LUCAS FERNANDES BRUNIALTI

Resolução do problema de coagrupamento em matrizes de dados esparsas usando fatoração de matrizes

São Paulo

2016

LUCAS FERNANDES BRUNIALTI

Resolução do problema de coagrupamento em matrizes de dados esparsas usando fatoração de matrizes

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

Ficha catalográfica

Dissertação de autoria de Lucas Fernandes Brunialti, sob o título **"Resolução do problema de coagrupamento em matrizes de dados esparsas usando fatoração de matrizes"**, 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: _____

Escreva aqui sua dedicatória, se desejar, ou remova esta página...

Agradecimentos

Texto de exemplo, texto de exemplo.

Texto de exemplo, texto de exemplo.

Texto de exemplo, texto de exemplo.

Texto de exemplo, texto de exemplo.

Texto de exemplo, texto de exemplo.

"Escreva aqui uma epígrafe, se desejar, ou remova esta página..."

(Autor da epígrafe)

Resumo

BRUNIALTI, Lucas Fernandes. **Resolução do problema de coagruapmento em matrizes de dados esparsas usando fatoração de matrizes**. 2016. 91 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 objetos, então denomidados cogrupos, que são similares entre si de acordo com um subconjunto dos seus atributos descritivos, e assim, um objeto pode pertencer a mais de um grupo se subconjuntos diferentes de atributos forem considerados. Essa característica pode ser particularmente útil para aplicações nas quais a similaridade parcial entre objetos faz sentido, conferindo ao resultado da análise de dados algumas características interessantes como serendipidade ou flexibilidade no modelo de agrupamento. Contextos de aplicação caracterizados por apresentar subjetividade, como mineração de textos, são candidatos a serem submetidos à estratégia de coagrupamento; a flexibilidade em associar textos de acordo com características parciais representa um tratamente mais adequado à tal subjetividade. Entretanto, análise de grupos considerando dados textuais representa um contexto no qual existe o problema de esparsidade de dados, que precisa ser adequadamente tratado para que os bons resultados sejam obtidos. 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, capazes de encontrar cogrupos organizados com sobreposição em uma matriz esparsa de valores reais positivos. As estratégias são apresentadas em termos de suas definições formais e seus algoritmos para implementação. Resultados experimentais são fornecidos a partir de problemas baseados em conjuntos de dados sintéticos e em conjuntos de dados reais contextualizados na área de mineração de textos. Os resultados confirmam a hipótese de que as estratégias propostas são capazes de descobrir cogrupos com sobreposição, e que tal organização de cogrupos fornece informação detalhada, e portanto de valor diferenciado, para a mineração de textos.

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

Abstract

BRUNIALTI, Lucas Fernandes. **Matrix Factorization for coclustering in sparse data matrices**. 2016. 91 p. Dissertation (Master of Science) – School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, DefenseYear.

Coagrupamento é uma estratégia para análise de dados capaz de encontrar grupos de objetos, então denomidados cogrupos, que são similares entre si de acordo com um subconjunto dos seus atributos descritivos, e assim, um objeto pode pertencer a mais de um grupo se subconjuntos diferentes de atributos forem considerados. Essa característica pode ser particularmente útil para aplicações nas quais a similaridade parcial entre objetos faz sentido, conferindo ao resultado da análise de dados algumas características interessantes como serendipidade ou flexibilidade no modelo de agrupamento. Contextos de aplicação caracterizados por apresentar subjetividade, como mineração de textos, são candidatos a serem submetidos à estratégia de coagrupamento; a flexibilidade em associar textos de acordo com características parciais representa um tratamente mais adequado à tal subjetividade. Entretanto, análise de grupos considerando dados textuais representa um contexto no qual existe o problema de esparsidade de dados, que precisa ser adequadamente tratado para que os bons resultados sejam obtidos. 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, capazes de encontrar cogrupos organizados com sobreposição em uma matriz esparsa de valores reais positivos. As estratégias são apresentadas em termos de suas definições formais e seus algoritmos para implementação. Resultados experimentais são fornecidos a partir de problemas baseados em conjuntos de dados sintéticos e em conjuntos de dados reais contextualizados na área de mineração de textos. Os resultados confirmam a hipótese de que as estratégias propostas são capazes de descobrir cogrupos com sobreposição, e que tal organização de cogrupos fornece informação detalhada, e portanto de valor diferenciado, para a mineração de textos.

Keywords: Coclustering. Matrix Factorization. Sparsity. Clustering Analysis. Text Mining.

Lista de figuras

Lista de algoritmos

Lista de tabelas

Lista de abreviaturas e siglas

Sigla/abreviatura 1	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 2	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 3	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 4	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 5	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 6	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 7	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 8	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 9	Definição da sigla ou da abreviatura por extenso
Sigla/abreviatura 10	Definição da sigla ou da abreviatura por extenso

Lista de símbolos

Γ	Letra grega Gama
Λ	Lambda
ζ	Letra grega minúscula zeta
$\in$	Pertence

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 intragrupo 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 (XU; WUNSCH, 2005; HAN; KAMBER, 2006), podem ser usadas para alcançar o objetivo da análise de agrupamento, e cada uma delas possui características que as fazem mais ou menos suscetiveis para conjuntos de dados de diferentes naturezas. Ainda, sob o contexto clássico da tarefa de agrupamento, os métodos precisam lidar com a similaridade intrínseca 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 (referência, referência, referência). Essa forma de tratar a tarefa de agrupamento permite melhorias no processo de descoberta de agrupamentos sobre 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 textos, por exemplo. Considere o contexto de um sistema de recomendação (SR) de notícias baseado em conteúdo (um conteúdo textual). Um SR de notícias simples e hipotético poderia apresentar a seguinte

estratégia para elaborar suas recomendações: dado um conjunto de notícias organizadas em grupos por um método de análise de agrupamento com base na similaridade de seus conteúdos; se um usuário visitar uma notícia, o SR verifica quais são as demais notícias pertencentes ao grupo daquela que foi lida pelo usuário e as recomenda para ele (Figura **??**a). Embora essas recomendações pareçam ser ideais, e sob algum aspecto de análise elas são, é factível assumir que esse usuário está recebendo um serviço de recomendação prático, mas talvez menos útil e interessante do que poderia ser e com baixa serendipidade (um resultado de alta serendipidade é aquele que é diferente do que é esperado e comumente praticado, e é mais ou igualmente útil ao contexto).

Uma possibilidade de melhoria nesse sistema hipotético seria usar um algoritmo de análise de agrupamento que permitisse descobrir uma organização de grupo de notícias baseada em **similaridades parciais** ou baseada **em partes** (FRANCA, 2010; HO, 2008). Assim o sistema seria dotado 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. A recomendação, nesse caso, seria potencialmente mais serendípita. 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 a esporte, notícias referentes a música, ou notícias referentes a esporte e música (Figura **??**(b, c, d)).

Na figura **??**(b,c,d) é introduzido graficamente o conceito de coagruapmento. Nesse contexto, o problema de mineração de textos é modelado como o problema de encontrar uma organização dos textos em grupos que considerem similaridades particiais. Assim, um texto pode pertencer a um ou mais cogrupos, a depender dos atributos descritivos sendo considerados. 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) são mutuamente submetidos a uma análise de similaridade, e portanto, grupos de dados (linhas) são estabelecidos com respeito a grupos de atributos (colunas).

A associação da análise de coagrupamentos a mineração de textos é interessante por diferentes aspectos. A mineração de textos constitui-se como um problema no qual é preciso lidar com a necessidade de apresentação de resultados com boa interpretabilidade e com um espaço dos dados de alta-dimensionalidade. O primeiro problema é 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. Ainda segundo (TJHI; CHEN, 2009), análise de coagrupamento pode apresentar bom desempenho em espaços de alta-dimensionalidade porque seu processo de agrupar atributos (características) pode ser visto com uma redução de dimensionalidade dinâmica para o espaço dos dados.

A despeito da capacidade intrínseca do processo de coagrupamento em lidar de forma diferenciada com o problema de alta-dimensionalidade, ainda se faz necessário notar que no contexto de mineração de dados, ocorre também o problema de esparsidade na representação dos dados. Assim, para implementar a estratégia de coagruapmento com alguma eficiência, é necessário adotar métodos que tenham a capacidade de lidar com esparsidade.

Dentre os diferentes métodos existentes na literatura referentes à implementação de análise de coagrupamento (citar artigos dos vários algoritmos que seguem outras linhas), 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 textos (XU; LIU; GONG, 2003; SHAHNAZ et al., 2006; YOO; CHOI, 2010).

1.1 Definição do problema

Eu estou entendendo que temos duas facetas do problema. Um deles é o mais obvio que é dar um jeito de descobrir os grupos com sobreposição e mostrar que é util para recomendação/textos etc. O outro é fazer a fatoração de matriz funcionar pra isso. Então acho que temos que dividir essa definição em duas partes: sobreposição nos cogrupos e fatoração funcionando nisso. Por isso dividi em duas partes, para ver se conseguimos mostrar isso.

1.1.1 Estruturas de coagrupamentos

Então aqui entra a parte de mostrar as estruturas de cogrupos possíveis e destacar aquela que fatoração já resolve e depois a que não resolve e a que queremos resolver. Também contextualizar no problema de recomendação ou análise de textos. Figuras precisam entrar aqui para mostra as estruturas.

1.1.2 Coagrupamento e fatorização 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 deles diz respeito a objetos que são similares entre si considerando um **grupo de atributos** (colunas), também associado ao cogrupo.

Com maior formalidade, dada uma matriz $X(N, M)$ (prefiro dizer $X \in \mathbb{R}^{N \times M}$) em que N é o número de linhas, M o número de colunas e $x(m, n)$ é, geralmente, um número real representando a relação entre a linha x_n (linha n?) e a coluna y_m (coluna m?), o problema de **coagrupamento** consiste em encontrar um conjunto $\mathcal{C}$ de submatrizes $G(I, J)$, onde $I = \{i_1, ..., i_r\}$ com $r \leq L$ e $J = \{j_1, ..., j_s\}$ com $s \leq C$, que maximize a similaridade entre os elementos $g\{i, j\}$. (Quem é L e C?)

A **esparsidade** em uma matriz é caracterizada pela existência de poucos elementos diferentes de zero (0). Em termos gerais, a esparsidade de uma matriz pode ser medida como a proporção de elementos iguais a zero (0) que ela contém. Problemas de otimização que envolvem matrizes esparsas são caracterizados por apresentarem alta complexidade combinatorial para os quais algoritmos eficientes em matrizes não esparsas tem seu desempenho bastante prejudicado.

Fatorar uma matriz consiste em encontrar duas, ou mais, novas matrizes que ao serem multiplicadas, reconstroem a matriz original. Considere uma matriz $R(N, M)$ em que N é o número de linhas, M o número de colunas. A fatoração desta matriz em duas novas matrizes consiste em encontrar duas matrizes $U(N, K)$ e $D(M, K)$, tal que $R = U \times D^t = \hat{R}$ (Por que colocar $\hat{R}$? Só se for para dizer $R \approx U \times D^t = \hat{R}$). Se K é escolhido tal que seja menor do que N e M, então é dito que U e D são representações compactas de R (Estranho, pois se $K = M - 1$ parece que precisaremos de mais números

e não haverá nenhuma compactação de dados). Se a matriz R, e as suas decomposições, são não negativas, tem-se o caso de fatoração de matriz não-negativa.

O problema de coagrupamento pode ser modelado de tal forma que a fatoração de matriz é capaz de fornecer uma aproximação da organização em cogrupos presente no conjunto de dados sob análise.

Considere que o conjunto de dados sob análise é representado pela matriz $X(N, M)$, a fatoração dessa matriz em duas (ou mais) novas matrizes $U(N, K)$ e $D(M, K)$ significa que K grupos de linhas foram descobertos, de acordo com K grupos de colunas.

Se três matriz são geradas na fatoração, $U(N, L)$, $S(L, K)$ e $D(M, K)$, a interpretação pode incluir uma noção de pesos (matriz S) que relacionam grupos de linhas e grupos de colunas, e dimensões diferentes para as matrizes U e D podem ser admitidas de modo que o número de grupos de linhas pode ser diferente do número de grupo de colunas.

Imagino que agora tem que entrar uma apresentação rápida do algoritmo novo.

1.2 Hipótese

Fatoração de matrizes considerando a decomposição da matriz original em ... como descrever em algo nível aqui?? ... possibilita a descoberta de cogrupos com sobreposição (de colunas); a partir das novas matrizes é possível extrair informação detalhada sobre a relação dos grupos de linhas em relação ao grupo de colunas que pode agregar valor à solução de um problema real de recomendação.

1.3 Objetivos

O objetivo geral desse trabalho é o desenvolvimento de novas estratégias de coagrupamento baseadas em fatoração de matrizes, que sejam capazes de descobrir cogrupos com sobreposição em uma dimensão da matriz, isto é, ou sobreposição de colunas ou sobreposição de linhas, considerando uma matriz de valores reais positivos. Com a proposição dessas novas estratégias, este trabalho cobre uma lacuna presente na área de coagrupamento baseado em algoritmos de fatoração de matrizes.

Este trabalho tem como objetivos específicos a aplicação das novas estratégias em um contexto de aplicacão real, de forma a ilustrar que elas

Como objetivos específicos, este trabalho mostra que a aplicação das novas estratégias em ambientes controlados (matrizes de dados com cogrupos sintéticas) e em um contexto de aplicação real:

- alcançam resultados tão bons quanto, ou melhores que, as estratégias correlatas que não permitem a sobreposição de dimensões nos cogrupos, quando medidas de qualidade quantitativas são consideradas;
- alcança resultados tão bons quanto, ou melhores que, estratégias de agrupamento quando análise de particionamento clássico e medidas de qualidade quantitativas são consideradas;
- é capaz de melhorar a interpretabilidade qualitativa dos resultados quando comparada aos resultaods fornecidos por estratégias de agrupamento clássico.

1.4 Metodologia

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 coagrupamento.

E não estou conseguindo encontrar uma forma de descrever a parte referente à concepção das estratégias propostas e também não sei como definir as estratégias referente às derivações.

A fim de permitir a validação das estratégias propostas e, portanto, a verificação da hipótese, fez-se necessário a definição de: (a) um ambiente de teste controlado, representado por uma coleção de conjuntos de dados sintéticos, contendo cada um dos conjuntos situações diferentes referentes às estrutura de coagrupamento e variações em relação à esparsidade; (b) um contexto para realização de uma prova de conceito, no qual um conjunto de dados real foi construído.

Para a prova de conceito foi escolhido usar o conteúdo referente à notícias publicadas no portal iG[1]. Trata-se de um portal de notícias brasileiro muito conhecido, com um volume de notícias bastante grande e com notícias categorizadas em canais, que representam os assuntos dessas notícias. Essas características conferem liberdade para a configuração

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

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*, filtragem de *stopwords*, remoção de sufixos (*stemming*), representação da relação "termos × documentos" usando estratégias de frequência de termos, como TF-IDF e *n-grams*.

Os resultados da aplicação das estratégias de coagrupamento foram validados utilizando técnicas de avaliação interna, para a verificação da consistência dos biclusters encontrados (SANTAMARíA; MIGUEL; THERóN, 2007), e externas (HOCHREITER et al., 2010), avaliando o quanto os biclusters encontrados estão em consenso com as classes de notícias (HOCHREITER et al., 2010).

Então precisaremos dizer aqui como fizemos a avaliação qualitativa.

1.5 Organização do documento

Esta dissertação é composta por XXX capítulos incluindo esta introdução. Os demais capítulos estão divididos em duas partes: a primeira é dedicada a explorar a estratégia de coagrupamento implementada com algoritmos baseadas em fatoração de matrizes; a segunda é dedicada a explorar o contexto de sistemas de recomendação baseados em conteúdo textual a partir da aplicação das estratégias de coagrupamento estudadas.

