LUCAS FERNANDES BRUNIALTI

Fatoração de matrizes no problema de coagrupamento com sobreposição de colunas

São Paulo

2016

LUCAS FERNANDES BRUNIALTI

Fatoração de matrizes no problema de coagrupamento com sobreposição de colunas

Versão original

Dissertação apresentada à Escola de Artes, Ciências e Humanidades da Universidade de São Paulo para obtenção do título de Mestre em Ciências pelo Programa de Pós-graduação em Sistemas de Informação.

Área de concentração: Metodologia e Técnicas da Computação

Orientador: Profa. Dra. Sarajane Marques Peres

São Paulo

2016

CATALOGAÇÃO-NA-PUBLICAÇÃO
(Universidade de São Paulo. Escola de Artes, Ciências e Humanidades. Biblioteca)

Brunialti, Lucas Fernandes
Fatoração de matrizes no problema de coagrupamento com sobreposição de colunas / Lucas Fernandes Brunialti ; orientadora, Sarajane Marques Peres. – São Paulo, 2016.
122 f. : il

Dissertação (Mestrado em Ciências) - Programa de Pós-Graduação em Sistemas de Informação, Escola de Artes, Ciências e Humanidades, Universidade de São Paulo.
Versão original

1. Aprendizado computacional. 2. Mineração de dados. I. Peres, Sarajane Marques, orient. II. Título

CDD 22.ed.– 006.3

Dissertação de autoria de Lucas Fernandes Brunialti, sob o título **Fatoração de matrizes no problema de coagrupamento com sobreposição de colunas**, apresentada à Escola de Artes, Ciências e Humanidades da Universidade de São Paulo, para obtenção do título de Mestre em Ciências pelo Programa de Pós-graduação em Sistemas de Informação, na área de concentração Sistemas de Informação, aprovada em ___ de _____ de _____ pela comissão julgadora constituída pelos doutores:

Prof. Dr. _____
Presidente
Instituição: _____

Prof. Dr. _____
Instituição: _____

Prof. Dr. _____
Instituição: _____

À minha noiva Beatriz, aos meus pais Sandra e Ronaldo, aos meus orientadores Sarajane e Valdinei, e aos amigos e empresas que me apoiaram nesses três anos

Resumo

BRUNIALTI, Lucas Fernandes. **Fatoração de matrizes no problema de coagrupamento com sobreposição de colunas**. 2016. 122 f. Dissertação (Mestrado em Ciências) – Escola de Artes, Ciências e Humanidades, Universidade de São Paulo, São Paulo, 2016.

Coagrupamento é uma estratégia para análise de dados capaz de encontrar grupos de dados, então denomidados cogrupos, que são formados considerando subconjuntos diferentes das características descritivas dos dados. Contextos de aplicação caracterizados por apresentar subjetividade, como mineração de texto, são candidatos a serem submetidos à estratégia de coagrupamento; a flexibilidade em associar textos de acordo com características parciais representa um tratamento adequado à tal subjetividade. Um método para implementação de coagrupamento capaz de lidar com esse tipo de dados é a fatoração de matrizes. Nesta dissertação de mestrado são propostas duas estratégias para coagrupamento baseadas em fatoração de matrizes não-negativas, capazes de encontrar cogrupos organizados com sobreposição de colunas em uma matriz de valores reais positivos. As estratégias são apresentadas em termos de suas definições formais e seus algoritmos para implementação. Resultados experimentais quantitativos e qualitativos são fornecidos a partir de problemas baseados em conjuntos de dados sintéticos e em conjuntos de dados reais, sendo esses últimos contextualizados na área de mineração de texto. Os resultados são analisados em termos de quantização do espaço e capacidade de reconstrução, capacidade de agrupamento utilizando as métricas Índice de Rand e Informação Mútua Normalizada e geração de informação (interpretabilidade dos modelos). Os resultados confirmam a hipótese de que as estratégias propostas são capazes de descobrir cogrupos com sobreposição de forma natural, e que tal organização de cogrupos fornece informação detalhada, e portanto de valor diferenciado, para as áreas de análise de agrupamento e mineração de texto.

Palavras-chaves: Coagrupamento. Fatoração de Matrizes Não-negativas. Análise de Agrupamento. Mineração de Texto.

Abstract

BRUNIALTI, Lucas Fernandes. **Matrix factorization for overlapping columns coclustering**. 2016. 122 p. Dissertation (Master of Science) – School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, DefenseYear.

Coclustering is a data analysis strategy which is able to discover data clusters, known as coclusters. This technique allows data to be clustered based on different subsets defined by data descriptive features. Application contexts characterized by subjectivity, such as text mining, are applicable candidates for applying coclustering strategy due to the flexibility to associate documents according to partial features. The coclustering method can be implemented by means of matrix factorization, which is suitable to handle this type of data. In this thesis two strategies are proposed in non-negative matrix factorization for coclustering. These strategies are able to find column overlapping coclusters in a given dataset of positive data and are presented in terms of their formal definitions as well as their algorithms' implementation. Quantitative and qualitative experimental results are presented through applying synthetic datasets and real datasets contextualized in text mining. This is accomplished by analyzing them in terms of space quantization, clustering capabilities and generated information (interpretability of models). The well known external metrics Rand Index and Normalized Mutual Information are used to achieve the analysis of clustering capabilities. Results confirm the hypothesis that the proposed strategies are able to discover overlapping coclusters naturally. Moreover, these coclusters produced by the new algorithms provide detailed information and are thus valuable for future research in cluster analysis and text mining.

Keywords: Coclustering. Non-negative Matrix Factorization. Cluster Analysis. Text Mining.

Lista de figuras

Lista de algoritmos

Lista de tabelas

Sumário

[1] De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

1 Introdução

Segundo Jain, Murty e Flynn (1999), a análise de agrupamento pode ser vista como uma tarefa exploratória que tem o objetivo de organizar uma coleção de dados em um conjunto de grupos, segundo a similaridade ou dissimilaridade existente entre esses dados. Tradicionalmente, os métodos usados para análise de agrupamento são desenvolvidos para minimizar a similaridade intragrupos e maximizar a similaridade intergrupos; e precisam encontrar uma organização "natural" em grupos que acomode cada dado do conjunto sob análise em um grupo específico.

Estratégias de diferentes naturezas – particional, hierárquica, baseada em densidade, etc (HAN; KAMBER; PEI, 2011; XU; WUNSCH, 2005) – podem ser usadas para alcançar o objetivo da análise de agrupamento, e cada uma delas possui características que as fazem mais ou menos suscetíveis para conjuntos de dados de diferentes naturezas. Ainda, sob o contexto clássico da tarefa de agrupamento, os métodos precisam lidar com a similaridade entre os dados tomando como base a comparação de todas as suas características descritivas e, de alguma forma, precisam ser capazes de descobrir quais características de fato tornam dados em um grupo de dados mais similares entre si.

Ao longo do tempo, pesquisadores da área de análise de agrupamento vêm propondo flexibilizações na definição da tarefa de agrupamento de forma a adequá-la a contextos mais realísticos, nos quais a organização natural dos dados em um conjunto de dados não pressupõe restrições como a pertinência de um dado a um único grupo ou a possibilidade de agrupar dados de acordo com similaridades em subconjuntos de atributos descritivos (BEZDEK, 1981; HAN; KAMBER; PEI, 2011; ROCHA et al., 2012). Essa forma de tratar a tarefa de agrupamento permite melhorias no processo de descoberta de agrupamentos sob dois aspectos: facilita o trabalho do método que busca os grupos, pois flexibiliza a maneira como os atributos descritivos dos dados ou a pertinência do dado aos grupos influencia o processo de agrupamento; fornece um conjunto de informações diferenciado que permite que análises mais refinadas sejam realizadas quando da interpretação dos grupos apresentados como resultado.

Esse diferencial pode ser especialmente útil quando o contexto da aplicação da análise de agrupamento apresenta alguma subjetividade em termos de interpretação de resultados, um fato bastante comum em tarefas de mineração de texto, por exemplo. Considere um contexto de uma aplicação de mineração de texto em que o objetivo é

encontrar notícias similares à uma notícia fornecida como entrada, seja para recomendação, aumentando o engajamento de usuários em um portal de notícias, ou até mesmo para descoberta de conhecimento, a fim de ajudar na tomada de decisões de negócio nesse portal de notícias, por exemplo. A análise de agrupamento com base na similaridade que considera todos os atributos (palavras), em um conjunto de notícias que representam os temas esportes (colunas #1, #2 e #3 na figura 1a) e música (colunas #4, #5 e #6 na figura 1a), seria capaz de particionar as notícias em dois grupos, que representariam esses dois temas. Embora essas recomendações pareçam ser ideais, e sob algum aspecto de análise elas são, é factível assumir que essa análise de agrupamento é prática, mas talvez menos útil e interessante do que poderia ser.

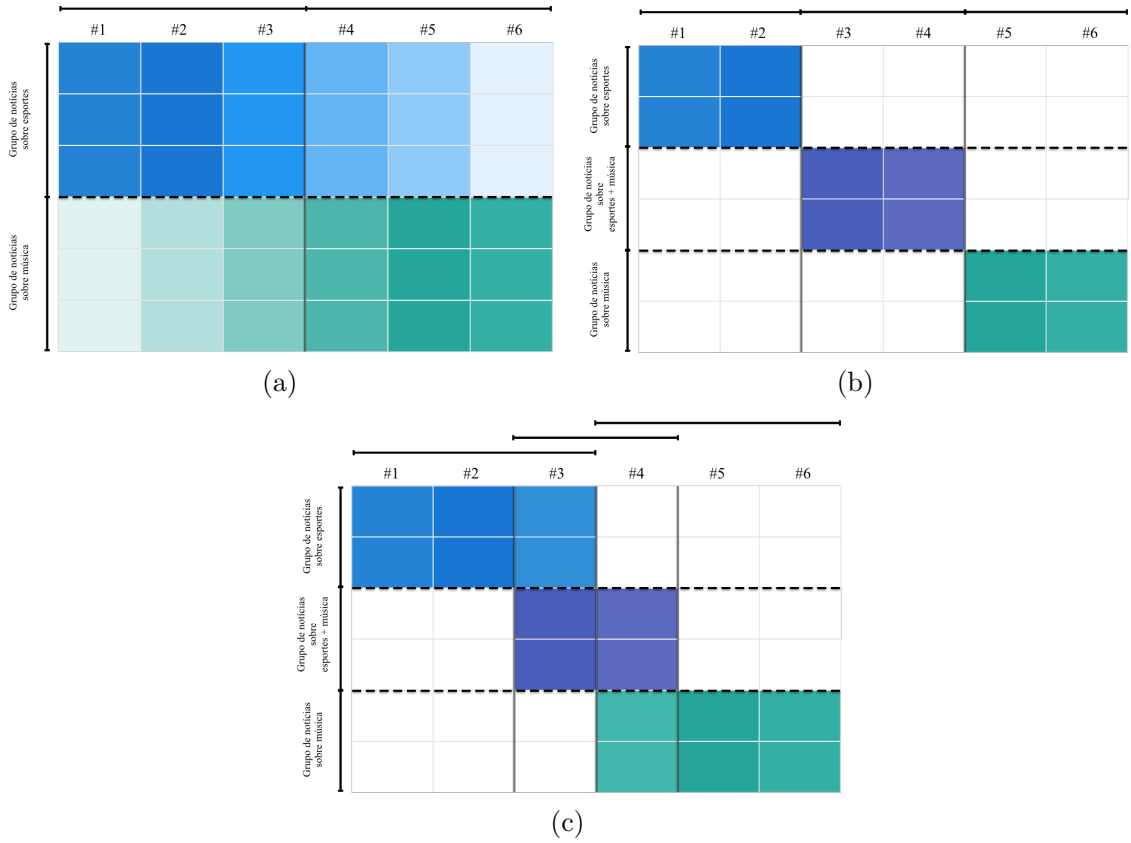
Uma possibilidade de melhoria dessa aplicação hipotética seria usar um algoritmo de análise de agrupamento que permitisse descobrir uma organização de grupos de notícias baseada em similaridades parciais ou baseada em partes (FRANCA, 2010; HO, 2008). Assim, seria possível encontrar grupos que não são encontrados quando se considera todas as características na análise de agrupamento.

Considerando tal possibilidade, a aplicação seria dotada da capacidade de perceber, por exemplo, que algumas notícias podem trazer conteúdo referente a diferentes contextos se forem analisadas apenas sob determinados aspectos. Nesse caso, os grupos formados durante a análise de agrupamento seriam capazes de refletir a diversidade de contextos abordados em uma notícia, fazendo-a pertencer a diferentes grupos, por diferentes motivos. Por exemplo, é sabido que eventos de beisebol – o *superbowl* – possuem uma abertura cultural na qual grandes artistas da música fazem apresentações; ou eventos de esportes radicais, como tirolesa, acontecem em eventos de música contemporâneos – *rock in rio*. Tais notícias deveriam aparecer em grupos caracterizados por notícias referentes à esportes, notícias referentes à música, ou notícias referentes à esporte e música (Figuras 1b e 1c).

Nas figuras 1b e 1c é introduzido graficamente o conceito de coagrupamento. Nesse contexto, o problema de mineração de texto é modelado como o problema de encontrar uma organização dos textos em grupos considerando similaridades parciais. Assim, um texto pode pertencer a um ou mais cogrupos, a depender dos atributos descritivos sendo considerados.

Portanto, nessa aplicação, uma análise de agrupamento com maior capacidade de extração de informação e interpretabilidade, consideraria que o grupo de notícias referentes

Figura 1 – Representação de uma aplicação de mineração de dados implementada a partir de análise de agrupamento com similaridade total (a) e similaridade parcial (b,c). Os grupos são diferenciados por cores.



Fonte: Lucas Fernandes Brunialti, 2016

aos assuntos esportes e música tem sobreposição nas palavras que formam o grupo de notícias sobre música e o grupo de notícias sobre esportes (Figura 1c).

A nomenclatura coagrupamento deriva da estratégia de análise de dados executada durante o processo de descoberta de grupos. Nesse caso, tanto os dados (linhas) quanto os seus atributos descritivos (colunas) são mutuamente submetidos a uma análise de similaridade, e portanto, grupos de dados (linhas) são estabelecidos com respeito a grupos de atributos (colunas), e grupos de atributos (colunas) também são estabelecidos com respeito a grupos de dados (linhas).

A associação da análise de coagrupamentos a mineração de texto é interessante pois essa tarefa constitui-se como um problema no qual é preciso lidar com a necessidade de apresentação de resultados que gerem informações passíveis de serem interpretadas. Esse problema pode ser bem resolvido com a estratégia de coagrupamento pois os grupos de atributos que são gerados por ela podem revelar informação antes escondida nos

dados (TJHI; CHEN, 2009), e que em um processo de agrupamento tradicional não poderiam ser, pelo menos diretamente, descobertas.

Dentre os diferentes métodos existentes na literatura referentes à implementação de análise de coagrupamento (FRANCA, 2010; MIRKIN, 1996; MADEIRA; OLIVEIRA, 2004), métodos que usam fatoração de matrizes não negativas (LEE; SEUNG, 2000; LEE; SEUNG, 1999) têm sido vistos como uma boa alternativa a ser aplicada no contexto de mineração de texto, dado que esses métodos têm vantagens para lidar com dados representados como matrizes positivas (XU; LIU; GONG, 2003; SHAHNAZ et al., 2006; YOO; CHOI, 2010).

Este trabalho explora o uso dos métodos de fatoração de matrizes não-negativas no contexto de coagrupamento, com atenção especial ao tratamento natural de cogrupos com sobreposição de colunas. Como forma de ilustrar a aplicabilidade das soluções propostas, o contexto de mineração de texto é analisado sob a ótica de coagrupamento.

1.1 Definição do problema

Para definição do problema é necessário o entendimento da tarefa de coagrupamento, e como a tarefa de coagrupamento se conecta com fatoração de matrizes.

A estratégia de coagrupamento pode ser apresentada como o processo de agrupamento simultâneo de linhas e colunas em uma matriz de dados, de forma que seja possível encontrar cogrupos nos quais um grupo de objetos (linhas) associado a um grupo de atributos (colunas) diz respeito a objetos que são similares entre si considerando este grupo de atributos (colunas), formando então, um cogrupo. Com maior formalidade, dada uma matriz X com n linhas e m colunas, a tarefa de coagrupamento pode ser vista como encontrar k grupos de linhas de X, denotados pelos conjuntos que contém as linhas da matriz de dados $\mathcal{K}_p, \forall p \in \{1, \ldots, k\}$, e l grupos de colunas de X, denotados pelos conjuntos que contém as colunas de atributos $\mathcal{L}_q, \forall q \in \{1, \ldots, l\}$.

Fatorar uma matriz consiste em encontrar duas, ou mais, novas matrizes que, ao serem combinadas, reconstroem a matriz original. Considerando a matriz X, a sua fatoração em duas novas matrizes consiste em encontrar duas matrizes, U com n linhas e k colunas e C com k linhas e m colunas, tal que $X \approx UC$. Se k é escolhido tal que seja muito menor do que n e m, então é dito que U e C são representações compactas de X. Se a matriz X e as suas decomposições são não-negativas, tem-se o caso de Fatoração de Matrizes Não-negativas (NMF - *Non-negative Matrix Factorization*) (LEE; SEUNG, 2000),

que pode ser interpretado como um método de agrupamento quando faz-se a análise sobre a matriz U, sendo k o número de grupos de linhas.

Se três matrizes são geradas na fatoração, U com n linhas e k colunas, S com k linhas e l colunas, e V com m linhas e l colunas, fazendo a aproximação $X \approx USV^T$, e se k e l são escolhidos tal que sejam menores que n e m, respectivamente, então é dito que U, S e V são representações compactas de X e imbutem a noção de k grupos de linhas e l grupos de colunas. A interpretação de S pode incluir uma noção de pesos que relacionam grupos de linhas e grupos de colunas, de modo que o número de grupos de linhas (k) pode ser diferente do número de grupo de colunas (l).

Desta maneira, o problema de coagrupamento pode ser modelado de tal forma que a fatoração de matrizes é capaz de fornecer uma aproximação da organização em cogrupos presente no conjunto de dados sob análise.

O problema deste trabalho é propor soluções algorítmicas que são capazes de solucionar a tarefa de coagrupamento no qual cogrupos de colunas têm intersecção (sobreposição): $\mathcal{L}_q \cap \mathcal{L}_{q'} \neq \emptyset$ para $q \neq q'$. Assim é possível que uma coluna pertença à dois ou mais grupos de colunas.

1.2 Hipótese

Fatorar matrizes considerando a decomposição da matriz original em uma matriz U, uma matriz S e um conjunto de matrizes V, isto é $X \approx USV_{(1)}^T \dots V_{(k)}^T$, tal que seja possível encontrar cogrupos, possibilita que o processo de busca seja beneficiado pela flexibilidade de ajuste de U e o conjunto de matrizes V, sendo mais adequado do que o processo tradicional de fatoração de matrizes, exposto na seção 1.1, para o tratamento da tarefa de coagrupamento com sobreposição de colunas. A adequação deste processo manterá informações da matriz original a fim de permitir a sua reconstrução, propiciará um resultado mais adequado em termos de agrupamento de linhas seguindo a avaliação clássica de quantização, e possibilitará a extração de informações diferenciadas a partir da análise da fatoração gerada, agregando valor à solução de um problema de mineração de texto.

1.3 Objetivos

O objetivo geral deste trabalho foi o desenvolvimento de novas estratégias de coagrupamento baseadas em fatoração de matrizes, aqui então nomeadas *OvNMTF* e *BinOvNMTF*, capazes de lidar com a existência de cogrupos de colunas com sobreposição, de maneira mais adequada que os algoritmos atualmente apresentados na literatura. A adequabilidade das estratégias propostas foi avaliada em termos de: capacidade de quantização do espaço e capacidade de reconstrução da matriz original, capacidade de agrupamento e capacidade de possibilitar a extração de informação.

Como objetivos específicos, este trabalho:

- apresentou a derivação formal das regras de atualização de matrizes usadas nas estratégias propostas (*OvNMTF* e *BinOvNMTF*);
- apresentou a aplicação das novas estratégias em ambientes controlados (bases de dados sintéticas) e em um contexto de aplicação real (análise de dado textuais);
- apresentou um novo conjunto de dados textual, em língua portuguesa, referente ao contexto de notícias.

O atendimento aos objetivos delineados permitiu a esse trabalho:

- aprimorar a área de coagrupamento baseado em algoritmos de fatoração de matrizes, de forma a contribuir com a pesquisa em análise de agrupamento;
- demonstrar que as estratégias de coagrupamento propostas têm potencial de revelar informações úteis provenientes da sobreposição natural dos atributos descritivos de um dado real, em especial para aplicações de mineração de texto.

1.4 Método

A análise exploratória da literatura especilizada foi escolhida como estratégia para a aquisição de conhecimento sobre a área de coagrupamento e fatoração de matrizes aplicada à resolução da tarefa de coagrupamento.

A partir da análise exploratória, os algoritmos *BVD* (LONG; ZHANG; YU, 2005), *ONMTF* (DING et al., 2006; YOO; CHOI, 2010) e *FNMTF* (WANG et al., 2011) foram estudados em profundidade com o fim de verificar a sua adequabilidade para tratar a

análise de coagrupamento sobre dados sintéticos e textuais, considerando a sobreposição de colunas. Nesse estudo verificou-se a possibilidade da proposição de melhorias no tratamento do problema, e então, os algoritmos *OvNMTF* e *BinOvNMTF* foram criados. Para cada um deles, o problema formal de otimização foi definido e uma derivação formal foi realizada usando cálculo em matrizes com o intuito de propor uma solução algorítmica para tais problemas.

A fim de permitir a validação das estratégias propostas e, portanto, a verificação da hipótese delineada neste trabalho, fez-se necessária a definição de: (a) um ambiente de teste controlado, representado por uma coleção de conjuntos de dados sintéticos, contendo em cada um dos conjuntos situações diferentes referentes às estruturas de coagrupamento; e (b) um contexto para realização de uma experimentação com dados reais.

Para tal experimentação foi escolhido usar o conteúdo referente à notícias publicadas no portal iG[1] e trabalhos acadêmicos publicadas na conferência *NIPS*. iG é um portal de notícias brasileiro muito conhecido, com um grande volume de notícias categorizadas em canais, os quais representam os assuntos dessas notícias. Essas características conferem liberdade para a configuração de experimentos de diferentes naturezas, como experimentos considerando determinadas categorias de notícias, tipos de notícias ou datas de publicação das notícias.

A partir do conteúdo de notícias do portal iG foi construído um corpus de dados textuais, categorizados de acordo com as categorias já usadas no referido portal. Todo o conteúdo do corpus passou por rotinas de pré-processamento comuns na área de mineração de texto: *tokenização* (LOPS; GEMMIS; SEMERARO, 2011), filtragem de *stopwords* (LOPS; GEMMIS; SEMERARO, 2011), representação da relação "termos × documentos" usando estratégias de frequência de termos, como *tf-idf* (SALTON; WONG; YANG, 1975).

Os resultados da aplicação das estratégias de coagrupamento foram validados utilizando:

- inspeção visual e erro de quantização para análise da capacidade de reconstrução;
- técnicas de avaliação externa representadas pelo índice de Rand e pela informação mútua normalizada, para análise da capacidade de agrupamento;
- análise empírica por meio de experimentos qualitativos para avaliação da capacidade de extração de informação e interpretabilidade dos modelos.

[1] http://ig.com.br/

1.5 Organização do documento

Esta dissertação é composta por 6 capítulos incluindo esta introdução.

No capítulo 2 são apresentados os principais conceitos referentes à teoria de matrizes, agrupamento e coagrupamento. O objetivo deste capítulo é fornecer diretrizes para entendimento dos assuntos tratados nos capítulos posteriores e indicar literatura na qual podem ser encontradas informações mais detalhadas sobre tais assuntos.

Estratégias de fatoração de matrizes aplicadas à resolução da tarefa de coagrupamento são discutidas no capítulo 3. A discussão é apresenta em termos da definição de problemas e apresentação de soluções algorítmicas para os problemas. Algumas discussões sobre os problemas e algoritmos são apresentadas sempre que um conceito é mais relevante para o entendimento da proposta deste trabalho. Detalhes sobre cada um dos problemas e algoritmos são encontrados na literatura sugerida nesse capítulo.

A principal contribuição deste trabalho, estratégias algorítmicas baseadas em fatoração de matrizes para encontrar cogrupos com sobreposição de colunas, é apresentada em detalhes no capítulo 4. As definições de problemas, regras de derivação e implementações algorítimicas foram formuladas originalmente neste estudo.

Os experimentos a cerca dos algoritmos apresentados nos outros capítulos são apresentados no capítulo 5. Os experimentos apresentados no capítulo 5 ilustram a aplicabilidade e adequabilidade das estratégias apresentadas nos capítulos 3 e 4, com destaque para as vantagens e limitações das estratégias propostas neste trabalho de mestrado. As análises apresentadas neste capítulo são de natureza quantitativa e qualitativa.

E, por fim, as conclusões deste trabalho são apresentadas no capítulo 6, destacando as contribuições aqui elaboradas e as questões em aberto na área, sendo algumas delas decorrentes de novas hipóteses, cuja formulação se tornou possível por conta das proposições realizadas no presente trabalho.

2 Conceitos Fundamentais

Este capítulo introduz os principais conceitos e definições necessárias para suportar a leitura dos demais capítulos deste trabalho. São apresentados os conceitos e notações referentes à teoria de matrizes e álgebra linear (seção 2.1), em seguida são mostrados técnicas e conceitos referentes à agrupamento (seção 2.2), e por fim, os conceitos referentes à coagrupamento (seção 2.3).

2.1 Teoria de Matrizes e Álgebra Linear

Considere $\mathbb{R}$ o conjunto de todos os números reais, então, uma matriz no espaço $\mathbb{R}^{n\times m}$ é definida como uma tabela com n linhas e m colunas, na forma:

$$A \in \mathbb{R}^{n\times m} \Leftrightarrow A = \begin{bmatrix} a_{11} & \dots & a_{m1} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nm} \end{bmatrix}$$

sendo $a_{ij} = (A)_{ij} \in \mathbb{R}, \forall i, j$ o elemento da i-ésima linha e j-ésima coluna da matriz A, e os índices $i \in \{1, \dots, n\}$ e $j \in \{1, \dots, m\}$ para indexar as linhas e colunas dessa matriz, respectivamente. Na notação usada neste trabalho, letras maiúsculas representam matrizes (ex.: A), e letras minúsculas com duas letras subscritas representam valores na matriz (ex.: a_{ij} representa um valor da matriz A). Também é possível definir essa mesma matriz em $\mathbb{R}_{+}^{n\times m}$, por meio da determinação da restrição $a_{ij} > 0, \forall i, j$.

De forma semelhante define-se uma notação para vetores no espaço $\mathbb{R}^n$, na forma:

$$\mathbf{a} \in \mathbb{R}^n \Leftrightarrow \mathbf{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}$$

sendo $a_i \in \mathbb{R}, \forall i$ o i-ésimo elemento do vetor. Ainda neste trabalho é usada a notação de vetores para representação das linhas e colunas de uma matriz, da seguinte forma:

$$A = \begin{bmatrix} — & \mathbf{a}_{1\cdot} & — \\ & \vdots & \\ — & \mathbf{a}_{n\cdot} & — \end{bmatrix} = \begin{bmatrix} | & & | \\ \mathbf{a}_{\cdot 1} & \dots & \mathbf{a}_{\cdot m} \\ | & & | \end{bmatrix}$$

sendo $\mathbf{a}_{i\cdot}$ e $\mathbf{a}_{\cdot j}, \forall i, j$ os vetores que representam as linhas e colunas da matriz A, respectivamente.

Neste trabalho as seguintes operações em matrizes são usadas (GOLUB; LOAN, 1996):

- Transposição de matrizes: $(A^T)_{ij} = (A)_{ji}, \forall i, j,\ A^T \in \mathbb{R}^{m \times n},\ A \in \mathbb{R}^{n \times m}$.
- Produto de Hadamard: $C = A \odot B$, onde $c_{ij} = a_{ij} b_{ij}, \forall i, j$, sendo $A, B, C \in \mathbb{R}^{n \times m}$.
- Divisão de matrizes ponto-a-ponto: $C = \frac{A}{B}$, onde $c_{ij} = \frac{a_{ij}}{b_{ij}}$, $\forall i, j$, sendo $A, B, C \in \mathbb{R}^{n \times m}$.
- Produto de matrizes: $C = AB$, onde $c_{ij} = \sum_{p=1}^{k} a_{ip} b_{pj}$, sendo $C \in \mathbb{R}^{n \times m}, A \in \mathbb{R}^{n \times k}, B \in \mathbb{R}^{k \times m}$.
- Matriz inversa: $AB = BA = I \Leftrightarrow B = A^{-1}$, sendo $A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times n}$, e I a matriz identidade.

Para definição de problemas em fatoração, também é usado neste trabalho um operador: traço da matriz. Considerando uma matriz quadrada $A \in \mathbb{R}^{n \times n}$ (MAGNUS; NEUDECKER, 1999), o traço da matriz é definido da seguinte forma:

$$tr(A) = \sum_{i=1}^{n} a_{ii}$$

Algumas igualdades são possíveis a partir da aplicação deste operador. Considerando as matrizes A, B, C, em um espaço que torne as multiplicações possíveis, definem-se as seguintes igualdades (MAGNUS; NEUDECKER, 1999):

- $tr(A^T) = tr(A)$.
- $tr(A^T B) = tr(B^T A)$.
- $tr(AB) = tr(BA)$.
- $tr(ABC) = tr(CAB) = tr(BCA)$ (propriedade circular).

Note que a matriz resultante das operações dentro do traço é sempre quadrada.

Definido o operador de traço, é possível definir o operador para o produto interno entre dois vetores $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ (BOYD; VANDENBERGHE, 2004), da seguinte forma:

$$\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{n} a_i b_i = \begin{bmatrix} a_1 & \dots & a_n \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} = \mathbf{a}^T \mathbf{b}$$

Da mesma forma define-se o produto interno entre duas matrizes $A \in \mathbb{R}^{n \times m}$ e $B \in \mathbb{R}^{n \times m}$, mostrando como os operadores de traço e produto interno se relacionam (BOYD; VANDENBERGHE, 2004):

$$\langle \mathbf{A}, \mathbf{B} \rangle = \sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij} b_{ij} = tr(A^T B)$$

2.1.1 Norma em matrizes

A norma é uma função que recebe como parâmetro de entrada um vetor ou matriz nos espaços $\mathbb{R}^n$ ou $\mathbb{R}^{n \times m}$, respectivamente, e gera um valor real, o qual representa a magnitude do vetor ou matriz. Uma das normas mais conhecidas é a norma de Frobenius, usada nas definições dos problemas de fatoração deste trabalho. A norma de Frobenius é definida da seguinte forma para um vetor $\mathbf{a} \in \mathbb{R}^n$ (BOYD; VANDENBERGHE, 2004):

$$\|\mathbf{a}\|_F = \sqrt{a_1^2 + \cdots + a_n^2} = \sqrt{\mathbf{a}^T \mathbf{a}} = \sqrt{\langle \mathbf{a}, \mathbf{a} \rangle}$$

Essa norma também é chamada de norma Euclidiana e norma-L_2. Neste caso, a norma é definida da seguinte forma para uma matriz $A \in \mathbb{R}^{n \times m}$ (BOYD; VANDENBERGHE, 2004):

$$\|A\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{m} |a_{ij}^2|} = \sqrt{tr(A^T A)}$$

Ainda, considerando dois vetores $\mathbf{a}$ e $\mathbf{b}$, é possível dizer que estes são ortogonais se o produto interno entre eles for 0, ou seja, não há projeção de $\mathbf{a}$ em $\mathbf{b}$, e vice-versa (MAGNUS; NEUDECKER, 1999):

$$\langle \mathbf{a}, \mathbf{b} \rangle = 0$$

Os vetores $\mathbf{a}$ e $\mathbf{b}$ também podem ser ditos ortonormais se, além da condição anterior, a suas normas forem igual à 1. A mesma definição pode ser extendida para matrizes, então, uma matriz é dita ortonormal se todos os seus vetores forem ortonormais (MAGNUS; NEUDECKER, 1999):

$$A^T A = I$$

2.1.2 Cálculo em matrizes

Cálculo em matrizes define uma notação que suportou a derivação de soluções para os problemas de otimização, baseados em fatoração de matrizes, propostos neste trabalho. Extendendo a definição de derivadas, tradicionalmente conhecida, é possível obter relações para derivação de funções em relação a matrizes (MAGNUS; NEUDECKER, 1999). Considere a seguinte notação para derivada de uma função de traço em relação à uma matriz A, sendo $F(A)$ uma função diferenciável em todos os elementos de A (PETERSEN; PEDERSEN, 2012):

$$\frac{\partial tr(F(A))}{\partial A} = f(A)^T$$

em que $f(\cdot)$ é a derivada escalar de $F(\cdot)$.

Diante desta definição, algumas relações podem ser mostradas (PETERSEN; PEDERSEN, 2012). Sendo A, B, C e D matrizes quaisquers definidas em um espaço que tornem as multiplicações possíveis, para cada caso:

- $\frac{\partial}{\partial A} tr(A) = I$ (matriz identidade).
- $\frac{\partial}{\partial A} tr(BA) = B^T$.
- $\frac{\partial}{\partial A} tr(BA^T) = \frac{\partial}{\partial A} tr(A^T B) = B$.
- $\frac{\partial}{\partial A} tr(BAC) = B^T C^T$.
- $\frac{\partial}{\partial A} tr(BA^T C) = CB$.
- $\frac{\partial}{\partial A} tr(BACA^T D) = B^T D^T AC^T + DBAC$.
- $\frac{\partial}{\partial A} tr(C^T A^T DAC) = C^T A^T DB + C^T A^T B^T D^T$.

2.2 Agrupamento

Classicamente, a tarefa de agrupamento de dados é definida como o processo de agrupar dados de acordo com a similaridade existente entre eles. Sendo assim, os dados alocados a um mesmo grupo são aqueles mais similares entre si, enquanto que os dados alocados em grupos diferentes são dissimilares entre si. Nesse contexto, similaridades, e dissimilaridades, são geralmente calculadas por meio de medidas de distâncias (HAN; KAMBER; PEI, 2011). A tarefa de agrupamento pode também ser vista como um processo de particionamento das linhas de uma matriz que representa um conjunto de dados, sendo que esse conjunto de dados contém dados e características desses dados (HAN; KAMBER; PEI, 2011). Ainda, os grupos podem ser vistos como uma forma de compressão de dados.

A tarefa de agrupamento pode ser resolvida a partir de técnicas provenientes da área de aprendizado de máquina, mais especificamente por um subconjunto de técnicas de aprendizado não-supervisionado, nas quais tem-se mecanismos para realizar análise de dados para os quais não se tem informação a priori sobre como os dados estão organizados. A resolução da tarefa de agrupamento pode ser útil em diversas áreas do conhecimento, tais como Mineração de Dados, Estatística, Tomada de Decisão, etc.

Formalmente, os algoritmos de agrupamento implementados sob estratégias de particionamento têm como entrada uma matriz de dados $X \in \mathbb{R}^{n \times m}$, com n linhas (dados) e m colunas (características) que representam um conjunto de dados em algum domínio de

aplicação. Essa matriz é formada por um conjunto de vetores de linhas $\mathcal{N} = \{\mathbf{x}_{1\cdot}, \ldots, \mathbf{x}_{n\cdot}\}$. O objetivo é encontrar k partições de $\mathcal{N}$, denotadas por subconjuntos ordenados $\mathcal{K}_p \subseteq \mathcal{N}$, considerando o índice $p \in \{1, \ldots, k\}$. Então, o conjunto $\mathscr{K} = \{\mathcal{K}_1, \ldots, \mathcal{K}_k\}$ é justamente os grupos de linhas resultantes de um algoritmo que soluciona a tarefa de agrupamento. Note que este problema é NP-difícil, então faz sentido buscar por algoritmos que se baseiam em alguma heurística, já que não existe uma solução algorítmica em tempo polinomial (ALOISE et al., 2009).

2.2.1 Algoritmos para Agrupamento

O problema de agrupamento *k-means* é provavelmente um dos problemas mais estudados para agrupamento, e o algoritmo de *Lloyd*, também chamado algoritmo *k-means*, é um dos algoritmos mais famosos para solução do problema de agrupamento. Trata-se de um problema de otimização do erro de quantização, formalizado no problema 1, proposto com o objetivo de encontrar grupos em um conjunto de dados, solucionando a tarefa de agrupamento.

