

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

*“Escreva aqui uma epígrafe, se desejar, ou remova esta página...”*

*(Autor da epígrafe)*

## Resumo

BRUNIALTI, Lucas Fernandes. **Resolução do problema de coagrupamento em matrizes de dados esparsas usando fatoração de matrizes.** 2016. 78 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 denominados 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 tratamento 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. 78 p. Dissertation (Master of Science) – School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, Defense Year.

Coagrupamento é uma estratégia para análise de dados capaz de encontrar grupos de objetos, então denominados 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 tratamento 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

Figura 1 – Fatoração da matriz original de dados $X$ em três outras matrizes: $U$ , $S$ e $V$ (Adaptado de Yoo e Choi (2010)). . . . .	32
Figura 2 – A reconstrução da primeira linha $x_1$ de $X$ , através da multiplicação da matriz indicadora de grupos de linhas $U$ pela matriz dos protótipos de linhas ( $SV^T$ ). . . . .	33
Figura 3 – Um exemplo sintético que compara o algoritmo para BVD contra o algoritmo para ONMTF, com pontos sendo dados, as linhas pontilhadas sendo os protótipos de linhas ( $SV^T$ ), e os três conjuntos de pontos sendo os grupos. (a) O algoritmo para BVD encontra uma solução com dois protótipos em um mesmo grupo, deixando um dos grupos sem nenhum protótipo para representá-lo. Apesar de ser uma solução correta, ou seja, encontra um mínimo local, não é a desejada. (b) O algoritmo para ONMTF é capaz de encontrar a solução em que cada protótipo aproxima cada grupo de dados, através das restrições referentes à ortogonalidade, este é capaz de restringir as possíveis soluções para a fatoração $X \approx USV^T$ . (YOO; CHOI, 2010) . . . . .	36
Figura 4 – Uma Variedade Stiefel no espaço Euclidiano, que quando $b = 1$ essa superfície será uma esfera, e um dos possíveis vetores que está contido no conjunto dessa variedade (tangente à esfera). . . . .	38
Figura 5 – 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. . . . .	47
Figura 6 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo <i>k-means</i> . . . . .	51
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 <i>fuzzy k-means</i> . . . . .	52

Figura 8 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo <i>ONMTF</i> . . . . .	53
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 <i>FNMTF</i> . . . . .	55
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 <i>OvNMTF</i> . . . . .	56
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 <i>BinOvNMTF</i> . . . . .	57
Figura 12 – Exemplo de uma notícia do canal arena e sua disposição quanto aos coclusters de notícias e palavras. . . . .	69
Figura 13 – Visualização <i>word cloud</i> de palavras para cada cocluster de palavras gerados pelo algoritmo <i>ONMTF</i> . . . . .	70
Figura 14 – Visualização <i>word cloud</i> de palavras para cada cocluster de palavras do cocluster de notícias “arena”, gerados pelo algoritmo <i>OvNMTF</i> . . . . .	71
Figura 15 – Visualização <i>word cloud</i> de palavras para cada cocluster de palavras do cocluster de notícias “esporte”, gerados pelo algoritmo <i>OvNMTF</i> . . . . .	71
Figura 16 – Visualização <i>word cloud</i> de palavras para cada cocluster de palavras do cocluster de notícias “jovem”, gerados pelo algoritmo <i>OvNMTF</i> . . . . .	71

## **Lista de algoritmos**

Algoritmo 1 – Algoritmo baseado em atualização multiplicativa para solução do BVD . . .	35
Algoritmo 2 – Algoritmo baseado em atualização multiplicativa para solução do ONMTF . . . . .	36
Algoritmo 3 – Algoritmo FNMTF . . . . .	40
Algoritmo 4 – Algoritmo baseado em atualização multiplicativa para solução do OvNMTF . . . . .	44
Algoritmo 5 – Algoritmo BinOvNMTF . . . . .	45

## Lista de tabelas

Tabela 1 – Resumo de qualidade de reconstrução: ok - permite reconstrução; * - sem informação sobre interseção; + - preserva informação de interseção	50
Tabela 2 – Avaliação da qualidade de agrupamento de linhas (l) e colunas (c) sob a medida de avaliação <i>Rand Index</i>	60
Tabela 3 – Estatísticas das bases de dados usadas nos experimentos.	62
Tabela 4 – Melhores resultados dos experimentos.	65
Tabela 5 – Melhores resultados dos experimentos.	66
Tabela 6 – Matriz $S$ para $ONMTF$ com $k = l = 3$ .	69
Tabela 7 – Matriz $S$ para $OvNMTF$ com $k = 3$ e $l = 6$ .	70