No capítulo 2 são apresentados os principais conceitos referentes à área de coagrupamentos. Especificar mais detalhes ... ...

Estratégias de fatorização de matrizes aplicadas à coagrupamentos são discutidas no capítulo .....

A principal contribuição deste trabalho, as estratégias ... ., é apresentada em detalhes no capítulo ... ...

2 Conceitos Fundamentais

Esses conceitos fundamentais ainda vem na qualificação. Não estão ajustados ao que está sendo discutido agora, para a fase final.

Técnicas e algoritmos de Biclusterização são usados, principalmente, no contexto de expressão genética. No entanto, algoritmos de Biclusterização se fazem úteis quando se deseja encontrar *modelos locais*. Ou seja, enquanto algoritmos de clusterização têm o intuito de encontrar *modelos globais*, que geram grupos de dados levando em consideração todas as características, algoritmos de Biclusterização geram grupos de dados em que as características tem alta correlação (FRANCA, 2010; MADEIRA; OLIVEIRA, 2004).

Para a descrição do problema formal de Biclusterização usa-se a seguinte definição (MADEIRA; OLIVEIRA, 2004): seja uma matriz A, de dimensão $N \times M$, um conjunto de linhas $X = \{x_1, \ldots, x_n, \ldots, x_N\}$ (aqui deveria ser $X = \{1, 2, \ldots, n, \ldots, N\}$) e um conjunto de colunas $Y = \{y_1, \ldots, y_m, \ldots, y_M\}$ (mesmo comentário anterior), em que a_{nm} geralmente é um número real e representa a relação entre a linha x_n e a coluna y_m; o problema de Biclusterização é encontrar biclusters, que são submatrizes de A, denotados por A_{IJ}, em que $I \subseteq X$ e $J \subseteq Y$. Assim, o bicluster A_{IJ} é um grupo dos objetos em I, perante as características com alta correlação J.

2.1 Tipos de cogrupos

Como a definição de bicluster não inclui uma prévia estrutura da matriz A e dos biclusters A_{IJ}, diversos algoritmos propostos na literatura diferem quanto ao tipo de bicluster que são capazes de encontrar. Uma taxonomia dos tipos de biclusters é proposta por Madeira e Oliveira (2004):

- *Biclusters com valores constantes*, se trata de biclusters em que todos os valores de A_{IJ} são constantes: $a_{ij} = \mu, \forall i, j \in I, J$, (aqui o valor de μ não deveria ser indexado por IJ, isto é, μ_{IJ}? O mesmo para os outros μ que aparecem abaixo.) onde μ é um valor constante dentro de A_{IJ}. Porém, em conjuntos de dados reais, esses biclusters estão presentes com algum tipo de ruído $\mu + \eta_{ij}$, onde η_{ij} é o ruído associado com os valores de μ e a_{ij} (MADEIRA; OLIVEIRA, 2004).

- *Biclusters com valores constantes nas linhas ou colunas*, se trata de biclusters com valores constantes nas linhas: $a_{ij} = \mu + \alpha_i, \forall i, j \in I, J$ ou $a_{ij} = \mu \cdot \alpha_i, \forall i, j \in I, J$, onde α_i é um fator aditivo ou multiplicativo para cada linha; ou ainda biclusters com valores constantes nas colunas: $a_{ij} = \mu + \beta_j, \forall i, j \in I, J$ ou $a_{ij} = \mu \cdot \beta_j, \forall i, j \in I, J$, onde β_j é um fator aditivo ou multiplicativo para cada coluna (MADEIRA; OLIVEIRA, 2004).
- *Biclusters com valores coerentes*, em que são considerados valores próximos entre si (coerentes) para definição de um bicluster: $a_{ij} = \mu + \alpha_i + \beta_j, \forall i, j \in I, J$, ou $a_{ij} = \mu' \cdot \alpha'_i \cdot \beta'_j, \forall i, j \in I, J$, sendo que se $\mu = \log \mu' \implies \alpha_i = \alpha'_i, \beta_j = \beta'_j$ (MADEIRA; OLIVEIRA, 2004).
- *Biclusters com evoluções coerentes* têm seus valores com evoluções coerentes, por exemplo, um bicluster com $a_{i4} \leq a_{i3} \leq a_{i2} \leq a_{i1}$ tem valores com evolução coerente na coluna (MADEIRA; OLIVEIRA, 2004) (estranho considerar a ordem das colunas ou das linhas, já que na maioria dos problemas pode ser bem arbitrário.). Seus valores podem ser gerados por uma função geradora de valores com evolução coerente $a_{ij} = g(a_{ij}), \forall i, j \in I, J$, sendo $g(\cdot)$ não linear e não constante, para que o tipo de bicluster não seja classificado nos casos anteriores. (Muito estranho $a = g(a)$, pois $g()$ deveria ser a identidade)

Os biclusters também diferem quanto as suas estruturas. Cada algoritmo usado para implementar Biclusterização faz uma suposição da estrutura de biclusters que é capaz de encontrar. A Figura **??** sumariza as diferentes estruturas de biclusters, com as linhas e colunas ordenadas para permitir a visualização dos biclusters por meio do mapa de calor dos valores de A, sendo os biclusters A_{IJ} representados por cores sólidas e o fundo da matriz ruído.

2.2 Algoritmos para Biclusterização

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

Um dos algoritmos de Biclusterização mais comum e simples. que encontra biclusters com valores coerentes, em estrutura com sobreposição e arbitrariamente posicionados, é o *Coupled Two-way Clustering* (CTWC) (GETZ; LEVINE; DOMANY, 2000). O algoritmo

CTWC é capaz de encontrar biclusters através da clusterização de objetos e atributos (linhas e colunas), separadamente. O algoritmo de clusterização usado por Getz, Levine e Domany (2000) foi o *Superparamagnetic Clustering* (SPC), o qual é capaz de determinar o número de clusters automaticamente, e com uma estratégia de clusterização hierárquica *top-down* é capaz de gerar clusters estáveis (GETZ; LEVINE; DOMANY, 2000). O SPC tem como entrada uma matriz de similaridade e um parâmetro temperatura, que controla o quão estáveis serão os clusters que o algoritmo gerará. Assim, o CTWC encontra clusters estáveis de linhas e colunas através do SPC, e iterativamente executa o SPC nos clusters de linhas e colunas encontrados, mantendo na memória um par do subconjunto de linhas e do subconjunto de colunas (biclusters), assim como os clusters estáveis de linhas e colunas, separadamente.

Já o algoritmo de Cheng e Church (2000) é capaz de encontrar o mesmo tipo de bicluster que o algoritmo CTWC, porém usando uma estratégia gulosa: biclusters com valores coerentes e estrutura com sobreposição e arbitrariamente posicionados. Este algoritmo esta sendo objeto de estudo desse projeto de mestrado para aplicação em dados textuais e por isso segue aqui descrito em mais detalhes. Nesse algoritmo, para encontrar biclusters, ou δ-biclusters, na matriz A, os autores definem o *Resíduo Quadrático Médio* (RQM):

$$H_{IJ} = \frac{1}{|I||J|} \sum_{i,j \in I,J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2$$

$$H_{iJ} = \frac{1}{|J|} \sum_{j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2$$

$$H_{Ij} = \frac{1}{|I|} \sum_{i \in I} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2$$

em que

$$a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}, \quad a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}, \quad a_{IJ} = \frac{1}{|I||J|} \sum_{i,j \in I,J} a_{ij}$$

onde H_{IJ} é o RQM de uma submatriz A_{IJ}, H_{iJ} o RQM da linha i, H_{Ij} o RQM da coluna j, a_{iJ} a média dos valores da linha i, a_{Ij} a média dos valores da coluna j e a_{IJ} a média dos valores da submatriz A_{IJ}, definida pelos subconjuntos I e J.

Então, um bicluster perfeito A_{IJ} teria o RQM $H_{IJ} = 0$, pois $a_{ij} = a_{ij}, \forall i, j \in I, J$, ($a_{ij} = a_{ij}$, hein? não seria $a_{ij} = a_{iJ} + a_{Ij} - a_{IJ}$?) fazendo $a_{iJ} = a_{Ij} = a_{IJ}$. No entanto, se apenas minizar o RQM, um bicluster com apenas um elemento seria perfeito, o que pode

não refletir a realidade. Além disso, em conjunto de dados reais existe ruído, podendo esconder o bicluster perfeito.

Para encontrar biclusters, ou δ-biclusters, Cheng e Church (2000) usam uma estratégia gulosa que retira linhas e colunas, visando a minimização do RQM, respeitando um parâmetro δ, que é calibrado pelo usuário. Então, um bicluster é encontrado quando o RQM de uma submatriz A_{IJ} é $H_{IJ} \leq \delta$, para algum $\delta \geq 0$. As etapas de remoções de elementos da matriz são apresentadas nos algoritmos 1 e 2.

O algoritmo 2 é usado para acelerar o processo de busca de um δ-bicluster, convergindo mais rapidamente para uma solução quanto maior for o parâmetro α, em que $\alpha \geq 0$. Ainda, para amenização do problema de encontrar δ-biclusters perfeitos com apenas um elemento, ou poucos elemento, é utilizado o algoritmo 3, que adiciona nós sem aumentar o RQM do bicluster.

Por fim, o algoritmo 4 é a consolidação dos algoritmos 3, 2 e 1 e a iteração para encontrar k δ-biclusters, um a um, sendo k fornecido pelo usuário.

Além dos algoritmos apresentados, existem outros algoritmos que são capazes de encontrar outros tipos de biclusters (Seção 2.1), além de serem recentes (FRANçA; ZUBEN, 2010; YANG; LESKOVEC, 2013; HOCHREITER et al., 2010; CABANES; BENNANI; FRESNEAU, 2012), mostrando que ainda há interesse na área de pesquisa de Biclusterização.

2.3 Avaliação de Biclusterização

Para determinar parâmetros, descobrir a qualidade e/ou estabilidade dos biclusters encontrados por algoritmos, é necessário estabelecer métricas de avaliação. Existem duas maneiras de avaliar biclusters (HOCHREITER et al., 2010): *interna*, usa os dados dos resultados dos algoritmos, juntamente com métricas de qualidade e/ou estabilidade, para avaliar as soluções geradas; *externa*, utiliza os dados reais das soluções de biclusters de um conjunto de dados, usando estratégias para comparação, obtendo assim, maior confiança nas soluções.

A avaliação interna pode não ser tão precisa quanto a avaliação externa, porém é útil para descobrir parâmetros ótimos. Apesar de PreliĆ et al. (2006) sugerirem não usar avaliações internas, por não estar claro como estender noções de separação e homogeneidade, Santamaría, Miguel e Therón (2007) descreveu métricas de consistência para verificar se

um bicluster é consistente com a sua definição, seja aditiva, multiplicativa e/ou constante, fazendo uma comparação dos elementos do bicluster:

$$C_l(A_{IJ}) = \frac{1}{|I|} \sum_{i=1}^{|I|-1} \sum_{j=i+1}^{|I|} \sqrt{\sum_{k=1}^{|J|} (a_{ik} - a_{jk})^2}$$

$$C_c(A_{IJ}) = \frac{1}{|J|} \sum_{i=1}^{|J|-1} \sum_{j=i+1}^{|J|} \sqrt{\sum_{k=1}^{|I|} (a_{ki} - a_{kj})^2}$$

em que $C_l(A_{IJ})$ é o índice de consistência das linhas do bicluster A_{IJ} e $C_c(A_{IJ})$ é o índice de consistência das colunas do bicluster A_{IJ}. Ainda, a consistência do bicluster inteiro C pode ser definida pela média:

$$C(A_{IJ}) = \frac{|I| \cdot C_l + |J| \cdot C_c}{|I| + |J|}$$

Uma das métricas externas que são usadas para comparar biclusters encontrados com biclusters reais em um conjunto de dados, é a métrica *concensus score* (HOCHREITER et al., 2010). Essa métrica calcula a maximização das similaridades entre biclusters encontrados e reais, usando o *índice de Jaccard* como medida de similaridade e o algoritmo Húngaro para solucionar o problema de maximização. A saída da avaliação é um *score* $\in [0, 1]$, em que 0 significa que os biclusters comparados são totalmente diferentes, e 1 o inverso.

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 (coclustering). Ainda, outros contextos são submetidos à análise sob a formulação de problemas de coagrupamento, sendo alguns exemplos a clusterização de genes e análise de microarry 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 textos, 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 $i = \{1, \ldots, n\}$ e $j = \{1, \ldots, m\}$, 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 $p = \{1, \ldots, k\}$ e $q = \{1, \ldots, l\}$. Então, os subconjuntos $\{\mathcal{K}_1, \ldots, \mathcal{K}_p\}$ e $\{\mathcal{L}_1, \ldots, \mathcal{L}_l\}$ são os cogrupos de linhas e colunas, respectivamente.

Para implementação da NMF como uma estratégia para resolução do problema de coagruamento, diferentes algoritmos foram apresentados na literatura. Cada uma delas 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.

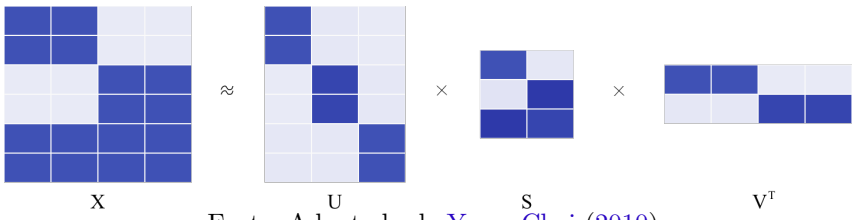
3.1 Decomposição de Valores em Blocos para Coagrupamento

A Decomposição de Valores em Blocos (*Block Value Decomposition* - BVD) foi proposto por Long, Zhang e Yu (2005) como uma abordagem para tratar o problema de coagruapmento, com base em fatoração de matrizes não-negativas. Esta 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 à partições encontradas no conjunto de linhas, e que associa as colunas à partições encontradas no conjunto de colunas, respectivamente; e S pode ser vista como uma representação compacta da matriz original de entrada 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 visto na figura 1.

Figura 1 – Fatoração da matriz original de dados X em três outras matrizes: U, S e V

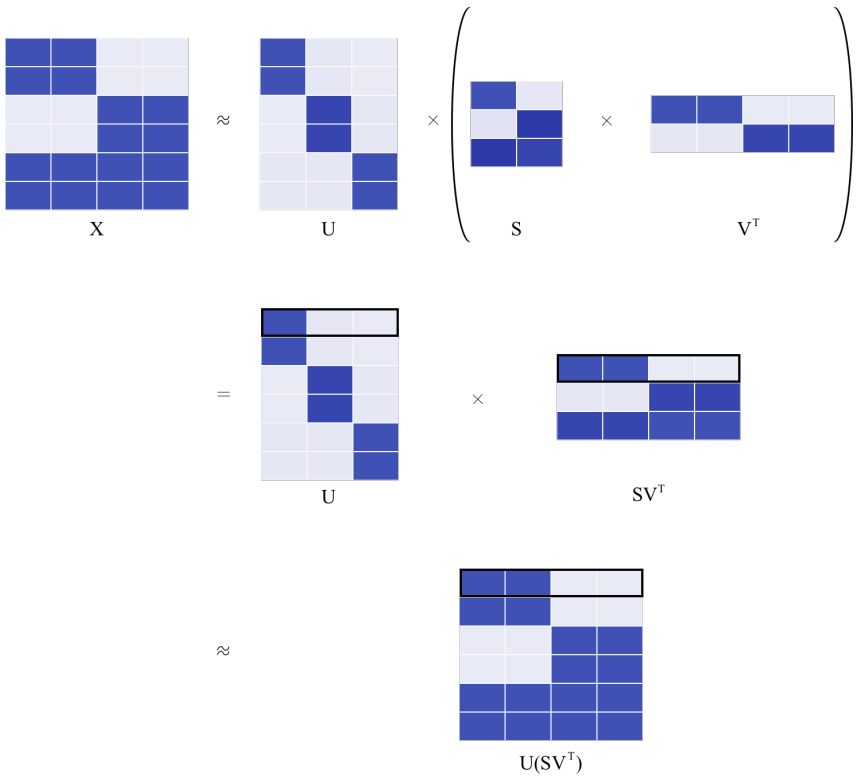


Fonte: Adaptado de Yoo e Choi (2010)

Na figura 1 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 a 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 2. 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.

