x_{11j} \ldots x_{11p_1}$	...	$x_{1l1} \ldots x_{1lj} \ldots x_{1lp_l}$	...	$x_{1s1} \ldots x_{1sj} \ldots x_{1sp_s}$
...	...	...	...	...	...
e_i	$x_{i11} \ldots x_{i1j} \ldots x_{i1p_1}$	...	$x_{il1} \ldots x_{ilj} \ldots x_{ilp_l}$	...	$x_{is1} \ldots x_{isj} \ldots x_{isp_s}$
...	...	...	...	...	...
e_n	$x_{n11} \ldots x_{n1j} \ldots x_{n1p_1}$	...	$x_{nl1} \ldots x_{nlj} \ldots x_{nlp_l}$	...	$x_{ns1} \ldots x_{nsj} \ldots x_{nsp_s}$

A partir da descrição dos elementos de entrada explicados acima, o algoritmo WVVHCM fornece como resultado:

- Uma partição $\mathcal{P} = (P_1, \ldots, P_K)$ dos E elementos em K *clusters*;
- Um vetor de relevância dos pesos da tabelas $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_l, \ldots, \pi_s)$ onde o componente π_l é a relevância do peso da tabela $\mathbf{X}_l$;
- Um vetor $\boldsymbol{\Lambda} = (\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_l, \ldots, \boldsymbol{\lambda}_s)$ onde o componente $\boldsymbol{\lambda}_l = (\lambda_{l1}, \ldots, \lambda_{lj}, \ldots, \lambda_{lp_l})$ é um vetor de relevância dos pesos das variáveis da tabela $\mathbf{X}_l$, com λ_{lj} sendo a relevância do peso da j-ésima variável da l-ésima tabela $\mathbf{X}_l$;
- Um vetor $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_k, \ldots, \mathbf{g}_K)$ onde o componente $\mathbf{g}_k = (\mathbf{g}_{k1}, \ldots, \mathbf{g}_{kl}, \ldots, \mathbf{g}_{ks})$ é o representante (protótipo) do *cluster* P_k, com $\mathbf{g}_{kl} = (g_{kl1}, \ldots, g_{klj}, \ldots, g_{klp_l})$ sendo o l-ésimo componente do protótipo $\mathbf{g}_k$ na tabela $\mathbf{X}_l$.

A partir de uma solução inicial, o vetor de protótipos $\mathbf{G}$, o vetor de relevância dos pesos das tabelas $\boldsymbol{\Lambda}$, o vetor de relevância de pesos das variáveis $\boldsymbol{\pi}$ e a partição de elementos $\mathcal{P}$ são obtidos de forma interativa nas quatro etapas do algoritmo (representação, peso das variáveis, peso das tabelas e alocação). Chega-se nesse resultado a partir da minimização de uma função objetivo que representa a heterogenidade total da partição.

Para facilitar a compreensão das equações e demonstrações existentes neste capítulo, a tabela 5 resume os símbolos e as definições necessárias ao modelo proposto.

Tabela 5 – Símbolos utilizados para descrever o algoritmo WVVHCM

Símbolo	Definição
E	O conjunto dos elementos que serão agrupados
n	A quantidade de elementos do conjunto
e_i	O i-ésimo elemento do conjunto
$\mathbf{X}_l$	A l-ésima tabela que descreve os elementos
s	A quantidade de tabelas que descrevem os elementos
$\mathbf{x}_i$	O vetor de tabelas que descrevem o i-ésimo elemento
$\mathbf{x}_{il}$	O vetor das variáveis do i-ésimo elemento na l-ésima tabela
$\mathbf{x}_{ilj}$	O valor da j-ésima variável do i-ésimo elemento na l-ésima tabela
p_l	A quantidade de variáveis da l-ésima tabela
$\mathcal{P}$	A partição final dos elementos
K	A quantidade de *clusters* da partição final
P_k	O k-ésimo *cluster* da partição final
$\boldsymbol{\pi}$	O vetor de relevância dos pesos das s tabelas que descrevem os elementos
π_l	A relevância do peso da l-ésima tabela que descreve os elementos
$\boldsymbol{\Lambda}$	O vetor de vetores das relevâncias dos pesos das variáveis das tabelas
$\boldsymbol{\lambda}_\mathbf{l}$	O vetor das relevâncias dos pesos das variáveis da l-ésima tabela
λ_{lj}	A relevância do peso da j-ésima variável da l-ésima tabela
$\mathbf{G}$	O vetor de protótipos dos *clusters*
$\mathbf{g}_k$	O vetor que representa o protótipo do k-ésimo *cluster*
$\mathbf{g_{kl}}$	O vetor que representa a l-ésima tabela do protótipo do k-ésimo *cluster*
$\mathbf{g_{klj}}$	O valor que representa a j-ésima variável da l-ésima tabela do protótipo do k-ésimo *cluster*

3.1 Função Objetivo

Os passos de otimização da solução gerada pelo modelo WVVHCM são feitos a partir da minimização da função objetivo denominada J_{WVVHCM}. Esta função calcula a heterogenidade total da partição somando as heterogenidades dos *clusters*. Sendo assim, é possível encontrar o agrupamento de dados mais homogêneo a partir da minimização da soma das dissimilaridades entre os elementos e os protótipos dos seus respectivos *cluters*. Esse cálculo é feito levando em consideração as ponderações das visões e variáveis existentes no conjunto de dados. A equação J_{WVVHCM} responsável por esse cálculo está definida em

3.1:

$$J_{WVVHCM} = \sum_{k=1}^{K} \sum_{e_i \in P_k} d_{(\boldsymbol{\pi},\boldsymbol{\Lambda})}(\mathbf{x}_i, \mathbf{g}_k) \tag{3.1}$$

Onde $d_{(\boldsymbol{\pi},\boldsymbol{\Lambda})}(\mathbf{x}_i, \mathbf{g}_k)$ representa a dissimilaridade (ou distância) entre a descrição $\mathbf{x}_i$ do elemento $\mathbf{e}_i$ e o protótipo $\mathbf{g}_k$ do *cluster* P_k. Essa dissimilaridade pode ser calculada como a soma ponderada da distância da representação do elemento $\mathbf{e}_i$ na tabela $\mathbf{X}_l$, denominado $\mathbf{x}_{il}$, e a representação do protótipo $\mathbf{g}_k$ na tabela $\mathbf{X}_l$, denominado $\mathbf{g}_{kl}$. Nesses casos, a ponderação é dada pela relevância da tabela $\mathbf{X}_l$, descrita como π_l. Com isso, a distância $d_{(\boldsymbol{\pi},\boldsymbol{\Lambda})}(\mathbf{x}_i, \mathbf{g}_k)$ pode ser escrita como:

$$d_{(\boldsymbol{\pi},\boldsymbol{\Lambda})}(\mathbf{x}_i, \mathbf{g}_k) = \sum_{l=1}^{s} \pi_l d_{\boldsymbol{\lambda}_l}(\mathbf{x}_{il}, \mathbf{g}_{kl}) \tag{3.2}$$

Considerando uma ideia semelhante a de ponderação dos pesos das tabelas, a distância $d_{\boldsymbol{\lambda}_l}(\mathbf{x}_{il}, \mathbf{g}_{kl})$ pode ser calculada pela equação 3.3 abaixo, onde $\mathbf{x}_{ilj}$ e $\mathbf{g}_{klj}$ são, respectivamente, os valores da j-ésima variável do elemento $\mathbf{e}_i$ e do protótipo $\mathbf{g}_k$ na tabela $\mathbf{X}_l$. A relevância dessa variável é descrita como λ_{lj}.

$$d_{\boldsymbol{\lambda}_l}(\mathbf{x}_{il}, \mathbf{g}_{kl}) = \sum_{j=1}^{p_l} \lambda_{lj}(x_{ilj} - g_{klj})^2 \tag{3.3}$$

Finalmente, a equação que representa a função objetivo J_{WVVHCM} pode ser reescrita como:

$$J_{WVVHCM} = \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{l=1}^{s} \pi_l \sum_{j=1}^{p_l} \lambda_{lj}(x_{ilj} - g_{klj})^2 \tag{3.4}$$

3.1.1 Restrições

Uma solução trivial, porém não adequada ao objetivo do modelo, seria considerar que a relevância dos pesos das variáveis ou das tabelas descritas na equação 3.4 seriam sempre iguais a 0 (zero), ou seja:

- $\pi_l = 0$, $\forall l = 1$ até s
- $\lambda_{lj} = 0$, $\forall l = 1$ até s e $\forall j = 1$ até p_l

Para evitar essas combinações de valores não adequados ao modelo, é necessário introduzir restrições que impeçam esse tipo de solução. Isso geralmente é feito utilizando restrições matemáticas baseadas na multiplicação ou na adição dos valores que devem ser restringidos. Muitas soluções existentes, a exemplo de alguns algoritmos utilizados como modelos concorrentes neste trabalho, adotam a abordagem que faz a restrição dos valores baseada na soma. Porém, esse tipo de abordagem acaba adicionando novos parâmetros

que precisam ser calculados para realizar o processo de agrupamento. Esses parâmetros não são simples de estimar. Por exemplo, o modelo WMCFS, que tem a função objetivo descrita pela equação 2.14 apresentada no capítulo anterior, possui as seguintes restrições baseadas na soma:

- $\sum_{l=1}^{s} \pi_l = 1$, $\pi_l \geq 0$
- $\sum_{j=1}^{p_l} \lambda_{lj} = 1$, $\lambda_{lj} \geq 0$, $\forall l = 1$ até s

Como pode ser visto em Xu, Wang e Lai (2016), essas restrições acabam por inserir dois novos parâmetros ao modelo WMCFS: β e p. Assim como mostrado nos experimentos conduzidos pelos próprios autores, esses parâmetros não são simples de calcular, dependem diretamente do conjunto de dados que serão utilizados e possuem um intervalo relativamente grande de possíveis valores.

No modelo proposto nesse trabalho, foi utilizada uma restrição baseada na multiplicação dos valores, como feito em Carvalho, Lechevallier e Melo (2012). Essa restrição é inspirada nos métodos de agrupamento que são baseados na distância de Mahalanobis e utilizam a normalização pelo determinante da matriz de co-variância. Fazendo uma analogia, as relevâncias dos pesos das tabelas podem ser representadas por uma matriz diagonal, em que o determinante dessa matriz é igual ao produto dos pesos das tabelas. A mesma analogia pode ser feita para a relevância dos pesos das variáveis. Nesse caso, por assumir que é necessário existir independência entre os pesos das variáveis de tabelas diferentes para o modelo ser considerado *multi-view*, cada matriz diagonal precisa representar apenas as relevâncias dos pesos das variáveis de uma única tabela.

A vantagem da utilização da abordagem que restringe valores a partir da multiplicação é que não são criados novos parâmetros que precisariam ser calculados para o bom funcionamento do modelo. Com isso, a função objetivo J_{WVVHCM} é definida com as seguintes restrições:

- $\prod_{l=1}^{s} \pi_l = 1$, $\pi_l > 0$
- $\prod_{j=1}^{p_l} \lambda_{lj} = 1$, $\lambda_{lj} > 0$, $\forall l = 1$ até s

3.2 Etapas da Otimização

Como explicado anteriormente, no modelo WVVHCM, a partir de uma solução inicial, o vetor $\mathbf{G}$ de protótipos, o vetor $\mathbf{\Lambda}$ de vetores de relevância dos pesos das variáveis das tabelas, o vetor $\boldsymbol{\pi}$ de relevância dos pesos das tabelas, e a partição $\mathcal{P}$ são obtidas interativamente em quatro etapas (representação, peso das variáveis, peso das tabelas ou visões e alocação) pela minimização da função objetivo J_{WVVHCM}. Essas etapas são interdependentes e a sequência de execução pode ser vista na figura 3. Esta seção irá detalhar cada uma dessas etapas.

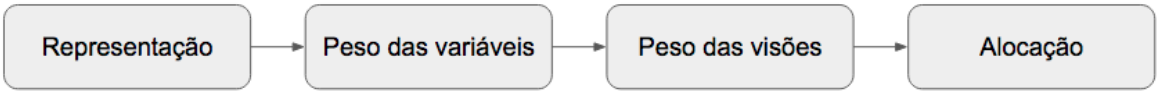


Figura 3 – Etapas que minimizam a função objetivo do modelo WVVHCM

3.2.1 Representação

A etapa de representação tem por objetivo encontrar os protótipos que representam da melhor forma os *clusters* da partição. Durante esta etapa, para que seja possível calcular o vetor $\mathbf{G}$ de protótipos que minimiza J_{WVVHCM}, o vetor $\mathbf{\Lambda}$ de vetores de relevância dos pesos das variáveis das tabelas, o vetor $\boldsymbol{\pi}$ de relevância dos pesos das tabelas, e a partição $\mathcal{P}$ são mantidos fixos. Com isso, a função objetivo do algoritmo é otimizada apenas para os protótipos.

O protótipo $\mathbf{g}_k = (\mathbf{g}_{k1}, \ldots, \mathbf{g}_{kl}, \ldots, \mathbf{g}_{ks})$ do *cluster* P_k pode ser obtido com o cálculo de $\mathbf{g}_{kl} = (g_{kl1}, \ldots, g_{klj}, \ldots, g_{klp_l})$, onde $g_{klj}\,(k = 1, \ldots, K; l = 1, \ldots, s; j = 1, \ldots, p_l)$ possui o valor da j-ésima variável da l-ésima tabela encontrada a partir da derivada parcial de J_{WVVHCM} em relação a g_{klj}, como apresentado a seguir:

$$\frac{\partial J_{WVVHCM}}{\partial g_{klj}} = 0 \tag{3.5}$$

$$\sum_{e_i \in P_k} (x_{ilj} - g_{klj}) = 0 \tag{3.6}$$

$$\sum_{e_i \in P_k} g_{klj} = \sum_{e_i \in P_k} x_{ilj} \tag{3.7}$$

$$n_k g_{klj} = \sum_{e_i \in P_k} x_{ilj} \tag{3.8}$$

Logo, o valor que melhor representa a j-ésima variável da l-ésima tabela do protótipo do *cluster* k pode ser obtido a partir da equação abaixo:

$$g_{klj} = \frac{\sum_{e_i \in P_k} x_{ilj}}{n_k} \tag{3.9}$$

3.2.2 Peso das Variáveis

O objetivo desta etapa é encontrar a melhor relevância dos pesos das variáveis em cada tabela para otimizar a solução gerada pelo modelo WVVHCM. Para isso, serão utilizados os protótipos dos *clusters* computados na etapa anterior. Durante o cálculo dos valores que representam os pesos das variáveis, o vetor $\mathbf{G}$ de protótipos, o vetor $\boldsymbol{\pi}$ de relevância dos pesos das tabelas e a partição $\mathcal{P}$ são mantidos inalterados.

Para que o modelo leve em consideração cada visão de forma independente, o cálculo do vetor $\boldsymbol{\Lambda} = (\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_l, \ldots, \boldsymbol{\lambda}_s)$ de vetores de relevância dos pesos das variáveis das tabelas é feito separadamente para cada visão. Com isso, cada valor $\boldsymbol{\lambda_{lj}}$ do vetor $\boldsymbol{\lambda}_l = (\boldsymbol{\lambda}_{l1}, \ldots, \boldsymbol{\lambda}_{lj}, \ldots, \boldsymbol{\lambda}_{lp_l})$ é calculado levando em consideração apenas os dados da l-ésima tabela, como demonstrado a seguir.

Para facilitar o entendimento da demonstração da equação que calcula λ_{lj}, a função objetivo do modelo proposto será reescrita da seguinte maneira:

$$\begin{aligned} J_{WVVHCM} &= \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{l=1}^{s} \pi_l \sum_{j=1}^{p_l} \lambda_{lj} (x_{ilj} - g_{klj})^2 \\ &= \sum_{l=1}^{s} \pi_l \sum_{j=1}^{p_l} \lambda_{lj} \sum_{k=1}^{K} \sum_{e_i \in P_k} (x_{ilj} - g_{klj})^2 \\ &= \sum_{l=1}^{s} \pi_l \sum_{j=1}^{p_l} \lambda_{lj} W_{lj} \end{aligned} \tag{3.10}$$

Para encontrar o valor de $\boldsymbol{\lambda}_{lj}$ que minimiza a função objetivo J_{WVVHCM} para l e j, onde l é o número da tabela e j é o número da variável da tabela X_l, foi utilizado o método de multiplicadores de Lagrange. Desta forma, derivar J_{WVVHCM} com as restrições $\prod_{h=1}^{p_l} \lambda_{lh} = 1$ e $\lambda_{lh} > 0$ é equivalente a minimizar a seguinte equação de Lagrange $\mathcal{L}_{(\lambda_{lj}, \omega)}$ para λ_{lj}:

$$\mathcal{L}_{(\lambda_{lj}, \omega)} = \pi_l \lambda_{lj} W_{lj} - \omega \left(\prod_{h=1}^{p_l} \lambda_{lh} - 1 \right) \tag{3.11}$$

Como:

$$\frac{\partial}{\partial \lambda_{lj}} \pi_l \lambda_{lj} W_{lj} = \pi_l W_{lj} \tag{3.12}$$

e

$$\frac{\partial}{\partial \lambda_{lj}} \omega \left(\prod_{h=1}^{p_l} \lambda_{lh} - 1 \right) = \omega \left(\prod_{h=1, h \neq j}^{p_l} \lambda_{lh} \right) \tag{3.13}$$

Logo:

$$\frac{\partial}{\partial \lambda_{lj}} \mathcal{L}_{(\lambda_{lj}, \omega)} = \pi_l W_{lj} - \omega \left(\prod_{h=1, h \neq j}^{p_l} \lambda_{lh} \right) = 0 \tag{3.14}$$

Que pode ser escrita da seguinte forma:

$$\omega = \frac{\pi_l \lambda_{lj} W_{lj}}{\prod_{h=1}^{p_l} \lambda_{lh}} \tag{3.15}$$

Como $\prod_{h=1}^{p_l} \lambda_{lh} = 1$:

$$\omega = \pi_l \lambda_{lj} W_{lj} \tag{3.16}$$

Isolando λ_{lj}:

$$\lambda_{lj} = \frac{\omega}{\pi_l W_{lj}} \tag{3.17}$$

Utilizando a seguinte sentença para ω:

$$\prod_{h=1}^{p_l} \omega = (\omega)^{p_l} = \prod_{h=1}^{p_l} \pi_l \lambda_{lh} W_{lh} = \prod_{h=1}^{p_l} \pi_l \prod_{h=1}^{p_l} \lambda_{lh} \prod_{h=1}^{p_l} W_{lh} = (\pi_l)^{p_l} \prod_{h=1}^{p_l} W_{lh} \tag{3.18}$$

Temos:

$$\lambda_{lj} = \frac{\pi_l \prod_{h=1}^{p_l} W_{lh}^{\frac{1}{p_l}}}{\pi_l W_{lj}} \tag{3.19}$$

$$\lambda_{lj} = \frac{\prod_{h=1}^{p_l} W_{lh}^{\frac{1}{p_l}}}{W_{lj}} \tag{3.20}$$

$$\lambda_{lj} = \frac{\left\{\prod_{h=1}^{p_l} \left[\sum_{k=1}^{K} \sum_{e_i \in P_k} (x_{ilh} - g_{klh})^2\right]\right\}^{\frac{1}{p_l}}}{\left[\sum_{k=1}^{K} \sum_{e_i \in P_k} (x_{ilj} - g_{klj})^2\right]} \tag{3.21}$$

Pode ser observado que o cálculo feito para o vetor das relevâncias dos pesos das variáveis $\boldsymbol{\lambda}_l = (\lambda_{l1}, \ldots, \lambda_{lj}, \ldots, \lambda_{lp_l})$ da visão l independe dos valores do vetor $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_l, \ldots, \pi_s)$ das relevância dos pesos das visões. Além disso, o cálculo de $\boldsymbol{\lambda}_l$ independe também das relevâncias dos pesos das variáveis de qualquer outra visão.