## **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**

$\Gamma$  Letra grega Gama

$\Lambda$  Lambda

$\zeta$  Letra grega minúscula zeta

$\in$  Pertence

# Sumário

<b>1</b>	<b>Introdução</b>	18
1.1	Definição do problema	20
1.1.1	Estruturas de coagrupamentos	21
1.1.2	Coagrupamento e fatorização de matrizes	21
1.2	Hipótese	22
1.3	Objetivos	22
1.4	Metodologia	23
1.5	Organização do documento	24
<b>2</b>	<b>Conceitos Fundamentais</b>	25
2.1	Tipos de cogrupo	25
2.2	Algoritmos para Biclusterização	26
2.3	Avaliação de Biclusterização	28
<b>3</b>	<b>Fatoração de matrizes na tarefa de coagrupamento</b>	30
3.1	Non-negative Matrix Factorization	30
3.2	Fatoração de Matrizes Não-Negativas para Coclustering	30
3.2.1	Decomposição de Valores em Blocos para Coclustering	31
3.2.2	Fatoração Ortogonal tripla de Matrizes Não-negativas	34
3.3	Fatoração tripla rápida de Matrizes Não-negativas	39
<b>4</b>	<b>Algoritmos propostos</b>	41
4.1	Fatoração Tripla de Matrizes Não-negativas Sobrepostas	41
4.2	Fatoração Binária Tripla de Matrizes Não-negativas Sobrepostas	44
<b>5</b>	<b>Experimentos e Resultados</b>	46
5.1	Experimentos com Base de Dados sintéticas	46
5.1.1	Análise da reconstrução	49
5.1.2	Reconstrução a partir dos resultados do algoritmo <i>k-means</i>	50

<b>5.1.3</b>	<b>Reconstrução a partir dos resultados do algoritmo <i>fuzzy k-means</i></b>	52
<b>5.1.4</b>	<b>Reconstrução a partir dos resultados do algoritmo <i>ONMTF</i></b>	53
<b>5.1.5</b>	<b>Reconstrução a partir dos resultados do algoritmo <i>FNMTF</i></b>	54
<b>5.1.6</b>	<b>Reconstrução a partir dos resultados do algoritmo <i>OvNMTF</i></b>	55
<b>5.1.7</b>	<b>Reconstrução a partir dos resultados do algoritmo <i>BinOvNMTF</i></b>	56
<b>5.1.8</b>	<b>Análise de particionamento</b>	58
<b>5.1.9</b>	<b>Análise de coagrupamento</b>	60
<b>5.2</b>	<b>Experimentos com base de dados reais</b>	60
<b>5.2.1</b>	<b>Descrição das bases de dados</b>	61
<b>5.2.2</b>	<b>Pré-processamento</b>	62
<b>5.2.3</b>	<b>Experimentos quantitativos</b>	64
5.2.3.1	Setup dos algoritmos	64
5.2.3.2	Setup dos experimentos	65
5.2.3.3	Base de dados <i>NIPS</i>	65
5.2.3.4	Base de dados <i>20Newsgroup</i>	65
5.2.3.5	Base de dados <i>IG</i>	65
5.2.3.6	Base de dados <i>IG toy</i>	65
<b>5.2.4</b>	<b>Análise qualitativa</b>	66
5.2.4.1	Análise de dados utilizando <i>ONMTF</i>	67
5.2.4.2	Análise de dados utilizando <i>FNMTF</i>	70
5.2.4.3	Análise de dados utilizando <i>OvNMTF</i>	70
5.2.4.4	Análise de dados utilizando <i>BinOvNMTF</i>	71
<b>Referências<sup>1</sup></b>		72
<b>Apêndice A–Mineração de Texto</b>		76
<b>A.1</b>	<b>Tarefas de pré-processamento</b>	76
A.1.1	Representação textual	76
A.1.2	Tokenização	77
A.1.3	Filtragem	77

<sup>1</sup> De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

A.1.4	Stemming . . . . .	78
A.1.5	Redução de Dimensionalidade . . . . .	78

## 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 suscetíveis 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 coagrupamento. Nesse contexto, o problema de mineração de textos é modelado como o problema de encontrar uma organização dos textos em grupos que considerem similaridades parciais. 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 coagrupamento 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 óbvio que é dar um jeito de descobrir os grupos com sobreposição e mostrar que é útil 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 cogerupos 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  $I = \{i_1, \dots, i_r\}$  com  $r \leq L$  e  $J = \{j_1, \dots, j_s\}$  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 cogrupo 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 cogrupo, 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 resultados fornecidos por estratégias de agrupamento clássico.