Figura 2 – 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

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 2 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.[2]

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, atravéz das matrizes U, S e V.

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

Problema 1 (Problema de Decomposição de Valores em Blocos)**.**

$$\begin{aligned} \mathcal{F}_1(U,S,V) \quad = \quad & \min_{U,S,V} \quad \left\|X - USV^T\right\|_F^2 \\ & \textit{suj. a} \qquad U \geq 0, \\ & \qquad\qquad S \geq 0, \\ & \qquad\qquad V \geq 0 \end{aligned}$$

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 1 é descrito no algoritmo 1, o qual é baseado em atualizações multiplicativas e tem sua convergência demonstrada via teoria de otimização não linear (veja (LONG; ZHANG; 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.

[2] Medidas de avaliação de agruapmento ou coagrupamento internas podem ser aplicadas às diferentes soluções encontradas de maneira a guiar um processo decisório no contexto de aplicação.

Algoritmo 1 Algoritmo baseado em atualização multiplicativa para solução do BVD

1: **function** BVD(X, t_{max}, k, l)
2: **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 $t \leftarrow 0$.
3: **while** (não convergiu) ou ($t \leq t_{max}$) **do**
4:
$$U^{(t+1)} \leftarrow U^{(t)} \odot \frac{XV^{(t)}S^{(t)^T}}{U^{(t)}S^{(t)}V^{(t)^T}V^{(t)}S^{(t)^T}}$$
5:
$$V^{(t+1)} \leftarrow V^{(t)} \odot \frac{X^TU^{(t+1)}S^{(t)}}{V^{(t)}S^{(t)^T}U^{(t+1)^T}U^{(t+1)}S^{(t)}}$$
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: **return** $U^{(t)}, S^{(t)}, V^{(t)}$
10: **end function**

A inicialização dos elementos das matrizes U, S e V são gerados através de uma distribuição uniforme que ignora zeros ($\mathcal{U}(0,1) \in]0,1]$). 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 t-ésima iteração seja igual ao número máximo de iterações (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, é o seguinte:

$$\mathcal{K}_p = \mathcal{K}_p + \{x_{i\cdot}\} \mid p = \arg\max_{p'} \mathbf{u}_{i\cdot}\ \forall i = \{1,\ldots,n\}, \forall p, p' = \{1,\ldots,k\}$$

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 $k \simeq l$, $n \simeq m$, $k << n$, $l << m$ e usando um algoritmo para otimizar a ordem das multiplicações (Cormen et al. (2001) discute algoritmos para tal otimização): $\mathcal{O}\Big(t_{max}\big(nl(m+k) + ml(n+k) + mk(n+l) + k^2(n+l) + l^2(m+k)\big)\Big)$.

3.2 Fatoração Ortogonal Tripla de Matrizes Não-negativas

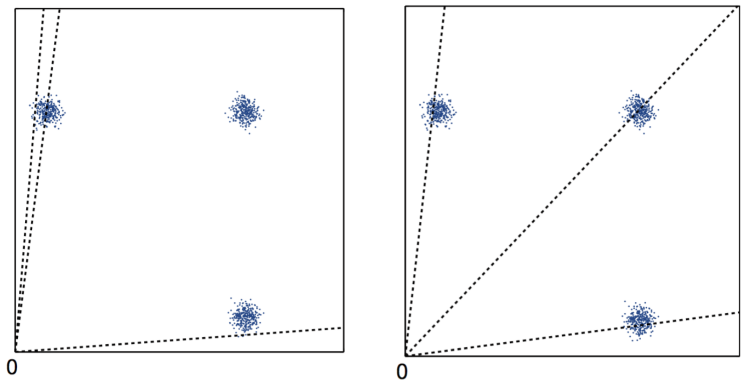
Baseado no problema 1, Ding et al. (2006) propõem o problema 2, e o chama de Fatoração Ortogonal Tripla de Matrizes Não-negativas (*Orthogonal Non-negative Matrix Tri-factorization* - ONMTF).

Problema 2 (Problema de Fatoração Ortogonal tripla de Matrizes Não-negativas)**.**

$$\begin{aligned} \mathcal{F}_2(U,S,V) \quad = \quad & \min_{U,S,V} \qquad \left\|X - USV^T\right\|_F^2 \\ & \textit{suj. a} \quad U \geq 0, S \geq 0, V \geq 0, \\ & \qquad\qquad U^TU = I, \\ & \qquad\qquad V^TV = I \end{aligned}$$

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 problemas, 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 3. No entanto, o algoritmo não garante que as restrições de ortonomalidade são respeitadas, apesar das restrições colocadas encontrarem soluções de particionamento com maior ortogonalidade.

Figura 3 – Base de protótipos obtidas com FMN sem restrições (a) e com restrições de ortogonalidade nas matrizes



(a) (b)

Fonte: Yoo e Choi (2010)

A figura 3 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.

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, atravéz da fatoração em U, S e V.

Ding et al. (2006) propõem uma solução para implementação do processo de minimização para o problema 2 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 2, 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 que ignora zeros. Também, a mesma condição de convergência aplicada no algoritmo 1 pode ser aplicada nesse caso.

Algoritmo 2 Algoritmo baseado em atualização multiplicativa para solução do ONMTF

function ONM3F(X, t_{max}, k, l)
  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) ou ($t \leq t_{max}$) **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

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 $k \simeq l$, $n \simeq m$, $k << n$, $l << m$ e usando um algoritmo para otimizar a ordem

das multiplicações, é possível encontrar a mesma complexidade antes calculada para o algoritmo 1.

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 $\nabla\mathcal{J}$:

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

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 2. 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 2 (ONMTF) alternativa às atualizações das equações 1, 2 e 3, através da atualização multiplicativa. Essas atualizações são apresentadas no algoritmo 3, 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 que ignora zeros, 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 das restrições $\left\|U^TU - I\right\|$ 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 3 Algoritmo baseado em atualização multiplicativa e na teoria de de derivação na superfície com restrições (Variedade Stiefel) para solução do ONMTF

function ONMTF(X, t_{max}, k, l)
  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) ou ($t \leq t_{max}$) **do**

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

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

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

    $t \leftarrow t + 1$
  end while

$$U^{(t)} \leftarrow U^{(t)}\,\mathrm{diag}(S^{(t)}\,\mathrm{diag}(\mathbf{1}^TV^{(t)})\mathbf{1})$$

$$V^{(t)} \leftarrow V^{(t)}\,\mathrm{diag}(\mathbf{1}^T\,\mathrm{diag}(\mathbf{1}^TU^{(t)})S^{(t)})$$

  return $U^{(t)}, S^{(t)}, V^{(t)}$
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 3. 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 1 e 2 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 3,proposto por Yoo e Choi (2010) para solução do problema 2 (ONMTF), é equivalente à complexidade apresentada para o algoritmo 1 e 2.

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 3, 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 3 (Fatoração tripla rápida de Matrizes Não-negativas)**.**

$$\begin{aligned} \mathcal{F}_3(U,S,V) \quad = \quad & \min_{U,S,V} \quad \left\|X - USV^T\right\|_F^2 \\ & U \in \Psi^{n\times k}, \\ & V \in \Psi^{m\times l} \end{aligned}$$

em que $S \in \mathbb{R}_+^{k\times l}$, $\Psi = \{0,1\}$, $\sum_{p=1}^{k} \mathbf{u}_{p\cdot} = 1$ e $\sum_{q=1}^{l} \mathbf{v}_{q\cdot} = 1$.

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 $\sum_{p=1}^{k} \mathbf{u}_{p\cdot} = 1$ e $\sum_{q=1}^{l} \mathbf{v}_{q\cdot} = 1$ 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.

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 $\mathcal{F}_3$:

$$\begin{aligned}
\nabla_S \mathcal{F}_3 &= U^T X V - U^T U S V^T V &&= 0 \\
&\implies U^T U S V^T V &&= U^T X V \\
&\implies (U^T U)^{-1} U^T U S V^T V (V^T V)^{-1} &&= (U^T U)^{-1} U^T X V (V^T V)^{-1}
\end{aligned}$$

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

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 3 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 4.

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 cálculo da média parcial 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.

$$\begin{aligned}(U^TX)V &= \left(\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} — & \mathbf{x}_{1\cdot} & — \\ — & \mathbf{x}_{2\cdot} & — \\ — & \mathbf{x}_{3\cdot} & — \\ — & \mathbf{x}_{4\cdot} & — \\ — & \mathbf{x}_{5\cdot} & — \\ — & \mathbf{x}_{6\cdot} & — \end{bmatrix} \right) \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{1\cdot} + \mathbf{x}_{2\cdot} + \mathbf{x}_{6\cdot} \\ \mathbf{x}_{3\cdot} + \mathbf{x}_{5\cdot} \\ \mathbf{x}_{4\cdot} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} x_{13} + x_{23} + x_{63} & x_{11} + x_{12} + x_{21} + x_{22} + x_{61} + x_{62} \\ x_{33} + x_{53} & x_{31} + x_{32} + x_{51} + x_{52} \\ x_{43} & x_{41} + x_{42} \end{bmatrix}\end{aligned}$$

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 4 ilustra o processo de minimização para o problema 3. Nesse algoritmo considere os índices $i = \{1, \ldots, n\}$, $j = \{1, \ldots, m\}$, $p = p' = \{1, \ldots, k\}$, e $q = q' = \{1, \ldots, l\}$, 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 que ignora zeros, 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 4 Algoritmo FNMTF

1: **function** FNMTF(X, t_{max}, k, l)
2: **Inicialize:** $U^{(0)}, V^{(0)} \leftarrow 0,1 \mid \sum_{p=1}^{k} \mathbf{u}_{p\cdot} = 1, \sum_{q=1}^{l} \mathbf{v}_{q\cdot} = 1$, $S^{(0)} \leftarrow \mathcal{U}(0,1)$ e $t \leftarrow 0$.
3: **while** (não convergiu) ou ($t \leq t_{max}$) **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 4 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: $\mathcal{O}\Big(t_{max}\big(nl(m+k) + kl(n+m) + nk + ml\big)\Big)$. Já é possível perceber

que a complexidade do algoritmo é menor das apresentadas nos algoritmos 1, 2 e 3. Isso se explica pois a atualização de U e V é feita de forma iterativa. Ainda, há multiplicações de matrizes para o cálculo dos centróides $\widetilde{U}$ e $\widetilde{V}$, e para atualização de S, que podem ser calculados 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. 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 organização 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 tem 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 simultaneamente ambas organizações na resolução do problema de minimização do erro de aproximação 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 unidimensional

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; 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 unidimensional, baseada na estratégia BVD;
- BinOvNMTF: uma estratégia de fatoração binária tripla de matrizes não-negativas com sobreposição unidimensional, 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 matriz U, 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 interseção de colunas na formação de cogrupos poderão ser adequadamente tratados pelas estratégias propostas.

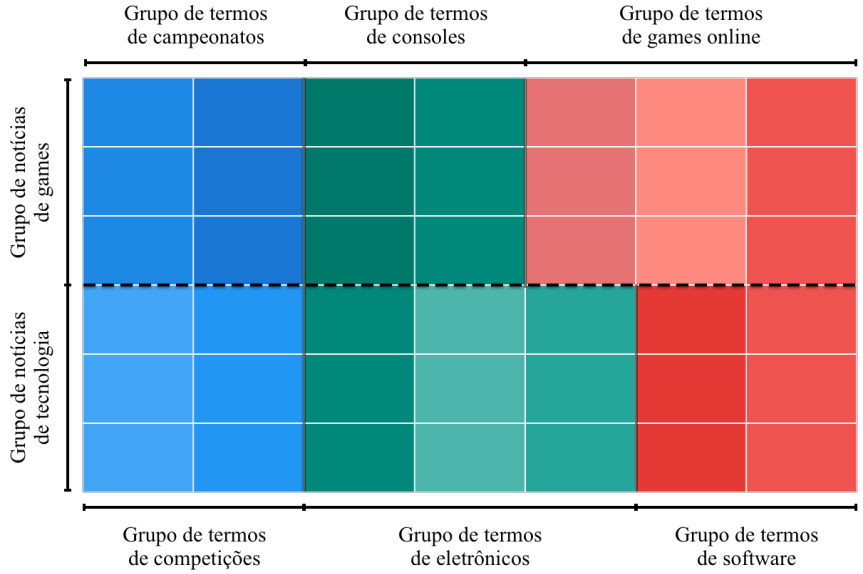
Formalmente, considere uma matriz de dados $X \in \mathbb{R}^{n \times m}$ contendo números reais positivos com n linhas e m colunas, formada por um conjunto de vetores de linhas $\mathcal{N} = \{\mathbf{x}_{1\cdot}, \ldots, \mathbf{x}_{n\cdot}\}$ e um conjunto de vetores de colunas $\mathcal{M} = \{\mathbf{x}_{\cdot 1}, \ldots, \mathbf{x}_{\cdot m}\}$, e as relações existentes entre cada linha x e cada coluna y são representadas por x_{ij} considerando os índices $i = \{1, \ldots, n\}$ e $j = \{1, \ldots, m\}$. O objetivo éEsta 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 $i = \{1, \ldots, n\}$ e $j = \{1, \ldots, m\}$, 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. Modificando o objetivo como foi apresentado na introdução ao capítulo 3, sendo então, 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 $p = \{1, \ldots, k\}$ e $q = \{1, \ldots, l\}$. Então, os subconjuntos $\{\mathcal{K}_1, \ldots, \mathcal{K}_p\}$ 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 4 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 unidimensional, ou seja, sobreposição de colunas ou de linhas, já que a sobreposição de linhas é justamente o problema transposto.

A figura 4 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”.

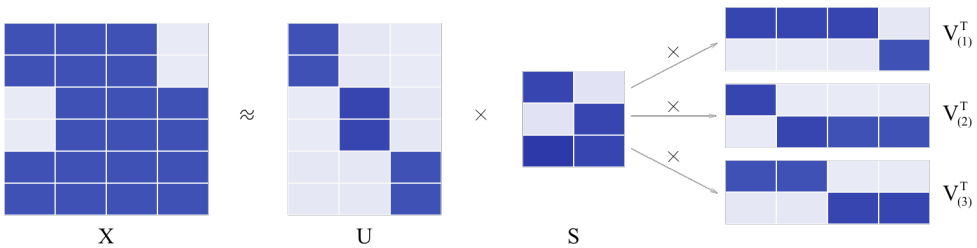
Figura 4 – Representação gráfica do problema de coagruamento com sobreposição unidimensional e contextualização do domínio de documentos (notícias)



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 representação a relação "documento por palavras" (linhas por colunas), cada coluna das k matrizes $UI_{(p)}S, \forall p = 1, \ldots, k$, captura a ideia de uma base para representação de grupos de palavras, descritos nas matrizes $V_{(p)}$; e cada linha em $\sum_{p=1}^{k} SV_{(p)}^{T}$, 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 5.