3.2.3 Peso das Visões

Nesta etapa, os valores para as relevâncias dos pesos das visões que otimizam a solução gerada pelo modelo WVVHCM são calculados. Para isso, serão utilizados os protótipos dos *clusters* e os valores dos pesos das variáveis computados nas etapas anteriores. Durante o cálculo dos valores que representam os pesos das visões, o vetor $\mathbf{G}$ de protótipos, o vetor $\boldsymbol{\Lambda}$ de vetores de relevância dos pesos das variáveis das tabelas e a partição $\mathcal{P}$ são mantidos inalterados.

Para encontrar o vetor de peso das tabelas $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_l, \ldots, \pi_s)$ que minimiza J_{WVVHCM}, as relevâncias dos pesos das variáveis calculadas na etapa anterior são utilizadas. Para calcular π_l, são levadas em consideração as distâncias entre os elementos e os protótipos dos *clusters*, além dos valores encontrados para os pesos das variáveis não só da visão l, mas de todas as s visões.

Para facilitar o entendimento da demonstração da equação que calcula π_l, a função objetivo J_{WVVHCM} será reescrita da seguinte maneira:

$$\begin{aligned} J_{WVVHCM} &= \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{l=1}^{s} \pi_l \sum_{j=1}^{p_l} \lambda_{lj} (x_{ilj} - g_{klj})^2 \\ &= \sum_{l=1}^{s} \pi_l \sum_{j=1}^{p_l} \lambda_{lj} \sum_{k=1}^{K} \sum_{e_i \in P_k} (x_{ilj} - g_{klj})^2 \\ &= \sum_{l=1}^{s} \pi_l W_l \end{aligned} \tag{3.22}$$

Assim como na abordagem desenvolvida para o cálculo dos pesos das variáveis, foi utilizado o método de multiplicadores de Lagrange para encontrar o valor de $\boldsymbol{\pi}_l$ que minimiza a equação J_{WVVHCM} para l, onde l é o número da tabela. Desta forma, derivar J_{WVVHCM} com as restrições $\prod_{h=1}^{s} \pi_h = 1$ e $\pi_h > 0$ é equivalente a minimizar a seguinte equação de Lagrange $\mathcal{L}_{(\pi_l,\omega)}$ para π_l:

$$\mathcal{L}_{(\pi_l,\omega)} = \pi_l W_l - \omega \left(\prod_{h=1}^{s} \pi_h - 1 \right) \tag{3.23}$$

Como:

$$\frac{\partial}{\partial \pi_l} \pi_l W_l = W_l \tag{3.24}$$

e

$$\frac{\partial}{\partial \pi_l} \omega \left(\prod_{h=1}^{s} \pi_h - 1 \right) = \omega \left(\prod_{h=1, h \neq l}^{s} \pi_h \right) \tag{3.25}$$

Logo:

$$\frac{\partial}{\partial \pi_l} \mathcal{L}_{(\pi_l,\omega)} = W_l - \omega \left(\prod_{h=1, h \neq l}^{s} \pi_h \right) = 0 \tag{3.26}$$

Que pode ser escrita da seguinte forma:

$$\omega = \frac{\pi_l W_l}{\prod_{h=1}^{s} \pi_h} \tag{3.27}$$

Como $\prod_{h=1}^{s} \pi_h = 1$:

$$\omega = \pi_l W_l \tag{3.28}$$

Isolando π_l:

$$\pi_l = \frac{\omega}{W_l} \tag{3.29}$$

Utilizando a seguinte sentença para ω:

$$\prod_{h=1}^{s} \omega = (\omega)^s = \prod_{h=1}^{s} \pi_h W_h = \prod_{h=1}^{s} \pi_h \prod_{h=1}^{s} W_h = \prod_{h=1}^{s} W_h \tag{3.30}$$

Temos:

$$\pi_l = \frac{\prod_{h=1}^{s} W_h^{\frac{1}{s}}}{W_l} \tag{3.31}$$

$$\pi_l = \frac{\left\{ \prod_{h=1}^{s} \left[\sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{j=1}^{p_h} \lambda_{hj} (x_{ihj} - g_{khj})^2 \right] \right\}^{\frac{1}{s}}}{\left[\sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{j=1}^{p_l} \lambda_{lj} (x_{ilj} - g_{klj})^2 \right]} \tag{3.32}$$

Pode ser observado que o cálculo utilizado para encontrar o valor de π_l, por levar em consideração a relevância dos pesos das variáveis das s visões existentes, só pode ser feito após o cálculo de todos os valores do vetor de vetores das relevâncias dos pesos das variáveis $\boldsymbol{\Lambda} = (\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_l, \ldots, \boldsymbol{\lambda}_s)$.

3.2.4 Alocação

Nesta etapa os elementos são alocados nos *clusters* levando em consideração os dados calculados nas etapas anteriores. Durante a alocação, o vetor $\mathbf{G}$ de protótipos, o vetor $\boldsymbol{\Lambda}$ de vetores de relevância dos pesos das variáveis das tabelas e o vetor $\boldsymbol{\pi}$ de relevância dos pesos das tabelas são mantidos inalterados.

Para encontrar a partição $\mathcal{P}$ que minimiza a equação da função objetivo J_{WVVHCM} cada elemento deve ser alocado ao *cluster* que melhor o representa. Isso é feito a partir do cálculo das distâncias entre os protótipos e os elementos, levando em consideração a relevância dos pesos das variáveis e a relevância dos pesos das tabelas. Cada elemento é alocado ao *cluster* representado pelo protótipo que possui a menor distância $d_{(\boldsymbol{\pi},\boldsymbol{\Lambda})}(\mathbf{x}_i, \mathbf{g}_k)$ detalhada em 3.34.

Com isso, os *clusters* $P_k\,(k = 1, \ldots, K)$ que minimizam a função objetivo J_{WVVHCM} são atualizados de acordo com a regra abaixo:

$$P_k = \left\{ e_i \in E : d_{(\boldsymbol{\pi},\boldsymbol{\Lambda})}(\mathbf{x}_i, \mathbf{g}_k) = \min_{h=1}^{K} d_{(\boldsymbol{\pi},\boldsymbol{\Lambda})}(\mathbf{x}_i, \mathbf{g}_h) \right\} \tag{3.33}$$

Onde:

$$d_{(\boldsymbol{\pi},\boldsymbol{\Lambda})}(\mathbf{x}_i, \mathbf{g}_k) = \sum_{l=1}^{s} \pi_l \sum_{j=1}^{p_l} \lambda_{lj} (x_{ilj} - g_{klj})^2 \tag{3.34}$$

3.3 Algoritmo

As etapas de representação, cálculo dos valores dos pesos das variáveis, cálculo dos valores dos pesos das visões e alocação devem ser iteradas até a convergência do modelo. Por serem interdependentes, esses passos devem sempre seguir a ordem definida na seção 3.2

apresentada. O critério de parada utilizado nos experimentos deste trabalho considera a convergência do modelo quando nenhum elemento é realocado para outro *cluster* entre duas iterações subsequentes.

No modelo WVVHCM, assim como no K-Means, a única entrada necessária, além dos elementos do conjunto de dados e seus atributos, é a quantidade final de *clusters*. Na inicialização, os protótipos são escolhidos aleatoriamente dentre os elementos do conjunto de dados. Além disso, é atribuído o valor 1 (um) para todos os pesos das variáveis e todos os pesos das visões, fazendo com que toda informação do conjunto de entrada possua, inicialmente, a mesma relevância, além de atender aos critérios descritos na subseção 3.1.1. O pseudo-código apresentado no algoritmo 5 resume todos os passos necessários para execução, mostrando os dados necessários para entrada e a inicialização do modelo. Também são apresentados o critério de parada e dos dados de saída.

Algorithm 5 WVVHCM

Entrada:
  $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$: o conjunto de dados;
  K: a quantidade de *clusters*;
Inicialização:
  Protótipos: escolhidos aleatoriamente K protótipos distintos $g_k \in \mathcal{D}\,(k = 1, \ldots, K)$, um para cada *cluster*;
  Peso das variáveis: atribuído $\lambda_{lj} = 1\,(l = 1, \ldots, s; j = 1, \ldots, p_l)$
  Peso das visões: atribuído $\pi_l = 1\,(l = 1, \ldots, s)$;
  Criar partição $\mathcal{P}$: criar os *clusters* $(P_1, \ldots, P_K)$ de acordo com as equações (3.33) and (3.34)
Iterar:
  Etapa da representação: atualizar o protótipo $\mathbf{g}_k = (\mathbf{g}_{k1}, \ldots, \mathbf{g}_{kl}, \ldots, \mathbf{g}_{ks})$ do *cluster* P_k, com $\mathbf{g}_{kl} = (g_{kl1}, \ldots, g_{klj}, \ldots, g_{klp_l})$, onde $g_{klj}\,(k = 1, \ldots, K; l = 1, \ldots, s; j = 1, \ldots, p_l)$ é calculado de acordo com a equação (3.9);
  Etapa dos pesos das variáveis: atualizar o vetor de relevância dos pesos das variáveis da tabela (visão) $\mathbf{X}_l$: $\boldsymbol{\lambda}_l = (\lambda_{l1}, \ldots, \lambda_{lj}, \ldots, \lambda_{lp_l})$, onde $\lambda_{lj}\,(l = 1, \ldots, s; j = 1, \ldots, p_l)$ é calculado de acordo com a equação (3.21);
  Etapa dos pesos das visões: atualizar o vetor de relevância dos pesos das tabelas $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_l, \ldots, \pi_s)$, onde $\pi_l\,(l = 1, \ldots, s)$ é calculado de acordo com a equação (3.32);
  Etapa de alocação: atualizar os clusters de acordo com as equações (3.33) and (3.34)
Até:
  Nenhum elemento mudar de *cluster*
Saída:
  $\mathbf{g}_1, \ldots, \mathbf{g}_K$: os protótipos que representam os *clusters*;
  $\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_s$: o vetor de vetores de relevância dos pesos das variáveis das tabelas;
  $\boldsymbol{\pi}$: o vetor de relevância dos pesos das tabelas (visões);
  $\mathcal{P} = (P_1, \ldots, P_K)$: a partição final de E nos K *clusters*.

3.4 Propriedades

Com base nas informações apresentadas, pode-se afirmar que o modelo proposto WVVHCM otimiza uma solução inicial a partir da minimização de uma função objetivo definida na equação 3.4. Essa otimização é feita de forma iterativa e em quatro etapas: representação, cálculo dos pesos das variáveis, cálculo dos pesos das visões e alocação. Também é possível verificar que a função objetivo J_{WVVHCM} leva em consideração o peso das visões e das variáveis para encontrar o *cluster* mais próximo de cada elemento, fazendo com que a partição final seja baseada na distância mínima ponderada entre os elementos de entrada e os centroides dos *clusters*.

Nas seções seguintes serão apresentadas as propriedades e prova de convergência, além da análise da complexidade do modelo WVVHCM.

3.4.1 Convergência do modelo

Seguindo o esquema apresentado em Diday e Simon (1976) para algoritmos de agrupamento dinâmicos, o modelo proposto neste trabalho busca a melhor partição $\mathcal{P}^* = (P_1^*, \ldots, P_K^*)$ dos E elementos em K *clusters* não vazios, os protótipos $\mathbf{G}^* = (\mathbf{g}_1^*, \ldots, \mathbf{g}_K)$ correspondentes, o vetor de relevância dos pesos da tabelas $\boldsymbol{\pi}^* = (\pi_1^*, \ldots, \pi_s^*)$ e o vetor de relevância dos pesos das variáveis $\boldsymbol{\Lambda}^* = (\boldsymbol{\lambda}_1^*, \ldots, \boldsymbol{\lambda}_s^*)$ com $\boldsymbol{\lambda}_l^* = (\lambda_{l1}^*, \ldots, \lambda_{lp_l}^*)$, onde $1 \leq l \leq s$ e $p_l = |\lambda_l|$, tal que:

$$J_{WVVHCM}(\mathbf{G}^*, \boldsymbol{\Lambda}^*, \boldsymbol{\pi}^*, \mathcal{P}^*) = \min\{J_{WVVHCM}(\mathbf{G}, \boldsymbol{\pi}, \boldsymbol{\Lambda}, \mathcal{P}) : \mathbf{G} \in \mathbb{L}^K, \boldsymbol{\Lambda} \in \mathbb{T}, \boldsymbol{\pi} \in \mathbb{M}, \mathcal{P} \in \mathbb{P}_K\} \tag{3.35}$$

Onde:

- $\mathbb{L}$ é o espaço que representa os protótipos, tal que $g_k \in \mathbb{L}$, com $1 \leq k \leq K$, e $\mathbf{G} \in \mathbb{L}^K$. Neste trabalho, $\mathbb{L} = \prod_{l=1}^{s} \mathbb{R}^{p_l}$
- $\mathbb{T}$ é a representação do espaço dos pesos das variáveis das visões, onde $\mathbb{T} = \boldsymbol{\tau}_1 \times \ldots \times \boldsymbol{\tau}_l \times \ldots \times \boldsymbol{\tau}_s$ e $\boldsymbol{\lambda}_l \in \boldsymbol{\tau}_l$. Neste trabalho, $\boldsymbol{\tau}_l = \{\mathbf{a}_l = (a_{l1}, \ldots, a_{lp_l}) \in \mathbb{R}^{p_l} : a_{lj} > 0 \text{ e } \prod_{j=1}^{p_l} a_{lj} = 1\}$.
- $\mathbb{M}$ é a representação do espaço dos pesos das visões. Neste trabalho, $\mathbb{M} = \{\boldsymbol{\mu} = (\mu_1, \ldots, \mu_s) \in \mathbb{R}^s : \mu_l > 0 \text{ e } \prod_{l=1}^{s} \mu_l = 1\}$
- $\mathbb{P}_K$ é o conjunto de todas as partições de E em K *clusters* não vazios, tal que $P_k \in (\rho(E) - \emptyset)$, onde $\rho(E)$ é o conjunto potência de E, e $\mathcal{P} \in \mathbb{P}_K$

Com esses dados, as propriedades de convergência do modelo proposto pode ser estudado a partir das séries:

- $v^{(t)}_{WVVHCM} = (\mathbf{G}^{(t)}, \mathbf{\Lambda}^{(t)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}) \in \mathbb{L}^K \times \mathbb{M} \times \mathbb{T} \times \mathbb{P}_K, t = 0, 1, \ldots;$
- $u^{(t)}_{WVVHCM} = J_{WVVHCM}(v^{(t)}_{WVVHCM}) = J_{WVVHCM}(\mathbf{G}^{(t)}, \mathbf{\Lambda}^{(t)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}), t = 0, 1, \ldots;$

A partir do termo inicial $v^{(0)}_{WVVHCM} = (\mathbf{G}^{(0)}, \mathbf{\Lambda}^{(0)}, \boldsymbol{\pi}^{(0)}, \mathcal{P}^{(0)})$, o algoritmo WVVHCM calcula os diferentes termos da série $v^{(t)}_{WVVHCM}$ até a convergência - que será mostrada -, quando a função objetivo J_{WVVHCM} atinge valores estacionários.

Sabendo que a função objetivo J_{WVVHCM} mede a hetereogenidade da partição a partir da soma das hetereogenidades de cada *cluster* e $J_{WVVHCM} \geq 0$, a preposição que a série $u^{(t)}_{WVVHCM} = J_{WVVHCM}(v^{(t)}_{WVVHCM}) = J_{WVVHCM}(\mathbf{G}^{(t)}, \mathbf{\Lambda}^{(t)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}), t = 0, 1, \ldots;$ decresce a cada iteração e converge pode ser provada a partir da confirmação das seguintes inequações:

$$\begin{aligned}
u^{(t)}_{WVVHCM} &= J_{WVVHCM}(\mathbf{G}^{(t)}, \mathbf{\Lambda}^{(t)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}) \\
&\geq J_{WVVHCM}(\mathbf{G}^{(t+1)}, \mathbf{\Lambda}^{(t)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}) \text{ (I)} \\
&\geq J_{WVVHCM}(\mathbf{G}^{(t+1)}, \mathbf{\Lambda}^{(t+1)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}) \text{ (II)} \\
&\geq J_{WVVHCM}(\mathbf{G}^{(t+1)}, \mathbf{\Lambda}^{(t+1)}, \boldsymbol{\pi}^{(t+1)}, \mathcal{P}^{(t)}) \text{ (III)} \\
&\geq J_{WVVHCM}(\mathbf{G}^{(t+1)}, \mathbf{\Lambda}^{(t+1)}, \boldsymbol{\pi}^{(t+1)}, \mathcal{P}^{(t+1)}) \text{ (IV)} \\
&= u^{(t+1)}_{WVVHCM}
\end{aligned}$$

A inequação (I) se confirma por: $J_{WVVHCM}(\mathbf{G}^{(t+1)}, \mathbf{\Lambda}^{(t)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}) = \sum_{k=1}^{K} \sum_{e_i \in P_k^{(t)}} \sum_{l=1}^{s} \pi_l^{(t)} \sum_{j=1}^{p_l} \lambda_{lj}^{(t)} (x_{ilj} - g_{klj}^{(t+1)})^2$ e pelo que foi apresentado em 3.2.1:

$$\begin{aligned}
\mathbf{G}^{(t+1)} &= (\mathbf{g}_1^{(t+1)}, \ldots, \mathbf{g}_K^{(t+1)}) \\
&= \underset{\mathbf{G}=(\mathbf{g}_1,\ldots,\mathbf{g}_K) \in \mathbb{L}^{\mathbb{K}}}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{e_i \in P_k^{(t)}} \sum_{l=1}^{s} \pi_l^{(t)} \sum_{j=1}^{p_l} \lambda_{lj}^{(t)} (x_{ilj} - g_{klj})^2
\end{aligned}$$

A inequação (II) se confirma por: $J_{WVVHCM}(\mathbf{G}^{(t+1)}, \mathbf{\Lambda}^{(t+1)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}) = \sum_{k=1}^{K} \sum_{e_i \in P_k^{(t)}} \sum_{l=1}^{s} \pi_l^{(t)} \sum_{j=1}^{p_l} \lambda_{lj}^{(t+1)} (x_{ilj} - g_{klj}^{(t+1)})^2$ e pelo que foi apresentado em 3.2.2:

$$\begin{aligned}
\mathbf{\Lambda}^{(t+1)} &= (\boldsymbol{\lambda}_1^{(t+1)}, \ldots, \boldsymbol{\lambda}_s^{(t+1)}) \\
&= \underset{\mathbf{\Lambda}=(\boldsymbol{\lambda}_1,\ldots,\boldsymbol{\lambda}_s) \in \mathbb{T}}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{e_i \in P_k^{(t)}} \sum_{l=1}^{s} \pi_l^{(t)} \sum_{j=1}^{p_l} \lambda_{lj} (x_{ilj} - g_{klj}^{(t+1)})^2
\end{aligned}$$

A inequação (III) se confirma por: $J_{WVVHCM}(\mathbf{G}^{(t+1)}, \mathbf{\Lambda}^{(t+1)}, \boldsymbol{\pi}^{(t+1)}, \mathcal{P}^{(t)}) = \sum_{k=1}^{K} \sum_{e_i \in P_k^{(t)}} \sum_{l=1}^{s} \pi_l^{(t+1)} \sum_{j=1}^{p_l} \lambda_{lj}^{(t+1)} (x_{ilj} - g_{klj}^{(t+1)})^2$ e pelo que foi apresentado em 3.2.3:

$$\begin{aligned}
\boldsymbol{\pi}^{(t+1)} &= (\pi_1^{(t+1)}, \ldots, \pi_s^{(t+1)}) \\
&= \underset{\boldsymbol{\pi}=(\pi_1,\ldots,\pi_s) \in \mathbb{M}}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{e_i \in P_k^{(t)}} \sum_{l=1}^{s} \pi_l \sum_{j=1}^{p_l} \lambda_{lj}^{(t+1)} (x_{ilj} - g_{klj}^{(t+1)})^2
\end{aligned}$$

A inequação (IV) se confirma por: $J_{WVVHCM}(\mathbf{G}^{(t+1)}, \mathbf{\Lambda}^{(t+1)}, \boldsymbol{\pi}^{(t+1)}, \mathcal{P}^{(t+1)}) =$
$\sum_{k=1}^{K} \sum_{e_i \in P_k^{(t+1)}} \sum_{l=1}^{s} \pi_l^{(t+1)} \sum_{j=1}^{p_l} \lambda_{lj}^{(t+1)} (x_{ilj} - g_{klj}^{(t+1)})^2$ e pelo que foi apresentado em 3.2.2:

$$\begin{aligned} \mathcal{P}^{(t+1)} &= (P_1^{(t+1)}, \ldots, P_K^{(t+1)}) \\ &= \underset{\mathcal{P}=(P_1,\ldots,P_K)\in\mathbb{P}_K}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{e_i \in P_k} \sum_{l=1}^{s} \pi_l^{(t+1)} \sum_{j=1}^{p_l} \lambda_{lj}^{(t+1)} (x_{ilj} - g_{klj}^{(t+1)})^2 \end{aligned}$$

Desta forma, pela série $u_{WVVHCM}^{(t)} = J_{WVVHCM}(v_{WVVHCM}^{(t)})$ decrescer e ser limitada por $J_{WVVHCM} \geq 0$, pode-se afirmar que a série converge.