O objetivo do *k-means* é encontrar k representantes da matriz de dados X. Esses representantes, também chamados centróides ou médias, são organizados em uma matriz C com k linhas e m colunas, sendo que cada vetor de linha é um representante dos dados, por isso estes têm a mesma dimensão dos dados da matriz X. De forma direta é possível perceber que serão formados k grupos de dados.

Neste trabalho o problema *k-means* é mostrado sob uma visão de fatoração de matrizes, assim como em Ding, He e Simon (2005), pois tem-se como objetivo encontrar uma solução para aproximar X pela multiplicação de duas matrizes U e C: $X \approx UC$. Além disso, em Ding, He e Simon (2005) também é mostrada a relação de aproximação entre os algoritmos de agrupamento mais tradicionais, como o *k-means*, o *fuzzy k-means* e o problema de Fatoração de Matrizes Não-negativas (NMF - *Non-negative Matrix Factorization*), que apesar de ter sido desenvolvido como um método de redução de dimensionalidade, também é considerado um método de agrupamento.

Problema 1 (Problema de *k-means*)**.**

$$\begin{aligned} \mathcal{F}_1(U,C) &= \min_{U,C} \sum_{i=1}^{n}\sum_{p=1}^{k} u_{ip}\|\mathbf{x_{i\cdot}} - \mathbf{c_{p\cdot}}\|^2 = \min_{U,C} \quad \|X - UC\|_F^2 \\ & \textit{suj. a} \qquad\qquad\qquad\qquad U \in \Psi^{n\times k}, \\ & \qquad\qquad\qquad\qquad\quad C \in \mathbb{R}^{k\times m}, \\ & \qquad\qquad\qquad\qquad\quad \textstyle\sum_{p=1}^{k} u_{ip} = 1, \forall i \end{aligned}$$

em que $\Psi = \{0,1\}$ *e* $\|\cdot\|_F$ *denota a norma de Frobenius para matrizes. A restrição da soma de cada linha da matriz* U *ser igual à* 1*, garante que cada dado de* X *pertença a apenas um grupo, já que o objetivo é particionar os dados.*

É possível perceber que o problema apresentado compacta a matriz X em duas outras matrizes: a matriz U, de dimensão n por k que contém apenas valores binários, e fará a associação de cada linha de X com um dos k grupos; e a matriz C, de dimensão k por m, que será uma base para representação das linhas em X. Portanto, a compactação transformará os nm elementos originais em $n + km$ novos elementos, desconsiderando os zeros da matriz U.

A implementação para o problema 1 apresentado é descrito no algoritmo 1 (ROCHA et al., 2012; HAN; KAMBER; PEI, 2011; BOTTOU; BENGIO, 1995), o qual é baseado no algoritmo de EM (Esperança-Maximização - *Expectation-Maximization*) e tem sua convergência para um mínimo local demonstrada (veja em Bottou e Bengio (1995)). Nesse algoritmo considere os índices $i \in \{1,\ldots,n\}$ e $p = p' \in \{1,\ldots,k\}$, t o contador de iterações, $U^{(t)}$ e $C^{(t)}$ as matrizes U e C na iteração t, respectivamente, $\mathcal{U}(0,1) \in]0,1]$ uma função que gera valores de uma distribuição uniforme, e $\|\cdot\|^2$ é a norma de Frobenius para vetores.

Como condições para assumir a convergência, neste trabalho, considera-se a diferença do erro de aproximação em duas iterações consecutivas menor ou igual a um ϵ:

$$\left\|X - U^{(t)}C^{(t)}\right\|_F^2 - \left\|X - U^{(t+1)}C^{(t+1)}\right\|_F^2 \leq \epsilon$$

O algoritmo também pára caso a t-ésima iteração seja igual ao número máximo de iterações (t_{max}). Note que o particionamento é direto, analisando a matriz U.

Ainda, a atualização de C também pode ser derivada de forma matricial, expandindo a função apresentada no problema 1 e derivando em relação à C, usando as relações definidas na seção 2.1.2. Assim, obtem-se a seguinte formulação, que é equivalente à apresentada no algoritmo 1 (BAUCKHAGE, 2015):

$$C = (U^T U)^{-1} U^T X$$

Algoritmo 1 Algoritmo para solução do *k-means*

function K-MEANS(X, k, t_{max})
  Inicialize: $C^{(0)} \leftarrow \mathcal{U}(0,1)$ e $t \leftarrow 0$.
  while (não convergiu) e ($t \leq t_{max}$) **do**

$$(C^{(t+1)})_{p\cdot} \leftarrow \frac{\sum_{i=1}^{n} u_{ip}^{(t)} \mathbf{x}_{i\cdot}}{\sum_{i=1}^{n} u_{ip}^{(t)}}, \forall p$$

▷ Etapa de Esperança

$$(U^{(t+1)})_{ip} \leftarrow \begin{cases} 1 & p = \arg\min_{p' \in \{1,\ldots,k\}} \left\| \mathbf{x}_{i\cdot} - \mathbf{c}_{p'\cdot}^{(t+1)} \right\|^2 \\ 0 & \textit{caso contrário} \end{cases} \quad \forall i, p$$

▷ Etapa de Maximização

    $t \leftarrow t + 1$
  end while
  return $U^{(t)}, C^{(t)}$
end function

O algoritmo *k-means* tem complexidade de tempo $\mathcal{O}(t_{max}nmk)$. Isso mostra a escalabilidade do algoritmo para grandes bases de dados, ou seja, matrizes com n e/ou m grandes (HAN; KAMBER; PEI, 2011).

Outro problema semelhante ao *k-means* é o *fuzzy k-means*, o qual, ao invés de restringir a pertinência de cada dado a exclusivamente um grupo, permite que o dado pertença a diferentes grupos com diferentes graus de pertinência. Então, a matriz U será a matriz responsável por armazenar a pertinência de cada dado (ou linha) de X, a cada um dos k grupos.

O problema de otimização do erro de quantização, com a finalidade da resolução da tarefa de agrupamento (*fuzzy k-means*), é apresentado formalmente no problema 2 (BEZDEK, 1981; ROCHA et al., 2012; DING; HE; SIMON, 2005).

Problema 2 (Problema de *fuzzy k-means*)**.**

$$\begin{aligned} \mathcal{F}_2(U,C) \quad = \quad & \min_{U,C} \quad \sum_{i=1}^{n} \sum_{p=1}^{k} u_{ip}^{w} \|\mathbf{x_i}_{\cdot} - \mathbf{c_p}_{\cdot}\|^2 \\ & \textit{suj. a} \qquad U \in \mathbb{R}^{n \times k}, \\ & \qquad\quad C \in \mathbb{R}^{k \times m}, \\ & \qquad\quad \sum_{p=1}^{k} u_{ip} = 1, \forall i \end{aligned}$$

em que $w \in [1, 2, \ldots, \infty]$.

O expoente w é chamado de parâmetro de "fuzificação", e tem a função de controlar a relação de pertinência de um dado entre os k grupos, isto é, pode se dizer que o parâmetro

controla a ortogonalidade dos grupos, quanto maior o seu valor, menor a ortogonalidade entre grupos. Isso pode ser percebido, observando a análise de limite de w na função de atualização de U, realizada em Rocha et al. (2012).

Se $w = 2$ é possível modificar o problema 2 tal que seja possível observá-lo sob uma ótica de fatoração de matrizes, que tem o objetivo de aproximar X das matrizes U e C, assim como no *k-means*: $X \approx UC$. O problema 2 é apresentado com $w = 2$ para obtenção da ótica de fatoração de matrizes (DING; HE; SIMON, 2005):

$$\begin{aligned} \mathcal{F}_2^{w=2}(U,C) &= \min_{U,C} \sum_{i=1}^{n}\sum_{p=1}^{k} u_{ip}^2 \|\mathbf{x_i}. - \mathbf{c_p}.\|^2 = \min_{U,C} \quad \|X - UC\|_F^2 \\ &\text{suj. a} \qquad\qquad\qquad\qquad\qquad\qquad U \in \mathbb{R}^{n\times k}, \\ &\qquad\qquad\qquad\qquad\qquad\qquad\quad C \in \mathbb{R}^{k\times m}, \\ &\qquad\qquad\qquad\qquad\qquad\qquad\quad \textstyle\sum_{p=1}^{k} u_{ip} = 1, \forall i \end{aligned}$$

Da mesma forma que o *k-means*, o *fuzzy k-means* compacta os dados de X, sendo um mapeamento de nm elementos para $nk + km$ elementos. Então, teoricamente, o *fuzzy k-means* é capaz de preservar mais informações do que o *k-means*.

A implementação para o problema 2 é descrita no algoritmo 2 (ROCHA et al., 2012; DING; HE; SIMON, 2005; BEZDEK, 1981), o qual é baseado na combinação do algoritmo *k-means* com conceitos da teoria de conjuntos fuzzy e tem sua convergência para um mínimo local demonstrada via teoria de otimização não linear (veja em Bezdek (1981)). Nesse algoritmo considere os índices $i \in \{1,\ldots,n\}$ e $p, p' \in \{1,\ldots,k\}$, t o contador de iterações, $U^{(t)}$ e $C^{(t)}$ as matrizes U e C na iteração t, respectivamente, $\mathcal{U}(0,1) \in]0,1]$ uma função que gera valores de uma distribuição uniforme, e $\|\cdot\|^2$ é a norma de Frobenius para vetores. Ainda, a mesma condição de convergência aplicada ao algoritmo 1 pode ser aplicada para esse algoritmo.

Note que como U possui valores no domínio dos reais, não é possível obter as partições diretamente, sem um processo de pós-processamento. Um método de pós-processamento é o seguinte:

$$\mathcal{K}_p = \{x_{i\cdot} \mid i \in \{1,\ldots,n\} \text{ e } p = \arg\max_{p' \in \{1,\ldots,k\}} u_{ip'}\}, \forall p \in \{1,\ldots,k\}$$

Isso significa que uma linha i pertencerá a um grupo p (ou partição) se para todos os k grupos, o grau de pertinência u_{ip} for maior que todos os outros graus de pertinência para os outros grupos, presentes no vetor $\mathbf{u}_{i\cdot}$. Analisando a complexidade de tempo do algoritmo é possível chegar em: $\mathcal{O}(t_{max}nmk^2)$.

Algoritmo 2 Algoritmo para solução do *fuzzy k-means*

function FUZZY K-MEANS(X, k, t_{max})
  Inicialize: $C^{(0)} \leftarrow \mathcal{U}(0,1)$ e $t \leftarrow 0$.
  while (não convergiu) e ($t \leq t_{max}$) **do**

$$(U^{(t+1)})_{ip} \leftarrow \left[\sum_{p'=1}^{k} \left(\frac{\left\| \mathbf{x}_{i\cdot} - \mathbf{c}_{p\cdot}^{(t)} \right\|}{\left\| \mathbf{x}_{i\cdot} - \mathbf{c}_{p'\cdot}^{(t)} \right\|} \right)^{\frac{1}{w-1}} \right]^{-1}, \forall i, p$$

$$(C^{(t+1)})_{p\cdot} \leftarrow \frac{\sum_{i=1}^{n} u_{ip}^{w^{(t+1)}} \mathbf{x}_{i\cdot}}{\sum_{i=1}^{n} u_{ip}^{w^{(t+1)}}}, \forall p$$

    $t \leftarrow t + 1$
  end while
  return $U^{(t)}, C^{(t)}$
end function

Além dos algoritmos apresentados, existem outros que buscam resolver diferentes tipos de problemas dentro da área de agrupamento, como agrupamento por densidade, agrupamento hierárquico, agrupamento baseado em grade e agrupamento baseado em modelos (HAN; KAMBER; PEI, 2011).

2.2.2 Validação de Agrupamento

Para avaliar a qualidade de um agrupamento obtido a partir da execução de um algoritmo, e consequentemente, avaliar a adequabilidade dos parâmetros usados nesse algoritmo, a capacidade de agrupamento ou a estabilidade do processo por ele executado, é necessário fazer uso de métricas de validação (ou avaliação). Como neste trabalho são feitas comparações entre diversos algoritmos quanto às suas capacidades de agrupamento, os métodos e métricas estudados na área de validação de agrupamento precisam ser estudados, pois trazem recursos e ferramentas que tornam possíveis análises quantitativas, considerando os resultados de agrupamento desses algoritmos.

Halkidi, Batistakis e Vazirgiannis (2002a), Halkidi, Batistakis e Vazirgiannis (2002b) propõem a seguinte taxonomia para métricas de validação de agrupamento: *validação interna*, que utiliza apenas o resultado de um agrupamento gerado para estabelecer métricas de qualidade ou estabilidade, permitindo avaliar as soluções geradas; *validação externa*, tem como entrada para o cálculo das medidas dois tipos de informação: o particionamento real dos dados, e o resultado de um agrupamento gerado - essas informações são usadas

para verificar o quanto o agrupamento gerado corresponde ao particionamento real; e *validação relativa*, que analisa diferentes soluções geradas por um mesmo algoritmo diante de diferentes parametrizações e estabelece métricas para encontrar a melhor solução dentre as soluções disponíveis.

Neste trabalho, são utilizadas apenas métricas de validação externa para validar a capacidade de agrupamento dos diferentes algoritmos apresentados. Dentre as muitas métricas de validação externa presentes na literatura (HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2002a; HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2002b), foram escolhidas as métricas Índice de Rand (*RI - Rand Index*) (RAND, 1971) e Informação Mútua Normalizada (*NMI - Normalized Mutual Information*).

O RI gera valores entre 0 e 1, com 0 indicando que o particionamento de dados gerado por um algoritmo não concorda em nenhum par de elementos com o particionamento real, e 1 indicando que o particionamento gerado pelo mesmo algoritmo é exatamente igual ao particionamento real.

Considere que um algoritmo encontrou um particionamento denotado pelo conjunto $\widetilde{\mathscr{K}} = \{\widetilde{\mathcal{K}}_1, \ldots, \widetilde{\mathcal{K}}_{\widetilde{k}}\}$ com $\widetilde{k}$ grupos, sendo $\widetilde{\mathcal{K}}_p, \forall p \in \{1, \ldots, \widetilde{k}\}$ um subconjuto que contém elementos pertencentes ao p-ésimo grupo. Considere também, o particionamento real: $\mathscr{K} = \{\mathcal{K}_1, \ldots, \mathcal{K}_k\}$ com k grupos. Então, determine as seguintes quantidades, sendo os dados as linhas da matriz de dados X denotado pelo conjunto $\mathcal{N}$:

- a: quantidade de pares de dados em $\mathcal{N}$ que estão nos mesmos subconjuntos em $\mathscr{K}$ e também nos mesmos subconjuntos em $\widetilde{\mathscr{K}}$;
- b: quantidade de pares de dados em $\mathcal{N}$ que estão em diferentes subconjuntos em $\mathscr{K}$ e também estão em diferentes subconjuntos em $\widetilde{\mathscr{K}}$;
- c: quantidade de pares de dados em $\mathcal{N}$ que estão nos mesmos subconjuntos em $\mathscr{K}$ e em diferentes subconjuntos em $\widetilde{\mathscr{K}}$;
- d: quantidade de pares de dados em $\mathcal{N}$ que estão em diferentes subconjuntos em $\mathscr{K}$ e estão nos mesmos subconjuntos em $\widetilde{\mathscr{K}}$.

Note então, que a e b denotam que o resultado de particionamento do algoritmo está de acordo com o particionamento real, e c e d denotam que o particionamento resultante do algoritmo não está de acordo com o particionamento real.

Sendo assim, o Índice de Rand (RI) pode ser definido pela fração com que o particionamento obtido está de acordo com o particionamento real, da seguinte forma:

$$RI(\mathscr{K}, \widetilde{\mathscr{K}}) = \frac{a+b}{a+b+c+d}$$

A métrica de validação externa, NMI, tem efeito semelhante ao RI (MANNING; RAGHAVAN; SCHüTZE, 2008). Utilizando a mesma notação, essa métrica é baseada na Informação Mútua (MI - *Mutual Information*) entre o resultado de particionamento de um algoritmo ($\widetilde{\mathscr{K}}$) e o particionamento real ($\mathscr{K}$), e gera valores entre 0 e 1, com 0 indicando que não há informação mútua entre $\mathscr{K}$ e $\widetilde{\mathscr{K}}$, e 1 indicando correlação perfeita entre $\mathscr{K}$ e $\widetilde{\mathscr{K}}$. Então MI é definido da seguinte forma:

$$MI(\mathscr{K}, \widetilde{\mathscr{K}}) = \sum_{p=1}^{k} \sum_{p'=1}^{\tilde{k}} \frac{|\mathcal{K}_p \cap \widetilde{\mathcal{K}}_{p'}|}{n} \log \frac{n|\mathcal{K}_p \cap \widetilde{\mathcal{K}}_{p'}|}{|\mathcal{K}_p||\widetilde{\mathcal{K}}_{p'}|}$$

A normalização de MI é realizada usando a entropia dos particionamentos. A entropia de um particionamento, $H(\cdot)$, é dada da seguinte forma:

$$H(\mathscr{K}) = \sum_{p=1}^{k} \frac{\mathcal{K}_p}{n} \log \frac{\mathcal{K}_p}{n}$$

Finalmente, a métrica NMI pode ser definida como:

$$NMI(\mathscr{K}, \widetilde{\mathscr{K}}) = \frac{MI(\mathscr{K}, \widetilde{\mathscr{K}})}{\sqrt{H(\mathscr{K})H(\widetilde{\mathscr{K}})}}$$

Note, então, que ambas as métricas têm capacidade de lidar com a caso em que a ordem do particionamento no subconjunto $\mathscr{K}$ não corresponde com a ordem do particionamento no subconjunto $\widetilde{\mathscr{K}}$. Além disso, ambas as métricas têm sido efetivas como um método de análise quantitativa na análise de qualidade de agrupamento entre diferentes algoritmos, principalmente no que diz respeito ao domínio de agrupamento de documentos (KUANG, 2014; HO, 2008; WANG et al., 2011; DING et al., 2006; XU; LIU; GONG, 2003; YOO; CHOI, 2010).

2.3 Coagrupamento

Grande parte dos esforços de pesquisa da área de mineração de dados é voltada ao estudo da tarefa de agrupamento (HAN; KAMBER; PEI, 2011), no entanto, ainda há muito

estudo a ser realizado quando se considera dados com alta dimensionalidade e/ou quando se deseja interpretabilidade do agrupamento gerado por um algoritmo.

Nesse contexto, faz-se interessante analisar de forma mais detalhada, as características que, dentre todas as disponíveis, tornam os dados mais similares entre si a ponto de poderem ser associados a um mesmo grupo. Mais do que isso, a ideia é saber se ao considerar apenas parte das características, um dado pode ser associado a um grupo diferente. Essa análise leva ao estudo de questões referentes a estratégias capazes de usar a similaridade baseada em partes, a qual analisa subconjuntos de características. Um dos primeiros trabalhos a tratar esse tipo de problema foi apresentado por Hartigan (1972). Nesse trabalho, o autor apresenta métodos de agrupamento de dados que particionam tanto os dados quanto as características, a análise das partições e uma taxonomia referente às diversas estruturas que essas partições podem compor.

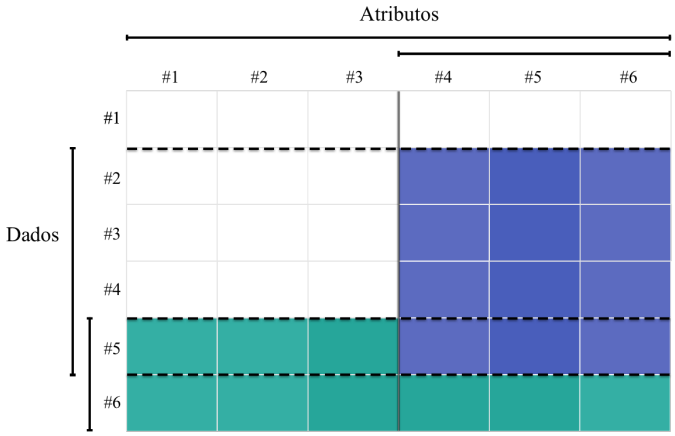
Com o passar do tempo, esse problema foi se tornando mais presente em aplicações reais, nas mais diferentes áreas, como Mineração de Textos, Mineração da Web, Bioinformática, Marketing, Ecologia, Arqueologia e Ciência da Computação (GOVAERT; NADIF, 2013). E hoje, o processo de agrupamento que se baseia na similaridade baseada em partes das características é conhecido como coagrupamento ou biagrupamento. O termo biagrupamento é usado principalmente no contexto de expressão genética, enquanto o termo Coagrupamento é usado principalmente em aplicações de mineração de texto.

Os algoritmos do processo de coagrupamento são capazes de gerar grupos de dados levando em consideração todas as características, ou apenas algumas características (FRANCA, 2010; MADEIRA; OLIVEIRA, 2004). Existem diversas maneiras de visualizar o problema de coagrupamento, uma visão importante é de encontrar submatrizes de uma matriz de dados, formalizando (MADEIRA; OLIVEIRA, 2004): tem-se uma matriz X, de dimensão $n \times m$, formada por um conjunto de vetores de linhas $\mathcal{N} = \{\mathbf{x}_{1\cdot}, \ldots, \mathbf{x}_{n\cdot}\}$ e um conjunto de vetores de colunas $\mathcal{M} = \{\mathbf{x}_{\cdot 1}, \ldots, \mathbf{x}_{\cdot m}\}$. O problema de coagrupamento, dentro desta visão, é encontrar cogrupos, que são submatrizes de X, denotados por $X_{\mathcal{K}_p\mathcal{L}_q}$, considerando k subconjuntos ordenados $\mathcal{K}_p \subseteq \mathcal{N}$, l subconjuntos ordenados $\mathcal{L}_q \subseteq \mathcal{M}$, e os índices $p \in \{1, \ldots, k\}$ e $q \in \{1, \ldots, l\}$. Assim, o cogrupo $X_{\mathcal{K}_p\mathcal{L}_q}$ é um grupo dos dados em $\mathcal{K}_p$, perante as características em $\mathcal{L}_q$.

Essa visão do problema de coagrupamento está ilustrada na figura 2, na qual é apresentado um conjunto de dados que possui as linhas (dados) $\mathbf{x}_{1\cdot}, \mathbf{x}_{2\cdot}, \mathbf{x}_{2\cdot}, \mathbf{x}_{4\cdot}, \mathbf{x}_{5\cdot}$ e $\mathbf{x}_{6\cdot}$ que são representados pelas colunas (características) $\mathbf{x}_{1\cdot}, \mathbf{x}_{\cdot 2}, \mathbf{x}_{\cdot 3}, \mathbf{x}_{\cdot 4}, \mathbf{x}_{\cdot 5}$ e $\mathbf{x}_{\cdot 6}$, sendo $n, m = 6$.

Na figura 2 também estão ilustrados dois cogrupos hipotéticos: o primeiro formado pelas linhas $\mathbf{x}_{5\cdot}$ e $\mathbf{x}_{6\cdot}$ e todos os atributos $\mathbf{x}_{\cdot 1}, \ldots, \mathbf{x}_{\cdot 6}$, representando um cogrupo com *modelo global*; e o segundo formado pelos objetos $\mathbf{x}_{2\cdot}, \mathbf{x}_{3\cdot}, \mathbf{x}_{4\cdot}$ e $\mathbf{x}_{5\cdot}$, e as características $\mathbf{x}_{\cdot 4}, \mathbf{x}_{\cdot 5}$ e $\mathbf{x}_{\cdot 6}$, representando um cogrupo com *modelo local*, que leva em consideração apenas um subconjunto dos atributos.

Figura 2 – Conjunto de dados com dois cogrupos



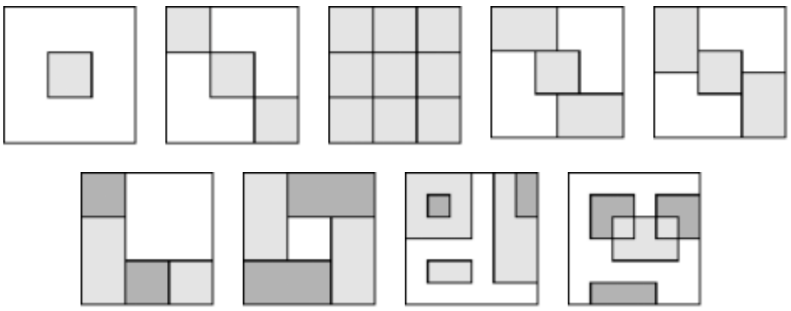
Fonte: Adaptado de Madeira e Oliveira (2004)

Como a definição apresentada não inclui uma estrutura a priori da matriz X e dos cogrupos $X_{\mathcal{K}_p\mathcal{L}_q}$, os algoritmos propostos na literatura diferem quanto ao tipo de cogrupo que são capazes de encontrar. Uma taxonomia dos tipos de cogrupos é proposta por Madeira e Oliveira (2004):

- Cogrupos com valores constantes.
- Cogrupos com valores constantes nas linhas ou colunas.
- Cogrupos com valores coerentes.
- Cogrupos com evoluções coerentes.

Os cogrupos também diferem quanto as suas estruturas. Cada algoritmo usado para implementar coagrupamento faz uma suposição da estrutura de cogrupos que é capaz de encontrar. A figura 3 sumariza as diferentes estruturas de cogrupos, com as linhas e colunas ordenadas para permitir a visualização dos cogrupos por meio de cada retângulo em cada uma das nove matrizes apresentadas.

Figura 3 – Estruturas de cogrupos



Fonte: Madeira e Oliveira (2004)

É possível nomear cada uma das estruturas apresentadas na figura 3, considerando as imagens da esquerda para a direita em linha: 1) único cogrupo; 2) cogrupos com linhas e colunas exclusivas; 3) cogrupos com estrutura em xadrez; 4) cogrupos com linhas exclusivas e sobreposição nas colunas; 5) cogrupos com colunas exclusivas e sobreposição nas linhas; 6) cogrupos com estrutura em árvore; 7) cogrupos não exclusivos sem sobreposição; 8) cogrupos com sobreposição e estrutura hierárquica; 9) cogrupos arbitrariamente posicionados. No capítulo 5 são mostrados experimentos considerando dados sintéticos baseados nessas estruturas, com ênfase nas estruturas que apresentam cogrupos com sobreposição e provendo mais detalhes sobre essas estruturas.

Diversos algoritmos para encontrar cogrupos, de diferentes tipos e estruturas, foram propostos na literatura (TANAY; SHARAN; SHAMIR, 2005; MADEIRA; OLIVEIRA, 2004).

Um dos algoritmos mais simples para a resolução da tarefa de Coagrupamento, que é capaz de encontrar cogrupos com valores coerentes em estrutura com sobreposição e arbitrariamente posicionados, é chamado de *Coupled Two-way Clustering* (CTWC) (GETZ; LEVINE; DOMANY, 2000). O algoritmo CTWC é capaz de encontrar cogrupos por meio do agrupamento de dados e atributos (linhas e colunas), separadamente. Getz, Levine e Domany (2000) utiliza o algoritmo de agrupamento *Superparamagnetic Clustering* (SPC), o qual é capaz de determinar o número de grupos automaticamente com uma estratégia de agrupamento hierárquica.

Já o algoritmo de Cheng e Church (2000) é capaz de encontrar o mesmo tipo de cogrupo que o algoritmo CTWC, porém usando uma estratégia gulosa: cogrupos com valores coerentes e estrutura com sobreposição e arbitrariamente posicionados. Para encontrar cogrupos, ou δ-cogrupos, Cheng e Church (2000) usam uma estratégia gulosa que retira linhas e colunas de X, respeitando um parâmetro δ, que é calibrado pelo usuário.

Além dos algoritmos mencionados, existem outros algoritmos que são capazes de encontrar outros tipos de cogrupos (FRANCA; ZUBEN, 2010; YANG; LESKOVEC, 2013; HOCHREITER et al., 2010; CABANES; BENNANI; FRESNEAU, 2012).

Ainda, de forma semelhante à tarefa de agrupamento, existem métricas de validação que propiciam a avaliação da qualidade e estabilidade de cogrupos, que são separadas em duas das categorias também presentes em agrupamento (HOCHREITER et al., 2010): *interna* e *externa*. Também, uma das formas de avaliar a qualidade e a estabilidade é pela própra medida de minimização de cada algoritmo, a qual pode não ser tão efetiva quando medidas de validação, mas apresenta-se como uma alternativa.

Resultados de pesquisa apresentados na literatura da área, que fazem uso da visão de coagrupamento discutida, são usados neste trabalho. Detalhes destes resultados são apresentados, em detalhes, no capítulo 3.

3 Fatoração de matrizes não-negativas para coagrupamento

Fatoração de matrizes não-negativas (*Non-negative Matrix Factorization* - NMF) foi estudada como um método para análise de dados capaz de extrair conhecimento sobre um objeto a partir do estudo de suas partes (LEE; SEUNG, 1999), como um contraponto a métodos mais populares como Análise de Componentes Principais (*Principal Component Analysis* - *PCA*) e Quantização Vetorial, porém, considerando a fatoração matrizes positivas ou negativas. Lee e Seung (1999) apresentam tal abordagem a partir de sua aplicação no aprendizado de características de faces (em dados do tipo imagem) e na análise de características semânticas de textos. A análise de textos também foi usada como aplicação na ilustração da aplicação de fatorização de matrizes por Ho (2008) que segue a ideia de aprendizado de partes, por Kuang (2014) que aplica fatoração de matrizes não-negativas sobre um problema formulado como análise de agrupamento (clustering), e por Long, Zhang e Yu (2005), Ding et al. (2006), Yoo e Choi (2010), que formulam o problema como coagrupamento. Ainda, outros contextos são submetidos à análise sob a formulação de problemas de coagrupamento, sendo alguns exemplos a clusterização de genes e análise de microarray em bioinformática (KLUGER et al., 2003) e a filtragem colaborativa em sistemas de recomendação (SALAKHUTDINOV; MNIH, 2008). Na aplicação de NMF em filtragem colaborativa em sistemas de recomendação, destaca-se o modelo baseado em NMF de Koren (2009) para predição de preferências de usuários por filmes, que ganhou em primeiro lugar o Prêmio Netflix (*Netflix Prize*)[1].

A adequação da fatoração de matrizes para tarefas modeladas como agrupamento ou coagrupamento ocorre porque muitas das representações usadas em aplicações nessas áreas se apresentam como uma relação entre pares de elementos pertencentes a conjuntos finitos, como apresentado em (LONG; ZHANG; YU, 2005). Por exemplo, na resolução da tarefa de agrupamento de documentos, em mineração de texto, usa-se, comumente, dois conjuntos finitos, documentos e palavras, sendo que a relação entre eles é representada pela ocorrência de uma determinada palavra em um determinado documento. Note ainda que a relação expressa entre os elementos, como no contexto de palavras e documentos, apresenta-se como uma matriz de dados positiva, característica que ilustra a aplicabilidade de NMF.

[1] http://www.netflixprize.com/

Formalmente, algoritmos de coagrupamento baseados em NMF têm como entrada uma matriz de dados $X \in \mathbb{R}_{+}^{n \times m}$, contendo números reais positivos com n linhas e m colunas. Esta matriz é formada por um conjunto de vetores de linhas $\mathcal{N} = \{\mathbf{x}_{1\cdot}, \ldots, \mathbf{x}_{n\cdot}\}$ e um conjunto de vetores de colunas $\mathcal{M} = \{\mathbf{x}_{\cdot 1}, \ldots, \mathbf{x}_{\cdot m}\}$, e as relações existentes entre cada linha x e cada coluna y são representadas por x_{ij} considerando os índices $i \in \{1, \ldots, n\}$ e $j \in \{1, \ldots, m\}$, que é justamente um valor da matriz X. Cada valor em x_{ij} representa, então, a relação existente entre pares de elementos em algum contexto de interesse. O objetivo é encontrar k partições de $\mathcal{N}$, denotadas pelos subconjuntos ordenados $\mathcal{K}_p \subseteq \mathcal{N}$, l partições para $\mathcal{M}$, denotadas pelos subconjuntos ordenados $\mathcal{L}_q \subseteq \mathcal{M}$, considerando os índices $p \in \{1, \ldots, k\}$ e $q \in \{1, \ldots, l\}$. Então, os subconjuntos $\{\mathcal{K}_1, \ldots, \mathcal{K}_k\}$ e $\{\mathcal{L}_1, \ldots, \mathcal{L}_l\}$ são os cogrupos de linhas e colunas, respectivamente. Da mesma forma que na tarefa de agrupamento, faz-se sentido a proposição de algoritmos baseados em heurísticas, pois sob diversas formas, o problema de coagrupamento baseado em particionamento, é um problema NP-difícil (BULTEAU et al., 2014).

Para implementação da NMF como uma estratégia para resolução do problema de coagruamento, diferentes algoritmos foram apresentados na literatura. Cada um deles considera o problema de NMF com diferentes restrições que permitem propor soluções para o problema de coagrupamento de diferentes naturezas. Este capítulo se destina a apresentar três das implementações existentes que são usadas como base para a proposta desta dissertação: decomposição de valores em blocos (Seção 3.1) introduzida por Long, Zhang e Yu (2005); fatoração ortogonal tripla de matrizes não-negativas (Seção 3.2) introduzida por Ding et al. (2006); e fatoração tripla rápida de matrizes não-negativas (Seção 3.3) introduzida por Wang et al. (2011). Outras implementações correlatas podem ser encontradas em (LI; DING, 2006).

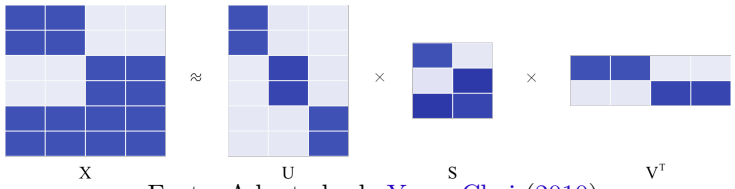
Todos algoritmos de NMF para resolução das tarefas de agrupamento e coagrupamento tem em comum a natureza recursiva, pois são encontradas partições de linhas a partir de partições de colunas, para então, encontrar melhores partições de colunas a partir das partições de linhas. Esse processo recursivo continua até que a aproximação entre a fatoração e a matriz fatorada (X) através de alguma medida atinja um mínimo local ou global, resultando em uma solução para o particionamento de linhas e colunas.

3.1 Decomposição de Valores em Blocos para Coagrupamento

A Decomposição de Valores em Blocos (*Block Value Decomposition* - *BVD*) foi proposta por Long, Zhang e Yu (2005) como uma abordagem para tratar o problema de coagrupamento, com base em fatoração de matrizes não-negativas. Essa decomposição recebe esse nome por ter a capacidade de encontrar estruturas em blocos escondidas na matriz de dados. Isso é possível porque o algoritmo *BVD* é capaz de explorar a relação entre linhas e colunas da matriz de dados por meio da decomposição dela em três matrizes: U uma matriz de coeficientes de linhas, S uma matriz com estrutura em blocos, e V uma matriz de coeficientes de colunas. Segundo os autores, tais coeficientes em U e V podem ser vistos como um fator que associa linhas a partições encontradas no conjunto de linhas, e que associa as colunas a partições encontradas no conjunto de colunas, respectivamente; e S pode ser vista como a representação média dos detalhes da matriz original e permite sua reconstrução aproximada a partir da operação USV^T. Sob o ponto de vista de resolução do problema de coagrupamento, então, o objetivo no *BVD* é encontrar grupos de linhas e colunas de forma simultânea, sendo k grupos de $\mathcal{N}$ (linhas) e l grupos de $\mathcal{M}$ (colunas).

Ainda, os autores proponentes da abordagem *BVD* defendem que interpretações intuitivas podem ser derivadas da análise das combinações das matrizes geradas na fatoração, quando aplicadas a um contexto específico. Um exemplo fornecido no trabalho de Long, Zhang e Yu (2005) é que, considerando uma matriz de entrada que representa a relação "documentos por palavras" (linhas por colunas) cada coluna de US captura a ideia de uma base para a representação de grupos de palavras; e cada linha em SV^T captura a ideia de uma base para a representação de grupos de documentos. Uma representação gráfica para o resultado de uma fatoração de matrizes pode ser vista na figura 4.

