Pós-Graduação em Ciência da Computação

Eduardo Cintra Simões

Algoritmo de Clusterização Difusa Multi-medoides para dados Relacionais com Múltiplas Visões

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

Recife
2019

Eduardo Cintra Simões

Algoritmo de Clusterização Difusa Multi-medoides para dados Relacionais com Múltiplas Visões

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.

Área de Concentração: Inteligência Computacional
Orientador: Francisco de Assis Tenório de Carvalho

Recife
2019

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

S593a Simões, Eduardo Cintra
Algoritmo de clusterização difusa multi-medoides para dados relacionais com múltiplas visões / Eduardo Cintra Simões. – 2019.
96 f.: il., fig., tab.

Orientador: Francisco de Assis Tenório de Carvalho.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2019.
Inclui referências e apêndices.

1. Inteligência computacional. 2. Clusterização difusa. 3. Dados relacionais. I. Carvalho, Francisco de Assis Tenório de (orientador). II. Título.

006.3 CDD (23. ed.) UFPE- MEI 2020-08

Eduardo Cintra Simões

“Algoritmo de Clusterização Difusa Multi-medoides para dados Relacionais com Múltiplas Visões”

Dissertação de Mestrado 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: 17 de dezembro de 2019.

BANCA EXAMINADORA

_____

Prof. Dr. Cleber Zanchettin
Centro de Informática/UFPE

_____

Prof. Dr. Byron Leite Dantas Bezerra
Escola Politécnica de Pernambuco / UPE

_____

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

Dedico este trabalho a minha família e amigos, que me apoiaram durante o caminho.

AGRADECIMENTOS

Agradeço aos meus familiares e amigos por todo o apoio durante esse caminho;

Agradeço ao meu orientador, o Prof. Francisco de Assis Tenorio de Carvalho, não só pelo tempo do Mestrado, mas também pela iniciação científica. Incluindo os ensinamentos da área e sobre pesquisas acadêmicas em geral, além de me apresentar à empresa Callere Document Solutions, onde estou atualmente trabalhando;

Agradeço aos demais professores do Centro de Informática, que me ensinaram os conhecimentos necessários;

Agradeço ao Dr. Everton Barbosa Lacerda pela ajuda para identificar elementos que estavam faltando na versão inicial dessa dissertação;

Agradeço ao Dr. Diogo Philippini Pontual Branco e ao M.e Rodrigo Cavalcanti de Araújo pelas bases *Corel 1*, *Corel 2*, *Reuters RCV1/RCV2 Multilingual* e *ALOI-Cars*, já separadas em matrizes de dissimilaridade e pela ajuda para utilizar o computador de testes do laboratório no Centro de Informática.

"A verdade é sempre encontrada na simplicidade, e não na multiplicidade e confusão das coisas." (NEWTON, apud MANUEL, 1974) [1]

RESUMO

Quando se deseja separar um conjunto de elementos sem um conhecimento prévio das suas classes, é necessário utilizar métodos de classificação não-supervisionada, que separam os elementos pelas suas similaridades, mantendo elementos muito distintos em grupos (*clusters*) diferentes. Contudo, obter um conjunto de características para os elementos pode ser mais custoso do que obter dados relacionais, como a similaridade entre os elementos. Muitas vezes é possível comparar os elementos com diferentes critérios (visões), o que oferece mais informações para distinguir os elementos, permitindo a geração de matrizes de dissimilaridade distintas. Contudo, nem sempre as visões são relevantes para a solução, por causa disso, vários modelos propostos ponderam as diferentes visões, reduzindo a influência daquelas irrelevantes. Em muitas situações, existem elementos com características de várias classes, de forma que classificar um elemento numa única classe não o representa bem. Para melhor representar os elementos nesse tipo de situação, foram desenvolvidos métodos de classificação difusa, que definem graus de pertinência dos elementos para cada classe (em oposição a classificação *crisp*, que mapeia cada elemento para apenas 1 classe). Os métodos de classificação, assim como vários outros métodos de inteligência artificial, precisam ser configurados para obter resultados melhores, utilizando um conjunto de parâmetros para isso. Contudo, quanto mais parâmetros um método possui, mais custoso se torna a busca por bons parâmetros. Dessa forma, reduzir o número de parâmetros sem afetar os resultados é uma excelente melhoria para um método. Esta dissertação tem como objetivo apresentar um modelo de classificação difusa não-supervisionada para dados relacionais com múltiplas visões. Sendo uma variação do modelo FW4M com um parâmetro a menos, mas com resultados similares ou melhores. O método proposto será comparado com o modelo FW4M e outros dois modelos com propósitos similares ($CARD_R$ e MFCMdd-RWG-P) sobre um conjunto de bases de dados, utilizando métricas relevantes para comparar as classificações considerando tanto a partição difusa quanto a *crisp*.

Palavras-chaves: Clusterização Difusa. Multi-Medoides. Dados Relacionais. Multi-Visões. Restrição de Produto.

ABSTRACT

When one wants cluster objects without a previous knowledgement of its classes, it's necessary to utilize non-supervised classification methods, which separate the elements based on their similarities, keeping the too distinct ones in diferent clusters. In many situations, it's too expensive to obtain a set of features for these elements, being cheaper to obtain relational data, like the distance between elements. Many times it's possible to compare elements using different criteria (visions), offering more information to distinguish, allowing the creation of distinct dissimilarity matrices. However, some visions can be irrelevants for the solution, because of that, multiple models were proposed that weight the visions, reducing the influence of the irelevant ones. In many situations, there are elements with caracteristics of multiple classes, in a way that to classify these elements in a single class doesn't represent them very well. To better represent the elements in these situations, fuzzy classifications methods were develop, they define the elements membership degree for each class (in oposition to the crisp classification, that maps them in only one). The classification methods, just like other artificial inteligence methods, need to be tunned to obtein better results, using parameters for that. However, The more parameters a method has, the more expensive it becomes to search for good parameters. That way, to reduce the parameters number without affecting the results is an excellent improvement for a method. This dissertation aims to present a non-supervised fuzzy classification model for relation data with multiple visions. Being a variation of the FW4M model with one parameters less, but with similar or better results. The proposed method will be tested against the F4M model and two other models that have similar proposes ($CARD_R$ and MFCMdd-RWG-P) on a group of datasets, using relevant metrics to compare the classifications considering both the fuzzy and crips partitions.

Keywords: Fuzzy Clustering. Multimedoids. Relational Data. Multi-Views. Product Restriction.

LISTA DE ILUSTRAÇÕES

Figura 1 – Visualização das distâncias mais conhecidas 17
Figura 2 – Visualização de uma busca em grade 53
Figura 3 – Distribuição de t-student 55
Figura 4 – Exemplos das espécies de Íris 63
Figura 5 – Visualização da mfeat-pix da base *Multiple Features* 63
Figura 6 – Exemplos da base Seeds 64
Figura 7 – Exemplos da base Corel 65
Figura 8 – Ângulos utilizados pela base ALOI 67
Figura 9 – Ângulos de iluminação utilizados pela base ALOI 67
Figura 10 – Cores das lâmpadas utilizadas pela base ALOI 67
Figura 11 – Exemplos da base ALOI 68
Figura 12 – Rand Frigui - Resultados Médios 83
Figura 13 – Rand Hullermeier - Resultados Médios 83
Figura 14 – Distância Mínima entre Centroides - Resultados Médios 83
Figura 15 – F-Measure - Resultados Médios 83
Figura 16 – Erro Geral - Resultados Médios 83
Figura 17 – NMI - Resultados Médios 83
Figura 18 – Tempo Médio - Resultados Médios 83
Figura 19 – Rand Frigui - Melhores Resultados 84
Figura 20 – Rand Hullermeier - Melhores Resultados 84
Figura 21 – Distância Mínima entre Centroides - Melhores Resultados 84
Figura 22 – F-Measure - Melhores Resultados 84
Figura 23 – Erro Geral - Melhores Resultados 84
Figura 24 – NMI - Melhores Resultados 84
Figura 25 – Tempo Médio - Melhores Resultados 84
Figura 26 – Impacto do número de *clusters* para classes com múltiplos agrupamentos 86
Figura 27 – Visualização de uma notação Big-O 96

LISTA DE ALGORITMOS

1 FANNY - Algoritmo 22
2 NERF - Algoritmo 25
3 FCMdd - Algoritmo 27
4 FMMdd - Algoritmo 29
5 CARD_R - Algoritmo 32
6 CARD_F - Algoritmo 35
7 Etapa de representação do MFCMdd 37
8 MFCMdd - Algoritmo 38
9 FW4M - Algoritmo 41
10 Estrutura dos algoritmos da área 42

11 MFMMdd - Alocação dos Elementos 45
12 MFMMdd - Representação dos Grupos 46
13 MFMMdd - Ponderação das Visões 46
14 MFMMdd - Algoritmo 48
15 MFMMdd - Alocação dos Elementos (Otimizado) 49
16 MFMMdd - Representação dos Grupos (Otimizado) 50
17 MFMMdd - Ponderação das Visões (Otimizado) 51

LISTA DE TABELAS

Tabela 1 – Variações do modelo MFCMdd 36
Tabela 2 – Atualização da Ponderação das Visões 37
Tabela 3 – Comparação dos modelos da literatura 43
Tabela 4 – Parâmetros Selecionados 54
Tabela 5 – coeficiente do teste bicaudal de Nemenyi 56
Tabela 6 – Bases de dados 61
Tabela 7 – Visões da base *Multiple Features* 63
Tabela 8 – Visões da base Corel 66
Tabela 9 – Visões da base ALOI 68
Tabela 10 – Parâmetros Selecionados 69
Tabela 11 – Rand Frigui - Resultado Médio 70
Tabela 12 – Rand Hullermeier - Resultado Médio 71
Tabela 13 – Distância Mínima entre Centroides - Resultado Médio 72
Tabela 14 – F-measure - Resultado Médio 73
Tabela 15 – Erro Geral - Resultado Médio 74
Tabela 16 – Informação Mútua Normalizada - Resultado Médio 75
Tabela 17 – Tempo Médio (em segundos) 76
Tabela 18 – Rand Frigui - Melhor Resultado 77
Tabela 19 – Rand Hullermeier - Melhor Resultado 77
Tabela 20 – Distância Mínima entre Centroides - Melhor Resultado 78
Tabela 21 – F-measure - Melhor Resultado 78
Tabela 22 – Erro Geral - Melhor Resultado 79
Tabela 23 – Informação Mútua Normalizada - Melhor Resultado 79
Tabela 24 – Tempo de execução (em segundos) - Melhor Resultado 80
Tabela 25 – *Ranks* médios - Resultados Médios 80
Tabela 26 – *Ranks* médios - Melhores Resultados 81
Tabela 27 – Teste de Friedman - Resultados Médios 81
Tabela 28 – Teste de Friedman - Melhores Resultados 81

LISTA DE SÍMBOLOS

$\in$	Pertence
$\leq$	Menor ou igual a
$>$	Maior que
$\sum$	Somatório
$\prod$	Produtório
$\forall$	Para todo
N	Número de elementos
K	Número de grupos
P	Número de visões
m	Parâmetro que controla a difusão das pertinências
n	Parâmetro que controla a suavidade das representações dos grupos
s	Parâmetro que controla a suavidade das ponderações das visões
q	Parâmetro que define o número de elementos na representação de um *cluster*
J	Energia do sistema (Entropia, Função-Objetivo)
D	Matriz de Dissimilaridade de um problema *single-view*
$D(i,j)$	Dissimilaridade entre o i-ésimo e j-ésimo elementos
D_p	p-ésima Matriz de Dissimilaridade de um problema *multi-view*
$D_p(i,j)$	Dissimilaridade entre o i-ésimo e j-ésimo elementos na p-ésima visão
u_{ic}	Pertinência do i-ésimo elemento no c-ésimo grupo
v_{cj}	Peso do j-ésimo elemento no c-ésimo grupo
δ_p	Peso da p-ésima visão
δ_{cp}	Peso da p-ésima visão para o c-ésimo *cluster* (ponderação local)

SUMÁRIO

1 INTRODUÇÃO 16
1.1 OBJETIVO DO TRABALHO 18
1.2 ESTRUTURA DA DISSERTAÇÃO 19

2 REVISÃO DA LITERATURA 20
2.1 MÉTODO DE MULTIPLICADORES DE LAGRANGE 20
2.2 MODELOS *SINGLE-VIEW* 20
2.2.1 *Fuzzy Analysis* (FANNY) 21
2.2.2 *Non-Euclidean Relational Fuzzy* (NERF) 23
2.2.3 *Fuzzy C-Medoids* (FCMdd) 25
2.2.4 *Fuzzy Relational Clustering with Weighted Medoids* (FMMdd) . . . 27
2.3 MODELOS *MULTI-VIEW* 30
2.3.1 *Clustering and Aggregating Relational Data* (CARD) 30
2.3.1.1 CARD para o modelo RFCM (CARD_R) 30
2.3.1.2 CARD para o modelo FANNY (CARD_F) 33
2.3.2 *Partitioning Fuzzy K-Medoids Clustering Algorithm Based on Multiple Dissimilarity Matrices* (MFCMdd) 36
2.3.3 *Fuzzy Clustering Algorithm Based on Weighted Multi-Medoids and Multi-Matrices* (FW4M) 39
2.4 RESUMO DOS MODELOS VISTOS 42

3 MODELO PROPOSTO 44
3.1 INICIALIZAÇÃO 44
3.2 OBTENÇÃO DAS EQUAÇÕES DA OTIMIZAÇÃO 45
3.3 COMPUTAÇÃO DAS PERTINÊNCIAS 45
3.4 COMPUTAÇÃO DA REPRESENTAÇÃO DOS MEDOIDES 45
3.5 COMPUTAÇÃO DA PONDERAÇÃO DAS VISÕES 46
3.6 ALGORITMO PROPOSTO 48
3.7 COMPLEXIDADE E OTIMIZAÇÃO 48
3.7.1 Complexidade da Função Objetivo 48
3.7.2 Complexidade da Alocação 49
3.7.3 Complexidade da Representação 50
3.7.4 Complexidade da Ponderação 51
3.7.5 Complexidade da Inicialização 52
3.7.6 Complexidade do Algoritmo 52

4 METODOLOGIA . . . 53
4.1 BUSCA DOS PARÂMETROS . . . 53
4.2 COMPARAÇÃO DOS MODELOS . . . 54
4.3 MÉTRICAS . . . 56
4.3.1 Partição *Crisp* . . . 56
4.3.1.1 Erro Geral . . . 57
4.3.1.2 F-Measure . . . 57
4.3.1.3 Informação Mútua Normalizada (NMI) . . . 58
4.3.1.4 Índice Rand . . . 58
4.3.2 Partição Difusa . . . 59
4.3.2.1 Rand Frigui . . . 59
4.3.2.2 Rand Hullermeier . . . 59
4.3.2.3 Distância Mínima entre Centroides . . . 60
4.3.2.4 Coeficiente de Partição Modificado . . . 60
4.4 BASES DE DADOS . . . 60
4.4.1 Glass Identification . . . 62
4.4.2 Image Segmentation . . . 62
4.4.3 Iris . . . 62
4.4.4 Multiple Features . . . 63
4.4.5 Phoneme . . . 64
4.4.6 Seeds . . . 64
4.4.7 Wine . . . 64
4.4.8 Corel Images . . . 65
4.4.9 Reuters RCV1/RCV2 Multilingual . . . 66
4.4.10 Amsterdam Library of Object Images (ALOI) . . . 66

5 AVALIAÇÃO EXPERIMENTAL . . . 69
5.1 MÉDIA DOS RESULTADOS . . . 70
5.1.1 Métricas da Partição Difusa . . . 70
5.1.2 Métricas da Partição *Crisp* . . . 73
5.1.3 Tempo de Execução . . . 75
5.2 MELHORES RESULTADOS . . . 76
5.2.1 Métricas da Partição Difusa . . . 77
5.2.2 Métricas da Partição *Crisp* . . . 78
5.2.3 Tempo de Execução . . . 80
5.3 *RANKS* MÉDIOS . . . 80
5.4 TESTE DE FRIEDMAN . . . 81
5.5 PÓS-TESTE DE NEMENYI . . . 82
5.5.1 Resultados Médios . . . 83
5.5.2 Melhores Resultados . . . 84

6 CONCLUSÕES E TRABALHOS FUTUROS 85
6.1 RESUMO DOS RESULTADOS 85
6.2 TRABALHOS FUTUROS 86

REFERÊNCIAS 87

APÊNDICE A – SOLUCIONANDO O SISTEMA DE LAGRANGE 90

APÊNDICE B – NOTAÇÃO BIG-O 96

1 INTRODUÇÃO

É de conhecimento geral que base de dados é um conjunto organizado de informações sobre elementos de um determinado domínio. Ela pode possuir descritores desses elementos ou comparações entre os mesmos. As bases de dados fazem parte do nosso dia a dia, mesmo que não diretamente (Agendas telefônicas, Todo cidadão é registrado numa base de dados do governo, dados pessoais armazenados em mídias sociais como Facebook e Twitter, dados utilizados para abrir uma conta de e-mail ou uma conta corrente num banco, entre outras situações).

Os elementos de uma base podem ser descritos pelas suas características ou pelas distâncias (ou similaridades) em relação aos demais elementos. Podendo ter essas descrições representando visões [1] diferentes do problema. Caso a base utilize apenas uma visão, ela é considerada *single-view* (visão única), já se ela utilizar 2 ou mais visões, a base é *multi-view* (múltiplas visões), o que permite uma ponderação das diferentes visões (uma granularidade entre não ter ponderação e ter uma ponderação de cada característica dos elementos).

Caso os elementos sejam descritos na base pelas suas características, elas podem possuir valores numéricos ou categóricos. Nesse tipo de descrição, a diferença entre dois elementos pode ser calculada de diversas formas, sendo as mais conhecidas a distância euclideana, a distância de Manhattan (ou *city-block*) e a distância de Chebyshev (Equação 1.1, Equação 1.2 e Equação 1.3, respectivamente).

$$D(i,j) = \sqrt{\sum_{p=1}^{P} d_p(i,j)^2} \tag{1.1}$$

$$D(i,j) = \sum_{p=1}^{P} |d_p(i,j)| \tag{1.2}$$

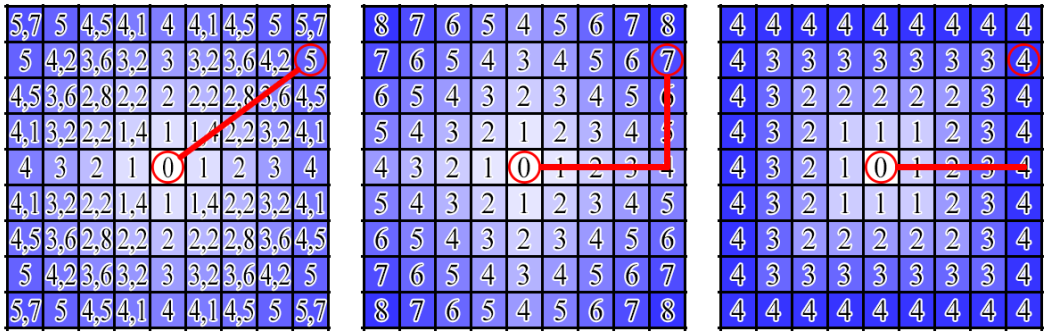
$$D(i,j) = \max_{1 \leq p \leq P} d_p(i,j) \tag{1.3}$$

Sendo:

$$d_p(i,j) = \begin{cases} x_{ip} - x_{jp} & \text{, para atributos numéricos} \\ 0 & \text{, para atributos da mesma categoria} \\ 1 & \text{, para atributos de categorias diferentes} \end{cases} \tag{1.4}$$

[1] uma visão representa uma forma de observar um problema para extrair um conjunto de características (por exemplo: examinar as dimensões dos elementos para extrair altura, largura e comprimento), servindo como um agrupamento de características relacionadas.

Figura 1 – Visualização das distâncias mais conhecidas



a) Distância Euclidiana b) Distância Manhattan c) Distância Chebyshev

Fonte: O Autor (2019)

Caso os elementos sejam descritos pelas suas distâncias ou similaridades para os demais elementos (normalmente organizados numa ou mais matrizes NxN), os dados são considerados relacionais e suas matrizes podem ser referidas como visões. Caso exista apenas uma visão, a base é considerada *single-view* (visão única), caso existam 2 ou mais visões, a base é *multi-view* (múltiplas visões), o que permite uma ponderação das diferentes visões.

Entre as utilizações possíveis para bases de dados, estão a validação dos dados (como num sistema de senhas), comparação de elementos (como para escolher qual vídeo será sugerido para o usuário assistir), obter novas informações (como calcular quanto de imposto um cidadão precisa pagar) e classificação (como decidir se um banco deve oferecer um empréstimo).

Classificação é uma subárea da inteligência artificial responsável pela separação de elementos em grupos, podendo ser utilizada numa etapa intermediária da solução de outras áreas (e.g.: classificar um documento para decidir quais as informações que podem ser extraídas) ou sendo a solução do problema em si (e.g.: decidir se um cidadão pode participar de algum programa de assistência social). A classificação pode ser feita encontrando representantes para as classes, ou criando uma hierarquia de divisões do grupo.

Um método de classificação pode ser supervisionado, ou seja, ele seria treinado em conjunto de elementos cujas classes são conhecidas, servindo de exemplo para o modelo a ser treinado aprender os padrões de cada classe, sendo usados quando se deseja construir um classificador para separar os elementos em classes já conhecidas (por exemplo: um algoritmo para definir o tipo de um documento escaneado). Esse tipo de método é o mais comum, contudo, existem situações onde obter um conjunto já classificado, e com tamanho suficiente para o treinamento, é muito custoso. Para esses casos é necessária a utilização de métodos que não utilizam as classes no treinamento: os métodos não supervisionados.

Os métodos não-supervisionados (também conhecidos como métodos de clusterização) separam os elementos pelo grau de similaridade entre os mesmos, tendo como objetivo colocar elementos muito distintos em grupos diferentes, e elementos semelhantes no mesmo grupo. Como não utilizam as classes no treinamento, é necessária alguma medida que

caracterize a qualidade dessa separação. Para tal, os métodos de clusterização costumam minimizar uma função objetivo que soma, de forma ponderada, as distâncias dos elementos dentro das classes (exemplos dessas funções objetivo podem ser encontrados no Capítulo 2).

Os primeiros métodos de classificação que surgiram colocavam cada elemento em apenas uma classe (o que é conhecido como método *crisp*), a ideia seria que cada elemento pertenceria a apenas uma classe. Contudo, existem elementos próximos das fronteiras das classes, com características presentes de ambas as classes. Por causa disso, foram desenvolvidos métodos de classificação difusa, métodos esses que atribuem para os N elementos um grau de pertinência u_{ic} para cada uma das K classes (sendo i e c os índices do elemento e da classe, respectivamente), o que permite uma representação mais precisa desses elementos das fronteiras. As pertinências de um elemento não podem ser negativas e devem respeitar as restrições dada na Equação 1.5 e Equação 1.6.

$$u_{ic} \geq 0, \forall i = 1, 2, ..., N, c = 1, 2, ..., K \tag{1.5}$$

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{1.6}$$

Várias soluções já foram propostas para o agrupamento de forma difusa de elementos descritos por matrizes de dissimilaridade. Algumas delas são os modelos *single-view* NERF [2], FANNY [3], FCMdd [4] e FMMdd [5] e os modelos *multi-view* CARD_R e CARD_F [6], MFCMdd [7] e FW4M [8]. Mais detalhes sobre esses modelos serão dados no Capítulo 2.

Todas as soluções são sensíveis aos seus parâmetros e, portanto, para se obter os melhores resultados, é necessário buscar os parâmetros ideais. Dessa forma, é necessário utilizar métodos de busca no conjunto de valores possíveis dos parâmetros para tentar obter bons parâmetros e, consequentemente, bons resultados. Contudo, o custo dessa busca aumenta de forma exponencial ao número de parâmetros a serem ajustados, ou seja, um aumento consideravelmente o tempo utilizado apenas para configurar o modelo. Portanto, é interessante a redução do número dos parâmetros, desde que os resultados não sejam muito prejudicados por essa redução.

1.1 OBJETIVO DO TRABALHO

Essa dissertação tem como objetivo apresentar um novo método de clusterização difusa com múltiplas visões: o *"Multi-view Relational Fuzzy Clustering Algorithm with Multi-medoids"* (MFMMdd). O modelo proposto foi baseado no modelo FW4M, mas com uma restrição por produto para ponderação das visões, com o intuito de reduzir o número de parâmetros, reduzindo o tempo de busca dos parâmetros, além de melhorar os resultados.

Os objetivos específicos desse trabalho são comparar o modelo proposto, o de referência FW4M, o MFCMdd-RWG-P, que inspirou a utilização da restrição por produto,

e o CARD_R, um modelo *multi-view* relevante da área que também possui 2 parâmetros, utilizando 11 bases de dados de classificação; comparar os resultados quantitativos dos modelos em 100 execuções para um conjunto de métricas [2] e validar a relevância do modelo proposto.