Figura 5 – 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 5 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 Unidimensional

Baseado no problema 1 (LONG; ZHANG; YU, 2005), o problema 4 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 4 (Problema de Fatoração Tripla de Matrizes Não-negativas Sobrepostas)**.**

$$\begin{aligned} \mathcal{F}_4(U,S,V_{(1)},\ldots,V_{(k)}) \quad = \quad & \min_{U,S,V_{(1)},\ldots,V_{(k)}} \quad \left\| X - U\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T \right\|_F^2 \\ & \textit{suj. a} \qquad\qquad U \geq 0, S \geq 0, \\ & \qquad\qquad\quad V_{(p)} \geq 0, \quad \forall p \end{aligned}$$

em que $U \in \mathbb{R}_+^{n\times k}$, $S \in \mathbb{R}_+^{k\times l}$, $V_{(p)} \in \mathbb{R}_+^{m\times l}$, $p = \{1,\ldots,k\}$ é o índice para o conjunto de matrizes $\{V_{(1)},\ldots,V_{(k)}\}$, $I_{(p)} \in \{0,1\}^{k\times k}$ é uma matriz identidade seletora, e $\|\cdot\|_F$ denota a norma de Frobenius. Na formulação, o papel das matrizes I_k 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_k.

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 4 é semelhante ao problema 2, é esperado que ele possa também ser resolvido por meio de regras de atualização multiplicativas. Assim, seguindo o exposto em (YOO; CHOI, 2010), a derivação das regras é aqui apresentada por meio de um abordagem baseada no cálculo do gradiente. Contudo, para o cálculo do gradiente de $\mathcal{F}_4$ a estratégia usada aqui é a apresentada em Yoo e Choi (2010), de forma que $\nabla\mathcal{F}_4 = [\nabla\mathcal{F}_4]^+ - [\nabla\mathcal{F}_4]^-$.

Expandindo $\mathcal{F}_4$, com base nas propriedade de traço de matrizes, para tornar o cálculo do gradiente mais simples, é possível obter:

$$\begin{aligned} \mathcal{F}_4 \quad &= \quad tr\big[(X - U\textstyle\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T)^T (U\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T)\big] \\ &= \quad tr(X^TX) - 2tr(X^TU\textstyle\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T) + tr(\sum_{p=1}^{k} V_{(p)} S^T I_{(p)} U^T U \sum_{p'=1}^{k} I_{(p')} S V_{(p')}^T) \end{aligned}$$

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

Considere as seguintes igualdades para o cálculo dos gradientes, sendo A, Q, B e C matrizes de quaisquers dimensões adequadas para a realização das multiplicações:

$$\begin{aligned} \nabla_Q tr(AQB) &= A^TB^T \\ \nabla_{Q^T} tr(AQB) &= BA \end{aligned} \tag{5}$$

$$\begin{aligned} \nabla_Q tr(AQBQ^TC) &= A^TC^TQB^T + CAQB \\ \nabla_{Q^T} tr(AQBQ^TC) &= BQ^TCA + B^TQ^TA^TC^T \end{aligned} \tag{6}$$

Para o cálculo de $\nabla_U \mathcal{F}_4$, considere as partes positiva e negativa desse gradiente, $[\nabla_U \mathcal{F}_4]^+$ e $[\nabla_U \mathcal{F}_4]^-$, respectivamente. Usando a igualdade da equação 5, com $A = X^T$, $B = \sum_{p=1}^{k} I_{(p)} S V_{(p)}^T$ e $Q = U$, é possível obter $[\nabla_U \mathcal{F}_4]^-$, como segue.

$$\begin{aligned}
[\nabla_U \mathcal{F}_4]^- &= -2\nabla_U \Big(tr\big[X^T U \textstyle\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T\big] \Big) \\
&= -2X \textstyle\sum_{p=1}^{k} V_{(p)} S^T I_{(p)}
\end{aligned}$$

Para o cálculo de $[\nabla_U \mathcal{F}_4]^+$, é utilizado a igualdade da equação 6, com $A = \sum_{p=1}^{k} V_{(p)} S^T I_{(p)}$, $B = I$, $Q = U^T$ e $C = \sum_{p'=1}^{k} I_{(p')} S V_{(p')}^T$. Então tem-se

$$\begin{aligned}
[\nabla_U \mathcal{F}_4]^+ &= \nabla_U \Big(tr(X^T X) + tr\big[\textstyle\sum_{p=1}^{k} V_{(p)} S^T I_{(p)} U^T U \sum_{p'=1}^{k} I_{(p')} S V_{(p')}^T \big] \Big) \\
&= U \textstyle\sum_{p'=1}^{k} I_{(p')} S V_{(p')}^T \sum_{p=1}^{k} V_{(p)} S^T I_{(p)} \\
&\quad + U \textstyle\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T \sum_{p'=1}^{k} V_{(p')} S^T I_{(p')} \\
&= 2U \textstyle\sum_{p=1}^{k} \sum_{p'=1}^{k} I_{(p)} S V_{(p)}^T V_{(p')} S^T I_{(p')}
\end{aligned}$$

De forma similar, para o cálculo de $\nabla_S \mathcal{F}_4$, considere as partes positiva e negativa, $[\nabla_S \mathcal{F}_4]^+$ e $[\nabla_S \mathcal{F}_4]^-$, respectivamente. Usando a igualdade da equação 5 para todas as partes da soma de $p = 1, \ldots, k$, com $A = X^T U I_{(p)}$, $Q = S$, e $B = V_{(p)}^T$, é possível obter $[\nabla_S \mathcal{F}_4]^-$, como segue:

$$\begin{aligned}
[\nabla_S \mathcal{F}_4]^- &= -2\nabla_S \Big(tr\big[X^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[X^T U I_{(k)} S V_{(k)}^T\big] \Big) \\
&= -2\Big(I_{(1)} U^T X V_{(1)} + \cdots + I_{(k)} U^T X V_{(k)} \Big) \\
&= -2 \textstyle\sum_{p=1}^{k} I_{(p)} U^T X V_{(p)}
\end{aligned}$$

Para o cálculo de $[\nabla_S \mathcal{F}_4]^+$, é utilizado a igualdade da equação 6 para todas as partes da soma de $p = p' = 1, \ldots, k$, com $A = V_{(p)}$, $Q = S^T$, $B = I_{(p)} U^T U I_{(p')}$ e $C = V_{(p')}^T$. Então tem-se

$$\begin{aligned}
[\nabla_S \mathcal{F}_4]^+ &= \nabla_S \Big(tr(X^T X) + tr\big[V_{(1)} S^T I_{(1)} U^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[V_{(1)} S^T I_{(1)} U^T U I_{(k)} S V_{(k)}^T\big] \\
&\quad + \cdots + tr\big[V_{(k)} S^T I_{(k)} U^T U I_{(1)} S V_{(1)}^T\big] + \cdots + tr\big[V_{(k)} S^T I_{(k)} U^T U I_{(k)} S V_{(k)}^T\big] \Big) \\
&= \big(I_{(1)} U^T U I_{(1)} S V_{(1)}^T V_{(1)} + I_{(1)} U^T U I_{(1)} S V_{(1)}^T V_{(1)} \big) \\
&\quad + \cdots + \big(I_{(1)} U^T U I_{(k)} S V_{(k)}^T V_{(1)} + I_{(k)} U^T U I_{(1)} S V_{(1)}^T V_{(k)} \big) \\
&\quad + \cdots + \big(I_{(k)} U^T U I_{(1)} S V_{(1)}^T V_{(k)} + I_{(1)} U^T U I_{(k)} S V_{(k)}^T V_{(1)} \big) \\
&\quad + \cdots + \big(I_{(k)} U^T U I_{(k)} S V_{(k)}^T V_{(k)} + I_{(k)} U^T U I_{(k)} S V_{(k)}^T V_{(k)} \big) \\
&= 2 \textstyle\sum_{p=1}^{k} \sum_{p'=1}^{k} I_{(p)} U^T U I_{(p')} S V_{(p')}^T V_{(p)}
\end{aligned}$$

Finalmente, para o cálculo de $\nabla_{V_{(p)}}\mathcal{F}_4$, considere as partes positiva e negativa, $[\nabla_{V_{(p)}}\mathcal{F}_4]^+$ e $[\nabla_{V_{(p)}}\mathcal{F}_4]^-$, respectivamente. Usando a igualdade da equação 5, com $A = X^TUI_{(p)}S$, $Q = V_{(p)}^T$, e $B = I$, $\forall p$, é possível obter $[\nabla_{V_{(p)}}\mathcal{F}_4]^-$, da seguinte forma:

$$\begin{aligned}[\nabla_{V_{(p)}}\mathcal{F}_4]^- &= -2\nabla_{V_{(p)}}\Big(tr\big[X^TUI_{(1)}SV_{(1)}^T\big] + \cdots + tr\big[X^TUI_{(k)}SV_{(k)}^T\big]\Big)\\ &= -2X^TUI_{(p)}S\end{aligned}$$

Fixando a derivação para $[\nabla_{V_{(p)}}\mathcal{F}_4]^+$ é 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 $A = I$, $Q = V_{(p)}$, $B = S^TI_{(p)}U^TUI_{(p')}S$ e $C = I$, $\forall p, p'$, de forma que

$$\begin{aligned}[\nabla_{V_{(p)}}\mathcal{F}_4]^+_{p=p'} &= \nabla_{V_{(p)}}\Big(tr\big[V_{(1)}S^TI_{(1)}U^TUI_{(1)}SV_{(1)}^T\big] + \cdots + tr\big[V_{(k)}S^TI_{(k)}U^TUI_{(k)}SV_{(k)}^T\big]\Big)\\ &= \nabla_{V_{(p)}}\Big(tr\big[V_{(p)}S^TI_{(p)}U^TUI_{(p)}SV_{(p)}^T\big]\Big)\\ &= 2V_{(p)}S^TI_{(p)}U^TUI_{(p)}S\end{aligned}$$

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

$$\begin{aligned}[\nabla_{V_{(p)}}\mathcal{F}_4]^+_{p\neq p'} &= \nabla_{V_{(p)}}\Big(tr\big[V_{(1)}S^TI_{(1)}U^TUI_{(2)}SV_{(2)}^T\big] + \cdots + tr\big[V_{(1)}S^TI_{(1)}U^TUI_{(k)}SV_{(k)}^T\big]\\ &\quad + \cdots + tr\big[V_{(k)}S^TI_{(k)}U^TUI_{(1)}SV_{(1)}^T\big] + \cdots + tr\big[V_{(k)}S^TI_{(k)}U^TUI_{(k-1)}SV_{(k-1)}^T\big]\Big)\\ &= \textstyle\sum_{p'\neq p}\Big[\nabla_{V_{(p)}}\Big(tr\big[V_{(p)}S^TI_{(p)}U^TUI_{(p')}SV_{(p')}^T\big]\Big)\\ &\quad + \nabla_{V_{(p)}}\Big(tr\big[V_{(p')}S^TI_{(p')}U^TUI_{(p)}SV_{(p)}^T\big]\Big)\Big]\\ &= \textstyle\sum_{p'\neq p} 2\big(V_{(p')}S^TI_{(p')}U^TUI_{(p)}S\big)\end{aligned}$$

Então, é possível calcular $[\nabla_{V_{(p)}}\mathcal{F}_4]^+$, $\forall p$, fazendo:

$$\begin{aligned}[\nabla_{V_{(p)}}\mathcal{F}_4]^+ &= [\nabla_{V_{(p)}}\mathcal{F}_4]^+_{p=p'} + [\nabla_{V_{(p)}}\mathcal{F}_4]^+_{p\neq p'}\\ &= 2\big(V_{(p)}S^TI_{(p)}U^TUI_{(p)}S + \textstyle\sum_{p'\neq p} V_{(p')}S^TI_{(p')}U^TUI_{(p)}S\big)\\ &= 2\big(\textstyle\sum_{p'=1}^{k} V_{(p')}S^TI_{(p')}U^TUI_{(p)}S\big)\end{aligned}$$

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.

$$\nabla_U\mathcal{F}_4 = 2\big(-X\textstyle\sum_{p=1}^{k} V_{(p)}S^TI_{(p)} + U\sum_{p=1}^{k}\sum_{p'=1}^{k} I_{(p)}SV_{(p)}^TV_{(p')}S^TI_{(p')}\big) \qquad (7)$$

$$\nabla_S \mathcal{F}_4 \;=\; 2\big(-\textstyle\sum_{p=1}^{k} I_{(p)}U^T X V_{(p)} + \sum_{p=1}^{k}\sum_{p'=1}^{k} I_{(p)}U^T U I_{(p')} S V_{(p')}^T V_{(p)}\big) \tag{8}$$

$$\nabla_{V_{(p)}} \mathcal{F}_4 \;=\; 2\big(-X^T U I_{(p)} S + \textstyle\sum_{p'=1}^{k} V_{(p')} S^T I_{(p')} U^T U I_{(p)} S\big) \tag{9}$$

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 5, o qual implementa o processo de minimização para o problema 4. 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 que ignora zeros, e $\odot$ é o produto de Hadamard.

Algoritmo 5 Algoritmo baseado em atualização multiplicativa para solução do OvNMTF

function OvNMTF(X, t_{max})
  Inicialize: $U^{(0)} \geq \mathcal{U}(0,1), S^{(0)} \geq \mathcal{U}(0,1), V_{(p)}^{(0)} \geq \mathcal{U}(0,1), \forall p$ e $t \leftarrow 0$.
  while (não convergiu) ou $(t \leq t_{max})$ **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 $p \leftarrow 1, k$ **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 que ignora zeros ($\mathcal{U}(0,1) \in]0,1]$). Como condições para assumir a convergência, assim como os algoritmos apresentados no capítulo 3, considera-se a diferença do erro de aproximação em 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 t-ésima iteração seja igual ao número máximo de iterações (t_{max}).

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:

$$\mathcal{K}_p = \mathcal{K}_p + \{x_{i\cdot}\} \mid p = \arg\max_{p'} \mathbf{u}_{i\cdot} \ \forall i = \{1, \ldots, n\}, \forall p, p' = \{1, \ldots, k\}$$

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

$$\mathcal{L}_{pq} = \mathcal{L}_{pq} + \{x_{\cdot j}\} \mid q = \arg\max_{q'} \mathbf{v}_{(p)_{j\cdot}} \ \forall j = \{1, \ldots, m\}, \forall q, q' = \{1, \ldots, l\}, \forall p = \{1, \ldots, k\}$$

4.2 Fatoração Binária Tripla de Matrizes Não-negativas com Sobreposição Unilateral

A segunda estratégia proposta nesta dissertação, segue os pressupostos estabelecidos por Wang et al. (2011) no problema 3. O problema 5, então denominado Fatoração Binária Tripla de Matrizes Não-negativas com Sobreposição Unilateral (Unilateral 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 5 (Problema de Fatoração Binária Tripla de Matrizes Não-negativas Sobrepostas)**.**

$$\begin{aligned} \mathcal{F}_5(U, S, V_{(1)}, \ldots, V_{(k)}) \quad = \quad & \min_{U,S,V_{(1)},\ldots,V_{(k)}} \quad \left\| X - U \textstyle\sum_{p=1}^{k} I_{(p)} S V_{(p)}^T \right\|_F^2 \\ & \textit{suj. a} \qquad\qquad U \in \Psi^{n \times k}, \\ & \qquad\qquad\quad V_{(p)} \in \Psi^{m \times l}, \quad \forall p \end{aligned}$$

em que $\Psi = \{0, 1\}$, $\sum_{p=1}^{k} \mathbf{u}_{p\cdot} = 1$ e $\sum_{q=1}^{l} \mathbf{v}_{(p)_{q\cdot}} = 1, \forall p$, $p = \{1, \ldots, k\}$ e $q = \{1, \ldots, l\}$ 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 $\sum_{p=1}^{k} \mathbf{u}_{p\cdot} = 1$ e $\sum_{q=1}^{l} \mathbf{v}_{(p)_{q\cdot}} = 1, \forall p$ garantem que uma linha e uma coluna de um grupo, respectivamente, têm que pertencer à algum grupo.