Figura 4 – Fatoração da matriz original de dados X em três outras matrizes: U, S e V



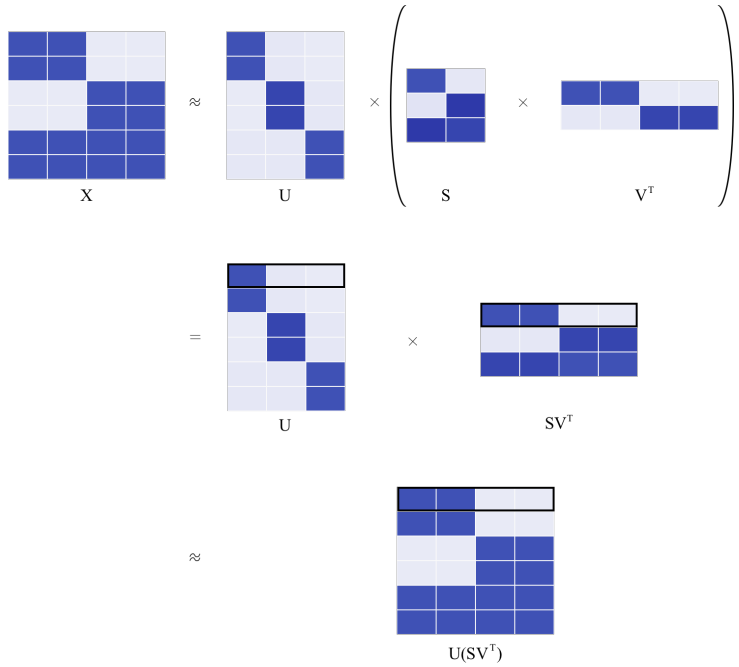
Fonte: Adaptado de Yoo e Choi (2010)

Na figura 4 considere que uma célula com cor escura representa a existência de uma relação entre linha e coluna, e uma célula com cor clara representa a inexistência de uma relação entre linha e coluna, e que essas relações são estabelecidas adequadamente em

cada contexto de aplicação. Transportando o exemplo da figura para o contexto de uma matriz de "documentos por palavras" tem-se um conjunto de seis documentos e quatro palavras, sendo que, por exemplo, o primeiro documento possui uma relação com as duas primeiras palavras, e não possui relação com as terceira e quarta palavras. A matriz U pode ser interpretada como uma matriz "documentos por grupos de documentos", sendo portanto uma situação em que seis documentos estão agrupados em três grupos ($k = 3$): os dois primeiros documentos no primeiro grupo, os dois próximos no segundo grupo e os dois últimos no terceiro grupo. A matriz V^T pode ser interpretada como uma matriz de "grupos de palavras por palavras", sendo portanto uma situação em que há dois grupos de palavras ($l = 2$) no contexto das quatro palavras existentes. Finalmente, a matriz S representa uma relação entre "grupos de documentos" e "grupos de palavras". A primeira linha da matriz S indica que há uma relação entre o primeiro grupo de linhas e o primeiro grupos de palavras, e que não há uma relação entre o primeiro grupo de linhas e o segundo grupo de palavras. Seguindo a interpretação intuitiva apresentada por (LONG; ZHANG; YU, 2005), uma das combinações possíveis, SV^T, está ilustrada na figura 5. Observe que a matriz SV^T representa "três grupos de documentos por quatro palavras", constituindo-se como uma base de representação para grupos de documentos.

Note ainda que nessa estrutura de matrizes fatoradas é possível identificar protótipos responsáveis por cada parte da reconstrução da matriz original X, visto que cada base de representação pode ser vista como um conjunto de vetores protótipos: as linhas de (SV^T) são vetores base (protótipos de grupos de linhas). O raciocínio apresentado para interpretação da figura 5 pode também ser feito para a interpretação da combinação US. Importante salientar que, assim como ocorre na resolução da tarefa de agrupamento, diferentes grupos de linhas e de colunas podem ser obtidos, representando diferentes soluções para o problema. E, neste caso ainda, diferentes matrizes S podem ser obtidas para uma mesma organização de grupos de linhas e colunas. Portanto, qualquer interpretação derivada dessa análise deve ser considerada como apenas uma das formas possíveis de análise dos dados provenientes do contexto de aplicação.

Figura 5 – A reconstrução da primeira linha $\mathbf{x}_{1.}$ de X, através da multiplicação da matriz indicadora de grupos de linhas U pela matriz dos protótipos de linhas (SV^T).



Fonte: Lucas Fernandes Brunialti, 2016

Semelhante ao *fuzzy k-means*, é possível observar esta fatoração como uma ótica de compactação. O BVD compacta a matriz de dados em uma matriz com fatores que correlacionam cada linha com cada grupo de linhas (U). Além disso, esta compactação adiciona a idéia de uma matriz de fatores V para compactar as colunas de X, e S que é uma visão compactada de X em kl elementos. Portanto, a compactação transforma nm elementos em $nk + kl + ml$ elementos, através das matrizes U, S e V, e supondo que $n \ll k$ e $m \ll l$, tem-se que $mm \ll nk + kl + ml$.

O problema de coagrupamento implementado sob a abordagem BVD é formalmente apresentado como (LONG; ZHANG; YU, 2005):

Problema 3 (Problema de decomposição de valores em blocos)**.**

$$\begin{aligned} \mathcal{F}_3(U,S,V) \quad = \quad & \min_{U,S,V} \quad \left\|X - USV^T\right\|_F^2 \\ & \textit{suj. a} \qquad U \geq 0, \\ & \qquad\qquad S \geq 0, \\ & \qquad\qquad V \geq 0 \end{aligned}$$

em que $U \in \mathbb{R}_+^{n\times k}$, $S \in \mathbb{R}_+^{k\times l}$, $V \in \mathbb{R}^{m\times l}$, *e* $\|\cdot\|_F$ *denota a norma de Frobenius para matrizes.*

A implementação para o processo de minimização do problema 3 é descrito no algoritmo 3, o qual é baseado em atualizações multiplicativas e tem sua convergência

demonstrada via teoria de otimização não linear (veja Long, Zhang e Yu (2005)). Nesse algoritmo considere t o contador de iterações, $U^{(t)}$, $S^{(t)}$ e $V^{(t)}$, as matrizes U, S e V, na iteração t, respectivamente, e $\odot$ é o produto de Hadamard.

Algoritmo 3 Algoritmo baseado em atualização multiplicativa para solução do *BVD*

function BVD(X, k, l, t_{max})
  Inicialize: $U^{(0)} \leftarrow \mathcal{U}(0,1), V^{(0)} \leftarrow \mathcal{U}(0,1), S^{(0)} \leftarrow \frac{1}{nm}\sum_{i,j} x_{ij}$ e $t \leftarrow 0$.
  while (não convergiu) e $(t \leq t_{max})$ **do**

$$U^{(t+1)} \leftarrow U^{(t)} \odot \frac{XV^{(t)}S^{(t)^T}}{U^{(t)}S^{(t)}V^{(t)^T}V^{(t)}S^{(t)^T}}$$

$$V^{(t+1)} \leftarrow V^{(t)} \odot \frac{X^TU^{(t+1)}S^{(t)}}{V^{(t)}S^{(t)^T}U^{(t+1)^T}U^{(t+1)}S^{(t)}}$$

$$S^{(t+1)} \leftarrow S^{(t)} \odot \frac{U^{(t+1)^T}XV^{(t+1)}}{U^{(t+1)^T}U^{(t+1)}S^{(t)}V^{(t+1)^T}V^{(t+1)}}$$

    $t \leftarrow t+1$
  end while
  return $U^{(t)}, S^{(t)}, V^{(t)}$
end function

A inicialização dos elementos das matrizes U, S e V são gerados através de uma distribuição uniforme ($\mathcal{U}(0,1) \in]0,1]$). Como condições para assumir a convergência, neste trabalho, considera-se a diferença do erro de aproximação em duas iterações consecutivas menor ou igual a um ϵ:

$$\left\|X - U^{(t)}S^{(t)}V^{(t)^T}\right\|_F^2 - \left\|X - U^{(t+1)}S^{(t+1)}V^{(t+1)^T}\right\|_F^2 \leq \epsilon$$

O algoritmo também pára caso a quantidade de iterações assuma o limite t_{max}.

Note que como U e V possuem valores no domínio dos reais, então, não é possível obter as partições diretamente, sem um processo de pós-processamento. Um modo simples de obter o particionamento para as linhas toma a seguinte forma:

$$\mathcal{K}_p = \{x_{i\cdot} \mid i \in \{1,\ldots,n\} \text{ e } p = \underset{p' \in \{1,\ldots,k\}}{\arg\max}\, u_{ip'}\}, \forall p \in \{1,\ldots,k\}$$

Isso significa que uma linha i pertencerá a um cogrupo p (ou partição) se para todos os k cogrupos, o fator u_{ip} for maior que todos os outros fatores para os outros cogrupos, presentes no vetor $\mathbf{u}_{i\cdot}$.

A complexidade de tempo do algoritmo é possível ser calculada fixando as condições $k \simeq l$, $n \simeq m$, $k \ll n$, $l \ll m$ e usando um algoritmo para otimizar a ordem das multi-

plicações (Cormen et al. (2001) discute algoritmos para tal otimização): $\mathcal{O}\Big(t_{max}\big(nl(m+k)+ml(n+k)+mk(n+l)+k^2(n+l)+l^2(m+k)\big)\Big) \simeq \mathcal{O}\Big(t_{max}\big(n^2k+nk^2+k^3\big)\Big)$.

3.2 Fatoração Ortogonal Tripla de Matrizes Não-negativas

Baseado no problema 3, Ding et al. (2006) propõem o problema 4, e o chama de Fatoração Ortogonal Tripla de Matrizes Não-negativas (*Orthogonal Non-negative Matrix Tri-factorization - ONMTF*).

Problema 4 (Problema de fatoração ortogonal tripla de matrizes não-negativas)**.**

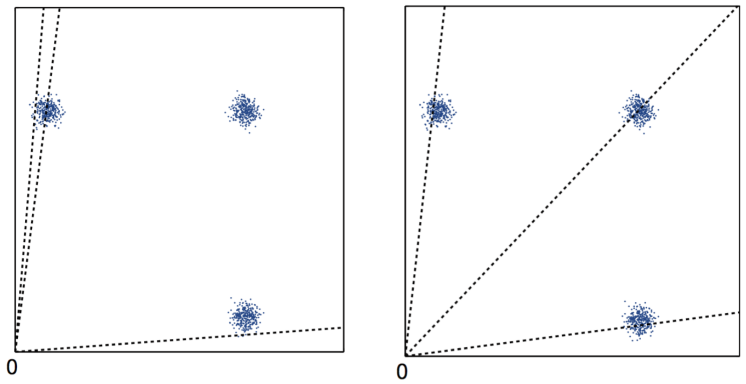
$$\begin{aligned} \mathcal{F}_6(U,S,V) \quad = \quad & \min_{U,S,V} \quad \left\|X-USV^T\right\|_F^2 \\ & \textit{suj. a} \quad U\geq 0, S\geq 0, V\geq 0, \\ & \qquad\qquad U^TU=I, \\ & \qquad\qquad V^TV=I \end{aligned}$$

em que $U\in\mathbb{R}_+^{n\times k}$, $S\in\mathbb{R}_+^{k\times l}$, $V\in\mathbb{R}_+^{m\times l}$ *e* $\|\cdot\|_F$ *denota a norma de Frobenius.*

Na formulação desse problema, duas restrições de ortonormalidade são acrescentadas, $U^TU=I$ e $V^TV=I$, em que I é a matriz identidade, para as matrizes de grupos de linhas e grupos de colunas, respectivamente. Tais restrições restringem o problema da fatoração $X\approx USV^T$ para um número menor de possíveis soluções, buscando a unicidade, como mostrado na figura 6. No entanto, o algoritmo não garante que as restrições de ortonomalidade são respeitadas, apesar das restrições colocadas encontrarem soluções de particionamento com maior ortogonalidade.

A figura 6 representa um problema com três grupos (três nuvens de pontos). As linhas pontilhadas representam os protótipos obtidos por FMN (a) sem restrição de ortogonalidade nas matrizes, como o *BVD* (a), e (b) com restrição de ortogonalidade nas matrizes, como o *ONMTF*. Note que a base obtida com FMN ortogonal tende a encontrar protótipos mais centralizados nos grupos. Já a base obtida com FMN sem restrições tende a encontrar uma região convexa que contém os pontos dos grupos (incluindo regiões que abrangem pontos de grupos originalmente projetados como sendo diferentes). A solução obtida em (a), embora correta, não é desejável.

Figura 6 – Base de protótipos obtidas com FMN sem restrições (a) e com restrições de ortogonalidade nas matrizes



(a) (b)

Fonte: Yoo e Choi (2010)

Em termos de compactação, igualmente ao *BVD*, o algoritmo para solução do *ONMTF* transforma os nm elementos da matriz X em $nk + kl + ml$ elementos, através da fatoração em U, S e V, e supondo que $n \ll k$ e $m \ll l$, tem-se que $mm \ll nk + kl + ml$.

Ding et al. (2006) propõem uma solução para implementação do processo de minimização para o problema 4 semelhante ao que foi apresentado na seção 3.1, também baseado em atualizações multiplicativas e com convergência demonstrada com base na teoria de otimização não linear. O algoritmo para tal processo de minimização é apresentado no algoritmo 4, no qual t o contador de iterações, $U^{(t)}$, $S^{(t)}$ e $V^{(t)}$, as matrizes U, S e V na iteração t, respectivamente, $\odot$ é o produto de Hadamard e $\mathcal{U}(0,1) \in]0,1]$ uma função que gera valores de uma distribuição uniforme. Também, a mesma condição de convergência aplicada no algoritmo 3 pode ser aplicada nesse caso.

Note que é possível obter o particionamento de linhas e colunas da mesma forma que foi descrita com o *BVD*. Calculando a complexidade de tempo, fixando as mesmas condições $k \simeq l$, $n \simeq m$, $k \ll n$, $l \ll m$ e usando um algoritmo para otimizar a ordem das multiplicações, é possível encontrar a mesma complexidade antes calculada para o algoritmo 3.

No artigo de Yoo e Choi (2010), é proposta uma abordagem mais simples para a derivação das regras de atualização multiplicativas, considere uma função de otimização qualquer $\mathcal{J}$ e seu respectivo gradiente $\nabla\mathcal{J}$:

$$\nabla\mathcal{J} = [\nabla\mathcal{J}]^+ - [\nabla\mathcal{J}]^-$$

Algoritmo 4 Algoritmo baseado em atualização multiplicativa para solução do *ONMTF*

function ONM3F(X, k, l, t_{max})
  Inicialize: $U^{(0)} \leftarrow \mathcal{U}(0,1), V^{(0)} \leftarrow \mathcal{U}(0,1), S^{(0)} \leftarrow \mathcal{U}(0,1)$ e $t \leftarrow 0$.
  while (não convergiu) e $(t \leq t_{max})$ **do**

$$U^{(t+1)} \leftarrow U^{(t)} \odot \sqrt{\frac{XV^{(t)}S^{(t)^T}}{U^{(t)}U^{(t)^T}XV^{(t)}S^{(t)^T}}} \tag{1}$$

$$V^{(t+1)} \leftarrow V^{(t)} \odot \sqrt{\frac{X^TU^{(t+1)}S}{V^{(t)}V^{(t)^T}X^TU^{(t+1)}S^{(t)}}} \tag{2}$$

$$S^{(t+1)} \leftarrow S^{(t)} \odot \sqrt{\frac{U^{(t+1)^T}XV^{(t+1)}}{U^{(t+1)^T}U^{(t+1)}S^{(t)}V^{(t+1)^T}V^{(t+1)}}} \tag{3}$$

    $t \leftarrow t + 1$
  end while
  return $U^{(t)}, S^{(t)}, V^{(t)}$
end function

onde $[\nabla\mathcal{J}]^+$ é a parte positiva do gradiente, $[\nabla\mathcal{J}]^-$ a parte negativa do gradiente. Se $[\nabla\mathcal{J}]^+ \geq 0$ e $[\nabla\mathcal{J}]^- \geq 0$, então, é possível definir uma regra de atualização multiplicativa, para otimizar os parâmetros Θ da função $\mathcal{J}$:

$$\Theta \leftarrow \Theta \odot \left(\frac{[\nabla\mathcal{J}]^-}{[\nabla\mathcal{J}]^+}\right)^{\cdot\eta} \tag{4}$$

onde $\odot$ representa o produto Hadamard, $(\cdot)^{\cdot\eta}$ representa a potência para cada elemento, e η uma taxa de aprendizado $(0 < \eta \leq 1)$. Então, se Θ for inicializado com elementos positivos, é possível verificar que a regra de atualização multiplicativa da equação 4 mantém a não-negatividade de Θ.

Também, é utilizada uma abordagem diferente para a derivação de regras de atualização multiplicativas, visando um algoritmo para a solução do problema 4. Neste caso, o gradiente é calculado com base em uma superfície com restrições que preserva a ortogonalidade. Essa superfície com restrições é chamada de Variedade de Stiefel (*Stiefel Manifold*).

Assim, Yoo e Choi (2010) faz o uso da estratégia da equação 4 e da teoria de derivação na superfície com restrições (Variedade Stiefel), para propor uma solução para o problema 4 (*ONMTF*) alternativa às atualizações das equações 1, 2 e 3, através da atualização multiplicativa. Essas atualizações são apresentadas no algoritmo 5, considere t o contador de iterações, $U^{(t)}$, $S^{(t)}$ e $V^{(t)}$, as matrizes U, S e V, na iteração t, respectivamente, $\odot$ o produto de Hadamard, $\mathcal{U}(0,1) \in]0,1]$ uma função que gera valores de uma distribuição

uniforme, diag$(\cdot)$ uma função que extrai a diagonal principal de uma matriz e a transforma em um vetor, e $\mathbf{1}$ um vetor de uns com dimensão que torne a multiplicação possível.

Considerando a preservação da ortonormalidade, o algoritmo apresentado é capaz de preservar melhor a ortonormalidade. Para tal afirmação, Yoo e Choi (2010) realizou um experimento que consistiu em medir as restrições de ortonormalidade em U e V, através dos erros $\left\|U^TU - I\right\|$ e $\left\|V^TV - I\right\|$, respectivamente, durante cada iteração do algoritmo proposto, comparando-o com o algoritmo proposto por Ding et al. (2006).

Algoritmo 5 Algoritmo baseado em atualização multiplicativa e na teoria de de derivação na superfície com restrições (Variedade Stiefel) para solução do *ONMTF*

1: **function** ONMTF(X, k, l, t_{max})
2: **Inicialize:** $U^{(0)} \leftarrow \mathcal{U}(0,1), V^{(0)} \leftarrow \mathcal{U}(0,1), S^{(0)} \leftarrow \mathcal{U}(0,1)$ e $t \leftarrow 0$.
3: **while** (não convergiu) e $(t \leq t_{max})$ **do**
4: $$U^{(t+1)} \leftarrow U^{(t)} \odot \frac{XV^{(t)}S^{(t)^T}}{U^{(t)}S^{(t)}V^{(t)^T}X^TU^{(t)}}$$
5: $$V^{(t+1)} \leftarrow V^{(t)} \odot \frac{X^TU^{(t+1)}S^{(t)}}{V^{(t)}S^{(t)^T}U^{(t+1)^T}XV^{(t)}}$$
6: $$S^{(t+1)} \leftarrow S^{(t)} \odot \frac{U^{(t+1)^T}XV^{(t+1)}}{U^{(t+1)^T}U^{(t+1)}S^{(t)}V^{(t+1)^T}V^{(t+1)}}$$
7: $t \leftarrow t + 1$
8: **end while**
9: $$U^{(t)} \leftarrow U^{(t)}\,\mathrm{diag}(S^{(t)}\,\mathrm{diag}(\mathbf{1}^TV^{(t)})\mathbf{1})$$
10: $$V^{(t)} \leftarrow V^{(t)}\,\mathrm{diag}(\mathbf{1}^T\,\mathrm{diag}(\mathbf{1}^TU^{(t)})S^{(t)})$$
11: **return** $U^{(t)}, S^{(t)}, V^{(t)}$
12: **end function**

Ainda, Yoo e Choi (2010) propõem uma normalização baseada numa interpretação probabilística da fatoração de X em USV^T, para então realizar o particionamento de linhas e colunas, como apresentado no algoritmo 5 (linhas 9 e 10). Note que o particionamento de linhas e colunas é realizado como nos outros algoritmos já apresentados e a mesma condição de convergência aplicada no algoritmo 3 e 4 pode ser aplicada nesse caso.

Calculando a complexidade de tempo, seguindo as mesmas restrições apresentadas anteriormente para os outros algoritmos, é possível verificar que o algoritmo 5, proposto por Yoo e Choi (2010) para solução do problema 4 (*ONMTF*), é equivalente à complexidade apresentada para o algoritmo 3.

3.3 Fatoração Tripla Rápida de Matrizes Não-negativas

O problema de Fatoração Tripla Rápida de Matrizes Não-negativas (*Fast Non-negative Matrix Tri Factorization* - FNMTF), formalizado no problema 5, foi proposto por Wang et al. (2011) com os seguintes argumentos contra o uso prático dos problemas até então propostos para encontrar cogrupos: eles exigem soluções algorítmicas iterativas, com intensas multiplicações de matrizes em cada passo do algoritmo; eles propõem encontrar coagrupamentos flexíveis (com restrições relaxadas), o que implica em encontrar inúmeras soluções para a tarefa de coagrupamento.

Problema 5 (Problema de fatoração tripla rápida de matrizes não-negativas)**.**

$$\begin{aligned} \mathcal{F}_7(U,S,V) \quad = \quad & \min_{U,S,V} \quad \left\|X - USV^T\right\|_F^2 \\ & \textit{suj. a} \quad U \in \Psi^{n\times k}, \\ & \qquad\quad V \in \Psi^{m\times l}, \\ & \qquad\quad \textstyle\sum_{p=1}^{k} u_{ip} = 1, \forall i, \\ & \qquad\quad \textstyle\sum_{q=1}^{l} v_{jq} = 1, \forall j \end{aligned}$$

em que $S \in \mathbb{R}_+^{k\times l}$ *e* $\Psi = \{0,1\}$ *e* $\|\cdot\|_F$ *denota a norma de Frobenius.*

Wang et al. (2011) denominam U como uma matriz indicadora dos grupos de linhas, V como uma matriz indicadora dos grupos de colunas, e S como uma matriz que contém os fatores que relacionam um grupo de linhas aos grupos de colunas, e um grupo de colunas aos grupos de linhas. Note que nesse caso não é necessário uma etapa de pós-processamento para particionamento como nos outros algoritmos apresentados.

Ainda, as restrições $\sum_{p=1}^{k} u_{ip} = 1$ e $\sum_{q=1}^{l} v_{jq} = 1$ indicam que uma linha e uma coluna, respectivamente, têm que pertencer à algum grupo. Então, apesar dessas restrições serem semelhantes à ortonormalidade, elas não são, pois não resolvem o caso em que há uma possibilidade de haver grupos vazios, ou seja, sem nenhum elemento pertencente a este grupo, enquanto a restrição de ortonormalidade, garante que não haverá grupos vazios, seja este grupo de linhas ou de colunas.

Como U e V neste caso têm as restrições descritas, e são capazes de fornecer o particionamento de linhas e colunas, respectivamente, de forma direta, a capacidade de compactação nesse caso é semelhante ao algoritmo *k-means*, brevemente descrito no capítulo 2. Porém, como a fatoração compacta as colunas e as relações entre grupos de

linhas e colunas (matriz S), a matriz X é compactada em $n + kl + m$ elementos, e, se $n \ll k$ e $m \ll l$, tem-se que $mm \ll n + kl + m$.

Como não há restrições em S, com exceção da positividade que é garantida pela positividade de X, é possível encontrar uma regra de atualização para S, e portanto, minimização de $\mathcal{F}_7$:

$$\begin{aligned} \nabla_S \mathcal{F}_7 &= U^T X V - U^T U S V^T V &&= 0 \\ &\implies U^T U S V^T V &&= U^T X V \\ &\implies (U^T U)^{-1} U^T U S V^T V (V^T V)^{-1} &&= (U^T U)^{-1} U^T X V (V^T V)^{-1} \end{aligned}$$

$$\therefore \quad S = (U^T U)^{-1} U^T X V (V^T V)^{-1}$$

Assim, o problema de minimização se transforma nos subproblemas de atualização de U e V. Uma estratégia semelhante aquela aplicada no clássico algoritmo *k-means* pode ser aplicada. Primeiramente, fixa-se S e V e resolve-se o problema 5 para U de forma iterativa, verificando quais dos protótipos de linhas (linhas de $\widetilde{V}$) mais se aproxima das linhas de X. Em seguida, fixa-se S e U para alcançar uma solução iterativa para V, através dos protótipos de colunas (colunas de $\widetilde{U}$), assim como mostrado no algoritmo 6.

Note que é possível interpretar a regra de atualização para S: $(U^T U)$ é uma matriz que na diagonal contém a contagem do número de linhas em cada grupo de linhas e zeros nas demais posições, $(U^T U)^{-1}$ é o fator que divide a soma dos valores para cada grupo de linhas, e $U^T X$ seleciona e soma as linhas de X para cada grupo de linhas. A mesma interpretação pode ser feita para $(V^T V)$, $(V^T V)^{-1}$ e XV. O seguinte exemplo mostra uma matriz de dados com 6 linhas e 3 colunas, particionada por 3 grupos de linhas e 2 grupos de colunas, ilustrando a interpretação da atualização de S:

$$X = \begin{bmatrix} — & \mathbf{x}_{1\cdot} & — \\ — & \mathbf{x}_{2\cdot} & — \\ — & \mathbf{x}_{3\cdot} & — \\ — & \mathbf{x}_{4\cdot} & — \\ — & \mathbf{x}_{5\cdot} & — \\ — & \mathbf{x}_{6\cdot} & — \end{bmatrix} = \begin{bmatrix} | & | & | \\ \mathbf{x}_{\cdot 1} & \mathbf{x}_{\cdot 2} & \mathbf{x}_{\cdot 3} \\ | & | & | \end{bmatrix}$$

$$U = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, U^TU = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}, V = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, V^TV = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

A matriz U do exemplo apresenta um particionamento das 6 linhas em 3 grupos, sendo que 3 linhas pertencem ao primeiro grupo, 2 linhas ao segundo grupo, e 1 linha ao terceiro grupo, como mostra a diagonal principal da matriz U^TU a mesma interpretação pode ser feita para V e V^TV.

$$(U^TX)V = \left(\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} — & \mathbf{x}_{1\cdot} & — \\ — & \mathbf{x}_{2\cdot} & — \\ — & \mathbf{x}_{3\cdot} & — \\ — & \mathbf{x}_{4\cdot} & — \\ — & \mathbf{x}_{5\cdot} & — \\ — & \mathbf{x}_{6\cdot} & — \end{bmatrix} \right) \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{1\cdot} + \mathbf{x}_{2\cdot} + \mathbf{x}_{6\cdot} \\ \mathbf{x}_{3\cdot} + \mathbf{x}_{5\cdot} \\ \mathbf{x}_{4\cdot} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} x_{13} + x_{23} + x_{63} & x_{11} + x_{12} + x_{21} + x_{22} + x_{61} + x_{62} \\ x_{33} + x_{53} & x_{31} + x_{32} + x_{51} + x_{52} \\ x_{43} & x_{41} + x_{42} \end{bmatrix}$$

A multiplicação em X por U^T pela esquerda, representa a soma da seleção de todas as linhas de X pertencentes à um mesmo grupo de linhas. O mesmo ocorre quando multiplica-se V pela direita de X, porém, representando a soma da seleção das colunas de X pertencentes à um mesmo grupo de colunas. Sendo assim, a operação U^TXV representa a soma de todos os elementos de X pertencentes à um mesmo grupo de linhas e colunas, ou seja, por exemplo o elemento da linha 1 e coluna 1 de U^TXV, que foi calculado no exemplo, é simplesmente a soma de todos os elementos de X que pertencem ao primeiro grupo de linhas e ao primeiro grupo de colunas.

$$(U^TU)^{-1} = \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}, (V^TV)^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$$

$$S = (U^T U)^{-1} U^T X V (V^T V)^{-1} = \begin{bmatrix} \frac{x_{13}+x_{23}+x_{63}}{3} & \frac{x_{11}+x_{12}+x_{21}+x_{22}+x_{61}+x_{62}}{3\times 2} \\ \frac{x_{33}+x_{53}}{2} & \frac{x_{31}+x_{32}+x_{51}+x_{52}}{2\times 2} \\ x_{43} & \frac{x_{41}+x_{42}}{2} \end{bmatrix}$$

Com o exemplo mostrado, é possível visualizar a atualização de S de forma mais intuitiva. Por exemplo, para calcular um elemento s_{pq} de S, será simplesmente o cálculo da média de todos elementos em X que pertencem ao grupo de linhas p e ao grupo de colunas q. Dessa forma, também é possível propor uma solução iterativa para S.

O algoritmo 6 ilustra o processo de minimização para o problema 5. Nesse algoritmo considere os índices $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, m\}$, $p = p' \in \{1, \ldots, k\}$, e $q = q' \in \{1, \ldots, l\}$, o contador de iterações t, $U^{(t)}$, $S^{(t)}$ e $V^{(t)}$ como sendo as matrizes U, S e V na iteração t, respectivamente, $\mathcal{U}(0,1) \in]0,1]$ uma função que gera valores de uma distribuição uniforme, e $\|\cdot\|^2$ é a norma frobenius para vetores. Também, as mesmas condições de convergência usada nos algoritmos anteriores pode ser aplicada nesse caso.

Algoritmo 6 Algoritmo iterativo para solução do *FNMTF*

1: **function** FNMTF(X, k, l, t_{max})
2: **Inicialize:** $U^{(0)}, V^{(0)} \leftarrow 0,1 \mid \sum_{p=1}^{k} u_{ip} = 1, \sum_{q=1}^{l} v_{jq} = 1, \forall i, j,\ S^{(0)} \leftarrow \mathcal{U}(0,1)$ e $t \leftarrow 0$.
3: **while** (não convergiu) e ($t \leq t_{max}$) **do**
4: $$S^{(t+1)} \leftarrow (U^{(t)^T} U^{(t)})^{-1} U^{(t)^T} X V^{(t)} (V^{(t)^T} V^{(t)})^{-1}$$
5: $$\widetilde{V} \leftarrow S^{(t+1)} V^{(t)^T}$$
6: $$(U^{(t+1)})_{ip} \leftarrow \begin{cases} 1 & p = \arg\min_{p' \in \{1,\ldots,k\}} \|\mathbf{x}_{i\cdot} - \widetilde{\mathbf{v}}_{p'\cdot}\|^2 \\ 0 & \textit{caso contrário} \end{cases} \quad \forall i, p$$
7: $$\widetilde{U} \leftarrow U^{(t+1)} S^{(t+1)}$$
8: $$(V^{(t+1)})_{jq} \leftarrow \begin{cases} 1 & q = \arg\min_{q' \in \{1,\ldots,l\}} \|\mathbf{x}_{\cdot j} - \widetilde{\mathbf{u}}_{\cdot q'}\|^2 \\ 0 & \textit{caso contrário} \end{cases} \quad \forall j, q$$
9: $t \leftarrow t + 1$
10: **end while**
11: **return** $U^{(t)}, S^{(t)}, V^{(t)}$
12: **end function**

A análise de complexidade de tempo do algoritmo 6 difere das demais apresentadas. Se for usado um algoritmo iterativo para o cálculo das matrizes inversas, aproveitando o fato que ambas contém elementos diferentes de 0 na diagonal principal, é possível chegar no

seguinte resultado: $\mathcal{O}\Big(t_{max}\big(nl(m+k)+kl(n+m)+nk+ml\big)\Big) \simeq \mathcal{O}\Big(t_{max}\big(n^2k+k^2n+nk\big)\Big)$. Já é possível perceber que a complexidade do algoritmo é menor que a complexidade dos algoritmos 3, 4 e 5. Isso se explica pois a atualização de U e V é feita de forma iterativa. Ainda, as multiplicações de matrizes para o cálculo dos centróides $\widetilde{U}$ e $\widetilde{V}$, e para atualização de S, que podem ser calculadas de forma iterativa, melhorando ainda mais a complexidade de tempo do algoritmo.

3.4 Considerações finais

É importante ressaltar, que nenhum dos algoritmos apresentados são capazes de garantir convergência para um mínimo global dos problemas apresentados, então, é possível encontrar diversas soluções em execuções diferentes dos algoritmos, devido à inicialização de forma aleatória. Essa é uma limitação também presente nos algoritmos de agrupamento clássicos.

Além disso, a fim de melhor motivar a proposta dos novos algoritmos apresentados no capítulo 4, é interessante compreender os algoritmos *ONMTF* e *FNMTF* em termos de suas capacidades de quantização do espaço dos dados e de geração de informação sobre os dados e em termos do processo de descoberta de cogrupos.

Do ponto de vista de quantização do espaço dos dados, a quantidade de informação que precisa ser armazenada é dependente da organização da matriz S, ou seja, do número de grupos de linhas k e do número de colunas l necessários para explicar a matriz de dados original. Como será discutido mais à frente neste trabalho, para determinados tipos de organizações de cogrupos, especificamente aqueles em que há sobreposição de colunas nos grupos de colunas, os algoritmos implementados sob essas estratégias exigem uma quantidade de grupos de colunas (l) que pode ser maior do que o número de grupos de colunas desejados. Os algoritmos propostos têm o objetivo de superar essa limitação, permitindo que a quantização do espaço seja mais próxima da desejada em termos de grupos de colunas.

Do ponto de vista de geração de informação sobre os dados, os algoritmos *ONMTF* e *FNMTF* são capazes de fornecer como as linhas da matriz se organizam em grupos e como as colunas da matriz se organizam em grupos. Transferindo essa informação para um contexto de aplicação, significa dizer que os algoritmos são capazes de explicar como os dados se organizam no espaço (matriz U) e, intuitivamente e sob uma forma de organização,

como grupos de atributos desses dados (matriz V) podem estar associados (por meio da matriz S) à essa organização dos dados. Esse tipo de informação, também presente nos algoritmos propostos, não é explicitamente fornecida por algoritmos de agrupamento como *k-means* e *fuzzy-k-means*, brevemente discutidos no capítulo 2.

Do ponto de vista do processo de descoberta dos cogrupos, os algoritmos *ONMTF* e *FNMTF* são capazes de considerar simultâneamente ambas organizações na resolução do problema de minimização do erro de quantização da matriz original. Porém, possuem um processo que pode ser caracterizado por um tipo de interdependência entre grupos de linhas, que é na realidade o causador da possibilidade de chegar a uma quantidade de grupos de colunas (l) maior do que é realmente necessário para explicar os dados que possuem uma organização de cogrupos com sobreposição de colunas. Esta é a segunda meta de superação obtida nos algoritmos propostos, que por sua formulação, são capazes de resolver o problema de fatoração das matrizes de maneira independente para cada grupo de linhas.

4 Fatoração de matrizes não-negativas para coagrupamento com sobreposição de colunas

Como discutido no capítulo 3, a fatoração de matrizes aplicada ao problema de agrupamento ou coagrupamento pode ser analisada sob, pelo menos, três aspectos: quantização do espaço dos dados; geração de informação sobre os dados; e pelo processo de descoberta de cogrupos. Também, para cada uma dessas análises, diferentes estratégias apresentam vantagens e desvantagens.

Com o intuito de apresentar uma alternativa às estratégias de fatoração de matrizes não-negativas para coagrupamento presentes na literatura, objetivando superar algumas das dificuldades apresentadas por eles, propõe-se duas novas estratégias que adicionam k matrizes V na fatoração, ao invés de uma única matriz V. Basicamente, cada uma dessas matrizes representará uma organização de cogrupos de colunas independente, de maneira que aumentar-se-á a flexibilidade para o estabelecimento de relações entre cogrupos de linhas e cogrupos de colunas. As estratégias são:

- *OvNMTF*: uma estratégia de fatoração tripla de matrizes não-negativas com sobreposição de colunas, baseada na estratégia *BVD*;
- *BinOvNMTF*: uma estratégia de fatoração binária tripla de matrizes não-negativas com sobreposição de colunas, baseada na estratégia *FNMTF*.

Sob o ponto de vista de resolução de problemas de coagrupamento, o objetivo dessas estratégias é o mesmo das estratégias já apresentadas neste texto, qual seja, encontrar grupos de linhas e colunas de forma simultânea. Entretanto, visto que se tem a liberdade de organizar um conjunto de matrizes V, que abstrai grupos e colunas, para ser associado a cada grupo de linhas, que abstrai os grupos de linhas, é possível que grupos de colunas diferentes sejam formados, a depender do grupo de linhas sendo considerado. Isso significa que problemas em que há sobreposição de colunas na formação de cogrupos poderão ser adequadamente tratados pelas estratégias propostas.