## 1.4 Metodologia

A análise exploratória da literatura especializada 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 à 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<sup>1</sup>. 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

---

<sup>1</sup> <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  $X = \{x_1, \dots, x_n, \dots, x_N\}$  (aqui deveria ser  $X = \{1, 2, \dots, n, \dots, N\}$ ) e um conjunto de colunas  $Y = \{y_1, \dots, y_m, \dots, y_M\}$  (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  $\mu$  não deveria ser indexado por  $IJ$ , isto é,  $\mu_{IJ}$ ? O mesmo para os outros  $\mu$  que aparecem abaixo.) onde  $\mu$  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  $\eta_{ij}$  é o ruído associado com os valores de  $\mu$  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  $\alpha_i$  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  $\beta_j$  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 está 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  $\delta$ -biclusters, na matriz  $A$ , os autores definem o *Resíduo Quadrático Médio* (RQM):

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

em que

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

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  $a_{ij} = a_{iJ} = a_{Ij} = a_{IJ}$ ,  $\forall i, j \in I, J$ , ( $a_{ij} = a_{iJ}$ , 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  $\delta$ -biclusters, [Cheng e Church \(2000\)](#) usam uma estratégia gulosa que retira linhas e colunas, visando a minimização do RQM, respeitando um parâmetro  $\delta$ , 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  $\delta$ -bicluster, convergindo mais rapidamente para uma solução quanto maior for o parâmetro  $\alpha$ , em que  $\alpha \geq 0$ . Ainda, para amenização do problema de encontrar  $\delta$ -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  $k$   $\delta$ -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:

$$\begin{aligned} C_l(A_{IJ}) &= \frac{1}{|I|} \sum_{i=1}^{|I|-1} \sum_{j=i+1}^{|I|} \sqrt{\sum_{k=1}^{|J|} (a_{ik} - a_{jk})^2} \\ C_c(A_{IJ}) &= \frac{1}{|J|} \sum_{i=1}^{|J|-1} \sum_{j=i+1}^{|J|} \sqrt{\sum_{k=1}^{|I|} (a_{ki} - a_{kj})^2} \end{aligned} \quad (3)$$

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:

$$C(A_{IJ}) = \frac{|I| \cdot C_l + |J| \cdot C_c}{|I| + |J|} \quad (4)$$

Uma das métricas externas que são usadas para comparar biclusters encontrados com biclusters reais em um conjunto de dados, é a métrica *consensus 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 na tarefa de coagrupamento

#### 3.1 Non-negative Matrix Factorization

#### 3.2 Fatoração de Matrizes Não-Negativas para Coclustering

Algoritmos de coclustering baseados em Fatoração de Matrizes Não-negativas (*Non-negative Matrix Factorization* - NMF) podem ser úteis em múltiplos contextos, aparecendo em diversos tipos de aplicações, como clusterização de genes e análise de microarray em bioinformática, filtragem colaborativa em sistemas de recomendação, e clusterização de documentos em mineração de textos. Isso acontece pois muitas das representações usadas nessas aplicações se apresentam em tuplas contendo um par de elementos (não seria uma tripla: documento, palavra, e ocorrência?, acho mais fácil falar que é uma matriz, já que em nosso caso temos os dados completos, faz mais sentido falar em tuplas quando os dados não são completos.), cada um pertencendo a um conjunto finito (LONG; ZHANG; YU, 2005). Por exemplo, na aplicação de clusterização de documentos em mineração de textos, usa-se, comumente, dois conjuntos: documentos e palavras, sendo cada observação representada pela contagem de uma determinada palavra em um determinado documento. Note ainda, que conjuntos de dados utilizados nos contextos citados, apresentam-se como uma matriz de dados positiva, expondo outra característica em que NMF podem ser úteis para esses tipos de dados. Além disso, essas técnicas, por reduzir a dimensionalidade original do conjunto de dados, são capazes de lidar com dados de alta dimensionalidade e/ou esparsos, que são geralmente presentes nessas aplicações. (Essa motivação que aparece em outras partes do texto não me convence muito, tem que deixar mais claro.)