1.2 ESTRUTURA DA DISSERTAÇÃO

Essa dissertação está dividida em 6 capítulos: No capítulo atual, foi apresentada uma introdução ao problema e o objetivo deste trabalho. No Capítulo 2, foi apresentada uma revisão da literatura, que inclui conceitos utilizados e modelos de clusterização relacionados (tanto *single-view*, quanto *multi-view*) dando uma melhor descrição ao modelo FW4M. No Capítulo 3, foi explicado o modelo proposto, suas diferenças ao modelo de referência, a complexidade do algoritmo e otimizações das equações para reduzir essa complexidade. No Capítulo 4, foram apresentas as métricas e bases utilizadas nas comparações entre o modelo de referência, o modelo proposto e dois outros modelos da literatura (CARD_R e MFCMdd-RWG-P). No Capítulo 5, foram apresentados os resultados dos experimentos descritos no Capítulo 4, comparando o modelo proposto com os modelos relacionados (FW4M, CARD_R e MFCMdd-RWG-P). No Capítulo 6, foram apresentadas as conclusões da comparação entre o modelo proposto com os demais, além de possíveis trabalhos futuros baseados neste trabalho.

[2] As métricas utilizadas foram: Rand Frigui, Rand Hullermeier, Distância mínima entre centroides, Erro geral, F-Measure, Informação mútua normalizada e o tempo de execução

2 REVISÃO DA LITERATURA

A seguir serão descritos alguns dos modelos existentes na literatura para o agrupamento não-supervisionado para dados relacionais descritos por matrizes de dissimilaridade, incluindo a função objetivo e suas restrições, o algoritmo com as equações utilizadas no passo iterativo para ajustar a solução, os parâmetros e a complexidade dos modelos (representada pela notação *Big-O*, descrita no Apêndice B). Além de alguns conceitos utilizados nessa descrição.

Para facilitar a compreensão do leitor, as siglas foram padronizadas, ao invés de continuar com as utilizadas pelos autores originais (Ver a lista de símbolos).

N e P são inerentes do problema (respectivamente: a dimensão de uma matriz de dissimilaridade e o número de matrizes na entrada) e, portanto, não serão considerados como parâmetros, mas sim como a entrada do problema. Já o parâmetro K (o número de *clusters* nos quais que se deseja dividir os elementos) foi considerado, nesse trabalho, igual ao número de classes do problema.

2.1 MÉTODO DE MULTIPLICADORES DE LAGRANGE

O Método de Multiplicadores de Lagrange [9] é uma técnica para encontrar os passos de otimização de uma função J, respeitando as suas restrições.

Primeiro, é preciso montar a função de Lagrange:

$$L_J = J - \sum_{r \in Restrições} lp_r \tag{2.1}$$

Na qual lp_r é a parte da função de Lagrange que representa a r-ésima restrição de J. Considerando a restrição $f_r(X) = C_r$ onde $f_r(X)$ é uma fórmula sobre as variáveis da função J e C_r uma constante, temos que:

$$lp_r = \alpha_r \left(f_r(X) - C_r \right) \tag{2.2}$$

Sendo α_r o multiplicador de Lagrange da r-ésima restrição.

A partir da função de Lagrange L_{func}, iguala-se a zero as derivadas das variáveis da função J e dos multiplicadores de Lagrange. Desse conjunto de equações, pode-se isolar as variáveis de J para se obter as funções de atualização das mesmas.

2.2 MODELOS *SINGLE-VIEW*

Os primeiros modelos de clusterização com dados relacionais utilizavam uma única visão. Naturalmente, esses modelos não possuem o passo de ponderação das visões. Mesmo assim, é possível utilizar esses modelos para problemas com múltiplas visões, só precisa

juntar as várias matrizes em uma única, utilizando algum cálculo de distância (e.g.: a distância demonstrada pela Equação 2.3).

$$D(i,j) = \sqrt{\sum_{p=1}^{P} D_p(i,j)^2},\ para\ 1 \leq i,j \leq N \tag{2.3}$$

A seguir serão apresentados os modelos *single-view Fuzzy Analysis* (FANNY), *Non-Euclidean Relational Fuzzy* (NERF), *Fuzzy C-Medoids* (FCMdd) e *Fuzzy Relational Clustering with Weighted Medoids* (FMMdd), que foram utilizados como inspiração pelos modelos *multi-view* utilizados nesse estudo.

2.2.1 *Fuzzy Analysis* (FANNY)

O algoritmo *"Fuzzy Analysis"* (FANNY) [3] é um dos primeiros modelos de clusterização difusa propostos e foi apresentado pelos autores do livro como um programa. Ele minimiza a função objetivo descrita pela Equação 2.4.

$$J = \sum_{c=1}^{K} \frac{\sum_{i=1}^{N} u_{ic}^2 \sum_{j=1}^{N} u_{jc}^2 D(i,j)}{2\sum_{j=1}^{N} u_{jc}^2} \tag{2.4}$$

Tendo uma inicialização aleatória das pertinências, respeitando a restrição de soma para as pertinências:

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.5}$$

com os passos de otimização obtidos com o método de multiplicadores de Lagrange [9], tendo a equação lagrangiana definida pela Equação 2.6, sendo λ_i os multiplicadores de Lagrange.

$$L_{Fanny} = \sum_{c=1}^{K} \frac{\sum_{i=1}^{N} u_{ic}^2 \sum_{j=1}^{N} u_{jc}^2 D(i,j)}{2\sum_{j=1}^{N} u_{jc}^2} - \sum_{i=1}^{N} \lambda_i \left(\sum_{c=1}^{K} u_{ic} - 1 \right) \tag{2.6}$$

Por causa da simplicidade do modelo, as únicas entradas necessárias são a matriz de dissimilaridades e o número de *clusters*, não possuindo nenhum parâmetro adicional.

Como pode-se perceber, o algoritmo FANNY não possuir um vetor específico para as ponderação das representações dos *clusters* (v_{cj}), dessa forma, foi utilizando o próprio vetor de pertinências como a ponderação das representações.

A pertinência dos elementos é dada pela Equação 2.7, zerando os resultados negativos, já que as pertinências não podem ser negativas.

$$u_{ic} = \begin{cases} \frac{1/a_{ic}}{\sum_{k \in V_i+} (1/a_{ik})} & \text{, caso } c \in V_i+ \\ 0 & \text{, caso contrário} \end{cases} \tag{2.7}$$

Sendo V_i+ o conjunto com todos os *clusters* no qual o elemento teria uma pertinência positiva ao utilizar a equação das pertinências (como demonstrado pela Equação 2.8).

$$V_i+ = \left\{ c : \frac{1/a_{ic}}{\sum_{k=1}^{K}(1/a_{ik})} > 0, \ \textit{sendo } c = 1, 2, ..., K \right\} \tag{2.8}$$

Já o termo a_{ic} presente em ambas as equações é definido pela Equação 2.9, e calculado previamente para evitar repetir esses cálculos custosos, reduzindo o custo computacional. Também é possível separar a computação de a_{ic} entre a parte que depende do índice i e a parte que não depende, permitindo calcular a parte independente anteriormente, sendo $\mathrm{O}(N^2K)$ o custo de calcular todos os valores de a_{ic}.

$$a_{ic} = \frac{2\sum_{j=1}^{N} u_{jc}^2 D(i,j)}{\sum_{j=1}^{N} u_{jc}^2} - \frac{\sum_{h=1}^{N} u_{hc}^2 \sum_{j=1}^{N} u_{jc}^2 D(h,j)}{\left(\sum_{j=1}^{N} u_{jc}^2\right)^2} \tag{2.9}$$

Com essas equações definidas, temos o algoritmo do modelo FANNY como descrito a seguir:

Algoritmo 1: FANNY - Algoritmo

ENTRADA

D: Matriz de Dissimilaridade NxN

K: Número de *Clusters*

T_{max}: Número máximo de iterações

ϵ: Variação Máxima da Função Objetivo para considerar a solução estabilizada

SAÍDA

A Matriz $U = [u_{ic}]$ de pertinências dos N elementos

INICIALIZAÇÃO

Define $t = 0$

Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1 (1 \leq i \leq N)$

J_0 = Função Objetivo (Equação 2.4)

ITERAÇÃO

Repetir

Define $t = t + 1$

Calcula os valores auxiliares a_{ic} com a Equação 2.9

Encontra os conjuntos V_i+ com a Equação 2.8

Atualiza as pertinências u_{ic} com a Equação 2.7

J_t = Função Objetivo (Equação 2.63)

Até $t \geq T_{MAX} \vee |J_t - J_{t-1}| < \epsilon$

Com o algoritmo em mãos, podemos calcular o custo computacional do modelo FANNY,

que é:

$$\begin{aligned} O_{FANNY} =& O_{inic} + T_{MAX} \times (O_{Iter}) \\ =& (O_{aleat} + O_J) + T_{MAX} \times (O_{a_{ic}} + O_{V_i+} + O_{u_{ic}} + O_J) \\ =& O(NK + KN^2) + T_{MAX} \times O(N^2K + NK + NK + KN^2) \\ =& O(KN^2) + T_{MAX} \times O(N^2K) \\ =& O(T_{MAX} \cdot N^2K) \end{aligned} \tag{2.10}$$

Os autores do livro também utilizaram o Coeficiente de Partição (e a versão modificada) e o método de conversão para a classificação crisp que será utilizada nesse trabalho (ambos serão explicados no Capítulo 4).

2.2.2 *Non-Euclidean Relational Fuzzy* (NERF)

O algoritmo *"Non-Euclidean Relational Fuzzy"* (NERF) [2] foi proposto como uma melhoria do algoritmo *"Relational fuzzy c-means"* (RFCM), contudo, só é garantido que o modelo RFCM irá convergir para os casos onde a matriz de dissimilaridade pode ser obtida com a distância euclidiana ao quadrado entre N pontos do $\Re^{N-1}$, o que se torna uma grande limitação do modelo. A melhoria do modelo NERF está relacionado com esse problema, tendo o objetivo de remover essa limitação.

O modelo NERF minimiza a função objetivo 2.11, tendo m como o único parâmetro.

$$J = \sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m ||x_i - g_c||^2 \tag{2.11}$$

Sendo g_c o centro do c-ésimo *cluster*. Contudo, como a entrada é uma matriz de dissimilaridade, a distância para o centro do *cluster* não pode ser calculada diretamente. Portanto, o algoritmo foi criado com isso em mente, podendo ter a sua função objetivo ajustada para a equação:

$$J = \sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m \frac{\sum_{j=1}^{N} u_{jc}^m D(i,j)}{2 \sum_{j=1}^{N} u_{jc}^m} \tag{2.12}$$

Além disso, como a matriz de dissimilaridade não é necessariamente euclidiana, uma matriz euclidiana auxiliar (D_β) foi utilizada nos cálculos, sendo ajustada em tempo de execução para suprir a deficiência do modelo RFCM. Na fase iterativa, as representações dos *clusters* são calculadas primeiro, com a Equação 2.13, possuindo uma restrição de soma (Equação 2.14).

$$v_{cj} = \frac{u_{jc}}{\sum_{i=1}^{N} u_{ic}} \tag{2.13}$$

$$\sum_{j=1}^{N} v_{cj} = 1, \forall c = 1, 2, ..., K \tag{2.14}$$

após isso, são calculados valores auxiliares d_{ic} (Equação 2.15), que serão utilizados no cálculo das pertinências.

$$d_{ic} = (D_\beta v_c)_i - (v_c^T D_\beta v_c)/2 \tag{2.15}$$

A matriz euclidiana D_β é obtida com a Equação 2.16.

$$D_\beta(i,j) = \begin{cases} D_p(i,j) & \text{, se i = j} \\ D_p(i,j) + \beta & \text{, caso contrário} \end{cases} \tag{2.16}$$

Sendo β o fator de espalhamento adicionado ao NERF para superar as limitações do modelo RFCM. Esse fator é atualizado caso exista um $d_{ic} < 0$ para qualquer um dos valores da iteração, sendo incrementado pelo fator $\Delta\beta$ (Equação 2.17). Os valores d_{ic} são então incrementados por Δd_{ic}, definido pela Equação 2.18.

$$\Delta\beta = max_{1\leq i\leq N, 1\leq c\leq K}\left[\frac{-2d_{ic}}{||v_i - e_c||^2}\right] \tag{2.17}$$

$$\Delta d_{ic} = \frac{\Delta\beta}{2}||v_i - e_c||^2 \tag{2.18}$$

Sendo e_c a c-ésima coluna da matriz de identidade com dimensão KxK e v_i as representações referentes ao i-ésimo elemento, de forma que:

$$||v_i - e_c||^2 = \sum_{k=1}^{K}(v_{ki} - I_{kc})^2 \tag{2.19}$$

Lembrando que a matriz de identidade I tem os valores definidos por:

$$I_{kc} = \begin{cases} 1 & \text{, se k = c} \\ 0 & \text{, caso contrário} \end{cases} \tag{2.20}$$

Caso $d_{ik} > 0, \forall k = 1, 2, ..., K$, então as pertinências são calculadas com a Equação 2.21. Caso contrário, as pertinências nos *clusters* referentes aos valores $d_{ik} < 0$ são zeradas, e as demais ajustadas respeitando a restrição de soma das pertinências (Equação 2.22)

$$u_{ic} = \left[d_{ic}\sum_{k=1}^{K}\left(\frac{1}{d_{ik}}\right)\right]^{\frac{-1}{m-1}} \tag{2.21}$$

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.22}$$

O modelo então se organiza no seguinte algoritmo:

Algoritmo 2: NERF - Algoritmo

ENTRADA

D: Matriz de Dissimilaridade NxN

m: Parâmetro de Difusão das Pertinências

K: Número de *Clusters*

T_{max}: Número máximo de iterações

ϵ: Variação Máxima da Função Objetivo para considerar a solução estabilizada

SAÍDA

A Matriz $U = [u_{ic}]$ de pertinências dos N elementos

INICIALIZAÇÃO

Define $t = 0$

Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1 (1 \leq i \leq N)$

J_0 = Função Objetivo (Equação 2.11)

ITERAÇÃO

Repetir

Define $t = t + 1$

Calcula as representações v_{cj} com a Equação 2.13

Calcula os valores auxiliares d_{ic} com a Equação 2.15

Caso necessário, atualiza o fator β e os valores d_{ic} com as Equação 2.17 e Equação 2.18

Atualiza as pertinências u_{ic} com a Equação 2.21

J_t = Função Objetivo (Equação 2.11)

Até $t \geq T_{MAX} \vee |J_t - J_{t-1}| < \epsilon$

O custo computacional do modelo NERF, considerando o algoritmo anterior, fica:

$$\begin{aligned} O_{NERF} =& O_{inic} + T_{MAX} \times O_{iter} \\ =& (O_{aleat} + O_J) + T_{MAX} \times (O_{Repr} + O_{d_{ic}} + O_{\Delta\beta} + O_{\Delta d_{ic}} + O_{Pert} + O_J) \\ =& O(NK + KN^2) \\ & + T_{MAX} \times O(NK + N^2K + NK + NK + NK + KN^2) \\ =& O(KN^2) + T_{MAX} \times O(N^2K) \\ =& O(T_{MAX} \cdot N^2K) \end{aligned} \tag{2.23}$$

2.2.3 *Fuzzy C-Medoids* (FCMdd)

O algoritmo *"Fuzzy C-Medoids"* (FCMdd) [4] foi proposto para mineração da Web com baixa complexidade, precisando ser robusto a *outliers*. Para reduzir a complexidade do cálculo das pertinências, o modelo FCMdd representa cada um dos *clusters* com um dos elementos como centroide (g_c). O que simplifica o cálculo das distâncias para os

clusters, principalmente para dados relacionais (em relação ao caso de utilizar os centros dos *clusters*), já que dessa forma, a distância de um elemento para um *cluster* é o valor que está na matriz de dissimilaridade, sendo obtido com um acesso direto (tempo constante). O modelo tem como objetivo minimiza a função objetivo descrita na Equação 2.24, tendo m como único parâmetro.

$$J = \sum_{i=1}^{N} \sum_{c=1}^{K} u_{ic}^{m} D(i, g_c) \tag{2.24}$$

Para tal, o modelo atualiza as pertinências com a Equação 2.25, tendo uma restrição de soma (Equação 2.26).

$$u_{ic} = \frac{[D(i, g_c)]^{\frac{-1}{m-1}}}{\sum_{k=1}^{K} [D(i, g_k)]^{\frac{-1}{m-1}}} \tag{2.25}$$

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.26}$$

E atualiza as representações dos *clusters* trocando, se necessário, o representante dos mesmos (utilizando a Equação 2.27).

$$g_c = argmin_{1 \leq j \leq N} \sum_{i=1}^{N} u_{ic}^{m} D(i, j) \tag{2.27}$$

O modelo primeiro seleciona representantes aleatórios e distintos para as classes. Então, os passos de atualização são repetidos, alternadamente, até que os representantes dos *clusters* não sejam mais modificados, ou até um número de repetições pré-definido (T_{MAX}) ser atingido. A condição de parada é essa pois as pertinências não mudam caso os representantes também não mudem (situação de convergência). Já o número limite de repetições serve para evitar casos onde os representantes possam ficar alternando indefinidamente (evitando um laço infinito).

Com essas informações, temos que o algoritmo fica:

Algoritmo 3: FCMdd - Algoritmo

ENTRADA

D: Matriz de Dissimilaridade NxN

m: Parâmetro de Difusão das Pertinências

K: Número de *Clusters*

T_{max}: Número máximo de iterações

SAÍDA

A Matriz $U = [u_{ic}]$ de pertinências dos N elementos

O Vetor $G = [g_c]$ dos representantes dos K *clusters*

INICIALIZAÇÃO

Define $t = 0$

Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1 (1 \leq i \leq N)$

Seleciona os representantes g_c aleatoriamente, evitando repetição

J_0 = Função Objetivo (Equação 2.24)

ITERAÇÃO

Repetir

Define $t = t + 1$

Atualiza as pertinências u_{ic} com a Equação 2.25

Atualiza os representantes g_c com a Equação 2.27

J_t = Função Objetivo (Equação 2.24)

Até $t \geq T_{MAX} \vee$ G não for alterado

Considerando essas etapas, podemos calcular a complexidade do algoritmo como:

$$\begin{aligned} O_{FCMdd} =& O_{inic} + T_{MAX} \times O_{Iter} \\ =& (O_{aleat} + O_J) + T_{MAX} \times (O_{Pert} + O_{Repr} + O_J) \\ =& O((K + NK) + NK) + T_{MAX} \times O(NK + KN^2 + NK) \\ =& O(NK) + T_{MAX} \times O(KN^2) \\ =& O(T_{MAX} \cdot KN^2) \end{aligned} \tag{2.28}$$

A complexidade acabou sendo a mesma do modelo FANNY, o que não parece ser interessante considerando que o objetivo do trabalho era reduzir a complexidade. Contudo, se considerarmos o uso do modelo para classificar novos elementos sem retreinar o modelo, a complexidade para classificar um elemento é O(K) (contra O(NK) do modelo FANNY).

2.2.4 *Fuzzy Relational Clustering with Weighted Medoids* (FMMdd)

O algoritmo *"Fuzzy Relational Clustering with Weighted Medoids"* (FMMdd) [5] utiliza de múltiplos medoides para fazer uma partição difusa que minimize a função objetivo

definida pela Equação 2.29, possuíndo os parâmetros 'm' e 'n'.

$$J = \sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D(i,j) \tag{2.29}$$

O Algoritmo FMMdd começa com uma solução aleatória para, então, alternar as atualizações das pertinências u_{ic} dos elementos e das representações v_{cj} dos *clusters*. Sendo a atualização das pertinências feita com a Equação 2.30, que possui uma restrição de soma (Equação 2.31).

$$u_{ic} = \left[\sum_{k=1}^{K} \left(\frac{\sum_{j=1}^{N} v_{cj}^{n} D(i,j)}{\sum_{j=1}^{N} v_{kj}^{n} D(i,j)} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{2.30}$$

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.31}$$

Já a atualização das representações dos *clusters* é feita com a Equação 2.32, e também possui uma restrição de soma (Equação 2.33).

$$v_{cj} = \left[\sum_{h=1}^{N} \left(\frac{\sum_{i=1}^{N} u_{ic}^{m} D(i,j)}{\sum_{i=1}^{N} u_{ic}^{m} D(i,h)} \right)^{\frac{1}{n-1}} \right]^{-1} \tag{2.32}$$

$$\sum_{j=1}^{N} v_{cj} = 1, \forall c = 1, 2, ..., K \tag{2.33}$$

Para obter essas equações de atualização, foi utilizado o método de multiplicadores de Lagrange [9]. Obtidas a partir da derivada da função Lagrangiana (Equação 2.34), baseada na sua função objetivo. Sendo λ_i e β_c os multiplicadores de Lagrange.

$$L_{FMMdd} = \sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D(i,j) - \sum_{i=1}^{N} \lambda_i \left(\sum_{c=1}^{K} u_{ic} - 1 \right) - \sum_{c=1}^{K} \beta_c \left(\sum_{j=1}^{N} v_{cj} - 1 \right) \tag{2.34}$$

Organizando essas etapas num algoritmo, temos:

Algoritmo 4: FMMdd - Algoritmo

ENTRADA

D: Matriz de Dissimilaridade NxN

m: Parâmetro de Difusão das Pertinências

n: Parâmetro de Ponderação das Representações

K: Número de *Clusters*

T_{max}: Número máximo de iterações

ϵ: Variação Máxima da Função Objetivo para considerar a solução estabilizada

SAÍDA

A Matriz $U = [u_{ic}]$ de pertinências dos N elementos

A Matriz $V = [v_{cj}]$ da representação dos Medoides dos *Clusters*

INICIALIZAÇÃO

Define $t = 0$

Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1 (1 \leq i \leq N)$

Atualiza as representações v_{ci} com a Equação 2.32

J_0 = Função Objetivo (Equação 2.29)

ITERAÇÃO

Repetir

Define $t = t + 1$

Atualiza as pertinências u_{ic} com a Equação 2.30

Atualiza as representações v_{ci} com a Equação 2.32

J_t = Função Objetivo (Equação 2.29)

Até $t \geq T_{MAX} \vee |J_t - J_{t-1}| < \epsilon$

Considerando essas etapas, podemos calcular a complexidade do algoritmo como:

$$\begin{aligned} O_{FMMdd} =& O_{inic} + T_{MAX} \times O_{Iter} \\ =& (O_{aleat} + O_J) + T_{MAX} \times (O_{Pert} + O_{Repr} + O_J) \\ =& O(NK + KN^2) + T_{MAX} \times O(NKN + KN^2 + KN^2) \\ =& O(KN^2) + T_{MAX} \times O(KN^2) \\ =& O(T_{MAX} \cdot KN^2) \end{aligned} \tag{2.35}$$

Esse artigo também falou sobre como as pertinências acabam ficando iguais caso qualquer um dos parâmetros 'm' ou 'n' seja muito grande, isso ocorre pois, se as representações fiquem distribuídas igualmente, as pertinências também ficarão (e vice-versa). Isso que demostra a importância da seleção desses parâmetros.

2.3 MODELOS *MULTI-VIEW*

Os modelos *multi-view* foram criados para os problemas que possuem múltiplas matrizes de dissimilaridade (cada uma representando uma visão diferente do problema), uma das vantagens de utilizar as múltiplas matrizes, ao invés de juntar todas em uma única, é a possibilidade de ponderar as diferentes visões (o que permite reduzir a influência de visões que não adicionam informações relevantes[1]).

A seguir serão apresentados os modelos *Clustering and Aggregating Relational Data based on RFCM* (CARD_R), *Clustering and Aggregating Relational Data based on FANNY* (CARD_F), (MFCMdd) e (FW4M), sendo esse último o modelo de referência para esse trabalho.

2.3.1 *Clustering and Aggregating Relational Data* (CARD)

O método *"Clustering and Aggregating Relational Data"* (CARD) foi proposto por Hichem Frigui et. al [6] como um método genérico para tornar modelos *single-view* em modelos modelos *multi-view*, permitindo utilizar diversos modelos já existentes para problemas *multi-view*, sem precisar juntar as visões numa só.

Primeiramente, precisa-se utilizar as múltiplas matrizes de dissimilaridade ao invés de apenas uma. Para tal, o método começa substituindo os usos de D(i, j) por uma soma ponderada das várias visões para cada *cluster* (como descrito pela Equação 2.36).

$$D(i,j) \leftarrow \sum_{p=1}^{P} \delta_{cp}^{s} D_p(i,j) \tag{2.36}$$

Esses pesos respeitam a restrição de soma (Equação 2.37) e possuem um parâmetro para controlar a sua suavidade (parâmetro s).

$$\sum_{p=1}^{P} \delta_{cp} = 1, \forall c = 1, 2, ..., K \tag{2.37}$$

Para obter a equação de atualização dos pesos das visões, deve-se utilizar a técnica de multiplicadores de Lagrange[9]. Essa equação é, então, adicionada ao passo iterativo do modelo.

Para Demostrar o funcionamento do método, o artigo o aplicou para os modelos RFCM e FANNY. Os modelos resultantes serão descritos a seguir.

2.3.1.1 CARD para o modelo RFCM (CARD_R)

O algoritmo CARD_R [6] é descrito pelo artigo como o resultado ao se aplicar o método CARD ao modelo RFCM. Contudo, ao observar suas equações, fica claro que o artigo

[1] Por informação relevante, se entende informações que ajudam a separar os *clusters*, por terem distribuições distintas entre os *clusters*

utilizou os ajustes feitos pelo modelo NERF, evitando a limitação do RFCM que só permite a utilização de matrizes euclidianas (assim como descrito na subseção 2.2.2).

O modelo CARD$_R$ minimiza a função objetivo descrita pela Equação 2.38, possuindo o parâmetro m do modelo NERF e o parâmetro s, que foi adicionado na conversão para *multi-view*.

$$J = \sum_{c=1}^{K} \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} u_{ic}^{m} u_{jc}^{m} \sum_{p=1}^{P} \delta_{cp}^{s} D_p(i,j)}{2\sum_{i=1}^{N} u_{ic}^{m}} \tag{2.38}$$

O modelo começa com uma inicialização aleatória das pertinências para, então, ser otimizado por duas equações que são executadas alternadamente: o de atualização dos pesos e o de atualização das pertinências.