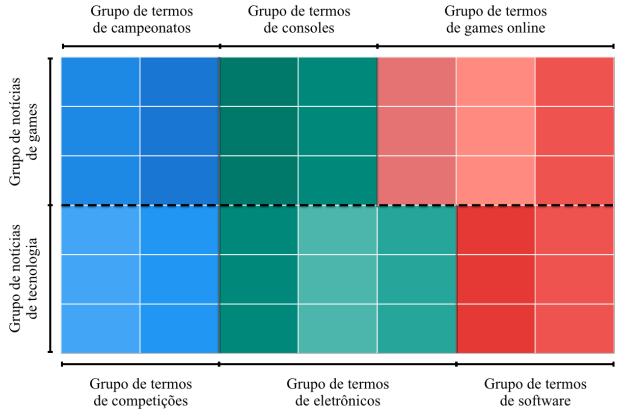
Formalmente, considere uma matriz de dados $X \in \mathbb{R}^{n \times m}$ contendo números reais positivos com n linhas e m colunas, formada por um conjunto de vetores de linhas $\mathcal{N} = \{\mathbf{x}_{1\cdot}, \ldots, \mathbf{x}_{n\cdot}\}$ e um conjunto de vetores de colunas $\mathcal{M} = \{\mathbf{x}_{\cdot 1}, \ldots, \mathbf{x}_{\cdot m}\}$, e as relações existentes entre cada linha x e cada coluna y são representadas por x_{ij} considerando os índices $i \in \{1, \ldots, n\}$ e $j \in \{1, \ldots, m\}$. Cada valor em x_{ij} representa, então, a relação existente entre pares de elementos em algum contexto de interesse. O objetivo da tarefa

de coagrupamento sob a estratégia de coagrupamento é encontrar k partições de $\mathcal{N}$, denotadas pelos subconjuntos ordenados $\mathcal{K}_p \subseteq \mathcal{N}$, $k \times l$ partições para $\mathcal{M}$ em cada $\mathcal{K}_p$, denotadas pelos subconjuntos ordenados $\mathcal{L}_{pq} \subseteq \mathcal{M}$, considerando os índices $p \in \{1, \ldots, k\}$ e $q \in \{1, \ldots, l\}$. Então, os subconjuntos $\{\mathcal{K}_1, \ldots, \mathcal{K}_k\}$ e $\{\mathcal{L}_{11}, \ldots, \mathcal{L}_{1l}, \ldots, \mathcal{L}_{k1}, \ldots \mathcal{L}_{kl}\}$ são os cogrupos de linhas e colunas, respectivamente.

Observe na figura 7 uma representação gráfica da estrutura de cogrupos com sobreposição de colunas, contextualizado em uma aplicação de análise de textos. Note que, na realidade, trata-se de busca por uma solução adequada para um problema com sobreposição, ou seja, sobreposição de colunas ou de linhas, já que a sobreposição de linhas é justamente o problema transposto.

A figura 7 mostra dois grupos de documentos, dos assuntos games e tecnologia, com três documentos em cada. Ainda, este exemplo contém 6 grupos de palavras, divididos igualmente entre os grupos de documentos. Então, pode-se dizer que os grupos de palavras entitulados “campeonatos”, “consoles” e “games online” caracterizam o grupo de documentos sobre games, assim como os grupos de palavras entitulados “competições”, “eletrônicos” e “software” caracterizam o grupo de documentos sobre tecnogia. Note que há sobreposição entre os primeiros três grupos de palavras do grupo de documentos com os três primeiros grupos de palavras do grupo de documentos de tecnologia. Também, é possível observar que apesar das palavras serem as mesmas para todos os documentos, como há independência entre os grupos de palavras para cada grupo de documentos, as palavras caracterizaram grupos de palavras diferentes, como no exemplo, os grupos de palavras entitulados como “games online” e “software”.

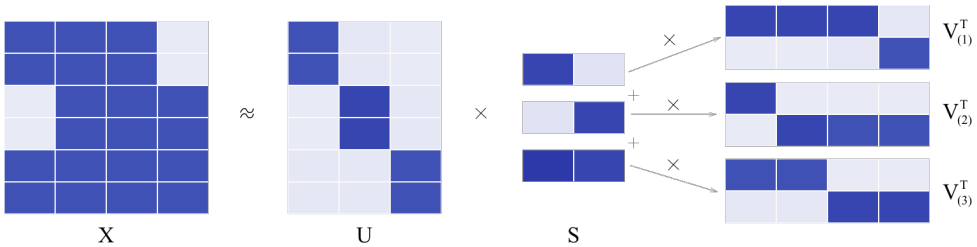
Figura 7 – Representação gráfica do problema de coagruamento com sobreposição de colunas e contextualização do domínio de documentos (notícias)



Fonte: Lucas Fernandes Brunialti, 2016

Da mesma maneira que foi ilustrado no capítulo 3 que é possível derivar interpretações intuitivas da análise das combinações de matrizes geradas na fatoração produzido pelos algoritmos lá discutidos, para os algoritmos discutidos no presente capítulo, interpretações análogas podem ser feitas, porém, consirando a existências das várias matrizes V. Assumindo novamente que uma matriz de entrada representa a relação "documento por palavras" (linhas por colunas), cada coluna das k matrizes $UI_{(p)}S, \forall p \in \{1, \ldots, k\}$, captura a ideia de uma base para representação de grupos de palavras, descritos nas matrizes $V_{(p)}$; e cada linha em $\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T$, captura a ideia de uma base de representação de grupos de documentos. A representação gráfica para o resultado de uma fatoração de matrizes que permite esse raciocínio é apresentada na figura 8.

Figura 8 – Fatoração da matriz original de dados X em cinco outras matrizes: U, S, V_1, V_2 e V_3



Fonte: Lucas Fernandes Brunialti, 2016

Todo o universo de interpretações delineado para os algoritmos da literatura podem ser estendidos para esse novo contexto de problema de agrupamento. No entanto, agora existem três matrizes para determinar os grupos de palavras, cada uma delas responsável por determinar os grupos de palavras para cada grupo de documentos. Na figura 8 considere que uma célula com cor escura representa a existência de uma relação entre linha e coluna, e uma célula com cor clara representa a inexistência de uma relação entre linha e coluna, e que essas relações são estabelecidas adequadamente em cada contexto de aplicação. Tem-se então, um conjunto de seis documentos e quatro palavras. A matriz U pode ser interpretada como descrito anteriormente, uma matriz "documentos por grupos de documentos", com seis documentos agrupados em três grupos ($k = 3$): os dois primeiros documentos no primeiro grupo, os dois próximos no segundo grupo e os dois últimos no terceiro grupo. Já as matrizes $V_{(p)}^T$ têm uma interpretação levemente diferente, ainda podem ser interpretadas como matrizes de "grupos de palavras por palavras", sendo dois grupos de palavras ($l = 2$), porém, a matriz $V_{(1)}^T$, por exemplo, contém os grupos de palavras para o primeiro grupo de documentos apenas (composto pelas linhas 1 e 2 de X), no primeiro

grupo de palavras estão as colunas 1, 2 e 3, e no segundo, a coluna 4. O mesmo raciocínio pode ser utilizado para as matrizes $V_{(2)}^T$ e $V_{(3)}^T$. Por fim, para a matriz S pode ser realizada a mesma interpretação, representando uma relação entre grupos de documentos e grupos de palavras, com a ressalva de que irão existir $k \times l$ grupos de palavras, ou seja, a linha p da matriz S irá representar a relação entre o p-ésimo grupo de documentos e os grupos de palavras encontrados em $V_{(p)}^T$.

O restante deste capítulo é destinado a apresentar a formulação dos problemas para cada uma das estratégias, incluindo a apresentação de uma derivação teórica para um algoritmo que implementa o processo de resolução para os problemas. Finalmente, as considerações finais apresentam uma discussão sobre o diferencial dessas estratégias no que diz respeito aos três aspectos citados no início do capítulo.

4.1 Fatoração Tripla de Matrizes Não-negativas com Sobreposição

Baseado no problema 3 (LONG; ZHANG; YU, 2005), o problema 6 é apresentado neste trabalho, e recebe aqui o nome de Fatoração Tripla de Matrizes Não-negativas Sobrepostas (Overlapped Non-negative Matrix Tri-factorization - *OvNMTF*).

Problema 6 (Problema de fatoração tripla de matrizes não-negativas sobrepostas)**.**

$$\begin{aligned} \mathcal{F}_6(U, S, V_{(1)}, \ldots, V_{(k)}) \quad = \quad & \min_{U,S,V_{(1)},\ldots,V_{(k)}} \quad \left\| X - U \sum_{p=1}^{k} I_{(p)} S V_{(p)}^T \right\|_F^2 \\ & \textit{suj. a} \qquad\qquad U \geq 0, S \geq 0, \\ & \qquad\qquad\quad V_{(p)} \geq 0, \quad \forall p \end{aligned}$$

em que $U \in \mathbb{R}_+^{n \times k}$, $S \in \mathbb{R}_+^{k \times l}$, $V_{(p)} \in \mathbb{R}_+^{m \times l}$, $p \in \{1, \ldots, k\}$ *é o índice para o conjunto de matrizes* $\{V_{(1)}, \ldots, V_{(k)}\}$, $I_{(p)} \in \{0, 1\}^{k \times k}$ *é uma matriz seletora com zeros em todos elementos exceto o elemento* $i_{(p)_{pp}}$ *que é igual à* 1, *e* $\|\cdot\|_F$ *denota a norma de Frobenius.*

Na formulação, o papel das matrizes $I_{(p)}$ seletoras é, justamente, organizar a base de grupos de linhas, de forma que cada um deles seja otimizado de acordo com uma das matrizes $V_{(p)}$.

É possível perceber, que neste caso, diferente dos apresentados no capítulo 3, têm menor capacidade de compactação, porém, com maior nível de detalhamento. Isso se explica, pois a partir dos nm elementos da matriz de dados, são gerados $nk + kl + klm$ para representá-los, diferente.

Desde que o problema 6 é semelhante ao problema 4, é esperado que ele possa também ser resolvido por meio de regras de atualização multiplicativas. Assim, a derivação das regras é aqui apresentada por meio de um abordagem baseada no cálculo do gradiente. Contudo, para o cálculo do gradiente de $\mathcal{F}_6$ a estratégia usada aqui é a apresentada em Yoo e Choi (2010), de forma que $\nabla\mathcal{F}_6 = [\nabla\mathcal{F}_6]^+ - [\nabla\mathcal{F}_6]^-$.

Expandindo $\mathcal{F}_6$, com base nas propriedade de traço de matrizes definidas no capítulo 2, para tornar o cálculo do gradiente mais simples, é possível obter:

$$\begin{aligned}\mathcal{F}_6 &= tr\big[(X - U\sum_{p=1}^{k} I_{(p)}SV_{(p)}^T)^T(X - U\sum_{p=1}^{k} I_{(p)}SV_{(p)}^T)\big] \\ &= tr(X^TX) - 2tr(X^TU\sum_{p=1}^{k} I_{(p)}SV_{(p)}^T) + tr(\sum_{p=1}^{k} V_{(p)}S^TI_{(p)}U^TU\sum_{p'=1}^{k} I_{(p')}SV_{(p')}^T)\end{aligned}$$

Note que a matriz seletora tem a seguinte propriedade: $I_{(p)}^T = I_{(p)}$.

Para o cálculo de $\nabla_U\mathcal{F}_6$, considere as partes positiva e negativa desse gradiente, $[\nabla_U\mathcal{F}_6]^+$ e $[\nabla_U\mathcal{F}_6]^-$, respectivamente. Usando a igualdade da equação 5, com $B = X^T$, $C = \sum_{p=1}^{k} I_{(p)}SV_{(p)}^T$ e $A = U$, é possível obter $[\nabla_U\mathcal{F}_6]^-$, como segue.

$$\begin{aligned}[\nabla_U\mathcal{F}_6]^- &= -2\nabla_U\Big(tr\big[X^TU\sum_{p=1}^{k} I_{(p)}SV_{(p)}^T\big]\Big) \\ &= -2X\sum_{p=1}^{k} V_{(p)}S^TI_{(p)}\end{aligned}$$

Para o cálculo de $[\nabla_U\mathcal{F}_6]^+$, é utilizado a igualdade da equação 6, com $B = \sum_{p=1}^{k} V_{(p)}S^TI_{(p)}$, $C = I$, $A = U^T$ e $D = \sum_{p'=1}^{k} I_{(p')}SV_{(p')}^T$. Então tem-se

$$\begin{aligned}[\nabla_U\mathcal{F}_6]^+ &= \nabla_U\Big(tr(X^TX) + tr\big[\sum_{p=1}^{k} V_{(p)}S^TI_{(p)}U^TU\sum_{p'=1}^{k} I_{(p')}SV_{(p')}^T\big]\Big) \\ &= U\sum_{p'=1}^{k} I_{(p')}SV_{(p')}^T\sum_{p=1}^{k} V_{(p)}S^TI_{(p)} \\ &\quad + U\sum_{p=1}^{k} I_{(p)}SV_{(p)}^T\sum_{p'=1}^{k} V_{(p')}S^TI_{(p')} \\ &= 2U\sum_{p=1}^{k}\sum_{p'=1}^{k} I_{(p)}SV_{(p)}^TV_{(p')}S^TI_{(p')}\end{aligned}$$

De forma similar, para o cálculo de $\nabla_S\mathcal{F}_6$, considere as partes positiva e negativa, $[\nabla_S\mathcal{F}_6]^+$ e $[\nabla_S\mathcal{F}_6]^-$, respectivamente. Usando a igualdade da equação 5 para todas as

Considere as seguintes igualdades para o cálculo dos gradientes, já definidas no capítulo 2, sendo A, B, C e D matrizes de quaisquers dimensões adequadas para a realização das multiplicações:

$$\begin{aligned}\nabla_A tr(BAC) &= A^TB^T \\ \nabla_{A^T} tr(BAC) &= CB\end{aligned} \tag{5}$$

$$\begin{aligned}\nabla_A tr(BACA^TD) &= B^TD^TAC^T + DBAC \\ \nabla_{A^T} tr(C^TA^TDAC) &= CA^TDB + C^TA^TB^TD^T\end{aligned} \tag{6}$$

partes da soma de $p \in \{1, \ldots, k\}$, com $B = X^T U I_{(p)}$, $A = S$, e $C = V_{(p)}^T$, é possível obter $[\nabla_S \mathcal{F}_6]^-$, como segue:

$$\begin{aligned}
[\nabla_S \mathcal{F}_6]^- &= -2\nabla_S \Big(tr\big[X^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[X^T U I_{(k)} S V_{(k)}^T\big] \Big) \\
&= -2\Big(I_{(1)} U^T X V_{(1)} + \cdots + I_{(k)} U^T X V_{(k)} \Big) \\
&= -2 \textstyle\sum_{p=1}^{k} I_{(p)} U^T X V_{(p)}
\end{aligned}$$

Para o cálculo de $[\nabla_S \mathcal{F}_6]^+$, é utilizado a igualdade da equação 6 para todas as partes da soma de $p, p' \in \{1, \ldots, k\}$, com $B = V_{(p)}$, $A = S^T$, $C = I_{(p)} U^T U I_{(p')}$ e $D = V_{(p')}^T$. Então tem-se

$$\begin{aligned}
[\nabla_S \mathcal{F}_6]^+ &= \nabla_S \Big(tr(X^T X) + tr\big[V_{(1)} S^T I_{(1)} U^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[V_{(1)} S^T I_{(1)} U^T U I_{(k)} S V_{(k)}^T\big] \\
&\quad + \cdots + tr\big[V_{(k)} S^T I_{(k)} U^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[V_{(k)} S^T I_{(k)} U^T U I_{(k)} S V_{(k)}^T\big] \Big) \\
&= \big(I_{(1)} U^T U I_{(1)} S V_{(1)}^T V_{(1)} + I_{(1)} U^T U I_{(1)} S V_{(1)}^T V_{(1)} \big) \\
&\quad + \cdots + \big(I_{(1)} U^T U I_{(k)} S V_{(k)}^T V_{(1)} + I_{(k)} U^T U I_{(1)} S V_{(1)}^T V_{(k)} \big) \\
&\quad + \cdots + \big(I_{(k)} U^T U I_{(1)} S V_{(1)}^T V_{(k)} + I_{(1)} U^T U I_{(k)} S V_{(k)}^T V_{(1)} \big) \\
&\quad + \cdots + \big(I_{(k)} U^T U I_{(k)} S V_{(k)}^T V_{(k)} + I_{(k)} U^T U I_{(k)} S V_{(k)}^T V_{(k)} \big) \\
&= 2 \textstyle\sum_{p=1}^{k} \sum_{p'=1}^{k} I_{(p)} U^T U I_{(p')} S V_{(p')}^T V_{(p)}
\end{aligned}$$

Finalmente, para o cálculo de $\nabla_{V_{(p)}} \mathcal{F}_6$, considere as partes positiva e negativa, $[\nabla_{V_{(p)}} \mathcal{F}_6]^+$ e $[\nabla_{V_{(p)}} \mathcal{F}_6]^-$, respectivamente. Usando a igualdade da equação 5, com $B = X^T U I_{(p)} S$, $A = V_{(p)}^T$, e $C = I$, $\forall p$, é possível obter $[\nabla_{V_{(p)}} \mathcal{F}_6]^-$, da seguinte forma:

$$\begin{aligned}
[\nabla_{V_{(p)}} \mathcal{F}_6]^- &= -2\nabla_{V_{(p)}} \Big(tr\big[X^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[X^T U I_{(k)} S V_{(k)}^T\big] \Big) \\
&= -2 X^T U I_{(p)} S
\end{aligned}$$

Na derivação para $[\nabla_{V_{(p)}} \mathcal{F}_6]^+$ nota-se que há dois casos diferentes para a derivação. O caso em que $p = p'$ e o caso em que $p \neq p'$. Para o caso em que $p = p'$, é utilizada a igualdade da equação 6, com $B = I$, $A = V_{(p)}$, $C = S^T I_{(p)} U^T U I_{(p)} S$ e $D = I$, $\forall p$, de forma que

$$\begin{aligned}
[\nabla_{V_{(p)}} \mathcal{F}_6]^+_{p=p'} &= \nabla_{V_{(p)}} \Big(tr\big[V_{(1)} S^T I_{(1)} U^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[V_{(k)} S^T I_{(k)} U^T U I_{(k)} S V_{(k)}^T\big] \Big) \\
&= \nabla_{V_{(p)}} \Big(tr\big[V_{(p)} S^T I_{(p)} U^T U I_{(p)} S V_{(p)}^T\big] \Big) \\
&= 2 V_{(p)} S^T I_{(p)} U^T U I_{(p)} S
\end{aligned}$$

Para o caso em que $p \neq p'$, é usada a igualdade da equação 5 em duas outras situações, aquela em que $V_{(p)}$ esta localizado à esquerda, então, $A = V_{(p)}$, $B = I$ e

$C = S^T I_{(p)} U^T U I_{(p')}$; e aquela em que $V_{(p)}$ esta localizado à direita, então, $A = V_{(p)}^T$, $B = V_{(p')} S^T I_{(p')} U^T U I_{(p)} S$, $C = I$, $\forall p, p'$. Assim,

$$\begin{aligned}
[\nabla_{V_{(p)}} \mathcal{F}_6]^+_{p \neq p'} &= \nabla_{V_{(p)}} \Big(tr\big[V_{(1)} S^T I_{(1)} U^T U I_{(2)} S V_{(2)}^T\big] + \cdots + tr\big[V_{(1)} S^T I_{(1)} U^T U I_{(k)} S V_{(k)}^T\big] \\
&\quad + \cdots + tr\big[V_{(k)} S^T I_{(k)} U^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[V_{(k)} S^T I_{(k)} U^T U I_{(k-1)} S V_{(k-1)}^T\big] \Big) \\
&= \textstyle\sum_{p' \neq p} \Big[\nabla_{V_{(p)}} \Big(tr\big[V_{(p)} S^T I_{(p)} U^T U I_{(p')} S V_{(p')}^T\big] \Big) \\
&\quad + \nabla_{V_{(p)}} \Big(tr\big[V_{(p')} S^T I_{(p')} U^T U I_{(p)} S V_{(p)}^T\big] \Big) \Big] \\
&= \textstyle\sum_{p' \neq p} 2\big(V_{(p')} S^T I_{(p')} U^T U I_{(p)} S \big)
\end{aligned}$$

Então, é possível calcular $[\nabla_{V_{(p)}} \mathcal{F}_6]^+$, $\forall p$, fazendo:

$$\begin{aligned}
[\nabla_{V_{(p)}} \mathcal{F}_6]^+ &= [\nabla_{V_{(p)}} \mathcal{F}_6]^+_{p=p'} + [\nabla_{V_{(p)}} \mathcal{F}_6]^+_{p \neq p'} \\
&= 2\big(V_{(p)} S^T I_{(p)} U^T U I_{(p)} S + \textstyle\sum_{p' \neq p} V_{(p')} S^T I_{(p')} U^T U I_{(p)} S \big) \\
&= 2\big(\textstyle\sum_{p'=1}^{k} V_{(p')} S^T I_{(p')} U^T U I_{(p)} S \big)
\end{aligned}$$

O resultado final dos gradientes para $U, S, V_{(p)}, \forall p \in \{1, \ldots, k\}$ são apresentados nas equações 7, 8 e 9, respectivamente.

$$\nabla_U \mathcal{F}_6 = 2\big(- X \textstyle\sum_{p=1}^{k} V_{(p)} S^T I_{(p)} + U \sum_{p=1}^{k} \sum_{p'=1}^{k} I_{(p)} S V_{(p)}^T V_{(p')} S^T I_{(p')} \big) \tag{7}$$

$$\nabla_S \mathcal{F}_6 = 2\big(- \textstyle\sum_{p=1}^{k} I_{(p)} U^T X V_{(p)} + \sum_{p=1}^{k} \sum_{p'=1}^{k} I_{(p)} U^T U I_{(p')} S V_{(p')}^T V_{(p)} \big) \tag{8}$$

$$\nabla_{V_{(p)}} \mathcal{F}_6 = 2\big(- X^T U I_{(p)} S + \textstyle\sum_{p'=1}^{k} V_{(p')} S^T I_{(p')} U^T U I_{(p)} S \big) \tag{9}$$

Sendo assim, as regras de atualização para as matrizes $U, V_{(p)}, S, \forall p \in \{1, \ldots, k\}$, são mostradas nas equações 10, 11 e 12, apresentadas no algoritmo 7, o qual implementa o processo de minimização para o problema 6. Nesse algoritmo t é o contador de iterações, $U^{(t)}$, $S^{(t)}$ e $V_{(p)}^{(t)}$ são as matrizes U, S e $V_{(p)}$, na iteração t, respectivamente, $\mathcal{U}(0,1) \in]0,1]$ uma função que gera valores de uma distribuição uniforme, e $\odot$ é o produto de Hadamard.

Algoritmo 7 Algoritmo baseado em atualização multiplicativa para solução do *OvNMTF*

function OvNMTF(X, k, l, t_{max})
  Inicialize: $U^{(0)} \leftarrow \mathcal{U}(0,1), S^{(0)} \leftarrow \mathcal{U}(0,1), V_{(p)}^{(0)} \leftarrow \mathcal{U}(0,1), \forall p$ e $t \leftarrow 0$.
  while (não convergiu) e ($t \leq t_{max}$) **do**

$$U^{(t+1)} \leftarrow U^{(t)} \odot \frac{\sum_{p=1}^{k} X V_{(p)}^{(t)} S^{(t)^T} I_{(p)}}{\sum_{p=1}^{k}\sum_{p'=1}^{k} U^{(t)} I_{(p)} S^{(t)} V_{(p)}^{(t)^T} V_{(p')}^{(t)} S^{(t)^T} I_{(p')}} \tag{10}$$

    for $p \leftarrow 1$ até k **do**

$$V_{(p)}^{(t+1)} \leftarrow V_{(p)}^{(t)} \odot \frac{X^T U^{(t+1)} I_{(p)} S^{(t)}}{\sum_{p'=1}^{k} V_{(p')} S^T I_{(p')} U^T U I_{(p)} S} \tag{11}$$

    end for

$$S^{(t+1)} \leftarrow S^{(t)} \odot \frac{\sum_{p=1}^{k} I_{(p)} U^{(t+1)^T} X V_{(p)}^{(t+1)}}{\sum_{p=1}^{k}\sum_{p'=1}^{k} I_{(p)} U^{(t+1)^T} U^{(t+1)} I_{(p')} S^{(t)} V_{(p')}^{(t+1)^T} V_{(p)}^{(t+1)}} \tag{12}$$

    $t \leftarrow t + 1$
  end while
  return $U^{(t)}, S^{(t)}, V_{(1)}^{(t)}, \ldots, V_{(k)}^{(t)}$
end function

A inicialização dos elementos das matrizes U, S e $V_{(1)}, \ldots, V_{(k)}$ são gerados através de uma distribuição uniforme ($\mathcal{U}(0,1) \in]0,1]$). Como condições para assumir a convergência, assim como os algoritmos apresentados no capítulo 3, considera-se a diferença do erro de aproximação entre duas iterações consecutivas menor ou igual a um ϵ:

$$\left\| X - U^{(t)} \sum_{p=1}^{k} I_{(p)} S^{(t)} V_{(p)}^{(t)^T} \right\|_F^2 - \left\| X - U^{(t+1)} \sum_{p=1}^{k} I_{(p)} S^{(t+1)} V_{(p)}^{(t+1)^T} \right\|_F^2 \leq \epsilon$$

O algoritmo também pára caso a quantidade de iterações (t) assuma o limite t_{max}.

A complexidade de tempo aproximada do algoritmo é possível ser calculada fixando as condições $k \simeq l$, $n \simeq m$, $k \ll n$, $l \ll m$ e usando um algoritmo para otimizar a ordem das multiplicações (Cormen et al. (2001) discute algoritmos para tal otimização): $\simeq \mathcal{O}\Big(t_{max}\big(n^2k^2 + nk + nk^2 + nk^3 + nk^4 + nk^5 + k^2 + k^4 + k^5\big)\Big) = \mathcal{O}\Big(t_{max}\big(n^2k^2 + nk^5 + k^5\big)\Big)$.

O particionamento, neste caso, não é direto. Então é necessário um processo de pós-processamento para a matriz U e para as matrizes $V_{(1)}, \ldots, V_{(k)}$. Um modo simples de obter o particionamento para as linhas, já descrito no capítulo 3, é o seguinte:

$$\mathcal{K}_p = \{x_{i\cdot} \mid i \in \{1, \ldots, n\} \text{ e } p = \underset{p' \in \{1,\ldots,k\}}{\arg\max}\, u_{ip'}\}, \forall p \in \{1, \ldots, k\}$$

Pode-se usar a mesma estratégia para o particionamento das colunas:

$$\mathcal{L}_{pq} = \{\mathbf{x}_{\cdot j} \mid j \in \{1,\ldots,m\} \text{ e } q = \arg\max_{q' \in \{1,\ldots,l\}} v_{(p)_{jq'}}\},$$

$$\forall j \in \{1,\ldots,m\}, \forall q, q' \in \{1,\ldots,l\}, \forall p \in \{1,\ldots,k\}$$

4.2 Fatoração Binária Tripla de Matrizes Não-negativas com Sobreposição

A segunda estratégia proposta nesta dissertação, segue os pressupostos estabelecidos por Wang et al. (2011) no problema 5. O problema 7, então denominado Fatoração Binária Tripla de Matrizes Não-negativas com Sobreposição (Overlapped Binary Non-negative Matrix Tri-factorization - *BinOvNMTF*) também está associado a diminuição da flexibilidade da solução de coagrupamento apresentada no que diz respeito à relação de associação entre grupos de linhas e grupos de colunas, a qual aqui é binária. Porém, mantém a independência do estabelecimento de diferentes bases para os grupos de linhas, uma vez que também assume a existência de k matrizes V.

Problema 7 (Problema de fatoração binária tripla de matrizes não-negativas sobrepostas)**.**

$$\begin{aligned} \mathcal{F}_7(U,S,V_{(1)},\ldots,V_{(k)}) \quad = \quad & \min_{U,S,V_{(1)},\ldots,V_{(k)}} \quad \left\| X - U\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T \right\|_F^2 \\ & \textit{suj. a} \qquad\qquad U \in \Psi^{n\times k}, \\ & \qquad\qquad V_{(p)} \in \Psi^{m\times l}, \quad \forall p, \\ & \qquad\qquad \sum_{p=1}^{k} u_{ip} = 1, \forall i, \\ & \qquad\qquad \sum_{q=1}^{l} v_{(p)_{jq}} = 1, \forall j \end{aligned}$$

em que $\Psi = \{0,1\}$, $p \in \{1,\ldots,k\}$ *e* $q \in \{1,\ldots,l\}$ *são os índices que iteram no número de linhas e colunas, respectivamente,* $I_{(p)} \in \{0,1\}^{k\times k}$ *é uma matriz identidade seletora, e* $\|\cdot\|_F$ *denota a norma de Frobenius para matrizes.*

Ainda, as restrições $\sum_{p=1}^{k} u_{ip} = 1$ e $\sum_{q=1}^{l} v_{(p)_{jq}} = 1, \forall p$ garantem que uma linha tem que pertencer à algum grupo, e uma coluna tem que pertencer à k grupos, respectivamente.

Em termos de compactação, a interpretação é semelhante à interpretação feita para o problema 6, porém, por causa das restrições serem binárias, os nm elementos da matriz de dados serão compactados em $n + kl + km$ elementos.

A mesma estratégia de derivação utilizada no algoritmo 6 também pode ser feita para propor uma solução para o problema 7: solucionar o problema para S para obter subproblemas que podem ser solucionados através de uma estratégia iterativa. Então, recapitulando o gradiente $[\nabla_S \mathcal{F}_6]^+$, já calculado para solução do problema 6, será o mesmo para $\mathcal{F}_7$, porém passível de simplificação, como (U^TU), para o problema 7 que tem restrições binárias em U, é uma matriz que contém zeros em todos elementos, com exceção da diagonal principal:

$$\begin{aligned}
[\nabla_S \mathcal{F}_7]^+ &= \sum_{p=1}^{k}\sum_{p'=1}^{k} I_{(p)}U^TUI_{(p')}SV_{(p')}^TV_{(p)} \\
&= \sum_{p=1}^{k}\sum_{p'=1}^{k}\left(I_{(p)}\begin{bmatrix}(U^TU)_{11} & & 0\\ & \ddots & \\ 0 & & (U^TU)_{kk}\end{bmatrix}I_{(p')}\right)SV_{(p')}^TV_{(p)}
\end{aligned}$$

Note que se $p \neq p'$, $I_{(p)}U^TUI_{(p)}$ e toda expressão, será uma matriz de zeros, por exemplo, se $p = 1$ e $p' = 2$:

$$I_{(1)}\begin{bmatrix}(U^TU)_{11} & & 0\\ & \ddots & \\ 0 & & (U^TU)_{kk}\end{bmatrix}I_{(2)} = \begin{bmatrix}1 & \dots & 0\\ \vdots & \ddots & \vdots\\ 0 & \dots & 0\end{bmatrix}\begin{bmatrix}(U^TU)_{11} & & 0\\ & \ddots & \\ 0 & & (U^TU)_{kk}\end{bmatrix}\begin{bmatrix}0 & \dots & 0\\ 0 & 1 & 0\\ \vdots & \ddots & \vdots\\ 0 & \dots & 0\end{bmatrix}$$

$$= [0]$$

$$\therefore \quad [\nabla_S \mathcal{F}_7]^+ = \sum_{p=1}^{k} I_{(p)}U^TUI_{(p)}SV_{(p)}^TV_{(p)}$$

Assim, é possível escrever o gradiente $\nabla_S \mathcal{F}_7$, a fim de encontrar uma expressão para atualização de S.

$$\begin{aligned}
\nabla_S \mathcal{F}_7 &= -2\sum_{p=1}^{k} I_{(p)}U^TXV_{(p)} + 2\sum_{p=1}^{k} I_{(p)}U^TUI_{(p)}SV_{(p)}^TV_{(p)} = 0 \\
&\implies \sum_{p=1}^{k} I_{(p)}U^TUI_{(p)}SV_{(p)}^TV_{(p)} = \sum_{p=1}^{k} I_{(p)}U^TXV_{(p)}
\end{aligned}$$

Como os termos $I_{(p)}U^TXV_{(p)}$ e $I_{(p)}U^TUI_{(p)}SV_{(p)}^TV_{(p)}$, $\forall p$, por serem multiplicados por $I_{(p)}$ pela esquerda, podem ser divididos em k equações, na qual cada equação será denotada por $\widetilde{S} = I_{(p)}S$. Observe que $I_{(p)}(U^TU)^{-1}I_{(p)} = I_{(p)}(U^TU)^{-1}$, dado a estrutura de (U^TU).

$$\begin{aligned}
I_{(p)}U^TU\widetilde{S}V_{(p)}^TV_{(p)} &= I_{(p)}U^TXV_{(p)} \\
\widetilde{S} &= I_{(p)}(U^TU)^{-1}I_{(p)}U^TXV_{(p)}(V_{(p)}^TV_{(p)})^{-1} \\
&= I_{(p)}(U^TU)^{-1}U^TXV_{(p)}(V_{(p)}^TV_{(p)})^{-1}
\end{aligned}$$

$$\therefore\ S = \sum_{p=1}^{k} I_{(p)}(U^T U)^{-1} U^T X V_{(p)} (V_{(p)}^T V_{(p)})^{-1}$$

É importante ressaltar que a mesma interpretação realizada para atualização de S no algoritmo 6 também é possível ser transferida para esse contexto.

Assim, é possível transformar o problema 7 em subproblemas para atualização de U e $V_{(1)}, \ldots, V_{(k)}$. Usando a mesma estratégia iterativa adotada no algoritmo 6: obtém-se os protótipos de linhas e verifica-se quais linhas de X mais se aproximam de cada um dos protótipos, para atualização de U; então, é feito o mesmo processo para atualização de $V_{(1)}, \ldots, V_{(k)}$, obtém-se os protótipos de colunas, relativo à $V_{(p)}, \forall p$, e verifica-se quais colunas de X mais se aproximam de cada um dos protótipos.

A implementação desse processo é apresentada no algoritmo 8. Considere t o contador de iterações, $U^{(t)}$, $S^{(t)}$ e $V_{(p)}^{(t)}$ são as matrizes U, S e $V_{(p)}$, na iteração t, respectivamente, $\mathcal{U}(0,1) \in]0,1]$ uma função que gera valores de uma distribuição uniforme, e $\|\cdot\|^2$ é a norma Frobenius para vetores.

Algoritmo 8 Algoritmo iterativo para solução do *BinOvNMTF*

function BinOvNMTF(X, k, l, t_{max})
  Inicialize: $U^{(0)} \leftarrow 0,1 \mid \sum_{p=1}^{k} u_{ip} = 1$, $V_{(p)}^{(0)} \leftarrow 0,1 \mid \sum_{q=1}^{l} v_{(p)_{jq}} = 1, \forall i, j, p$ e $t \leftarrow 0$.
  while (não convergiu) e ($t \leq t_{max}$) **do**

$$S^{(t+1)} \leftarrow \sum_{p=1}^{k} I_{(p)}(U^{(t)^T} U^{(t)})^{-1} U^{(t)^T} X V_{(p)}^{(t)} (V_{(p)}^{(t)^T} V_{(p)}^{(t)})^{-1} \tag{13}$$

$$\widetilde{V} \leftarrow \sum_{p=1}^{k} I_{(p)} S^{(t+1)} V_{(p)}^{(t)^T}$$

$$(U^{(t+1)})_{ip} \leftarrow \begin{cases} 1 & p = \arg\min_{p' \in \{1,\ldots,k\}} \|\mathbf{x}_{i\cdot} - \widetilde{\mathbf{v}}_{p'\cdot}\|^2 \\ 0 & \textit{caso contrário} \end{cases} \quad \forall i, p \tag{14}$$

$$\widetilde{U}_{(p)} \leftarrow U^{(t+1)} I_{(p)} S^{(t+1)}, \forall p$$

$$(V_{(p)}^{(t+1)})_{jq} \leftarrow \begin{cases} 1 & q = \arg\min_{q' \in \{1,\ldots,l\}} \left\| \mathbf{x}_{\cdot j} - \widetilde{\mathbf{u}}_{(p)\cdot q'} \right\|^2 \\ 0 & \textit{caso contrário} \end{cases} \quad \forall j, p, q \tag{15}$$

    $t \leftarrow t + 1$
  end while
  return $U^{(t)}, S^{(t)}, V_{(1)}^{(t)}, \ldots, V_{(k)}^{(t)}$
end function

Note que nesse tipo de fatoração, semelhante à apresentada no problema 5, não necessita de uma fase de pós-processamento para obter o particionamento de linhas e colunas, pois o mesmo é direto a partir das matrizes U e $V_{(1)}, \ldots, V_{(k)}$. Ainda, usa-se a mesma estratégia de convergência que a apresentada no algoritmo 7.