Formalizando, algoritmos de Coclusterização baseados em Fatoração de Matrizes Não-negativas, têm como entrada uma matriz de dados  $X$  que representa uma aplicação em algum contexto, sendo  $X \in \mathbb{R}_+^{n \times m}$ , contendo números reais positivos com  $n$  linhas e  $m$  colunas. Esta matriz é formada por um conjunto de vetores de linhas  $\mathcal{X} = \{x_1, \dots, x_n\}$  e um conjunto de vetores de colunas  $\mathcal{Y} = \{y_1, \dots, y_m\}$ , e cada observação representada por  $x_{ij}$ , que é justamente um valor da matriz  $X$ . (Em outra parte do texto utilizava  $N$  como tamanho e  $n$  como índice, aqui já está utilizando  $n$  como tamanho.)

O objetivo é particionar  $\mathcal{X}$  em  $k$  grupos de linhas, denotados pelos subconjuntos ordenados  $I_p \subseteq \mathcal{X}$ , e  $l$  grupos de colunas, subconjuntos  $J_q \subseteq \mathcal{Y}$ , podendo assim, pela junção

desses subconjuntos, formar cogrupos ( $I_p, J_q$ ). Note então, que os algoritmos tradicionais presentes na literatura são capazes de formar  $kl$  cogrupos. (Como escrevi anteriormente, prefiro dizer que  $\mathcal{X} = \{1, 2, \dots, n\}$  e o mesmo para  $\mathcal{Y}$ , então faz mais sentido formar os cogrupos com  $I \subseteq \mathcal{X}$  e o mesmo para  $J$ . Além disso, os índices  $p$  e  $q$  não foram mencionados. Finalmente, aqui já está considerando que os clusters de linha são separados dos cluster de colunas, como é utilizado tradicionalmente na literatura. Dessa forma, deve-se dizer que os conjuntos  $I$  devem formar uma partição de  $\mathcal{X}$ .)

As próximas subseções irão mostrar definições dos algoritmos de coclusterização baseados em NMF presentes na literatura, assim como estratégias usadas para selecionar os cogrupos mais relevantes dos  $kl$  possíveis. (  $kl$  possíveis? existem muito mais cogrupos possíveis.)

### 3.2.1 Decomposição de Valores em Blocos para Coclustering

Um dos primeiros algoritmos propostos na literatura para a resolução do problema de *coclustering* baseado em NMF, é o Decomposição de Valores em Blocos (*Block Value Decomposition* - BVD). Esta decomposição recebe esse nome, justamente, por ter capacidade de encontrar estruturas em blocos na matriz de dados. Isso é possível pois o algoritmo é capaz de explorar a relação entre linhas e colunas, através da decomposição em três matrizes,  $U$  a matriz de coeficientes de linhas,  $S$  a matriz com estrutura em blocos, e  $V$  a matriz de coeficientes de colunas, sendo os coeficientes o grau com que cada vetor (linha ou coluna) de  $X$  pertence a um grupo, e a estrutura em blocos, uma representação compacta da matriz original. Sendo assim, multiplicando as três matrizes geradas pela fatoração, é possível reconstruir a matriz original  $X$  por partes.

O objetivo é encontrar grupos de forma simultânea, sendo  $k$  grupos de  $\mathcal{X}$  (linhas) e  $l$  grupos de  $\mathcal{Y}$  (colunas) (LONG; ZHANG; YU, 2005).