A atualização dos pesos é dada pela Equação 2.39:

$$\delta_{cp} = \left[\sum_{q=1}^{P} \left(\frac{\sum_{i=1}^{N}\sum_{j=1}^{N} u_{ic}^{m} u_{jc}^{m} D_p(i,j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} u_{ic}^{m} u_{jc}^{m} D_q(i,j)} \right)^{\frac{1}{s-1}} \right]^{-1} \tag{2.39}$$

Tendo uma restrição de soma local:

$$\sum_{p=1}^{P} \delta_{cp} = 1, \forall c = 1, 2, ..., K \tag{2.40}$$

e foi encontrado utilizando a técnica de multiplicadores de Lagrange[9], sendo λ_i e β_c os multiplicadores de Lagrange.

$$L_{CARD_R} = \sum_{c=1}^{K} \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} u_{ic}^{m} u_{jc}^{m} \sum_{p=1}^{P} \delta_{cp}^{s} D_p(i,j)}{2\sum_{i=1}^{N} u_{ic}^{m}} - \sum_{i=1}^{N} \lambda_i \left(\sum_{c=1}^{K} u_{ic} - 1 \right) - \sum_{c=1}^{K} \beta_c \left(\sum_{p=1}^{P} \delta_{cp} - 1 \right) \tag{2.41}$$

Já a atualização das pertinências (Equação 2.42) foi obtida aplicando a conversão 2.36 à equação utilizada pelo algoritmo RFCM.

$$u_{ic} = \left[\sum_{k=1}^{K} \left(\frac{d_{ic}^2}{d_{ik}^2} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{2.42}$$

Sendo d_{ic}^2 calculado como:

$$d_{ic}^2 = \sum_{j=1}^{N} R_{\beta c}(i,j) v_{cj} - \frac{\sum_{j=1}^{N} v_{cj} \sum_{k=1}^{N} R_{\beta c}(j,k) v_{ck}}{2} \tag{2.43}$$

Com v_{cj} definido pela Equação 2.44 e $R_{\beta c}(i,j)$ pela Equação 2.45

$$v_{cj} = \frac{u_{jc}}{\sum_{i=1}^{N} u_{ic}} \tag{2.44}$$

$$R_{\beta c}(i,j) = \begin{cases} \sum_{p=1}^{P} \delta_{cp}^{s} D_p(i,j) & \text{, caso i = j} \\ \sum_{p=1}^{P} \delta_{cp}^{s} D_p(i,j) + \beta & \text{, caso contrário} \end{cases} \tag{2.45}$$

Se existir algum fator $d_{ic} < 0$, tanto o vetor β quanto os d_{ic} são atualizados da mesma forma da feita pelo modelo NERF (rever Equação 2.17 e Equação 2.18, na subseção 2.2.2).

Assim como com o modelo NERF, as pertinências respeitam uma restrição de soma:

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.46}$$

Organizando essas informações, temos o algoritmo:

Algoritmo 5: CARD$_R$ - Algoritmo

ENTRADA

$D = \{D_1, D_2, .., D_P\}$: P Matrizes de Dissimilaridade NxN

m: Parâmetro de Difusão das Pertinências

s: Parâmetro de Ponderação das visões

K: Número de *Clusters*

T_{max}: Número máximo de iterações

ϵ: Variação Máxima da Função Objetivo para considerar a solução estabilizada

SAÍDA

A Matriz $U = [u_{ic}]$ de pertinências dos N elementos

A Matriz $\Delta = [\delta_{cp}]$ de Pesos das Visões

INICIALIZAÇÃO

Define $t = 0$

Define $\delta_{cp} = 1/P (1 \leq c \leq K; 1 \leq p \leq P)$

Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1 (1 \leq i \leq N)$

J_0 = Função Objetivo (Equação 2.38)

ITERAÇÃO

Repetir

Define $t = t + 1$

Atualiza as ponderações δ_{cp} com a Equação 2.39

Calcula as representações v_{cj} com a Equação 2.44

Calcula os valores auxiliares d_{ic} com a Equação 2.43

Caso necessário, atualiza o fator β e os valores d_{ic} com as equações utilizadas pelo NERF (Equação 2.17 e Equação 2.18)

Atualiza as pertinências u_{ic} com a Equação 2.42

J_t = Função Objetivo (Equação 2.38)

Até $t \geq T_{MAX} \vee |J_t - J_{t-1}| < \epsilon$

Já o custo computacional do modelo CARD$_R$, considerando o algoritmo descrito an-

teriormente, fica:

$$\begin{aligned} O_{CARD_R} =& O_{inic} + T_{MAX} \times O_{Iter} \\ =& (O_\delta + O_{aleat} + O_J) \\ & + T_{MAX} \times (O_{Pond} + O_{Repr} + O_{d_{ic}} + O_{\Delta\beta} + O_{\Delta d_{ic}} + O_{Pert} + O_J) \\ =& O(KP + NK + KN^2P) \\ & + T_{MAX} \times O(KPN^2 + NK + N^2KP + NK + NK + NK + KN^2P) \\ =& O(KN^2P) + T_{MAX} \times O(KN^2P) \\ =& O(T_{MAX} \cdot KN^2P) \end{aligned} \tag{2.47}$$

Que é equivalente a $P \times O_{NERF}$.

2.3.1.2 CARD para o modelo FANNY ($CARD_F$)

O algoritmo $CARD_F$ [6] é o resultado ao se aplicar o método CARD no algoritmo FANNY (descrito na subseção 2.2.1). Ele minimiza a função objetivo descrita pela Equação 2.48, possuindo apenas o parâmetro s, que foi adicionado na conversão para *multi-view*.

$$J = \sum_{c=1}^{K} \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} u_{ic}^2 u_{jc}^2 \sum_{p=1}^{P} \delta_{cp}^s D_p(i,j)}{2\sum_{i=1}^{N} u_{ic}^2} \tag{2.48}$$

Podemos perceber que, assim como o FANNY, o $CARD_F$ sempre utiliza o parâmetro m = 2 e pondera as representações dos *clusters* utilizando as pertinências dos elementos. O modelo começa com uma inicialização aleatória para, então, ter as ponderações das visões e as pertinências atualizadas alternadamente.

A atualização dos pesos (Equação 2.49) possui uma restrição de soma (Equação 2.50).

$$\delta_{cp} = \left[\sum_{q=1}^{P} \left(\frac{\sum_{i=1}^{N}\sum_{j=1}^{N} u_{ic}^2 u_{jc}^2 D_p(i,j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} u_{ic}^2 u_{jc}^2 D_q(i,j)} \right)^{\frac{1}{s-1}} \right]^{-1} \tag{2.49}$$

$$\sum_{p=1}^{P} \delta_{cp} = 1, \forall c = 1, 2, ..., K \tag{2.50}$$

E foi encontrada utilizando a técnica de multiplicadores de Lagrange[9], sendo λ_i e β_c os multiplicadores de Lagrange.

$$L_{CARD_F} = \sum_{c=1}^{K} \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} u_{ic}^2 u_{jc}^2 \sum_{p=1}^{P} \delta_{cp}^s D_p(i,j)}{2\sum_{i=1}^{N} u_{ic}^2} - \sum_{i=1}^{N} \lambda_i \left(\sum_{c=1}^{K} u_{ic} - 1 \right) - \sum_{c=1}^{K} \beta_c \left(\sum_{p=1}^{P} \delta_{cp} - 1 \right) \tag{2.51}$$

Já a atualização das pertinências (Equação 2.52) foi obtida aplicando a conversão 2.36 à equação utilizada pelo algoritmo FANNY.

$$u_{ic} = \begin{cases} \frac{1/a_{ic}}{\sum_{k \in V_i+}(1/a_{ik})} & \text{, para } c \in V_i+ \\ 0 & \text{, caso contrário} \end{cases} \tag{2.52}$$

Na qual o conjunto V_i+ é definido na Equação 2.53 e a_{ic} pela Equação 2.54.

$$V_i+ = \left\{ c; \frac{1/a_{ic}}{\sum_{k=1}^{K}(1/a_{ik})} > 0, \ \textit{sendo } c = 1, 2, ..., K \right\} \tag{2.53}$$

$$a_{ic} = \frac{2\sum_{j=1}^{N} u_{jc}^2 \sum_{p=1}^{P} \delta_{cp}^s D(i,j)}{\sum_{j=1}^{N} u_{jc}^2} - \frac{\sum_{k=1}^{N}\sum_{j=1}^{N} u_{kc}^2 u_{jc}^2 \sum_{p=1}^{P} \delta_{cp}^s D(k,j)}{\left(\sum_{j=1}^{N} u_{jc}^2\right)^2} \tag{2.54}$$

Assim como no modelo FANNY, o modelo CARD_F possui uma restrição de soma para as pertinências (Equação 2.55), além de uma condição de parada similar.

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.55}$$

Organizando essas informações, temos o algoritmo:

Algoritmo 6: CARD$_F$ - Algoritmo

ENTRADA

$D = \{D_1, D_2, .., D_P\}$: P Matrizes de Dissimilaridade NxN

s: Parâmetro de Ponderação das visões

K: Número de *Clusters*

T_{max}: Número máximo de iterações

ϵ: Variação Máxima da Função Objetivo para considerar a solução estabilizada

SAÍDA

A Matriz $U = [u_{ic}]$ de pertinências dos N elementos

A Matriz $\Delta = [\delta_{cp}]$ de Pesos das Visões

INICIALIZAÇÃO

Define $t = 0$

Define $\delta_{cp} = 1/P (1 \leq c \leq K; 1 \leq p \leq P)$

Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1 (1 \leq i \leq N)$

J_0 = Função Objetivo (Equação 2.48)

ITERAÇÃO

Repetir

Define $t = t + 1$

Atualiza as ponderações δ_{cp} com a Equação 2.49

Calcula os valores auxiliares a_{ic} com a Equação 2.54

Encontra os conjuntos V_i+ com a Equação 2.53

Atualiza as pertinências u_{ic} com a Equação 2.52

J_t = Função Objetivo (Equação 2.48)

Até $t \geq T_{MAX} \vee |J_t - J_{t-1}| < \epsilon$

Já o custo computacional do modelo CARD$_F$, considerando o algoritmo descrito anteriormente, fica:

$$\begin{aligned} O_{CARD_F} =& O_{inic} + T_{MAX} \times O_{Iter} \\ =& (O_\delta + O_{aleat} + O_J) + T_{MAX} \times (O_{Pond} + O_{a_{ic}} + O_{V_i+} + O_{u_{ic}} + O_J) \\ =& O(KP + NK + KN^2P) \\ & + T_{MAX} \times O(KN^2P + N^2K + NKP + NK + KN^2P) \\ =& O(KN^2P) + T_{MAX} \times O(KN^2P) \\ =& O(T_{MAX} \cdot KN^2P) \end{aligned} \tag{2.56}$$

Que é equivalente a $P \times O_{FANNY}$.

2.3.2 *Partitioning Fuzzy K-Medoids Clustering Algorithm Based on Multiple Dissimilarity Matrices* (MFCMdd)

O algoritmo *"Partitioning Fuzzy K-Medoids Clustering algorithm Based on Multiple Dissimilarity Matrices"* MFCMdd [7] é uma modificação *multi-view* do algoritmo FCMdd (descrito na subseção 2.2.3) cujos *clusters* são descritos por um subconjunto G_c com tamanho pré-definido de elementos (em oposição a um único elemento, como no caso do FCMdd). Foram apresentadas, no artigo, cinco variações do modelo, cada uma com uma restrição diferente para as ponderações das visões (Como demostrado na Tabela 1).

Tabela 1 – Variações do modelo MFCMdd

Variação	Ponderação	Restrição
MFCMdd	$w_{cp} = 1$	—
MFCMdd-RWG-P	$w_{cp} = \delta_p$	$\prod_{p=1}^{P} \delta_p = 1$
MFCMdd-RWG-S	$w_{cp} = \delta_p^s$	$\sum_{p=1}^{P} \delta_p = 1$
MFCMdd-RWL-P	$w_{cp} = \delta_{cp}$	$\prod_{p=1}^{P} \delta_{cp} = 1, \forall c = 1, 2, ..., K$
MFCMdd-RWL-S	$w_{cp} = \delta_{cp}^s$	$\sum_{p=1}^{P} \delta_{cp} = 1, \forall c = 1, 2, ..., K$

Fonte: Informações de [7]

A função objetivo foi modificada para refletir tanto a mudança na representação dos *clusters* quanto a utilização de múltiplas visões (como demonstrado pela Equação 2.57). Todas as variações possuem os parâmetros 'm' e 'q', e as variações de soma (MFCMdd-RWG-S e MFCMdd-RWL-S) também possuem o parâmetro 's'. Já as ponderações por produto não possuem esse parâmetro.

$$J = \sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} w_{cp} \sum_{e \in G_c} D_p(i, e) \tag{2.57}$$

A atualização das pertinências também teve modificações semelhantes (Equação 2.58), mantendo a restrição de soma (Equação 2.59).

$$u_{ic} = \left[\sum_{k=1}^{K} \left(\frac{\sum_{p=1}^{P} w_{cp} \sum_{e \in G_c} D_p(i, e)}{\sum_{p=1}^{P} w_{kp} \sum_{e \in G_k} D_p(i, e)} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{2.58}$$

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.59}$$

A atualização das pertinências e das representações precisam ser trocados (Equação 2.58 e algoritmo 7, respectivamente).

Algoritmo 7: Etapa de representação do MFCMdd

1: **for** c = 1,2,...,K **do**
2: $G_c \leftarrow \emptyset$
3: **repeat**
4: e = $\text{argmin}_{1 \leq j \leq N} \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} w_{cp} D_p(i,j)$, sendo que $j \notin G_c$
5: $G_c \leftarrow G_c \cup e$
6: **until** $|G_c|$ = q
7: **end for**

Já a etapa de atualização dos pesos das visões vai, naturalmente, depender da variação do modelo (atualizações listadas na Tabela 2).

Tabela 2 – Atualização da Ponderação das Visões

Variação	Atualização das ponderações
MFCMdd	—
MFCMdd-RWG-P	$\delta_p = \frac{\left\{\prod_{h=1}^{P}\left[\sum_{c=1}^{K}\sum_{i=1}^{N} u_{ic}^{m} \sum_{e \in G_c} D_h(i,e)\right]\right\}^{1/P}}{\sum_{c=1}^{K}\sum_{i=1}^{N} u_{ic}^{m} \sum_{e \in G_c} D_p(i,e)}$
MFCMdd-RWG-S	$\delta_p = \left[\sum_{h=1}^{P}\left(\frac{\sum_{c=1}^{K}\sum_{i=1}^{N} u_{ic}^{m} \sum_{e \in G_c} D_p(i,e)}{\sum_{c=1}^{K}\sum_{i=1}^{N} u_{ic}^{m} \sum_{e \in G_c} D_h(i,e)}\right)^{\frac{1}{s-1}}\right]^{-1}$
MFCMdd-RWL-P	$\delta_{cp} = \frac{\left\{\prod_{h=1}^{P}\left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{e \in G_c} D_h(i,e)\right]\right\}^{1/P}}{\sum_{i=1}^{N} u_{ic}^{m} \sum_{e \in G_c} D_p(i,e)}$
MFCMdd-RWL-S	$\delta_{cp} = \left[\sum_{h=1}^{P}\left(\frac{\sum_{i=1}^{N} u_{ic}^{m} \sum_{e \in G_c} D_p(i,e)}{\sum_{i=1}^{N} u_{ic}^{m} \sum_{e \in G_c} D_h(i,e)}\right)^{\frac{1}{s-1}}\right]^{-1}$

Fonte: Informações de [7]

O algoritmo é organizado da forma:

Algoritmo 8: MFCMdd - Algoritmo

ENTRADA

$D = \{D_1, D_2, .., D_P\}$: P Matrizes de Dissimilaridade NxN

m: Parâmetro de Difusão das Pertinências

q: Parâmetro do tamanho dos representantes de cada *cluster*

s: Parâmetro de Ponderação das visões (para as variantes com restrição de soma)

K: Número de *Clusters*

T_{max}: Número máximo de iterações

ϵ: Variação Máxima da Função Objetivo para considerar a solução estabilizada

SAÍDA

A Matriz $U = [u_{ic}]$ de pertinências dos N elementos

O Vetor $\Delta = [\delta_p]$ de Pesos das Visões (para as variantes com ponderação global)

A Matriz $\Delta = [\delta_{cp}]$ de Pesos das Visões (para as variantes com ponderação local)

INICIALIZAÇÃO

Define $t = 0$

Define a ponderação inicial:

Define $\delta_p = 1 (1 \leq p \leq P)$, para a variante MFCMdd-RWG-P

Define $\delta_p = 1/P (1 \leq p \leq P)$, para a variante MFCMdd-RWG-S

Define $\delta_{cp} = 1 (1 \leq c \leq K; 1 \leq p \leq P)$, para a variante MFCMdd-RWL-P

Define $\delta_{cp} = 1/P (1 \leq c \leq K; 1 \leq p \leq P)$, para a variante MFCMdd-RWL-S

Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1 (1 \leq i \leq N)$

Seleciona os representantes G_c aleatoriamente, evitando repetição

J_0 = Função Objetivo (Equação 2.57)

ITERAÇÃO

Repetir

Define $t = t + 1$

Atualiza as pertinências u_{ic} com a Equação 2.58

Atualiza os representantes G_c com o algoritmo 7

Atualiza as ponderações δ_p ou δ_{cp} com a equação na Tabela 2 referente à variante

J_t = Função Objetivo (Equação 2.57)

Até $t \geq T_{MAX} \vee |J_t - J_{t-1}| < \epsilon$

Como os passos podem mudar entre as variações, precisamos calcular a complexidade para cada tipo de ponderação (sem ponderação, ponderação global e ponderação local).

Dessa forma, temos que a complexidade do modelo sem ponderação fica:

$$\begin{aligned} O_{MFCMdd} =& O_{inic} + T_{MAX} \times O_{iter} \\ =& (O_{aleat} + O_J) + T_{MAX} \times (O_{Pert} + O_{Repr} + O_J) \\ =& O((NK + Kq) + KNPq) + T_{MAX} \times O(NKPq + KN^2P + KqNP) \\ =& O(KNPq) + T_{MAX} \times O(KN^2P) \\ =& O(T_{MAX} \cdot KN^2P) \end{aligned} \tag{2.60}$$

A complexidade dos modelos com restrição global fica:

$$\begin{aligned} O_{MFCMdd-RWG} =& O_{inic} + T_{MAX} \times O_{iter} \\ =& (O_{aleat} + O_{\delta_p} + O_J) + T_{MAX} \times (O_{Pert} + O_{Repr} + O_{Pond} + O_J) \\ =& O((NK + Kq) + P + KNPq) \\ & + T_{MAX} \times O(NKPq + KN^2P + PKNq + KqNP) \\ =& O(KNPq) + T_{MAX} \times O(KN^2P) \\ =& O(T_{MAX} \cdot KN^2P) \end{aligned} \tag{2.61}$$

Já a complexidade dos modelos com restrição local fica:

$$\begin{aligned} O_{MFCMdd-RWL} =& O_{inic} + T_{MAX} \times O_{iter} \\ =& (O_{aleat} + O_{\delta_{cp}} + O_J) + T_{MAX} \times (O_{Pert} + O_{Repr} + O_{Pond} + O_J) \\ =& O((NK + Kq) + KP + KNPq) \\ & + T_{MAX} \times O(NKPq + KN^2P + PKNq + KqNP) \\ =& O(KNPq) + T_{MAX} \times O(KN^2P) \\ =& O(T_{MAX} \cdot KN^2P) \end{aligned} \tag{2.62}$$

Como podemos observar, a complexidade de todas as variantes é O($T_{MAX} \cdot KN^2P$), sendo definida pelo passo de atualização das representações.

Esse artigo apresentou diferentes restrições possíveis para as ponderações das visões. É possível perceber que as variantes que utilizaram uma restrições de produto (RWG-P e RWL-P) não possuem o parâmetro s utilizado pelas variantes com a restrição de soma (RWG-S e RWL-S). Isso serviu de inspiração para o objetivo principal do modelo proposto (a redução do número de parâmetro do modelo de referência FW4M, que será descrito a seguir).

2.3.3 *Fuzzy Clustering Algorithm Based on Weighted Multi-Medoids and Multi-Matrices* (FW4M)

O modelo *"Fuzzy Clustering Algorithm Based on Weighted Multi-Medoids and Multi-Matrices"* (FW4M) [8] combina a representação do *clusters* apresentada pelo modelo FMMdd (subseção 2.2.4) com a ponderação de múltiplas visões apresentada pelo modelo

MFCMdd-RWG-S (subseção 2.3.2), ficando com uma função objetivo que possui ambas as características (como demonstrada pela a Equação 2.63), possuindo 3 parâmetros: 'm', 'n' e 's'.

$$J = \sum_{c=1}^{K}\sum_{i=1}^{N} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p^s D_p(i,j) \tag{2.63}$$

Sendo que as pertinências dos elementos, as representações dos *clusters* e as ponderações das visões respeitam restrições de soma (Equação 2.64, Equação 2.65 e Equação 2.66, respectivamente).

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.64}$$

$$\sum_{j=1}^{N} v_{cj} = 1, \forall c = 1, 2, ..., K \tag{2.65}$$

$$\sum_{p=1}^{P} \delta_p = 1 \tag{2.66}$$

Para se obter as equações da otimização, eles também utilizaram o método de Multiplicadores de Lagrange[9] (com a equação de Lagrangiana 2.67).

$$\begin{aligned} L_{FW4M} = &\sum_{c=1}^{K}\sum_{i=1}^{N} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p^s D_p(i,j) - \sum_{i=1}^{N} \alpha_i \left(\sum_{c=1}^{K} u_{ic} - 1 \right) \\ &- \sum_{c=1}^{K} \beta_c \left(\sum_{j=1}^{N} v_{cj} - 1 \right) - \gamma \left(\sum_{p=1}^{P} \delta_p - 1 \right) \end{aligned} \tag{2.67}$$

Sendo α_i, β_c e γ os multiplicadores de Lagrange. Para buscar os valores que minimizam a função, deve-se zerar as derivadas de L_{FW4M}. Dessa forma, são obtidas as equações de atualização das pertinências, das representações e dos pesos (Equação 2.68, Equação 2.69 e Equação 2.70, respectivamente).

$$u_{ic} = \frac{\left[\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right]^{\frac{-1}{m-1}}}{\sum_{k=1}^{K} \left[\sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right]^{\frac{-1}{m-1}}} \tag{2.68}$$

$$v_{cj} = \frac{\left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j) \right]^{\frac{-1}{n-1}}}{\sum_{k=1}^{N} \left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,k) \right]^{\frac{-1}{n-1}}} \tag{2.69}$$

$$\delta_p = \frac{\left[\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j) \right]^{\frac{-1}{s-1}}}{\sum_{h=1}^{P} \left[\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_h(i,j) \right]^{\frac{-1}{s-1}}} \tag{2.70}$$

A organização do modelo FW4M pode ser melhor observada no algoritmo seguinte:

Algoritmo 9: FW4M - Algoritmo

ENTRADA

$D = \{D_1, D_2, .., D_P\}$: P Matrizes de Dissimilaridade NxN

m: Parâmetro de Difusão das Pertinências

n: Parâmetro de Ponderação das Representações

s: Parâmetro de Ponderação das visões

K: Número de *Clusters*

T_{max}: Número máximo de iterações

ϵ: Variação Máxima da Função Objetivo para considerar a solução estabilizada

SAÍDA

A Matriz $U = [u_{ic}]$ de pertinências dos N elementos

A Matriz $V = [v_{cj}]$ da representação dos Medoides dos *Clusters*

O Vetor $\Delta = [\delta_p]$ de Pesos das Visões

INICIALIZAÇÃO

Define $t = 0$

Define $\delta_p = 1/P (1 \leq p \leq P)$

Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1 (1 \leq i \leq N)$

Atualiza as representações v_{ci} com a Equação 2.69

J_0 = Função Objetivo (Equação 2.63)

ITERAÇÃO

Repetir

Define $t = t + 1$

Atualiza as pertinências u_{ic} com a Equação 2.68

Atualiza as representações v_{ci} com a Equação 2.69

Atualiza as ponderações δ_p com a Equação 2.70

J_t = Função Objetivo (Equação 2.63)

Até $t \geq T_{MAX} \vee |J_t - J_{t-1}| < \epsilon$

Computando a complexidade do modelo FW4M, temos:

$$
\begin{aligned}
O_{FW4M} =& O_{inic} + T_{MAX} \times O_{Iter} \\
=& (O_{\delta_p} + O_{aleat} + O_{Repr} + O_J) + T_{MAX} \times (O_{Pert} + O_{Repr} + O_{Pond} + O_J) \\
=& O(P + NK + KN^2P + KN^2P) \\
& + T_{MAX} \times O(N^2KP + KN^2P + PLN^2 + KN^2P) \\
=& O(KN^2P) + O(T_{MAX} \cdot KN^2P) \\
=& O(T_{MAX} \cdot KN^2P)
\end{aligned}
\tag{2.71}
$$

2.4 RESUMO DOS MODELOS VISTOS

Como podemos perceber, todos os modelos possuem uma estrutura similar, sendo diferentes na função objetivo que será minimizada e nas equações de atualização de cada etapa para atingir essa minimização (essa estrutura pode ser representada pelo algoritmo 10).