Semelhante ao *FNMTF*, se for usado um algoritmo iterativo para o cálculo das matrizes inversas, aproveitando o fato que ambas contém elementos diferentes de 0 na diagonal principal, é possível chegar na seguinte complexidade de tempo aproximada: $\simeq \mathcal{O}\Big(t_{max}\big(n^2k^2 + n^2k + nk^3 + nk^4 + k^2 + k^4\big)\Big) = \mathcal{O}\Big(t_{max}\big(n^2k^2 + nk^4 + k^4\big)\Big)$.

4.3 Considerações Finais

A motivação para a apresentação dos novos problemas de fatoração de matrizes (*OvNMTF* e *BinOvNMTF*), era a possibilidade de superar algumas das dificuldades apresentas nas fatorações da literatura (*ONMTF* e *FNMTF*), no que diz respeito à solução do problema de coagrupamento.

Do ponto de vista de quantização do espaço dos dados, a quantidade de informação armazenada nas novas fatorações tem o potencial de superar as fatorações da literatura. Isso ocorre porque nas fatorações propostas, o número de colunas l necessários para explicar a matriz de dados original deve ser mais próxima daquele necessário para obter o número de grupos de colunas desejado, mesmo para os casos em que há sobreposição de colunas entre as diferentes bases dos grupos de linhas. Entretanto, as fatorações propostas necessitam criar k matrizes para a abstração dos grupos de colunas, o que incorre em um custo maior de armazenamento dos protótipos dos grupos de colunas, e portanto, menor capacidade de compactação.

Do ponto de vista de geração de informação sobre os dados, as fatorações propostas são capazes de fornecer o mesmo tipo de informação: como as linhas da matriz de entrada se organizam em grupos e como as colunas da matriz de entrada se organizam em grupo. Porém, desde que se tem k organizações de grupos de colunas, sabe-se que há várias possibilidades dessa organização ocorrer para cada um dos grupos de linhas. Transferindo essa informação para um contexto de aplicação, entende-se que há um conjunto de organização de atributos que estão associados à organização assumida pelos dados no espaço dos dados. Intuitivamente, pode-se dizer que há diferentes maneiras de justificar o

agrupamento de dados descoberto, com base nas similaridades parciais dos atributos que os descrevem.

Do ponto de vista do processo de descoberta dos cogrupos, as fatorações propostas seguem a mesma ideia das fatorações na literatura, qual seja, considerar a informação de similaridade de dados e de atributos simultaneamente para resolver o problema de minimização do erro de quantização da matriz original, e consequentemente apresentar protótipos que expliquem os cogrupos. Porém, as fatorações propostas liberam os algoritmos de minimização da necessidade de considerar todos os grupos de linhas na otimização dos grupos de colunas. Desta forma, implementa-se um processo no qual não existe mais a interdependência entre grupos de linhas. Esse fato é que, na realidade, possibilita aproximar a quantidade de cogrupos utilizada para explicar os dados daquela que é a desejada, ainda que a sobreposição de colunas ocorra nos dados.

Em termos de complexidade de tempo, assim como esperado, os algoritmos para solução das fatorações propostas tem maior complexidade que os algoritmos da literatura. No entanto, o grau polinomial com maior diferença é em k, o que torna os algoritmos possíveis de serem utilizados em um problema real, já que normalmente utiliza-se valores de k de forma que $k \ll n$.

Essas considerações, assim como aquelas delineadas no capítulo 3, são ilustradas no próximo capítulo, no qual resultados experimentais são apresentados.

5 Experimentos e Resultados

Para fins de validação dos algoritmos propostos foram realizados experimentos utilizando bases de dados sintéticas e bases de dados textuais reais, sendo que sobre as bases de dados reais, foram criadas versões simplificadas para permitir análises mais detalhadas.

Esses experimentos foram projetados e executados com o fim de ilustrar as capacidades e limitações dos algoritmos de coagrupamento baseados em fatoração de matrizes presentes na literatura e dos algoritmos propostos neste trabalho, todos já apresentados nos capítulos 3 e 4, respectivamente. Tais capacidades e limitações são discutidas neste capítulo em termos de resultados obtidos sobre ambientes controlados (bases de dados sintéticas), ambientes semi-controlados (bases de dados textuais simplificadas) e ambientes aqui denominados não controlados (bases de dados textuais originais). O intuito com a experimentação desses algoritmos em bases de dados textuais é ilustrar seu desempenho como um método de resolução da tarefa de agrupamento de textos, da área de mineração de texto. Também, como os algoritmos clássicos de agrupamento, *k-means* e *fuzzy k-means*, são equivalentes à fatoração de matrizes (como foi mostrado no capítulo 2), estes foram aplicados sobre as mesmas bases de dados, para servir como base de comparação dos algoritmos *ONMTF* (algoritmo 5 de Yoo e Choi (2010)), *FNMTF*, *OvNMTF* e *BinOvNMTF*.

Para proporcionar uma visão organizada das capacidades e limitações dos algoritmos, primeiramente são apresentados os experimentos e os resultados obtidos com as bases de dados sintéticas. Tais resultados são apresentados em termos de qualidade de reconstrução, a qual é analisada com apoio de visualizacão gráfica, e de capacidade de *quantização*, a qual é avaliada em termos do erro de quantização dos algoritmos. Então, a capacidade de agrupamento é validada nos resultados obtidos com as bases de dados textuais reais originais e simplificadas, fazendo uso de medidas de qualidade de agrupamento. Para essas últimas, uma análise qualitativa é delineada de forma a ilustrar o valor agregado que a flexibilidade de modelos de coagrupamento pode trazer ao contexto de mineração de texto, mostrando o funcionamento do processo de geração de cogrupos.

5.1 Experimentos com bases de dados sintéticas

As bases de dados sintéticas foram criadas com inspiração nas diferentes estruturas de cogrupos, propostas por Madeira e Oliveira (2004), descritas com maior detalhamento no capítulo 2. Dentre as nove possíveis estruturas de cogrupos apresentadas por esses autores, foram escolhidas para uso nesses experimentos as três mais simples que, comprovadamente na literatura, os algoritmos *ONMTF* e *FNMTF* são capazes de tratar, e duas estruturas que apresentam sobreposição parcial de linhas ou colunas, que representam os casos que tais algoritmos não são capazes de tratar de forma natural, ou seja, usando os parâmetros de k e l, número de grupos de linhas e número de grupos de colunas, respectivamente, quando escolhidos manualmente analisando as matrizes da figura 9. Portanto, os casos que apresentam sobreposição parcial de colunas são o alvo dos algoritmos propostos neste trabalho (*OvNMTF* e *BinOvNMTF*). Na figura 9 são apresentas as visualizações gráficas de cada uma das estruturas de cogrupos em estudo.

Figura 9 – Dados sintéticos gerados a partir das diferentes estruturas de cogrupos. (a) Um único cogrupo. (b) Cogrupos com linhas e colunas sem sobreposição. (c) Cogrupos com estrutura em xadrez. (d) Cogrupos sem sobreposição nas linhas e com sobreposição nas colunas. (e) Cogrupos com sobreposição nas linhas e sem sobreposição nas colunas.

(a) (b) (c) (d) (e)

Fonte: Lucas Fernandes Brunialti, 2016

Para compreender a representação gráfica apresentada na figura 9 considere que cada um dos cinco quadrados maiores representa uma base de dados sintética, que a quantidade de dados da base está representada pela altura do quadrado, que a quantidade de atributos na base está representada pela largura do quadrado. Todas as bases de dados possuem 150 dados (linhas) e 150 atributos (colunas). Considere ainda que cada quadrado ou retângulo, representados com diferentes tons da cor azul, representam um cogrupo, também com suas alturas e larguras representando, respectivamente, a quantidade de linhas e colunas que compõem cada cogrupo. As diferentes tonalidades da cor azul revelam

a similaridade entre os valores assumidos pelos dados em subconjuntos de atributos, o que também revela a existência dos cogrupos. A intensidade da cor é proporcional à intensidade dos valores associados a cada atributo em cada dado, então valores maiores são representados por tonalidades mais escuras e vice-versa. Por exemplo, a seguinte matriz poderia representar o caso da figura 9b, porém em dimensões menores que 150 linhas por 150 colunas:

$$\begin{bmatrix}
\mathbf{30,3} & \mathbf{31,6} & \mathbf{30,9} & 1,0 & 0,8 & 0,9 & 0,7 & 0,7 & 0,1 \\
\mathbf{29,1} & \mathbf{30,7} & \mathbf{30,0} & 0,7 & 0,0 & 0,6 & 0,8 & 0,1 & 0,9 \\
\mathbf{30,8} & \mathbf{29,5} & \mathbf{31,5} & 0,2 & 0,7 & 0,2 & 0,9 & 0,7 & 0,5 \\
0,4 & 0,9 & 1,0 & \mathbf{10,5} & \mathbf{9,2} & \mathbf{10,8} & 0,8 & 0,8 & 0,8 \\
0,5 & 0,7 & 0,5 & \mathbf{11,1} & \mathbf{10,0} & \mathbf{9,2} & 0,9 & 0,7 & 0,6 \\
0,3 & 0,4 & 0,5 & \mathbf{10,8} & \mathbf{11,2} & \mathbf{10,9} & 0,5 & 0,3 & 0,1 \\
0,0 & 0,7 & 0,4 & 0,4 & 0,1 & 0,4 & \mathbf{20,2} & \mathbf{19,6} & \mathbf{20,4} \\
0,8 & 0,5 & 1,0 & 0,4 & 0,7 & 0,3 & \mathbf{21,2} & \mathbf{20,7} & \mathbf{19,4} \\
0,0 & 0,6 & 0,4 & 0,6 & 0,1 & 0,1 & \mathbf{19,9} & \mathbf{20,2} & \mathbf{20,9}
\end{bmatrix}$$

Todas as bases de dados da figura 9 são matrizes de valores reais positivos geradas artificialmente. Primeiramente, uma matriz de tamanho 150×150 é preenchida com valores ponto flutuante, gerados aleatoriamente a partir de uma função $\mathcal{U}(0,1)$, que gera números de uma distribuição uniforme no intervalo $]0,1]$. Em seguida, o tamanho em termo de linhas e colunas e disposição dos cogrupos na base de dados foram determinados de acordo com cada estrutura de cogrupos desejada. Para instanciar cada cogrupo, um conjunto de valores foi criado e distribuído pelas células do cogrupo da seguinte forma:

- um valor central $c \in \mathcal{C}$, sendo $\mathcal{C} = \{20, 40, 60, 80, 100, 120, 140, 160, 180\}$, foi aleatoriamente escolhido, e retirado de $\mathcal{C}$ para não haver cogrupos com o mesmo c em uma mesma matriz;
- o conjunto de valores usado para instanciar as células do cogrupo foi estabelecido por meio da adição de c a valores reais, gerados a partir da função $\mathcal{U}(0,10)$, que gera números de uma distribuição uniforme no intervalo $]0,10]$;
- cada um dos valores nesse conjunto foi atribuído às células previamente definidas como pertecentes ao cogrupo.

Assim, considerando que um cogrupo é uma submatriz da matriz original X, ele pode ser gerado pela equação:

$$x_{ij} = c + \mathcal{U}(0, 10)$$

sendo i e j os índices das linhas e colunas de X escolhidos para compor o cogrupo.

Ainda sobre a interpretação da figura 9, alguns detalhes precisam ser observados. A depender do objetivo da análise de coagruamento, na figura 9a, por exemplo, mais de um cogrupo pode ser observado, além daquele que é de interesse de análise neste trabalho. Para essa interpretação, considera-se que todo agrupamento de linhas e de colunas, independente de serem úteis ou não, ou de serem interesse para análise ou não, é um cogrupo. Assim, tem-se os seguintes cogrupos na base de dados sintética (a):

- O mais evidente, destacado em azul, é formado pelas linhas [60, ..., 109] e pelas colunas [70, ..., 139]. Esse é o cogrupo de interesse, nesse projeto, para a resolução da tarefa de coagrupamento aplicada a dados textuais, e é representado na figura 9a pelo quadrado em cor azul.
- O segundo, que não está destacado na visualização, é formado por todas as linhas e as colunas [0, ..., 69] e [140, ..., 149].
- O terceiro, também não destacado, é formado por todas as colunas e as linhas [0, ..., 59] e [110, ..., 149].

Sob essa ótica de interpretação, as demais bases de dados possuem:

- (b): três cogrupos de principal interesse e seis cogrupos não destacados na figura;
- (c): seis cogrupos de principal interesse;
- (d): três cogrupos de principal interesse e oito cogrupos não destacados na figura;
- (e): três cogrupos de principal interesse e oito cogrupos não destacados na figura.

Para cada uma das bases de dados sintéticas foram executados os seguintes algoritmos: *k-means*[1], *fuzzy k-means*[2], *ONMTF*, *FNMTF*, *OvNMTF* e *BinOvNMTF*[3]. Os resultados foram analisados em termos de qualidade de reconstrução e capacidade de quantização.

[1] Para os experimentos com *k-means* foi usada a implementação da biblioteca *scikit-learn* (PEDREGOSA et al., 2011) da linguagem Python.

[2] Para experimentos com *fuzzy k-means* foi usado a implementação do algoritmo *fuzzy k-means* da biblioteca *scikit-fuzzy* da linguagem Python.

[3] Os algoritmos baseadas em fatoração de matrizes foram implementados pelo autor deste trabalho usando a linguagem Python.

5.1.1 Análise da reconstrução

Uma forma de analisar o resultado dos algoritmos estudados neste trabalho é analisá-los quanto a sua capacidade de reconstrução. A reconstrucão é feita tomando as matrizes resultantes da fatoração e combinando-as da mesma forma que o seu problema foi proposto, de forma a reconstruir a matriz original.

Essa análise se torna importante quando se tem como objetivo avaliar o comportamento dos algoritmos em diferentes estruturas de organização de cogrupos, com destaque para a análise de maior interesse neste trabalho – coagrupamento com sobreposição parcial de linhas ou colunas.

Um resumo sobre os resultados obtidos nessa análise é apresentado na tabela 1. O restante dessa seção se destina a detalhar as análise de qualidade de reconstrução.

Tabela 1 – Resumo de qualidade de reconstrução: *ok* - permite reconstrução de forma natural; × - sem informação sobre sobreposição parcial; + - preserva informação de sobreposição parcial

	base (a)	**base (b)**	**base (c)**	**base (d)**	**base (e)**
k-means	*ok*	*ok*	*ok*	*ok*, ×	
fuzzy k-means	*ok*	*ok*	*ok*	*ok*, ×	+
ONMTF	*ok*	*ok*	*ok*	+	+
FNMTF	*ok*	*ok*	*ok*		
OvNMTF	*ok*	*ok*	*ok*	*ok*, +	*ok*, +
BinOvNMTF	*ok*	*ok*	*ok*	*ok*, +	

Fonte: Lucas Fernandes Brunialti, 2016

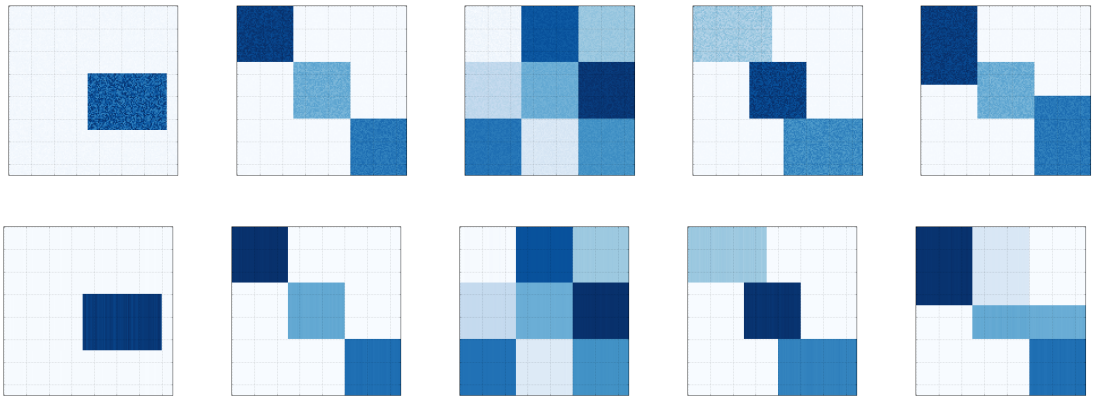
5.1.2 Reconstrução a partir dos resultados do algoritmo *k-means*

Para execução dos experimentos com o *k-means* os seguintes parâmetros foram estabelecidos: número máximo de iterações (300) ou a diferença do erro de quantização em duas iterações consecutivas ($\leq 1 \times 10^{-4}$), o que ocorrer primeiro, como critérios de parada; número de grupos $k = 2$ para a base de dados sintética (a) e $k = 3$ para as bases de dados sintéticas (b), (c), (d) e (e). A escolha de k foi realizada com a prerrogativa de que o algoritmo *k-means* deveria encontrar os grupos considerando todos os atributos descritivos, e desta forma, agrupar os dados de acordo com a similaridade total. Assim, k foi escolhido a partir da quantidade de agrupamento de linhas presente na base de dados.

A escolha de k maiores levaria o algoritmo a, necessariamente, dividir em grupos diferentes os dados que deveriam pertencer a um mesmo agrupamento de linhas. Foram executadas 10 rodadas do algoritmo, com inicialização de centróides aleatória, sendo que o melhor modelo resultante nessas rodadas foi escolhido para ilustrar a avaliação da qualidade da reconstrução.

As reconstruções obtidas a partir dos centróides encontrados pelo algoritmo *k-means* são ilustradas na figura 12. As matrizes originais são repetidas na figura, nas cinco primeiras posições, de forma a facilitar a análise visual dos resultados. Para um melhor entendimento da representação gráfica, note que cada linha recebe cores de acordo com sua pertinência a um agrupamento de linhas (ou grupo, na visão de agrupamento) representado por um centróide. Ou seja, se a linha 10 pertencer ao centróide 1, então os valores da linha 10 serão substituídos pelos valores do centróide 1. Note que o centróide é um vetor com o mesmo número de coordenadas de um dado da base de dados, sendo assim, a substituição é direta. O centróide que representa o agrupamento de linhas referente ao cogrupo de interesse (em azul na figura), possui, claramente, valores similares aos dados que pertencem a esse cogrupo, e por isso o procedimento proposto pode ser visto como uma representação de reconstrução.

Figura 10 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *k-means*.



(a) (b) (c) (d) (e)

Fonte: Lucas Fernandes Brunialti, 2016

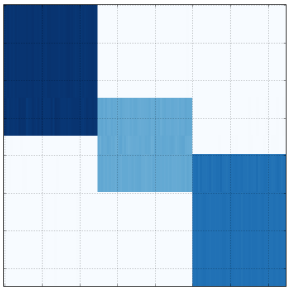
O modelo resultante da execução do *k-means* permite representar perfeitamente a reconstrução para as bases de dados (a) à (d). Porém, é preciso lembrar que não há informação sobre agrupamento de colunas no resultado do algoritmo, sendo que a

visualização gráfica de cada coagrupamento é possível apenas por conta do algoritmo levar em conta todos os atributos para realização do agrupamento.

O modelo não permite a descoberta de grupos com sobreposição de linhas (um dado não pode pertencer a mais de um grupo), e portanto não permite uma boa representação para a reconstrução no caso da base de dados (e). Esse resultado já era esperado devido à natureza da solução apresentada pelo *k-means*.

No entanto, se mudar o valor de k para descobrir 5 grupos ao invés dos 3 grupos desejados, é possível realizar a reconstrução da base de dados (e), como mostra a figura 11. Porém, não é uma reconstrução natural, ou seja, a informação que seria desejado é que há 3 grupos de linhas com sobreposição parcial entre o primeiro e o segundo grupos, e sobreposição parcial entre o segundo e terceiro grupos, diferentemente do obtido, que existe 5 grupos diferentes.

Figura 11 – Resultado da reconstrução da base de dados (e) utilizando o algoritmo *k-means* com $k = 5$.



Fonte: Lucas Fernandes Brunialti, 2016

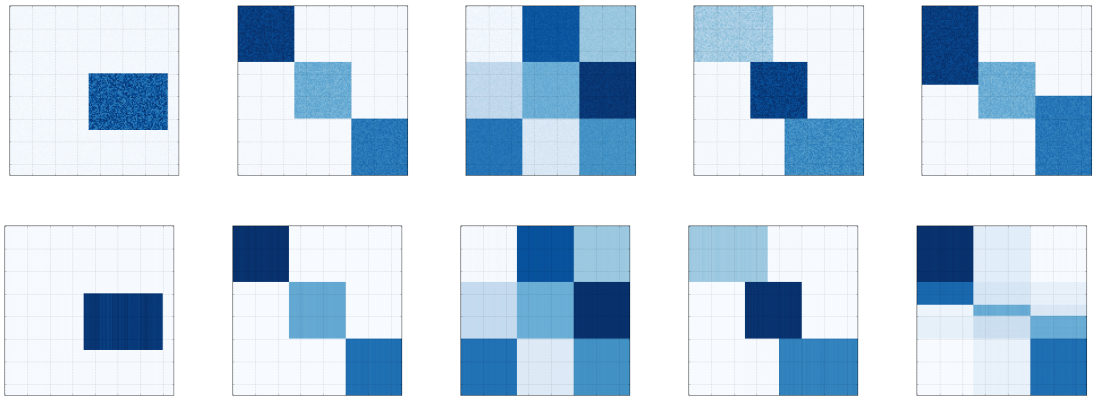
5.1.3 Reconstrução a partir dos resultados do algoritmo *fuzzy k-means*

Os mesmos critérios de parada e número de grupos usados para o *k-means* foram também usados nos experimentos com o *fuzzy k-means*. O valor para o parâmetro de fuzzificação m foi mantido em 2, como indicado em Rocha et al. (2012). Também, 10 execuções foram realizadas, com iniciação aleatória de pesos e o melhor modelo obtido foi usado para avaliação da qualidade de reconstrução obtida.

As reconstruções apresentadas na figura 12 foram obtidas por meio da multiplicação da matriz de partição pela matriz dos protótipos UC, resultantes da execução do algoritmo, e as matrizes originais foram repetidas para facilitar a análise visual. Note que a matriz U

do *fuzzy k-means* tem justamente o mesmo papel que a matriz de pertinências de linhas U das fatorações *BVD*, *ONMTF* e *OvNMTF*.

Figura 12 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algorimto *fuzzy k-means*.

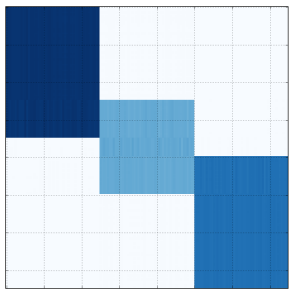


(a) (b) (c) (d) (e)

Fonte: Lucas Fernandes Brunialti, 2016

De forma semelhante à análise de reconstrução proporcionada pelo *k-means*, o *fuzzy k-means* também permite representar perfeitamente a reconstrução para as bases de dados (a) à (d). Porém, a reconstrução para o caso (e) não foi obtida com sucesso. No entanto, o algoritmo preservou parte da sobreposição, isso pode ser visto pelas regiões sombreadas onde há sobreposição de grupos.

Figura 13 – Resultado da reconstrução da base de dados (e) utilizando o algoritmo *fuzzy k-means* com $k = 5$.



Fonte: Lucas Fernandes Brunialti, 2016

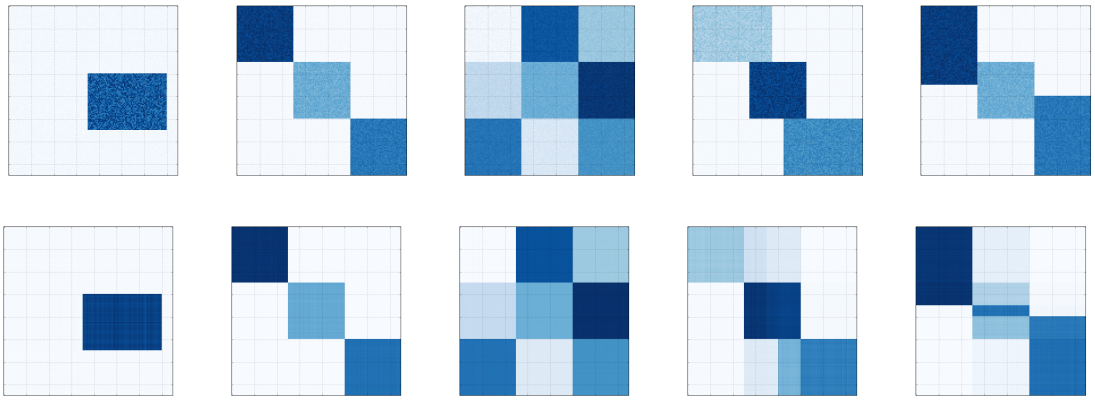
Da mesma forma que o *k-means*, é possível mudar o valor de k para 5, para assim, obter uma reconstrução perfeita (Figura 13). Fazendo as mesmas ressalvas, que a informação obtida a partir da interpretação desse modelo, não é a desejada.

5.1.4 Reconstrução a partir dos resultados do algoritmo *ONMTF*

Para execução dos experimentos com o ONMTF a seguinte parametrização foi estabelecida: número máximo de iterações (1000) ou a diferença do erro de quantização em duas iterações consecutivas ($\leq 1 \times 10^{-4}$), o que ocorrer primeiro, como critérios de parada; o número de cogrupos de linhas (k) e colunas (l) configurados da seguinte maneira: $k = l = 2$ para (a) e $k = l = 3$ (b), (c), (d) e (e). Novamente, as escolhas são baseadas no conhecimento *apriori* que se tem sobre a quantidade de cogrupos, e quais cogrupos, deseja-se obter. Para a visualização da reconstrução do algoritmo *ONMTF* mantendo os valores (e portanto as cores) das matrizes de dados, não foi realizada a normalização baseada em probabilidade proposta por Yoo e Choi (2010) (apresentada no algoritmo 5), pois esta muda a escala dos valores da reconstrução, dada a interpretação probabilística que pode ser feita.

A partir da realização de 10 execuções do algoritmo, foi escolhido o modelo que alcançou o menor erro de quantização e a partir do resultado obtido a reconstrução foi avaliada. A qualidade das resconstruções pode ser visualmente observada na figura 14. A resconstrução foi obtida a partir da multiplicação das matrizes fatoradas, ou seja, USV^T, conforme explicado no capítulo 3.

Figura 14 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *ONMTF*.



(a) (b) (c) (d) (e)

Fonte: Lucas Fernandes Brunialti, 2016

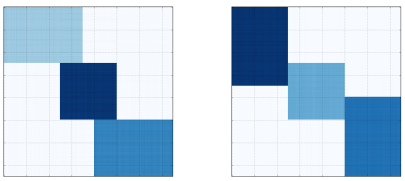
Analisando os resultados, é possível perceber que a reconstrução é realizada com êxito nos casos (a), (b) e (c). O algoritmo falhou em reconstruir a matriz no caso (d)

pois não foi capaz de associar corretamente as colunas que pertencem a mais de um agrupamento de colunas (sobreposição parcial nas colunas). O mesmo efeito ocorre com o caso (e), no qual há sobreposição parcial de linhas. A falha da reconstrução é percebida na coloração diferenciada, formando regiões com sombras, nas colunas, ou linhas, envolvidas nas interseções.

A coloração diferenciada daquela esperada (seguindo a matriz original) é resultante do estado da matriz V, que contém as relações de associação de linhas e/ou colunas aos cogrupos. Para o caso (d), por exemplo, é possível deduzir, a partir da análise visual, que há valores de magnitude diferentes estabelecendo a associação das colunas 50 à 79 e 80 à 99, que pertencem aos primeiro e terceiro cogrupos, respectivamente, com o cogrupo de colunas do meio. Ou seja, a reconstrução preservou parte da sobreposição. Esse comportamento é observado pois o algoritmo força a ortogonalidade entre cogrupos de linhas e cogrupos de colunas.

Ainda, é possível fazer a reconstrução perfeita, se mudar o valor de l para 5 na base de dados (d), e o valor de k para 5 na base de dados (e) (Figura 15). Porém, como nos outros casos, a interpretação do modelo gerado não é seria a desejada.

Figura 15 – Resultado da reconstrução da base de dados (d) com $k = 5$ e (e) com $l = 5$, respectivamente, utilizando o algoritmo *ONMTF*.



Fonte: Lucas Fernandes Brunialti, 2016

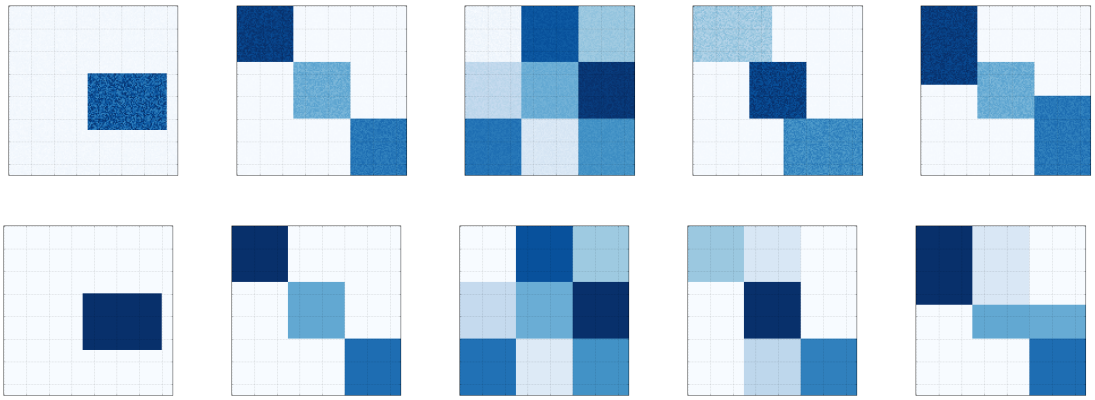
5.1.5 Reconstrução a partir dos resultados do algoritmo *FNMTF*

Para execução dos experimentos com o *FNMTF* a seguinte parametrização foi estabelecida: número máximo de iterações (300) ou a diferença do erro de minimização em duas iterações consecutivas ($\leq 1 * 10^{-4}$), o que ocorrer primeiro, como critérios de parada; o número de cogrupos de linhas (k) e colunas (l) configurados da seguinte maneira: $k = l = 2$ para (a), $k = l = 3$ para (b), (c), (d) e (e).

A partir da realização de 10 execuções do algoritmo, foi escolhido o modelo que alcançou o menor erro de quantização e a partir do resultado obtido a reconstrução foi

avaliada. A qualidade das reconstruções pode ser visualmente observada na figura 16. A reconstrução foi obtida a partir da multiplicação das matrizes fatoradas, ou seja, USV^T, conforme explicado no capítulo 3.

Figura 16 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *FNMTF*.



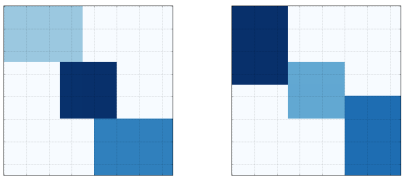
(a) (b) (c) (d) (e)

Fonte: Lucas Fernandes Brunialti, 2016

Este caso é semelhante ao anterior (algoritmo *ONMTF*). O algoritmo permitiu a reconstrução com êxito dos casos (a), (b) e (c), falhando na reconstrução dos casos (d) e (e), nos quais há sobreposição de colunas ou linhas nos cogrupos de interesse. No entanto, nesse caso o algoritmo restringe a associação de linhas (ou colunas), a apenas um agrupamento de linhas (ou de colunas), e por isso a informação referente a sobreposição em cogrupos é totalmente descaracterizada. Observe, por exemplo, que a reconstrução no caso (d) se assemelha à reconstrução do caso (c), embora as matrizes originais, em cada caso, representem informação sobre associação de dados e atributos em cogrupos também diferenciada.

Ainda, é possível fazer a reconstrução perfeita, se mudar o valor de l para 5 na base de dados (d), e o valor de k para 5 na base de dados (e) (Figura 15). Porém, como nos outros casos, a interpretação do modelo gerado não é seria a desejada.

Figura 17 – Resultado da reconstrução da base de dados (d) com $k = 5$ e (e) com $l = 5$, respectivamente, utilizando o algoritmo *FNMTF*.



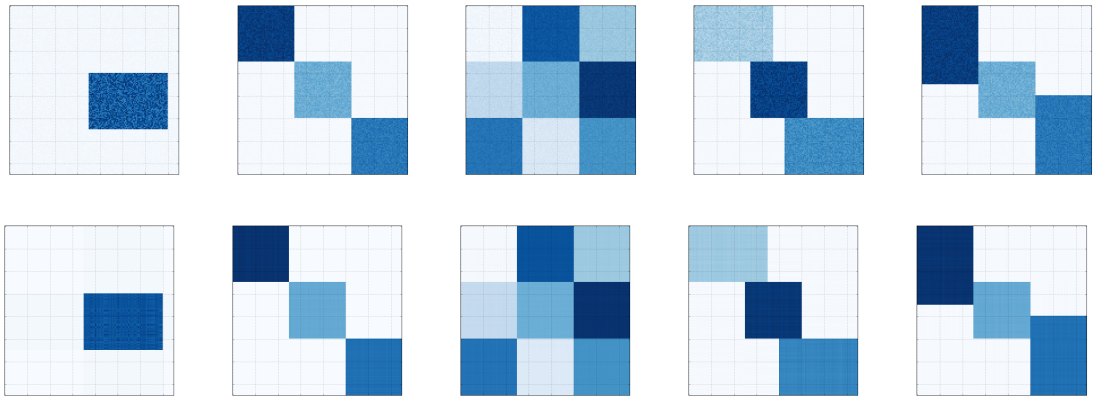
Fonte: Lucas Fernandes Brunialti, 2016

5.1.6 Reconstrução a partir dos resultados do algoritmo *OvNMTF*

Para execução dos experimentos com o *OvNMTF* a seguinte parametrização foi estabelecida: número máximo de iterações (1000) ou a diferença do erro de quantização ($\leq 1 * 10^{-4}$), o que ocorrer primeiro, como critérios de parada; o número de cogrupos de linhas (k) e colunas (l) configurados da seguinte maneira: $k = l = 2$ para (a), $k = 3$ e $l = 2$ para (b), (d) e (d), e $k = l = 3$ para (c). Note que o parâmetro l mudou em relação às outras reconstruções, pois no caso dos algoritmos propostos, para escolher o parâmetro l, é necessário responder a pergunta: quantos cogrupos de coluna existem para cada cogrupo de linhas?

A partir da realização de 10 execuções do algoritmo, foi escolhido o modelo que alcançou o menor erro de quantização e a partir do resultado obtido a reconstrução foi avaliada. A qualidade das reconstruções pode ser visualmente observada na figura 18. A reconstrução foi obtida a partir da multiplicação das matrizes fatoradas, ou seja, $U \sum_{p=1}^{k} I_{(p)} S V_{(p)}^{T}$, conforme explicado no capítulo 4.

Figura 18 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *OvNMTF*.



(a) (b) (c) (d) (e)

Fonte: Lucas Fernandes Brunialti, 2016

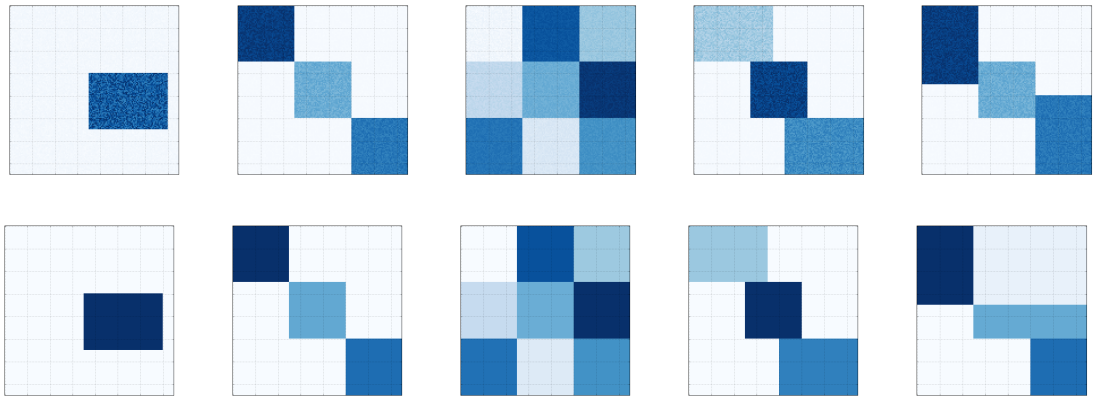
O algoritmo conseguiu realizar a reconstrução com êxito em todos os casos. Note que o algoritmo *OvNMTF* é capaz de lidar com cogrupos com sobreposição de colunas, e portanto é capaz de resolver o caso (d). Ainda, como não há restrições de ortogonalidade e existe fatores para controlar a pertinência de uma mesma linha ou coluna à multiplos cogrupos, o algoritmo é capaz de recontruir perfeitamente a base de dados (e).