A aproximação  $X \approx USV^T$  é capaz de gerar diversas informações interpretáveis através da análise do resultado da fatoração. Os valores da matriz  $U$  podem ser interpretados como o grau com que cada objeto  $x_i$ .  $\forall i = \{1, \dots, n\}$  pertence à um dos  $k$  grupos de linhas, da mesma forma, os valores da matriz  $V$  podem ser interpretados como o grau com que cada objeto  $y_j$ .  $\forall j = \{1, \dots, m\}$  pertence à um dos  $l$  grupos de colunas. A matriz  $S$  também permite interpretação quando são considerados os grupos formados por  $U$  e  $V$ , cada valor  $s_{pq}$   $\forall p = \{1, \dots, k\}, \forall q = \{1, \dots, l\}$  representa a intensidade em que o

grupo de objetos  $J_p$  e o grupo de características  $J_q$  se relacionam, ou seja, fixando um grupo de objetos  $I_p$ ,  $s_{pq}$  representa a intensidade que  $J_q$  colaborou para a sua formação, alternadamente, fixando um grupo de características  $J_q$ ,  $s_{pq}$  representa a intensidade na qual o grupo de objetos  $I_p$  colaborou para a sua formação. (Parece que aqui é a primeira vez aparece "características" e "objetos", essa interpretação da matriz tem que ser dada mais força anteriormente. Além disso, essa interpretação da matriz  $S$  não me convence muito. Se restringirmos as matrizes  $U$  e  $V$  a serem binárias, a interpretação de  $S$  fica sendo o valor de  $\mu$  comentado no capítulo de "Tipos de Cocluster".)

Um exemplo de fatoração em três matrizes e sua interpretação pode ser visto na Figura 1.

$$X \approx U \times S \times V^T$$

Figura 1 – Fatoração da matriz original de dados  $X$  em três outras matrizes:  $U$ ,  $S$  e  $V$  (Adaptado de [Yoo e Choi \(2010\)](#)).

Note ainda, que é possível obter os protótipos responsáveis por cada parte da reconstrução da matriz original  $X$ . Os vetores das colunas de  $(US)$  são vetores base (protótipos de colunas), geradores de vetores de colunas de  $X$ . Assim como os vetores de linhas de  $(SV^T)$ , que são vetores base (protótipo de linhas), geradores de vetores e linhas de  $X$ . A Figura 2 mostra um exemplo de como os protótipos são responsáveis pela reconstrução de  $X$ .

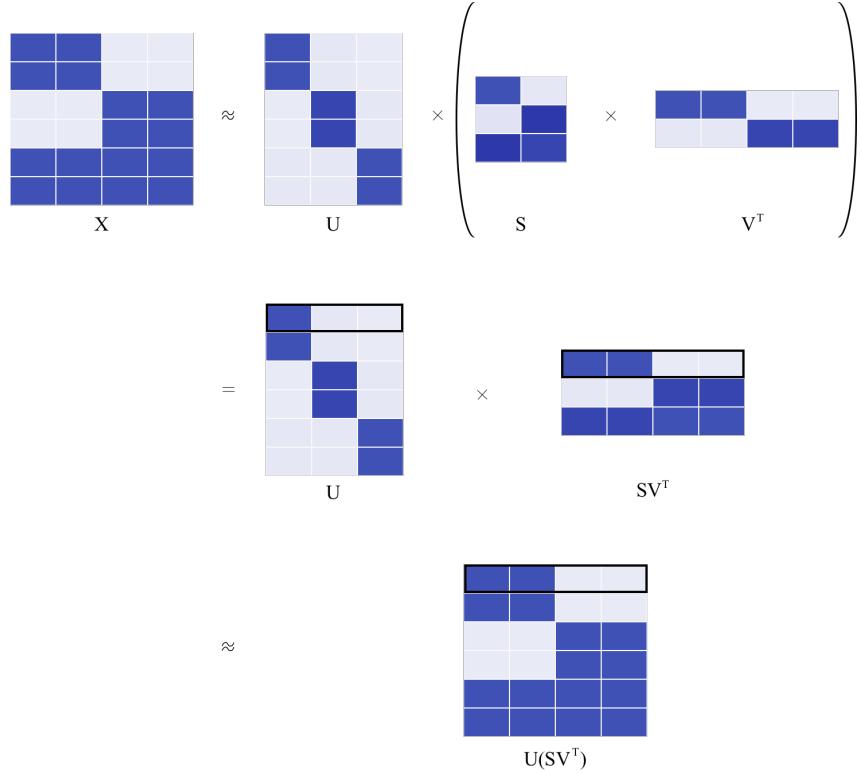


Figura 2 – A reconstrução da primeira linha  $x_1$  de  $X$ , através da multiplicação da matriz indicadora de grupos de linhas  $U$  pela matriz dos protótipos de linhas ( $SV^T$ ).