Algoritmo 10: Estrutura dos algoritmos da área

1: inicialização aleatória da solução
2: **Repetir**
3: Atualiza as pertinências dos elementos u_{ic}
4: Atualiza as representações dos *clusters* v_{cj} (se possuir)
5: Atualiza os pesos das visões δ (caso seja *multi-view*)
6: **Até** t $= T_{MAX}$ ou $|J_t - J_{t-1}| < \epsilon$

Possuindo restrições que limitam as pertinências (Equação 2.72 e Equação 2.73)

$$u_{ic} \geq 0, \forall i, c \tag{2.72}$$

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{2.73}$$

e restrições para as representações (Equação 2.74 e Equação 2.75)

$$v_{cj} \geq 0, \forall c, j \tag{2.74}$$

$$\sum_{j=1}^{N} v_{cj} = 1, \forall c = 1, 2, ..., K \tag{2.75}$$

Já a restrição das ponderações vai depender do modelo, como vimos com as variações do modelo MFCMdd. Podendo ser de soma global (Equação 2.76), de soma local (Equação 2.77), de produto global (Equação 2.78) ou de produto local (Equação 2.79).

$$\sum_{p=1}^{P} \delta_p = 1 \tag{2.76}$$

$$\sum_{p=1}^{P} \delta_{cp} = 1, \forall c = 1, 2, ..., K \tag{2.77}$$

$$\prod_{p=1}^{P} \delta_p = 1 \tag{2.78}$$

$$\prod_{p=1}^{P} \delta_{cp} = 1, \forall c = 1, 2, ..., K \tag{2.79}$$

Sendo que as ponderações também não podem ser negativas, além de que as ponderações nas restrições de produto também não podem ser nulas (pois se existir uma ponderação nula, o produto também vai ser nulo).

Para uma melhor comparação dos modelos, eles foram listados a seguir, com suas funções-objetivo e seus parâmetros:

Tabela 3 – Comparação dos modelos da literatura

Modelo	Função Objetivo	Parâmetros
FANNY	$\sum_{c=1}^{K} \frac{\sum_{i=1}^{N} u_{ic}^2 \sum_{j=1}^{N} u_{jc}^2 D(i,j)}{2\sum_{j=1}^{N} u_{jc}^2}$	—
NERF	$\sum_{c=1}^{K} \frac{\sum_{i=1}^{N} u_{ic}^m \sum_{j=1}^{N} u_{jc}^m D(i,j)}{2\sum_{i=1}^{N} u_{ic}^m}$	m
FCMdd	$\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m D(i, g_c)$	m
FMMdd	$\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m \sum_{j=1}^{N} v_{cj}^n D(i,j)$	m, n
CARD_R	$\sum_{c=1}^{K} \frac{\sum_{i=1}^{N} u_{ic}^m \sum_{j=1}^{N} u_{jc}^m \sum_{p=1}^{P} \delta_{cp}^s D_p(i,j)}{2\sum_{i=1}^{N} u_{ic}^m}$	m, s
CARD_F	$\sum_{c=1}^{K} \frac{\sum_{i=1}^{N} u_{ic}^2 \sum_{j=1}^{N} u_{jc}^2 \sum_{p=1}^{P} \delta_{cp}^s D_p(i,j)}{2\sum_{i=1}^{N} u_{ic}^2}$	s
MFCMdd	$\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \sum_{e \in G_c} D_p(i,e)$	m, q
MFCMdd-RWG-P	$\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_p \sum_{e \in G_c} D_p(i,e)$	m, q
MFCMdd-RWG-S	$\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_p^s \sum_{e \in G_c} D_p(i,e)$	m, q, s
MFCMdd-RWL-P	$\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_{cp} \sum_{e \in G_c} D_p(i,e)$	m, q
MFCMdd-RWL-S	$\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_{cp}^s \sum_{e \in G_c} D_p(i,e)$	m, q, s
FW4M	$\sum_{c=1}^{K} \sum_{i=1}^{N} u_{ic}^m \sum_{j=1}^{N} v_{cj}^n \sum_{p=1}^{P} \delta_p^s D_p(i,j)$	m, n, s

Fonte: Informações de [3], [2], [4], [5], [6], [7] e [8]

Também é possível perceber que todos os modelos *single-view* citados possuem uma complexidade $\text{O}(T_{MAX} \cdot KN^2)$, já os modelos *multi-view* citados possuem uma complexidade $\text{O}(T_{MAX} \cdot KN^2P)$. Isso faz sentido, já que a complexidade de iterar as P matrizes de dissimilaridade com tamanho NxN é $\text{O}(N^2P)$. A complexidade encontrada é atingida ao se repetir essa iteração para cada um dos K *clusters* T_{MAX} vezes.

3 MODELO PROPOSTO

Esse capítulo apresentará o modelo proposto, incluindo o seu algoritmo e parâmetros, a função objetivo com suas restrições, a inicialização da solução, as equações para otimizar a solução e a complexidade de cada passo e do algoritmo como um todo.

Nesse trabalho estamos propondo o modelo *"Multi-view Relational Fuzzy Clustering Algorithm with Multi-medoids"* (MFMMdd), que foi publicado na conferência ISNN 2019 [10]. Ele é baseado no modelo FW4M (subseção 2.3.3) porém reduzindo o número de parâmetros. Para tal, foram estudadas as variações MFCMdd-RWG-S e MFCMdd-RWG-P do algoritmo MFCMdd (subseção 2.3.2) que demostraram ser possível remover o parâmetro s ao substituir a restrição de soma (Equação 2.66) utilizada pelo modelo FW4M pela de produto (Equação 3.1).

$$\prod_{p=1}^{P} \delta_p = 1 \tag{3.1}$$

Com essa substituição, a função objetivo que será minimizada, tem o parâmetro s removido (como demonstrado pela Equação 3.2), continuando com os parâmetros m e n.

$$J = \sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \tag{3.2}$$

Essa função objetivo possui as restrições:

$$\sum_{c=1}^{K} u_{ic} = 1, \forall i = 1, 2, ..., N \tag{3.3}$$

$$\sum_{j=1}^{N} v_{cj} = 1, \forall c = 1, 2, ..., K \tag{3.4}$$

Além da restrição de produto da ponderação das visões, descrita previamente pela Equação 3.1.

3.1 INICIALIZAÇÃO

Assim como nos modelos da literatura, as pertinências são inicializadas aleatoriamente, de forma que a sua restrição seja respeitada. Já as ponderações das visões são inicializadas com 1, já que não se sabe ainda qual delas é mais relevante. Isso respeita a sua restrição. Por fim, as representações são calculadas com o mesmo passo utilizado na iteração para atualizá-las.

3.2 OBTENÇÃO DAS EQUAÇÕES DA OTIMIZAÇÃO

Assim como foi feito para os métodos explicados no Capítulo 2, também foi utilizado o método dos multiplicadores de Lagrange[9] para se obter as equações de otimização das suas variáveis (u_{ic}, v_{cj} e δ_p), o método de multiplicadores de Lagrange garante que os passos encontrados minimizam a função objetivo, respeitando as restrições.

Juntando a função objetivo e suas restrições, a função de Lagrange do modelo fica:

$$\begin{aligned} L_{MFMMdd} = & \sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) - \sum_{i=1}^{N} \alpha_i \left(\sum_{c=1}^{K} u_{ic} - 1 \right) \\ & - \sum_{c=1}^{K} \beta_c \left(\sum_{j=1}^{N} v_{cj} - 1 \right) - \gamma \left(\prod_{p=1}^{P} \delta_p - 1 \right) \end{aligned} \tag{3.5}$$

sendo α_i, β_c e γ os multiplicadores de Lagrange.

Os pontos de mínimo de uma equação zeram as suas derivada, pois se a derivada não fosse zero, ainda seria possível encontrar um ponto inferior ao seguir a direção oposta da derivada. Dessa forma, devemos então encontrar as soluções que zerem as derivadas de L_{MFMMdd} para obter as equações que minimizam a função. Ou seja, fazer $\frac{\partial L_{MFMMdd}}{\partial u_{ic}} = 0$, $\frac{\partial L_{MFMMdd}}{\partial v_{cj}} = 0$, $\frac{\partial L_{MFMMdd}}{\partial \delta_p} = 0$, $\frac{\partial L_{MFMMdd}}{\partial \alpha_i} = 0$, $\frac{\partial L_{MFMMdd}}{\partial \beta_c} = 0$ e $\frac{\partial L_{MFMMdd}}{\partial \gamma} = 0$

Solucionando esse sistema, obtemos as equações para otimização das pertinências dos elementos (u_{ic}), das representações dos *clusters* (v_{cj}) e das ponderações das visões (δ_p). O passo-a-passo dessa solução pode ser encontrado no Apêndice A, para evitar sobrecarregar esse capítulo com contas.

3.3 COMPUTAÇÃO DAS PERTINÊNCIAS

Nesse passo, as ponderações $\Delta = [\delta_p]$ e as representações $V = [[v_{cj}]]$ são mantidas constantes e a Matriz de pertinência dos elementos $U = [[u_{ic}]]$ é atualizada utilizando o algoritmo a seguir.

Algoritmo 11: MFMMdd - Alocação dos Elementos

1: **Repetir para cada** $i = 1, ..., N$
2: **Repetir para cada** $c = 1, ..., K$
3: $$u_{ic} = \frac{\left[\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{m-1}}}{\sum_{k=1}^{K} \left[\sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{m-1}}}$$
4: **Fim da Repetição**
5: **Fim da Repetição**

3.4 COMPUTAÇÃO DA REPRESENTAÇÃO DOS MEDOIDES

Nesse passo, as ponderações $\Delta = [\delta_p]$ e as pertinências $U = [[u_{ic}]]$ são mantidas constantes e a Matriz de representação dos grupos $V = [[v_{cj}]]$ é atualizada utilizando o algoritmo a

seguir.

Algoritmo 12: MFMMdd - Representação dos Grupos

1: **Repetir para cada** $c = 1, ..., K$
2: **Repetir para cada** $j = 1, ..., N$
3: $$v_{cj} = \frac{\left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{n-1}}}{\sum_{k=1}^{N}\left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,k)\right]^{\frac{-1}{n-1}}}$$
4: **Fim da Repetição**
5: **Fim da Repetição**

3.5 COMPUTAÇÃO DA PONDERAÇÃO DAS VISÕES

Nesse passo, as representações $V = [[v_{cj}]]$ e as pertinências $U = [[u_{ic}]]$ são mantidas constantes e o vetor de ponderação das visões $\Delta = [\delta_p]$ é atualizado utilizando o algoritmo a seguir (O que respeita a restrição do produto definida na Equação 3.1).

Algoritmo 13: MFMMdd - Ponderação das Visões

1: $V2 = \emptyset$
2: **Repetir para cada** $p = 1$ to P
3: **Se** $\sum_{i=1}^{N} \sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j) > \theta$ **então**
4: $V2 = V2 \cup \{p\}$
5: **Fim do Se**
6: **Fim da Repetição**
7: $X = \left[\prod_{p \in V2} \delta_p\right]^{1/|V2|}$
8: **Repetir para cada** p em $V2$
9: $$\delta_p = X \cdot \frac{\prod_{q \in V2}\left[\sum_{i=1}^{N} \sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_q(i,j)\right]^{1/|V2|}}{\sum_{i=1}^{N} \sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)}$$
10: **Fim da Repetição**

O uso do grupo V2 foi necessário, já que o produtório pode zerar se um dos elementos for igual a zero, ou tão próximo de zero que um *underflow*[1] ocorra. Esse tipo de problema pode ser ignorado na lógica matemática, mas não num sistema computacional. Uma alternativa seria somar um pequeno valor no fator calculado, mas como os fatores não costumam zerar, isso adicionaria um ruído desnecessariamente. Dessa forma, é melhor não atualizar os elementos que zeram o cálculo (sendo θ o limiar que define o quão próximo o número pode ser de zero).

Para que os pesos continuem respeitando a restrição de produto, os valores precisam ser ajustados pelo fato X (o produto dos valores antigos dos elementos que serão atualizados), a prova de que essa atualização respeita essa restrição pode ser feita utilizando uma prova

[1] *underflow* é a situação onde o módulo do número calculado é menor do que o valor mínimo, não zero, que a estrutura de dados utilizada consegue representar, fazendo com que o número seja aredondado para zero.

iterativa. Sendo a prova do passo iterativo dada por:

$$
\begin{aligned}
\prod_{p=1}^{P} \delta_p &= \left(\prod_{p\in V2} \delta_p\right)\left(\prod_{p\notin V2} \delta_p\right) \\
&= \prod_{p\in V2}\left[X \cdot \frac{\prod_{q\in V2}\left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_q(i,j)\right]^{1/|V2|}}{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)}\right]\left(\prod_{p\notin V2} \delta'_p\right) \\
&= X^{|V2|} \cdot \frac{\prod_{p\in V2}\prod_{q\in V2}\left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_q(i,j)\right]^{1/|V2|}}{\prod_{p\in V2}\left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right]}\left(\prod_{p\notin V2} \delta'_p\right) \\
&= X^{|V2|} \cdot \frac{\prod_{q\in V2}\left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_q(i,j)\right]}{\prod_{p\in V2}\left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right]}\left(\prod_{p\notin V2} \delta'_p\right) \\
&= X^{|V2|}\left(\prod_{p\notin V2} \delta'_p\right) \\
&= \left(\prod_{p\in V2} \delta'_p\right)\left(\prod_{p\notin V2} \delta'_p\right) \\
&= \prod_{p=1}^{P} \delta'_p \\
&= 1
\end{aligned}
\tag{3.6}
$$

Com δ'_p o valor anterior da p-ésima ponderação e δ_p o seu novo valor. Já para provar o caso base, as ponderações precisam ser inicializadas respeitando a restrição (O que também é verdade, já que elas são inicializadas com 1, e é trivial de demonstrar que $\prod_{p=1}^{P} 1 = 1$). Sendo assim, a atualização das ponderações respeita a restrição sobre as mesmas.

3.6 ALGORITMO PROPOSTO

Algoritmo 14: MFMMdd - Algoritmo

ENTRADA
$D = \{D_1, D_2, .., D_P\}$: P Matrizes de Dissimilaridade NxN
m: Parâmetro de Difusão das Pertinências
n: Parâmetro de Ponderação das Representações
K: Número de *Clusters*
T_{max}: Número máximo de iterações
ϵ: Variação Máxima da Função Objetivo para parar a execução
θ: Limite para evitar divisão por zero na atualização das ponderações das visões
SAÍDA
A Matriz $U = [[u_{ic}]]$ de pertinências dos N elementos
A Matriz $V = [[v_{cj}]]$ da representação dos Medoides dos *Clusters*
O Vetor $\Delta = [\delta_p]$ de Pesos das Visões
INICIALIZAÇÃO
Define $t = 0$
Define $\delta_p = 1, p = 1, 2, ..., P$
Define u_{ic} com valores aleatórios não negativos, tal que $\sum_{c=1}^{K} u_{ic} = 1, i = 1, 2, ..., N$
Atualiza as representações v_{ci} com o algoritmo 12
J_0 = Função Objetivo (Equação 3.2)
ITERAÇÃO
Repetir
Define $t = t + 1$
Passo de Alocação
Atualiza as pertinências u_{ic} com o algoritmo 11
Atualiza as representações v_{ci} com o algoritmo 12
Atualiza as ponderações δ_p com o algoritmo 13
J_t = Função Objetivo (Equação 3.2)
Até $t \geq T_{MAX} \vee |J_t - J_{t-1}| < \epsilon$

3.7 COMPLEXIDADE E OTIMIZAÇÃO

3.7.1 Complexidade da Função Objetivo

A complexidade da função objetivo é:

$$\begin{aligned} O_J &= O_{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^m \sum_{j=1}^{N} v_{cj}^n \sum_{p=1}^{P} \delta_p D_p(i,j)} \\ &= O(NKNPc) \\ &= O(N^2KP) \end{aligned} \tag{3.7}$$

3.7.2 Complexidade da Alocação

A complexidade do passo de Alocação seria:

$$\begin{aligned} O_{Aloc} &= NK \times O\left\{ \frac{\left[\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{m-1}}}{\sum_{k=1}^{K}\left[\sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{m-1}}} \right\} \\ &= NK \times O(NPc_1 + KNPc_2) \\ &= NK \times O(KNP) \\ &= O(N^2K^2P) \end{aligned} \tag{3.8}$$

Contudo, é possível otimizar o passo ao calcular apenas os numeradores, somá-los para, então, fazer a divisão (como demostrado no algoritmo 15)

Algoritmo 15: MFMMdd - Alocação dos Elementos (Otimizado)

1: **Repetir para cada** $i = 1, ..., N$
2: Define denominador = 0
3: **Repetir para cada** $c = 1, ..., K$
4: $u_{ic} = \left[\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{m-1}}$
5: Define denominador = denominador + u_{ic}
6: **Fim da Repetição**
7: **Repetir para cada** $c = 1, ..., K$
8: $u_{ic} = \frac{u_{ic}}{denominador}$
9: **Fim da Repetição**
10: **Fim da Repetição**

Dessa forma, a complexidade do passo fica:

$$\begin{aligned} O_{Aloc} &= N \times \left[K \times O_{\left[\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{m-1}}} + K \times O_{\frac{u_{ic}}{denominador}} \right] \\ &= N \times [K \times O(NP) + K \times O(1)] \\ &= N \times O(KNP) \\ &= O(N^2KP) \end{aligned} \tag{3.9}$$

3.7.3 Complexidade da Representação

A complexidade do passo de Representação seria:

$$\begin{aligned} O_{Repr} &= KN \times O\left\{ \frac{\left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{n-1}}}{\sum_{k=1}^{N}\left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,k)\right]^{\frac{-1}{n-1}}} \right\} \\ &= KN \times O(NP + NNP) \\ &= KN \times O(N^2 P) \\ &= O(KN^3 P) \end{aligned} \tag{3.10}$$

Contudo, é possível otimizar o passo ao calcular apenas os numeradores, somá-los para, então, fazer a divisão (como demostrado no algoritmo 16)

Algoritmo 16: MFMMdd - Representação dos Grupos (Otimizado)

1: **Repetir para cada** $c = 1, ..., K$
2: Define denominador = 0
3: **Repetir para cada** $j = 1, ..., N$
4: $v_{cj} = \left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{n-1}}$
5: Define denominador = denominador + v_{cj}
6: **Fim da Repetição**
7: **Repetir para cada** $j = 1, ..., N$
8: $v_{cj} = \frac{v_{cj}}{denominador}$
9: **Fim da Repetição**
10: **Fim da Repetição**

Dessa forma, a complexidade do passo fica:

$$\begin{aligned} O_{Repr} &= K \times \left[N \times O_{\left[\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j)\right]^{\frac{-1}{n-1}}} + N \times O_{\frac{v_{cj}}{denominador}} \right] \\ &= K \times [N \times O(NP) + N \times O(1)] \\ &= K \times O(N^2 P) \\ &= O(KN^2 P) \end{aligned} \tag{3.11}$$

3.7.4 Complexidade da Ponderação

A complexidade do passo de Ponderação seria:

$$\begin{aligned} O_{Pond} &= P \times O_{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)} + O_{\left[\prod_{p\in V2}\delta_p\right]^{1/|V2|}} \\ &\quad + P \times O\left\{ X \cdot \frac{\prod_{q\in V2}\left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_q(i,j)\right]^{1/|V2|}}{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)} \right\} \\ &= P \times O(NKN) + O(1) + P \times O(PNKN + NKN) \\ &= O(PN^2K) + O(1) + P \times O(PN^2K) \\ &= O(PN^2K) + O(1) + O(P^2N^2K) \\ &= O(P^2N^2K) \end{aligned} \tag{3.12}$$

Contudo, é possível otimizar o passo ao calcular apenas os denominadores, multiplicá-los para, então, fazer a divisão (como demostrado no algoritmo 17)

Algoritmo 17: MFMMdd - Ponderação das Visões (Otimizado)

1: $numerador = 1$
2: $V2 = \emptyset$
3: **Repetir para cada** $p = 1, ..., P$
4: $\quad temp = \sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)$
5: $\quad$ **Se** $temp > \theta$ **então**
6: $\quad\quad numerador = numerador \times \delta_p \times temp$
7: $\quad\quad \delta_p = temp$
8: $\quad\quad V2 = V2 \cup \{p\}$
9: $\quad$ **Fim do Se**
10: **Fim da Repetição**
11: **Se** $|V2| > 0$ **então**
12: $\quad numerador = numerador^{1/|V2|}$
13: $\quad$ **Repetir para cada** p in $V2$
14: $\quad\quad \delta_p = \frac{numerador}{\delta_p}$
15: $\quad$ **Fim da Repetição**
16: **Fim do Se**

Dessa forma, a complexidade do passo fica:

$$\begin{aligned} O_{Pond} &= P \times \left[O_{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m}\sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)} + O_{numerador\times\delta_p\times temp} + O_{V2\cup\{p\}} \right] \\ &\quad + O_{numerador^{1/|V2|}} + P \times O_{\frac{numerador}{\delta_p}} \\ &= P \times [O(NKN) + O(1) + O(1)] + O(1) + P \times O(1) \\ &= P \times O(NKN) + O(P) \\ &= O(PN^2K) + O(P) \\ &= O(PN^2K) \end{aligned} \tag{3.13}$$

3.7.5 Complexidade da Inicialização

A complexidade da Inicialização é:

$$\begin{aligned} O_{Inic} &= O_{init(\delta_p)} + O_{init(u_{ic})} + O_{Repr} + O_J \\ &= O(P) + O(NK) + O(KN^2P) + O(N^2KP) \\ &= O(KN^2P) \end{aligned} \tag{3.14}$$

3.7.6 Complexidade do Algoritmo

Para calcularmos a complexidade do algoritmo, precisamos da complexidade da sua inicialização (O_{Inic}) e da iteração (O_{Iter}).

$$O_{MFMMdd} = O_{Inic} + T_{max} \times O_{Iter} \tag{3.15}$$

Sendo que, a complexidade da iteração é a soma das complexidades do passo de Alocação (O_{Aloc}), de Representação (O_{Repr}) e de Ponderação (O_{Pond}) além da verificação da condição de parada (que possui a complexidade do cálculo da função objetivo). Com isso temos:

$$\begin{aligned} O_{MFMMdd} &= O_{Inic} + T_{max} \times [O_{Aloc} + O_{Repr} + O_{Pond} + O_J] \\ &= O(KN^2P) + T_{max} \times \Big[O(N^2KP) + O(KN^2P) + O(PN^2K) + O(N^2KP)\Big] \\ &= O(KN^2P) + T_{max} \times O(KN^2P) \\ &= O(KN^2P) + O(T_{max} \cdot KN^2P) \\ &= O(T_{max} \cdot KN^2P) \end{aligned} \tag{3.16}$$

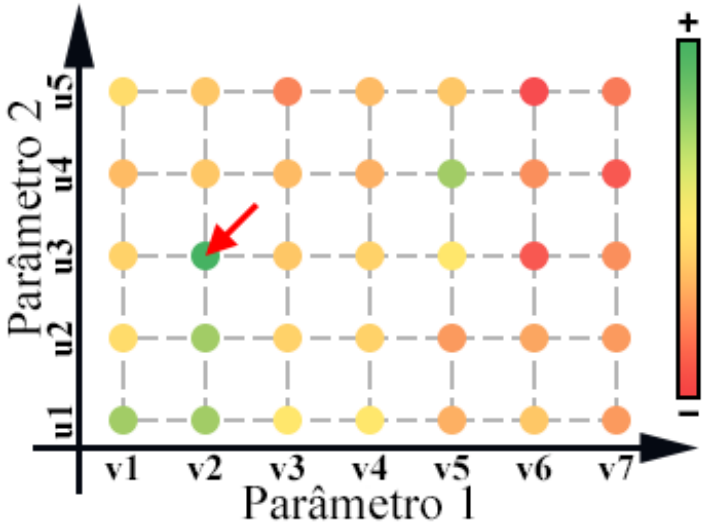
4 METODOLOGIA

Esse capítulo apresentará a metodologia utilizada pelo trabalho para validar o modelo proposto e como ele será comparado com modelos relacionados. Incluindo informações sobre a busca dos parâmetros, a forma de comparar os modelos, as métricas calculadas (tanto para a partição *crisp*, quanto para a difusa) e as bases de dados utilizadas.

4.1 BUSCA DOS PARÂMETROS

Os parâmetros para cada um dos modelos testados (MFMMdd, FW4M, CARD-R e MFCMdd-RWG-P) foram encontrados utilizando uma busca em grade, que é uma busca exaustiva dentro de um subconjunto de valores para os parâmetros do modelo, sendo o método mais conhecido de obtenção de parâmetros. Cada combinação de valores para os parâmetros é testada e a que obtiver o melhor resultado para uma determinada métrica é escolhida como os parâmetros a serem utilizados pelo modelo na execução principal.

Figura 2 – Visualização de uma busca em grade



Fonte: O Autor (2019)

Os parâmetros 'm', 'n' e 's' costumam possuir valores entre 1 e 2, aumentando a separação das distribuições referentes aos mesmo o quanto mais próximo forem de 1, sendo equivalente a um modelo *crisp* se forem iguais a 1. Considerando isso, as opções escolhidas na busca para cada métrica foram:

- m = 1.1, 1.3, 1.5, 1.7
- n = 1.1, 1.3, 1.5
- s = 1.1, 1.3, 1.5

- q = 2, 3, 5, 7

Adicionando as opções m=1.05, n=1.05 e s=1.05 caso nenhum modelo obtenha um resultado satisfatório para a base. Esses valores foram escolhidos de forma empírica com conhecimentos de outros modelos que implementamos no passado. Os valores de m, n e s precisam ser maiores do que 1 para evitar uma divisão por zero nas etapas de otimização.