Em termos de compactação, a interpretação é semelhante à interpretação feita para o problema 4, 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 4, também pode ser feita para propor uma solução para o problema 5: solucionar o problema para S para obter subproblemas que podem ser solucionados através de uma estratégia iterativa. Então, recapitulando o gradiente $[\nabla_S \mathcal{F}_4]^+$, já calculado para solução do problema 4, será o mesmo para $\mathcal{F}_5$, porém passível de simplificação, como $(U^T U)$, para o problema 5 que tem restrições binárias em U, é uma matriz que contém zeros em todos elementos, com exceção da diagonal principal:

$$\begin{aligned}
[\nabla_S \mathcal{F}_5]^+ &= \sum_{p=1}^{k}\sum_{p'=1}^{k} I_{(p)} U^T U I_{(p')} S V_{(p')}^T V_{(p)} \\
&= \sum_{p=1}^{k}\sum_{p'=1}^{k} \left(I_{(p)} \begin{bmatrix} (U^T U)_{11} & & 0 \\ & \ddots & \\ 0 & & (U^T U)_{kk} \end{bmatrix} I_{(p')} \right) S V_{(p')}^T V_{(p)}
\end{aligned}$$

Note que se $p \neq p'$, $I_{(p)} U^T U I_{(p)}$ e toda expressão, será uma matriz de zeros, por exemplo, se $p = 1$ e $p' = 2$:

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

$$\therefore \quad [\nabla_S \mathcal{F}_5]^+ = \sum_{p=1}^{k} I_{(p)} U^T U I_{(p)} S V_{(p)}^T V_{(p)}$$

Assim, é possível escrever o gradiente $\nabla_S \mathcal{F}_5$, a fim de encontrar uma expressão para atualização de S.

$$\begin{aligned}
[\nabla_S \mathcal{F}_5]^+ &= -2\sum_{p=1}^{k} I_{(p)} U^T X V_{(p)} + 2\sum_{p=1}^{k} I_{(p)} U^T U I_{(p)} S V_{(p)}^T V_{(p)} = 0 \\
&\implies \sum_{p=1}^{k} I_{(p)} U^T U I_{(p)} S V_{(p)}^T V_{(p)} = \sum_{p=1}^{k} I_{(p)} U^T X V_{(p)}
\end{aligned}$$

Como os termos $I_{(p)} U^T X V_{(p)}$ e $I_{(p)} U^T U I_{(p)} S V_{(p)}^T V_{(p)}$, $\forall p$, por serem multiplicados por $I_{(p)}$ pela esquerda, correspondem a cada linha da matriz que será a atualização de

S, aqui denotado 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).

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

$$\therefore\ S = \sum_{p=1}^{k} I_{(p)}(U^TU)^{-1}U^TXV_{(p)}V_{(p)}^TV_{(p)}$$

É importante ressaltar que a mesma interpretação realizada para atualização de S no algoritmo 4, também é possível ser transferida para esse contexto.

Assim, é possível transformar o problema 5 em subproblemas para atualização de U e $V_{(1)}, \ldots, V_{(k)}$. Usando a mesma estratégia iterativa adotada no algoritmo 4: 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.

O implementação desse processo é apresentado no algoritmo 6). 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 que ignora zeros, e $\|\cdot\|^2$ é a norma frobenius para vetores.

Note que nesse tipo de fatoração, semelhante à apresentada no problema 3, 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 5.

4.3 Considerações Finais

A motivação para a apresentação das novas estratégias (OvNMTF e BinOvNMTF), era a possibilidade de superar algumas das dificuldades apresentas nas estratégias da literatura (ONMTF e FNMTF).

Do ponto de vista de quantização do espaço dos dados, a quantidade de informação armazenada nas novas estratégias tem o potencial de superar as estratégias da literatura no que diz respeito ao tamanho da matriz S. Isso ocorre porque nas estratégias propostas,

Algoritmo 6 Algoritmo iterativo para solução do BinOvNMTF

function BinOvNMTF(X, t_{max}, k, l)
  Inicialize: $U^{(0)} \geq 0,1 \mid \sum_{p=1}^{k} \mathbf{u}_{p\cdot} = 1$, $V_{(p)}^{(0)} \geq 0,1 \mid \sum_{q=1}^{l} \mathbf{v}_{(p)_{q\cdot}} = 1, \forall p$, $S^{(0)} \leftarrow \mathcal{U}(0,1)$ e $t \leftarrow 0$.
  while (não convergiu) ou ($t \leq t_{max}$) **do**

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

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

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

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

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

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

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 interseção de colunas entre as diferentes bases dos grupos de linhas. Entretanto, as estratégias 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.

Do ponto de vista de geração de informação sobre os dados, as estratégias propostas são capazes de fornece 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 organização 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 estratégias propostas seguem o mesmo principio seguido nas estratégias da literatura, qual seja, considerar a informação de similaridade de dados e de atributos simultaneamente para resolver o problema de minimização do erro de aproximação da matriz original, e consequentemente apresentar protótipos que expliquem os cogrupos. Porém, as estratégias 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.

Essa considerações, assim como aquelas delineadas nos capítulos 2 e 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 textos. Também, como forma de melhorar a compreensão sobre o desempenho dos algoritmos de coagrupamento, dois algoritmos clássicos de agrupamento foram aplicados sobre as mesmas bases de dados: *k-means* e *fuzzy k-means*.

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, de qualidade de *particionamento*, a qual é avaliada em termos de medidas de qualidade de grupos de linhas e qualidade de grupos de colunas, e de qualidade de *coagrupamento*, a qual é avaliada via medida xxxxx. Então, a qualidade de particionamento e de coagrupamento é avalidada nos resultados obtidos com as bases de dados textuais reais originais e simplificadas. 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 textos.

5.1 Experimentos com Base 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 intersecção de linhas ou colunas, que representam os casos que tais algoritmos não são capazes de tratar e que são, portanto, o alvo dos algoritmos propostos neste trabalho (*OvNMTF* e *BinOvNMTF*). Na figura 6 são apresentas as visualizações gráficas de cada uma das estruturas de cogrupos em estudo.

Figura 6 – Dados sintéticos gerados a partir das diferentes estruturas de cogrupos. (a) Um único cogrupo. (b) Cogrupos com linhas e colunas sem intersecção. (c) Cogrupos com estrutura em xadrez. (d) Cogrupos sem intersecção nas linhas e com intersecção nas colunas. (e) Cogrupos com intersecção nas linhas e sem intersecção nas colunas.

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

Fonte: Lucas Fernandes Brunialti, 2016

Para compreender a representação gráfica apresentada na figura 6 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.

Todas as bases de dados da figura 6 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 distribuição uniforme

$unif(0, 0, 1, 0)$, no intervalo $[0, 0, 1, 0]$[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;
- 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 $unif(0, 0, 10, 0)$, sendo que $unif$ é uma função que escolhe um valor aleatório dentro do intervalo $[0, 0, 10, 0]$, seguindo uma distribuição uniforme;
- 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

$$x_{ij} = c + unif(0, 0, 10, 0)$$

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

Ainda sobre a interpretação da figura 6, alguns detalhes precisam ser observados. A depender do objetivo da análise de coagruamento, na figura 6a, 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 6a 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].

[1] Este processo evita que ocorram divisões por 0 nos algoritmos. Que processo? O que exatamente evita? Ser positivo, ser uniforme? Sugiro que essa observação vá para um rodapé, mas precisa ser mais específica.

- 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 princial interesse e oito cogrupos não destacados na figura;
- (e): três cogrupos de princial interesse e oito cogrupos não destacados na figura.

Para cada uma das bases de dados sintéticas foram executados os seguintes algoritmos: *k-means*[2], *fuzzy k-means*[3], *ONMTF*, *FNMTF*, *OvNMTF* e *BinOvNMTF*[4]. Os resultados foram analisados em termos de qualidade de reconstrução, qualidade de particionamento e qualidade de coagraumento.

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. Para os algoritmos baseados em fatoração de matrizes, 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. Para o caso dos algoritmos *k-means* e *fuzzy-k-means*, a resconstrução foi realizada assumindo que os centróides obtidos pelos algoritmos representam os dados originais; assim a substituição de um dado original pelo centróide do grupo ao qual ele pertence, seguindo as regras dos algoritmos *k-means* e *fuzzy k-means*, constitui a reconstrução aplicada nesses casos.

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 interseção de linhas ou colunas. A capacidade de reconstrução .... vamos ver se colocamos alguma coisa a mais para justificar esse análise.

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

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

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

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; * - sem informação sobre interseção; + - preserva informação de interseção

	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+	+
BinOvNMTF	ok	ok	ok	ok+	

Fonte: Lucas Fernandes Brunialti, 2016

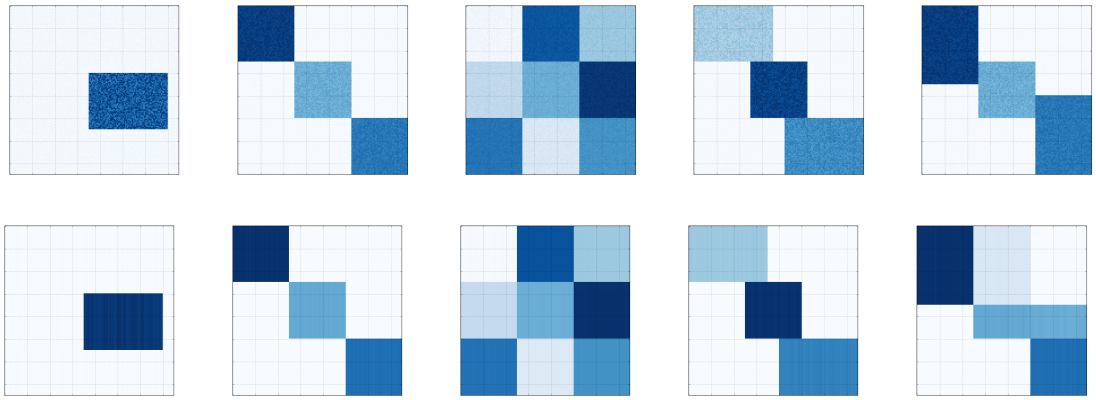
5.1.2 Reconstrução a partir dos resultados do algoritmo *k-means*

Para execução dos experimentos com o *k-means* os seguintes parâmetros foram estabelecidos: número máximo de iterações (300) ou a diferença do erro de minimização em duas iterações consecutivas ($\leq 1,0 * 10^{-4}$), 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 Figura 8. 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 7 – 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 da similaridade do centróide em relação aos dados de cada agrupamento de linhas.

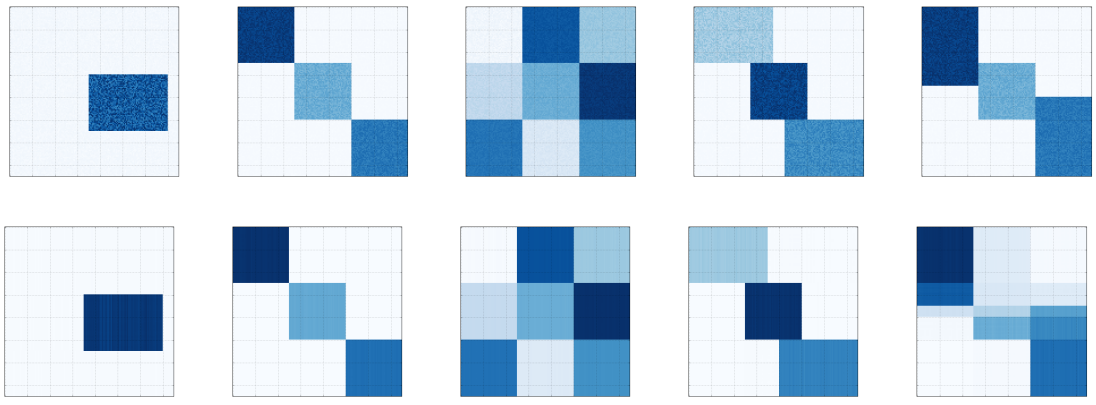
O modelo não permite a descoberta de grupos com interseção de linhas (um dado não pode pertencer a mais de um grupo), e portanto não permite uma boa representação para a reconstrução no caso da base de dados (e). Esse resultado já era esperado devido à natureza da solução apresentada pelo *k-means*. Tem um erro aqui, pois o centróide deveria ser único para cada um dos 3 grupos, não há como termos esse quarto quadrado em azul claro.

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.,). Também, 10 execuções foram realizadas, com iniciação aleatória de pesos e o melhor modelo obtido foi usado para avaliação da qualidade de reconstrução obtida.

Da mesma forma que o experimento anterior, as reconstruções apresentadas na figura 8 foram obtidas por meio da coloração de linhas de acordo com os centróides resultantes da execução do algoritmo, e as matrizes originais foram repetidas para facilitar a análise visual. Contudo, desde que a pertinência de um dado a mais de um grupo é possível nesse algoritmo, escolheu-se o grupo ao qual o dado tem maior pertinência para usar como base para a reconstrução.

Figura 8 – 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. Embora o *fuzzy c-means* lide, em sua concepção, com sobreposição de grupos, neste caso representada pelas pertinências fuzzy dos dados a vários grupos, o procedimento de escolha da maior pertinência anula essa capacidade na representação da reconstrução; muito embora .... é preciso fazer alguma ressalva ou análise diferente aqui.

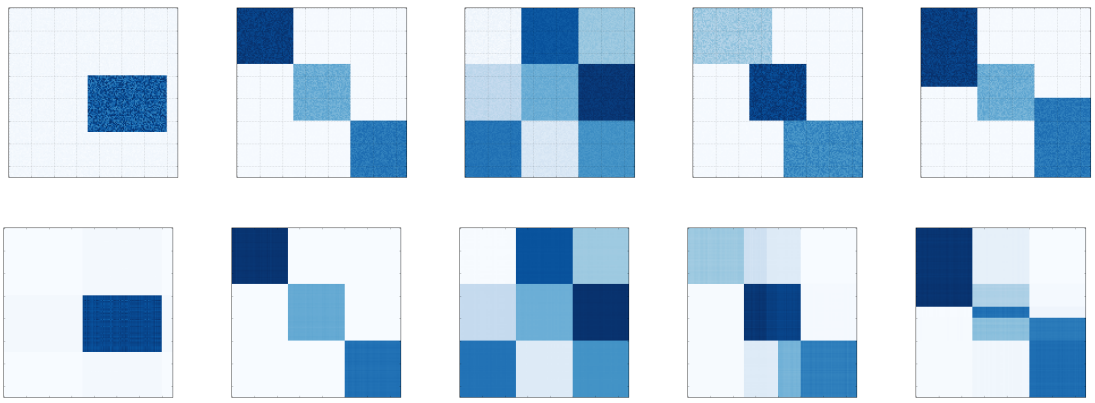
¿¿¿¿¿¿¿ 2792448c20174b8a33ff41ecb8de077f0d58b4c5

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 minimização em duas iterações consecutivas ($\leq 1,0 * 10^{-4}$), 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 (e), e $k = l = 3$ para (c). Novamente, as escolhas são baseadas no conhecimento *apriori* que se tem sobre a quantidade de cogrupos, e quais cogrupos, deseja-se obter. Para a execução do algoritmo *ONMTF* é necessária a normalização dos dados, então, todas as matrizes de dados sintéticas foram normalizadas para que a norma das linhas fosse igual à 1.