Apresentando o problema formalmente, temos:

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

$$\begin{aligned} \mathcal{F}_1(U, S, V) &= \min_{U, S, V} \|X - USV^T\|_F^2 \\ &\text{suj. a} \quad U \geq 0, \\ &\quad S \geq 0, \\ &\quad V \geq 0 \end{aligned} \tag{5}$$

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

Introduzindo a função lagrangeana, associada à  $\mathcal{F}_1$ :

$$\mathcal{L}(U, S, V, \Lambda_1, \Lambda_2, \Lambda_3) = \|X - USV^T\|_F^2 - \text{tr}(\Lambda_1 U^T) - \text{tr}(\Lambda_2 S^T) - \text{tr}(\Lambda_3 V^T)$$

onde  $\Lambda_1 \in \mathbb{R}^{n \times k}$ ,  $\Lambda_2 \in \mathbb{R}^{k \times l}$  e  $\Lambda_3 \in \mathbb{R}^{m \times l}$  são os multiplicadores de Lagrange.

Pela teoria de otimização não-linear com restrições,  $\Theta = (U^*, S^*, V^*, \Lambda_1^*, \Lambda_2^*, \Lambda_3^*)$  será um mínimo local estacionário de  $\mathcal{F}_1$ , se e somente se, respeitar as condições de regularidade de *Karush-Kuhn-Tucker* (KKT) ([BAZARAA; SHERALI; SHETTY, 2006](#)):

$$U^* \geq 0, \quad S^* \geq 0, \quad V^* \geq 0 \quad (6a)$$

$$\nabla_{\Theta} \mathcal{L} = 0 \quad (6b)$$

$$\Lambda_1^* \odot U^* = 0, \quad \Lambda_2^* \odot S^* = 0, \quad \Lambda_3^* \odot V^* = 0 \quad (6c)$$

onde  $\odot$  denota o produto de Hadamard.

É possível expandir as equações em [6b](#), calculando as derivadas:

$$\begin{aligned} \nabla_U \mathcal{L} &= USV^T VS^T - XV^T S^T - \Lambda_1 = 0 \\ \nabla_S \mathcal{L} &= U^T USV^T V - U^T XV - \Lambda_2 = 0 \\ \nabla_V \mathcal{L} &= S^T U^T USV^T - S^T U^T X - \Lambda_3 = 0 \end{aligned}$$

Aplicando o produto Hadamard dos dois lados de cada uma das equações e utilizando das condições em [6c](#):

$$\begin{aligned} \nabla_U \mathcal{L} &= U \odot USV^T VS^T - U \odot XV^T S^T = 0 \\ \nabla_S \mathcal{L} &= S \odot U^T USV^T V - S \odot U^T XV = 0 \\ \nabla_V \mathcal{L} &= V \odot S^T U^T USV^T - V \odot S^T U^T X = 0 \end{aligned}$$

Desta forma, é possível resolver para  $U$  de forma algébrica:

$$\begin{aligned} U \odot USV^T VS^T &= U \odot XV^T S^T \\ \implies U \odot \frac{USV^T VS^T}{USV^T VS^T} &= U \odot \frac{XV^T S^T}{USV^T VS^T} \\ \therefore U &= U \odot \frac{XV^T S^T}{USV^T VS^T} \end{aligned}$$

Da mesma maneira é possível resolver para  $S$  e  $V$ , assim como mostram as equações [9](#) e [8](#), descritas no algoritmo [1](#) ([LONG; ZHANG; YU, 2005](#)). Considere  $t$  o contador de iterações, e  $U^{(t)}$ ,  $S^{(t+1)}$  e  $V^{(t+1)}$ , as matrizes  $U$ ,  $S$  e  $V$ , na iteração  $t$ , respectivamente.