5.1.7 Reconstrução a partir dos resultados do algoritmo *BinOvNMTF*

Para execução dos experimentos com o *BinOvNMTF* a seguinte parametrização foi estabelecida: número máximo de iterações (300) ou a diferença do erro de quantização ($\leq 1 * 10^{-4}$), o que ocorrer primeiro, como critérios de parada; o número de cogrupos de linhas (k) e colunas (l) configurados da mesma maneira que a reconstrução com o algoritmo *OvNMTF*: $k = l = 2$ para (a), $k = 3$ e $l = 2$ para (b), (d) e (d), e $k = l = 3$ para (c).

A partir da realização de 10 execuções do algoritmo, foi escolhido o modelo que alcançou o menor erro de quantização e a partir do resultado obtido a reconstrução foi avaliada. A qualidade das reconstruções pode ser visualmente observada na figura 19. A reconstrução foi obtida a partir da multiplicação das matrizes fatoradas, ou seja, $U \sum_{p=1}^{k} I_{(p)} S V_{(p)}^{T}$, conforme explicado no capítulo 4.

Figura 19 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *BinOvNMTF*.



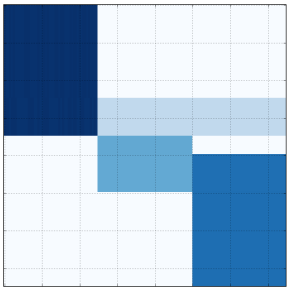
(a) (b) (c) (d) (e)

Fonte: Lucas Fernandes Brunialti, 2016

O algoritmo *BinOvNMTF* conseguiu realizar a reconstrução com êxito dos casos (a), (b), (c) e (d). Como o algoritmo é capaz de lidar com cogrupos com sobreposição de colunas, ele é capaz de resolver o caso (d). No entanto, o algoritmo não é capaz de lidar com cogrupos com sobreposição nas linhas, então, não é capaz de realizar a reconstrução do caso (e). E nesse caso, também não preserva qualquer informação sobre a sobreposição, já que possui a restrição de associação binária (cada linha/coluna associadas a um único agrupamento de linhas/colunas).

Ao contrário dos algoritmos *kmeans*, *fuzzy kmeans*, *ONMTF* e *FNMTF*, não é possível fazer a reconstrução perfeita utilizando $k = 5$ para a base de dados (e) (Figura 20). Uma possível explicação para isso é que o algoritmo, dada todas as suas restrições, é incapaz de encontrar a solução que permite a reconstrução perfeita para a base de dados (e).

Figura 20 – Resultado da reconstrução da base de dados (e) utilizando o algoritmo *BinOvNMTF* com $k = 5$.



Fonte: Lucas Fernandes Brunialti, 2016

5.1.8 Análise da capacidade de quantização

Nos experimentos com as bases de dados sintéticas, fazendo o uso da mesma parametrização descrita para a análise da reconstrução, foi feita a medida do erro de quantização. A tabela 2 sumariza o melhor erro de quantização resultante das 10 execuções de cada algoritmo para cada base de dados.

Tabela 2 – Avaliação da capacidade de quantização segundo o erro de quantização com os melhores destacados em negrito.

	base (a)	**base (b)**	**base (c)**	**base (d)**	**base (e)**
k-means	30.336, 0	62.776, 5	184.992, 8	**79.238, 5**	2.886.245, 5
fuzzy k-means	29.402, 8	63.768, 5	183.991, 4	**78.340, 0**	2.307.168, 6
ONMTF	30.555, 4	60.794, 8	184.255, 9	579.136, 7	781.131, 6
FNMTF	30.924, 7	64.636, 1	186.224, 4	1.634.328, 5	2.881.172, 0
OvNMTF	30.439, 8	61.863, 2	178.886, 8	**75.533, 6**	**75.931, 2**
BinOvNMTF	31.239, 0	63.660, 4	187.579, 8	**79.968, 0**	3.160.391, 0

Fonte: Lucas Fernandes Brunialti, 2016

Através dos resultados obtidos é possível perceber que a capacidade de quantização para todos os algoritmos para as bases de dados (a), (b) e (c), são semelhante, assim como a foi percebido na análise da reconstrução.

Para a base de dados (d), que contém cogrupos de colunas organizados com sobreposição parcial, os algoritmos que obtiveram erro de quantização entre 75.000 e 80.000 foram destacados na tabela 2, isso indica que eles foram capazes de quantizar a informação de sobreposição parcial. É importante perceber que capacidade de quantização é diferente de possibilidade de interpretação da quantização realizada, isto é, os algoritmos

kmeans e *fuzzy k-means* foram capazes de quantizar a sobreposição parcial de cogrupos de colunas, mas isso não quer dizer que é possível interpretar essa informação. Como foi mostrado na análise de reconstrução, o algoritmo *ONMTF* não quantizou toda a informação de sobreposição parcial da base de dados (d), mas parte dessa informação foi quantizada, diferentemente do algoritmo *FNMTF*, que perdeu totalmente essa informação. Isso também esta evidenciado na diferença do erro de quantização entre os algoritmos *ONMTF* e *FNMTF*.

O único algoritmo que foi capaz de quantizar a sobreposição parcial entre cogrupos de linhas da base de dados (e) foi o *OvNMTF*, isso ocorre pois não há restrição de ortogonalidade entre cogrupos. Da mesma forma que na base de dados (d), o algoritmo *ONMTF* não quantizou toda a informação de sobreposição, evidenciado pela diferença do erro de quantização obtido pelo algoritmo *OvNMTF*.

5.2 Experimentos com Bases de Dados Reais

Três bases de dados reais foram usadas a fim de ilustrar a aplicação dos algoritmos de coagrupamento no contexto da resolução de uma tarefa de mineração de texto. Dessas bases, uma (*NIPS14-17*) é base de referência, usada pela comunidade científica para experimentação de algoritmos de mineração de texto. Duas das bases de dados, *IG* e sua versão reduzida *IG toy*, foram elaboradas no contexto deste trabalho, e constituem-se como uma das contribuições desta pesquisa, já que podem constituir-se como mais duas bases de referência baseadas em dados reais, em língua portuguesa. Essas bases de dados estão descritas nessa seção, juntamente com o pré-processamento realizado sobre elas, e as análises quantitativas e qualitativas obtidas a partir da aplicação dos algoritmos de coagrupamento sobre elas.

5.2.1 Descrição das bases de dados

As três bases de dados reais são compostas por textos referentes à textos acadêmicos e notícias de portais postagem. O contexto referente a cada uma das bases é brevemente apresentados nesta seção, sendo que na tabela 3 são listadas algumas estatísticas para cada um deles. A esparsidade foi calculada da seguinte forma: $1 - \frac{\#\text{ Total de palavras}}{\#\text{ Documentos} \times \#\text{ Palavras únicas}}$.

1. ***NIPS14-17***[4] Esta base de dados contém uma coleção de trabalhos acadêmicos publicados no congresso *NIPS* (*Neural Information Processing Systems*) no período de 2001 a 2003, dos volumes 14 a 17. A construção da base de dados *NIPS14-17* foi realizada por *Sam Roweis*[5], a partir de um processamento aplicado aos dados adquiridos por *Yann LeCun* usando um dispositivo de reconhecimento ótico de caracteres (OCR) (GLOBERSON et al., 2007). A fonte de dados original, usada na construção desta base, possui os trabalhos científicos publicados em 18 volumes (0 a 17), porém apenas os trabalhos dos volumes 14 a 17 estão rotulados. Tais documentos estão organizados sob tópicos que compreendem áreas técnicas (teoria de aprendizado, neurociência, algoritmos e arquiteturas e etc), e estão distribuídos de forma desbalanceada entre os grupos caracterizados por cada um desses tópicos, por isso, foram usados documentos das 9 áreas técnicas com mais documentos, das 13 áreas técnicas originalmente disponibilizadas.
2. ***IG*** Esta base de dados foi criada como uma contribuição deste trabalho, e consiste em uma coleção de notícias extraídas do portal iG[6]. Cada documento nesta base contém o endereço eletrônico no qual a notícia está publicada, título, subtítulo, corpo e canal da notícia, sendo que o canal representa uma classificação para a notícia, atribuída pelos construtores do portal. As notícias que compõem essa base foram publicadas no período de 2 de janeiro de 2012 à 11 de outubro de 2014 e estão distribuídas em 13 canais, de maneira desbalanceada.
3. ***IG toy (ou reduzido)*** Esta base de dados é um subconjunto do conjunto de dados *IG*, composto por 300 notícias dispostas de forma balanceada em três canais (*esporte*, *arena* e *jovem*).

Tabela 3 – Estatísticas das bases de dados usadas nos experimentos.

Base de dados	**# Palavras únicas**	**# Total de palavras**	**# Documentos**	**# Grupos**	**Esparsidade**
NIPS14-17	6.881	746.826	555	9	0,804
IG	19.563	1.187.334	4.593	13	0,987
IG toy	6.764	70.169	300	3	0,965

Fonte: Lucas Fernandes Brunialti, 2016

[4] http://robotics.stanford.edu/~gal/data.html
[5] http://www.cs.nyu.edu/~roweis/data.html
[6] http://ig.com.br/

A extração das notícias do portal iG foi realizada através da implementação de um *web crawler* utilizando a linguagem Python[7]. As notícias foram capturadas a partir de uma página de início, fornecida para o *web crawler*, que era selecionada a fim de equalizar a distribuição de notícias por ano e por categoria (canal), mostradas nas tabelas 4 e 5. Todas as notícias coletadas para compor a base tem no mínimo 250 caracteres no corpo.

Tabela 4 – Distribuição de notícias por ano (base de dados *IG*).

Ano	# Notícias
2012	1.551
2013	1.933
2014	1.109

Fonte: Lucas Fernandes Brunialti, 2016

Tabela 5 – Distribuição de notícias por canal (base de dados *IG*)

Canal	**# Notícias**
economia	907
ultimosegundo	555
igirl	527
jovem	524
arena	421
tecnologia	359
esporte	342
delas	252
igay	210
gente	196
deles	141
saude	88
luxo	71

Fonte: Lucas Fernandes Brunialti, 2016

Analisando a distribuição de notícias por ano foi possível verificar que os links escolhidos como partida para o *web crawler* realizar a extração de notícias foram efetivos para deixar a distribuição perto de uniforme, com média e desvio padrão de aproximadamente 1.531 ± 337 notícias. Contrariamente, a distribuição de notícias por canal não ficou perto do uniforme, com média e desvio padrão de aproximadamente 353 ± 225 notícias. Uma hipótese é que isso se deve à idade e popularidade do canal, por exemplo, os canais *luxo* e *deles* são muito mais recentes e menos populares que o *ultimosegundo*.

[7] https://www.python.org/

Para a base de dados *NIPS*, da mesma forma, é mostrada, nas tabelas 6 e 7, as distribuições de trabalhos acadêmicos por ano e trabalhos acadêmicos por áreas técnicas, respectivamente. Note que os documentos rotulados como as seguintes áreas técnicas não foram usados nos experimentos: *Control & Reinforcement Learning*, *Vision (Biological)*, *Brain Imaging* e *Speech and Signal Processing*.

Tabela 6 – Distribuição de trabalhos acadêmicos por ano (base de dados *NIPS*)

Ano	**# Notícias**
2001	192
2002	204
2003	197

Fonte: Lucas Fernandes Brunialti, 2016

Tabela 7 – Distribuição de trabalhos acadêmicos por áreas técnicas (base de dados *NIPS*)

Áreas técnicas	**# Trabalhos acadêmicos**
Algorithms & Architectures	209
Cognitive Science & AI	68
Implementations	66
Vision	50
Neuroscience	47
Vision (Machine)	40
Emerging Technologies	33
Applications	26
Learning Theory	16
Control & Reinforcement Learning	15
Vision (Biological)	9
Brain Imaging	7
Speech and Signal Processing	7

Fonte: Lucas Fernandes Brunialti, 2016

5.2.2 Pré-processamento

A fim de estruturar a informação nas bases de dados para construção dos modelos, foi necessária uma fase de pré-processamento. As tarefas de pré-processamento comumente executadas em dados textuais são necessárias para melhorar a qualidade dos dados que serão submetidos à análise e também adequar a representação dos textos às necessidades dos algoritmos de análise (no caso deste trabalho, é necessário criar uma representação numérica e vetorial para os textos).

As ações executadas neste trabalho para realizar o pré-processamento dos textos foram:

- tokenização: o objetivo nesta ação é criar um "dicionário de termos" para a coleção de documentos. Para isso, um procedimento de quebra do texto em termos (*tokens* ou palavras) é realizado por meio da determinação de caracteres delimitadores (espaço em branco) e eliminação de caracteres que não se constituem como termos significativos para representação do texto (pontuação, caracteres especiais, etc). A decisão sobre como decidir os caracteres delimitadores e quais símbolos serão considerados insignificantes depende do contexto dos textos. Para a base de dados *IG* e *IG toy* foi usada uma expressão regular que separa caracteres não contíguos: (w+). Essa fase de pré-processamento não foi necessária para a base de dados *NIPS*, já que a base de dados é fornecida com essa etapa.
- filtragem: na filtragem são retirados os termos (*stopwords*) que não contribuem para a descrição ou identificação de um texto. Tradicionalmente, palavras das seguintes classes gramaticais são retiradas por esse filtro: conjunções, artigos, preposições e advérbios. Podem ser adicionadas a essa lista estão outras classes de palavras como numerais, nomes próprios, elementos da web, tokens monetários, e também palavras que aparecem em todos os textos por uma questão de padrão/formato dos mesmos. Algumas palavras foram acrescentadas à lista tradicional de *stopwords* no tratamento da base de dados IG: *leia*, *lendo*, *twitter*, *facebook*, *mais*, *divulgação*, *agnews*, *ler*, *ig*, *reprodução*, *siga*, *curta*, *images*, *imagens*, *veja*, *tambem*, *também*, *saiba*, *informações*, *thinkstock*, *getty*, *photos*, *foto*, *fotos*, *ap*, *gettyimages*, *infográfico*, *aí*. Para a base de dados *NIPS*, não foi necessário a etapa de filtragem de *stopwords*. Ainda, como parte da filtragem, palavras cuja frequência de ocorrência nos documentos é muito pequena ou muito grande, podem ser acrescentadas à lista. Em todas as bases usadas neste trabalho, todas as palavras com ocorrência menor ou igual à dois em todos os documentos, foram retiradas.

A representação vetorial objetivada é composta por uma relação de documentos e termos, organizada em modelo conhecimento como modelo do espaço vetorial, ou *Vector Space Model* (SALTON; WONG; YANG, 1975; SEBASTIANI, 2002; LOPS; GEMMIS; SEMERARO, 2011). Nesta representação, cada documento é representado por um vetor composto por tantas dimensões quanto forem os termos presentes no "dicionário de termos". Formalmente,

há um conjunto de n documentos $\{d_1, \ldots, d_n\}$, e um conjunto de m termos $\{t_1, \ldots, t_m\}$, e para cada par (d_i, t_j), sendo os índices $i \in \{1, \ldots, n\}$ e $j \in \{1, \ldots, m\}$, estabelece-se uma relação que expressa a maneira como um termo será usado na representação de um documento.

A relação $tf(t_j, d_i)$ expressa a frequência de ocorrência do termo t_j no documento d_i, e o vetor de representação de um documento, que será uma linha da matriz de dados X, e portanto, a entrada para os algoritmos, é representado por $\mathbf{x_{i\cdot}} = \left[tf(d_i, t_1), \ldots, tf(d_i, t_m)\right], \forall i$. Contudo, Salton, Wong e Yang (1975) mostram a partir de experimentação em diversos conjuntos de dados textuais, que o uso da relação conhecida como Frequência de Termos-Frequência de Documentos Inversa (*Term Frequency-Inversed Document Frequency - tfidf*) é capaz de melhorar a separação de documentos. Essa relação é calculado como descrito na equação 16, e tem o efeito de fazer com que a frequência dos termos que aparecem em muitos documentos seja enfraquecida, e a frequência dos termos que aparecem em alguns raros documentos seja fortalecida, gerando um efeito de normalização.

$$\begin{aligned} tfidf(d_i, t_j) &= tf(d_i, t_j) \times idf(t_j) \\ &= tf(d_i, t_j) \times \left(log_2 \frac{n}{df(t_j)+1}\right) \end{aligned} \tag{16}$$

em que $idf(t_j)$ representa a frequência de documentos inversa do termo t_j, e $df(t_j)$ a frequência de documentos que contém t_j.

Nos experimentos presentes neste trabalho, também é usada normalização norma-L_2 para que todos os vetores de termos $\mathbf{x}_{i\cdot}$ tenham comprimento iguais, ou seja, $\|\mathbf{x}_{i\cdot}\| = 1$. A partir dessa normalização aplicada aos vetores gerados com as relações $tf(d_i, t_j)$ e $tfidf(d_i, t_j)$, obtem-se respectivamente, as relações $tf_{norm}(d_i, t_j)$ e $tfidf_{norm}(d_i, t_j)$.

Sendo assim, para os experimentos presentes neste trabalho, foram utilizadas quatro formas de representação de documentos: *tf*, *tfidf*, tf_{norm} e $tfidf_{norm}$.

5.2.3 Análises quantitativas

Para avaliar os algoritmos propostos de forma quantitativa foram realizados experimentos considerando a tarefa de agrupamento de documentos.

Usando as classes presentes nas bases de dados *NIPS*, *IG* e *IG toy* foi possível avaliar e comparar os algoritmos *k-means*, *fuzzy k-means*, *ONMTF* (algoritmo 5 de Yoo e

Choi (2010)), *FNMTF*, *OvNMTF* e *BinOvNMTF* perante métricas clássicas de avalição de agrupamento (descritas no capítulo 2): índice de Rand e informação mútua normalizada.

A fim de organizar as informações sobre os experimentos e resultados, nesta seção, primeiramente são apresentadas as configurações usadas para esses experimentos para então apresentar os resultados obtidos para cada base de dados e uma discussão acerca dos resultados obtidos.

5.2.3.1 Configuração dos experimentos

Os algoritmos usados nesses experimentos foram os mesmos submetidos à experimentação com as base de dados sintéticas (seção 5.1), porém, as implementações para os algoritmos de coagrupamento foram alteradas para viabilizar a geração de modelos para as bases de dados reais. Para os experimentos com os algoritmos de agrupamento tradicional (*kmeans* e *fuzzy k-means*), não foi necessário o uso de outras implementações, visto que as disponíveis já viabizavam os testes pretendidos, já que tais algoritmos têm menor complexidade de tempo comparando com os algoritmos para coagrupamento.

Para os experimentos com os algoritmos de FM para coagrupamento (capítulos 3 e 4), foram feitas implementações usando a linguagem de programação *C++*[8]. Os algoritmos *FNMTF* e *BinOvNMTF* foram reimplementados com o apoio da biblioteca de álgebra linear *Armadillo*[9] (SANDERSON, 2010), para manipulação de vetores e matrizes (algebra linear). Como motor de multiplicação de matrizes dessa biblioteca, foram usadas as rotinas do software *OpenBLAS*[10] (*Open Basic Linear Algebra Subprograms* (WANG et al., 2013)), que proporcionaram o uso de computação paralela de alta performance.

Já os algoritmos *ONMTF* e *OvNMTF*, que envolvem intensas multiplicações de matrizes no processo de otimização, além da biblioteca *Armadillo* para manipulação de vetores e matrizes, também foi usada a plataforma *CUDA*[11] com a biblioteca *cuBLAS*[12] para multiplicação de matrizes. A plataforma *CUDA* é uma interface para interação com Unidades de Processamento Gráfico (*Graphics Processing Unit* - GPU) da *Nvidia*, então, como a sua arquitetura é massivamente paralela, problemas paralelizáveis como multiplicação de matrizes, são beneficiados em performance com esse hardware (FATAHA-

[8] https://isocpp.org/
[9] http://arma.sourceforge.net/
[10] http://www.openblas.net/
[11] https://developer.nvidia.com/cuda-zone
[12] http://docs.nvidia.com/cuda/cublas/

LIAN; SUGERMAN; HANRAHAN, 2004). A biblioteca *cuBLAS* fornece a mesma interface de rotinas para multiplicação de matrizes que as rotinas do software *OpenBLAS*. Então, o uso dessas estratégias foi uma maneira efetiva para melhorar a performance computacional dos algoritmos *ONMTF* e *OvNMTF*.

Ainda, com o intuito de melhorar o resultado da fatoração, para a representação numérica da matriz de dados fornecida de entrada para os algoritmos, foi utilizada a representação dupla precisão no formato de ponto flutuante (*float 64 bits*).

Na implementação dos algoritmos *ONMTF* e *OvNMTF*, foi utilizado um algoritmo para otimização da ordem das multiplicações de matrizes (CORMEN et al., 2001). Essa estratégia de implementação faz sentido já que, normalmente, $k \ll n$ e $l \ll m$, podendo diminuir consideravelmente o tempo de execução dos algoritmos. Para verificar a utilidade dessa estratégia, considere o seguinte exemplo de multiplicação de matrizes:

$$UU^TX$$

em que $U \in \mathbb{R}^{n \times k}$, $X \in \mathbb{R}^{n \times m}$, sendo $k \approx l$, $n \approx m$, $k \ll n$ e $l \ll m$. Se a ordem das multiplicacões for $(U(U^TX))$, serão realizadas $nmk + nk^2 \approx n^2k + nk^2$ operações de multiplicação de ponto flutuante; contrariamente, se a ordem das multiplicações for $((UU^T)X)$, serão realizadas $n^2k + n^2m \approx n^3 + n^2k$ operações, que são, claramente, mais operações que as $n^2k + nk^2$ operações.

Ainda na configuração dos experimentos para parametrização dos algoritmos, foi usado o número de classes presentes em cada base de dados, para determinar o número de grupos de documentos (k), seguindo, portanto, a seguinte configuração:

- *NIPS*: $k = 9$;
- *IG*: $k = 13$;
- *IG toy*: $k = 3$.

Já o número de grupos de termos (l), presentes nos algoritmos *ONMTF*, *FNMTF*, *OvNMTF* e *BinOvNMTF*, como não existe nenhuma evidência de quantos grupos de termos existem nas bases de dados, foram gerados modelos de agrupamento considerando os seguintes valores de l para cada base de dados:

- *NIPS*: $l \in \{6, 9, 12, 15, 18\}$;
- *IG*: $l \in \{7, 10, 13, 16, 19\}$;

- *IG toy*: $l \in \{2, 3, 4, 5, 6\}$.

Os valores de l foram escolhidos desta forma a fim de verificar como os algoritmos se comportariam para $l = k$, $l < k$ e $l > k$.

Para todas as bases de dados foram criadas quatro representações que serviram de entrada para os algoritmos: *tf*, *tfidf*, tf_{norm} e $tfidf_{norm}$, visando comparar o efeito que os algoritmos têm sob cada representação.

Como nenhum dos algoritmos é capaz de garantir um mínimo global para minimização do erro de quantização, foram gerados 10 modelos para cada configuração, lembrando que serão geradas diferentes soluções, devido à inicialização aleatória das matrizes U, C, S e V (como descrito na apresentação dos algoritmos nos capítulos 2, 3 e 4).

Como condições de parada dos algoritmos, foi configurado o número máximo de iterações como 1.000 para os experimentos com a base de dados *IG toy* e 10.000 para as bases de dados *NIPS* e *IG*. Essa decisão foi tomada com base na observação empírica que bases de dados com maior número de grupos de documentos necessitam de mais iterações para convergência. Também, o valor 1×10^{-4} para a diferença do erro de quantização em duas iterações consecutivas foi estabelecido como critério de parada adicional (se ele ocorrer primeiro).

5.2.3.2 Resultados

Como informado, 10 modelos de coagrupamento foram criados para cada parametrização dos experimentos quantitativos. Um exemplo de combinação é: representação *tf*, com $k = 3$, $l = 3$, para a base de dados *IG toy* e o algoritmo *ONMTF*. Os resultados para as três bases de dados reais são apresentados nesta seção.

Base de dados *IG toy*

Os resultados médios aplicando as métricas índice de Rand e informação mútua normalizada para a base de dados *IG toy* são sumarizados nas tabelas 8 e 9, e nas figuras 21 e 22. Analisando o conteúdo das tabelas 8 e 9, a primeira conclusão é que a avaliação obtida a partir do índice de Rand é consistente com a avaliação obtida a partir informação mútua

normalizada para esta base de dados. Na maioria dos casos, os melhores resultados médios representados pelas duas métricas, para todas as representações testadas, ocorrem no algoritmo *OvNMTF*. Considerando apenas esse algoritmo, os melhores resultados médios ocorrem para as representações *tf* e tf_{norm}, e com um número reduzido de grupos (l) de termos, quando comparado os número de grupos exigido pelas representações *tfidf* e $tfidf_{norm}$.

Tabela 8 – Índice de Rand médio por experimento com conjunto de dados *IG toy*: com $k = 3$, e variações de l e de representações para os textos

Algoritmo	*tf*	tf_{norm}	$tfidf_{norm}$	*tfidf*
k-means	$0,7017$	$0,7086$	$0,4701$	$0,3869$
fuzzy k-means	$0,4966$	$0,4970$	$0,4673$	$0,4390$
ONMTF	$0,3372 : l = 5$	$0,6479 : l = 5$	$0,5717 : l = 3$	$0,1758 : l = 3$
FNMTF	$0,2615 : l = 3$	$0,2590 : l = 3$	$0,1535 : l = 6$	$0,1543 : l = 3$
OvNMTF	$\mathbf{0,7466} : l = 4$	$\mathbf{0,7487} : l = 3$	$\mathbf{0,6755} : l = 6$	$\mathbf{0,6674} : l = 6$
BinOvNMTF	$0,4360 : l = 5$	$0,4818 : l = 5$	$0,2943 : l = 5$	$0,4079 : l = 3$

Fonte: Lucas Fernandes Brunialti, 2016

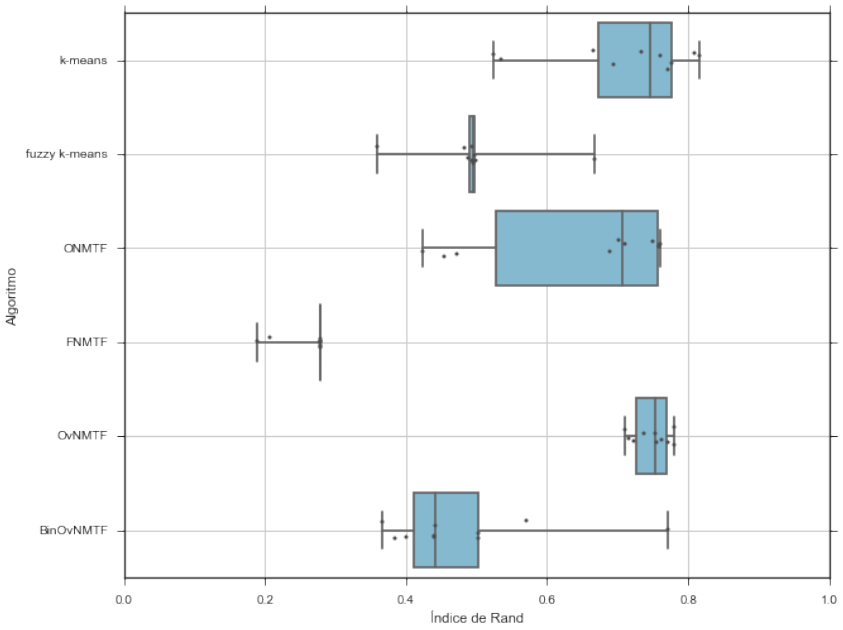
Tabela 9 – Informação mútua normalizada média por experimento com conjunto de dados *IG toy*: com $k = 3$, e variações de l e de representações para os textos

Algoritmo	*tf*	tf_{norm}	$tfidf_{norm}$	*tfidf*
k-means	$0,7169$	$0,7134$	$0,5467$	$0,4668$
fuzzy k-means	$0,0694$	$0,5421$	$0,4701$	$0,0836$
ONMTF	$0,3704 : l = 5$	$0,6720 : l = 5$	$0,6411 : l = 3$	$0,2143 : l = 3$
FNMTF	$0,2770 : l = 3$	$0,2734 : l = 3$	$0,2039 : l = 6$	$0,1690 : l = 3$
OvNMTF	$\mathbf{0,7257} : l = 4$	$\mathbf{0,7288} : l = 3$	$\mathbf{0,7033} : l = 6$	$\mathbf{0,6964} : l = 6$
BinOvNMTF	$0,4975 : l = 5$	$0,5500 : l = 5$	$0,3559 : l = 5$	$0,4343 : l = 3$

Fonte: Lucas Fernandes Brunialti, 2016

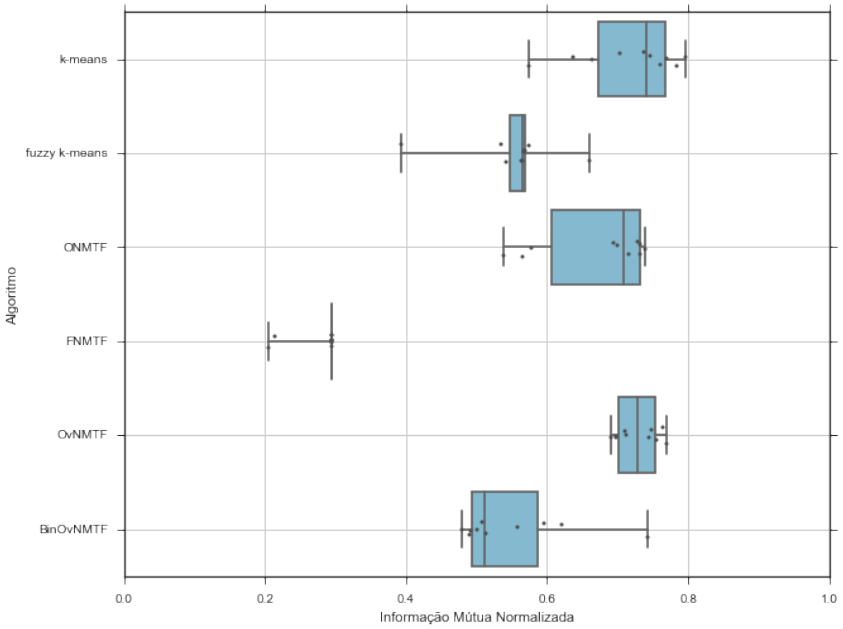
O algoritmo *OvNMTF*, proposto nesse trabalho, além de apresentar os melhores resultados médios nos experimentos com a base de dados *IG toy*, também é o que apresentou maior estabilidade, como pode ser observado nas tabelas e especialmente nas figuras 21 e 22. Essas figuras mostram as distribuições das medidas índice de Rand e informação mútua normalizada para os melhores resultados médios para cada algoritmo. Essa afirmação também é suportada pela análise das figuras 23 e 24, as quais organizam os resultados das avaliações sob as métricas índice de Rand e informação mútua normalizada, respectivamente, considerando os resultados obtidos para todos os modelos.

Figura 21 – Distribuições dos valores do índice de Rand para os melhores resultados médios para cada algoritmo na base de dados *IG toy*



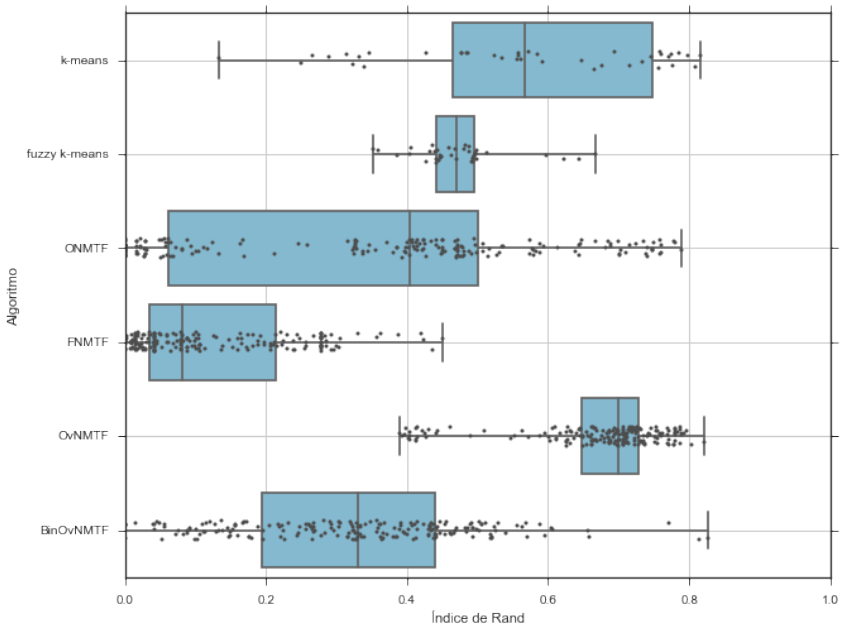
Fonte: Lucas Fernandes Brunialti, 2016

Figura 22 – Distribuições dos valores de informação mútua normalizada para os melhores resultados médios para cada algoritmo na base de dados *IG toy*



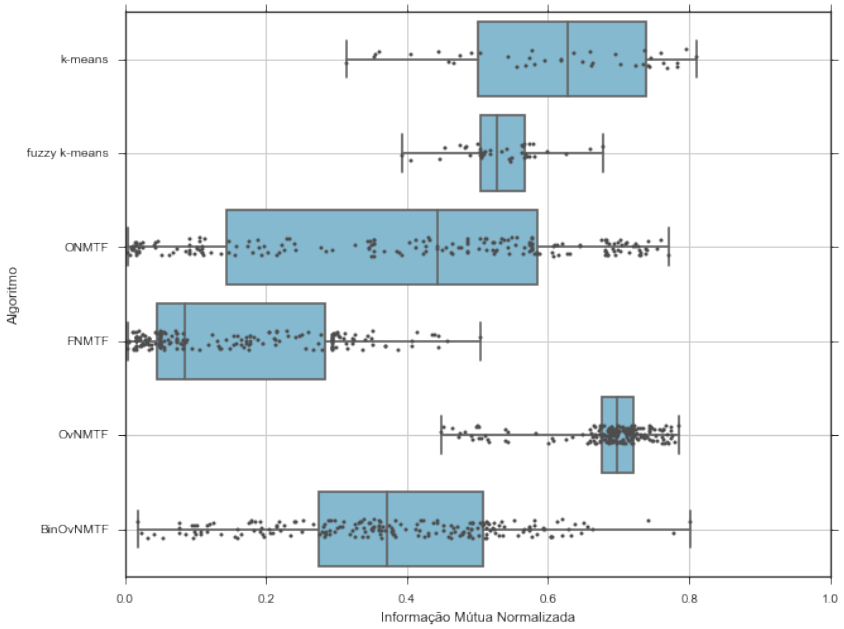
Fonte: Lucas Fernandes Brunialti, 2016

Figura 23 – Distribuições dos valores do índice de Rand para todas as execuções para cada algoritmo na base de dados *IG toy*



Fonte: Lucas Fernandes Brunialti, 2016

Figura 24 – Distribuições dos valores de informação mútua normalizada para todas as execuções para cada algoritmo na base de dados *IG toy*



Fonte: Lucas Fernandes Brunialti, 2016

Analisando os melhores resultados obtidos, ao invés do resultado médio, os algoritmos que apresentam melhor desempenho são o *BinOvNMTF* considerando o índice de Rand, e o *k-means* considerando o índice informação mútua normalizada, embora os resultados sejam semelhantes. A tabela 10 lista esses resultados.

Tabela 10 – Melhores (máximos) resultados obtidos para o conjunto de dados *IG toy* com $k = 3$, para cada algoritmo.