A partir da realização de 10 execuções do algoritmo, foi escolhido o modelo que alcançou a menor taxa de erro e a partir do resultado obtido a reconstrução foi avaliada. A qualidade das resconstruções pode ser visualmente observada na figura 9. A resconstrução foi obtida a partir da multiplicação das matrizes fatoradas, ou seja, USV^T, conforme explicado no capítulo 3.

Figura 9 – 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 (interseção nas colunas). O mesmo efeito ocorre com o caso (e), no qual há interseção de linhas. A falha da reconstrução é percebida na coloração diferenciada, formando regiões com sombras, nas colunas, ou linhas, envolvidas nas interseções.

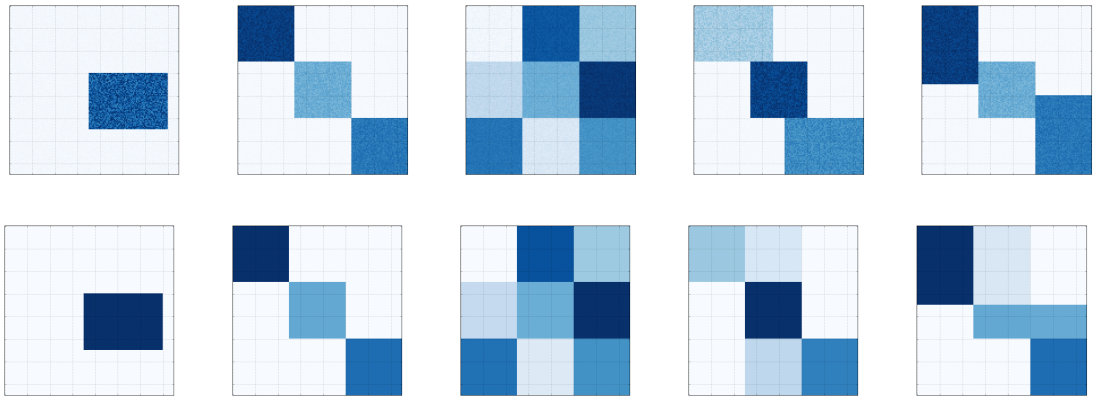
A coloração diferenciada daquela esperada (seguindo a matriz original) é resultante do estado da matriz V, que contém as relações de associação de linhas e/ou colunas aos cogrupos. Para o caso (d), por exemplo, é possível deduzir, a partir da análise visual, que há valores de magnitude diferentes estabelecendo a associação das colunas X x X, Y x Y e Z x Z, ao grupo de linhas X x X. Isso é possível pois o algoritmo não força a associação única e exclusiva entre uma linha e um agrupamento de linhas ou uma coluna e um agrupamento de colunas.Em uma análise qualitativa, poder-se-ia então dizer que o algoritmo está indicando a formação de quatro grupos de colunas em relação ao mesmo grupo de linhas X x X.

5.1.5 Reconstrução a partir dos resulatdos 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 ($\leq 1,0 * 10^{-4}$), 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). Para a execução do algoritmo ONMTF é necessária a normalização dos dados, então, todas as matrizes de dados sintéticas foram normalizadas para que a norma das linhas fosse igual à 1.

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

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 *FNMTF*.



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

Fonte: Lucas Fernandes Brunialti, 2016

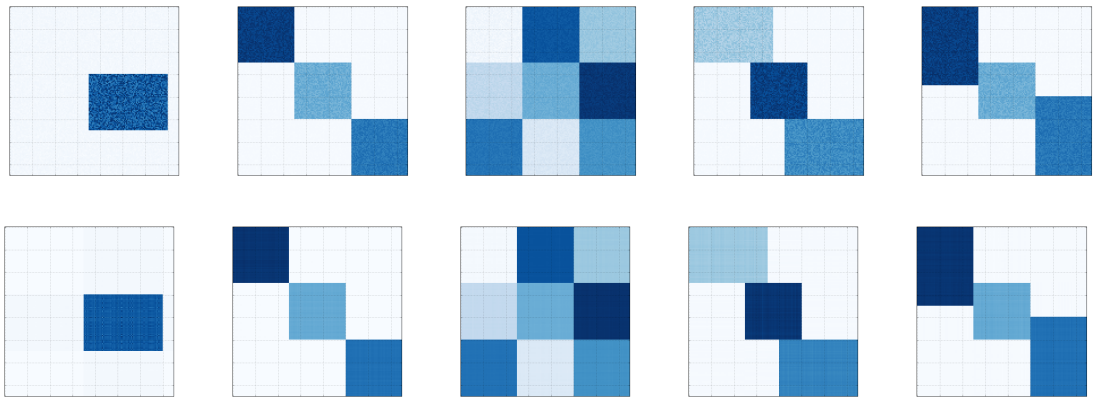
Este caso é semelhante ao anterior (algoritmo *ONMTF*). O algoritmo permitiu a reconstrução com êxito dos casos (a), (b) e (c), falhando na reconstrução dos casos (d) e (e), nos quais há intersecção de colunas ou linhas nos cogrupos de interesse. No entanto, nesse caso o algoritmo restringe a associação de linhas (ou colunas), a apenas um agrupamento de linhas (ou de colunas), e por isso a informação referente a intersecção 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.

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 minimização ($\leq 1,0 * 10^{-4}$), 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). Para a execução do algoritmo *OvNMTF* é necessária a normalização dos dados, então, todas as matrizes de dados sintéticas foram normalizadas para que a norma das linhas fosse igual à 1.

A partir da realização de 10 execuções do algoritmo, foi escolhido o modelo que alcançou a menor taxa de erro e a partir do resultado obtido a reconstrução foi avaliada. A qualidade das reconstruções pode ser visualmente observada na figura 11. 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 11 – 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 dos casos (a), (b), (c) e (d), com valores suavemente diferentes (isso pode ser notado pela cor das matrizes reconstruídas), devido à normalização necessária para ..... Note que o algoritmo *OvNMTF* é capaz de lidar com cogrupos com intersecção de colunas, e portanto é capaz de resolver o caso (d).

Como o algoritmo não é capaz de lidar com cogrupos com intersecção nas linhas, ele não é capaz de realizar a reconstrução do caso (e). No entanto, assim como o algoritmo *ONMTF*, o *OvNMTF* possui a informação sobre o agrupamento de linhas e permite a associação de linhas a vários agrupamentos. Assim, ele é capaz de preservar a informação referente às interseções nas linhas, porém de maneira fragmentada em vários grupos.

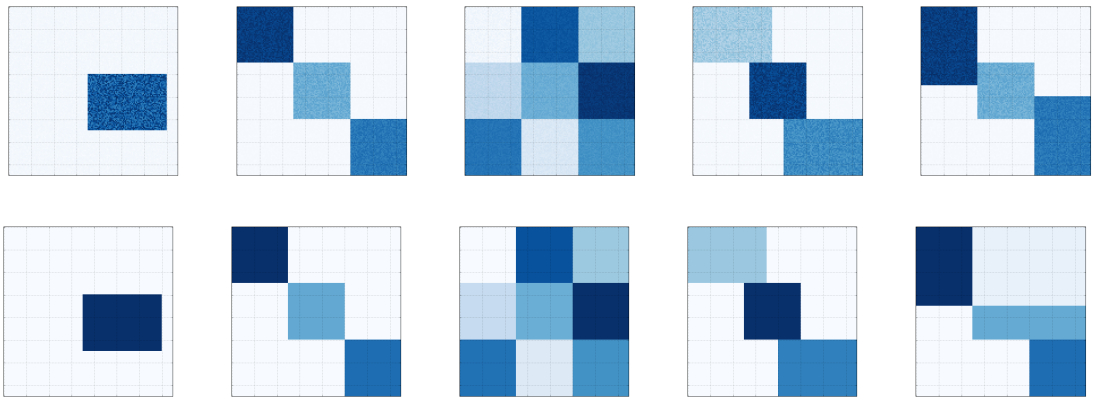
5.1.7 Reconstrução a partir dos resultados do algoritmo *BinOvNMTF*

Para execução dos experimentos com o *BinOvNMTF* a seguinte parametrização foi estabelecida: número máximo de iterações (300) ou a diferença do erro de minimização

($\leq 1,0 * 10^{-4}$), 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). Para a execução do algoritmo *BinOvNMTF* é necessária a normalização dos dados, então, todas as matrizes de dados sintéticas foram normalizadas para que a norma das linhas fosse igual à 1.

A partir da realização de 10 execuções do algoritmo, foi escolhido o modelo que alcançou a menor taxa de erro e a partir do resultado obtido a reconstrução foi avaliada. A qualidade das reconstruções pode ser visualmente observada na figura 12. A reconstrução foi obtida a partir da multiplicação das matrizes fatoradas, ou seja, $F \sum_{p=1}^{k} I_{(p)} S G_{(p)}^{T}$, conforme explicado no capítulo 4.

Figura 12 – 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

Semelhante ao experimento anterior, 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 intersecção de colunas, eke é capaz de resolver o caso (d). No entanto, o algoritmo não é capaz de lidar com cogrupos com intersecção nas linhas, ele não é capaz de realizar a reconstrução do caso (e). E nesse caso, também não preserva qualquer informação sobre a intersecção, já que possui a restrição de associação binária (cada linha/coluna associadas a um único agrupamento de linhas/colunas).

5.1.8 Análise de particionamento

A análise de particionamento avaliada nesta seção diz respeito ao entendimento do quanto aos agrupamentos de linhas ou agrupamento de colunas descoberto pelos algoritmos estariam adequados em relação a um modelo conhecido *apriori*. Os mesmos modelos que foram analisados quanto à qualidade de reconstrução, foram usados para a análise apresentada aqui. Como as bases de dados foram sintetizadas, foi possível rotulá-las de forma que se sabe a qual cogrupo uma determinada linha ou uma determinada coluna pertence, permitindo uma análise de particionamento via um índice de avaliação externa de agrupamentos.

Apesar da escolha por manter a análise sobre os mesmos modelos usados na análise de reconstrução, para o caso da análise de agrupamento, foi também verificada a qualidade obtida quando da execução dos algoritmos a partir de uma entrada de dados em ordem aleatória de linhas ou colunas. Os resultados obtidos mostram que o desempenho dos algoritmos é independente da organização das linhas e colunas da matriz de entrada.

O índice usado para a avaliação de particionamento foi o *Rand Index* (RAND, 1971). Esse índice gera valores entre 0 e 1, com 0 indicando que o particionamento de dados gerado pelo modelo não concorda em nenhum par de pontos com o particionamento real e 1 indicando que o particionamento gerado pelo modelo é exatamente igual ao particionamento real. O *Rand Index* é definido como

$$(A + D)/(A + B + C + D)$$

em que

- A: quantidade de pares de pontos (pontos distintos) para os quais os pontos pertencem a uma mesma partição no particionamento gerado e a uma mesma partição no particionamento real;
- B: quantidade de pares de pontos (pontos distintos) para os quais os pontos pertencem a uma mesma partição no particionamento gerado e a partições diferentes no particionamento real;
- C: quantidade de pares de pontos (pontos distintos) para os quais os pontos pertencem à partições diferentes no particionamento gerado e a uma mesma partição no particionamento real;

- A: quantidade de pares de pontos para os quais os pontos pertencem à partições diferentes no particionamento gerado e a partições diferentes no particionamento real.

No caso da avaliação da qualidade de agrupamentos para bases em que há interseção de linhas ou de colunas, o *Rand Index* foi modificado de forma a considerar que um ponto pode pertencer a mais de um agrupamento. Assim, na avaliação de pares de pontos nos quais um deles, ou os dois, pertenciam a mais de um agrupamento, duas possibilidades de análise foram consideradas:

- caso médio: todas as situações possíveis foram consideradas, podendo este par de pontos entrar na contagem das somas de A e B, C e D, ou A e D;
- pior caso: das situações possíveis, a menos favorável foi considerada, ou seja, o part de pontos entrou na soma de B ou C, em detrimento de A ou D, e na soma de A em detrimento de D.

A avaliação externa foi realizada para o particionamento de linhas e para o particionamento de colunas e os resultados estão listados na tabela 2. Observando os valores destacados nos valores de *Rand Index* para avaliação de agrupamento de linhas, é possível notar que nenhum dos algoritmos foi capaz de particionar perfeitamente (com valor 1.0) as linhas para a base de dados (e), que contém intersecção de linhas. Claramente, isso ocorre pois nenhum dos algoritmos é capaz de encontrar cogrupos com intersecção de linhas.

Em relação aos valores de *Rand index* para avaliação de agrupamentos de colunas, não há resultados para os algoritmos *k-means* e *fuzzy k-means*, pois os mesmos não resolvem o problema de particionamento de colunas. Os valores destacados para essa avaliação mostram que os algoritmos propostos neste trabalho (*OvNMTF* e *BinOvNMTF*) são capazes de particionar as colunas corretamente tanto para o caso (d) quanto para o caso (e), embora continue havendo a perda de qualidade para o caso (e) quando se observa o resultado dos algoritmos em relação ao agrupamento de linhas (assim como ocorre para os algoritmos da literatura).

Tabela 2 – Avaliação da qualidade de agrupamento de linhas (l) e colunas (c) sob a medida de avaliação *Rand Index*

		base (a)	**base (b)**	**base (c)**	**base (d)**	**base (e)**
k-means	l	1,0	1,0	1,0	1,0	**0,78**
	c	–	–	–	–	–
fuzzy-k-means	l	1,0	1,0	1,0	1,0	**0,78**
	c	–	–	–	–	–
ONMTF	l	1,0	1,0	1,0	1,0	**0,78**
	c	1,0	1,0	1,0	**0,78**	1,0
FNMTF	l	1,0	1,0	1,0	1,0	**0,53**
	c	1,0	1,0	1,0	**0,78**	1,0
OvNMTF	l	1,0	1,0	1,0	1,0	**0,78**
	c	1,0	1,0	1,0	**1,0**	1,0
BinOvNMTF	l	1,0	1,0	1,0	1,0	**0,53**
	c	1,0	1,0	1,0	**1,0**	1,0

Fonte: Lucas Fernandes Brunialti, 2016

Os resultados obtidos na avaliações de reconstrução corroboram com os resultados obtidos na validação de agrupamento. Então tem que colocar algo mais direto ..... depois de rever as medidas, é claro.

5.1.9 Análise de coagrupamento

Uma forma simples de resolver esse problema seria nós fazermos uma adaptação do índice de Rand. A mesma lógica usada para pertinência a grupos pode ser usada para a pertinência a co grupos, não? Seria parecido com o concensus score ????

avaliar de acordo com valores coerentes talvez fosse possível tb.

fazer a avaliação interna seria avaliada como Santamaria miguel e theron propuseam

5.2 Experimentos com base de dados reais

Quatro 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, duas (*20Newsgroups* e *NIPS14-17*) são bases de referência, usadas pela comunidade científica para experimentação de algoritmos de mineração de textos. 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 baseada 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 quatro bases de dados reais são compostas por textos referentes ao domínio de postagens em grupos discussão, 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.