Foram escolhidos os parâmetros que maximizem a média do coeficiente de partição modificado, já que essa métrica representa a separação das partições (ver seção 4.3.2.4) de 20 execuções com inicializações aleatórias (para reduzi a influência da inicialização na escolha). Com isso, temos que o número de execuções de cada modelo para busca de parâmetros foi:

Tabela 4 – Parâmetros Selecionados

Modelo	Parâmetros	Número de execuções
MFMMdd	m, n	240
FW4M	m, n, s	720
CARD_R	m, s	240
MFCMdd-RWG-P	m, q	320

Fonte: O Autor (2019)

Como pode-se observar, o modelo proposto necessitou de menos execuções para busca de parâmetros, sendo o mesmo para o modelo CARD_R. Já o modelo de referência precisou de 3x mais execuções (o número de valores testados para o parâmetro s).

4.2 COMPARAÇÃO DOS MODELOS

Com os parâmetros escolhidos, os algoritmos são executados 100 vezes com inicializações aleatórias, calculando as médias e desvios padrão das métricas escolhidas (equações 4.1 e 4.2, respectivamente).

$$\mu_x = \frac{\sum_{i=1}^{n} x_i}{n} \tag{4.1}$$

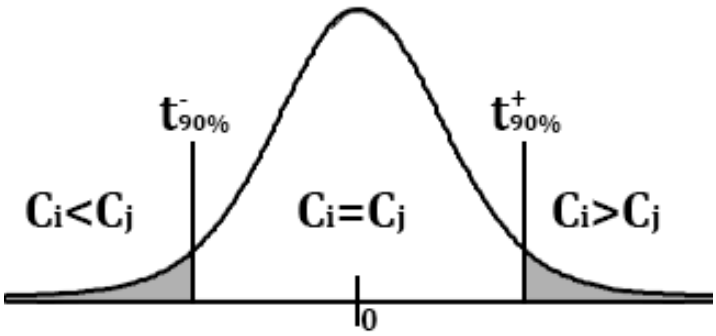
$$\sigma_x = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \mu_x)^2}{n-1}} \tag{4.2}$$

Os modelos então são ordenados para cada métrica, sendo a ordenação (o *rank*) determinado utilizando uma comparação de médias utilizando a distribuição t-student (equação

4.3) com 90% de confiança, como descrito na equação 4.4, sendo C_i o i-ésimo classificador e $t^-_{90\%}$ e $t^+_{90\%}$ os limiares opostos que contêm 90% uma distribuição t-student.

$$t(C_i, C_j) = \frac{\mu_i - \mu_j}{\sqrt{\frac{\sigma_i^2}{N_i} + \frac{\sigma_j^2}{N_j}}} \tag{4.3}$$

Figura 3 – Distribuição de t-student



Fonte: O Autor (2019)

$$rank_i = \begin{cases} 1 + \sum_{j \neq i} compare(C_i, C_j) & \text{, se o maior resultado é o melhor} \\ 1 + \sum_{j \neq i} compare(C_j, C_i) & \text{, se o menor resultado é o melhor} \end{cases} \tag{4.4}$$

Sendo:

$$compare(C_i, C_j) = \begin{cases} 1 & \text{, se } t(C_i, C_j) > t^+_{90\%} \\ 0.5 & \text{, se } t^-_{90\%} \leq t(C_i, C_j) \leq t^+_{90\%} \\ 0 & \text{, se } t(C_i, C_j) < t^-_{90\%} \end{cases} \tag{4.5}$$

A equação 4.4 simplifica o cálculo dos *ranks*, sem precisar ordenar os resultados. Ela obtêm valores similares ao de simplesmente ordenar, obtendo também a média das posições para classificadores que empatam (e.g: se o 2º e 3º classificadores possuírem resultados similares, o rank dos mesmos será 2.5).

Vale ressaltar que, quanto menor o *rank*, "melhor"é o algoritmo em relação aos demais, pelo menos para a métrica referente.

Após calcular os *ranks*, são calculadas as médias dos *ranks* dos modelos para cada métrica, para então ser utilizado o teste de Friedman[11] para identificar se existe algum modelo que obteve um resultado estatisticamente distinto. O calcula a estatística descrita pela equação 4.6, que possui (c-1) graus de liberdade. Sendo "b"o número de bases, "c"o número de classificadores testados.

$$\chi^2_F = \frac{12b}{c(c+1)} \left[\sum_{i=1}^{c} rank_i^2 - \frac{c(c+1)^2}{4} \right] \tag{4.6}$$

Contudo, Iman e Daveport[12] demonstraram que a métrica χ_F^2 é muito conservadora, e propuseram uma variação da estatística:

$$F_F = \frac{(b-1)\chi_F^2}{b(c-1)-\chi_F^2} \tag{4.7}$$

Que é baseada na distribuição F de Fisher-Snedecor, com (c-1) e (c-1)(b-1) graus de liberdade.

Como o teste de Friedman só detecta se existe um modelo com resultado distindo, mas não identifica qual o modelo, é necessário executar um pós-teste. Para esse trabalho foi utilizado o teste de Nemenyi[13], que lista os *ranks* médios e compara as distâncias dos modelos com a Distância Crítica (calculada com a equação 4.8) para agrupar os modelos que obtiveram resultados estatisticamente semelhantes.

$$CD = q_\alpha \sqrt{\frac{c(c+1)}{6b}} \tag{4.8}$$

Sendo q_α o coeficiente de Nemenyi, que é definido pela tabela a seguir.

Tabela 5 – coeficiente do teste bicaudal de Nemenyi

Num. de classificadores	2	3	4	5	6	7	8	9	10
$q_{0.05}$	1.960	2.343	2.569	2.728	2.850	2.949	3.031	3.102	3.164
$q_{0.10}$	1.645	2.052	2.291	2.459	2.589	2.693	2.780	2.855	2.920

Fonte: Valores obtidos de [14]

Considerando $q_{0.05}$, c = 4 e b = 11, temos a distância crítica:

$$CD = q_{0.05}\sqrt{\frac{4(4+1)}{6 \cdot 11}} = 2.569\sqrt{\frac{10}{33}} = 1,414 \tag{4.9}$$

4.3 MÉTRICAS

Para validar os modelos, foram utilizadas tanto métricas para a partição difusa[1], quanto métricas para partição *crips*[2].

4.3.1 Partição *Crisp*

Para calcular as métricas *crisp*, primeiro é necessário encontrar a partição *crips* dos elementos, utilizando as partições difusas dos mesmos. A partição *crips* p_i do i-ésimo elemento é o *cluster* no qual o elemento possui a maior pertinência.

[1] As métricas difusas utilizadas foram: Rand Frigui, Rand Hullermeier, Distância Mínima entre Centroides e Informação Mútua Normalizada

[2] As métricas *crisp* utilizadas foram: Erro Geral e F-Measure

4.3.1.1 Erro Geral

O Erro Geral [15] é a métrica que contabiliza a fração de elementos que foram mapeados erroneamente. Como os métodos são não supervisionados, não podemos considerar que o índices dos *clusters* e das classes possuem alguma relação, além de ser possível utilizar um número de *clusters* diferente do número de classes. Dessa forma, foi considerado que um elemento foi corretamente mapeado caso a sua classe seja a classe com mais elementos no seu *cluster*).

$$e_i = \begin{cases} 0 & \text{, se classe(i) = classe predominante de } p_i \\ 1 & \text{, caso contrário} \end{cases} \tag{4.10}$$

$$erro = \frac{\sum_{i=1}^{N} e_i}{N}$$

o erro será 0 caso todos os elementos foram corretamente mapeados e, teoricamente, 1 caso todos os elementos tenham sido mapeados erroneamente. Contudo, não é possível o erro ser exatamente 1, pois os elementos da classe predominante estarão mapeados corretamente. Dessa forma, o pior caso ocorre quando todos os elementos se encontram num único cluster, resultando no erro:

$$max_{error} = 1 - max_{c \in classes}|c| \tag{4.11}$$

4.3.1.2 F-Measure

F-Measure [16] é uma métrica que leva em consideração a precisão e revocação, sendo a média harmônica[3] dessas duas métricas (equação 4.12).

$$F - Measure = \left[\frac{precisão^{-1} + revocação^{-1}}{2}\right]^{-1} = 2\frac{precisão \cdot revocação}{precisão + revocação} \tag{4.12}$$

Sendo a precisão a taxa de elementos selecionados que são da classe positiva (equação 4.13) e a revocação a taxa da classe positiva que foi selecionada (equação 4.14).

$$precisão = \frac{Verdadeiros\ Positivos}{(Verdadeiros\ Positivos) + (Falsos\ Positivos)} \tag{4.13}$$

$$revocação = \frac{Verdadeiros\ Positivos}{(Verdadeiros\ Positivos) + (Falsos\ Negativos)} \tag{4.14}$$

Que utilizam informações obtidas da matriz de confusão[4] da solução.

Contudo, como as bases testadas possuem mais de 2 classes, foi necessário utilizar uma versão modificada.

$$F - measure = \frac{1}{N} \sum_{C\ \in\ Classes} |C| \left[max_{K\ \in\ Clusters} \left(\frac{2|C \cap K|}{|C| + |K|} \right) \right] \tag{4.15}$$

[3] a média harmônica é definida como o inverso da média dos inversos

[4] A matriz de confusão contabiliza os elementos de cada classe que foram mapeados para cada *cluster*, tendo os valores corretamente mapeados na diagonal principal.

Que é o equivalente a uma média ponderada das medidas F-measure para cada uma das classes.

4.3.1.3 Informação Mútua Normalizada (NMI)

A informação mútua[17] é uma medida da relação entre duas variáveis aleatórias (nesse caso, as Classes e *Clusters* dos elementos), representando a quantidade de informação obtida para uma variável ao se examinar a outra (ou seja, a informação da classe ao se obter o *cluster*).

$$MI = \sum_{c \in C} \sum_{k \in K} p(c,k) log \left(\frac{p(c,k)}{p(c)p(k)} \right) \tag{4.16}$$

sendo p(c) a probabilidade de um elemento pertencer à classe c, p(k) a probabilidade de um elemento pertencer ao *cluster* k, e p(c,k) a probabilidade de um elemento pertencer a classe c e ao *cluster* k, simultâneamente.

Para obter os valores num intervalo entre 0 e 1, foi utilizada a Informação Mútua Normalizada, sendo esta descrita como:

$$NMI = \frac{2MI}{S_C + S_K} \tag{4.17}$$

Sendo S_C e S_K as entropias das classes e dos *clusters*, que podem ser calculadas com a equação a seguir (substituindo A por C e K, respectivamente).

$$S_A = -\sum_{a \in A} p(a) log\left(p(a)\right) \tag{4.18}$$

4.3.1.4 Índice Rand

o índice Rand [18] foi nomeado em homenagem ao seu criador (William M. Rand) e é utilizado quando se deseja comparar duas classificações. Podendo ser usado para comparar os *clusters* obtidos com as classes originais dos elementos.

$$c(C,K) = \frac{\sum_{i=2}^{N} \sum_{j=1}^{i} \gamma_{ij}}{N(N-1)/2} \tag{4.19}$$

Sendo:

$$\gamma_{ij} = \begin{cases} 1, & \text{se } x_i \text{ e } x_j \text{ são da mesma classe e foram mapeados no mesmo } \textit{clusters} \\ 1, & \text{se } x_i \text{ e } x_j \text{ são de classes diferentes e foram mapeados em } \textit{clusters} \text{ diferentes} \\ 0, & \text{caso contrário} \end{cases} \tag{4.20}$$

Esse índice foi pensado para a partição *crisp*. Porém surgiram variações do mesmo para a partição difusa (como o Rand Frigui e Rand Hullermeier). Dessa forma, esse índice não foi utilizado por esse trabalho, sendo utilizado as duas variações citadas no seu lugar (mais detalhes adiante, nas seções 4.3.2.1 e 4.3.2.2, respectivamente).

4.3.2 Partição Difusa

4.3.2.1 Rand Frigui

Uma variação difusa, proposta por Frigui et al. [6], da métrica Rand.

$$Q_{Rand}(\psi^{(1)}, \psi^{(2)}) = \frac{N_{SS} + N_{DD}}{N_{SS} + N_{SD} + N_{DS} + N_{DD}} \tag{4.21}$$

Considerando:

$$\begin{aligned} N_{SS}(\psi^{(1)}, \psi^{(2)}) &= \sum_{i=2}^{N} \sum_{j=1}^{i-1} \psi_{ij}^{(1)} \psi_{ij}^{(2)} \\ N_{SD}(\psi^{(1)}, \psi^{(2)}) &= \sum_{i=2}^{N} \sum_{j=1}^{i-1} \psi_{ij}^{(1)} (1 - \psi_{ij}^{(2)}) \\ N_{DS}(\psi^{(1)}, \psi^{(2)}) &= \sum_{i=2}^{N} \sum_{j=1}^{i-1} (1 - \psi_{ij}^{(1)}) \psi_{ij}^{(2)} \\ N_{DD}(\psi^{(1)}, \psi^{(2)}) &= \sum_{i=2}^{N} \sum_{j=1}^{i-1} (1 - \psi_{ij}^{(1)}) (1 - \psi_{ij}^{(2)}) \end{aligned} \tag{4.22}$$

Sendo:

$$\psi_{ij} = \sum_{k=1}^{K} u_{ik} u_{jk} \tag{4.23}$$

com $\psi^{(1)}$ calculado com as pertinências obtidas pelo algoritmo, e $\psi^{(2)}$ com as pertinências verdadeiras dos elementos. Lembrando que as pertinências verdadeiras de um elemento é 1 para a sua classe e 0 para as demais classes.

4.3.2.2 Rand Hullermeier

Uma variação difusa, proposta por Hullermeier et al. [19], da métrica Rand.

$$R_E(P, Q) = 1 - \frac{\sum_{i=2}^{N} \sum_{j=1}^{i-1} |E_P(i,j) - E_Q(i,j)|}{N(N-1)/2} \tag{4.24}$$

Sendo:

$$E(i,j) = 1 - \frac{\sum_{c=1}^{K} |u_{ic} - u_{jc}|}{2} \tag{4.25}$$

com E_P calculado com as pertinências obtidas pelo algoritmo, e E_Q com as pertinências verdadeiras dos elementos. Lembrando que as pertinências verdadeiras de um elemento é 1 para a sua classe e 0 para as demais classes.

4.3.2.3 Distância Mínima entre Centroides

Métrica que representa a menor distância entre os centroides de dois clusters distintos (a distância entre dois centroides pode ser calculada com a equação 4.26). Maximizar a distância mínima entre os centroides indica numa melhor separação.

$$Dist(c,k) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{p=1}^{P} v_{ci}v_{kj}\delta_p D_p(i,j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{p=1}^{P} v_{ci}v_{kj}\delta_p} \tag{4.26}$$

Contudo, como as representações das classes respeitam a restrição de soma (equação 4.27) e a ponderação das visões não depende dos elementos, então podemos simplificar a equação, ficando com a equação 4.28.

$$\sum_{i=1}^{N} v_{ic} = 1, \forall c = 1, 2, ...K \tag{4.27}$$

$$Dist(c,k) = \frac{\sum_{p=1}^{P}\delta_p\left[\sum_{i=1}^{N} v_{ci}\sum_{j=1}^{N} v_{kj}D_p(i,j)\right]}{\sum_{p=1}^{P}\delta_p} \tag{4.28}$$

Como podemos perceber, a distância entre os centroides é a média ponderada da distância entre os centroides para para cada visão.

4.3.2.4 Coeficiente de Partição Modificado

O coeficiente de partição indica a separação das partições difusas dos elementos [20] (apud [21]) e é descrito pela equação 4.29. Sendo 1 quando $u_{ic} = 1$ para uma das classes e $u_{ic} = 0$ para as demais, e 1/K quando as partições são iguais.

$$PC = \frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^2 \tag{4.29}$$

Como o valor é dependente do número de *clusters*, Rajesh N. Dave propôs uma modificação [21] para padronizar o intervalo do coeficiente de [1/K, 1] para [0, 1] (utilizando a equação 4.30)

$$MPC = 1 - \frac{K}{K-1}(1 - PC) \tag{4.30}$$

Essa padronização é uma transformação linear entre os intervalos.

Nesse Trabalho, o Coeficiente de Partição Modificada foi utilizado na busca de parâmetros e, portanto, não foi comparado entre os modelos testados.

4.4 BASES DE DADOS

Para testar os métodos, foram utilizadas 11 bases de dados[5] (descritas na tabela 6).

[5] Por limitações computacionais e temporais, não foi possível testar bases maiores, ficando esses testes como trabalhos futuros.

Tabela 6 – Bases de dados

Base	Elementos	Classes	Visões	Características
Glass	214	6	9	9
Image Segm.	2310	6	2	19
Iris	150	3	4	4
mFeat	2000	10	6	649
Phoneme	2000	5	3	447
Seeds	210	3	7	7
Wine	178	3	13	13
Corel 1	400	4	7	338
Corel 2	400	4	7	338
Reuters	1200	6	5	500
ALOI-Cars	1413	13	4	106

Fonte: Informações obtidas de [22], [23], [24], [25], [26], [27], [28], [29], [30] e [31]

As bases testadas possuem os elementos descritos por características, contudo, os modelos testados utilizam dados relacionas. Considerando isso, os dados precisam ser convertidos. Para fazer essa conversão primeiro é necessário separar as características nas visões (Felizmente, os autores dessas bases informaram, nos pacotes das bases, quais são as características referentes a cada visão). Com as visões separadas, foram calculadas as distâncias euclidianas entre os elementos em relação a cada visão, gerando P matrizes de dissimilaridade NxN (equação 4.31).

$$D'_p(i,j) = \sqrt{\sum_{v \in V_p} (v_i - v_j)^2} \tag{4.31}$$

Sendo V_p a p-ésima visão e v_i o valor da variável v para o i-ésimo elemento. Vale resaltar que $D'_p(i,j) \leq 0$ por se tratar de uma distância. Essas matrizes são então padronizadas utilizando a equação 4.32 [32], isso serve para evitar valores muito grandes (que podem causar overflows nos algoritmos) ou muito pequenos (que podem causar underflows nos algoritmos).

$$D_p(i,j) = \frac{D'_p(i,j)}{W_p} \tag{4.32}$$

Com o fator de padronização definido por:

$$W_p = min_{1 \leq i \leq N} \left(\sum_{j=1}^{N} D'_p(i,j) \right) \tag{4.33}$$

Como a padronização se baseia no mínimo, ao invés de uma média ou máximo, então ela não é influenciada por *outliers*[6], de forma que, normalmente, o ponto escolhido pelo mínimo se encontra próximo ao centro da base. De forma que a soma das distâncias de um elemento para os demais será maior ou igual a 1.

4.4.1 Glass Identification

A base *Glass Identification*[22] foi criada com o propósito forense de identificar a origem de cacos de vidro, o que ajuda a reconstruir os acontecimentos de um crime.

As classes são: Janelas com vidro float, janelas com vidro não float, janelas de veículo, recipientes, copos e faróis. As características são o índice de refração além das concentrações de sódio, magnésio, alumínio, silício, potássio, cálcio, bário e ferro e foram assumidas como visões neste trabalho.

4.4.2 Image Segmentation

A base *Image Segmentation*[23] foi formada pelo Grupo de Visão da Universidade de Massachusetts, selecionando 2310 pixels de imagens aleatórias de exteriores, classificando-os manualmente com uma de 7 classes possíveis (tijolos, céu, folhagem, cimento, janelas, caminhos e grama). A descrição dos pontos foi feita com duas visões, uma que se refere às cores (com 10 características) e outra se referindo às formas (com 9 características).

4.4.3 Iris

A base *Iris*[24] é uma das bases de classificação mais conhecidas, sendo normalmente testada primeiro por ser considerada fácil e pequena, o que permite identificar problemas de implementação ou conceito caso os resultados obtidos sejam ruins ou caso a convergência não seja atingida.

As classes das bases são 3 espécies do gênero de flores Íris: *Iris Setosa*, *Iris Versicolour* e *Iris Virginica*. Sendo as características a largura e o comprimento das pétalas e das Sépalas e, assim como para a base *Glass*, foram assumidas como visões.

[6] um elemento é considerado outlier se ele for muito distante dos demais, não pertencendo a uma das classes

Figura 4 – Exemplos das espécies de Íris

Fonte: Imagens obtidas da Wikipédia

4.4.4 Multiple Features

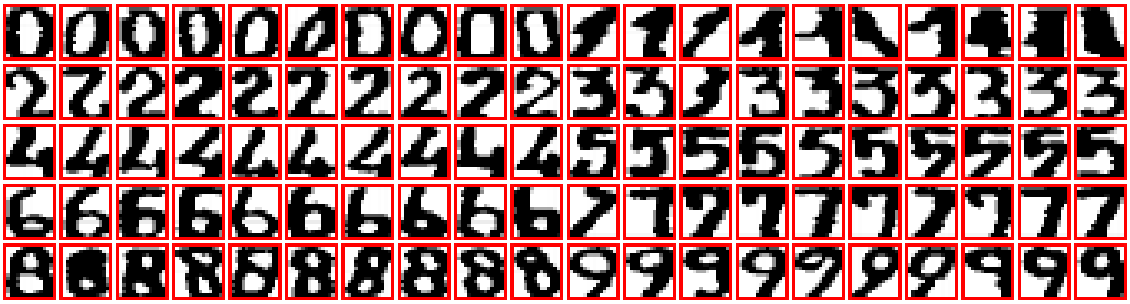
A base *Multiple Features* (mFeat)[25] é uma base de dígitos (0-9) manuscritos, com 200 entradas para cada valor de dígito, descritos por 7 visões, cada uma se referindo a uma categoria de características (ver tabela 7)

Tabela 7 – Visões da base *Multiple Features*

Visão	Descrição	Variaveis
mfeat-fou	Coeficientes de Fourier da forma	76
mfeat-fac	Correlação do perfil	216
mfeat-kar	Coeficientes de Karhunen-Loève	64
mfeat-pix	Soma dos pixels de uma janela 2x3	240
mfeat-zer	Momentos de Zernike	47
mfeat-mor	Características Morfológicas	6

Fonte: Informações obtidas de [25]

Figura 5 – Visualização da mfeat-pix da base *Multiple Features*



Fonte: O Autor (2019)

4.4.5 Phoneme

A base *Phoneme*[26] possui uma visão que é uma série temporal (com 150 características) referente a pronúncia de um de 5 fonemas da língua inglesa ("aa", "ao", "dcl", "iy"e "sh"). Para expandir o número de visões, foram criadas 2 visões novas, que representam a velocidade e aceleração que essa série temporal possui (sendo a primeira e segunda derivadas discretas da sequência, possuindo 149 e 148 características, respectivamente).

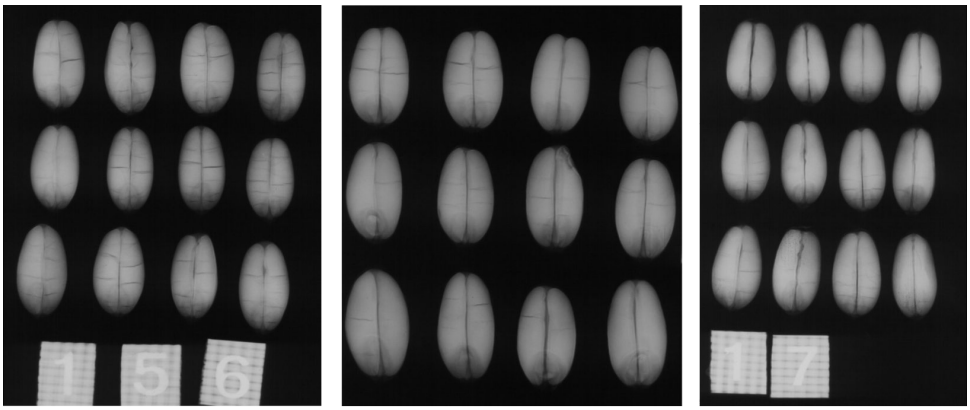
$$v'[t] = \frac{v[t+1] - v[t]}{\textit{período do sinal}} \tag{4.34}$$

Como a base será padronizada, o período do sinal pode ser ignorado, já que é uma constante que divide todos os valores da visão. Dessa forma, as derivadas foram calculadas apenas como $v'[t] = v[t+1] - v[t]$.

4.4.6 Seeds

A base *Seeds*[27] descreve sementes de 3 variedades de trigo, utilizando propriedades geométricas obtidas a partir de imagens de raio-X [7] de um pacote de grãos. A base descreve 70 grãos de cada variedade, utilizando 7 características contínuas (área, perímetro, o quão compacto é o grão, comprimento, largura, coeficiente de assimetria e comprimento do sulco) que foram assumidas como visões para este trabalho.