Para mostrar que a série $v_{WVVHCM}^{(t)} = (\mathbf{G}^{(t)}, \mathbf{\Lambda}^{(t)}, \boldsymbol{\pi}^{(t)}, \mathcal{P}^{(t)}), t = 0, 1, \ldots;$ converge, é assumido que a série $u_{WVVHCM}^{(t)}$ atinge um valor estacionário na iteração $t = T$. Desta forma, temos que $u_{WVVHCM}^{(T)} = u_{WVVHCM}^{(T+1)}$ e $J_{WVVHCM}(v_{WVVHCM}^{(T)}) = J_{WVVHCM}(v_{WVVHCM}^{(T+1)})$. Sendo assim, pode-se chegar a $J_{WVVHCM}(\mathbf{G}^{(T)}, \mathbf{\Lambda}^{(T)}, \boldsymbol{\pi}^{(T)}, \mathcal{P}^{(T)}) =$
$J_{WVVHCM}(\mathbf{G}^{(T+1)}, \mathbf{\Lambda}^{(T+1)}, \boldsymbol{\pi}^{(T+1)}, \mathcal{P}^{(T+1)})$. Essa equação pode ser reescrita como as igualdades:

$$\begin{aligned} u_{WVVHCM}^{(T)} &= J_{WVVHCM}(\mathbf{G}^{(T)}, \mathbf{\Lambda}^{(T)}, \boldsymbol{\pi}^{(T)}, \mathcal{P}^{(T)}) \\ &= J_{WVVHCM}(\mathbf{G}^{(T+1)}, \mathbf{\Lambda}^{(T)}, \boldsymbol{\pi}^{(T)}, \mathcal{P}^{(T)}) \text{ (I)} \\ &= J_{WVVHCM}(\mathbf{G}^{(T+1)}, \mathbf{\Lambda}^{(T+1)}, \boldsymbol{\pi}^{(T)}, \mathcal{P}^{(T)}) \text{ (II)} \\ &= J_{WVVHCM}(\mathbf{G}^{(T+1)}, \mathbf{\Lambda}^{(T+1)}, \boldsymbol{\pi}^{(T+1)}, \mathcal{P}^{(T)}) \text{ (III)} \\ &= J_{WVVHCM}(\mathbf{G}^{(T+1)}, \mathbf{\Lambda}^{(T+1)}, \boldsymbol{\pi}^{(T+1)}, \mathcal{P}^{(T+1)}) \text{ (IV)} \\ &= u_{WVVHCM}^{(T+1)} \end{aligned}$$

A partir da igualdade (I), tem-se que $\mathbf{G}^{(T)} = \mathbf{G}^{(T+1)}$. Isso acontece por $\mathbf{G}$ ser único quando J_{WVVHCM} é minimizado com os valores de $\mathbf{\Lambda}^{(T)}, \boldsymbol{\pi}^{(T)}$ e $\mathcal{P}^{(T)}$ mantidos fixos. Em (II), $\mathbf{\Lambda}^{(T)} = \mathbf{\Lambda}^{(T+1)}$ é verdade quando $\mathbf{\Lambda}$ é único, que acontece quando se tenta minimizar J_{WVVHCM} fixando $\mathbf{G}^{(T)}, \boldsymbol{\pi}^{(T)}$ e $\mathcal{P}^{(T)}$. Na terceira igualdade (III), pode-se dizer que $\boldsymbol{\pi}^{(T)} = \boldsymbol{\pi}^{(T+1)}$, pois $\boldsymbol{\pi}$ é único quando se minimiza J_{WVVHCM} com $\mathbf{G}^{(T)}, \mathbf{\Lambda}^{(T)}$ e $\mathcal{P}^{(T)}$ mantidos fixos. Finalmente e de forma semelhante, a partir de (IV), nota-se que $\mathcal{P}^{(T)} = \mathcal{P}^{(T+1)}$. Essa igualdade acontece por $\mathcal{P}$ ser único quando se minimiza J_{WVVHCM} fixando $\mathbf{G}^{(T)}, \mathbf{\Lambda}^{(T)}$ e $\boldsymbol{\pi}^{(T)}$.

Desta forma, pode-se concluir que $v_{WVVHCM}^{(T)} = v_{WVVHCM}^{(T+1)}$ acontece para todo $t \geq T$. Logo, $v_{WVVHCM}^{(t)} = v_{WVVHCM}^{(T)}, \forall t \geq T$, o que indica que $v_{WVVHCM}^{(t)}$ converge.

3.4.2 Análise da complexidade

Para chegar à complexidade computacional do algoritmo WVVHCM, o custo de cada etapa da otimização do modelo e o custo da inicialização devem ser calculados. Para

facilitar o entendimento e a comparação entre o modelo proposto e os modelos concorrentes, nesta seção m representará a quantidade total de variáveis do conjunto de dados de entrada. Desta forma, $m = \sum_{l=1}^{s} p_l$.

Para a inicialização, os pesos de todas as visões e de todas as variáveis devem receber o valor 1 (um), que possui a complexidade $\mathcal{O}(s+m)$. Além disso, o protótipo de cada *cluster* receberá os valores das variáveis de um elemento escolhido aleatoriamente sem repetição. Essa operação possui complexidade $\mathcal{O}(Km\log(Km))$. Após a definição dos protótipos, a partição inicial - que será otimizada nas iterações do algoritmo - é criada com complexidade computacional $\mathcal{O}(Knm)$. Nas etapas de cálculo dos pesos das variáveis e cálculo dos pesos das visões, os custos computacionais são, respectivamente, de $\mathcal{O}(Knm\log(m))$ e $\mathcal{O}(s^2Kn\log(m))$. Já a alocação tem complexidade de $\mathcal{O}(Knm)$, que é o custo do cálculo da função objetivo.

Desta forma, caso o modelo necessite de r repetições para convergência, a complexidade computacional total do algoritmo WVVHCM é de $\mathcal{O}((m+s^2)rKn\log(m))$. Esta é a mesma complexidade dos algoritmos já existentes que levam em consideração os pesos das visões e variáveis para otimização da solução inicial.

4 ANÁLISE EXPERIMENTAL

Este capítulo explica como foram feitas as análises das performances do modelo proposto *Multi-View Hard C-Means With Automated Weighting Of Views And Variables* (WVVHCM) e dos modelos concorrentes selecionados. Serão detalhadas a metodologia utilizada para realização dos experimentos, os indíces que foram calculados para avaliar a performance dos modelos e os conjuntos de dados utilizados como entrada para os algoritmos.

Assim como explicado no capítulo 2, o modelo WVVHCM será comparado com quatro outros modelos: *K-Means* tradicional (MACQUEEN, 1967), *Multi-view Kernel K-Means* (MVKKM) (TZORTZIS; LIKAS, 2009a), *Automated Two-Level Variable Weighting Clustering Algorithm for Multiview Data* (TW-K-Means) (CHEN X., 2013) e *Weighted Multi-view Clustering with Feature Selection* (WMCFS) (XU; WANG; LAI, 2016). O primeiro desses modelos (K-Means) opera apenas em conjuntos de dados que possuem uma única tabela e sem fazer distinção entre as variáveis existentes. O segundo algoritmo (MVKKM) faz distinção apenas entre as relevâncias das diferentes tabelas, fazendo com que qualquer atributo dentro de cada tabela tenha a mesma influência para a construção da partição final. Os dois últimos modelos concorrentes (TW-K-Means e WMCFS), assim como o modelo proposto (WVVHCM), fazem distinção entre as tabelas e os atributos, atribuindo diferentes relevâncias às tabelas e aos atributos do conjunto de dados.

A escolha dos modelos concorrentes foi feita com a intenção de mostrar a importância da evolução dos métodos centralizados de agrupamento *multi-view*, dando destaque a importância de se levar em consideração as relevâncias das visões e variáveis das visões no processo de agrupamento de dados com múltiplas tabelas. Além disso, também será possível comparar o desempenho do modelo proposto com dois algoritmos que atuam de forma semelhante.

4.1 Metodologia

Para obtenção dos resultados que irão medir a performance dos modelos, cada algoritmo foi executado um total de 100 vezes para cada conjunto de dados. Em cada execução, o modelo pôde otimizar localmente a solução inicial até chegar ao agrupamento final. A quantidade de execuções é necessária pelo fato da performance dos modelos depender diretamente do ponto de partida dos algoritmos (solução inicial a ser otimizada). Para todos os modelos e para cada execução, diferentes elementos - escolhidos aleatoriamente dentro dos conjuntos de dados - foram utilizados como protótipos iniciais dos clusters. Com isso, as várias execuções são necessárias para que sejam dadas as mesmas condições

iniciais aos algoritmos.

Os resultados das execuções serão analisados de duas formas no próximo capítulo. Uma maneira será a comparação do melhor resultado de cada algoritmo, extraído a partir da avaliação do índice interno de cada modelo. No caso dos modelos analisados, a melhor execução selecionada, ou seja, a de melhor índice interno, será a que obtiver o menor resultado da função objetivo utilizada no processo de otimização da solução inicial (busca local da partição ótima). Além do melhor resultado, também serão comparadas as médias obtidas a partir das 100 execuções. Para ambas as comparações foram utilizados diferentes índices que avaliam a performance dos agrupamentos.

O K-Means é o único modelo testado que funciona apenas com uma tabela de dados. Logo, nos experimentos realizados para este algoritmo, os conjuntos de dados *multi-view* tiveram suas visões concatenadas, formando uma única tabela com todos os atributos. Já para os modelos MVKKM, TW-K-Means e WMCFS, uma etapa anterior a da execução dos experimentos é imprescindível. Isso ocorre devido à necessidade de se calcular valores de entrada de parâmetros que interferem na distribuição dos pesos das tabelas ou variáveis durante o processo de agrupamento. Esses parâmetros, que não são necessários para os modelos K-Means e WVVHCM, possuem um valor ideal diferente para cada conjunto de dados e uma metodologia específica para cálculo que não necessite dos rótulos dos elementos não foi proposta pelos autores. Nos artigos originais dos modelos, a ajuste desses parâmetros só foi possível ser feito a partir de uma etapa supervisionada, o que não é possível em aplicações reais no contexto não supervisionado. Neste trabalho, para encontrar os valores ideais dos parâmetros, foram feitas as seguintes variações, sugeridas pelos autores, para cada um dos três modelos:

- MVKKM: Seguindo as recomendações feitas em Tzortzis e Likas (2012), o parâmetro que controla a esparsidade dos pesos das tabelas, chamado pelo autor de p, foi variado entre 1.0 e 6.0, com diferença de 0.5 entre cada valor testado.

- TW-k-Means: Como explicado por Chen X. (2013), o parâmetro λ, que controla a distribuição dos pesos das tabelas, foi testado para os seguintes valores: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110 e 120. O parâmetro η, que controla a distribuição dos pesos dos atributos foi variado de 1 a 30, com diferença de 1 para cada valor testado.

- WMCFS: A partir do que foi explicado em Xu, Wang e Lai (2016), o parâmetro p, que controla a esparsidade dos pesos das tabelas, foi variado de 1 a 30, com diferença de 1 para cada valor testado. Já para β, parâmetro que controla a esparsidade dos pesos das variáveis, foram testados os seguintes valores entre 0.0001 e 1.0: 0.0001, 0.001, 0.01, 0.1 e 1.0.

Para cada um desses três modelos, todas as possíveis combinações dos valores desses parâmetros foram testadas para todos os conjuntos de dados. Foram feitas 10 execuções para cada possibilidade de combinação desses valores. Para todas as execuções efetuadas dentro de cada conjunto, 10 diferentes variações de protótipos foram apresentados como solução inicial a ser otimizada. Desta forma, toda combinação de valores de parâmetros dentro de cada modelo obteve as mesmas condições iniciais 10 vezes. Os valores selecionados dos parâmetros para os experimentos que comparam a performance dos modelos foram os que obtiveram o melhor *Normalized Mutual Information* (NMI), um dos índices utilizados para avaliação dos agrupamentos gerados neste trabalho. Os valores obtidos para os parâmetros estão detalhados na explicação dos conjuntos de dados que será feita na seção 4.3.

Para cada uma das 100 execuções realizadas nos experimentos, a solução inicial apresentada para os modelos foi obtida a partir da escolha aleatória de K diferentes elementos (dentro do conjunto de dados) para representar os K *clusters* existentes. No caso do modelo MVKKM, seguindo a sugestão do autor, a solução inicial apresentada ao algoritmo foi a partição de saída do modelo *Global Kernel K-Means*. Logo, nesse caso, a solução com protótipos escolhidos aleatoriamente era apresentada ao algoritmo *Global Kernel K-Means* que apresentava sua saída ao algoritmo MVKKM. Dentro de cada execução, os quatro modelos fizeram repetições para otimizar a solução e, para todos os modelos, o critério de parada utilizado foi que não houvesse mudança nos elementos dos *clusters* em duas repetições subsequentes.

Todos os algoritmos foram implementados utilizando a linguagem C e os experimentos foram executados em uma máquina Windows de 64 bits com processador de 3.47 GHz e 16 GB de memória RAM.

4.2 Análise da Performance

Pelo fato dos rótulos dos elementos dos conjuntos de dados não serem conhecidos na aprendizagem não supervisionada, a avaliação dos modelos que seguem essa abordagem não é uma tarefa simples. Nos experimentos realizados na aprendizagem supervisionada, o objetivo dos modelos é classificar elementos com o rótulo correto. Nesse contexto, os algoritmos são treinados e os resultados obtidos em experimentos controlados podem ser avaliados através de uma comparação entre as classificações recebidas e as classificações reais dos elementos. Já na aprendizagem não supervisionada, os rótulos dos elementos não podem ser levados em consideração durante os experimentos realizados. Desta forma, o objetivo dos algoritmos de *clustering* é a formação de grupos de objetos que possuem máxima semelhança entre seus atributos, independente da classificação real desses elementos. Esses grupos podem, muitas vezes, ser equivalentes aos rótulos dos objetos, porém não existe necessariamente uma dependência entre essas duas informações.

Hoje em dia, os índices de avaliação de performance dos algoritmos de *clustering* podem ser classificados em três categorias: interno, externo e relativo (HALKIDI; VAZIRGIANNIS, 2001; RENDON et al., 2011). Os índices internos geralmente são utilizados pelos proprios algoritmos de agrupamento para chegar a uma partição local ótima. No caso dos modelos avaliados neste trabalho, a função objetivo representa o índice interno de cada modelo. Para os índices externos, deve-se assumir um conhecimento prévio dos dados. Nesse caso, os rótulos dos elementos precisam ser conhecidos para que uma comparação entre os *clusters* e a classificação real seja feita. Já os índices relativos são utilizados para verificar a estrutura dos grupos resultantes com o objetivo de checar se os elementos estão agrupados de forma concisa.

Pelo fato de todos os conjuntos de dados utilizados neste trabalho possuírem os elementos rotulados, foram selecionados quatro índices externos amplamente utilizados para tentar avaliar a performance dos modelos testados. Para analisar o resultado, os valores dos índices foram calculados para todas as 100 execuções de todos os modelos. A partir dos valores encontrados, testes estatísticos paramétricos e não paramétricos foram utilizados para comparação da perfomance.

Os índices utilizados neste trabalho foram: *Normalized Mutual Information - NMI* (STREHL; GHOSH, 2002), *Adjusted Rand Index* (HUBERT; ARABIE, 1985), *F-Measure* (RIJISBERGEN, 1979) e *Purity* (TAN M. STEINBACH, 2005). As subseções abaixo explicam de forma resumida cada um desses índices.

4.2.1 Normalized Mutual Information - NMI

O NMI é um índice baseado na informação mútua dos agrupamentos, uma definição que pode ser extendida da entropia dos *clusters* (WAGNER; WAGNER, 2007). O conceito de entropia aplicada a *clusters* pode ser definida como uma forma quantitativa de medir a incerteza de um elemento escolhido aleatoriamente pertencer a um determinado *cluster*. Como definido em Meila (2003), ao assumir que todos os elementos de um agrupamento C possuem a mesma chance de serem selecionados, a probabilidade de um elemento escolhido aleatoriamente pertencer ao *cluster* C_i pode ser definida como $P(i)$. Com isso, a entropia associada ao agrupamento C pode ser calculada utilizando a equação abaixo:

$$H(C) = -\sum_{i=1}^{k} P(i) \log_2 P(i) \tag{4.1}$$

Caso exista mais de um agrupamento dos mesmos elementos, a informação mútua pode ser descrita como uma forma de minimizar a incerteza de um elemento aleatório pertencer a um *cluster* em uma partição sabendo o *cluster* dele em outra partição. Com isso, considerando dois agrupamentos C e C', podemos definir $P(i, j)$ como a probabilidade de um elemento escolhido aleatoriamente pertencer ao *cluster* C_i sabendo que ele pertence

ao *cluster* C'_j. Formalmente, a informação mútua entre os agrupamentos C e C' pode ser calculada pela equação abaixo:

$$I(C, C') = -\sum_{i=1}^{k}\sum_{j=1}^{l} P(i,j)\log_2 \frac{P(i,j)}{P(i)P(j)} \tag{4.2}$$

Nesse trabalho, foi utilizada uma variação que normaliza o valor da informação mútua, chamada de NMI. Esse índice possui um valor entre 0 e 1, onde 0 é encontrado quando os dois agrupamentos comparados não possuem nenhuma informação mútua e 1 quando os agrupamentos possuem uma correlação perfeita (agrupamentos idênticos).

O cálculo do índice que realiza a média geométrica dos valores encontrados para a informação mútua foi criado em Strehl e Ghosh (2002) e está detalhado na equação 4.3:

$$NMI_{sg} = \frac{I(C, C')}{\sqrt{H(C)H(C')}} \tag{4.3}$$

Para que o índice seja utilizado como uma medida efetiva de performance dos modelos, a classificação original dos elementos nos conjuntos de dados utilizados nesse trabalho foi considerada como o agrupamento conhecido para o cálculo dos valores do NMI.