Algoritmo	**Índice de Rand**	**Informação Mútua Normalizada**
k-means	$0,8152 : tf_{norm}$	$\mathbf{0,8110} : tf_{norm}$
fuzzy k-means	$0,6669 : tf_{norm}$	$0,6785 : tf$
ONMTF	$0,7885 : l = 5,\ tfidf_{norm}$	$0,7719 : l = 5,\ tfidf_{norm}$
FNMTF	$0,4502 : l = 4,\ tfidf$	$0,5041 : l = 4,\ tfidf$
OvNMTF	$0,8208 : l = 2,\ tf_{norm}$	$0,7855 : l = 2,\ tf_{norm}$
BinOvNMTF	$\mathbf{0,8261} : l = 3,\ tfidf$	$0,8024 : l = 3,\ tfidf$

Fonte: Lucas Fernandes Brunialti, 2016

No que diz respeito às representações usadas para os textos, observa-se que as representações normalizada se destacam. As tabelas 8 e 9 mostram que os algoritmos que têm como entrada a base de dados *IG toy* na representação tf_{norm} apresentam resultados médios melhores.

Analisando o parâmetro de número de grupos de termos nos resultados para algoritmos de FM para coagrupamento, é possível perceber que os melhores resultados médios são obtidos quando $l \geq k$. Ainda, é possível observar um padrão analisando a configuração do parâmetro l para os melhores resultados em geral (tabela 10): os valores de l para os resultados obtidos com os algoritmos propostos *OvNMTF* e *BinOvNMTF* são, em todos os casos, menores que os valores obtidos para os algoritmos da literatura *ONMTF* e *FNMTF*. Essa questão impacta diretamente na qualidade dos cogrupos de documentos, pois os cogrupos de termos são usados para formá-los, sendo portanto um ponto interessante para desenvolvimento de estudos específicos.

Os resultados para a base de dados *IG toy* também mostram a instabilidade e capacidade de agrupamento média inferior dos algoritmos que trabalham sob restrições binárias (ou ortogonais) nas matrizes indicadoras de grupos de documentos e termos. A rápida convergência desses algoritmos, proporcionada pelo processo de busca restrito à soluções no espaço binário (ou ortogonal), pode impossibilitar o encontro de um mínimo (local) que está em um espaço de soluções contínuo. Esse fato pode ser o causador dos problemas citados.

Base de dados *IG*

Para a base de dados *IG*, os resultados apontam uma superioridade dos algoritmos *OvNMTF* e *BinOVNMTF*. As tabelas 11 e 12 mostram os resultados médios obtidos para essa base considerando cada algoritmo e cada representação para os textos, sob a avaliação dos índices de Rand e informação mútua normalizada, respectivamente. Na tabela 13 são mostrados os melhores resultados. O algoritmo *BinOvNMTF* obteve o melhor resultado, considerando as duas métricas dessas tabelas.

Tabela 11 – Índice de Rand médio por experimento com o conjunto de dados *IG*: com $k = 13$, e variações de l e de representação para os textos

Algoritmo	*tf*	tf_{norm}	$tfidf_{norm}$	*tfidf*
k-means	$0,3137$	$0,3049$	$0,2750$	$0,2784$
fuzzy k-means	$0,1694$	$0,1619$	$0,2429$	$0,2662$
ONMTF	$0,1437 : l = 16$	$0,1802 : l = 19$	$0,1184 : l = 7$	$0,1279 : l = 7$
FNMTF	$0,2327 : l = 19$	$0,2399 : l = 19$	$0,2165 : l = 19$	$0,2124 : l = 13$
OvNMTF	$0,3384 : l = 10$	$0,3455 : l = 16$	$\mathbf{0,3554} : l = 16$	$\mathbf{0,3534} : l = 7$
BinOvNMTF	$\mathbf{0,3784} : l = 16$	$\mathbf{0,3591} : l = 7$	$0,2807 : l = 19$	$0,2868 : l = 19$

Fonte: Lucas Fernandes Brunialti, 2016

Tabela 12 – Informação Mútua Normalizada média por representação do conjunto de dados *IG* com $k = 13$, e variações de l e de representação para os textos

Algoritmo	*tf*	tf_{norm}	$tfidf_{norm}$	*tfidf*
k-means	$0,5235$	$0,5240$	$0,5361$	$0,5350$
fuzzy k-means	$0,2548$	$0,2518$	$0,3769$	$0,3929$
ONMTF	$0,4186 : l = 19$	$0,4312 : l = 19$	$0,4338 : l = 19$	$0,4416 : l = 19$
FNMTF	$0,4412 : l = 19$	$0,4492 : l = 19$	$0,4518 : l = 19$	$0,4593 : l = 19$
OvNMTF	$0,4930 : l = 7$	$0,5001 : l = 16$	$\mathbf{0,5493} : l = 16$	$\mathbf{0,5451} : l = 7$
BinOvNMTF	$\mathbf{0,5563} : l = 16$	$\mathbf{0,5424} : l = 7$	$0,5423 : l = 19$	$0,5327 : l = 19$

Fonte: Lucas Fernandes Brunialti, 2016

Tabela 13 – Melhores resultados obtidos para o conjunto de dados *IG* com $k = 13$, destacando os melhores resultados por algoritmo

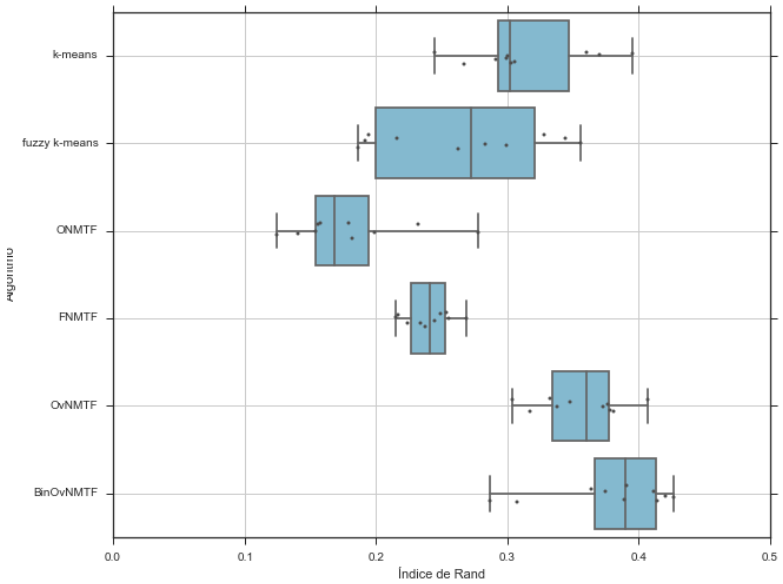
Algoritmo	**Índice de Rand**	**Informação Mútua Normalizada**
k-means	$0,3955 : tf$	$0,5826* : tfidf_{norm}$
fuzzy k-means	$0,3557 : tfidf$	$0,4365 : tfidf_{norm}$
ONMTF	$0,2884 : l = 10,\ tf_{norm}$	$0,4938 : l = 16,\ tfidf$
FNMTF	$0,2813 : l = 16,\ tfidf_{norm}$	$0,5047 : l = 19,\ tfidf$
OvNMTF	$0,4251* : l = 10,\ tfidf_{norm}$	$0,5778 : l = 16,\ tfidf_{norm}$
BinOvNMTF	$\mathbf{0,5743} : l = 10,\ tf$	$\mathbf{0,6064} : l = 7,\ tfidf_{norm}$

Fonte: Lucas Fernandes Brunialti, 2016

As figuras 25 e 26 mostram os resultados utilizados para obtenção das melhores médias nas tabelas 11 e 13, para as métricas índice de Rand e informação mútua normalizada, repectivamente. Ainda, nas figuras 27 e 28 são mostrados todos os resultados obtidos para todas as configurações considerando as duas métricas de qualidade de agrupamento.

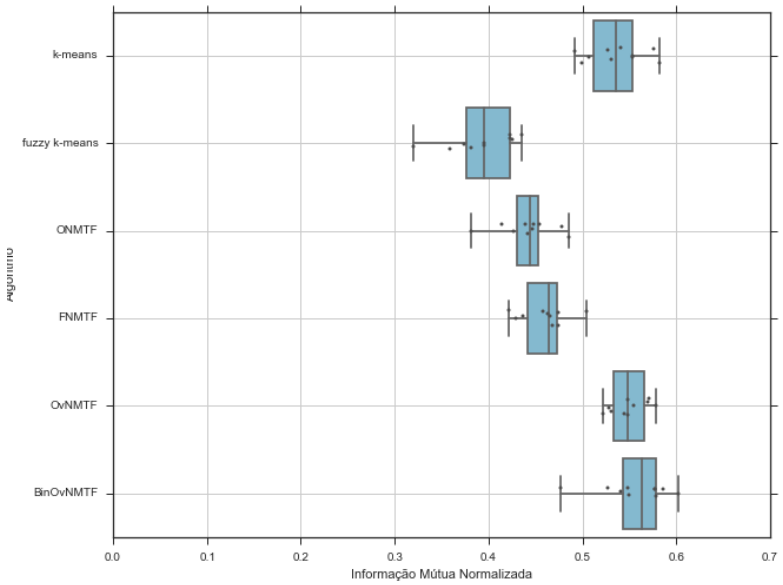
Diferentemente dos resultados apresentados para a base de dados *IG toy*, apesar de ter o mesmo comportamento instável, os resultados obtidos com os experimentos com a base de dados *IG* apontam que o algoritmo *BinOvNMTF* obteve os melhores resultados, uma possível hipótese é que os algoritmos com as restrições binárias, quando o número de grupos de linhas buscado é grande, as restrições binárias têm efeito positivo na capacidade de agrupamento. Isso é possível notar quando os resultados com os algoritmos *ONMTF* e *FNMTF* são comparados, sendo o algoritmo *FNMTF* melhor na maioria dos resultados. Esse comportamento também pode ser observado nas figuras 27 e 28, que mostram as distribuições considerando todos os experimentos com as métricas índice de Rand e informação mútua normalizada, repectivamente.

Figura 25 – Distribuições dos valores do índice de Rand para os melhores resultados médios para cada algoritmo na base de dados *IG*



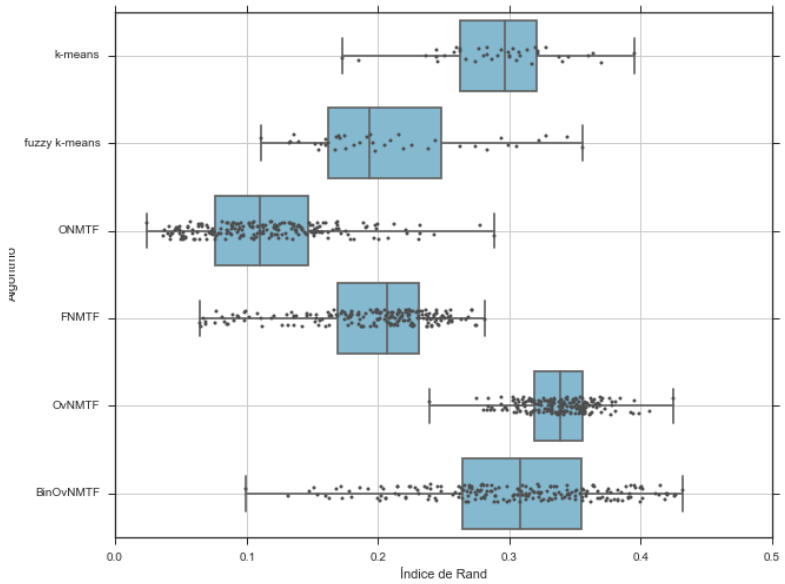
Fonte: Lucas Fernandes Brunialti, 2016

Figura 26 – Distribuições dos valores de informação mútua normalizada para os melhores resultados médios para cada algoritmo na base de dados *IG*



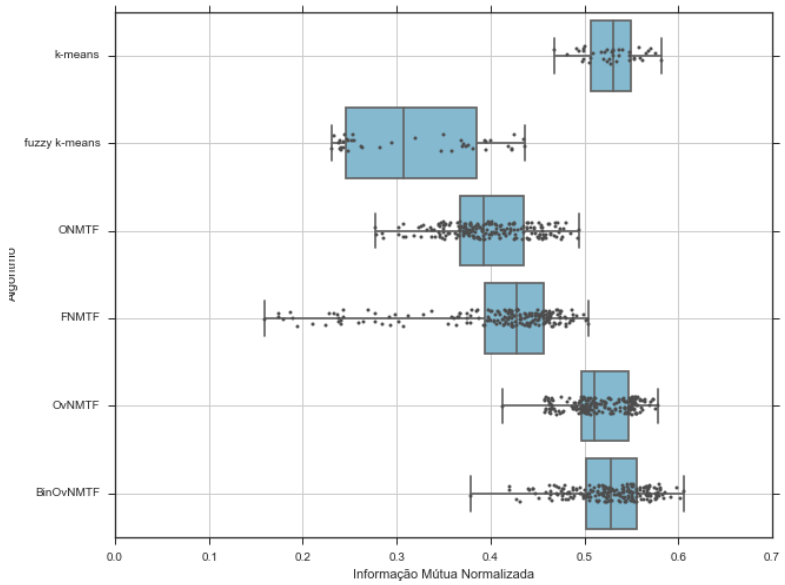
Fonte: Lucas Fernandes Brunialti, 2016

Figura 27 – Distribuições dos valores do índice de Rand para todas as execuções para cada algoritmo na base de dados *IG*



Fonte: Lucas Fernandes Brunialti, 2016

Figura 28 – Distribuições dos valores de informação mútua normalizada para todas as execuções para cada algoritmo na base de dados *IG*



Fonte: Lucas Fernandes Brunialti, 2016

Ainda, faz-se interessante observar que, comparando os resultados do algoritmo *BinOvNMTF* com aqueles obtidos para o algoritmo do qual ele se origina, o *FNMTF*, há melhoras significativas (conforme as figuras 27 e 28).

Base de dados *NIPS*

Para a base de dados *NIPS*, os resultados apontam uma superioridade do algoritmo *BinOvNMTF*. As tabelas 14 e 15 mostram os resultados médios obtidos para essa base considerando cada algoritmo e cada representação para os textos, sob a avaliação dos índices de Rand e informação mútua normalizada, respectivamente. Na tabela 16 são mostrados os melhores resultados.

Tabela 14 – Índice de Rand médio por experimento com o conjunto de dados *NIPS*: com $k = 9$, e variações de l e de representação para os textos

Algoritmo	*tf*	tf_{norm}	$tfidf_{norm}$	*tfidf*
k-means	$0,1573$	$0,1527$	$0,1519$	$0,1368$
fuzzy k-means	$0,1223$	$0,1240$	$0,1736$	$0,1882$
ONMTF	$0,1579 : l = 6$	$0,1352 : l = 15$	$0,1318 : l = 9$	$0,1442 : l = 18$
FNMTF	$0,1293 : l = 18$	$0,1325 : l = 18$	$0,2128 : l = 18$	$0,2199 : l = 18$
OvNMTF	$0,1672 : l = 6$	$0,1641 : l = 9$	$0,1742 : l = 12$	$0,1711 : l = 9$
BinOvNMTF	$\mathbf{0,2247} : l = 9$	$\mathbf{0,2118} : l = 15$	$\mathbf{0,2811} : l = 6$	$\mathbf{0,2813} : l = 15$

Fonte: Lucas Fernandes Brunialti, 2016

Tabela 15 – Informação Mútua Normalizada média por representação do conjunto de dados *NIPS* com $k = 9$, e variações de l e de representação para os textos

Algoritmo	*tf*	tf_{norm}	$tfidf_{norm}$	*tfidf*
k-means	$0,3226$	$0,3106$	$0,3506$	$0,3476$
fuzzy k-means	$0,1876$	$0,1929$	$0,2575$	$0,2496$
ONMTF	$0,2930 : l = 15$	$0,2832 : l = 18$	$0,3361 : l = 18$	$0,3441 : l = 18$
FNMTF	$0,2272 : l = 18$	$0,2312 : l = 18$	$0,3109 : l = 18$	$0,3017 : l = 18$
OvNMTF	$0,3090 : l = 6$	$0,3092 : l = 9$	$0,3541 : l = 12$	$0,3493 : l = 15$
BinOvNMTF	$\mathbf{0,3255} : l = 15$	$\mathbf{0,3139} : l = 9$	$\mathbf{0,4013} : l = 12$	$\mathbf{0,4009} : l = 15$

Fonte: Lucas Fernandes Brunialti, 2016

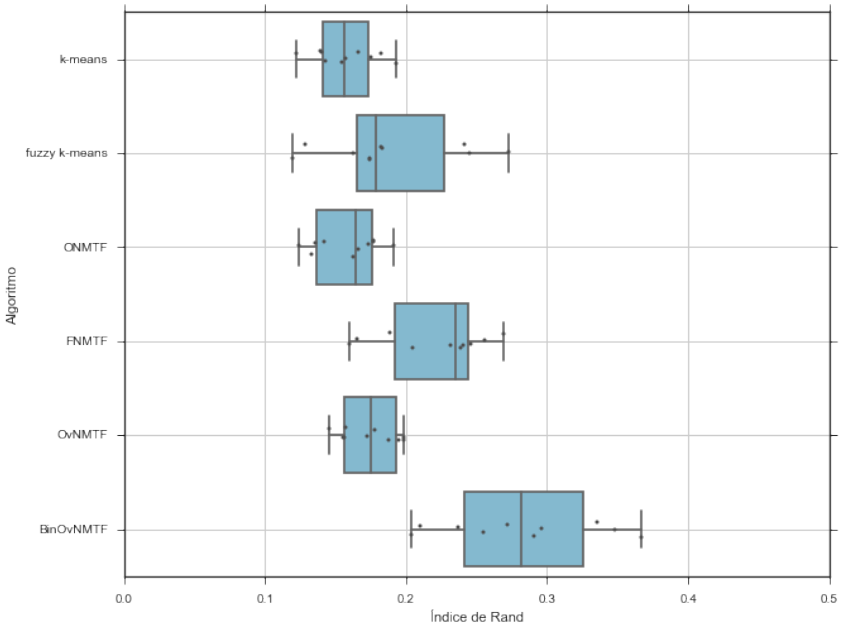
Tabela 16 – Melhores resultados obtidos para o conjunto de dados *NIPS* com $k = 9$, destacando os melhores resultados por algoritmo

Algoritmo	**Índice de Rand**	**Informação Mútua Normalizada**
k-means	$0,2006 : tfidf_{norm}$	$0,3952 : tfidf_{norm}$
fuzzy k-means	$0,2728 : tfidf$	$0,3115 : tfidf$
ONMTF	$0,2704 : l = 18,\ tfidf$	$0,3997* : l = 18,\ tfidf$
FNMTF	$0,3009* : l = 15,\ tfidf$	$0,3744 : l = 18,\ tfidf_{norm}$
OvNMTF	$0,1992 : l = 9,\ tfidf$	$0,3870 : l = 9,\ tfidf$
BinOvNMTF	$\mathbf{0,3670} : l = 15,\ tfidf$	$\mathbf{0,4589} : l = 9,\ tfidf_{norm}$

Fonte: Lucas Fernandes Brunialti, 2016

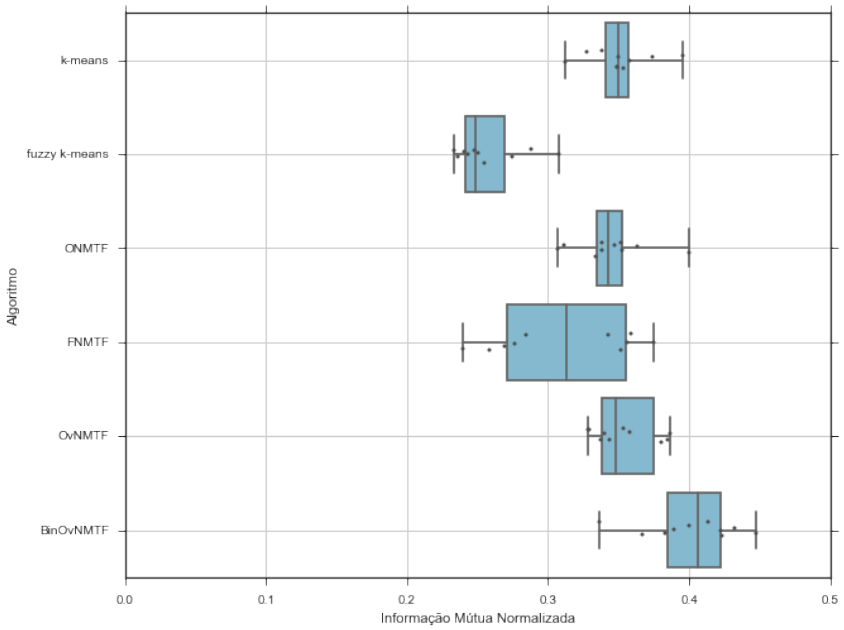
Observando todos os resultados para a base de dados *NIPS* é visível que o algoritmo *BinOvNMTF* foi o que mais se destacou, considerando ambas as medidas. Tanto para os melhores resultados (máximos - tabela 16) quanto para os resultados médios (tabelas 14 e 15). No entanto, esse foi também o algoritmo que apresentou a maior instabilidade nos resultados (observada nas distribuições das figuras 29, 30, 31 e 32), devido possivelmente às razões já comentadas neste texto.

Figura 29 – Distribuições dos valores do índice de Rand para os melhores resultados médios para cada algoritmo na base de dados *NIPS*



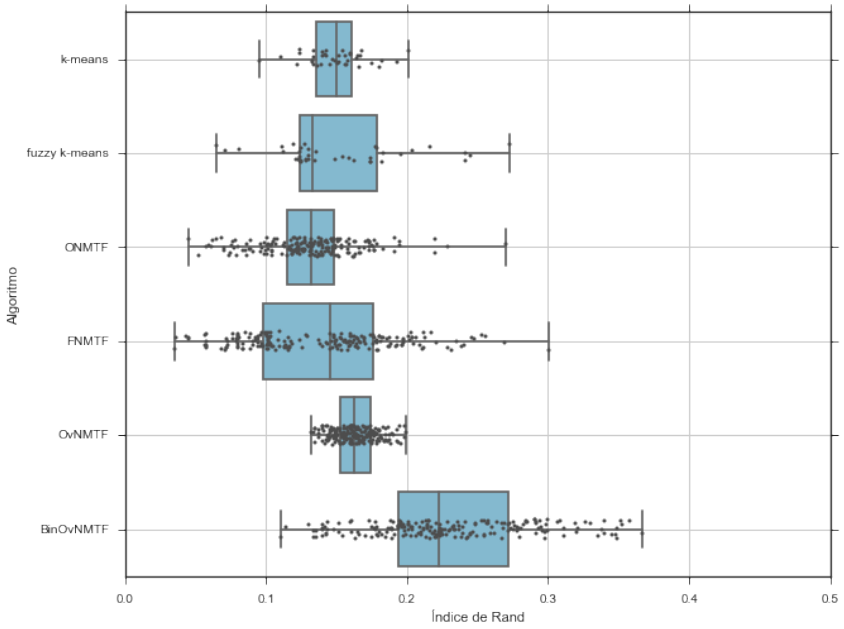
Fonte: Lucas Fernandes Brunialti, 2016

Figura 30 – Distribuições dos valores de informação mútua normalizada para os melhores resultados médios para cada algoritmo na base de dados *NIPS*



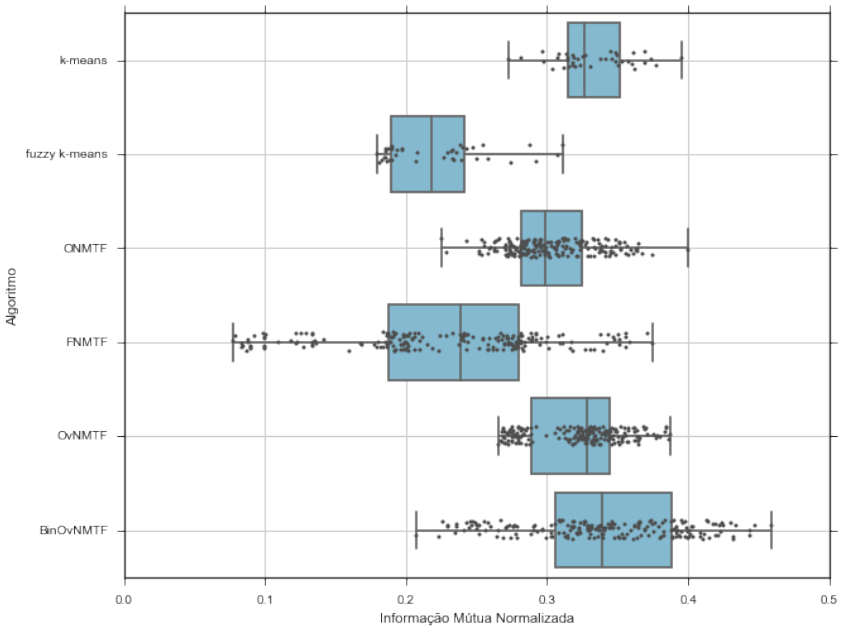
Fonte: Lucas Fernandes Brunialti, 2016

Figura 31 – Distribuições dos valores do índice de Rand para todas as execuções para cada algoritmo na base de dados *NIPS*



Fonte: Lucas Fernandes Brunialti, 2016

Figura 32 – Distribuições dos valores de informação mútua normalizada para todas as execuções para cada algoritmo na base de dados *NIPS*



Fonte: Lucas Fernandes Brunialti, 2016

Observando os melhores resultados e os resultados médios, o parâmetro l se apresentou ligeiramente menor para os algoritmos *OvNMTF* e *BinOvNMTF*, comparando com os algoritmos *ONMTF* e *FNMTF*, como esperado. Ainda, analisando os resultados por representação para os textos, é possível perceber uma tendência de destaque para a representação *tfidf*, tanto não normalizada quanto a normalizada.

Contrariamente ao que ocorreu na base de dados *IG toy*, e semelhante ao que ocorreu na base de dados *IG*, o algoritmo *OvNMTF* não apresentou resultados tão bons. Os algoritmos *ONMTF* e *FNMTF* se apresentaram competitivos ao *OvNMTF*, apresentando melhores resultados máximos. Essa inferioridade para o algoritmo *OvNMTF* levanta a hipótese que restrições binárias e de ortogonalidade podem ser mais interessantes em situações de menor esparsidade, como é o caso da base de dados *NIPS* (veja tabela 3).

E, por fim, faz-se interessante observar que, comparando os resultados do algoritmo *BinOvNMTF* com aqueles obtidos para o algoritmo do qual ele se origina, o *FNMTF*, há melhoras significativas considerando todas as bases de ados (conforme as figuras 23, 27 e 31). Essa constatação indica que o mecanismo de independência para formação do cogrupos de palavras, criado nas abordagens propostas neste trabalho, é capaz de melhorar, com eficiência, a resolução da tarefa de análise de agrupamentos em dados textuais.

5.2.4 Análises qualitativas

As análises apresentadas nesta seção mostram a aplicabilidade dos algoritmos discutidos neste texto no domínio de mineração de texto. Essa análise qualitativa é importante para mostrar que os algoritmos são capazes de encontrar tópicos (grupos de palavras) e, possivelmente, explicar os grupos de notícias (documentos) descobertos, ou até mesmo, realizar rotulação desses grupos.

Especificamente, trata-se aqui das informações que podem ser extraídas dos modelos gerados pelos algoritmos de coagrupamento baseados em fatoração de matrizes: *ONMTF* e *OvNMTF*. Os algoritmos *FNMTF* e *BinOvNMTF* não foram considerados nesta análise pois a restrições binárias inviabilizam a análise semi-automática de palavras nos cogrupos de palavras, como proposta aqui. O problema é que a estratégia proposta depende da existência de um fator que relaciona palavras, documentos e grupos, em diferentes magnitudes ou com diferentes relevâncias. Essa questão se apresenta como uma desvantagem dos algoritmos com restrições binárias.

A fim de possibilitar a análise, foi estabelecido o propósito de estudar a relação existente entre cogrupos de palavras e cogrupos de notícias. Para isso, foi usado o conjunto de dados *IG toy*, que, de fato, foi construído justamente para viabiliar estudos mais detalhados, dentro de um ambiente de análise com algum grau de controle.

O portal IG[13] é um dos maiores portais de notícias brasileiro (SITES..., 2016), composto por sites importantes como o noticiário Último Segundo, o iG Gente, o iG Esportes, a TV iG, o iG Economia, o Delas etc. Cada um desses sites é caracterizado como um canal que contém notícias de um determinado assunto macro. O conjunto de dados *IG toy* é composto por 100 notícias de cada um dos três canais: **esportes**, que contém notícias, principalmente, dos esportes mais populares no Brasil, como o futebol e o UFC; **jovem**, que contém notícias mais interessantes para o público jovem em geral, como notícias sobre esportes radicais, filmes e músicas voltados para o público jovem; **arena**, que compõem notícias de games, novidades de todos os tipos de games, consoles e coberturas de eventos de games.

Esses canais foram escolhidos para formar o conjunto de dados *IG toy* por serem de assuntos similares. Notícias do canal jovem podem apresentar semelhanças com notícias do canal esportes: notícias sobre surfe, por exemplo. Notícias do canal arena podem ser

[13] http://www.ig.com.br/

similares a notícias do canal de esportes, já que existem *games* que simulam esportes, e para os quais existem campeonatos oficiais, como no caso dos esportes. Ainda, o canal arena possui notícias de interesse dos jovens e, portanto, notícias desse canal poderiam, naturalmente, estar classificadas no canal jovem. Essa escolha de canais torna possível verificar como os algoritmos tratam essa sobreposição entre palavras (ou tópicos) nas notícias.

As subseções seguintes mostram como os algoritmos *ONMTF* e *OvNMTF* são úteis para análise de tópicos e palavras no conjunto de dados *IG toy*.

5.2.4.1 Análise de dados utilizando *ONMTF*

Para desenvolver análises a partir dos resultados da aplicação do algoritmo *ONMTF* foi escolhido o modelo que obteve o melhor resultado considerando o *índice de Rand* na base de dados *IG toy* (conforme apresentado na seção 5.2.3.2): o modelo com $k = 3$, $l = 5$ e representação para dos dados textuais $tfidf_{norm}$.

O algoritmo *ONMTF* é capaz de encontrar cogrupos de notícias e cogrupos de palavras, além disso, cada cogrupo de notícias é relacionado com os cogrupos de palavras por um fator, assim como cada cogrupo de palavras está relacionado com os cogrupos de notícias pelo mesmo fator.

Os fatores que relacionam os cogrupos de notícias com os cogrupos de palavras podem ser observados na matriz S. A matriz S que foi usada nesta análise está apresentada na tabela 17. Note que os valores estão normalizados para que a soma dos elementos de cada linha seja igual à 1. A normalização foi feita para tornar claro que cada cogrupo de palavras foi usado pelo algoritmo para caracterizar um grupo de notícias. Os valores mostram que o cogrupo de notícias “arena” está relacionado com um maior fator ao cogrupo de palavras #4, o cogrupo de notícias “esporte” está relacionado com fatores semelhantes aos cogrupos de palavras #2 e #5, e o cogrupo de notícias “jovem” está relacionado com fatores semelhantes aos cogrupos de palavras #1 e #3.

Tabela 17 – Matriz S normalizada para o algoritmo *ONMTF* com $k = 3$ e $l = 5$ executado sobre a base de dados *IG toy*

	CP #1	**CP #2**	**CP #3**	**CP #4**	**CP #5**
CN "esportes"	$0,0$	$\mathbf{0,5}$	$0,1$	$0,0$	$\mathbf{0,4}$
CN "arena"	$0,0$	$0,05$	$0,05$	$\mathbf{0,9}$	$0,0$
CN "jovem"	$\mathbf{0,4}$	$0,1$	$\mathbf{0,5}$	$0,0$	$0,0$

Fonte: Lucas Fernandes Brunialti, 2016

Uma ressalva precisa ser feita no contexto desse exemplo. Ao realizar a normalização, a solução original apresentada pelo algoritmo é alterada, de forma que, por exemplo, a reconstrução da matriz original já não é mais possível. No entanto, para fins de interpretação da solução de coagrupamento, a normalização é interessante para melhorar a legibilidade dos relacionamentos entre os cogrupos.

Uma rotulação para grupos de notícias foi então possível por meio da análise da relações expressas pela da matriz S, e da análise das palavras em cada cogrupo de palavras. Esse cogrupos de palavras são obtidos a partir do particionamento (descrito no capítulo 3) que atribui cada palavra ao cogrupo que possui o maior fator para cada linha na matriz V. O resultado das palavras mais relevantes para cada cogrupo de palavras gerado pelo algoritmo *ONMTF*, é mostrado na tabela 18. A ordenação foi feita usando os fatores (na matriz V) que relacionam cada palavra para cada cogrupo de palavras. Os rótulos atribuídos a cada cogrupo de palavras foram determinados com base na inspeção visual das listas de palavras.

Tabela 18 – Top-20 palavras para cada cogrupo de palavras, após a realização do particionamento baseado na matriz V

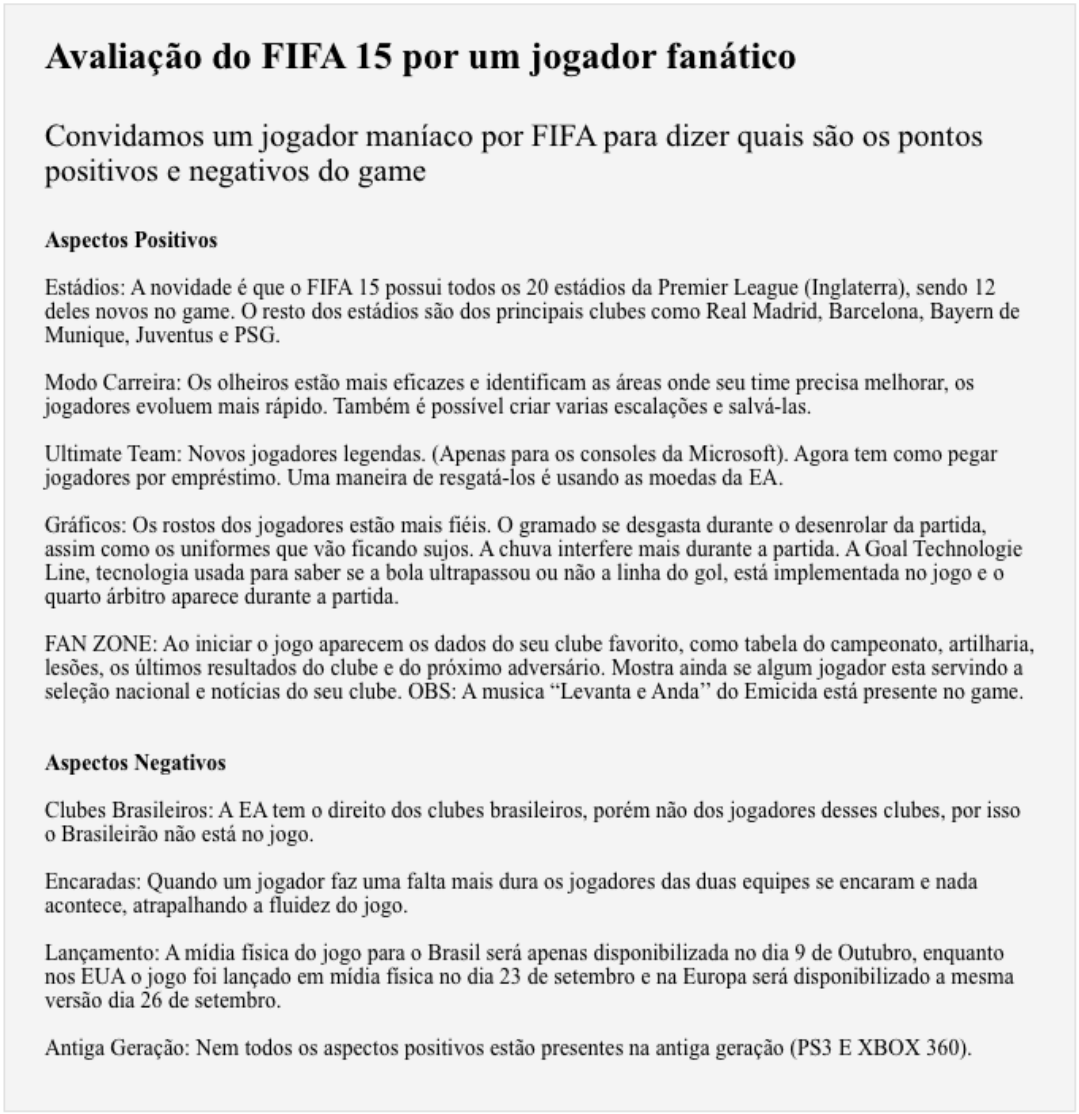
CP #1 *"esportes radicais"*	**CP #2** *"futebol"*	**CP #3** *"esportes em geral"*	**CP #4** *"games"*	**CP #5** *"futebol"*
skate	real	anos	jogos	gol
surfe	breno	mundial	xbox	madrid
bob	time	brasil	playstation	gols
burnquist	barcelona	etapa	wii	messi
circuito	partida	brasileiro	jogo	euro
games	equipe	jovem	of	guardiola
mineirinho	minutos	rio	console	ronaldo
slater	jogador	dias	sony	itália
rampa	campeonato	música	legends	cristiano
medina	liga	pedro	nintendo	bola
manobras	futebol	atleta	game	bayern
mega	casa	americano	league	pontos
megarampa	temporada	gente	one	espanhol
kelly	grupo	esporte	novo	zagueiro
prova	feira	campeão	ps	atacante
bmx	dois	competição	microsoft	defesa
pista	vitória	mundo	the	técnico
barros	tempo	ano	usmonetáriointerno	placar
skatistas	área	janeiro	controle	atlético
rony	copa	ufc	versão	polônia

Fonte: Lucas Fernandes Brunialti, 2016

O conteúdo apresentado na tabela 18 associado aos fatores da matriz S (tabela 17) permitem inferir que os cogrupos de palavras #2 e #5 (rotulados como "futebol") foram responsáveis por caracterizar o cogrupo de notícias que representa o canal "esportes", o cogrupo de palavras #4 (rotulado como "games") foi responsável por caracterizar o cogrupo de notícias que representa o canal "arena", e por fim, os cogrupos de palavras #1 e #3 (rotulados como "esportes radicais" e "esportes em geral", respectivamente) foram responsáveis por caracterizar as notícias do cogrupo que representa o canal "jovem".