Figura 6 – Exemplos da base Seeds



Fonte: Imagens obtidas de [33]

4.4.7 Wine

A base *Wine*[28] utiliza uma análise química para determinar a origem de vinhos da mesma região da Itália, mas de 3 cultivares distintos. As características representam 13 medidas do vinho (álcool, ácido málico, cinzas, alcalinidade das cinzas, magnésio, fenóis totais, flavonoides, fenóis não flavonoides, pro-antocianinas, intensidade da cor, matiz da

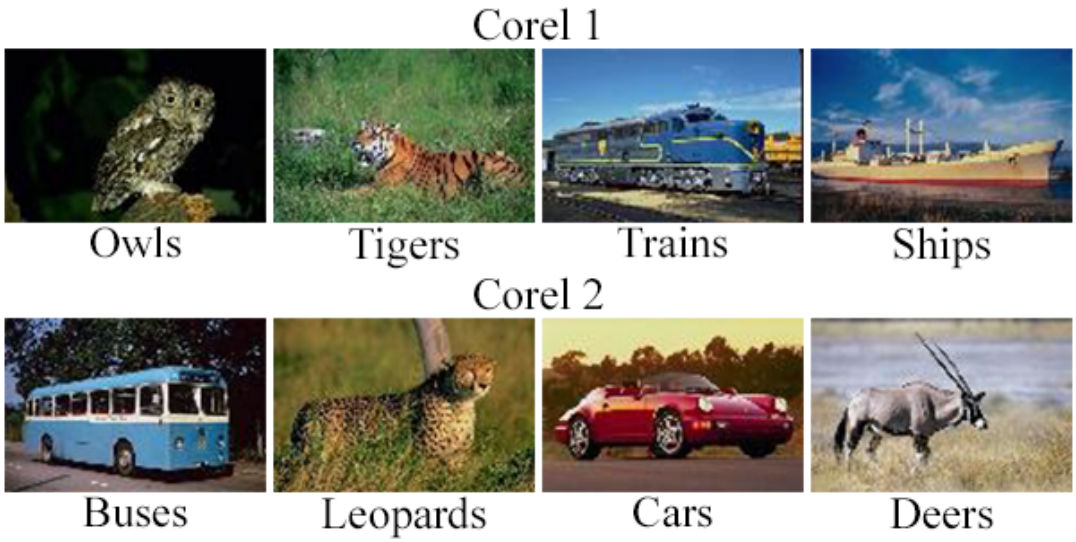
[7] as imagens de raio-X foram obitidas com placas da Kodak com 13x18 centímetros

cor, OD280/OD315 de vinhos diluídos e prolina) que foram assumidas como visões para este trabalho.

4.4.8 Corel Images

Corel 1 e Corel 2 são subconjuntos da base de imagens *Corel*[29], que possui 34 classes com 100 elementos cada. Infelizmente eu não consegui acessar a base original, contudo Rodrigo Cavalcanti de Araújo disponibilizou a base separada que ele utilizou na sua dissertação de mestrado [34]. Pela base possuir muitas classes, Rodrigo a separou em bases com 4 classes, testando duas dessas bases: a "Corel 1"com as classes *owls*, *tigers*, *trains* e *ships*, e a "Corel 2"com as classes *buses*, *leopards*, *cars* e*deers*. Para uma melhor visualização da base, Rodrigo selecionou uma imagem para cada uma das classes selecionadas (Figura 7).

Figura 7 – Exemplos da base Corel



Fonte: Imagens obtidas de [34]

A base possui 7 visões, referentes a diferentes características da imagem, incluindo cor e textura.

Tabela 8 – Visões da base Corel

Visão	Características
Histograma HSV	64
Momento LUV	9
Coerência HSV	128
Aspereza da Textura	10
Direção da Textura	8
Wavelet da Textura	104
Textura MRSAR	15

Fonte: Informações obtidas de [29]

4.4.9 Reuters RCV1/RCV2 Multilingual

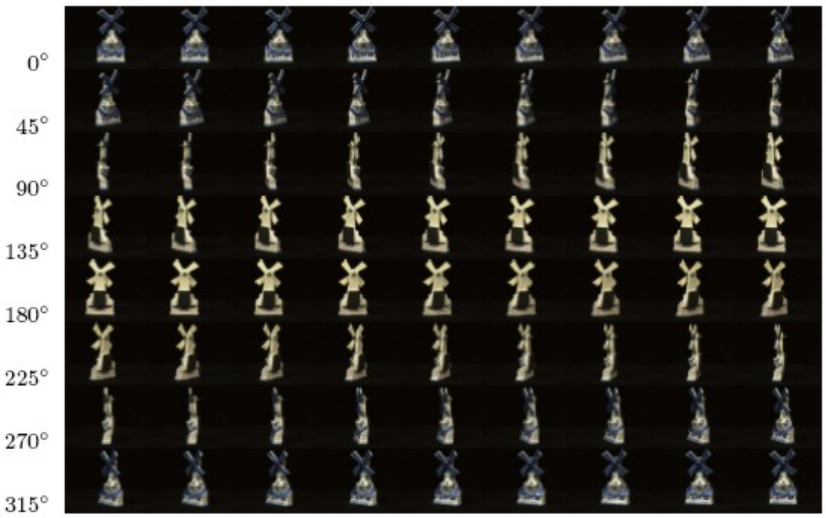
A base *Reuters*[30] possui 111740 documentos escritos em 5 idiomas (inglês, alemão, espanhol, francês e italiano) e traduzidos para os demais idiomas. Os documentos são separados em 6 classes (C15, CCAT, E21, ECAT, GCAT e M11).

Como a base possui muitos elementos e características, Rodrigo Cavalcanti de Araújo [34] selecionou 1200 documentos aleatórios, e reduziu o número de características para 100 para cada idioma, utilizando um método proposto por Abhishek Kumar e Hal Daumé III[35]. Este trabalho, assim como o de Rodrigo, assume cada idioma como sendo uma visão.

4.4.10 Amsterdam Library of Object Images (ALOI)

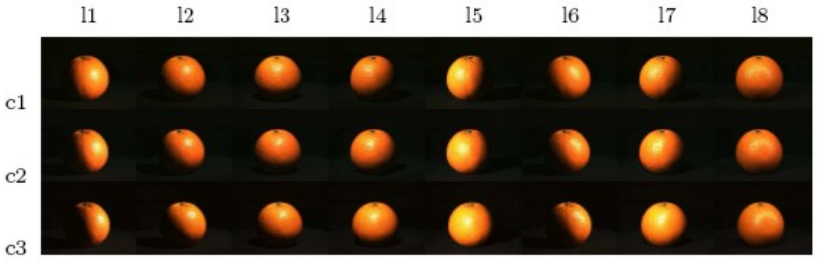
A base *Amsterdam Library of Object Images* (ALOI)[31] é uma base com fotos de objetos pequenos, aplicando variações sistemáticas no ângulo da câmera e da iluminação, além da cor da iluminação, resultando em centenas de fotos para cada objeto. Os autores da base publicaram imagens demonstrando essas variações.

Figura 8 – Ângulos utilizados pela base ALOI



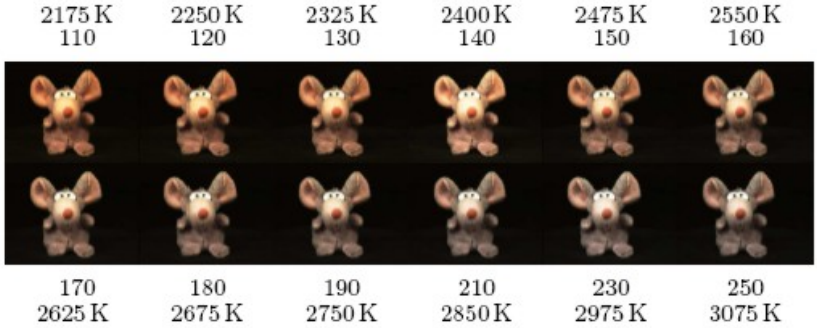
Fonte: Imagens obtidas de [31]

Figura 9 – Ângulos de iluminação utilizados pela base ALOI



Fonte: Imagens obtidas de [31]

Figura 10 – Cores das lâmpadas utilizadas pela base ALOI



Fonte: Imagens obtidas de [31]

A base ALOI possui 4 visões, referentes a diferentes características extraídas dessas imagens (como pode ser observado pela tabela 9).

Tabela 9 – Visões da base ALOI

Visão	Características
Histograma da Cor	8
Momentos da Cor	77
Cor no modelo HSB	8
Características de Haralick da Textura	13

Fonte: Informações obtidas de [31]

Rodrigo Cavalcanti de Araújo selecionou as imagens de 13 veículos de brinquedo da base, considerando cada um desses veículos como uma classe distinta [34], alguns desses veículos podem ser observados na imagem 11.

Figura 11 – Exemplos da base ALOI

Fonte: Imagens obtidas de [31]

5 AVALIAÇÃO EXPERIMENTAL

Após calcular a média do Coeficiente de Partição Modificado para cada uma das possíveis combinações de parâmetros, foram selecionados os seguintes parâmetros de cada modelo para cada uma das bases:

Tabela 10 – Parâmetros Selecionados

Datasets	MFMMdd	FW4M	$CARD_R$	MFCMdd-RWG-P
Glass	m = 1.1 n = 1.1	m = 1.1 n = 1.1 s = 1.1	m = 1.3 s = 1.5	m = 1.1 q = 2
Image Segm.	m = 1.1 n = 1.3	m = 1.1 n = 1.1 s = 1.05	m = 1.1 s = 1.5	m = 1.1 q = 2
Iris	m = 1.1 n = 1.1	m = 1.1 n = 1.1 s = 1.5	m = 1.1 s = 1.5	m = 1.1 q = 5
mFeat	m = 1.1 n = 1.1	m = 1.1 n = 1.1 s = 1.1	m = 1.1 s = 1.05	m = 1.1 q = 7
Phonema	m = 1.05 n = 1.05	m = 1.1 n = 1.1 s = 1.1	m = 1.1 s = 1.05	m = 1.1 q = 5
Seeds	m = 1.1 n = 1.1	m = 1.1 n = 1.1 s = 1.1	m = 1.1 s = 1.5	m = 1.1 q = 5
Wine	m = 1.1 n = 1.1	m = 1.1 n = 1.1 s = 1.3	m = 1.1 s = 1.3	m = 1.1 q = 7
Corel 1	m = 1.1 n = 1.1	m = 1.1 n = 1.05 s = 1.1	m = 1.1 s = 1.3	m = 1.1 q = 2
Corel 2	m = 1.1 n = 1.1	m = 1.1 n = 1.05 s = 1.1	m = 1.1 s = 1.1	m = 1.1 q = 5
Reuters	m = 1.1 n = 1.1	m = 1.1 n = 1.1 s = 1.5	m = 1.1 s = 1.5	m = 1.1 q = 2
ALOI	m = 1.1 n = 1.1	m = 1.1 n = 1.3 s = 1.1	m = 1.1 s = 1.3	m = 1.1 q = 5

Fonte: O Autor (2019)

5.1 MÉDIA DOS RESULTADOS

Foram calculados média e desvio padrão de 100 execuções de cada modelo para cada base, com o objetivo de validar a robustez dos modelos. Os resultados dos experimentos para métricas da partição difusa e da partição *crisp* estão descritos nas tabelas seguintes, com a organização:

média *rank*

(desvio padrão)

Esses resultados serão comparados nas seções 5.3, 5.4 e 5.5.

5.1.1 Métricas da Partição Difusa

Tabela 11 – Rand Frigui - Resultado Médio

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.713016**	1	0.672337	3	0.482732	4	0.680145	2
	$(8.229440 \cdot 10^{-8})$		$(1.157600 \cdot 10^{-2})$		$(2.251680 \cdot 10^{-5})$		$(2.693990 \cdot 10^{-2})$	
Image Segm.	**0.850405**	1	0.838008	2	0.823075	3.5	0.817357	3.5
	$(6.234620 \cdot 10^{-3})$		$(2.202590 \cdot 10^{-2})$		$(3.334500 \cdot 10^{-2})$		$(3.312370 \cdot 10^{-2})$	
Iris	0.926521	3	**0.946013**	1.5	**0.942251**	1.5	0.877568	4
	$(4.211550 \cdot 10^{-2})$		$(2.101130 \cdot 10^{-2})$		$(2.836660 \cdot 10^{-2})$		$(9.258800 \cdot 10^{-2})$	
mFeat	**0.907604**	1.5	0.821114	4	0.858236	3	**0.908235**	1.5
	$(5.364950 \cdot 10^{-3})$		$(5.276320 \cdot 10^{-3})$		$(1.596400 \cdot 10^{-3})$		$(9.939330 \cdot 10^{-3})$	
Phonema	0.820001	3	0.858130	2	**0.893794**	1	0.777517	4
	$(6.155940 \cdot 10^{-3})$		$(2.395430 \cdot 10^{-2})$		$(2.375900 \cdot 10^{-2})$		$(1.084090 \cdot 10^{-2})$	
Seeds	**0.852132**	1	0.839794	4	0.842159	3	0.847160	2
	$(7.884950 \cdot 10^{-8})$		$(4.097140 \cdot 10^{-7})$		$(9.797120 \cdot 10^{-3})$		$(5.340890 \cdot 10^{-4})$	
Wine	**0.904725**	1.5	0.800693	3	0.786115	4	**0.906918**	1.5
	$(6.409230 \cdot 10^{-8})$		$(4.222550 \cdot 10^{-6})$		$(8.094320 \cdot 10^{-2})$		$(2.290900 \cdot 10^{-2})$	
Corel 1	**0.770629**	1.5	0.769555	2.5	0.761928	4	0.771224	2
	(0.000000)		$(5.034580 \cdot 10^{-3})$		$(6.330790 \cdot 10^{-3})$		$(1.631450 \cdot 10^{-2})$	
Corel 2	0.714218	4	0.733918	3	0.747397	2	**0.760080**	1
	$(8.940700 \cdot 10^{-8})$		$(3.454740 \cdot 10^{-2})$		$(1.131470 \cdot 10^{-2})$		$(1.039870 \cdot 10^{-2})$	
Reuters	0.722686	3	0.722686	3	0.722686	3	**0.727838**	1
	$(2.107340 \cdot 10^{-8})$		(0.000000)		$(1.053670 \cdot 10^{-8})$		$(2.348080 \cdot 10^{-3})$	
ALOI	**0.930055**	1	0.875642	2	0.855127	3.5	0.858453	3.5
	$(4.859560 \cdot 10^{-3})$		$(7.336720 \cdot 10^{-3})$		$(1.503050 \cdot 10^{-2})$		$(4.199710 \cdot 10^{-2})$	

Fonte: O Autor (2019)

Tabela 12 – Rand Hullermeier - Resultado Médio

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.692723**	1	0.671488	2.5	0.482734	4	0.673169	2.5
	$(1.649260 \cdot 10^{-7})$		$(1.123550 \cdot 10^{-2})$		$(1.808940 \cdot 10^{-5})$		$(2.462640 \cdot 10^{-2})$	
Image Segm.	**0.847628**	1	0.837955	2	0.822980	3.5	0.815255	3.5
	$(5.922580 \cdot 10^{-3})$		$(2.175480 \cdot 10^{-2})$		$(3.337320 \cdot 10^{-2})$		$(3.376200 \cdot 10^{-2})$	
Iris	0.926821	3	**0.945941**	1.5	**0.942245**	1.5	0.877689	4
	$(4.212350 \cdot 10^{-2})$		$(2.099500 \cdot 10^{-2})$		$(2.836560 \cdot 10^{-2})$		$(9.256440 \cdot 10^{-2})$	
mFeat	0.894247	2	0.114840	4	0.858156	3	**0.899708**	1
	$(5.417480 \cdot 10^{-3})$		(0.106144)		$(1.592670 \cdot 10^{-3})$		$(1.164820 \cdot 10^{-2})$	
Phonema	0.780696	3	0.846640	2	**0.892369**	1	0.690653	4
	$(4.223240 \cdot 10^{-3})$		$(2.940870 \cdot 10^{-2})$		$(2.415660 \cdot 10^{-2})$		$(2.299350 \cdot 10^{-2})$	
Seeds	**0.851840**	1	0.839739	4	0.842063	3	0.846943	2
	$(7.300050 \cdot 10^{-8})$		$(3.971940 \cdot 10^{-7})$		$(9.790880 \cdot 10^{-3})$		$(5.051300 \cdot 10^{-4})$	
Wine	**0.908419**	1.5	0.800274	3	0.786233	4	**0.910140**	1.5
	$(4.942160 \cdot 10^{-8})$		$(4.243990 \cdot 10^{-6})$		$(8.106630 \cdot 10^{-2})$		$(2.289450 \cdot 10^{-2})$	
Corel 1	0.733476	4	**0.761668**	1	0.760344	2.5	0.757107	2.5
	(0.000000)		$(4.699210 \cdot 10^{-3})$		$(6.321870 \cdot 10^{-3})$		$(2.096690 \cdot 10^{-2})$	
Corel 2	0.620925	4	0.730336	2.5	**0.745074**	1	0.724537	2.5
	$(1.928530 \cdot 10^{-7})$		$(3.497080 \cdot 10^{-2})$		$(1.132370 \cdot 10^{-2})$		$(1.074200 \cdot 10^{-2})$	
Reuters	0.165972	3.5	0.165972	3.5	0.165982	2	**0.384955**	1
	$(7.684400 \cdot 10^{-8})$		$(1.131160 \cdot 10^{-7})$		$(1.563440 \cdot 10^{-6})$		$(3.175790 \cdot 10^{-2})$	
ALOI	**0.927517**	1	0.875418	2	0.854968	3.5	0.855407	3.5
	$(4.843800 \cdot 10^{-3})$		$(7.355530 \cdot 10^{-3})$		$(1.505710 \cdot 10^{-2})$		$(4.189200 \cdot 10^{-2})$	

Fonte: O Autor (2019)

Tabela 13 – Distância Mínima entre Centroides - Resultado Médio

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	0.001385	3	0.000875	4	**0.002226**	1.5	**0.002087**	1.5
	$(5.294980 \cdot 10^{-10})$		$(3.073690 \cdot 10^{-4})$		$(1.907250 \cdot 10^{-4})$		$(1.131480 \cdot 10^{-3})$	
Image Segm.	**0.000271**	1	0.000205	4	0.000250	2	0.000215	3
	$(3.016760 \cdot 10^{-5})$		$(3.568600 \cdot 10^{-5})$		$(2.168200 \cdot 10^{-5})$		$(3.810290 \cdot 10^{-5})$	
Iris	0.006186	2.5	0.006192	2.5	**0.006882**	1	0.005490	4
	$(7.416530 \cdot 10^{-4})$		$(4.585530 \cdot 10^{-4})$		$(2.751370 \cdot 10^{-4})$		$(1.695470 \cdot 10^{-3})$	
mFeat	0.000315	2	**0.000555**	1	0.000195	4	0.000218	3
	$(3.598820 \cdot 10^{-5})$		$(5.556770 \cdot 10^{-5})$		$(1.327960 \cdot 10^{-5})$		$(4.053510 \cdot 10^{-5})$	
Phonema	**0.000490**	1	0.000383	4	0.000477	2	0.000405	3
	$(7.642840 \cdot 10^{-6})$		$(2.307700 \cdot 10^{-5})$		$(1.155090 \cdot 10^{-5})$		$(1.200000 \cdot 10^{-5})$	
Seeds	0.005200	3	0.005529	2	**0.005600**	1	0.004856	4
	$(8.808420 \cdot 10^{-10})$		$(5.816110 \cdot 10^{-9})$		$(2.246280 \cdot 10^{-4})$		$(1.681810 \cdot 10^{-4})$	
Wine	**0.007346**	1	0.007115	2	0.006750	4	0.006991	3
	$(1.164150 \cdot 10^{-10})$		$(3.287140 \cdot 10^{-7})$		$(7.947120 \cdot 10^{-4})$		$(2.693410 \cdot 10^{-4})$	
Corel 1	**0.002088**	1.5	0.001576	4	**0.002084**	1.5	0.001824	3
	(0.000000)		$(7.942010 \cdot 10^{-5})$		$(4.226670 \cdot 10^{-5})$		$(4.469460 \cdot 10^{-4})$	
Corel 2	0.002438	2	0.001753	4	**0.002632**	1	0.001942	3
	$(6.507810 \cdot 10^{-11})$		$(1.239930 \cdot 10^{-4})$		$(7.602300 \cdot 10^{-5})$		$(2.835680 \cdot 10^{-4})$	
Reuters	0.000905	2.5	0.000905	2.5	**0.000918**	1	0.000614	4
	$(1.455190 \cdot 10^{-11})$		$(1.028980 \cdot 10^{-11})$		$(1.584090 \cdot 10^{-10})$		$(2.038310 \cdot 10^{-5})$	
ALOI	0.000204	2	0.000057	4	0.000097	3	**0.000234**	1
	$(3.338100 \cdot 10^{-5})$		$(9.727270 \cdot 10^{-6})$		$(2.226130 \cdot 10^{-5})$		$(5.736270 \cdot 10^{-5})$	

Fonte: O Autor (2019)

5.1.2 Métricas da Partição *Crisp*

Tabela 14 – F-measure - Resultado Médio

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.549631**	1	0.419852	3	0.346121	4	0.499767	2
	$(3.494630 \cdot 10^{-8})$		$(2.931260 \cdot 10^{-2})$		$(1.976290 \cdot 10^{-3})$		$(4.725910 \cdot 10^{-2})$	
Image Segm.	**0.647941**	1	0.604803	2	0.578919	3.5	0.566572	3.5
	$(2.272640 \cdot 10^{-2})$		$(4.358810 \cdot 10^{-2})$		$(5.893510 \cdot 10^{-2})$		$(7.163450 \cdot 10^{-2})$	
Iris	0.943143	2.5	**0.957750**	1	0.948861	2.5	0.881813	4
	$(4.971540 \cdot 10^{-2})$		$(2.222850 \cdot 10^{-2})$		$(3.130540 \cdot 10^{-2})$		(0.112752)	
mFeat	**0.767144**	1	0.297269	4	0.447069	3	0.725101	2
	$(2.814860 \cdot 10^{-2})$		$(2.291800 \cdot 10^{-2})$		$(2.355650 \cdot 10^{-3})$		$(5.302410 \cdot 10^{-2})$	
Phonema	**0.836313**	2	**0.834529**	2	**0.843888**	2	0.788788	4
	$(3.261380 \cdot 10^{-2})$		$(7.006470 \cdot 10^{-2})$		$(4.839430 \cdot 10^{-2})$		$(5.301710 \cdot 10^{-2})$	
Seeds	**0.874717**	1	0.855688	4	0.863418	2.5	0.865385	2.5
	(0.000000)		$(2.580960 \cdot 10^{-8})$		$(2.185360 \cdot 10^{-2})$		$(2.925020 \cdot 10^{-4})$	
Wine	**0.954541**	1.5	0.833111	3	0.797136	4	**0.957339**	1.5
	(0.000000)		$(1.053670 \cdot 10^{-8})$		$(9.432030 \cdot 10^{-2})$		$(2.986160 \cdot 10^{-2})$	
Corel 1	**0.772143**	1	0.707638	2.5	0.669628	4	0.709138	2.5
	$(4.537430 \cdot 10^{-4})$		$(1.466220 \cdot 10^{-2})$		$(8.031220 \cdot 10^{-3})$		$(3.378880 \cdot 10^{-2})$	
Corel 2	**0.753003**	1	0.556504	4	0.604160	3	0.715560	2
	$(1.490120 \cdot 10^{-8})$		$(3.630190 \cdot 10^{-2})$		$(3.419880 \cdot 10^{-2})$		$(3.159890 \cdot 10^{-2})$	
Reuters	0.420456	3	0.430050	2	0.395470	4	**0.468519**	1
	$(3.813350 \cdot 10^{-2})$		$(3.781280 \cdot 10^{-2})$		$(3.567560 \cdot 10^{-2})$		$(3.825180 \cdot 10^{-2})$	
ALOI	**0.719731**	1	0.392920	3.5	0.391184	3.5	0.527940	2
	$(2.368590 \cdot 10^{-2})$		$(1.071650 \cdot 10^{-2})$		$(2.143360 \cdot 10^{-2})$		$(6.768730 \cdot 10^{-2})$	

Fonte: O Autor (2019)

Tabela 15 – Erro Geral - Resultado Médio

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.331776**	1	0.493972	3	0.629393	4	0.430467	2
	$(2.172200 \cdot 10^{-8})$		$(2.503640 \cdot 10^{-2})$		$(4.797180 \cdot 10^{-3})$		$(4.552790 \cdot 10^{-2})$	
Image Segm.	**0.382039**	1	0.419056	2	0.436766	3	0.455225	4
	$(1.852620 \cdot 10^{-2})$		$(3.914530 \cdot 10^{-2})$		$(4.689610 \cdot 10^{-2})$		$(6.562800 \cdot 10^{-2})$	
Iris	0.058133	2.5	**0.042933**	1	0.052533	2.5	0.123467	4
	$(5.617500 \cdot 10^{-2})$		$(2.918630 \cdot 10^{-2})$		$(4.106670 \cdot 10^{-2})$		(0.122007)	
mFeat	**0.248235**	1	0.793935	4	0.574080	3	0.273985	2
	$(3.776420 \cdot 10^{-2})$		$(3.284910 \cdot 10^{-2})$		$(4.507060 \cdot 10^{-3})$		$(5.506020 \cdot 10^{-2})$	
Phonema	**0.161590**	1.5	0.161150	2	0.172575	2.5	0.223115	4
	$(2.863000 \cdot 10^{-2})$		$(5.745000 \cdot 10^{-2})$		$(5.566960 \cdot 10^{-2})$		$(5.817090 \cdot 10^{-2})$	
Seeds	**0.123810**	1	0.142857	4	0.135429	2.5	0.133333	2.5
	$(9.497660 \cdot 10^{-9})$		$(7.450580 \cdot 10^{-9})$		$(2.084740 \cdot 10^{-2})$		$(6.715860 \cdot 10^{-9})$	
Wine	**0.044944**	1.5	0.162921	3	0.198989	4	**0.042921**	1.5
	(0.000000)		$(1.086100 \cdot 10^{-8})$		$(9.240230 \cdot 10^{-2})$		$(3.577480 \cdot 10^{-2})$	
Corel 1	**0.225025**	1	0.295625	2	0.338350	4	0.307575	3
	$(4.322900 \cdot 10^{-4})$		$(1.362160 \cdot 10^{-2})$		$(6.530890 \cdot 10^{-3})$		$(3.976880 \cdot 10^{-2})$	
Corel 2	**0.245000**	1	0.437000	4	0.401000	3	0.287125	2
	(0.000000)		$(3.700000 \cdot 10^{-2})$		$(3.404040 \cdot 10^{-2})$		$(3.379970 \cdot 10^{-2})$	
Reuters	0.609425	3	0.594367	2	0.652267	4	**0.540433**	1
	$(4.865710 \cdot 10^{-2})$		$(4.838760 \cdot 10^{-2})$		$(5.077530 \cdot 10^{-2})$		$(3.788460 \cdot 10^{-2})$	
ALOI	**0.292519**	1	0.598054	4	0.594154	3	0.472611	2
	$(2.322640 \cdot 10^{-2})$		$(9.905150 \cdot 10^{-3})$		$(2.008270 \cdot 10^{-2})$		$(6.138860 \cdot 10^{-2})$	