4.2.2 Adjusted Rand Index

Como o nome explica, o *Adjusted Rand Index* (ARI) é um ajuste do *Rand Index* (RI) (RAND, 1971), um índice que calcula a proporção de classificações concordantes de pares de elementos em dois agrupamentos. Sendo assim, considerando n elementos agrupados de duas formas diferentes, onde a é a quantidade de pares classificados da mesma forma nos dois agrupamentos e b é a quantidade de elementos classificados de forma diferente nos dois agrupamentos, o cálculo do RI pode ser feito a partir da equação abaixo:

$$RI = \frac{a+b}{\binom{n}{2}} \tag{4.4}$$

Como mostrado em Morey e Agresti (1981) e Fowlkes E. B. (1983), a quantidade de *clusters* pode influenciar diretamente no cálculo do *Rand Index*. Para evitar esse problema e com o objetivo de normalizar os valores encontrados entre 0 (nenhum par classificado da mesma forma nos dois agrupamentos) e 1 (agrupamentos idênticos), o índice utilizado neste trabalho foi o *Adjusted Rand Index*. Considerando m_{ij} o número de pares que obtiveram classificação comum nas partições C e C', o ARI pode ser calculado segundo a equação abaixo:

$$ARI = \frac{\sum_{i=1}^{k}\sum_{j=1}^{l}\binom{m_{ij}}{2} - t_3}{\frac{1}{2}(t_1 + t_2) - t_3} \tag{4.5}$$

Onde:

$$t_1 = \sum_{i=1}^{k} \binom{|C_i|}{2}, t_2 = \sum_{j=1}^{l} \binom{|C'_j|}{2}, t_3 = \frac{2t_1 t_2}{n(n-1)} \tag{4.6}$$

Da mesma forma que o índice NMI, a classificação original dos elementos nos conjuntos de dados utilizados nesse trabalho foi considerada como um dos agrupamentos a serem comparados.

4.2.3 Purity

Outro índice calculado para avaliar o modelo proposto é o *Purity*. Para que seu valor seja encontrado, as classes dos elementos da partição precisam ser conhecidas e cada *cluster* resultante deve ser associado à classe de maior frequência dentro do grupo (MANNING; RAGHAVAN; SCHüTZE, 2008). A quantidade de elementos classificados corretamente dentro de cada associação de *cluster* e classe é somada e dividida pela quantidade total de elementos.

Dessa forma, sendo C'_j uma classe e C_i um *cluster* da partição gerada, a equação do *Purity* pode ser escrita da seguinte forma:

$$Purity(C, C') = \frac{1}{n} \sum_{i=1}^{k} \max_j |C_i \cap C'_j| \tag{4.7}$$

Os valores obtidos como resultado da equação deste índice podem variar entre 0 e 1, onde 0 é encontrado quando os grupos resultantes e as classes dos elementos não possuem nenhuma informação relacionada e 1 quando os grupos e as classses dos elementos possuem uma correlação perfeita.

4.2.4 F-Measure

O índice *F-Measure* (RIJISBERGEN, 1979) foi originalmente criado para avaliar a categorização de documentos utilizando classificação não supervisionada em experimentos onde os rótulos dos documentos eram conhecidos. O índice é obtido comparando *clusters* da partição final com as classes dos documentos. O valor *F-Measure* obtido para cada classe C'_j descreve se ela é bem representada pelo *cluster* C_i. A ideia desse índice é semelhante ao *Purity*, porém é calculado a partir da média harmônica da precisão (fração de elementos recuperados que são relevantes) e da revocação (fração de elementos relevantes que são recuperados).

Sendo C'_j uma classe e C_i um *cluster* da partição gerada, a equação do *F-Measure* pode ser escrita da seguinte forma:

$$F(C_i, C'_j) = \frac{2|C_i||C'_j|}{|C_i| + |C'_j|} \tag{4.8}$$

O cálculo do *F-Measure* geral da partição resultante pode ser definido como a soma ponderada dos valores máximos obtidos para os clusters da partição C:

$$F(C) = \frac{1}{n} \sum_{i=1}^{k} n_i \max_{j=1}^{l} F(C_i, C'_j) \tag{4.9}$$

Assim como o índice *Purity*, os valores obtidos como resultado da equação deste índice podem variar entre 0 e 1, onde 0 é encontrado quando os grupos resultantes e as classes dos elementos não possuem nenhuma informação relacionada e 1 quando os grupos e as classses dos elementos possuem uma correlação perfeita.

4.3 Conjuntos de Dados

Para chegar aos resultados obtidos neste trabalho, foram realizados experimentos com oito conjuntos de dados com configurações que variam a quantidade de visões e a quantidade de variáveis de cada visão. Esses conjuntos foram extraídos a partir de sete fontes de dados que possuem diferentes tipos de informação, como imagens, digitos manuscritos, fonemas e textos. Em uma das fontes (Corel Images), dois subconjuntos foram gerados a partir dos dados existentes. A tabela 6 apresenta um resumo das configurações dos oito conjuntos.

Tabela 6 – Resumo dos conjuntos de dados utilizados

Conjunto de dados	**Elementos**	**Classes**	**Visões**	**Variáveis**
Multiple features	2000	10	6	649
Animals with attributes	2000	50	6	10934
Phoneme	150	5	3	447
Reuters	1200	6	5	500
Image Segmentation	2310	7	2	19
ALOI	1413	13	4	106
Corel 1	400	4	7	338
Corel 2	400	4	7	338

Esta seção irá apresentar cada um desses conjuntos, detalhando a origem, a quantidade de classes e de elementos, entre outras informações.

4.3.1 Multiple Features (Mfeat)

O conjunto *Multiple Features* [1] descreve 2000 dígitos escritos à mão ('0'–'9') extraídos de uma coleção de mapas holandeses de utilidade. Nele existem 10 classes e cada classe possui 200 imagens. Esses dígitos são apresentados em seis diferentes conjuntos de variáveis (*Fourier coefficients, Profile correlations, Karhunen-Love coefficients, Pixel averages in 2 x 3 windows, Zernike moments e Morphological features*).

Para os experimentos feitos neste trabalho, as variáveis de cada visão foram normalizadas pela subtração de sua média e divisão pela unidade de desvio padrão. A tabela 7 detalha as múltiplas visões do conjunto com a quantidade de variáveis existentes para cada tabela.

Tabela 7 – Visões do Conjunto Multiple Features

View	**Description**	**Variables**
mfeat-fou	Fourier coefficients	76
mfeat-fac	Profile correlations	216
mfeat-kar	Karhunen-Love coefficients	64
mfeat-pix	Pixel averages in 2 x 3 windows	240
mfeat-zer	Zernike moments	47
mfeat-mor	Morphological features	6

A figura abaixo apresenta exemplos de dígitos descritos no conjunto de dados *Multiple Features*.

Figura 4 – Exemplo de dígitos existentes no conjunto Mfeat

Para este conjunto, os valores encontrados para os parâmetros de entrada que controlam a dispersão da distribuição dos pesos nos modelos MVKKM, TW-K-Means e WMCFS foram:

[1] https://archive.ics.uci.edu/ml/datasets/Multiple+Features

- MVKKM: $p = 4.0$
- TW-K-Means: $\lambda = 30.0$ e $\eta = 16.0$
- WMCFS: $p = 4.0$ e $\beta = 0.1$

4.3.2 Image Segmentation

No conjunto *Image Segmentation*[2] 2310 instâncias foram selecionadas aleatoriamente de um banco de dados de 7 imagens (classes). As imagens foram segmentadas à mão para criar uma classificação para cada pixel. No total, existem 19 variáveis que foram divididas em duas visões. Uma visão é a exibição RGB, que consiste em 10 variáveis e a outra é a exibição *Shape*, que consiste de 9 variáveis.

Para os experimentos feitos neste trabalho, as variáveis de cada visão foram normalizadas pela subtração de sua média e divisão pela unidade de desvio padrão. A tabela 8 detalha as duas visões do conjunto com a quantidade de variáveis existentes para cada tabela.

Tabela 8 – Visões do Conjunto Image Segmentation

View	**Description**	**Variables**
RGB	Features related to the color	10
Shape	Features related to the shape	9

Para este conjunto, os valores encontrados para os parâmetros de entrada que controlam a dispersão da distribuição dos pesos nos modelos MVKKM, TW-K-Means e WMCFS foram:

- MVKKM: $p = 2.0$
- TW-K-Means: $\lambda = 60.0$ e $\eta = 21.0$
- WMCFS: $p = 11.0$ e $\beta = 0.1$

4.3.3 Phoneme

O conjunto *Phoneme*[3], que possui originalmente uma visão com 150 variáveis, consiste em 2000 objetos divididos em cinco fonemas (classes): “sh”, “iy”, “dcl”, “aa” e “ao”. Cada classe tem 400 instâncias (objetos). Para este conjunto, duas visões adicionais (velocidade e aceleração) foram calculadas através da aplicação da distância euclidiana ao quadrado das variáveis originais (log-periodogramas discretizados).

[2] https://archive.ics.uci.edu/ml/datasets/Image+Segmentation
[3] http://www.math.univ-toulouse.fr/staph/npfda/npfda-datasets.html

Para os experimentos feitos neste trabalho, as variáveis de cada visão foram normalizadas pela subtração de sua média e divisão pela unidade de desvio padrão. A tabela 9 detalha as múltiplas visões do conjunto com a quantidade de variáveis existentes para cada tabela.

Tabela 9 – Visões do Conjunto Phoneme

View	**Description**	**Variables**
npfda-position	Position (discretized log-periodograms)	150
npfda-velocity	Velocity	149
npfda-acceleration	Acceleration	148

Para o conjunto *Phoneme*, os valores encontrados para os parâmetros de entrada que controlam a dispersão da distribuição dos pesos nos modelos MVKKM, TW-K-Means e WMCFS foram:

- MVKKM: $p = 1.5$
- TW-K-Means: $\lambda = 30.0$ e $\eta = 19.0$
- WMCFS: $p = 8.0$ e $\beta = 0.1$

4.3.4 Reuters RCV1/RCV2 Multilingual

O conjunto original *Reuters*[4] descreve 111.740 documentos escritos em cinco idiomas diferentes (inglês, francês, alemão, espanhol e italiano). Esses documentos estão separados em 6 grandes categorias da *Reuters* (CCAT, C15, ECAT, E21, GCAT e M11). Para produzir versões multilingues dos documentos, cada documento original foi traduzido para os outros 4 idiomas usando um sistema de tradução automática. Para este projeto, 1200 documentos (200 de cada classe ou categoria) escritos em inglês e suas traduções para outras línguas foram selecionados aleatoriamente. Uma vez que em dados de alta dimensão os métodos de agrupamento ou classificação geralmente não obtém bons resultados, seguindo a abordagem descrita em Kumar e Daumé (2011), os dados foram primeiro reduzidos para um espaço de 100 dimensões. O processo de redução dimensional já produz dados normalizados.

A tabela 10 detalha as múltiplas visões do conjunto com a quantidade de variáveis existentes para cada tabela.

Para este conjunto, os valores encontrados para os parâmetros de entrada que controlam a dispersão da distribuição dos pesos nos modelos MVKKM, TW-K-Means e WMCFS foram:

[4] http://ama.liglab.fr/~amini/DataSets/Classification/Multiview/ReutersMulti LingualMultiView.htm

Tabela 10 – Visões do Conjunto Reuters RCV1/RCV2 Multilingual

View	Description	Variables
IndexEN-EN	Original text in english	100
IndexEN-FR	Translated from english	100
IndexEN-GR	Translated from english	100
IndexEN-IT	Translated from english	100
IndexEN-SP	Translated from english	100

- MVKKM: $p = 1.5$
- TW-K-Means: $\lambda = 20.0$ e $\eta = 30.0$
- WMCFS: $p = 10.0$ e $\beta = 0.1$

4.3.5 Corel Images

Os subconjuntos de dados Corel utilizados neste trabalho foram extraídos da base de dados *Corel Images*[5], que possui 34 categorias, cada uma com 100 imagens. No conjunto existem 7 diferentes visões: *Color Histogram, Color Moment, Color Coherence, Coarsness - Tamura Texture, Directionality - Tamura Texture Wavelet Texture, MASAR Texture.* Para este trabalho, 2 subconjuntos distintos de dados foram selecionados, cada um com quatro categorias. A tabela 11 detalha esses subconjuntos.

Para os experimentos executados, as variáveis de cada visão foram normalizadas pela subtração de sua média e divisão pela unidade de desvio padrão. A tabela 12 detalha as múltiplas visões do conjunto com a quantidade de variáveis existentes para cada tabela.

Tabela 11 – Subconjuntos extraídos do conjunto Corel Images

Dataset	Categories			
Corel 1	owls	tiger	trains	ships
Corel 2	buses	leopards	cars	deers

Uma imagem exemplo que representa cada categoria selecionada para este trabalho é mostrada na figura 5.

Para as variações do conjunto *Corel*, os valores encontrados para os parâmetros de entrada que controlam a dispersão da distribuição dos pesos nos modelos MVKKM, TW-K-Means e WMCFS foram:

- MVKKM: $p = 2.0$ (Corel 1); $p = 3.0$ (Corel 2)

[5] http://www.cs.virginia.edu/~xj3a/research/CBIR/Download.htm

Tabela 12 – Visões do Conjunto Corel Images

View	Description	Variables
ColorHsvHistogram64	Color Histogram	64
ColorLuvMoment123	Color Moment	9
ColorHsvCoherence64	Color Coherence	128
CoarsnessVector	Coarsness - Tamura Texture	10
Directionality	Directionality - Tamura Texture	8
WaveletTwtTexture	Wavelet Texture	104
MRSAR	MASAR Texture	15

Figura 5 – Exemplo de imagens do conjunto Corel utilizadas nos experimentos

- TW-K-Means: $\lambda = 100.0$ e $\eta = 23.0$ (Corel 1); $\lambda = 110.0$ e $\eta = 29.0$ (Corel 2)
- WMCFS: $p = 17.0$ e $\beta = 0.1$ (Corel 1); $p = 8.0$ e $\beta = 0.1$ (Corel 2)

4.3.6 Amsterdam Library of Object Images (ALOI)

Este conjunto de dados foi extraído da coleção ALOI[6], uma base de 110.250 imagens de mil pequenos objetos que foram gravadas variando o ângulo, a cor e a direção da iluminação

[6] http://aloi.science.uva.nl/

(entre outras formas de captura). Para este trabalho, foram utilizados 13 objetos que representam veículos, onde cada objeto equivale a uma classe e as imagens gravadas equivalem a um elemento dessa classe. Os grupos (visões) de atributos utilizados foram: *Color Histogram, Color Moment, Color HSB Model (Hue, Saturation and Brightness) e Haralick texture features.*

Para os experimentos executados, as variáveis de cada visão foram normalizadas pela subtração de sua média e divisão pela unidade de desvio padrão. A tabela 13 detalha as múltiplas visões do conjunto com a quantidade de variáveis existentes para cada tabela.

Tabela 13 – Visões do Conjunto ALOI

View	**Description**	**Variables**
Color-8d	Color Histogram	8
Colorsim77	Color Moment	77
Color-HSB-2x2x2	Color HSB Model	8
Haralick	Haralick texture features	13

Exemplos de alguns objetos utilizados para este trabalho são apresentados na figura 6.

Figura 6 – Exemplos de objetos utilizados nos experimentos do conjunto ALOI

Para o conjunto ALOI, os valores encontrados para os parâmetros de entrada que controlam a dispersão da distribuição dos pesos nos modelos MVKKM, TW-K-Means e WMCFS foram:

- MVKKM: $p = 5.0$
- TW-K-Means: $\lambda = 110.0$ e $\eta = 29.0$

- WMCFS: $p = 9.0$ e $\beta = 0.1$

4.3.7 Animals with Attributes (AWA)

Este conjunto de dados foi extraído da coleção *Animals with Attributes* (AWA)[7], uma base de 30475 imagens de 50 categorias de animais diferentes. Na coleção, cada categoria contém ao menos 92 imagens. Para este trabalho, foram utilizadas 40 imagens de cada categoria, num total de 2000 imagens. Seis conjuntos de atributos com tipos diferentes de características foram extraídos dessas imagens: histogramas de cor RGB, SIFT, rgSIFT, PHOG, SURF e histogramas de auto-similaridade locais.

Para os experimentos feitos neste trabalho, as variáveis de cada visão foram normalizadas pela subtração de sua média e divisão pela unidade de desvio padrão. A tabela 14 detalha as múltiplas visões do conjunto com a quantidade de variáveis existentes para cada tabela.

Tabela 14 – Visões do Conjunto AWA

View	**Description**	**Variables**
awa-cq	Color Histogram features	2687
awa-phog	PyramidHOG (PHOG) features	251
awa-sift	SIFT features	1999
awa-rgsift	Color SIFT features	1999
awa-surf	SURF features	1999
awa-lss	Local Self-Similarity features	1999

Para o conjunto AWA, os valores encontrados para os parâmetros de entrada que controlam a dispersão da distribuição dos pesos nos modelos MVKKM, TW-K-Means e WMCFS foram:

- MVKKM: $p = 6.0$
- TW-K-Means: $\lambda = 30.0$ e $\eta = 20.0$
- WMCFS: $p =$ e 11.0 $\beta = 0.1$

[7] http://attributes.kyb.tuebingen.mpg.de/

5 RESULTADOS

Neste capítulo serão apresentados os resultados obtidos nos experimentos realizados durante o trabalho de pesquisa. Os dados serão analisados sob duas perspectivas. A primeira perspectiva é o estudo da média, desvio padrão e intervalo de confiança dos índices obtidos em cada conjunto de dados para todas as execuções efetuadas. A segunda é a análise dos melhores resultados para cada índice em cada conjunto de dados dentre todas as execuções realizadas. A análise do melhor resultado tem como meta verificar qual modelo conseguiu obter localmente, a partir das repetições que otimizam sua função objetivo, o resultado de melhor desempenho em relação aos outros modelos. Já a análise da média e desvio padrão poderá indicar, principalmente, o modelo que possui comportamento mais estável, ou seja, que a partir de diferentes pontos de partida (diferentes soluções iniciais a serem otimizadas), obteve resultados com performance semelhante (com pouca variação dos valores obtidos para os índices avaliados).

Antes da apresentação dos resultados finais, é importante destacar que foi necessária uma etapa anterior aos experimentos para os modelos MVKKM, TW-K-Means e WMCFS. Essa etapa é de extrema importância para os 3 modelos e tem como objetivo a escolha dos parâmetros de entrada que influenciam na performance desses algoritmos. Uma metodologia não supervisionada não foi fornecida pelos autores e, até onde foi pesquisado, não existe procedimento específico para descoberta dos valores desses parâmetros. Desta forma, o método de seleção dos valores de entrada, assim como indicado nos artigos originais, só pôde ser realizado com o conhecimento prévio dos rótulos dos elementos dos conjuntos de dados que serão agrupados. A má escolha dos valores de entrada dos parâmetros pode influenciar na performance final desses modelos.

Para mostrar um exemplo da importância dessa etapa anterior, foi realizado um segundo experimento utilizando esses três modelos citados apenas com o conjunto de dados *Image Segmentation*. Nesse segundo experimento, também foram feitas 100 execuções para cada variação possível dos parâmetros dos modelos MVKKM, TW-K-Means e WMCFS (de acordo com a seção 4.1 do capítulo 4). O índice externo NMI foi calculado para o melhor resultado de cada combinação desses valores (de acordo com a função objetivo de cada algoritmo). A escolha de um único conjunto de dados e do cálculo de um único índice se deram devido à excessiva quantidade de execuções necessárias para cada modelo. Como exemplo, para o WMCFS, foram necessárias 36.000 execuções com várias iterações cada. Logo, o menor conjunto de dados e apenas o índice NMI, que foi utilizado para escolha dos parâmetros de entrada desses modelos para comparação de performance, foram selecionados para a realização desse segundo experimento. A figura 7 apresenta gráficos que indicam a variação do valor NMI encontrado de acordo com a variação dos parâmetros de entrada. Com isso, pode-se afirmar que a falta de conhecimento prévio dos dados que

serão agrupados pode levar a uma má escolha dos parâmetros de entrada, o que influencia diretamente na performance desses três algoritmos.