### 3.2.2 Fatoração Ortogonal tripla de Matrizes Não-negativas

Baseado no Problema de decomposição em blocos, [Ding et al. \(2006\)](#) propõem o problema [2](#), e o chama de Fatoração Ortogonal Tripla de Matrizes Não-negativas

---

**Algoritmo 1** Algoritmo baseado em atualização multiplicativa para solução do BVD

---

```

1: function BVD( $X$ ,  $maxIter$ )
2:   Inicialize:  $U^{(0)} \geq 0, V^{(0)} \geq 0, S^{(0)} \geq 0$  e  $t \leftarrow 0$ .
3:   while (não convergiu (tem que comentar no texto alguma condição de con-
      vergência)) ou ( $t \leq maxIter$ ) do
4:

$$U^{(t+1)} \leftarrow U^{(t)} \odot \frac{XV^{(t)^T}S^{(t)^T}}{U^{(t)}S^{(t)}V^{(t)^T}V^{(t)}S^{(t)^T}} \quad (7)$$

5:

$$V^{(t+1)} \leftarrow V^{(t)} \odot \frac{S^{(t)^T}U^{(t+1)^T}X}{S^{(t)^T}U^{(t+1)^T}U^{(t+1)}S^{(t)}V^{(t)^T}} \quad (8)$$

6:

$$S^{(t+1)} \leftarrow S^{(t)} \odot \frac{U^{(t+1)^T}XV^{(t+1)}}{U^{(t+1)^T}U^{(t+1)}S^{(t)}V^{(t+1)^T}V^{(t+1)}} \quad (9)$$

7:    $t \leftarrow t + 1$ 
8: end while
9: return  $U^{(t+1)}, S^{(t+1)}, V^{(t+1)}$ 
10: end function

```

---

(*Orthogonal Non-negative Matrix Tri-factorization - ONMTF*). Colocando duas restrições para ortogonalidade nas matrizes indicadoras de grupos de linhas e colunas, restringe o problema da fatoração  $X \approx USV^T$  para um número menor de possíveis soluções, buscando a unicidade, assim como mostra a figura 3.

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

$$\begin{aligned} \mathcal{F}_2(U, S, V) = \min_{U, S, V} & \|X - USV^T\|_F^2 \\ \text{suj. a } & U \geq 0, S \geq 0, V \geq 0, \\ & U^T U = I, \\ & V^T V = I \end{aligned} \quad (10)$$

onde  $U \in \mathbb{R}_+^{n \times k}$ ,  $S \in \mathbb{R}_+^{k \times l}$ ,  $V \in \mathbb{R}^{m \times l}$ ,  $U \geq 0, S \geq 0, V \geq 0$  sendo todos os elementos de  $U$ ,  $S$  e  $V$ , maior que 0, respectivamente,  $U^T U = I$  e  $V^T V = I$  as restrições de ortonormalidade para as matrizes indicadoras de grupos de linhas e colunas, respectivamente, e  $\|\cdot\|_F$  denota a norma de Frobenius para matrizes.

Ding et al. (2006) propõem uma solução semelhante ao que foi apresentado na subseção 3.2.1, fazendo a derivação através da função lagrangeana e a introdução dos multiplicadores de lagrange, utilizando as condições de otimização não-linear de KKT, derivando as regras para atualização multiplicativa para  $U$ ,  $S$  e  $V$ , apresentadas no algoritmo 2. Considere  $t$  o contador de iterações, e  $U^{(t)}$ ,  $S^{(t+1)}$  e  $V^{(t+1)}$ , as matrizes  $U$ ,  $S$  e  $V$ , na iteração  $t$ , respectivamente.