Fonte: O Autor (2019)

Tabela 16 – Informação Mútua Normalizada - Resultado Médio

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.362588**	1	0.200405	3	0.114287	4	0.327978	2
	$(1.053670 \cdot 10^{-8})$		$(2.313260 \cdot 10^{-2})$		$(3.491360 \cdot 10^{-3})$		$(5.817410 \cdot 10^{-2})$	
Image Segm.	**0.614119**	1	0.585772	2	0.565368	3	0.528509	4
	$(2.040520 \cdot 10^{-3})$		$(4.346980 \cdot 10^{-2})$		$(5.486320 \cdot 10^{-2})$		$(7.511660 \cdot 10^{-2})$	
Iris	0.841816	2.5	**0.862206**	1	0.840665	2.5	0.776761	4
	$(3.901710 \cdot 10^{-2})$		$(1.969520 \cdot 10^{-2})$		$(4.105030 \cdot 10^{-2})$		(0.106957)	
mFeat	**0.703021**	1	0.308716	4	0.480500	3	0.670708	2
	$(7.397600 \cdot 10^{-3})$		$(2.543940 \cdot 10^{-2})$		$(9.421900 \cdot 10^{-4})$		$(3.685480 \cdot 10^{-2})$	
Phonema	0.749868	3	0.768676	2	**0.789201**	1	0.699169	4
	$(2.094240 \cdot 10^{-2})$		$(6.885140 \cdot 10^{-2})$		$(5.242640 \cdot 10^{-2})$		$(5.057530 \cdot 10^{-2})$	
Seeds	**0.680460**	1	0.662326	2	0.648711	4	0.656602	3
	(0.000000)		(0.000000)		$(1.938140 \cdot 10^{-2})$		$(7.591120 \cdot 10^{-3})$	
Wine	0.847290	2	0.585296	3.5	0.561004	3.5	**0.859748**	1
	$(6.143910 \cdot 10^{-8})$		$(4.012260 \cdot 10^{-8})$		(0.168319)		$(3.562760 \cdot 10^{-2})$	
Corel 1	**0.578512**	1	0.456549	3	0.424130	4	0.552073	2
	$(4.094500 \cdot 10^{-4})$		$(9.013590 \cdot 10^{-4})$		$(2.439390 \cdot 10^{-2})$		$(3.992900 \cdot 10^{-2})$	
Corel 2	**0.568072**	1	0.353502	4	0.393526	3	0.525163	2
	(0.000000)		$(8.702460 \cdot 10^{-2})$		$(3.618390 \cdot 10^{-2})$		$(2.481850 \cdot 10^{-2})$	
Reuters	0.232043	2.5	0.241386	2.5	0.195836	4	**0.274791**	1
	$(4.480450 \cdot 10^{-2})$		$(4.363120 \cdot 10^{-2})$		$(4.215610 \cdot 10^{-2})$		$(3.925490 \cdot 10^{-2})$	
ALOI	**0.740322**	1	0.401320	4	0.421650	3	0.612628	2
	$(1.192740 \cdot 10^{-2})$		$(2.189610 \cdot 10^{-3})$		$(1.236030 \cdot 10^{-2})$		$(4.911030 \cdot 10^{-2})$	

Fonte: O Autor (2019)

5.1.3 Tempo de Execução

O tempo médio de execução foi medido com implementações *single-thread* em C++ para todos os modelos. Os modelos foram executados no mesmo computador[1], de forma serial e sem nenhum outro programa pesado sendo executado, isso para não ocorrer concorrência pelos recursos da máquina, o que poderia interferir na medição do tempo médio.

[1] Informações do computador utilizado no teste: Sistema: Windows 10 Education 64-bits, Memória: 12GB, Processador: Intel Core i7-4790 CPU @ 3.60GHz

Tabela 17 – Tempo Médio (em segundos)

Base	FW4M		MFMMdd		$CARD_R$		MFCMdd-RWG-P	
Glass	0.859220	4	0.339970	3	**0.149700**	1	0.243060	2
(N:214, K:6, V:9)	(0.187223)		$(8.997180 \cdot 10^{-2})$		$(4.062230 \cdot 10^{-2})$		$(5.792490 \cdot 10^{-2})$	
Image Segm.	24.724900	4	**17.690200**	1	21.384600	2.5	21.878800	2.5
(N:2310, K:6, V:2)	(6.072850)		(4.691120)		(7.691660)		(5.332390)	
Iris	0.045150	4	0.036200	3	**0.020810**	1	0.031080	2
(N:150, K:3, V:4)	$(5.750430 \cdot 10^{-3})$		$(4.420410 \cdot 10^{-3})$		$(2.820270 \cdot 10^{-3})$		$(5.813230 \cdot 10^{-3})$	
mFeat	155.835000	2	201.737000	3	329.357000	4	**129.227000**	1
(N:2000, K:10, V:6)	(56.422700)		(38.010400)		(119.189000)		(25.844400)	
Phonema	35.405400	4	30.533500	3	**11.788000**	1	20.966000	2
(N:2000, K:5, V:3)	(16.935900)		(18.483100)		(7.802880)		(3.528920)	
Seeds	0.087220	2	0.094110	3.5	0.096970	3.5	**0.060950**	1
(N:210, K:3, V:7)	$(2.217730 \cdot 10^{-2})$		$(1.719350 \cdot 10^{-2})$		$(1.648300 \cdot 10^{-2})$		$(1.189910 \cdot 10^{-2})$	
Wine	0.153480	2.5	1.423980	4	0.168030	2.5	**0.073170**	1
(N:178, K:3, V:13)	$(4.521760 \cdot 10^{-2})$		(0.253771)		(0.106955)		$(1.544280 \cdot 10^{-2})$	
Corel 1	0.910080	2	3.082670	4	1.290950	3	**0.381820**	1
(N:400, K:4, V:7)	(0.162698)		(0.363418)		(0.234558)		$(5.123250 \cdot 10^{-2})$	
Corel 2	3.320610	3	3.684440	4	1.386800	2	**0.473880**	1
(N:400, K:4, V:7)	(0.571978)		(1.700420)		(0.552017)		$(8.193020 \cdot 10^{-2})$	
Reuters	1.964630	2	**1.317680**	1	2.745300	3	3.280510	4
(N:1200, K:6, V:5)	(0.230820)		$(6.855990 \cdot 10^{-2})$		(0.246800)		(0.588498)	
ALOI	**32.029600**	1	39.446000	3	81.219200	4	34.427600	2
(N:1413, K:13, V:4)	(8.781580)		(10.310000)		(22.688700)		(10.224400)	

Fonte: O Autor (2019)

É possível observar que o tempo do modelo MFCMdd-RWG-P. O motivo principal disso é o fato dele utilizar um número limitado de elementos como representante dos grupos, o que reduz o número de operações nos seus cálculos.

5.2 MELHORES RESULTADOS

Foi mantido os resultados da execução que obteve o menor valor para a função objetivo, dentre as 100 execuções para cada combinação modelo x base. Os resultados dos experimentos para métricas da partição difusa e da partição *crisp* estão descritos nas tabelas seguintes, com a organização:

média	*rank*

Assim como os resultados da seção anterior, esses também serão comparados nas seções 5.3, 5.4 e 5.5.

5.2.1 Métricas da Partição Difusa

Tabela 18 – Rand Frigui - Melhor Resultado

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.713016**	1	0.642745	3	0.482734	4	0.673466	2
Image Segm.	0.847028	4	0.847143	3	0.849823	2	**0.856254**	1
Iris	0.935116	4	**0.948125**	1	0.946303	2	0.937671	3
mFeat	0.914150	2	0.858048	4	0.866057	3	**0.924918**	1
Phonema	0.820881	3	0.866115	2	**0.904846**	1	0.785086	4
Seeds	**0.852132**	1	0.839794	4	0.843144	3	0.846477	2
Wine	0.904725	2	0.800708	3	0.732982	4	**0.908935**	1
Corel 1	0.770629	2	0.747609	4	0.760329	3	**0.773623**	1
Corel 2	0.714219	4	0.746328	2	0.736123	3	**0.762718**	1
Reuters	**0.722686**	2	**0.722686**	2	**0.722686**	2	0.721074	4
ALOI	0.918639	2	0.861921	3	0.838541	4	**0.930452**	1

Fonte: O Autor (2019)

Tabela 19 – Rand Hullermeier - Melhor Resultado

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.692723**	1	0.642523	3	0.482734	4	0.670449	2
Image Segm.	0.844470	4	0.846982	3	0.849748	2	**0.855512**	1
Iris	0.935418	4	**0.948051**	1	0.946297	2	0.937734	3
mFeat	0.900846	2	0.857850	4	0.865959	3	**0.918277**	1
Phonema	0.781299	3	0.856443	2	**0.903780**	1	0.708697	4
Seeds	**0.851839**	1	0.839739	4	0.843048	3	0.846298	2
Wine	0.908419	2	0.800289	3	0.732973	4	**0.911950**	1
Corel 1	0.733476	4	0.741185	3	0.758746	2	**0.766596**	1
Corel 2	0.620925	4	**0.743451**	1	0.734487	2	0.723124	3
Reuters	0.165972	3.5	0.165972	3.5	0.165979	2	**0.448165**	1
ALOI	0.916093	2	0.861665	3	0.838360	4	**0.928085**	1

Fonte: O Autor (2019)

Tabela 20 – Distância Mínima entre Centroides - Melhor Resultado

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	0.001385	4	0.001494	3	**0.002396**	1	0.002218	2
Image Segm.	**0.000295**	1	0.000220	4	0.000267	2	0.000264	3
Iris	0.006337	2	0.006238	4	**0.006674**	1	0.006308	3
mFeat	**0.000276**	1	0.000166	4	0.000198	3	0.000264	2
Phonema	**0.000491**	1	0.000390	4	0.000473	2	0.000412	3
Seeds	0.005200	3	0.005529	2	**0.005584**	1	0.005071	4
Wine	0.007346	2	0.007116	3	**0.007451**	1	0.007083	4
Corel 1	0.002088	2	0.001922	4	**0.002559**	1	0.002028	3
Corel 2	0.002438	2	0.001710	4	**0.002910**	1	0.001741	3
Reuters	0.000905	2.5	0.000905	2.5	**0.000918**	1	0.000652	4
ALOI	**0.000218**	1	0.000079	4	0.000084	3	0.000189	2

Fonte: O Autor (2019)

5.2.2 Métricas da Partição *Crisp*

Tabela 21 – F-measure - Melhor Resultado

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.549631**	1	0.452337	3	0.345579	4	0.460211	2
Image Segm.	0.631178	2	0.621512	3	0.620977	4	**0.654487**	1
Iris	0.953291	4	**0.959984**	1	0.953329	2.5	0.953329	2.5
mFeat	0.801552	2	0.457695	4	0.458609	3	**0.817655**	1
Phonema	0.840972	3	0.857884	2	**0.862974**	1	0.821366	4
Seeds	**0.874717**	1	0.855688	4	0.865614	2	0.865011	3
Wine	0.954541	2	0.833111	3	0.742510	4	**0.960340**	1
Corel 1	**0.772155**	1	0.643727	4	0.667599	3	0.695337	2
Corel 2	**0.753003**	1	0.552573	4	0.576010	3	0.676378	2
Reuters	0.421746	2	0.409430	3	0.370049	4	**0.467502**	1
ALOI	0.673647	2	0.368287	3	0.340332	4	**0.722081**	1

Fonte: O Autor (2019)

Tabela 22 – Erro Geral - Melhor Resultado

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.331776**	1	0.509346	3	0.630841	4	0.448598	2
Image Segm.	0.396104	2	0.403463	4	0.402597	3	**0.371861**	1
Iris	0.046667	3	**0.040000**	1	0.046667	3	0.046667	3
mFeat	0.202000	2	0.564000	4	0.552000	3	**0.185000**	1
Phonema	0.157500	3	0.142000	2	**0.137000**	1	0.183000	4
Seeds	**0.123810**	1	0.142857	4	0.133333	2.5	0.133333	2.5
Wine	0.044944	2	0.162921	3	0.252809	4	**0.039326**	1
Corel 1	**0.225000**	1	0.355000	4	0.340000	3	0.322500	2
Corel 2	**0.245000**	1	0.440000	4	0.422500	3	0.327500	2
Reuters	0.582500	3	0.575833	2	0.694167	4	**0.526667**	1
ALOI	0.334749	2	0.612880	3	0.643312	4	**0.277424**	1

Fonte: O Autor (2019)

Tabela 23 – Informação Mútua Normalizada - Melhor Resultado

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	**0.362588**	1	0.207895	3	0.113213	4	0.336707	2
Image Segm.	0.615720	2	0.601361	4	0.604714	3	**0.623842**	1
Iris	0.849781	2	**0.864186**	1	0.846483	3.5	0.846483	3.5
mFeat	0.711329	2	0.486772	3	0.475884	4	**0.721823**	1
Phonema	0.752860	3	0.791626	2	**0.798306**	1	0.730282	4
Seeds	**0.680460**	1	0.662326	3	0.650659	4	0.666298	2
Wine	0.847290	2	0.585296	3	0.453138	4	**0.863329**	1
Corel 1	**0.578575**	1	0.452620	3	0.417967	4	0.571108	2
Corel 2	**0.568072**	1	0.382361	3	0.371647	4	0.488173	2
Reuters	0.221831	3	0.261790	2	0.161014	4	**0.318022**	1
ALOI	0.708259	2	0.397914	3	0.389871	4	**0.733236**	1

Fonte: O Autor (2019)

5.2.3 Tempo de Execução

Tabela 24 – Tempo de execução (em segundos) - Melhor Resultado

Datasets	MFMMdd		FW4M		$CARD_R$		MFCMdd-RWG-P	
Glass	0.582000	4	0.351000	3	**0.175000**	1	0.212000	2
Image Segm.	32.791000	4	19.712000	2	28.046000	3	**14.569000**	1
Iris	0.058000	4	0.028000	3	**0.023000**	1	0.027000	2
mFeat	**103.605000**	1	178.449000	3	235.402000	4	116.910000	2
Phonema	50.359000	4	23.450000	2	**6.181000**	1	32.883000	3
Seeds	0.082000	3	0.080000	2	0.090000	4	**0.050000**	1
Wine	0.152000	3	1.017000	4	**0.064000**	1	0.081000	2
Corel 1	0.913000	2	1.698000	4	1.254000	3	**0.415000**	1
Corel 2	3.298000	4	1.939000	3	0.700000	2	**0.416000**	1
Reuters	1.924000	2	**1.295000**	1	2.621000	3	3.006000	4
ALOI	28.125000	2	38.949000	3	75.847000	4	**25.881000**	1

Fonte: O Autor (2019)

5.3 *RANKS* MÉDIOS

Os *ranks* podem variar de acordo com a base de dados testada. Para facilitar a comparação, foram calculados os *ranks* médios de cada modelo para cada uma das métricas.

Tabela 25 – *Ranks* médios - Resultados Médios

Métrica	MFMMdd	FW4M	$CARD_R$	MFCMdd-RWG-P
Rand Frigui	**1.955**	2.727	2.955	2.364
Rand Hullermeier	**2.273**	2.545	2.636	2.545
Distância Mínima entre Centroides	**1.955**	3.091	2.000	2.955
F-measure	**1.455**	2.818	3.273	2.455
Erro Geral	**1.409**	2.818	3.227	2.545
Informação Mútua Normalizada	**1.545**	2.818	3.182	2.455
Tempo Médio	2.773	2.955	2.500	**1.773**

Fonte: O Autor (2019)

Tabela 26 – *Ranks* médios - Melhores Resultados

Métrica	MFMMdd	FW4M	$CARD_R$	MFCMdd-RWG-P
Rand Frigui	2.455	2.818	2.818	**1.909**
Rand Hullermeier	2.773	2.773	2.636	**1.818**
Distância Mínima entre Centroides	1.955	3.500	**1.545**	3.000
F-measure	1.909	3.091	3.136	**1.864**
Erro Geral	1.909	3.091	3.136	**1.864**
Informação Mútua Normalizada	**1.818**	2.727	3.591	1.864
Tempo Médio	3.000	2.727	2.455	**1.818**

Fonte: O Autor (2019)

Lembrando que quanto menor o *rank*, melhor o resultado.

5.4 TESTE DE FRIEDMAN

Aplicando o Teste de Friedman para os resultados anteriores, obtemos:

Tabela 27 – Teste de Friedman - Resultados Médios

Métrica	F_F	Rejeita H_0
Rand Frigui	1.17887	não
Rand Hullermeier	0.126077	não
Distância Mínima entre Centroides	2.56316	não
F-measure	5.05241	sim
Erro Geral	5.13742	sim
Informação Mútua Normalizada	3.78517	sim
Tempo Médio	1.7529	não

Fonte: O Autor (2019)

Tabela 28 – Teste de Friedman - Melhores Resultados

Métrica	F_F	Rejeita H_0
Rand Frigui	1.12044	não
Rand Hullermeier	1.30380	não
Distância Mínima entre Centroides	8.70991	sim
F-measure	3.88495	sim
Erro Geral	3.88495	sim
Informação Mútua Normalizada	6.57864	sim
Tempo Médio	1.63497	não

Fonte: O Autor (2019)

Como podemos observar, os modelos foram considerados equivalentes para as métricas Rand Frigui, Rand Hullermeier e em relação ao Tempo Médio de execução, tanto para o melhor caso, quanto para os casos médios. Além de serem equivalentes para a distância mínima entre os centroides em relação aos resultados médios;

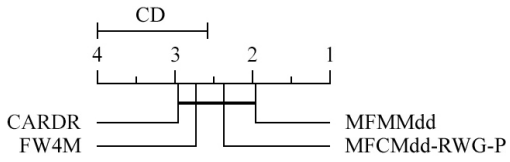
5.5 PÓS-TESTE DE NEMENYI

Após executar o teste de Friedman, foi executado o pós-teste de Nemenyi com o coeficiente CD = 1,414 (como calculado pela Equação 4.8). Esse pós-teste só precisava ser feito para os casos nos quais a hipótese nula do teste de Friedman foi rejeitada, contudo ele foi feito com todas as métricas por completude.

5.5.1 Resultados Médios

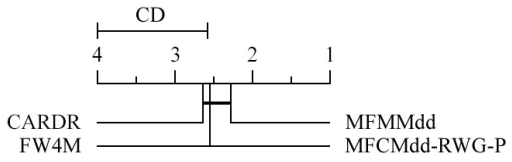
Os resultados do pós-teste de Nemenyi, para os resultados médios, foram:

Figura 12 – Rand Frigui - Resultados Médios



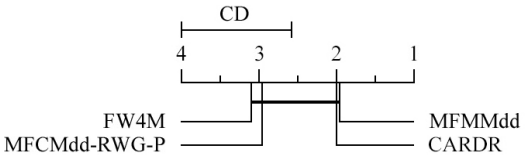
Fonte: O Autor (2019)

Figura 13 – Rand Hullermeier - Resultados Médios



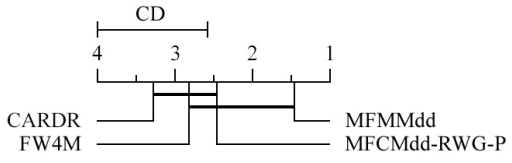
Fonte: O Autor (2019)

Figura 14 – Distância Mínima entre Centroides - Resultados Médios



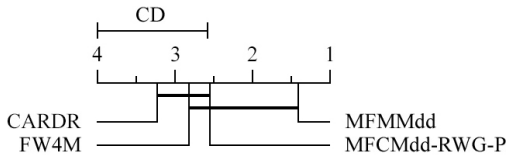
Fonte: O Autor (2019)

Figura 15 – F-Measure - Resultados Médios



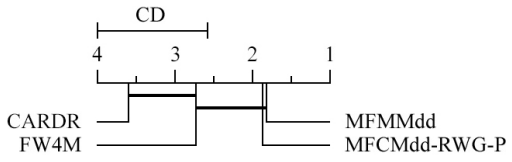
Fonte: O Autor (2019)

Figura 16 – Erro Geral - Resultados Médios



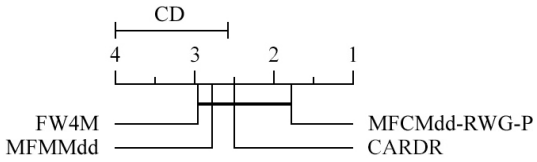
Fonte: O Autor (2019)

Figura 17 – NMI - Resultados Médios



Fonte: O Autor (2019)

Figura 18 – Tempo Médio - Resultados Médios



Fonte: O Autor (2019)

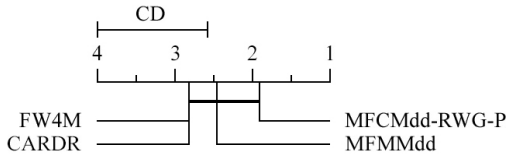
Esses resultados demostram que o modelo proposto obteve o melhor resultado médio para as métricas *crisps* e difusas (tendo pelo menos um dos outros modelos com resultados equivalentes ao mesmo).

Já o tempo dos modelos foi equivalente entre os mesmos, sendo o MFCMdd-RWG-P o mais rápido entre eles. Um dos motivos disso está no fato do modelo MFCMdd-RWG-P utilizar um conjunto reduzido de representantes para cada grupo, ao contrário dos demais modelos que utilizam uma ponderação de todos os elementos da base.

5.5.2 Melhores Resultados

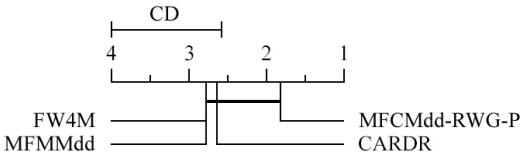
Os resultados do teste de Nemenyi, para os melhores resultados, foram:

Figura 19 – Rand Frigui - Melhores Resultados



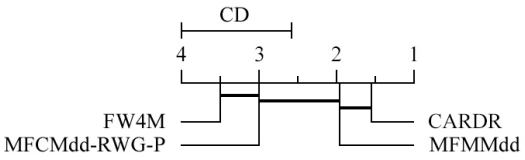
Fonte: O Autor (2019)

Figura 20 – Rand Hullermeier - Melhores Resultados



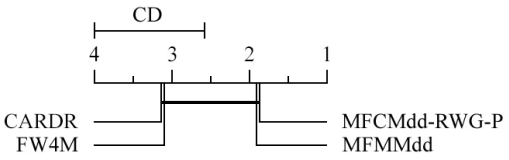
Fonte: O Autor (2019)

Figura 21 – Distância Mínima entre Centroides - Melhores Resultados



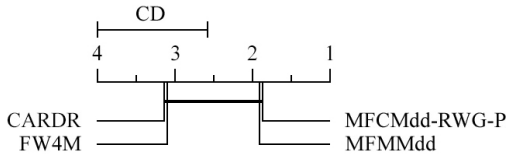
Fonte: O Autor (2019)

Figura 22 – F-Measure - Melhores Resultados



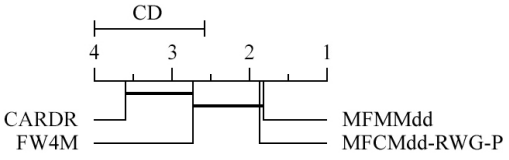
Fonte: O Autor (2019)

Figura 23 – Erro Geral - Melhores Resultados



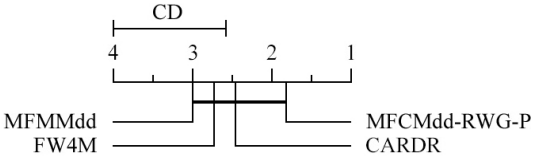
Fonte: O Autor (2019)

Figura 24 – NMI - Melhores Resultados



Fonte: O Autor (2019)

Figura 25 – Tempo Médio - Melhores Resultados



Fonte: O Autor (2019)

O Teste de Nemenyi para os melhores resultados demonstra que o modelo proposto obteve resultados semelhantes ao melhor modelo para todas as métricas. Tendo nenhum dos *ranks* médios pior do que os obtidos pelo modelo de referência (FW4M).

A medição do melhor resultado é mais sensível à inicialização aleatória do que o do resultado médio e, por tanto, mais susceptível à variações. Esse pode ser um dos motivos do comportamento dos melhores resultados diferir do comportamento dos resultados médios.