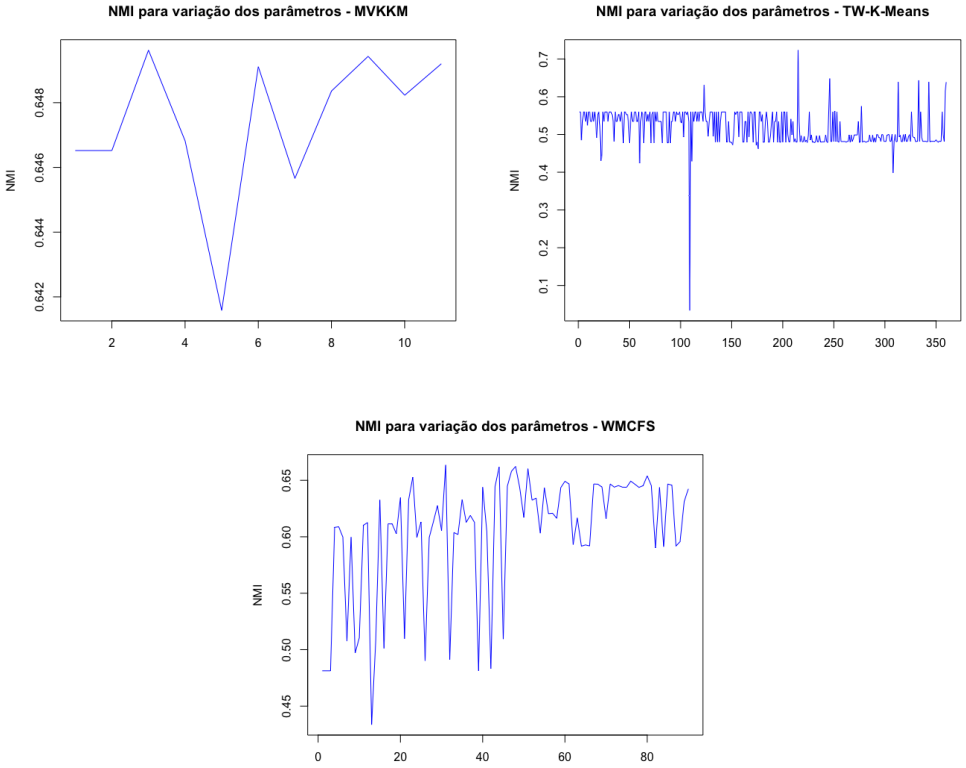


Figura 7 – Variação dos valores do NMI de acordo com a variação dos parâmetros de entrada dos modelos MVKKM, TW-K-Means e WMCFS

As seções seguintes detalharão as análises feitas a partir dos resultados encontrados nos experimentos realizados para todos os modelos em todos os conjuntos de dados.

5.1 Análise da média e desvio padrão

Com o objetivo de verificar a estabilidade dos modelos em relação aos diferentes pontos de partida utilizados, inicialmente foram calculados e analisados, a partir das médias e desvios padrão obtidos, os intervalos de confiança dos índices utilizados para medir a performance. Posteriormente, também foi utilizado o teste paramétrico *One-way* ANOVA (MILLER, 1997) para cada combinação de índice e conjunto de dados para verificar se existia diferença estatística significativa entre os modelos. No caso da hipótese nula ser rejeitada no teste *One-way* ANOVA, o teste *post-hoc Least Significant Difference (LSD)* (WILLIAMS; ABDI, 2010) foi aplicado em cada resultado para cada par de modelo com o objetivo de fazer uma comparação direta entre os algoritmos.

Os valores encontrados para as médias e desvios padrão são apresentados nas tabelas 15 a 18. Como pontuado, antes da execução dos testes estatísticos, foram calculados os

intervalos de confiança que podem ser vistos nos gráficos apresentados nas figuras 8 a 11. Os valores dos intervalos apresentados nesses gráficos estão detalhados no apêndice B. A partir desses dados, analisando apenas a estabilidade dos modelos, pode ser verificado que o modelo TW-K-Means, por possuir os maiores intervalos de confiança na maioria dos índices e conjuntos de dados, aparenta ser o modelo menos estável dentre todos. A exceção acontece quando todos os modelos apresentam um intervalo de confiança mais aberto, o que pode ser visto no conjunto *Reuters*, onde o modelo WVVHCM possui os intervalos de maior amplitude em todos os índices. Ainda em relação a estabilidade das execuções, também pode ser destacado que, com exceção dos conjuntos da base *Corel*, o MVKKM, se comparado aos outros modelos, apresenta os resultados mais estáveis para todos os índices. Já o modelo K-Means, apesar de apresentar as piores médias para todos os índices nos conjuntos da base *Corel*, apresenta os resultados mais estáveis nestes casos. Ao analisar as médias juntamente com os intervalos de confiança obtidos, os modelos WMCFS e WVVHCM se apresentam, em pelo menos metade dos conjuntos de dados e para todos os índices analisados, com os melhores resultados ou sem uma diferença significativa para a melhor média obtida. Os outros três modelos apresentam resultados piores nesse sentido.

A partir dos valores nas tabelas 15 a 18, foi realizado o teste *One-way* ANOVA com nível de significância de 0.05 para comparar os resultados dos modelos em cada combinação de índice e conjunto de dados. A hipótese nula, que indica não existir diferença significativa entre as médias obtidas, foi rejeitada para todos os casos. A partir dessa informação, foi executado o teste *post-hoc* LSD para comparar cada par de algoritmo dentro de cada índice e conjunto de dados com o objetivo de identificar os modelos que obtiveram as melhores médias. Os resultados do teste LSD, indicados por uma letra que pode variar entre 'a' e 'e', também podem ser vistos nas tabelas 15 a 18. No caso desse teste, a letra 'a' indica o melhor resultado, enquanto a letra 'e' indica o pior resultado. Os modelos que possuem a mesma letra não possuem uma diferença significativa entre suas médias. Caso um modelo possua mais de uma letra, ele não possui diferença significativa para todos os modelos que possuem pelo menos uma dessas letras.

Pode ser visto que, para metade dos conjuntos e em todos os índices avaliados, o modelo WVVHCM apresentou os melhores resultados ou uma média que não teve diferença significativa para o melhor resultado. O mesmo aconteceu para o modelo WMCFS, que em um dos índices (*Purity*) ainda obteve esse desempenho para 5 dos 8 conjuntos. Já o modelo MVKKM apresentou o melhor resultado ou média sem diferença significativa para o melhor em 2 conjuntos (AWA e *Segmentation*) para todos os índices, e em um conjunto (*Corel 1*) nos índices NMI e *Adjusted Rand*. O K-Means obteve o melhor resultado - ou média sem diferença significativa para o melhor - em todos os índices para o conjunto *Reuters*.

Para facilitar a visualização do desempenho dos algoritmos, a tabela 19 apresenta a

Tabela 15 – Média dos valores de NMI obtidos para 100 execuções dos modelos

Dados	K-Means	MVKKM	TW-k-Means	WMCFS	WVVHCM
Mfeat	$0.764(0.039)^{b}$	$0.744(0.018)^{c}$	$0.582(0.093)^{d}$	$\mathbf{0.841(0.039)^{a}}$	$\mathbf{0.848(0.025)^{a}}$
AWA	$0.193(0.007)^{bc}$	$\mathbf{0.235(0.004)^{a}}$	$0.194(0.036)^{b}$	$0.186(0.007)^{d}$	$0.189(0.007)^{cd}$
Phoneme	$0.641(0.077)^{b}$	$0.633(0.069)^{b}$	$0.488(0.171)^{c}$	$\mathbf{0.736(0.098)^{a}}$	$\mathbf{0.709(0.101)^{a}}$
Reuters	$\mathbf{0.371(0.032)^{a}}$	$0.344(0.029)^{b}$	$\mathbf{0.367(0.035)^{a}}$	$\mathbf{0.367(0.031)^{a}}$	$0.347(0.059)^{b}$
Segm	$0.620(0.033)^{b}$	$\mathbf{0.643(0.012)^{a}}$	$0.560(0.127)^{c}$	$0.615(0.052)^{b}$	$\mathbf{0.625(0.041)^{ab}}$
ALOI	$0.755(0.024)^{d}$	$0.770(0.029)^{bc}$	$0.764(0.046)^{c}$	$0.773(0.024)^{b}$	$\mathbf{0.784(0.024)^{a}}$
Corel 1	$0.515(0.011)^{d}$	$\mathbf{0.589(0.008)^{a}}$	$0.565(0.035)^{c}$	$0.569(0.015)^{c}$	$0.575(0.019)^{b}$
Corel 2	$0.321(0.041)^{d}$	$0.440(0.065)^{c}$	$\mathbf{0.491(0.107)^{ab}}$	$\mathbf{0.508(0.055)^{a}}$	$0.486(0.081)^{b}$

Tabela 16 – Média dos valores de F-Measure obtidos para 100 execuções dos modelos

Dados	K-Means	MVKKM	TW-k-Means	WMCFS	WVVHCM
Mfeat	$0.786(0.059)^{c}$	$0.781(0.027)^{c}$	$0.582(0.093)^{d}$	$0.824(0.062)^{b}$	$\mathbf{0.855(0.048)^{a}}$
AWA	$0.117(0.004)^{b}$	$\mathbf{0.123(0.003)^{a}}$	$0.102(0.014)^{e}$	$0.112(0.004)^{d}$	$0.115(0.004)^{c}$
Phoneme	$0.709(0.077)^{c}$	$0.735(0.049)^{b}$	$0.614(0.102)^{d}$	$\mathbf{0.784(0.083)^{a}}$	$\mathbf{0.768(0.080)^{a}}$
Reuters	$\mathbf{0.535(0.044)^{a}}$	$0.492(0.032)^{b}$	$\mathbf{0.535(0.043)^{a}}$	$\mathbf{0.533(0.040)^{a}}$	$0.503(0.056)^{b}$
Segm	$0.643(0.040)^{b}$	$\mathbf{0.661(0.018)^{a}}$	$0.588(0.085)^{d}$	$0.625(0.051)^{c}$	$0.639(0.044)^{bc}$
ALOI	$0.700(0.045)^{c}$	$0.710(0.051)^{bc}$	$\mathbf{0.720(0.064)^{ab}}$	$\mathbf{0.719(0.049)^{ab}}$	$\mathbf{0.733(0.049)^{a}}$
Corel 1	$0.695(0.017)^{c}$	$0.718(0.042)^{b}$	$\mathbf{0.735(0.051)^{a}}$	$\mathbf{0.729(0.047)^{ab}}$	$\mathbf{0.730(0.048)^{a}}$
Corel 2	$0.569(0.045)^{d}$	$0.653(0.059)^{c}$	$0.685(0.099)^{b}$	$\mathbf{0.709(0.070)^{a}}$	$0.682(0.083)^{b}$

Tabela 17 – Média dos valores de Adjusted Rand obtidos para 100 execuções dos modelos

Dados	K-Means	MVKKM	TW-k-Means	WMCFS	WVVHCM
Mfeat	$0.665(0.067)^{c}$	$0.650(0.029)^{c}$	$0.427(0.099)^{d}$	$0.729(0.086)^{b}$	$\mathbf{0.770(0.061)^{a}}$
AWA	$0.021(0.002)^{b}$	$\mathbf{0.026(0.002)^{a}}$	$0.010(0.006)^{d}$	$0.020(0.002)^{c}$	$0.020(0.002)^{c}$
Phoneme	$0.523(0.119)^{c}$	$0.573(0.069)^{b}$	$0.384(0.153)^{d}$	$\mathbf{0.640(0.123)^{a}}$	$\mathbf{0.614(0.120)^{a}}$
Reuters	$\mathbf{0.253(0.039)^{a}}$	$0.218(0.027)^{c}$	$\mathbf{0.254(0.038)^{a}}$	$\mathbf{0.252(0.034)^{a}}$	$0.231(0.056)^{b}$
Segm	$0.478(0.044)^{b}$	$\mathbf{0.517(0.023)^{a}}$	$0.414(0.115)^{d}$	$0.456(0.071)^{c}$	$0.470(0.060)^{bc}$
ALOI	$0.578(0.053)^{b}$	$0.584(0.062)^{b}$	$0.587(0.077)^{b}$	$0.588(0.062)^{b}$	$\mathbf{0.609(0.066)^{a}}$
Corel 1	$0.451(0.009)^{b}$	$\mathbf{0.502(0.033)^{a}}$	$\mathbf{0.492(0.057)^{a}}$	$\mathbf{0.496(0.039)^{a}}$	$\mathbf{0.498(0.042)^{a}}$
Corel 2	$0.286(0.041)^{d}$	$0.384(0.073)^{c}$	$\mathbf{0.446(0.117)^{ab}}$	$\mathbf{0.461(0.076)^{a}}$	$0.436(0.094)^{b}$

posição média de cada modelo em cada índice externo. Essa tabela também apresenta a média das posições dos modelos em todos os índices. A partir dessas informações, pode-se considerar que os modelos WVVHCM e WMCFS obtiveram resultados superiores, com uma clara vantagem para o algoritmo WVVHCM, que, além de obter o melhor

Tabela 18 – Média dos valores de Purity obtidos para 100 execuções dos modelos

Dados	K-Means	MVKKM	TW-k-Means	WMCFS	WVVHCM
Mfeat	0.771(0.066)^{c}	0.775(0.030)^{c}	0.560(0.090)^{d}	0.813(0.066)^{b}	**0.844(0.054)^{a}**
AWA	0.113(0.005)^{b}	**0.121(0.004)^{a}**	0.100(0.018)^{d}	0.108(0.004)^{c}	0.111(0.005)^{b}
Phoneme	0.671(0.097)^{c}	0.717(0.044)^{b}	0.599(0.115)^{d}	**0.745(0.107)^{a}**	**0.736(0.095)^{ab}**
Reuters	**0.532(0.042)^{a}**	0.504(0.027)^{b}	**0.534(0.041)^{a}**	**0.530(0.037)^{a}**	0.508(0.055)^{b}
Segm	0.617(0.040)^{b}	**0.671(0.016)^{a}**	0.591(0.096)^{c}	0.613(0.053)^{b}	0.621(0.049)^{b}
ALOI	0.674(0.048)^{c}	0.685(0.051)^{bc}	**0.699(0.066)^{ab}**	**0.696(0.050)^{ab}**	**0.708(0.052)^{a}**
Corel 1	0.691(0.018)^{c}	0.706(0.048)^{b}	**0.717(0.063)^{ab}**	**0.720(0.052)^{a}**	**0.716(0.055)^{ab}**
Corel 2	0.568(0.044)^{d}	0.649(0.058)^{c}	**0.689(0.094)^{ab}**	**0.708(0.069)^{a}**	0.680(0.083)^{b}

resultado em 3 dos 4 índices calculados, obteve a melhor posição geral entre todos os modelos. É importante destacar que o desempenho do algoritmo WMCFS, que obteve melhor resultado apenas em um dos índices, só foi possível pelo fato do modelo utilizar informação supervisionada para ajustar a sua performance para as execuções realizadas. Como explicado na seção 4.2 do capítulo 4, os índices utilizados para avaliação, em geral, fazem uma comparação entre os *clusters* e as classificações do elementos. Desta forma, por utilizarem os rótulos dos elementos para ajuste do desempenho, os algoritmos MVKKM, TW-k-Means e WMCFS levam uma vantagem que não pode ser obtida em um contexto de aprendizado não supervisionado. Essa vantagem ocorre tanto na análise da estabilidade como na análise dos melhores resultados e médias.

Tabela 19 – Posições dos modelos na análise pela média e desvio padrão

Modelo	NMI	F-Measure	Adj. Rand	Purity	Posição média
WVVHCM	**1,7500**	**1,6875**	**1,6875**	1,6250	**1,687500**
WMCFS	1,8750	1,8750	1,7500	**1,5000**	1,750000
MVKKM	1,9375	2,0625	2,0000	2,0625	2,015625
K-Means	2,6875	2,6250	2,3750	2,6250	2,578125
TW-k-Means	2,5625	2,8125	2,6875	2,5625	2,656250

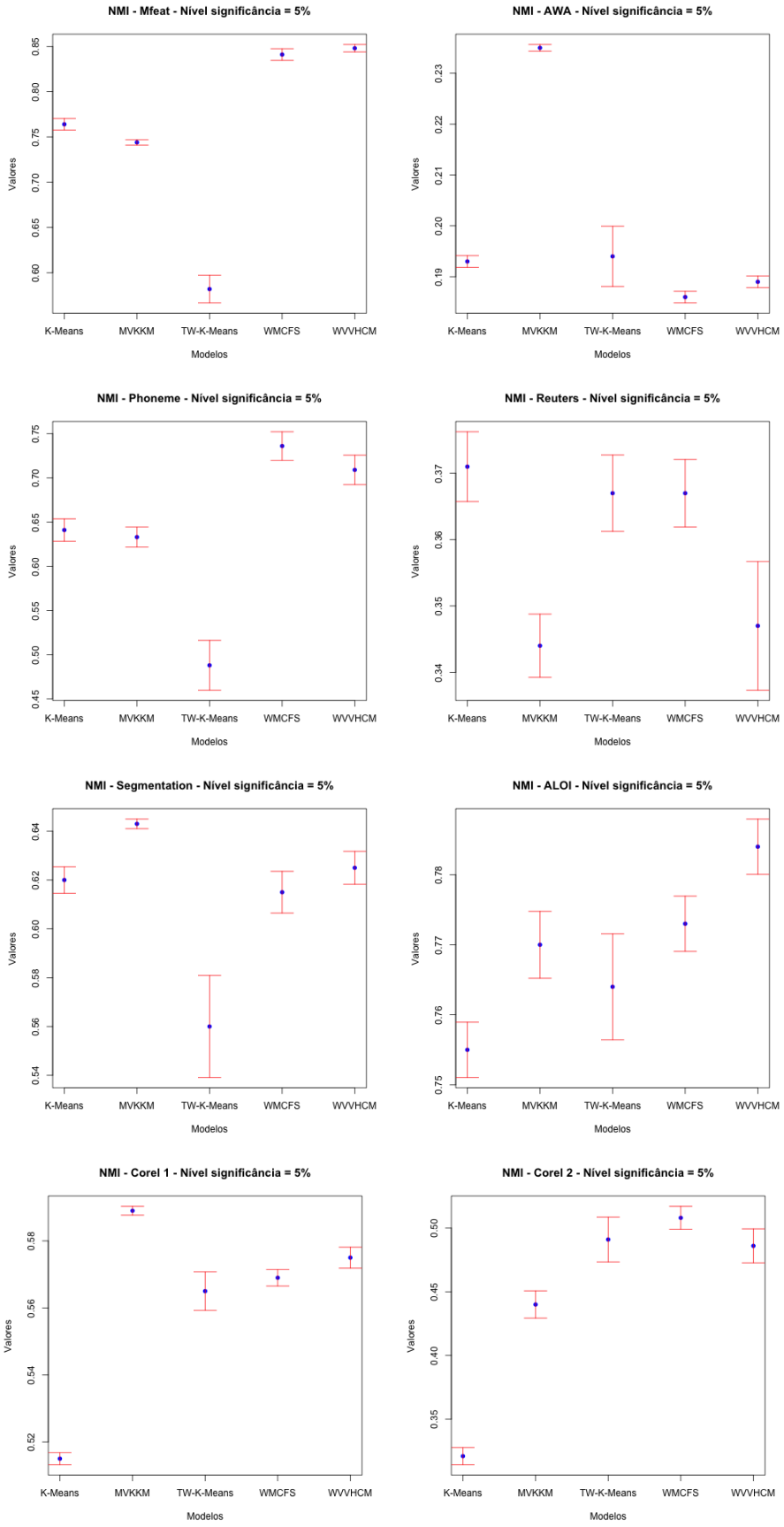


Figura 8 – Diagrama com comparação do intervalo de confiança do índice NMI com $alpha = 0.05$