Com o objetivo de exemplificar as informações que um modelo gerado pelo algoritmo *ONMTF* provê, a notícia "Avaliação do FIFA 15 por um jogador fanático", mostrada na figura 33, foi usada como exemplo.

Figura 33 – Exemplo de uma notícia do canal “arena”



Avaliação do FIFA 15 por um jogador fanático

Convidamos um jogador maníaco por FIFA para dizer quais são os pontos positivos e negativos do game

Aspectos Positivos

Estádios: A novidade é que o FIFA 15 possui todos os 20 estádios da Premier League (Inglaterra), sendo 12 deles novos no game. O resto dos estádios são dos principais clubes como Real Madrid, Barcelona, Bayern de Munique, Juventus e PSG.

Modo Carreira: Os olheiros estão mais eficazes e identificam as áreas onde seu time precisa melhorar, os jogadores evoluem mais rápido. Também é possível criar varias escalações e salvá-las.

Ultimate Team: Novos jogadores legendas. (Apenas para os consoles da Microsoft). Agora tem como pegar jogadores por empréstimo. Uma maneira de resgatá-los é usando as moedas da EA.

Gráficos: Os rostos dos jogadores estão mais fiéis. O gramado se desgasta durante o desenrolar da partida, assim como os uniformes que vão ficando sujos. A chuva interfere mais durante a partida. A Goal Technologie Line, tecnologia usada para saber se a bola ultrapassou ou não a linha do gol, está implementada no jogo e o quarto árbitro aparece durante a partida.

FAN ZONE: Ao iniciar o jogo aparecem os dados do seu clube favorito, como tabela do campeonato, artilharia, lesões, os últimos resultados do clube e do próximo adversário. Mostra ainda se algum jogador esta servindo a seleção nacional e notícias do seu clube. OBS: A musica “Levanta e Anda” do Emicida está presente no game.

Aspectos Negativos

Clubes Brasileiros: A EA tem o direito dos clubes brasileiros, porém não dos jogadores desses clubes, por isso o Brasileirão não está no jogo.

Encaradas: Quando um jogador faz uma falta mais dura os jogadores das duas equipes se encaram e nada acontece, atrapalhando a fluidez do jogo.

Lançamento: A mídia física do jogo para o Brasil será apenas disponibilizada no dia 9 de Outubro, enquanto nos EUA o jogo foi lançado em mídia física no dia 23 de setembro e na Europa será disponibilizado a mesma versão dia 26 de setembro.

Antiga Geração: Nem todos os aspectos positivos estão presentes na antiga geração (PS3 E XBOX 360).

Fonte: Lucas Fernandes Brunialti, 2016

Segundo o resultado de coagrupamento obtido para essa notícia, seguindo os fatores presentes na linha correspondente a ela na matriz U, ela tem relações mais altas aos cogrupos de notícias que representa os canais “arena” e “esporte”, e nenhuma relação ao cogrupo de notícias que representa o canal “jovem”; sendo que o fator que a relaciona ao primeiro cogrupo (canal “arena”) é um pouco mais alto do que aquele que a relaciona ao segundo cogrupo (canal “esporte”).

Uma forma de visualização alternativa é aqui proposta para permitir a interpretação da relação das palavras com os grupos de palavras. Essa visualização se apresenta como uma *nuvem de palavras* na qual o tamanho das palavras é definido pelos fatores que as

associam aos cogrupos (matriz V). A figura 34 mostra essa visualização a partir de nuvens de 100 palavras.

Figura 34 – Visualização em nuvem de palavras das top-100 palavras, para cada cogrupo gerados pelo algoritmo *ONMTF*

(a) CP #1 *"esportes radicais"*

(b) CP #2 *"futebol"*

(c) CP #3 *"esportes em geral"*

(d) CP #4 *"games"*

(e) CP #5 *"futebol"*

Fonte: Lucas Fernandes Brunialti, 2016

Note que uma palavra pode aparecer em diferentes cogrupos de palavras. Como exemplo, considere a palavra *jogo* presente nos cogrupos #4 e #5 (a maior palavra do cogrupo #4 e no centro do 'o' da palavra *gol* para o cogrupo #5). Esse fato significa que existem notícias, tanto no cogrupo de notícias do canal "arena" quanto no cogrupo de notícias do canal "esportes", nas quais a palavra *jogo* ocorre, sendo que a maior ocorrência é no primeiro cogrupo. É possível observar esse fato porque nessa visualização foram usados os 100 maiores fatores de cada uma das l colunas na matriz V, gerada pela fatoração *ONMTF*. Diante desse contexto, pode-se interpretar que a palavra *jogo*, por exemplo, possui diferentes significados em cada cogrupo de palavras.

Associando essa interpretação à interdependência que o algoritmo *ONMTF* impõe na formação dos cogrupos (de notícias e de palavras), a ocorrência da palavra "jogo" no contexto do cogrupo "futebol" pode afetar as relações estabelecidas na matriz S de maneira a causar um efeito direto no estabelecimento do cogrupo de palavras "games", e vice-versa.

5.2.4.2 Análise de dados utilizando *OvNMTF*

De forma semelhante ao que foi realizado na análise com o modelo gerado pelo algoritmo *ONMTF*, a escolha do modelo usado para a análise com o algoritmo *OvNMTF* foi baseada no melhor resultado considerando o índice de Rand para a base de dados *IG toy* (conforme apresentado na seção 5.2.3.2): o modelo com $k = 3$, $l = 2$ e representação tf_{norm}.

O algoritmo *OvNMTF*, proposto neste trabalho, é capaz de encontrar cogrupos de notícias e cogrupos de palavras, com a particularidade de que há um conjunto de l cogrupos de palavras associados à cada um dos k cogrupos de notícias. Desta forma, os cogrupos de palavras de diferentes cogrupos de notícias são formados de maneira independente.

A matriz S usada nessa análise é apresentada na tabela 19. A mesma normalização da análise com o algoritmo *ONMTF* foi aplicada a essa matriz e as mesmas ressalvas quanto a normalização também servem para este caso. No presente caso, é preciso lembrar que cada linha da matriz S está associada aos cogrupos de palavras correspondentes à essa linha, isto é, cada fator da linha 1 da matriz S tem relação com todos os cogrupos de palavras da linha 1 (extraídos da matriz resultante da fatoração $V_{(1)}$). Assim, existem 6 cogrupos de palavras – 2 cogrupos de palavras para cada cogrupo de notícia:

- os cogrupos #1 e #2 referentes ao cogrupo de notícias relacionado ao canal "arena";
- os cogrupos #3 e #4 referentes ao cogrupo de notícias relacionado ao canal "jovem";
- os cogrupos #5 e #6 referentes ao cogrupos de notícias relacionado ao canal "esportes".

Tabela 19 – Matriz S normalizada para o algoritmo *OvNMTF* com $k = 3$ e $l = 2$ executado sobre a base de dados *IG toy*

	CP #1, #3, #5	**CP #2, #4, #6**
CN "arena"	**0.38**	**0.62**
CN "jovem"	**0.46**	**0.54**
CN "esportes"	**0.94**	0.06

Fonte: Lucas Fernandes Brunialti, 2016

Interpretando os fatores da matriz S da tabela 19, verifica-se que o cogrupo de notícias referente ao canal "arena" esta associado com um fator maior ao cogrupo de palavras #2 e com um fator menor ao cogrupo de palavras #1, o cogrupo de notícias referente ao canal "jovem" esta associado com fatores similares aos cogrupos de palavras #3 e #4, e o cogrupo de notícias referente ao canal "esportes" é caracterizado, principalmente, pelo cogrupo de palavras #5.

A rotulação dos cogrupos de notícias foi feita por meio da análise de palavras presentes em cada cogrupo de palavras, e relação especificamente associado aos cogrupos de notícias. As palavras mais relevantes presentes em cada cogrupo de palavras são mostradas na tabela 20.

A estratégia para interpretação usada foi a mesma da análise com o algoritmo *ONMTF*: foi realizado o particionamento das palavras (como descrito no capítulo 4) considerando os fatores presentes nas matrizes $V_{(p)}, \forall p$ para realizar a ordenação; as 20 palavras com maiores fatores para cada cogrupo de palavras são aquelas listadas na tabela 20.

Note que no caso do *OvNMTF* a mesma palavra também pode aparecer em diferentes cogrupos de palavras, como a palavra *games* que aparece nos cogrupos #2 e #3, ou *jogo* que aparece nos cogrupos #2 e #6. Porém, neste caso, não há dependência entre os l-grupos de grupos: #1 #2, #3 #4, #5 e #6, o que minimiza os efeitos potencialmente nocivos à interpretação dos significados atribuídos às palavras e aos grupos de notícias que ocorria no algoritmo *ONMTF*. Porém, ainda há interdependência entre os cogrupos de palavras relacionados à um mesmo cogrupo de notícias.

O exemplo referente à palavra *jogo* nos resultados apresentados pelo algoritmo *OvNMTF* pode suportar uma interpretação: no contexto do cogrupo de palavras #2, rotulado como "games" a palavra *jogo* tem um significado particular; esse significado é independente daquele que pode ser atribuído a palavra *jogo* no contexto particular do

Tabela 20 – Top-20 palavras para cada cogrupo de palavras, após a realização do particionamento

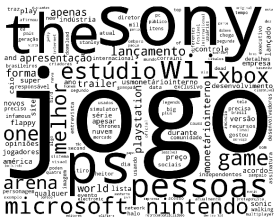
CP #1 *"games"*	**CP #2** *"games"*	**CP #3** *"esportes em geral"*	**CP #4** *"esportes radicais + música"*	**CP #5** *"futebol"*	**CP #6** *"futebol"*
jogos	jogo	games	anos	time	breno
sony	of	jovem	skate	real	casa
ps	playstation	brasileiro	mundial	feira	gol
the	game	paulo	brasil	final	bayern
pessoas	novo	dia	surfe	gols	minutos
wii	console	mundo	música	madrid	clube
microsoft	xbox	ano	rio	jogador	partida
nintendo	games	esporte	conta	tempo	técnico
estúdio	league	vai	dias	pontos	título
one	legends	janeiro	primeira	grupo	livre
arena	brasil	bem	final	liga	jogo
melhor	além	burnquist	atleta	fez	meia
apenas	nova	além	pessoas	brasileiro	volta
lançamento	jogadores	gente	casa	jogadores	segunda
versão	dia	bob	paulista	campo	técnica
opiniões	lançado	série	ficou	rodada	casillas
caio	usmonetáriointerno	etapa	fim	três	espanha
site	personagens	história	monetáriointerno	cristiano	equipe
and	feira	melhor	melhores	copa	argentino
forma	dois	circuito	amigos	deixe	semana

Fonte: Lucas Fernandes Brunialti, 2016

cogrupo de palavras #6, rotulado como “futebol”, visto que eles são formados de maneira independentes dentro da estratégia algorítmica.

Ainda, foram criadas visualizações de cada cogrupo de palavras no formato de nuvem de palavras. Essas visualizações foram feitas considerando as 100 palavras com maiores fatores para cada coluna de cada matriz $V_{(p)}$, sem a realização do particionamento (Figuras 35, 36, 37).

Figura 35 – Visualização em nuvem de palavras para cada cogrupo de palavras do cogrupo de notícias “arena”, gerados pelo algoritmo *OvNMTF*.



(a) #1 “games”

(b) #2 “games”

Fonte: Lucas Fernandes Brunialti, 2016

Figura 36 – Visualização em nuvem de palavras para cada cogrupo de palavras do cogrupo de notícias “jovem”, gerados pelo algoritmo *OvNMTF*.

(a) #3 “esportes em geral”

(b) #4 “esportes radicais + música”

Fonte: Lucas Fernandes Brunialti, 2016

Figura 37 – Visualização em nuvem de palavras para cada cogrupo de palavras do cogrupo de notícias “esportes”, gerados pelo algoritmo *OvNMTF*.

(a) #5 “futebol” (b) #6 “futebol”

Fonte: Lucas Fernandes Brunialti, 2016

5.3 Considerações finais

Os experimentos deste trabalho trouxeram evidências de como avaliar os algoritmos aqui discutidos do ponto de vista de quantização. Os experimentos serviram para mostrar a capacidade das estratégias propostas de superar algumas das dificuldades apresentas nas fatorações da literatura (*ONMTF* e *FNMTF*), no que diz respeito à solução do problema de coagrupamento.

Do ponto de vista de reconstrução e quantização do espaço dos dados, foi visto que o único algoritmo capaz de fazer a reconstrução com perfeição e obter o menor erro de quantização, considerando a base de dados com estrutura de cogrupos com sobreposição de linhas, foi o *OvNMTF*. No entanto, o algoritmo *OvNMTF* foi capaz de preservar parcialmente as informações de sobreposição de linhas e colunas, isso pode indicar que este algoritmo é capaz de lidar com sobreposição de cogrupos, porém, não de forma tão natural quanto o *OvNMTF*. Já o algoritmo *BinOvNMTF* foi capaz de lidar apenas com cogrupos com sobreposição de colunas, que é o seu objetivo principal, isso devido às suas restrições binárias que diminuir o espaço de busca e fazer com que o algoritmo não encontre as mesmas soluções que o *OvNMTF*. Além disso, foi visto que os algoritmos tradicionais de agrupamento são capazes de preservar as informações de sobreposição de colunas, no entanto, não são capazes de prover interpretações semelhantes quando a tarefa de coagrupamento é levada em consideração.

Do ponto de vista de capacidade de agrupamento, as fatorações propostas foram capazes, na maioria dos experimentos, segundo as medidas de validação *RI* e *NMI*, foram capazes de obter os melhores resultados. Foi percebido que a restrição binária pode ter

maior influência na capacidade de agrupamento a depender da base de dados, por exemplo, para a base de dados *IG toy* o algoritmo *BinOvNMTF* obteve resultados piores que o *OvNMTF*, contrariamente, para a base de dados *NIPS*, o algoritmo *BinOvNMTF* obteve resultados melhores que o *OvNMTF*.

Do ponto de vista de geração de informação sobre os dados, além do algoritmo *OvNMTF* ser capaz de fornecer o mesmo tipo de informação que o algoritmo *ONMTF*, considerando a aplicação de mineração de texto, isto é, como as notícias se organizam em grupos, como as palavras se organizam em grupos e como esses grupos se relacionam, o algoritmo *OvNMTF* é capaz de fornecer a informação de cada grupo de notícias se organiza em termos de grupos de palavras. Ainda, o algoritmo *OvNMTF* tem maior capacidade de separação dos assuntos presentes em cada grupo de documentos, pois permite a independência entre grupos de palavras referentes à diferentes grupos de notícias.

No geral, foi possível perceber que o processo no qual existe a independência entre grupos de colunas, que foi proposto nos algoritmos *OvNMTF* e *BinOvNMTF* deste trabalho, de fato, possibilita maior capacidade de reconstrução, de agrupamento e de geração de informação sobre os dados.

6 Conclusão

Esta dissertação teve o objetivo principal de apresentar uma solução para o problema de resolução da tarefa de coagrupamento com sobreposição de colunas, apresentado no capítulo 1, na forma de uma aplicação de mineração de texto hipotética. A principal motivação para buscar uma solução para esse problema advém dos contextos de análises de dados em que se faz necessário encontrar a organização "natural" dos dados em grupos. A análise de subconjuntos dos atributos descritivos desses dados podem trazer benefícios para a análise de agrupamento, como interpretações mais detalhadas, maior preservação de detalhes na quantização dos grupos e, portanto, melhor capacidade de agrupamento.

Um contexto de aplicação no qual a resolução do problema de coagrupamento se faz interessante é na mineração de texto, já que se tem a necessidade de apresentar resultados que gerem informações passíveis de serem interpretadas. Como estratégias de análise de coagrupamento formam grupos de documentos a partir de grupos de termos (ou características), informações antes escondidas nos dados, podem acabar sendo reveladas por esse processo.

Diante desta motivação, uma busca exploratória na literatura da área revelou a utilidade de algoritmos de fatoração de matrizes não-negativas para aplicação na solução do problema de coagrupamento, especialmente as fatorações em três matrizes: $X \approx USV^T$. Algoritmos apresentados como estado da arte, descritos no capítulo 3 foram estudados e, a partir desse estudo, constatou-se que existiam algumas lacunas nas quais poder-se-ia trabalhar para que as estratégias neles implementadas pudessem tratar de forma mais adequada algumas características da tarefa de coagrupamento, a saber: a sobreposição de linhas ou de colunas na composição dos cogrupos.

Dessa constatação, a hipótese de propor a fatoração da matriz de dados em $X \approx USV_{(1)}^T \dots V_{(k)}^T$, apresentada em no capítulo 1, motivou a proposição de problemas de fatoração de matrizes que possibilitassem uma separação que considera independência e sobreposição entre os cogrupos de colunas formados pela fatoração. A formulação de problemas com essa características e de algoritmos para sua resolução foram apresentados, acompanhados de suas derivações formais, no capítulo 4. Experimentos e análise de resultados de testes para os algoritmos propostos, com comparações diretas com algoritmos já consolidados na literatura, foram detalhados no capítulo 5. As análises realizadas consideraram bases de dados sintéticas, como prova de conceito, e bases de dados reais,

como exemplo de aplicação em mineração de texto. Também, as análises foram delineadas tanto sob o aspecto quantitativo como sob o aspecto qualitativo, apresentando considerações referentes a capacidade de reconstrução, de quantização e de extração de informação dos algoritmos. Então, em relação à capacidade de reconstrução, de quantização e de extração de informação dos algoritmos, os resultados apresentados indicaram que a fatoração $X \approx USV_{(1)}^T \dots V_{(k)}^T$, assim como os algoritmos para solução da fatoração proposta, agregam valor à área de análise de agrupamento e à solução de problemas referentes à mineração de texto.

Assim, a pesquisa discutida neste trabalho é de natureza aplicada, se caracterizando como sendo do gênero teórico-prática, uma vez que apresenta contribuições de ordem teórica para a área de coagrupamento, e também coloca o conhecimento científico produzido em condições de intervir na realidade, a partir da experimentação com dados reais. O método seguido na pesquisa teve um caráter exploratório, procurando proporcionar maior familiaridade com um problema, e adotou uma abordagem de análise mista (quali-quanti).

6.1 Contribuições

Nesse contexto, destacam-se como as principais contribuições deste trabalho de mestrado:

- proposição do problema e um algoritmo para *OvNMTF*, derivado com base na fatoração *BVD* já proposta na literatura de fatoração de matrizes não-negativas para coagrupamento, acompanhado de uma derivação formal das regras de atualização para as matrizes $U, S, V_{(1)}, \dots V_{(k)}$;
- proposição do problema e um algoritmo para *BinOvNMTF*, derivado com base na fatoração *FNMTF* já proposta na literatura de fatoração de matrizes não-negativas para coagrupamento, acompanhado de uma derivação formal das regras de atualização para as matrizes $U, S, V_{(1)}, \dots V_{(k)}$;
- apresentação de uma interpretação semântica para o novo problema de fatoração de matrizes não-negativas com sobreposição de colunas aplicados à dados textuais;
- construção e apresentação das bases de dados de notícias *IG* e *IG toy*, que se mostram importantes para o desenvolvimento da pesquisa de mineração de texto no idioma português (brasileiro).

Os resultados obtidos com essa pesquisa permitiram mostrar que foi possível propor novos tipos de fatoração de matrizes para a solução da tarefa de coagrupamento com sobreposição de colunas de forma natural, isto é, com independência de cogrupos de colunas.

Em termos de reconstrução e capacidade de quantização do espaço, por meio dos experimentos em bases de dados sintéticas, foi verificado a superioridade ou equivalência dos algoritmos desenvolvidos quando comparados aos algoritmos da literatura. Essa análise foi realizada usando inspeção visual de reconstruções de conjuntos de dados e o erro de quantização.

Em termos de capacidade de agrupamento, experimentos em bases de dados reais com o uso de medidas externas clássicas para validação de agrupamento (RI e NMI), permitiram verificar a superioridade das estratégias desenvolvidas em relação às outras estratégias de fatoração. Em relação às técnicas de agrupamento tradicionais (*k-means* e *fuzzy k-means*), as estratégias mostraram capacidade de agrupamento superior ou equivalente considerando as bases de dados textuais aplicadas (*IG toy*, *IG* e *NIPS*).

Em termos de geração de informação, foi verificado que os algoritmos desenvolvidos são capazes de gerar novos tipos de informações que podem ser úteis para aplicações de mineração de texto. Tal tipo de informação, ou forma de interpretação de resultados, não havia ainda sido proposta no contexto de fatoração de matrizes para a tarefa de coagrupamento. A análise dos experimentos qualitativos para o *OvNMTF*, para bases de dados textuais, permite fazer essa afirmação.

Embora os objetivos tenham sido alcançados e os resultados obtidos tenham se mostrado promissores, há algumas desvantagens e limitações a serem consideradas nas soluções apresentadas:

- tempo de execução dos algoritmos: nos experimentos realizados foi percebido que os algoritmos desenvolvidos exigem maior tempo de execução que os demais, no entanto, são necessários experimentos para medir quantitativamente essa diferença;
- validade dos resultados considerando testes estatísticos: nenhum dos experimentos apresentados neste trabalho fez o uso de testes estatísticos para afirmar com maior segurança quais os melhores algoritmos em termos de capacidade de agrupamento;
- convergência dos algoritmos desenvolvidos: foram realizados testes empíricos verificando a convergência dos algoritmos *OvNMTF* e *BinOvNMTF*, e portanto ainda

são necessárias as verificações matemáticas de convergência, assim como experimentos considerando a convergência na execução dos algoritmos para diferentes parametrizações.

6.2 Trabalhos futuros

Durante a realização dessa pesquisa, algumas hipóteses para explicar alguns comportamentos advindos dos algoritmos foram levantadas. Essas hipóteses abrem espaço para o desenvolvimento de pesquisas mais específicas, direcionadas a análise de pontos que não foram analisados neste trabalho. Tais pontos são:

- estudar a determinação do parâmetro k (número de cogrupos de linhas): como em todos os experimentos já se conhecia de antemão esse parâmetro, considerando-o como o número de classes existentes no conjunto de dados, a sua variação não foi considerada nos experimentos. Seria necessário aplicar índices de validação internos de agrupamento e experimentos qualitativos a fim de verificar a qualidade dos agrupamentos que seriam criados a partir da variação do k;
- estudar com mais detalhes a vantagem da independência dos cogrupos de colunas diante de contextos de aplicação: foram realizados experimentos qualitativos para ilustrar a vantagem da influência da independência em cogrupos com sobreposição de colunas. Contudo, considera-se que a formulação de experimentos que permitam a análise quantitativa detalhada desse quesito, pode melhorar a motivação para o uso das estratégias propostas;
- estudar o efeito da estratégia de aplicação de múltiplas matrizes ao problema *NMF*: esse seria um outro tipo de fatoração que poderia trazer resultados interessantes para a tarefa de agrupamento, até possivelmente, para a tarefa de coagrupamento;
- estudar com maior profundidade as restrições de ortogonalidade dos algoritmos: em alguns casos as restrições de ortogonalidade contribuíram muito para obter resultados melhores considerando a capacidade de agrupamento, então, são necessários estudos para verificar as vantagens dessa estratégia. Além disso, ainda é possível a proposição de um novo problema e algoritmo considerando restrições ortogonais no problema *OvNMTF*, semelhante ao que foi feito nos trabalhos que apresentaram o *ONMTF*.

Referências[1]

ALOISE, D. et al. Np-hardness of euclidean sum-of-squares clustering. *Machine Learning*, v. 75, n. 2, p. 245–248, 2009. ISSN 1573-0565. Citado na página 27.

BAUCKHAGE, C. *k-Means Clustering Is Matrix Factorization*. 2015. Citado na página 28.

BEZDEK, J. C. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Norwell, MA, USA: Kluwer Academic Publishers, 1981. ISBN 0306406713. Citado 3 vezes nas páginas 15, 29 e 30.

BOTTOU, L.; BENGIO, Y. Convergence properties of the K-means algorithm. In: . [S.l.: s.n.], 1995. p. 585–592. Citado na página 28.

BOYD, S.; VANDENBERGHE, L. *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004. ISBN 0521833787. Citado 2 vezes nas páginas 24 e 25.

BULTEAU, L. et al. Co-clustering under the maximum norm. In: _____. *Algorithms and Computation: 25th International Symposium, ISAAC 2014, Jeonju, Korea, December 15-17, 2014, Proceedings*. Cham: Springer International Publishing, 2014. p. 298–309. Citado na página 39.

CABANES, G.; BENNANI, Y.; FRESNEAU, D. Enriched topological learning for cluster detection and visualization. *Neural Networks*, v. 32, p. 186–195, 2012. Citado na página 37.

CHENG, Y.; CHURCH, G. M. Biclustering of expression data. In: *Procedures of the 8th ISMB*. [S.l.]: AAAI Press, 2000. p. 93–103. Citado na página 36.

CORMEN, T. H. et al. *Introduction to Algorithms*. 2nd. ed. [S.l.]: McGraw-Hill Higher Education, 2001. ISBN 0070131511. Citado 3 vezes nas páginas 44, 61 e 89.

DING, C.; HE, X.; SIMON, H. D. *On the equivalence of nonnegative matrix factorization and spectral clustering*. 2005. Citado 3 vezes nas páginas 27, 29 e 30.

DING, C. et al. Orthogonal nonnegative matrix tri-factorizations for clustering. In: *In SIGKDD*. [S.l.]: Press, 2006. p. 126–135. Citado 7 vezes nas páginas 20, 33, 38, 39, 44, 45 e 47.

FATAHALIAN, K.; SUGERMAN, J.; HANRAHAN, P. Understanding the efficiency of gpu algorithms for matrix-matrix multiplication. In: *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*. New York, NY, USA: ACM, 2004. (HWWS '04), p. 133–137. ISBN 3-905673-15-0. Citado na página 89.

FRANCA, F. de. *Biclusterização na Análise de Dados Incertos*. Tese (Doutorado) — Universidade Estadual de Campinas, Campinas, SP, BR, 11 2010. Citado 3 vezes nas páginas 16, 18 e 34.

FRANCA, F. de; ZUBEN, F. V. Finding a high coverage set of 5-biclusters with swarm intelligence. In: *Evolutionary Computation (CEC), 2010 IEEE Congress on*. [S.l.: s.n.], 2010. p. 1–8. Citado na página 37.

[1] De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

GETZ, G.; LEVINE, E.; DOMANY, E. Coupled two-way clustering analysis of gene microarray data. *Proc. Natl. Acad. Sci. USA*, v. 97, p. 12079–12084, 2000. Citado na página 36.

GLOBERSON, A. et al. Euclidean embedding of co-occurrence data. *The Journal of Machine Learning Research*, MIT Press Cambridge, MA, USA, v. 8, p. 2265–2295, 2007. Citado na página 83.

GOLUB, G. H.; LOAN, C. F. V. *Matrix Computations (3rd Ed.)*. Baltimore, MD, USA: Johns Hopkins University Press, 1996. ISBN 0-8018-5414-8. Citado na página 24.

GOVAERT, G.; NADIF, M. *Co-Clustering*. [S.l.]: Wiley, 2013. ISBN 9781848214736. Citado na página 34.

HALKIDI, M.; BATISTAKIS, Y.; VAZIRGIANNIS, M. Cluster validity methods: Part i. *SIGMOD Record*, ACM Press, v. 31, n. 2, p. 40–45, 2002. Citado 2 vezes nas páginas 31 e 32.

HALKIDI, M.; BATISTAKIS, Y.; VAZIRGIANNIS, M. Clustering validity checking methods: Part ii. *SIGMOD Rec.*, New York, NY, USA, v. 31, n. 3, p. 19–27, set. 2002. ISSN 0163-5808. Citado 2 vezes nas páginas 31 e 32.

HAN, J.; KAMBER, M.; PEI, J. *Data Mining: Concepts and Techniques*. 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN 0123814790, 9780123814791. Citado 6 vezes nas páginas 15, 26, 28, 29, 31 e 33.

HARTIGAN, J. A. Direct clustering of a data matrix. *Journal of the American Statistical Association*, American Statistical Association, v. 67, n. 337, p. 123–129, 1972. Citado na página 34.

HO, N.-D. *Nonnegative Matriz Factorization Algorithms and Applications*. Tese (Doutorado) — Université Catholique de Louvain, Louvain-la-Neuve, Belgique, 6 2008. Citado 3 vezes nas páginas 16, 33 e 38.

HOCHREITER, S. et al. Fabia: factor analysis for bicluster acquisition. *Bioinformatics*, v. 26, n. 12, p. 1520–1527, 2010. Citado na página 37.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: A review. *ACM Computing Surveys*, ACM, v. 31, p. 264 – 323, September 1999. Citado na página 15.

KLUGER, Y. et al. Spectral biclustering of microarray data: Coclustering genes and conditions. *Genome Research*, v. 13, n. 4, p. 703–716, 2003. Citado na página 38.

KOREN, Y. *1 The BellKor Solution to the Netflix Grand Prize*. 2009. Citado na página 38.

KUANG, D. *Nonnegative Matrix Factorization For Clustering*. Tese (Doutorado) — University of Tampere, Tampere, Finlândia, 2014. Citado 2 vezes nas páginas 33 e 38.

LEE, D. D.; SEUNG, H. S. Learning the parts of objects by nonnegative matrix factorization. *Nature*, v. 401, p. 788–791, 1999. Citado 2 vezes nas páginas 18 e 38.

LEE, D. D.; SEUNG, H. S. Algorithms for non-negative matrix factorization. In: *NIPS*. [S.l.: s.n.], 2000. p. 556–562. Citado na página 18.

LI, T.; DING, C. The relationships among various nonnegative matrix factorization methods for clustering. In: *Proceedings of the Sixth International Conference on Data Mining*. Washington, DC, USA: IEEE Computer Society, 2006. (ICDM '06), p. 362–371. ISBN 0-7695-2701-9. Citado na página 39.

LONG, B.; ZHANG, Z. M.; YU, P. S. *Co-clustering by block value decomposition*. [S.l.]: ACM Press, 2005. 635–640 p. Citado 8 vezes nas páginas 20, 38, 39, 40, 41, 42, 43 e 57.

LOPS, P.; GEMMIS, M. de; SEMERARO, G. Content-based recommender systems: State of the art and trends. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. [S.l.]: Springer, 2011. p. 73–105. Citado 2 vezes nas páginas 21 e 86.

MADEIRA, S. C.; OLIVEIRA, A. L. Biclustering algorithms for biological data analysis: A survey. *IEEE Transactions on Computational Biology and Bioinformatics*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 1, p. 24–45, January 2004. Citado 5 vezes nas páginas 18, 34, 35, 36 e 68.

MAGNUS, J.; NEUDECKER, H. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. [S.l.]: Wiley, 1999. ISBN 9780471986331. Citado 2 vezes nas páginas 24 e 25.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008. ISBN 0521865719, 9780521865715. Citado na página 33.

MIRKIN, B. *Mathematical Classification and Clustering*. [S.l.]: Kluwer Academic Pubishers, 1996. Citado na página 18.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 70.

PETERSEN, K. B.; PEDERSEN, M. S. *The Matrix Cookbook*. [S.l.]: Technical University of Denmark, 2012. Citado 2 vezes nas páginas 25 e 26.

RAND, W. M. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, v. 66, n. 336, p. 846–850, December 1971. Citado na página 32.

ROCHA, T. et al. Tutorial sobre fuzzy-c-means e fuzzy learning vector quantization: Abordagens híbridas para tarefas de agrupamento e classificação. *RITA - Revista de Informática Teórica e Aplicada*, v. 19, n. 1, p. 120–163, March 2012. Citado 5 vezes nas páginas 15, 28, 29, 30 e 73.

SALAKHUTDINOV, R.; MNIH, A. Probabilistic matrix factorization. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2008. v. 20. Citado na página 38.

SALTON, G.; WONG, A.; YANG, C. S. A vector space model for automatic indexing. *Communications of the ACM*, ACM, New York, NY, USA, v. 18, n. 11, p. 613–620, 1975. Citado 3 vezes nas páginas 21, 86 e 87.

SANDERSON, C. *Armadillo: An Open Source C++ Linear Algebra Library for Fast Prototyping and Computationally Intensive Experiments*. [S.l.], 2010. Citado na página 88.

SEBASTIANI, F. Machine learning in automated text categorization. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 34, n. 1, p. 1–47, 2002. Citado na página 86.

SHAHNAZ, F. et al. Document clustering using nonnegative matrix factorization. *Information Processing & Management*, v. 42, n. 2, p. 373 – 386, 2006. Citado na página 18.

SITES mais acessados do Brasil. 2016. Disponível em: <http://www.alexa.com/topsites-/countries;1/BR>. Citado na página 103.

TANAY, A.; SHARAN, R.; SHAMIR, R. Biclustering algorithms: A survey. In: *In Handbook of Computational Molecular Biology Edited by: Aluru S. Chapman & Hall/CRC Computer and Information Science Series*. [S.l.: s.n.], 2005. Citado na página 36.

TJHI, W.-C.; CHEN, L. Dual fuzzy-possibilistic coclustering for categorization of documents. *IEEE Transactions on Fuzzy Systems*, IEEE, v. 17, p. 533 – 543, June 2009. Citado na página 18.

WANG, H. et al. Fast nonnegative matrix tri-factorization for large-scale data co-clustering. In: *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*. [S.l.]: AAAI Press, 2011. (IJCAI'11), p. 1553–1558. ISBN 978-1-57735-514-4. Citado 5 vezes nas páginas 20, 33, 39, 48 e 62.

WANG, Q. et al. Augem: Automatically generate high performance dense linear algebra kernels on x86 cpus. In: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. New York, NY, USA: ACM, 2013. (SC '13), p. 25:1–25:12. ISBN 978-1-4503-2378-9. Citado na página 88.

XU, R.; WUNSCH, D. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, IEEE, v. 16, p. 645 – 678, May 2005. Citado na página 15.

XU, W.; LIU, X.; GONG, Y. Document clustering based on non-negative matrix factorization. In: *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*. New York, NY, USA: [s.n.], 2003. (SIGIR '03), p. 267–273. ISBN 1-58113-646-3. Citado 2 vezes nas páginas 18 e 33.

YANG, J.; LESKOVEC, J. Overlapping community detection at scale: A nonnegative matrix factorization approach. In: *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*. New York, NY, USA: ACM, 2013. (WSDM '13), p. 587–596. Citado na página 37.

YOO, J.; CHOI, S. Orthogonal nonnegative matrix tri-factorizations for co-clustering: multiplicative updates on stiefel manifolds. *Information Processing and Management*, v. 46, p. 559–570, 2010. Citado 12 vezes nas páginas 18, 20, 33, 38, 40, 45, 46, 47, 58, 67, 75 e 88.