6 CONCLUSÕES E TRABALHOS FUTUROS

Esse trabalho apresentou o modelo *"Multi-view Relational Fuzzy Clustering Algorithm with Multi-medoids"* (MFMMdd), um modelo de clusterização difusa para dados relacionais descritos com múltiplas visões, que pondera as visões com uma restrição de produto. Esse modelo começa com uma solução aleatória e melhora o resultado de forma iterativa, até as pertinências estabilizarem ou um número limite de iterações for atingido.

O modelo proposto foi comparado com 3 outros modelos: o modelo CARD_R (que não possui uma representação distinta para os *clusters*, utilizando as pertinências dos elementos no lugar), o modelo MFCMdd-RWG-P (que também utiliza a restrição do produto para as ponderações das visões) e FW4M (o modelo de referência, que utiliza a restrição da soma para as ponderações das visões).

Os parâmetros de cada modelo foram encontrados, para cada uma das bases testadas, utilizando uma busca em grade com o objetivo de minimizar a média dos Coeficientes de Partição Modificado de 20 execuções. Com os parâmetros definidos, cada modelo foi executado 100 vezes para cada uma das bases testadas (Glass, Image Segmentation, Iris, multiple Features, Phonema, Seeds, Wine, Corel 1, Corel 2, Reuters, ALOI-Cars), sendo calculadas métricas para partição difusa (Rand Frigui, Rand Hullermeier e Distância Mínima entre Centroides) e para partição *crisp* (F-Measure, Erro Geral e Informação Mútua Normalizada), além do tempo médio da execução dos modelos. Esses resultados então foram ordenados em *ranks* utilizando uma comparação baseada na distribuição t-student para, então, obter os *ranks* médios de cada métrica para cada modelo, permitindo a comparação dos modelos.

6.1 RESUMO DOS RESULTADOS

Os testes de Friedman e de Nemenyi demonstraram que, para as bases testadas, o modelo proposto obteve resultados similares ou superiores ao modelo de referência FW4M e ao modelo MFCMdd-RWG-P para todas as métricas. O modelo obteve o melhor *rank* médio para 6 métricas dos resultados médios (todas menos o tempo) e para 1 métrica do melhor resultado (NMI), e o segundo melhor para 4 métricas do melhor resultado (Rand Frigui, Distância mínima entre centroides, F-Measure e Erro Geral).

Além dos resultados satisfatórios encontrados, o modelo proposto também possui um parâmetro a menos em relação ao modelo de referência, o que reduz o número de dimensões do espaço de busca na seleção de parâmetros. Com isso, o tempo necessário para seleção dos parâmetros é reduzindo. Com essas considerações, o objetivo do trabalho foi atingido. Mesmo assim, nenhum dos modelos foi o melhor para todas as bases, o que demonstra a importância da busca de modelos diferentes, permitindo um leque de opções para a

solução de cada problema.

6.2 TRABALHOS FUTUROS

Trabalhos futuros possíveis são: Expandir os testes para outras bases de dados, principalmente bases grandes com dezenas de classes e/ou visões, com o objetivo de verificar se os resultados encontrados também são válidos para esse tipo de base; Utilizar uma ponderação das visões para cada *cluster* (como as ponderações locais de soma e produto utilizadas pelos modelos MFCMdd-RWL-S e MFCMdd-RWL-P, respectivamente); Permitir que o número de *clusters* varie durante a execução, permitindo uma melhor divisão para quando o número de classes não é conhecido previamente ou para quando existem classes não contínuas, podendo fragmentar essas classes em múltiplos *clusters* (como demonstrado pela figura 26).

Figura 26 – Impacto do número de *clusters* para classes com múltiplos agrupamentos

Fonte: O Autor (2019)

Também pode ser buscada uma forma de agrupar as soluções dos diferentes modelos, assim como pode ser feito para agrupar classificadores treinados de forma supervisionada num único classificador, o que costuma obter resultados melhores do que os das partes. A maior dificuldade dessa junção seria determinar o mapeamento entre os *clusters* de cada solução, principalmente se não possuir as classes a priori dos elementos.

REFERÊNCIAS

1 MANUEL, F. E. Rules for methodizing the apocalypse, rule 9. In: The Religion of Isaac Newton. [S.l.]: Clarendon Press, 1974. p. 120.

2 HATHAWAY, R. J.; BEZDEK, J. C. Nerf c-means: non-euclidean relational fuzzy clustering. Pattern Recognition, v. 27, n. 3, p. 429–437, 1994.

3 KAUFMAN, L.; ROUSSEEUW, P. J. Fuzzy analysis (program fanny). In: Finding Groups in Data. [S.l.]: Wiley, 1990. cap. 4.

4 KRISHNAPURAM, R.; JOSHI, A.; NASRAOUI, O.; YI, L. Low-complexity fuzzy relational clustering algorithms for web mining. IEEE Transactions on Fuzzy Systems, v. 9, p. 595–607, 2001.

5 MEI, J.-P.; CHEN, L. Fuzzy relational clustering around medoids: A unified view. Fuzzy Sets and Systems, v. 183, n. 1, p. 44–56, 2011.

6 FRIGUI, H.; HWANG, C.; RHEE, F. C.-H. Clustering and aggregation of relational data with applications to image database categorization. Pattern Recognition, v. 40, p. 3053–3068, 2007.

7 CARVALHO, F. de Assis Tenorio de; LECHEVALLIER, Y.; MELO, F. M. Relational partitioning fuzzy clustering algorithms based on multiple dissimilarity matrices. Fuzzy Sets and Systems, v. 215, p. 1–28, 2013.

8 GAO, Y.; SUN, C.; QI, H.; WANG, S. Fuzzy clustering based on weighted multi-medoids and multi-matrices. International Journal of Advancements in Computing Technology, v. 6, n. 3, p. 61–70, 2014.

9 The Khan Academy team. Lagrange multipliers, introduction. <https://www.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/constrained-optimization/a/lagrange-multipliers-single-constraint>. [Online; acessado 15 de Junho, 2019].

10 SIMÕES, E. C.; CARVALHO, F. de A. T. de. A fuzzy clustering algorithm with multi-medoids for multi-view relational data. Advances in Neural Networks, ISNN 2019, p. 469–477, 2019.

11 FRIEDMAN, M. A comparison of alternative tests of significance for the problem of m rankings. Annals of Mathematical Statistics, v. 11, p. 86–92, 1940.

12 IMAN, L.; DAVENPORT, J. M. Approximations of the critical region of the friedman statistic. Communications in Statistics, p. 571–595, 1980.

13 NEMENYI, B. Distribution-free multiple comparisons. Tese (Doutorado) — Princeton University, 1963.

14 ZHU, C.; JI, X.; CHEN, C.; ZHOU, R.; WEI, L.; ZHANG, X. Critical values for the two-tailed Nemenyi test. PLOS ONE, 2018. Disponível em: <https://plos.figshare.com/articles/Critical_values_for_the_two-tailed_Nemenyi_test_/7299293/1>.

15 BREIMAN, L.; FRIEDMAN, J.; STONE, C. J.; OLSHEN, R. A. Classification and regression trees. Chapman and Hall/CRC, Boca Raton, 1984.

16 MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. Introduction to information retrieval. Cambridge University Press, 2008.

17 COVER, T. M.; THOMAS, J. A. Entropy, relative entropy, and mutual information. In: Elements of information theory. [S.l.]: Wiley, 1991. cap. 2.

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

19 HÜLLERMEIER, E.; RIFQI, M.; HENZGEN, S.; SENGE, R. Comparing fuzzy partitions: A generalization of the rand index and related measures. IEEE Transactions on Fuzzy Systems, v. 20, n. 3, p. 546–556, 2012.

20 BEZDEK, J. C. Cluster validity with fuzzy sets. Journal of Cybernetics, v. 3, p. 58–73, 1974.

21 DAVE, R. N. Validating fuzzy partitions obtained through c-shells clustering. Pattern Recognition Letters, v. 17, p. 613–623, 1996.

22 GERMAN, B. Glass Identification Data Set. <https://archive.ics.uci.edu/ml/datasets/glass+identification>. [Online; acessado 28 de Agosto, 2018].

23 Vision Group, University of Massachusetts . Image Segmentation Data Set. <http://archive.ics.uci.edu/ml/datasets/image+segmentation>. [Online; acessado 19 de Setembro, 2018].

24 FISHER, R. A. Iris Data Set. <http://archive.ics.uci.edu/ml/datasets/iris>. [Online; acessado 28 de Agosto, 2018].

25 ROBERT P. W. DUIN, Delft University of Technology. Multiple Features Data Set. <https://archive.ics.uci.edu/ml/datasets/Multiple+Features>. [Online; acessado 28 de Agosto, 2018].

26 BUJA, A.; STUETZLE, W.; MAECHLER, M. Phoneme dataset. <http://www.math.univ-toulouse.fr/staph/npfda/npfda-datasets.html>. [Online; acessado 28 de Agosto, 2018].

27 CHARYTANOWICZ, M.; NIEWCZAS, J. Seeds Data Set. <https://archive.ics.uci.edu/ml/datasets/seeds>. [Online; acessado 28 de Agosto, 2018].

28 FORINA, M. et al, PARVUS. Wine Data Set. <https://archive.ics.uci.edu/ml/datasets/wine>. [Online; acessado 28 de Agosto, 2018].

29 JIN, X. Corel Images. <http://www.cs.virginia.edu/~xj3a/research/CBIR/Download.htm>. [Online; acessado 20 de Março, 2018].

30 AMINI, M.-R.; GOUTTE, C. Reuters RCV1 RCV2 Multilingual, Multiview Text Categorization Test collection Data Set. <http://archive.ics.uci.edu/ml/datasets/Reuters+RCV1+RCV2+Multilingual%2C+Multiview+Text+Categorization+Test+collection>. [Online; acessado 20 de Junho, 2019].

31 GEUSEBROEK, J.-M.; BURGHOUTS, G. Amsterdam Library of Object Images (ALOI). <http://aloi.science.uva.nl>. [Online; acessado 20 de Junho, 2019].

32 CHAVENT, M. Normalized k-means clustering of hyper-rectangles. Proceedings of the XI International Symposium of Applied Stochastic Models and Data Analysis (ASMDA 2005), Brest, p. 670–677, 2005.

33 CHARYTANOWICZ, M.; KULCZYCKI, P.; KOWALSKIB, P. A.; ŁUKASIK, S.; CZABAK-GARBACZ, R. An evaluation of utilizing geometric features for wheat grain classicationusing x-ray images. Computers and Electronics in Agriculture, v. 114, p. 260–268, 2018.

34 DE ARAÚJO, R. C. Um modelo de agrupamento multi-view com ponderação simultânea de tabelas e variáveis. Dissertação (Mestrado) — Universidade Federal de Pernambuco, Recife, Pernambuco, Brasil, 2018.

35 KUMAR, A.; DAUMÉ, H. A co-training approach for multi-view spectral clustering. Proceedings of the 28th International Conference on Machine Learning, ICML 2011, p. 393–400, 2011.

36 CORMEN, T.; BALKCOM, D.; The Khan Academy computing curriculum team. Big-O notation. <https://www.khanacademy.org/computing/computer-science/algorithms/asymptotic-notation/a/big-o-notation>. [Online; acessado 08 de Junho, 2019].

APÊNDICE A – SOLUCIONANDO O SISTEMA DE LAGRANGE

Função de Lagrange do modelo proposto:

$$\begin{aligned} L_{MFMMdd)} = & \sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) - \sum_{i=1}^{N} \alpha_i \left(\sum_{c=1}^{K} u_{ic} - 1 \right) \\ & - \sum_{c=1}^{K} \beta_c \left(\sum_{j=1}^{N} v_{cj} - 1 \right) - \gamma \left(\prod_{p=1}^{P} \delta_p - 1 \right) \end{aligned} \tag{A.1}$$

As propriedades utilizadas por essa demonstração são:

1. $(AB)^n = A^n B^n$
2. $\left(\frac{A}{B}\right)^n = \frac{A^n}{B^n}$
3. $\frac{A}{B} = AB^{-1} = \frac{1}{A^{-1}} B^{-1} = \frac{B^{-1}}{A^{-1}}$
4. $\sum_{i=1}^{N} cf(x_i) = c \sum_{i=1}^{N} f(x_i)$, sendo c independente de i
5. $A = B \iff f(A) = f(B)$
   a) $A + C = B \iff A = B - C$
   b) $A \cdot C = B \iff A = B/C$
   c) $A^C = B \iff A = B^{1/C}$
6. $\prod_{i=1}^{N} cf(x_i) = c^N \prod_{i=1}^{N} f(x_i)$, sendo c independente de i
7. $\prod_{i=1}^{N} f(x_i)^n = \left[\prod_{i=1}^{N} f(x_i)\right]^n$

A.1 OBTENDO A OTIMIZAÇÃO DAS PERTINÊNCIAS

Para encontrar a equação de otimização das pertinências, precisamos utilizar as derivadas relacionadas às variáveis u_{ic}: $\frac{\partial L_{MFMMdd)}}{\partial u_{ic}} = 0$ e $\frac{\partial L_{MFMMdd)}}{\partial \alpha_i} = 0$:

$$\begin{aligned} \frac{\partial L_{MFMMdd)}}{\partial u_{ic}} &= 0 \\ m \cdot u_{ic}^{(m-1)} \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) - \alpha_i &= 0 [derivada] \\ m \cdot u_{ic}^{(m-1)} \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) &= \alpha_i [5a] \\ u_{ic}^{(m-1)} &= \frac{\alpha_i}{m \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)} [5b] \\ u_{ic} &= \left(\frac{\alpha_i}{m \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{m-1}} [5c] \end{aligned} \tag{A.2}$$

$$\begin{aligned} \frac{\partial L_{MFMMdd)}}{\partial \alpha_i} &= 0 \\ 1 - \sum_{k=1}^{K} u_{ik} &= 0[derivada] \\ \sum_{k=1}^{K} u_{ik} &= 1[5a] \end{aligned} \tag{A.3}$$

Substituindo as equações A.2 em A.3 temos:

$$\begin{aligned} \sum_{k=1}^{K} \left(\frac{\alpha_i}{m \sum_{j=1}^{N} v_{kj}^n \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{m-1}} &= 1 \\ \sum_{k=1}^{K} \alpha_i^{\frac{1}{m-1}} \left(\frac{1}{m \sum_{j=1}^{N} v_{kj}^n \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{m-1}} &= 1[1] \\ \alpha_i^{\frac{1}{m-1}} \sum_{k=1}^{K} \left(\frac{1}{m \sum_{j=1}^{N} v_{kj}^n \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{m-1}} &= 1[4] \\ \alpha_i^{\frac{1}{m-1}} = \left[\sum_{k=1}^{K} \left(\frac{1}{m \sum_{j=1}^{N} v_{kj}^n \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{m-1}} \right]^{-1} & [5b] \\ \alpha_i = \left[\sum_{k=1}^{K} \left(\frac{1}{m \sum_{j=1}^{N} v_{kj}^n \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{m-1}} \right]^{-(m-1)} & [5c] \end{aligned} \tag{A.4}$$

Agora que isolamos α_i, podemos Substituir a equação A.4 em A.2:

$$
\begin{aligned}
u_{ic} &= \left(\frac{\left[\sum_{k=1}^{K} \left(\frac{1}{m \sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{m-1}} \right]^{-(m-1)}}{m \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{m-1}} \\
u_{ic} &= \frac{\left[\sum_{k=1}^{K} \left(m \sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{-1}{m-1}} \right]^{-1}}{\left(m \sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{1}{m-1}}} [2] \\
u_{ic} &= \frac{\left[\sum_{k=1}^{K} m^{\frac{-1}{m-1}} \left(\sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{-1}{m-1}} \right]^{-1}}{m^{\frac{1}{m-1}} \left(\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{1}{m-1}}} [1] \\
u_{ic} &= \frac{m^{\frac{1}{m-1}} \left[\sum_{k=1}^{K} \left(\sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{-1}{m-1}} \right]^{-1}}{m^{\frac{1}{m-1}} \left(\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{1}{m-1}}} [4,1] \\
u_{ic} &= \frac{\left[\sum_{k=1}^{K} \left(\sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{-1}{m-1}} \right]^{-1}}{\left(\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{1}{m-1}}} \\
u_{ic} &= \frac{\left(\sum_{j=1}^{N} v_{cj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{-1}{m-1}}}{\sum_{k=1}^{K} \left(\sum_{j=1}^{N} v_{kj}^{n} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{-1}{m-1}}} [3]
\end{aligned}
\tag{A.5}
$$

A.2 OBTENDO A OTIMIZAÇÃO DAS REPRESENTAÇÕES

Para encontrar a equação de otimização das representações, precisamos utilizar as derivadas relacionadas às variáveis v_{cj}: $\frac{\partial L_{MFMMdd)}}{\partial v_{cj}} = 0$, $\frac{\partial L_{MFMMdd)}}{\partial \beta_c} = 0$ e

$$
\begin{aligned}
\frac{\partial L_{MFMMdd)}}{\partial v_{cj}} &= 0 \\
n \cdot v_{cj}^{(n-1)} \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j) - \beta_c &= 0 [derivada] \\
n \cdot v_{cj}^{(n-1)} \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j) &= \beta_c [5a] \\
v_{cj}^{(n-1)} &= \frac{\beta_c}{n \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j)} [5b] \\
v_{cj} &= \left(\frac{\beta_c}{n \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{n-1}} [5c]
\end{aligned}
\tag{A.6}
$$

$$\begin{aligned} \frac{\partial L_{MFMMdd)}}{\partial \beta_c} &= 0 \\ 1 - \sum_{h=1}^{N} v_{ch} &= 0[derivada] \\ \sum_{h=1}^{N} v_{ch} &= 1[5a] \end{aligned} \tag{A.7}$$

Substituindo as equações A.6 em A.7 temos:

$$\begin{aligned} \sum_{h=1}^{N} \left(\frac{\beta_c}{n \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_p D_p(i,h)} \right)^{\frac{1}{n-1}} &= 1 \\ \sum_{h=1}^{N} \beta_c^{\frac{1}{n-1}} \left(\frac{1}{n \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_p D_p(i,h)} \right)^{\frac{1}{n-1}} &= 1[1] \\ \beta_c^{\frac{1}{n-1}} \sum_{h=1}^{N} \left(\frac{1}{n \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_p D_p(i,h)} \right)^{\frac{1}{n-1}} &= 1[4] \\ \beta_c^{\frac{1}{n-1}} = \left[\sum_{h=1}^{N} \left(\frac{1}{n \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_p D_p(i,h)} \right)^{\frac{1}{n-1}} \right]^{-1} & [5b] \\ \beta_c = \left[\sum_{h=1}^{N} \left(\frac{1}{n \sum_{i=1}^{N} u_{ic}^m \sum_{p=1}^{P} \delta_p D_p(i,h)} \right)^{\frac{1}{n-1}} \right]^{-(n-1)} & [5c] \end{aligned} \tag{A.8}$$

Agora que isolamos β_c, podemos Substituir a equação A.8 em A.6:

$$
\begin{aligned}
v_{cj} &= \left(\frac{\left[\sum_{h=1}^{N} \left(\frac{1}{n \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,h)} \right)^{\frac{1}{n-1}} \right]^{-(n-1)}}{n \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j)} \right)^{\frac{1}{n-1}} \\
v_{cj} &= \frac{\left[\sum_{h=1}^{N} \left(n \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,h) \right)^{\frac{-1}{n-1}} \right]^{-1}}{\left(n \sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{1}{n-1}}} [2] \\
v_{cj} &= \frac{\left[\sum_{h=1}^{N} n^{\frac{-1}{n-1}} \left(\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,h) \right)^{\frac{-1}{n-1}} \right]^{-1}}{n^{\frac{1}{n-1}} \left(\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{1}{n-1}}} [1] \\
v_{cj} &= \frac{n^{\frac{1}{n-1}} \left[\sum_{h=1}^{N} \left(\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,h) \right)^{\frac{-1}{n-1}} \right]^{-1}}{n^{\frac{1}{n-1}} \left(\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{1}{n-1}}} [4,1] \\
v_{cj} &= \frac{\left[\sum_{h=1}^{N} \left(\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,h) \right)^{\frac{-1}{n-1}} \right]^{-1}}{\left(\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{1}{n-1}}} \\
v_{cj} &= \frac{\left(\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,j) \right)^{\frac{-1}{n-1}}}{\sum_{h=1}^{N} \left(\sum_{i=1}^{N} u_{ic}^{m} \sum_{p=1}^{P} \delta_p D_p(i,h) \right)^{\frac{-1}{n-1}}} [3]
\end{aligned}
\tag{A.9}
$$

A.3 OBTENDO A OTIMIZAÇÃO DAS PONDERAÇÕES

Para encontrar a equação de otimização das ponderações das visões, precisamos utilizar as derivadas relacionadas às variáveis δ_p: $\frac{\partial L_{MFMMdd)}}{\partial \delta_p} = 0$, $\frac{\partial L_{MFMMdd)}}{\partial \gamma} = 0$

$$
\begin{aligned}
\frac{\partial L_{MFMMdd)}}{\delta_p} &= 0 \\
\sum_{i=1}^{N} \sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j) - \gamma \frac{\prod_{h=1}^{P} \delta_h}{\delta_p} &= 0 [derivada] \\
\sum_{i=1}^{N} \sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j) &= \gamma \frac{\prod_{h=1}^{P} \delta_h}{\delta_p} [5a] \\
\delta_p &= \gamma \frac{\prod_{h=1}^{P} \delta_h}{\sum_{i=1}^{N} \sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)} [5b]
\end{aligned}
\tag{A.10}
$$

$$\begin{aligned}
\frac{\partial L_{MFMMdd)}}{\partial \gamma} &= 0 \\
1 - \prod_{h=1}^{P} \delta_h &= 0[derivada] \\
\prod_{h=1}^{P} \delta_h &= 1[5a]
\end{aligned} \tag{A.11}$$

Podemos simplificar a equação A.10 substituindo o produtório pela equação A.11, ficando com:

$$\delta_p = \frac{\gamma}{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)} \tag{A.12}$$

Para encontrar o valor de γ, devemos substituir A.12 em A.11, resultando em:

$$\begin{aligned}
\prod_{h=1}^{P} \gamma \frac{1}{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)} &= 1 \\
\prod_{h=1}^{P} \gamma \left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right]^{-1} &= 1 \\
\gamma^P \prod_{h=1}^{P} \left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right]^{-1} &= 1[6] \\
\gamma^P \left\{\prod_{h=1}^{P} \left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right]\right\}^{-1} &= 1[7] \\
\gamma^P = \prod_{h=1}^{P} \left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right] &[5b] \\
\gamma = \left\{\prod_{h=1}^{P} \left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right]\right\}^{1/P} &[5c]
\end{aligned} \tag{A.13}$$

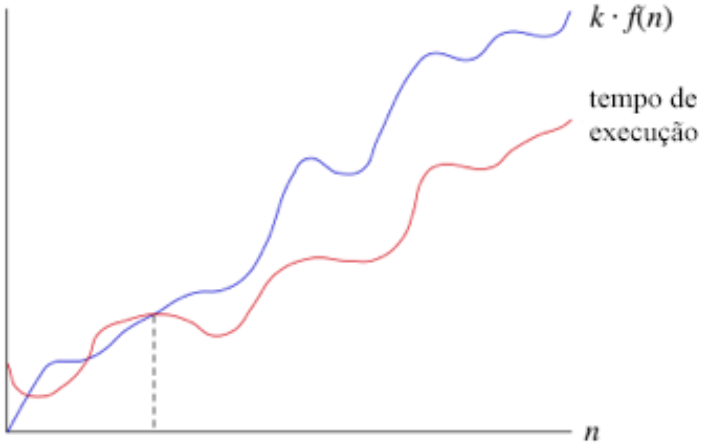
Finalmente, podemos substituir o valor de γ encontrado em A.13 em A.12. Obtendo:

$$\begin{aligned}
\delta_p &= \frac{\left\{\prod_{h=1}^{P}\left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right]\right\}^{1/P}}{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)} \\
\delta_p &= \frac{\prod_{h=1}^{P}\left[\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)\right]^{1/P}}{\sum_{i=1}^{N}\sum_{c=1}^{K} u_{ic}^{m} \sum_{j=1}^{N} v_{cj}^{n} D_p(i,j)} [7]
\end{aligned} \tag{A.14}$$

APÊNDICE B – NOTAÇÃO BIG-O

A Notação Big-O representa a relação entre o maior custo computacional de um sistema e as suas entradas, sendo um limite superior ao custo computacional multiplicado por uma constante [36]. A notação é dada como O(f(n)), sendo n as características da entrada (como o tamanho) e f(n) uma fórmula utiliza essas características como variáveis (como pode ser observado na figura 27)

Figura 27 – Visualização de uma notação Big-O



Fonte: Adaptada de [36]

Algumas propriedades da Notação são:

1. $O_{f_A(x)f_B(x)} = O_{f_A(x)} + O_{f_B(x)}$
2. $O_{f_A(x)+f_B(x)} = O_{f_A(x)} + O_{f_B(x)}$
3. $O_{\frac{1}{f(x)}} = O_{f(x)}$
4. $O_{\sum_{i=1}^{N} f(x_i)} = \sum_{i=1}^{N} O_{f(x_i)} = NO_{f(x)}$ (baseado da propriedade 1)
5. $O_{\prod_{i=1}^{N} f(x_i)} = \sum_{i=1}^{N} O_{f(x_i)} = NO_{f(x)}$ (baseado da propriedade 2)
6. A × O(B) = O(AB)
7. $c_1O(c_2A + c_3) = O(A)$, sendo c_1, c_2 e c_3 constantes
8. O(A)O(B) = O(AB)
9. O(A) + O(B) = O(A + B)
10. O(A + B) = O(A), se $A \gg B$