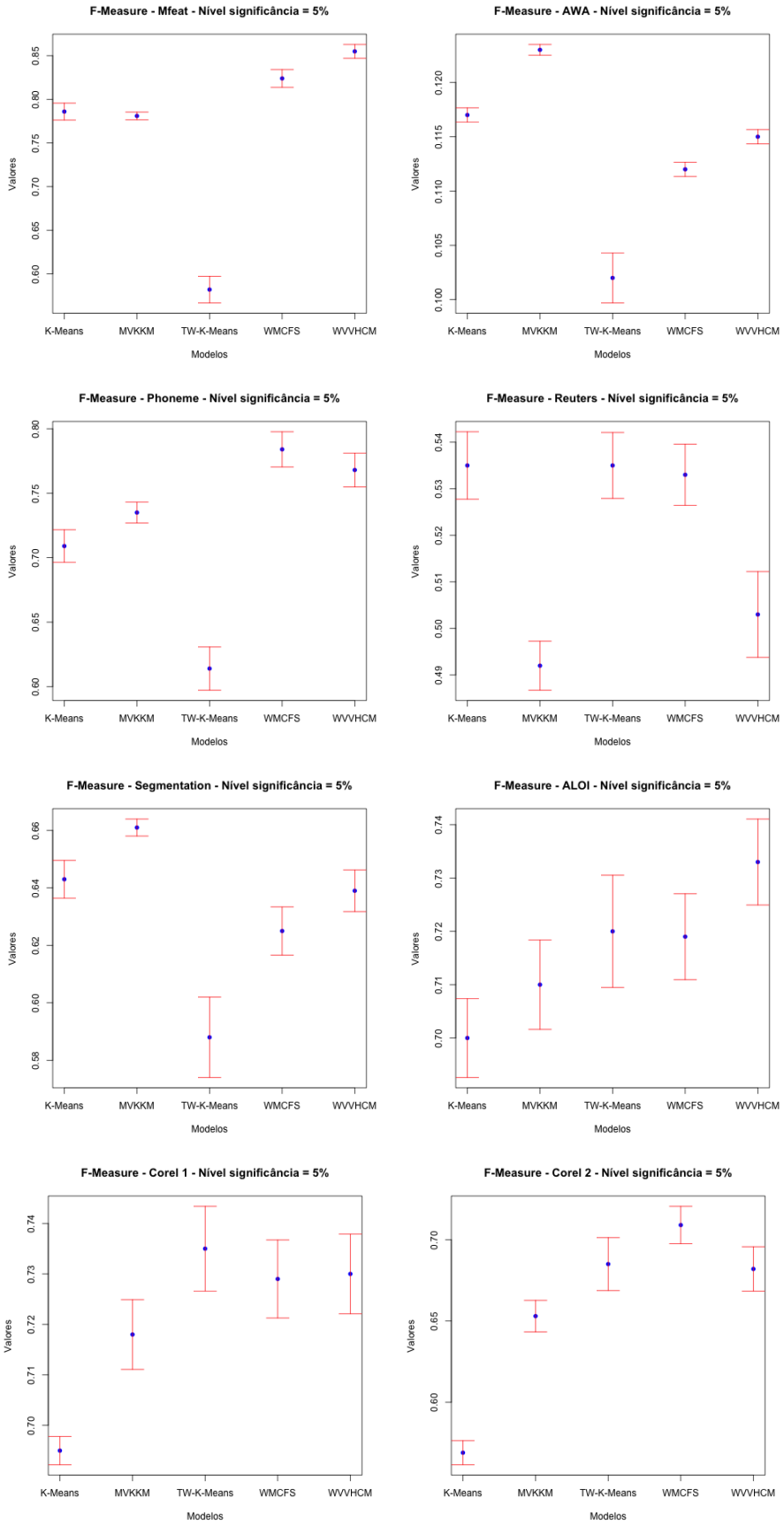


Figura 9 – Diagrama com comparação do intervalo de confiança do índice F-Measure com $alpha = 0.05$

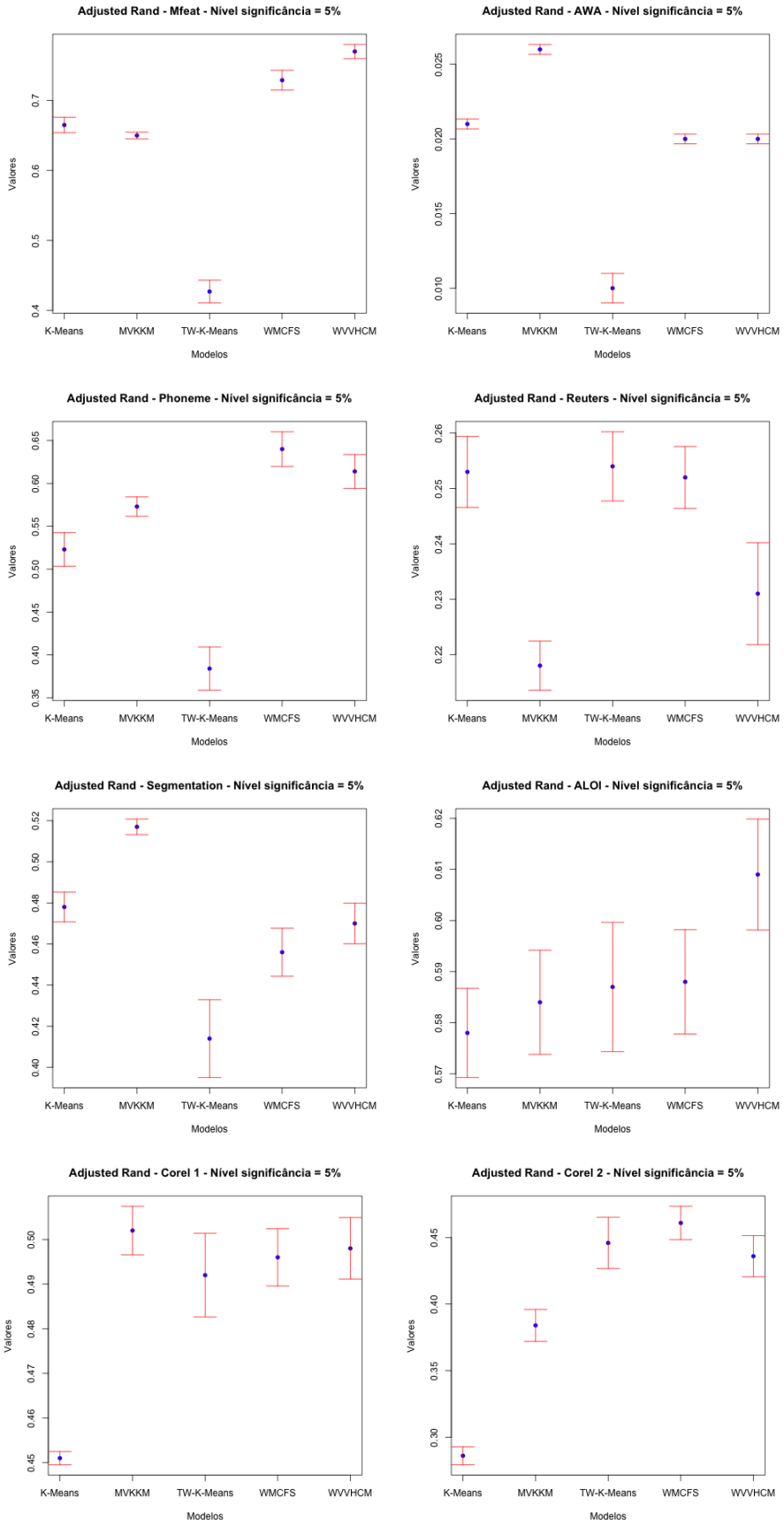


Figura 10 – Diagrama com comparação do intervalo de confiança do índice Adjusted Rand com $alpha = 0.05$

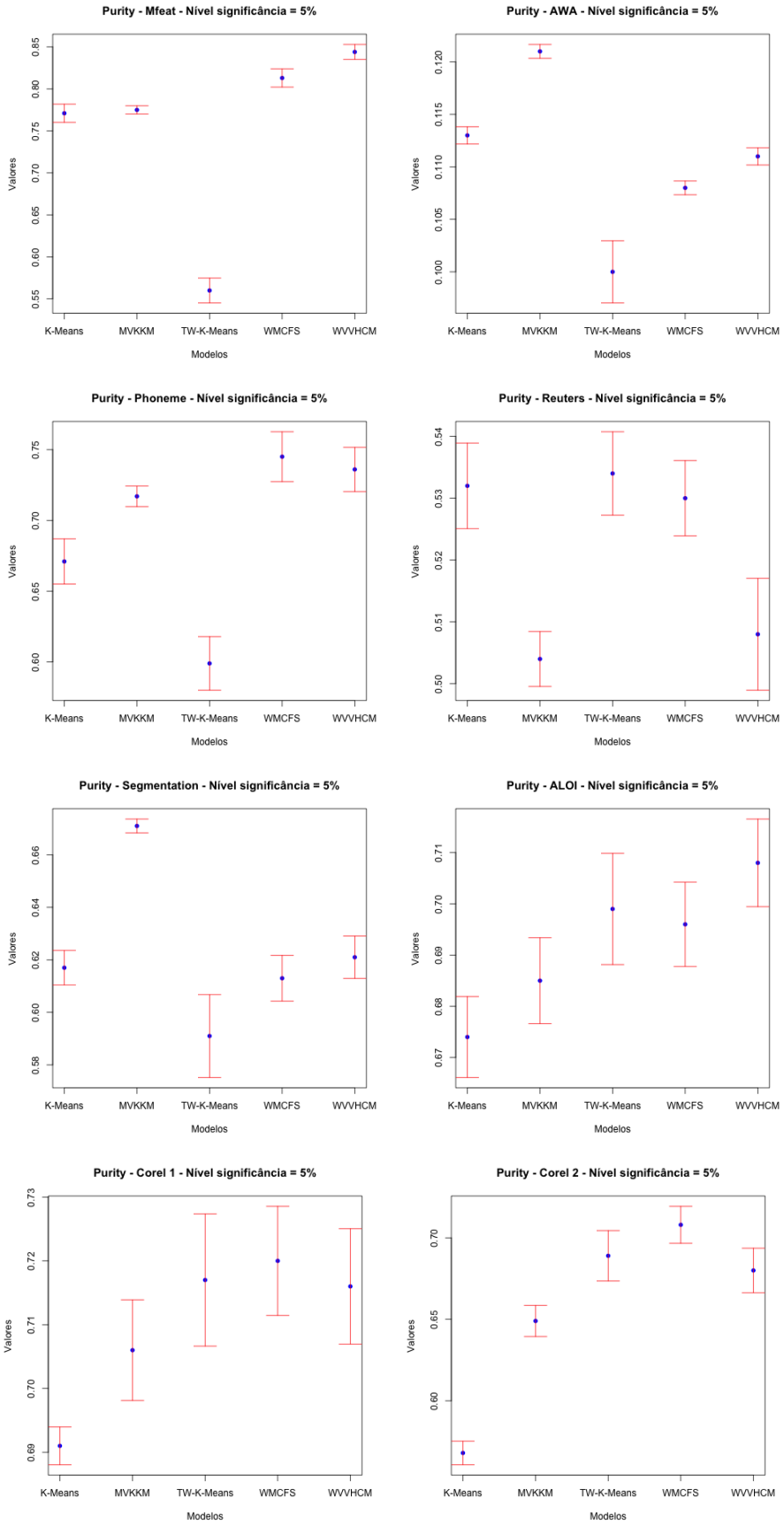


Figura 11 – Diagrama com comparação do intervalo de confiança do índice Purity com $alpha = 0.05$

5.2 Análise do melhor resultado selecionado

Para os melhores resultados selecionados de cada algoritmo, foi feita uma comparação direta entre os modelos em todos os índices. Posteriormente, foi utilizado, para cada índice, o teste de *Friedman* para múltiplos conjuntos de dados e o teste *post-hoc* de *Nemenyi* a partir dos resultados obtidos (DEMSAR, 2006).

Analisando o melhor resultado obtido de acordo com o critério interno de cada modelo, é possível checar na tabela 20 que o modelo MVKKM obteve o melhor índice NMI em 3 dos 8 conjuntos existentes. Nesse mesmo índice, os modelos WMCFS e WVVHCM obtiveram, cada um, o melhor resultado para 2 conjuntos. Já o modelo TW-K-Means obteve melhor resultado para apenas um conjunto. A tabela 21 apresenta os valores obtidos para o índice *F-Measure*, onde se destaca o modelo WVVHCM possuindo resultado superior em 3 dos 8 conjuntos de dados. Os modelos TW-K-Means e WMCFS obtiveram melhor resultado em 2 conjuntos e o K-Means em um conjunto. Os modelos K-Means, WMCFS e WVVHCM repetem seus resultados para o índice *Adjusted Rand*, que pode ser visto na tabela 22. Nesse índice, os modelos MVKKM e TW-K-Means se igualam ao K-Means em relação a quantidade de resultados com melhor valor. Já para o índice *Purity*, detalhado na tabela 23, os modelos MVKKM, WMCFS e WVVHCM obtiveram, cada um, o melhor resultado em 2 conjuntos, enquanto os modelos K-Means e TW-K-Means apresentam melhor resultado em um conjunto cada. De forma geral, podemos ver que em dois (*F-Measure* e *Adjusted Rand*) dos quatro índices calculados, o modelo proposto WVVHCM obteve melhor performance se comparado aos outros modelos. O mesmo aconteceu com o modelo MVKKM para o índice NMI. Já para o índice *Purity*, aconteceu um empate entre os modelos MVKKM, WMCFS e WVVHCM. É importante destacar, também, que o modelo WVVHCM mesmo quando não obteve o melhor resultado em um conjunto de dados, conseguiu um valor de índice competitivo, ficando com o segundo melhor resultado em quatro conjuntos nos índices NMI e F-Measure, em três conjuntos no índice *Adjusted Rand* e em dois conjuntos no índice *Purity*. Dessa forma, o modelo WVVHCM ficou entre os dois melhores modelos na maioria dos conjuntos em todos os índices avaliados. A performance superior desse modelo pode ser vista na tabela 24, que apresenta a posição média dos algoritmos em cada um dos índices externos calculados. Nessa tabela também é apresentada a posição média geral de cada modelo em todos os índices, mostrando mais uma vez uma clara vantagem para o modelo WVVHCM.

Fazendo uma análise com foco nos conjuntos de dados, podemos notar que para o conjunto AWA, que possui uma grande quantidade de variáveis (10934 atributos diferentes), os modelos que fazem o cálculo automático dos pesos das variáveis não obtiveram um bom resultado. Nesse caso, o K-Means, que não considera qualquer tipo de peso na formação dos grupos, e o MVKKM, que calcula apenas os pesos das visões no processo de agrupamento, foram mais bem sucedidos. Já para os conjuntos *Mfeat, Phoneme, Reuters* e *Corel 2*, os modelos que obtiveram melhores resultados são os que calculam automati-

camente os pesos das visões e das variáveis. Isso pode indicar que existem variações de importância que podem influenciar o resultado final tanto nas visões como nas variáveis dentro das visões desses conjuntos de dados. Nos conjuntos *Segmentation, ALOI* e *Corel 1*, os melhores resultados variaram entre o modelo MVKKM, que pondera apenas as visões e os modelos que poderam visões e variáveis durante o processo de agrupamento. Esse fato pode indicar que podem existir visões com menor importância, porém, dentre as visões com mais importância, a variação da relevância dos atributos não é crucial para obtenção da partição final.

Tabela 20 – Valores do NMI para o melhor resultado das 100 execuções dos modelos

Dados	K-Means	MVKKM	TW-k-Means	WMCFS	WVVHCM
Mfeat	0.805	0.736	0.627	**0.920**	0.890
AWA	0.220	**0.239**	0.234	0.197	0.205
Phoneme	0.771	0.780	0.657	0.790	**0.798**
Reuters	0.369	0.375	0.362	0.369	**0.421**
Segm	0.586	**0.619**	0.605	0.605	0.610
ALOI	0.765	0.774	0.674	**0.780**	0.770
Corel 1	0.509	**0.584**	0.561	0.566	0.575
Corel 2	0.322	0.534	**0.598**	0.543	0.557

Tabela 21 – Valores do F-Measure para o melhor resultado das 100 execuções dos modelos

Dados	K-Means	MVKKM	TW-k-Means	WMCFS	WVVHCM
Mfeat	0.834	0.757	0.599	**0.963**	0.941
AWA	**0.130**	0.120	0.117	0.116	0.126
Phoneme	0.855	0.858	0.726	0.858	**0.864**
Reuters	0.511	0.501	0.502	0.511	**0.532**
Segm	0.632	0.634	0.582	0.582	**0.646**
ALOI	0.698	0.710	0.582	**0.716**	0.678
Corel 1	0.669	0.698	**0.710**	0.687	0.699
Corel 2	0.560	0.740	**0.785**	0.755	0.759

Para demonstrar a importância do cálculo das relevâncias das tabelas e atributos no processo de agrupamento de dados *multi-view*, os valores finais obtidos para os pesos de cada visão e cada variável do modelo proposto para o conjunto *Phoneme* são apresentados,

Tabela 22 – Valores do Adjusted Rand para o melhor resultado das 100 execuções dos modelos

Dados	K-Means	MVKKM	TW-k-Means	WMCFS	WVVHCM
Mfeat	0.722	0.628	0.440	**0.921**	0.879
AWA	**0.029**	0.024	0.019	0.019	0.020
Phoneme	0.724	0.735	0.540	0.740	**0.751**
Reuters	0.227	0.207	0.223	0.226	**0.253**
Segm	0.460	0.448	0.414	0.414	**0.466**
ALOI	0.570	0.561	0.396	**0.571**	0.535
Corel 1	0.446	**0.485**	0.464	0.465	0.465
Corel 2	0.289	0.494	**0.574**	0.511	0.522

Tabela 23 – Valores de Purity para o melhor resultado das 100 execuções dos modelos

Dados	K-Means	MVKKM	TW-k-Means	WMCFS	WVVHCM
Mfeat	0.826	0.773	0.538	**0.964**	0.942
AWA	**0.126**	0.117	0.123	0.110	0.119
Phoneme	0.852	0.857	0.728	0.859	**0.864**
Reuters	0.528	0.514	0.519	0.527	**0.553**
Segm	0.554	**0.640**	0.604	0.604	0.592
ALOI	0.675	**0.716**	0.560	0.708	0.641
Corel 1	0.687	0.680	0.675	**0.688**	0.685
Corel 2	0.563	0.738	**0.788**	0.753	0.758

Tabela 24 – Posições dos modelos na análise pelo melhor resultado

Modelo	NMI	F-Measure	Adj. Rand	Purity	Posição média
WVVHCM	**2,1250**	**1,8750**	**2,0625**	2,5000	**2,140625**
WMCFS	2,7500	2,9375	2,6875	**2,4375**	2,703125
MVKKM	2,2500	3,1875	3,1250	3,2500	2,953125
K-Means	4,0625	3,3125	3,0000	3,1250	3,375000
TW-k-Means	3,8125	3,6875	4,1250	3,6875	3,828125

respectivamente, nas figuras 12 e 13. Na primeira figura, é possível verificar que houve uma grande variação entre os pesos das três visões existentes. O mesmo pode ser visto em

relação às variáveis na segunda figura, onde é possível observar uma grande variação dos pesos dentro de cada tabela. Como o modelo WVVHCM obteve os melhores resultados em todos os índices calculados a partir dos grupos gerados para o conjunto *Phoneme*, é possível dizer que caso as visões ou as variáveis dessas visões não fossem ponderadas para a definição do partição final, a performance do algoritmo poderia ser bastante influenciada.