1. ***20Newsgroups***[5] Trata-se de uma coleção de documentos do tipo texto que se tornou referência para avaliação de algoritmos de aprendizado de máquina aplicados a tarefas de mineração de textos (classificação e agrupamento) e outros tipos de análise como redução de dimensionalidade e recuperação de informação. Os documentos compreendem postagens de usuários anônimos do *Usenet newsgroup* (um repositório para grupos de discussões de um sistema distribuído de comunicação chamado *Usenet*). A coleção de documentos está organizada de forma particionada em 20 diferentes grupos temáticos, como computação, ciências, política e etc., com uma distribuição uniforme de documentos nos grupos.
2. ***NIPS14-17***[6] Esta base de dados contém uma coleção de trabalhos acadêmicos publicados no congresso *NIPS* (*Neural Information Processing Systems*) no período de 2003 a 2003, dos volumes 14 a 17. A construção da base de dados *NIPS14-17* foi realizada por *Sam Roweis*[7], 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

[5] http://qwone.com/~jason/20Newsgroups/
[6] http://robotics.stanford.edu/~gal/data.html
[7] http://www.cs.nyu.edu/~roweis/data.html

de aprendizado, neurociência, algoritmos e arquiteturas e etc), e estão distribuídos de forma desbalanceada entre os grupos caracterizados por cada um desses tópicos.

3. ***IG*** Esta base de dados foi criada como uma contribuição deste trabalho, e consiste em uma coleção de notícias extraídas do portal iG[8]. 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.
4. ***IG toy (ou reduzido)*** Esta base de dados é um subconjunto do conjunto de dados *IG*, composto por 100 notícias de três canais (esporte, arena e jovem). Esta base foi criada para possibilitar a análise mais detalhada dos resultados obtidos com os algoritmos aplicados sobre ela.

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

Base de dados	# Palavras	# Documentos	# Grupos reais	Distribuição por grupo
20Newsgroup	12.998	18.221	20	
NIPS14-17	17.583	420	9	
IG	19.563	4.593	13	
IG toy	6.764	300	3	

Fonte: Lucas Fernandes Brunialti, 2016

A fim de estruturar a informação nas bases de dados para construção dos modelos, foi necessária uma fase de pré-processamento, descrita na subseção 5.2.2.

5.2.2 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:

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

- 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. No caso deste trabalho foi usada uma expressão regular que separa caracteres não contíguos. Colocar a expressão regular aqui ...
- 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: ..... colocar as palavras. Ainda, como parte da filtragem, palavras cuja frequência de ocorrência nos documentos é muito pequena ou muito grande, podem ser acrescentadas à lista. Neste trabalho todas as palavras com ocorrência menor que dois em um documento 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 $D = \{d_1, \ldots, d_i, \ldots d_n\}$, e um conjunto de m termos $\{t_1, \ldots, t_j, \ldots t_m\}$, e para cada par $\langle t_j, d_i \rangle$, estabelece-se uma relação que expressa a maneira como um termo será usado na representação de um documento.

A relação entre documentos e termos pode ser representada de diversas formas e, neste trabalho, foram utilizadas quatro formas: $tf(t_j, d_i)$, $tf\_idf(t_j, d_i)$, $tf_{norm}(t_j, d_i)$ e $tf\_idf_{norm}(t_j, d_i)$.

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 é $\mathbf{v}_{d_i} = \left[tf(t_1, d_i), \ldots, tf(t_j, d_i), \ldots tf(t_m, d_i)\right]$. 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* - *tf_idf*) 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.

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

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 $\mathbf{v}_{d_i}$ tenham comprimento iguais, ou seja, $\|\mathbf{v}_{d_i}\| = 1$. A partir dessa normalização aplicada aos vetores gerados com as relações $tf(t_j, d_i)$ e $tf\_idf(t_j, d_i)$, obtem-se respectivamente, as relações $tf_{norm}(t_j, d_i)$ e $tf\_idf_{norm}(t_j, d_i)$.

5.2.3 Experimentos quantitativos

Em todas as bases foram usados os rótulos verdadeiros para avaliar a qualidade dos clusters, usadas apenas após a fase de criação dos modelos, ou seja, não serviram de entrada para os algoritmos.

5.2.3.1 Setup dos algoritmos

Foi utilizado o algoritmo do CCR para otimizar as multiplicações de matrizes.

5.2.3.2 Setup dos experimentos

5.2.3.3 Base de dados ***NIPS***

5.2.3.4 Base de dados ***20Newsgroup***

5.2.3.5 Base de dados ***IG***

$$k = 13$$

$$l = 7, 10, 13, 16, 19$$

Tabela 4 – Melhores resultados dos experimentos.

Algoritmo	Representação	l	*Rand Index*
k-means	TF	-	0.3082
k-means	TF norm	-	0.3350
k-means	TF-IDF	-	0.2750
k-means	TF-IDF norm	-	0.2262
fuzzy k-means	TF	-	0.2681
fuzzy k-means	TF norm	-	0.1757
fuzzy k-means	TF-IDF	-	0.2347
fuzzy k-means	TF-IDF norm	-	0.3832
ONMTF	TF	19	0.1883
ONMTF	TF norm	13	0.2023
ONMTF	TF-IDF	13	0.1814
ONMTF	TF-IDF norm	7	0.2037
FNMTF	TF	13	0.2367
FNMTF	TF norm	19	0.2678
FNMTF	TF-IDF	7	0.2410
FNMTF	TF-IDF norm	16	0.2480
OvNMTF	TF	-	0.0000
OvNMTF	TF norm	-	0.0000
OvNMTF	TF-IDF	-	0.0000
OvNMTF	TF-IDF norm	-	0.0000
BinOvNMTF	TF	16	0.4415
BinOvNMTF	TF norm	7	0.4081
BinOvNMTF	TF-IDF	7	0.3681
BinOvNMTF	TF-IDF norm	16	0.3883

5.2.3.6 Base de dados ***IG toy***

$$k = 3$$

$$l = 2, 3, 4, 5, 6$$

Tabela 5 – Melhores resultados dos experimentos.

Algoritmo	Representação	l	*Rand Index*
k-means	TF	-	0.0002
k-means	TF norm	-	0.6496
k-means	TF-IDF	-	0.0004
k-means	TF-IDF norm	-	0.3381
fuzzy k-means	TF	-	0.0044
fuzzy k-means	TF norm	-	0.0043
fuzzy k-means	TF-IDF	-	0.0413
fuzzy k-means	TF-IDF norm	-	0.0778
ONMTF	TF	6	0.3910
ONMTF	TF norm	5	0.7485
ONMTF	TF-IDF	5	0.4099
ONMTF	TF-IDF norm	3	0.7098
FNMTF	TF	2	0.0383
FNMTF	TF norm	3	0.3810
FNMTF	TF-IDF	5	0.0387
FNMTF	TF-IDF norm	5	0.2966
OvNMTF	TF	4	0.4190
OvNMTF	TF norm	3	0.8230
OvNMTF	TF-IDF	6	0.0111
OvNMTF	TF-IDF norm	6	0.7172
BinOvNMTF	TF	3	0.0509
BinOvNMTF	TF norm	5	0.7197
BinOvNMTF	TF-IDF	5	0.2263
BinOvNMTF	TF-IDF norm	4	0.7133

Fonte: Lucas Fernandes Brunialti, 2016

5.2.4 Análise qualitativa

Estes experimentos mostram a aplicabilidade dos algoritmos no domínio de mineração de textos, com ênfase nas informações que são possíveis de extrair dos modelos gerados por algoritmos de coclustering baseados em fatoração de matrizes: *ONMTF*, *FNMTF*, *OvNMTF* e *BinOvNMTF*.

Essa análise qualitativa se faz importante, para mostrar que os algoritmos são capazes de encontrar tópicos (grupos de palavras), e possivelmente, explicar os grupos de notícias formados, ou até mesmo, realizar rotulação dos grupos de notícias formados.

Foi necessária a construção de um estudo de caso para avaliar os algoritmos de Coclustering baseados em Fatoração de Matrizes. O estudo de caso é composto por uma análise dos coclusters de palavras correlacionando-os com os coclusters de notícias.

Para isso, foi usado o conjunto de dados ***IG toy***, que foi construído, justamente, para fazer esse tipo de análise.

O portal IG[9] é um dos maiores portais de notícias brasileiro (SITES..., 2016) que é composto por sites importantes como o noticiário Último Segundo, o iG Gente, o iG Esportes, a TV iG, o iG Economia, o Delas e 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 três desses 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 filmes, esportes 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. Sendo 100 notícias para cada canal para compor o conjunto de dados, totalizando 300 notícias.

Esses canais foram escolhidos para formar o conjunto de dados ***IG toy*** por serem de assuntos similares, notícias do canal jovem podem ter semelhanças com notícias do canal esportes, como notícias sobre surfe, por exemplo, notícias do canal arena podem ser similares com notícias do canal de esportes, por existerem games que simulam esportes, ou até mesmo, possuírem similaridades com notícias do canal jovem, pois games é um assunto ligado ao público jovem na sua maioria. Essa escolha torna possível verificar como os algoritmos tratam essa intersecção entre palavras nas notícias.

As subseções a seguir irão mostrar como cada um dos algoritmos de coclusterização baseados em fatoração de matrizes: são úteis para análise de tópicos e palavras no conjunto de dados ***IG toy***.

5.2.4.1 Análise de dados utilizando *ONMTF*

O algoritmo *ONMTF* é capaz de encontrar coclusters de notícias e coclusters de palavras, além disso, cada cocluster de notícias é relacionado com os coclusters de palavras por um fator, assim como cada cocluster de palavras está relacionado com os coclusters de notícias pelo mesmo fator. Estes fatores são extraídos da matriz S.

Isso significa que o algoritmo permite que palavras ou notícias estejam presentes em múltiplos coclusters, com um fator de pertinência para cada cocluster, ou seja, é possível

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

realizar *soft clustering*.Uma abordagem para realizar *hard coclustering* é atribuir uma notícia ou palavra ao cocluster com maior pertinência, a qual foi utilizada nos experimentos quantitativos (Subseção 5.2.3).

Note também que cada notícia esta presente em cada um dos coclusters com um fator associado, assim como cada palavra esta presente em cada um dos coclusters com um fator associado.

Para as análises construídas com o *ONMTF* foi utilizado o modelo que obteve a melhor taxa segundo a métrica *Rand Index* no experimento descrito na seção 5.2.3.6, que foi o modelo com $k = 3$ e $l = 3$ com a representação TF-IDF normalizado.

A figura 13 exemplifica as informações que um modelo gerado pelo algoritmo *ONMTF* provê. A notícia "Avaliação do FIFA 15 por um jogador fanático" foi usada como exemplo. O resultado da coclusterização de notícias foi que dos três coclusters de notícias, a notícia pertence ao cocluster (rotulado manualmente) esportes com um fator equivalente à 40%, também pertence ao cocluster rotulado como arena com um fator equivalente à 60%, e não pertence ao cocluster rotulado como jovem. Cada cocluster de notícias é formado pela combinação dos coclusters de palavras com os fatores que indicam a relação entre coclusters de notícias e coclusters de colunas, denotado pelas linhas que os conecta. Cada cocluster de palavras é ilustrado pelas palavras mais relevantes que o compõem, ou seja, as palavras que contém os maiores fatores (representado em porcentagem) para aquele determinado cocluster.

Ainda sobre a Figura 13, note que as palavras do corpo da notícia foram coloridas de acordo com as cores dos coclusters de palavras as quais pertencem. É possível perceber que existem mais palavras que caracterizam a notícia como sendo sobre o assunto games, o que vai de acordo com a coclusterização realizada.

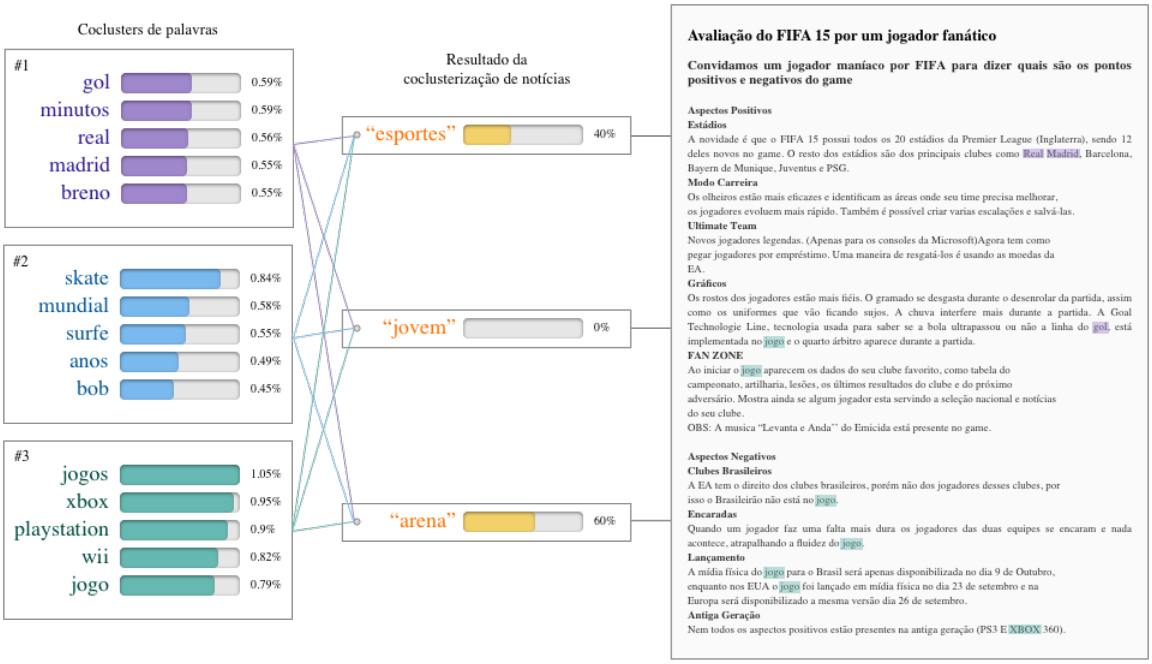


Figura 13 – Exemplo de uma notícia do canal arena e sua disposição quanto aos coclusters de notícias e palavras.

Os fatores que conectam os coclusters de notícias com os coclusters de palavras podem ser observados na matriz S. A matriz S que foi usada nesta análise pode ser observada na Tabela ??. Note que os valores estão normalizados para que a soma dos elementos de cada linha tenha resultado da soma igual a 1. A normalização foi feita para tornar claro que cada cocluster de palavras foi usado pelo algoritmo para caracterizar um grupo de notícias, os valores mostram que o cocluster de notícias "arena" está associado com um maior fator ao cocluster de palavras #3, o cocluster de notícias "esporte" está associado com um maior fator ao cocluster de palavras #1, e o cocluster de notícias "jovem" está associado com um maior fator ao cocluster de palavras #2.

Tabela 6 – Matriz S para $ONMTF$ com $k = l = 3$.

-	**Cocluster de palavras #1**	**Cocluster de palavras #2**	**Cocluster de palavras #3**
Cocluster de notícias "arena"	0.0068	0.0188	**0.9744**
Cocluster de notícias "esporte"	**0.9604**	0.0348	0.0048
Cocluster de notícias "jovem"	0.0444	**0.9324**	0.0232

Os grupos de palavras foram sumarizados através da visualização em *word cloud*, em que o tamanho das palavras é definido pelo seu fator de pertinência ao cocluster correspondente. A Figura 14 mostra essa visualização contendo as 100 palavras com maior fator para cada cogrupo.

(a) Cocluster de palavras #1 (b) Cocluster de palavras #2 (c) Cocluster de palavras #3

Figura 14 – Visualização *word cloud* de palavras para cada cocluster de palavras gerados pelo algoritmo *ONMTF*.

A Figura 14 mostra claramente que cada cocluster de palavras ficou responsável por caracterizar cada um dos três coclusters de notícias. O cocluster de palavras #1 contém palavras sobre esportes, principalmente sobre futebol, como gol, minutos, nomes de seleções, times de futebol e campeonatos. Note que nesse cocluster aparece a palavra jogo, de tamanho pequeno, localizada entre as letras i e n da palavra minutos, que também aparece no cocluster de palavras #3, porém, de tamanho claramente maior. No cocluster de palavras #2 aparecem palavras de esportes, principalmente sobre surfe e skate, como nomes de atletas desses esportes e campeonatos. O cocluster de palavras #3, como esperado, contém palavras ligadas ao assunto games, como jogo, nomes de consoles (xbox, playstation e wii), nomes de grandes empresas do ramo e nomes de games.