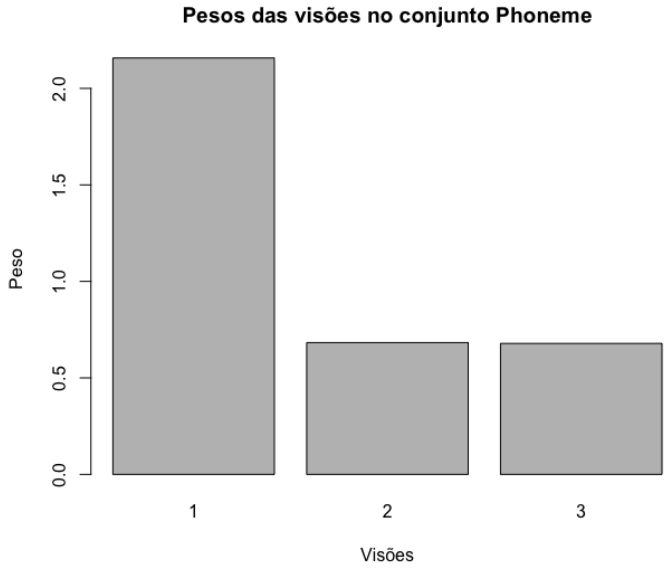


Figura 12 – Variação dos pesos das visões no conjunto de dados Phoneme

Para checar se existe uma diferença estatística significativa entre os modelos dentro dos melhores resultados de cada índice, foi realizado o teste não paramétrico de *Friedman* por bloco (conjunto de dados) com nível de confiança de 95%. A hipótese nula do teste, que indica que não existe diferença entre os modelos avaliados, foi rejeitada apenas para o índice NMI. Apesar disso, foi calculado o teste *post-hoc* de *Nemenyi* para todos os índices e os resultados encontram-se nos gráficos da figura 14. Nesses gráficos, os algoritmos estão ordenados da esquerda para direita, onde o algoritmo mais a esquerda possui melhor resultado. Uma linha acima de cada gráfico indica a diferença crítica que precisa ser atingida para que exista uma diferença estatística sifnificativa entre os modelos. Também são mostradas conexões entre os algoritmos que não apresentam diferença significativa em seus resultados. Desta forma, é possível verificar que o teste de *Nemenyi* confirma a igualdade do teste de *Friedman* nos índices *F-Measure, Adjusted Rand* e *Purity*, indicando, inclusive, que não existe diferença significativa para o índice NMI. Sendo assim, pode-se verificar que, apesar do algoritmo WVVHCM se destacar entre os outros modelos tanto em uma comparação direta, quanto checando os resultados dos testes *Nemenyi*, nenhum modelo apresenta uma diferença estatística significativa em seus resultados.

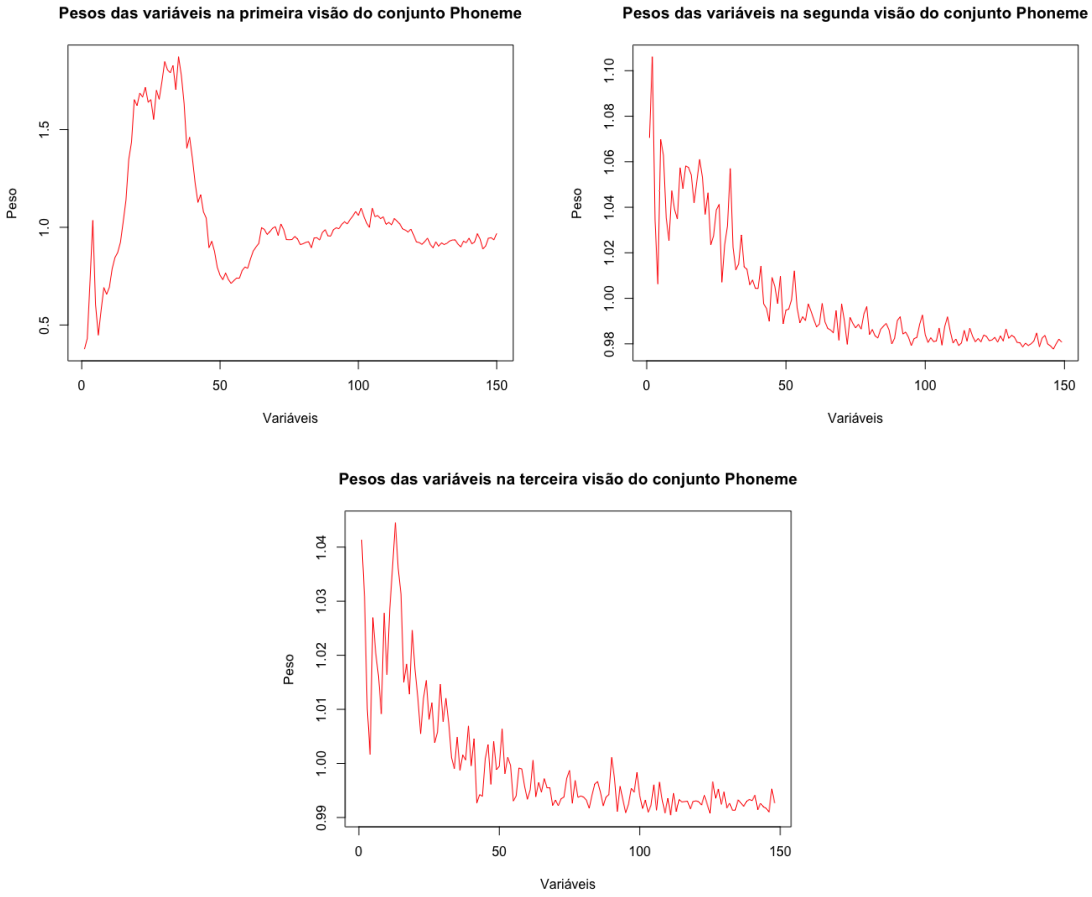


Figura 13 – Variação dos pesos das variáveis em cada visão no conjunto de dados Phoneme

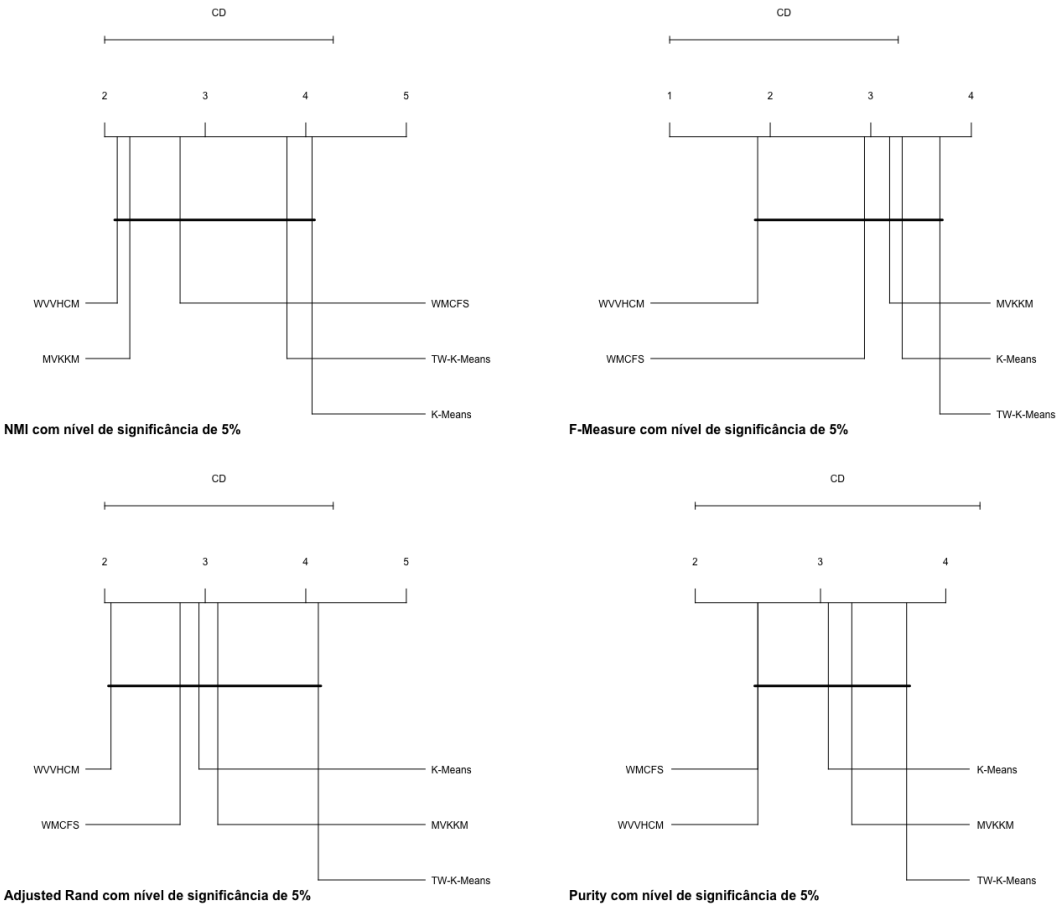


Figura 14 – Diagrama com comparação entre modelos usando teste post-hoc Nemenyi com $\alpha = 0.05$

6 CONCLUSÃO

Este trabalho apresentou o modelo *Multi-View Hard C-Means With Automated Weighting Of Views And Variables* (WVVHCM), um algoritmo de agrupamento *multi-view* exclusivo c-means que calcula de forma automática e simutânea as relevâncias das visões e das variáveis com o objetivo de selecionar os melhores recursos para o agrupamento dos elementos. A partir de uma solução inicial, o modelo proposto fornece um agrupamento final otimizando uma função objetivo. Essa otimização é feita de forma iterativa em quatro etapas: cálculo dos representantes, cálculo dos pesos das variáveis, cálculo dos pesos das visões e alocação dos objetos aos grupos.

A performance do algoritmo WVVHCM foi comparada com quatro outros modelos: O K-Means tradicional, que não calcula qualquer tipo de peso para visões e variáveis dos conjuntos de dados, o MVKKM, um algoritmo que calcula apenas as relevâncias das visões durante o processo de agrupamento, e os modelos TW-K-Means e WMCFS, que, assim como o modelo proposto, computam as relevâncias não só das visões, mas também das variáveis das visões durante o agrupamento dos dados. Os experimentos foram realizados utilizando diversos conjuntos *benchmark* de dados *multi-view* e os resultados obtidos foram comparados sob duas perspectivas: análise dos melhores resultados dos modelos em cada conjunto de dados e análise da média, desvio padrão e intervalo de confiança de cada modelo em cada conjunto de dados para todas as execuções. Testes paramétricos e não paramétricos foram executados para verificar se as diferenças observadas nos resultados finais eram estatisticamente significativas.

Apesar dos testes estatísticos não indicarem uma diferença significativa entre os melhores resultados de cada modelo, foi possível observar que o modelo proposto (WVVHCM) e um dos modelos que calculam as relevâncias dos pesos das visões e das variáveis (WMCFS) obtiveram resultados superiores ao K-Means e ao MVKKM na grande maioria dos conjuntos e índices avaliados. O único conjunto de dados onde pode ser vista uma clara vantagem dos modelos K-Means e MVKKM possui uma quantidade total de dimensões muito grande (acima de 10 mil variáveis), o que pode indicar uma dificuldade em encontrar variáveis relevantes nesses casos. Também foi visto que em três conjuntos de dados o modelo MVKKM, que só computa a relevância das visões, obteve resultados competitivos com os modelos que computam os pesos das visões e das variáveis. Pode-se concluir que nesses conjuntos não existem variáveis com pouca relevância nas visões de maior relevância e/ou não existem variáveis relevantes nas visões com pouca relevância. Dentre os três modelos que computam pesos das visões e variáveis, também foi possível perceber que os modelos WVVHCM e WMCFS possuem um comportamento mais estável quando comparados ao modelo TW-K-Means.

A performance superior do algoritmo WVVHCM indica uma vantagem clara em relação aos modelos MVKKM, TW-K-Means e WMCFS, porque, assim como o K-Means, além de receber como entrada a quantidade de *clusters*, a performance foi obtida sem a necessidade de nenhum outro conhecimento prévio sobre o conjunto de dados a ser agrupado. Os modelos MVKKM, TW-K-Means e WMCFS precisam de uma etapa anterior para ajuste de parâmetros de entrada que, assim como foi mostrado, pode influenciar consideravelmente os resultados desses algoritmos. Como a forma conhecida para ajuste desses parâmetros necessita do conhecimento prévio da classificação dos elementos dos conjuntos de dados, que não é o caso nas aplicações da vida real no contexto não supervisionado, a escolha dos valores desses parâmetros é propensa a erros. Desta forma, o algoritmo proposto neste trabalho deve ser considerada a melhor escolha para realizar agrupamento de conjuntos descritos por múltiplas tabelas.

Como trabalhos futuros, serão realizados mais testes no modelo apresentado em novos conjuntos de dados *multi-view* que possuem diferentes características: alta dimensionalidade, visões com todas as variáveis possuindo máxima relevância, entre outras. A partir de uma avaliação desses resultados, será possível entender o contexto onde o modelo pode ser melhor aplicado. Também é possível criar variações do algoritmo proposto para obtenção dos pesos das visões e variáveis a partir de técnicas de inteligência de enxames ou modificar a função objetivo para que o cálculo das dissimilaridades seja feito utilizando alguma função de *kernel*. Além disso, pode-se criar derivações do modelo WVVHCM para que os pesos obtidos para as visões e variáveis sejam locais. Ou seja, cada *cluster* do agrupamento pode possuir um peso diferente para cada visão e suas variáveis. Sendo assim, ainda é possível dar continuidade ao trabalho com o objetivo de contribuir na pesquisa de algoritmos de agrupamento *multi-view*.

REFERÊNCIAS

BICKEL, S.; SCHEFFER, T. Multi-view clustering. *Proceedings of the Fourth IEEE International Conference on Data Mining*, v. 31, p. 19–26, 2004.

BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 0387310738.

CALDERS, T.; ESPOSITO, F.; HÜLLERMEIER, E.; MEO, R. *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings*. [S.l.]: Springer Berlin Heidelberg, 2014.

CARVALHO, F. A. T. de; LECHEVALLIER, Y.; MELO, F. M. de. Partitioning hard clustering algorithms based on multiple dissimilarity matrices. *Pattern Recognition*, v. 45, p. 447–464, 2012.

CHEN X., X. X. Y. Y. H. J. Tw-k-means: Automated two-level variable weighting clustering algorithm for multi-view data. *IEEE Transactions on Knowledge and Data Engineering*, v. 25, p. 932–944, 2013.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, JMLR.org, v. 7, p. 1–30, dez. 2006. ISSN 1532-4435.

DESARBO, W.; CARROLL, J.; CLARK, L.; GREEN, P. Synthesized clustering: A method for amalgamating clustering bases with differential weighting variables. *Psychometrika*, v. 49, n. 1, p. 57–78, 1984.

DIDAY, E.; SIMON, J. Clustering analysis, in: K.s. fu (ed.). *Digital Pattern Classification*, Springer, Berlin, v. 10, p. 47–94, 1976.

FERREIRA, M. R. P.; CARVALHO, F. D. A. T. D. Kernel fuzzy c-means with automatic variable weighting. *Fuzzy Sets Syst.*, Elsevier North-Holland, Inc., v. 237, p. 1–46, fev. 2014. ISSN 0165-0114.

FOWLKES E. B., M. C. L. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, v. 383, n. 78, p. 553–569, 1983.

HALKIDI, M.; VAZIRGIANNIS, M. Clustering validity assessment: finding the optimal partitioning of a data set. In: *Proceedings 2001 IEEE International Conference on Data Mining*. [S.l.: s.n.], 2001. p. 187–194.

HAYKIN, S. Neural networks - a compreensive foundation. *Prentice-Hall, New Jersey, 2nd edition*, 1999.

HIGHAM, D. J.; KALNA, G.; KIBBLE, M. Spectral clustering and its use in bioinformatics. *J. Comput. Appl. Math.*, Elsevier Science Publishers B. V., v. 204, n. 1, p. 25–37, jul. 2007. ISSN 0377-0427.

HUBERT, L.; ARABIE, P. Comparing partitions. *Journal of classification*, v. 2, n. 1, p. 193–218, 1985.

JAIN, A. K. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, v. 31, p. 651–666, 2010.

JIANG F. QIU, S. Y. B.; WANG, L. Evolutionary multi-objective optimization for multi-view clustering. *IEEE Congress on Evolutionary Computation (CEC)*, 2016.

JOULIN, A.; BACH, F. R.; PONCE, J. Discriminative clustering for image co-segmentation. In: *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*. [S.l.: s.n.], 2010. p. 1943–1950.

KAUFMAN, L.; ROUSSEEUW, P. *Finding Groups in Data: an introduction to cluster analysis*. [S.l.]: Wiley, 1990.

KING, R. S. *Cluster Analysis and Data Mining: An Introduction*. USA: Mercury Learning & Information, 2014. ISBN 1938549384, 9781938549380.

KUMAR, A.; DAUMé, H. A co-training approach for multi-view spectral clustering. In: . [S.l.: s.n.], 2011. p. 393–400.

LINTAS, A.; ROVETTA, S.; VERSCHURE, P.; VILLA, A. *Artificial Neural Networks and Machine Learning – ICANN 2017: 26th International Conference on Artificial Neural Networks, Alghero, Italy, September 11-14, 2017, Proceedings*. [S.l.]: Springer International Publishing, 2017.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, p. 281–297, 1967.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008. ISBN 0521865719, 9780521865715.

MEILA, M. Comparing clusterings by the variation of information. *COLT*, v. 3, p. 173–187, 2003.

MILLER, R. *Beyond ANOVA: Basics of Applied Statistics*. [S.l.]: Taylor & Francis, 1997. (Chapman & Hall/CRC Texts in Statistical Science).

MOREY, L.; AGRESTI, A. An adjustment to the rand statistic for chance agreement. *The Classification Society Bulletin*, v. 5, p. 9–10, 1981.

RAND, W. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, v. 336, n. 66, p. 846– 850, 1971.

RENDON, E.; ABUNDEZ, I. M.; GUTIERREZ, C.; ZAGAL, S. D.; ARIZMENDI, A.; QUIROZ, E. M.; ARZATE, H. E. A comparison of internal and external cluster validation indexes. In: . [S.l.]: World Scientific and Engineering Academy and Society (WSEAS), 2011. p. 158–163.

REZA, G.; NASIR, S. M.; HAMIDAH, I.; NORWATI, M. A survey: Clustering ensembles techniques. *Proceedings of World Academy of Science, Engineering and Technology*, v. 38, p. 644–653, 2009.

RIJISBERGEN, C. van. Information retrieval. *Butterworth-Heinemann*, 1979.

SAEMAIL P. W. GALLAGHER, J. M. L. V. R. de; MALAVE, V. L. Multi-view kernel construction. *Mach Learn*, v. 79, p. 47, 2010.

SCHOLKOPF, B.; SMOLA, A.; MULLER, K.-R. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, MIT Press, Cambridge, MA, USA, v. 10, n. 5, p. 1299–1319, jul. 1998. ISSN 0899-7667.

STREHL, A.; GHOSH, J. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, v. 3, p. 583–617, 2002.

TAN M. STEINBACH, V. K. P.-N. *Introduction to Data Mining*. [S.l.]: Addison-Wesley, 2005.

TRIVEDI, A.; RAI, P.; III, H. D.; DUVALL, S. Multiview clustering with incomplete views. *NIPS 2010: Workshop on Machine Learning for Social Computing*, 2010.

TZORTZIS, G.; LIKAS, A. Convex mixture models for multi-view clustering. *Proc. 19th Int. Conf. Artif. Neural Netw*, p. 205–214, 2009.

TZORTZIS, G.; LIKAS, A. The global kernel k -means algorithm for clustering in feature space. *IEEE transactions on neural networks*, v. 20 7, p. 1181–94, 2009.

TZORTZIS, G.; LIKAS, A. Kernel-based weighted multi-view clustering. *Proceedings of the 12th International Conference on Data Mining*, p. 675–684, 2012.

WAGNER, S.; WAGNER, D. *Comparing clusterings: an overview*. [S.l.]: Univ., Fak. für Informatik, 2007. (Interner Bericht).

WAHID, X. G. A.; ANDREAE, P. Multi-objective multi-view clustering ensemble based on evolutionary approach. *IEEE Congress on Evolutionary Computation (CEC)*, 2015.

WANG, H.; NIE, F.; HUANG, H. Multi-view clustering and feature learning via structured sparsity. *Proceedings of the 30th International Conference on Machine Learning*, R. Oldenbourg Verlag, Mtlnchen, Wien, p. 352–360, 2013.

WANG, Y.; CHEN, L. Multi-view fuzzy clustering with minimax optimization for effective clustering of data from multiple sources. *Expert Systems with Applications*, v. 72, p. 457–466, 2016.