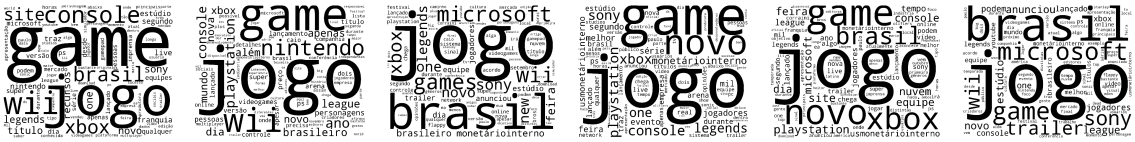
5.2.4.2 Análise de dados utilizando *FNMTF*

5.2.4.3 Análise de dados utilizando *OvNMTF*

O algoritmo *OvNMTF* é capaz de encontrar coclusters de notícias e coclusters de palavras, semelhante aos outros.

Tabela 7 – Matriz S para *OvNMTF* com $k = 3$ e $l = 6$.

-	#1	#2	#3	#4	#5	#6
Cocluster de notícias "arena"	0.1465	**0.2516**	0.0535	**0.3294**	0.1499	0.0691
Cocluster de notícias "esporte"	0.0949	0.0943	**0.2239**	0.1287	**0.2878**	0.1703
Cocluster de notícias "jovem"	0.0381	**0.2545**	**0.3324**	**0.2220**	0.0932	0.0597



(a) #1 (b) #2 (c) #3 (d) #4 (e) #5 (f) #6

Figura 15 – Visualização *word cloud* de palavras para cada cocluster de palavras do cocluster de notícias “arena”, gerados pelo algoritmo *OvNMTF*.

(a) #1 (b) #2 (c) #3 (d) #4 (e) #5 (f) #6

Figura 16 – Visualização *word cloud* de palavras para cada cocluster de palavras do cocluster de notícias “esporte”, gerados pelo algoritmo *OvNMTF*.

(a) #1 (b) #2 (c) #3 (d) #4 (e) #5 (f) #6

Figura 17 – Visualização *word cloud* de palavras para cada cocluster de palavras do cocluster de notícias “jovem”, gerados pelo algoritmo *OvNMTF*.

5.2.4.4 Análise de dados utilizando *BinOvNMTF*

Finalmente, é possível aprender os perfis dos usuários utilizando de técnicas e algoritmos de Aprendizado de Máquina (ADOMAVICIUS; TUZHILIN, 2005; LOPS; GEMMIS; SEMERARO, 2011; JANNACH et al., 2011), por exemplo, para prever se o usuário gosta ou não de um determinado item (classificação).

Referências[10]

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, v. 17, n. 6, p. 734–749, 2005. Citado na página 84.

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 28.

CHENG, Y.; CHURCH, G. M. Biclustering of expression data. In: *Procedures of the 8th ISMB*. [S.l.]: AAAI Press, 2000. p. 93–103. Citado 2 vezes nas páginas 27 e 28.

CORMEN, T. H. et al. *Introduction to Algorithms*. 2nd. ed. [S.l.]: McGraw-Hill Higher Education, 2001. ISBN 0070131511. Citado na página 35.

DING, C. et al. Orthogonal nonnegative matrix tri-factorizations for clustering. In: *In SIGKDD*. [S.l.]: Press, 2006. p. 126–135. Citado 5 vezes nas páginas 30, 31, 36, 37 e 39.

FELDMAN, R.; SANGER, J. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge, MA, USA: Cambridge University Press, 2006. Hardcover. 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 2 vezes nas páginas 19 e 25.

FRANçA, 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 28.

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 2 vezes nas páginas 26 e 27.

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 74.

HAN, J.; KAMBER, M. *Data mining: Concepts and Techniques*. 2. ed. [S.l.]: Morgan Kaufmann San Francisco, Calif, USA, 2006. Citado na página 18.

HAN, J.; KAMBER, M.; PEI, J. *Data Mining: Concepts and Techniques*. 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. Citado na página 91.

HAYKIN, S. *Neural Networks and Learning Machines (3rd Edition)*. 3. ed. [S.l.]: Prentice Hall, 2008. Hardcover. Citado na página 91.

HO, N.-D. *Nonnegative Matriz Factorization Algorithms and Applications*. Tese (Doutorado) — Université Catholique de Louvain, Louvain-la-Neuve, Belgique, 6 2008. Citado 2 vezes nas páginas 19 e 30.

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

HOCHREITER, S. et al. Fabia: factor analysis for bicluster acquisition. *Bioinformatics*, v. 26, n. 12, p. 1520–1527, 2010. Citado 3 vezes nas páginas 24, 28 e 29.

HOTHO, A.; NÜRNBERGER, A.; PAAß, G. A brief survey of text mining. *LDV Forum - GLDV Journal for Computational Linguistics and Language Technology*, v. 20, n. 1, p. 19–62, maio 2005. Citado na página 89.

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 18.

JANNACH, D. et al. *Recommender Systems An Introduction*. [S.l.]: Cambridge University Press, 2011. Citado na página 84.

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 30.

KOREN, Y. *1 The BellKor Solution to the Netflix Grand Prize*. 2009. Citado na página 30.

KUANG, D. *Nonnegative Matrix Factorization For Clustering*. Tese (Doutorado) — University of Tampere, Tampere, Finlândia, 2014. Citado na página 30.

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 20 e 30.

LEE, D. D.; SEUNG, H. S. Algorithms for non-negative matrix factorization. In: *NIPS*. [s.n.], 2000. p. 556–562. Disponível em: <citeseer.ist.psu.edu/lee01algorithms.html>. Citado na página 20.

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. Disponível em: <http://dx.doi.org/10.1109/ICDM.2006.160>. Citado na página 31.

LONG, B.; ZHANG, Z. M.; YU, P. S. *Co-clustering by block value decomposition*. [S.l.]: ACM Press, 2005. 635–640 p. Citado 6 vezes nas páginas 30, 31, 32, 33, 34 e 49.

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 4 vezes nas páginas 76, 84, 89 e 90.

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 3 vezes nas páginas 25, 26 e 59.

MINER, G. et al. *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications*. 1st. ed. [S.l.]: Academic Press, 2012. Citado na página 91.

MURPHY, K. P. *Machine Learning: A Probabilistic Perspective*. [S.l.]: The MIT Press, 2012. Citado na página 91.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 62.

PRELIĆ, A. et al. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, Oxford University Press, Oxford, UK, v. 22, n. 9, p. 1122–1129, maio 2006. Citado na página 28.

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 71.

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. Citado na página 65.

SALAKHUTDINOV, R.; MNIH, A. Probabilistic matrix factorization. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2008. v. 20. Citado na página 30.

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 4 vezes nas páginas 76, 77, 89 e 90.

SANTAMARíA, R.; MIGUEL, L.; THERóN, R. Methods to bicluster validation and comparison in microarray data. *Lecture Notes in Computer Science: Proceedings of IDEAL'07*, v. 4881, p. 780–789, 2007. Citado 2 vezes nas páginas 24 e 28.

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 3 vezes nas páginas 76, 89 e 90.

SHAHNAZ, F. et al. Document clustering using nonnegative matrix factorization. *Information Processing & Management*, v. 42, n. 2, p. 373 – 386, 2006. Citado na página 20.

SITES mais acessados do Brasil. 2016. Disponível em: <http://www.alexa.com/topsites-/countries;1/BR>. Citado na página 80.

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 26.

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 20.

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*. AAAI Press, 2011. (IJCAI'11), p. 1553–1558. ISBN 978-1-57735-514-4. Disponível em: <http://dx.doi.org/10.5591/978-1-57735-516-8-/IJCAI11-261>. Citado 3 vezes nas páginas 31, 40 e 54.

WEISS, S. M.; INDURKHYA, N.; ZHANG, T. *Fundamentals of predictive text mining*. London; New York: Springer-Verlag, 2010. Citado na página 90.

XU, R.; WUNSCH, D. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, IEEE, v. 16, p. 645 – 678, May 2005. Citado na página 18.

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: ACM, 2003. (SIGIR '03), p. 267–273. ISBN 1-58113-646-3. Disponível em: <http://doi.acm.org/10.1145/860435.860485>. Citado na página 20.

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 28.

YOO, J.; CHOI, S. Orthogonal nonnegative matrix tri-factorizations for co-clustering: multiplicative updates on stiefel manifolds. *Information Proessing and Management*, v. 46, p. 559–570, 2010. Citado 7 vezes nas páginas 20, 30, 32, 36, 38, 39 e 50.

Apêndice A – Mineração de Texto

Técnicas de Mineração de Texto são muito usadas para SRs baseados em conteúdo textual (LOPS; GEMMIS; SEMERARO, 2011), principalmente quando o contexto do SR trata de informações não-estruturadas. Mineração de Texto lida com análise de texto, suportando a sua natureza não-estruturada, imprecisa, incerta e difusa, para extração de informação e conhecimento (HOTHO; NÜRNBERGER; PAAß, 2005). Além disso, a área de Mineração de Texto utiliza de técnicas das áreas de Recuperação de Informação e Processamento de Linguagem Natural (PLN), conectando essas técnicas com algoritmos e métodos de Descoberta de Conhecimento em Banco de Dados, Mineração de Dados, Aprendizado de Máquina e Estatística (HOTHO; NÜRNBERGER; PAAß, 2005).

Feldman e Sanger (2006) apresentam uma arquitetura geral para aplicações de Mineração de Textos composta por quatro etapas: *tarefas de pré-processamento*, que preparam os dados para a central de operações de mineração; *central de operações de mineração*, que incluem algoritmos para a descoberta de padrões, tendências e conhecimentos por meio de técnicas e algoritmos; *componentes de apresentação*, que incluem interfaces para o usuário, apresentando visualizações dos conhecimentos gerados na etapa anterior; e *técnicas de refinamento*, também descritas como uma fase de pós-processamento, que incluem métodos para filtrar informações redundantes.

A.1 Tarefas de pré-processamento

As tarefas de pré-processamento incluem rotinas, processos e métodos para a estruturação dos textos presentes nos documentos. A estruturação se faz necessária para a extração de informações e descoberta de conhecimento por meio de técnicas e algoritmos (HOTHO; NÜRNBERGER; PAAß, 2005).

A.1.1 Representação textual

Para a estruturação dos textos é necessário a definição da representação textual dos documentos. O vetor de termos, ou *Vector Space Model* (SALTON; WONG; YANG, 1975), é a representação clássica usada para representar documentos textuais (SEBASTIANI, 2002; LOPS; GEMMIS; SEMERARO, 2011). Cada dimensão desse vetor está associada a um termo, sendo que todas as dimensões representam todos os termos do conjunto de documentos. Formalmente, há um conjunto de documentos $D = \{d_1, d_2, \ldots, d_n\}$, em que d_i representa um documento e n o número total de documentos, e um conjunto de termos $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$, em que t_j representa um termo e m o número de termos presentes em todos os documentos. Representando a frequência de um termo pelo número de vezes

que t_j aparece em um documento d_i, denotado por $ft(t_j, d_i)$, o vetor de termos pode ser construído e representado da seguinte forma: $\vec{vt}_{d_i} = (TF(t_1, d_i), TF(t_2, d_i), \ldots, TF(t_m, d_i))$. Salton, Wong e Yang (1975) argumentam que a representação textual de documentos em vetor de termos é suficiente para separar documentos. Ao invés de frequência de termos, também é usado, a representação binária (SEBASTIANI, 2002), ou seja, t_j aparecendo em d_i corresponde à entrada 1 na dimensão j em $\vec{vt}_{d_i}$. Há também outros métodos para representação textual, como *n-gramas* e *ontologias* (LOPS; GEMMIS; SEMERARO, 2011).

Ainda sobre o vetor de termos, Salton, Wong e Yang (1975) mostram com experimentos em diversos conjuntos de dados, que o uso da normalização nos vetores usando a técnica de Frequência de Termos-Frequência de Documentos Inversa (*Term Frequency-Inversed Document Frequency* – TF-IDF) é capaz de melhorar a separação de documentos:

$$\begin{aligned} TF\text{-}IDF(t_j, d_i) &= TF(t_j, d_i) \cdot IDF(t_j) \\ TF\text{-}IDF(t_j, d_i) &= TF(t_j, d_i) \cdot \left(log_2 \frac{n}{DF(t_j) + 1} \right) \end{aligned} \tag{17}$$

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. Essa normalização faz com que a frequência dos termos que aparecem em muitos documentos seja reduzida, e a frequência dos termos que aparecem em alguns raros documentos seja aumentada, com um fator de $\log_2$.

A.1.2 Tokenização

Para realizar a estruturação de textos e representar os textos dos documentos em vetores de termos, o primeiro processo a ser realizado é a *tokenização*, que cria um dicionário de termos para cada documento através da quebra dos textos desses documentos. A quebra do texto pode ser feita através de caracteres delimitadores de termos, como espaços em branco, pontuações e etc. No entanto, existem casos que esses caracteres podem não ser delimitadores de termos, como por exemplo os termos *Prof.* e *Sr.*. Este problema é chamado de determinação de fim de sentença, e pode ser resolvido por métodos estáticos (*hard-coded*), baseados em regras e métodos de Aprendizado de Máquina (WEISS; INDURKHYA; ZHANG, 2010).

A.1.3 Filtragem

Métodos de filtragem têm a função de retirar termos do conjunto $\mathcal{T}$ que não contribuem para distinguir ou identificar documentos, como exemplo, conjunções (*e*, *pois*, *que*), artigos (*um*, *o*, *a*), preposições (*de*, *para*) e etc. A técnica de retirar determinados

termos de $\mathcal{T}$ a partir de uma lista, é chamada de *stopwords*. Também são usadas outras técnicas, como a eliminação de termos com a frequência muito alta ou muito baixa.

A.1.4 Stemming

A fim de reduzir a ambiguidade de termos, o método de *stemming* é capaz de juntar, em uma única forma, termos relacionados (MINER et al., 2012). Por exemplo, o verbo *fazer* pode se apresentar em diversas formas, como *fazendo*, *fez*, etc. Esse processo é capaz de aumentar a capacidade da representação em distinguir ou identificar documentos, além de reduzir a dimensionalidade, reduzindo também a esparsidade.

A.1.5 Redução de Dimensionalidade

A representação em vetor de termos pode resultar em vetores esparsos num espaço de alta dimensão, que pode fazer com que algoritmos sofram do problema de *Maldição de Dimensionalidade*, que diz respeito à perda de densidade em espaços de alta dimensão, isto significa que medidas de distância se tornam incapazes de detectar padrões em um conjunto de dados (HAYKIN, 2008). Para amenização desse problema, são usados métodos de *redução de dimensionalidade*. A técnica mais comum de *redução de dimensionalidade* é chamada *Análise dos Componentes Principais* (*Principal Component Analysis - PCA*) (MURPHY, 2012). Esta técnica tem o objetivo de encontrar uma representação compacta através da descoberta de k vetores n-dimensionais ortogonais aos dados ($\vec{v}$), em que $k \leq m$. Os vetores são encontrados a partir da minimização da projeção dos dados em $\vec{v}$. Depois de encontrados os vetores $\vec{v}$, é feita a projeção dos dados nesses vetores, resultando em uma representação num espaço mais compacto (HAN; KAMBER; PEI, 2011). É possível aplicar o algoritmo *PCA*, no vetor de termos, diminuindo a dimensionalidade e esparsidade, superando o problema de *Maldição de Dimensionalidade*.