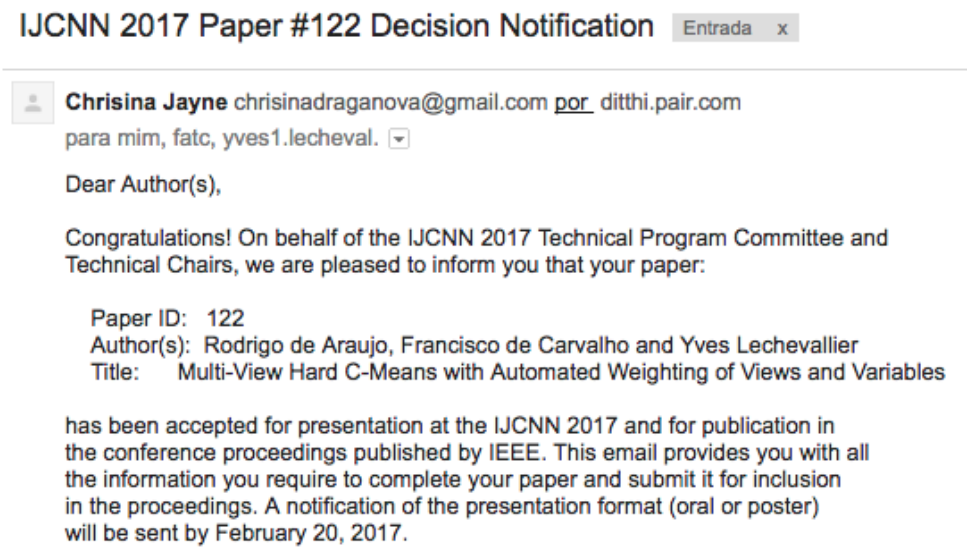
WILLIAMS, L. J.; ABDI, H. *Fisher's Least Significant Difference (LSD) Test*. [S.l.]: SAGE Publications, Inc, 2010. (Encyclopedia of Research Design).

XIE, X.; SUN, S. Multi-view clustering ensembles. *ICMLC*, p. 51–56, 2013.

XU, Y. M.; WANG, C. D.; LAI, J. H. Weighted multi-view clustering with feature selection. *Pattern Recognition*, v. 53, p. 25–35, 2016.

YAMANISHI, Y.; VERT, J. P.; KANEHISA, M. Protein network inference from multiple genomic data: a supervised approach. *Bioinformatics*, v. 20, n. 1, p. 363–370, 2004.

APÊNDICE A – ARTIGO PUBLICADO NA CONFERÊNCIA IJCNN 2017



IJCNN 2017 Paper #122 Decision Notification Entrada x

Chrisina Jayne chrisinadraganova@gmail.com por ditthi.pair.com
para mim, fatc, yves1.lecheval.

Dear Author(s),

Congratulations! On behalf of the IJCNN 2017 Technical Program Committee and Technical Chairs, we are pleased to inform you that your paper:

Paper ID: 122
Author(s): Rodrigo de Araujo, Francisco de Carvalho and Yves Lechevallier
Title: Multi-View Hard C-Means with Automated Weighting of Views and Variables

has been accepted for presentation at the IJCNN 2017 and for publication in the conference proceedings published by IEEE. This email provides you with all the information you require to complete your paper and submit it for inclusion in the proceedings. A notification of the presentation format (oral or poster) will be sent by February 20, 2017.

Figura 15 – Mensagem de aceite do artigo na International Joint Conference on Neural Networks (IJCNN 2017)

Figura 16 – Página web da IEEE com os dados do artigo publicado

Multi-View Hard C-Means with Automated Weighting of Views and Variables

Rodrigo C. de Araújo
Centro de Informática
Universidade Federal de Pernambuco
Recife - PE, Brazil
Email: rca7@cin.ufpe.br

Francisco de A.T. de Carvalho
Centro de Informática
Universidade Federal de Pernambuco
Recife - PE, Brazil
Email: fatc@cin.ufpe.br

Yves Lechevallier
INRIA Paris-Rocquencourt
Domaine de Voluceau
Le Chesnay, France
Email: yves1.lechevallier@orange.fr

***Abstract*—Multi-View Clustering models can be viewed as a way to extract information from different data representations to improve the clustering accuracy. In multi-view clustering, some views are irrelevant and among the relevant ones, some may be more or less relevant than others. This is why the most part of existing algorithms assign a weight to each view aiming to compute its relevance in the clustering process. However very few algorithms computes also the relevance weight of variables inside each view aiming to achieve automated feature selection. This paper proposes a muti-view hard c-means clustering algorithm with automated computation of weights for both views and variables in such a way that the relevant views as well as the relevant variables in each view are selected for clustering. Compared to previous similar works, an advantage of the proposed method is that, apart the need to know previously the number of clusters, there are no additional parameters to tune. Experiments with benchmark data sets corroborate the usefulness of the proposed method.**

I. Introduction

Clustering analysis is one of the most important methods that accomplishes unsupervised learning from data which is widely used in areas such as pattern recognition, bioinformatics, data mining, image processing, among others. Clustering aims to provide homogeneous and well separated clusters such that the similarity between the elements within the same group is high and the similarity between elements belonging to different groups is low [1].

As problems involving classification of data appears every day, clustering models emerge taking into account different perspectives to solve these problems. Today is more common to note the existence of separate tables that describe objects from different perspectives. For example, in bio-medicine, different types of information can be obtained from patients, such as magnetic resonance imaging, protein interaction data, blood test results, genetic data, etc. Each of these datasets may be treated as a distinct view that can influence differently the groups of patients. You can also extract different views knowing that information describing objects can be drawn considering different points of view. For example, the extraction of color and texture attributes from a set of images can contribute to classify these images. Knowing this, each perspective can be treated in a more individualized manner during the data collation process, helping to solve search and classification problems.

Currently, there are three approaches to manage multi-view data in the clustering task.

Concatenation: performs a previous treatment in existing views, which may simply be the concatenation of these views, resulting in a single table that will be used to cluster the individuals in a final partition [2], [3].

Distributed methods: they aim to cluster the views independently, using the same or different algorithms. The resulting groups provided by each view are combined to obtain a final partition [4], [5].

Centralized methods: they take into account all views simultaneously to perform the grouping of data. Each view has a relevance weight that will influence the formation of the groups of individuals [6].

In this work we present a centralized method where weights are assigned simultaneously to the views and to the variables of each view during the partitioning process. The proposed method is a multi-view hard c-means clustering algorithm with automated computation of weights for both views and variables in such a way that the relevant views as well as the relevant variables in each view are selected for clustering. A method where views and variables weights are taken into account has been presented by Xu in [7], but the corresponding algorithm depends on two input parameters. However, a procedure for tuning these parameters exists only when the individuals are previously labeled. To the best of our knowledge, the algorithm proposed in this work is the first of its kind that, apart the need to know previously the number of clusters, needs the tuning of no parameters.

The rest of this paper is organized as follows. Next section briefly reviews previous and related work on multi-view clustering. Section III presents the details of the proposed multi-view clustering algorithm. Section IV discusses the performance and usefulness of the proposed algorithm in comparison with three others algorithms: K-Means, Multi-view Kernel K-Means and Weighted Multi-view Clustering with Feature Selection (WMCFS). Finally section V gives the main conclusions and final remarks.

II. Related Work

Research about clustering of multi-view datasets was done extensively in the last years, resulting in the emergence of

978-1-5090-6182-2/17/$31.00 ©2017 IEEE

Figura 17 – Primeira página do artigo publicado na International Joint Conference on Neural Networks (IJCNN 2017)

APÊNDICE B – MÉDIAS COM INTERVALOS DE CONFIANÇA OBTIDOS COM NÍVEL DE SIGNIFICÂNCIA DE 0.05

Tabela 25 – Valores das médias e dos intervalos de confiança de NMI para 100 execuções dos modelos

Dados	**K-Means**	**MVKKM**	**TW-k-Means**	**WMCFS**	**WVVHCM**
Mfeat	0.764±0.006	0.744±0.003	0.582±0.015	0.841±0.006	0.848±0.004
(IC)	(0.758 a 0.770)	(0.741 a 0.747)	(0.567 a 0.597)	(0.835 a 0.847)	(0.844 a 0.852)
AWA	0.193±0.001	0.235±0.001	0.194±0.006	0.186±0.001	0.189±0.001
(IC)	(0.192 a 0.194)	(0.234 a 0.236)	(0.188 a 0.200)	(0.185 a 0.187)	(0.188 a 0.190)
Phoneme	0.641±0.013	0.633±0.011	0.488±0.028	0.736±0.016	0.709±0.017
(IC)	(0.628 a 0.654)	(0.622 a 0.644)	(0.460 a 0.516)	(0.720 a 0.752)	(0.692 a 0.726)
Reuters	0.371±0.005	0.344±0.005	0.367±0.006	0.367±0.005	0.347±0.010
(IC)	(0.366 a 0.376)	(0.339 a 0.349)	(0.361 a 0.373)	(0.362 a 0.372)	(0.337 a 0.357)
Segm	0.620±0.005	0.643±0.002	0.560±0.021	0.615±0.009	0.625±0.007
(IC)	(0.615 a 0.625)	(0.641 a 0.645)	(0.539 a 0.581)	(0.606 a 0.624)	(0.618 a 0.632)
ALOI	0.755±0.004	0.770±0.005	0.764±0.008	0.773±0.004	0.784±0.004
(IC)	(0.751 a 0.759)	(0.765 a 0.775)	(0.756 a 0.772)	(0.769 a 0.777)	(0.780 a 0.788)
Corel 1	0.515±0.002	0.589±0.001	0.565±0.006	0.569±0.002	0.575±0.003
(IC)	(0.513 a 0.517)	(0.588 a 0.590)	(0.559 a 0.571)	(0.567 a 0.571)	(0.572 a 0.578)
Corel 2	0.472±0.005	0.643±0.003	0.557±0.011	0.565±0.008	0.596±0.007
(IC)	(0.467 a 0.477)	(0.640 a 0.646)	(0.546 a 0.568)	(0.557 a 0.573)	(0.589 a 0.603)
Corel 3	0.321±0.007	0.440±0.011	0.491±0.018	0.508±0.009	0.486±0.013
(IC)	(0.314 a 0.328)	(0.429 a 0.451)	(0.473 a 0.509)	(0.499 a 0.517)	(0.473 a 0.499)
Corel 4	0.544±0.011	0.640±0.009	0.496±0.014	0.569±0.011	0.550±0.012
(IC)	(0.533 a 0.555)	(0.631 a 0.649)	(0.482 a 0.510)	(0.558 a 0.580)	(0.538 a 0.562)
Corel 5	0.263±0.003	0.344±0.009	0.225±0.013	0.289±0.005	0.290±0.005
(IC)	(0.260 a 0.266)	(0.335 a 0.353)	(0.212 a 0.238)	(0.284 a 0.294)	(0.285 a 0.295)

Tabela 26 – Valores das médias e dos intervalos de confiança de F-Measure para 100 execuções dos modelos

Dados	**K-Means**	**MVKKM**	**TW-k-Means**	**WMCFS**	**WVVHCM**
Mfeat	0.786±0.010	0.781±0.004	0.582±0.015	0.824±0.010	0.855±0.008
(IC)	(0.776 a 0.796)	(0.777 a 0.785)	(0.567 a 0.597)	(0.814 a 0.834)	(0.847 a 0.863)
AWA	0.117±0.001	0.123±0.000	0.102±0.002	0.112±0.001	0.115±0.001
(IC)	(0.116 a 0.118)	(0.123 a 0.123)	(0.100 a 0.104)	(0.111 a 0.113)	(0.114 a 0.116)
Phoneme	0.709±0.013	0.735±0.008	0.614±0.017	0.784±0.014	0.768±0.013
(IC)	(0.696 a 0.722)	(0.727 a 0.743)	(0.597 a 0.631)	(0.770 a 0.798)	(0.755 a 0.781)
Reuters	0.535±0.007	0.492±0.005	0.535±0.007	0.533±0.007	0.503±0.009
(IC)	(0.528 a 0.542)	(0.487 a 0.497)	(0.528 a 0.542)	(0.526 a 0.540)	(0.494 a 0.512)
Segm	0.643±0.007	0.661±0.003	0.588±0.014	0.625±0.008	0.639±0.007
(IC)	(0.636 a 0.650)	(0.658 a 0.664)	(0.574 a 0.602)	(0.617 a 0.633)	(0.632 a 0.646)
ALOI	0.700±0.007	0.710±0.008	0.720±0.011	0.719±0.008	0.733±0.008
(IC)	(0.693 a 0.707)	(0.702 a 0.718)	(0.709 a 0.731)	(0.711 a 0.727)	(0.725 a 0.741)
Corel 1	0.695±0.003	0.718±0.007	0.735±0.008	0.729±0.008	0.730±0.008
(IC)	(0.692 a 0.698)	(0.711 a 0.725)	(0.727 a 0.743)	(0.721 a 0.737)	(0.722 a 0.738)
Corel 2	0.694±0.007	0.774±0.007	0.690±0.010	0.726±0.009	0.735±0.007
(IC)	(0.687 a 0.701)	(0.767 a 0.781)	(0.680 a 0.700)	(0.717 a 0.735)	(0.728 a 0.742)
Corel 3	0.569±0.007	0.653±0.010	0.685±0.016	0.709±0.012	0.682±0.014
(IC)	(0.562 a 0.576)	(0.643 a 0.663)	(0.669 a 0.701)	(0.697 a 0.721)	(0.668 a 0.696)
Corel 4	0.773±0.014	0.820±0.012	0.714±0.014	0.791±0.012	0.753±0.014
(IC)	(0.759 a 0.787)	(0.808 a 0.832)	(0.700 a 0.728)	(0.779 a 0.803)	(0.739 a 0.767)
Corel 5	0.519±0.004	0.581±0.008	0.471±0.007	0.512±0.006	0.511±0.006
(IC)	(0.515 a 0.523)	(0.573 a 0.589)	(0.464 a 0.478)	(0.506 a 0.518)	(0.505 a 0.517)

Tabela 27 – Valores das médias e dos intervalos de confiança de Adjusted Rand para 100 execuções dos modelos

Dados	**K-Means**	**MVKKM**	**TW-k-Means**	**WMCFS**	**WVVHCM**
Mfeat	0.665±0.011	0.650±0.005	0.427±0.016	0.729±0.014	0.770±0.010
(IC)	(0.654 a 0.676)	(0.645 a 0.655)	(0.411 a 0.443)	(0.715 a 0.743)	(0.760 a 0.780)
AWA	0.021±0.000	0.026±0.000	0.010±0.001	0.020±0.000	0.020±0.000
(IC)	(0.021 a 0.021)	(0.026 a 0.026)	(0.009 a 0.011)	(0.020 a 0.020)	(0.020 a 0.020)
Phoneme	0.523±0.020	0.573±0.011	0.384±0.025	0.640±0.020	0.614±0.020
(IC)	(0.503 a 0.543)	(0.562 a 0.584)	(0.359 a 0.409)	(0.620 a 0.660)	(0.594 a 0.634)
Reuters	0.253±0.006	0.218±0.004	0.254±0.006	0.252±0.006	0.231±0.009
(IC)	(0.247 a 0.259)	(0.214 a 0.222)	(0.248 a 0.260)	(0.246 a 0.258)	(0.222 a 0.240)
Segm	0.478±0.007	0.517±0.004	0.414±0.019	0.456±0.012	0.470±0.010
(IC)	(0.471 a 0.485)	(0.513 a 0.521)	(0.395 a 0.433)	(0.444 a 0.468)	(0.460 a 0.480)
ALOI	0.578±0.009	0.584±0.010	0.587±0.013	0.588±0.010	0.609±0.011
(IC)	(0.569 a 0.587)	(0.574 a 0.594)	(0.574 a 0.600)	(0.578 a 0.598)	(0.598 a 0.620)
Corel 1	0.451±0.001	0.502±0.005	0.492±0.009	0.496±0.006	0.498±0.007
(IC)	(0.450 a 0.452)	(0.497 a 0.507)	(0.483 a 0.501)	(0.490 a 0.502)	(0.491 a 0.505)
Corel 2	0.428±0.007	0.599±0.007	0.471±0.016	0.508±0.012	0.530±0.009
(IC)	(0.421 a 0.435)	(0.592 a 0.606)	(0.455 a 0.487)	(0.496 a 0.520)	(0.521 a 0.539)
Corel 3	0.286±0.007	0.384±0.012	0.446±0.019	0.461±0.013	0.436±0.015
(IC)	(0.279 a 0.293)	(0.372 a 0.396)	(0.427 a 0.465)	(0.448 a 0.474)	(0.421 a 0.451)
Corel 4	0.544±0.017	0.617±0.014	0.442±0.017	0.551±0.015	0.516±0.017
(IC)	(0.527 a 0.561)	(0.603 a 0.631)	(0.425 a 0.459)	(0.536 a 0.566)	(0.499 a 0.533)
Corel 5	0.189±0.003	0.265±0.010	0.139±0.007	0.189±0.005	0.186±0.006
(IC)	(0.186 a 0.192)	(0.255 a 0.275)	(0.132 a 0.146)	(0.184 a 0.194)	(0.180 a 0.192)

Tabela 28 – Valores das médias e dos intervalos de confiança de Purity para 100 execuções dos modelos

Dados	**K-Means**	**MVKKM**	**TW-k-Means**	**WMCFS**	**WVVHCM**
Mfeat	0.771±0.011	0.775±0.005	0.560±0.015	0.813±0.011	0.844±0.009
(IC)	(0.760 a 0.782)	(0.770 a 0.780)	(0.545 a 0.575)	(0.802 a 0.824)	(0.835 a 0.853)
AWA	0.113±0.001	0.121±0.001	0.100±0.003	0.108±0.001	0.111±0.001
(IC)	(0.112 a 0.114)	(0.120 a 0.122)	(0.097 a 0.103)	(0.107 a 0.109)	(0.110 a 0.112)
Phoneme	0.671±0.016	0.717±0.007	0.599±0.019	0.745±0.018	0.736±0.016
(IC)	(0.655 a 0.687)	(0.710 a 0.724)	(0.580 a 0.618)	(0.727 a 0.763)	(0.720 a 0.752)
Reuters	0.532±0.007	0.504±0.004	0.534±0.007	0.530±0.006	0.508±0.009
(IC)	(0.525 a 0.539)	(0.500 a 0.508)	(0.527 a 0.541)	(0.524 a 0.536)	(0.499 a 0.517)
Segm	0.617±0.007	0.671±0.003	0.591±0.016	0.613±0.009	0.621±0.008
(IC)	(0.610 a 0.624)	(0.668 a 0.674)	(0.575 a 0.607)	(0.604 a 0.622)	(0.613 a 0.629)
ALOI	0.674±0.008	0.685±0.008	0.699±0.011	0.696±0.008	0.708±0.009
(IC)	(0.666 a 0.682)	(0.677 a 0.693)	(0.688 a 0.710)	(0.688 a 0.704)	(0.699 a 0.717)
Corel 1	0.691±0.003	0.706±0.008	0.717±0.010	0.720±0.009	0.716±0.009
(IC)	(0.688 a 0.694)	(0.698 a 0.714)	(0.707 a 0.727)	(0.711 a 0.729)	(0.707 a 0.725)
Corel 2	0.686±0.008	0.773±0.008	0.690±0.009	0.724±0.009	0.728±0.007
(IC)	(0.678 a 0.694)	(0.765 a 0.781)	(0.681 a 0.699)	(0.715 a 0.733)	(0.721 a 0.735)
Corel 3	0.568±0.007	0.649±0.010	0.689±0.015	0.708±0.011	0.680±0.014
(IC)	(0.561 a 0.575)	(0.639 a 0.659)	(0.674 a 0.704)	(0.697 a 0.719)	(0.666 a 0.694)
Corel 4	0.777±0.013	0.818±0.013	0.701±0.015	0.787±0.013	0.745±0.015
(IC)	(0.764 a 0.790)	(0.805 a 0.831)	(0.686 a 0.716)	(0.774 a 0.800)	(0.730 a 0.760)
Corel 5	0.500±0.003	0.556±0.007	0.460±0.005	0.493±0.004	0.493±0.005
(IC)	(0.497 a 0.503)	(0.549 a 0.563)	(0.455 a 0.465)	(0.489 a 0.497)	(0.488 a 0.498)