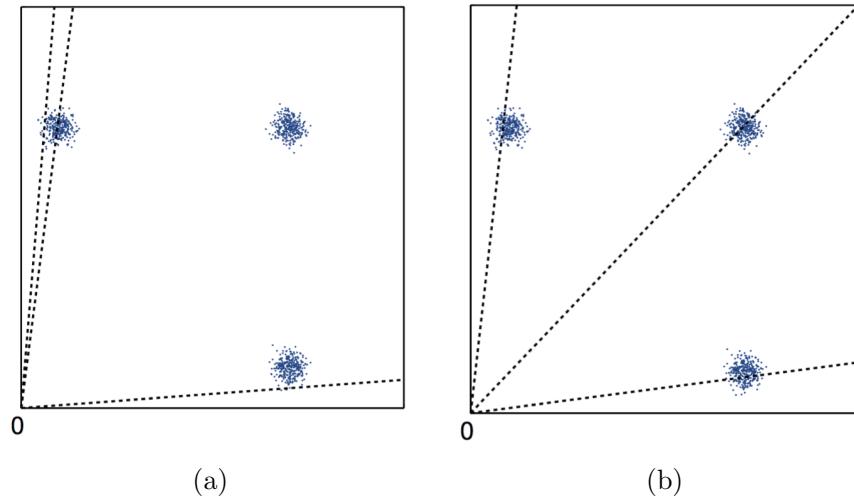


Figura 3 – Um exemplo sintético que compara o algoritmo para BVD contra o algoritmo para ONMTF, com pontos sendo dados, as linhas pontilhadas sendo os protótipos de linhas ( $SV^T$ ), e os três conjuntos de pontos sendo os grupos. (a) O algoritmo para BVD encontra uma solução com dois protótipos em um mesmo grupo, deixando um dos grupos sem nenhum protótipo para representá-lo. Apesar de ser uma solução correta, ou seja, encontra um mínimo local, não é a desejada. (b) O algoritmo para ONMTF é capaz de encontrar a solução em que cada protótipo aproxima cada grupo de dados, através das restrições referentes à ortogonalidade, este é capaz de restringir as possíveis soluções para a fatoração  $X \approx USV^T$ . (YOO; CHOI, 2010)

**Algoritmo 2** Algoritmo baseado em atualização multiplicativa para solução do ONMTF

```

1: function ONMTF( $X$ ,  $maxIter$ )
2:   Inicialize:  $U^{(0)} \geq 0$ ,  $V^{(0)} \geq 0$ ,  $S^{(0)} \geq 0$  e  $t \leftarrow 0$ .
3:   while (não convergiu) ou ( $t \leq maxIter$ ) do
4:     
$$U^{(t+1)} \leftarrow U^{(t)} \odot \frac{XV^{(t)}S^{(t)T}}{U^{(t)}U^{(t)T}XV^{(t)}S^{(t)T}} \quad (11)$$

5:     
$$V^{(t+1)} \leftarrow V^{(t)} \odot \frac{X^TU^{(t+1)}S}{V^{(t)}V^{(t)T}X^TU^{(t+1)}S^{(t)}} \quad (12)$$

6:     
$$S^{(t+1)} \leftarrow S^{(t)} \odot \frac{U^{(t+1)T}XV^{(t+1)}}{U^{(t+1)T}U^{(t+1)}S^{(t)}V^{(t+1)T}V^{(t+1)}} \quad (13)$$

7:      $t \leftarrow t + 1$ 
8:   end while
9:   return  $U^{(t+1)}$ ,  $S^{(t+1)}$ ,  $V^{(t+1)}$ 
10: end function

```

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

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

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  $\Theta$  da função  $\mathcal{J}$ :

$$\Theta \leftarrow \Theta \odot \left( \frac{[\nabla \mathcal{J}]^-}{[\nabla \mathcal{J}]^+} \right)^{\cdot \eta} \quad (14)$$

onde  $\odot$  representa o produto Hadamard,  $(\cdot)^{\cdot \eta}$  representa a potência para cada elemento, e  $\eta$  uma taxa de aprendizado ( $0 < \eta \leq 1$ ). Então, se  $\Theta$  for inicializado com elementos positivos, é possível verificar que a regra de atualização multiplicativa da equação 14 mantém a não-negatividade de  $\Theta$ .

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*), neste caso é usada a Variedade de Stiefel no espaço euclidiano, denotada por  $\mathcal{H}_{a,b}$ , sendo essa variedade o conjunto de  $a \times b$  matrizes ortonormais no espaço  $\mathbb{R}^a$ , formalmente:

$$\mathcal{H}_{a,b} = \{Y \in \mathbb{R}^{a \times b} : Y^T Y = I\}$$

Note que quando  $b = 1$ , a superfície se torna uma esfera. Para otimização, considerando que essa esfera sejam as restrições do problema, o ideal é propor métodos que permanecem na esfera, então, todos os vetores tangentes à essa esfera, podem ser direções possíveis para um algoritmo de otimização iterativo, como mostra a Figura 4.

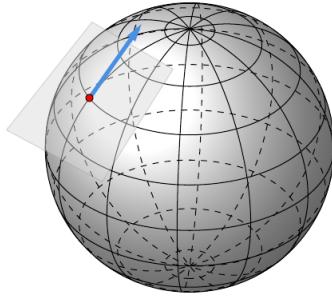


Figura 4 – Uma Variedade Stiefel no espaço Euclidiano, que quando  $b = 1$  essa superfície será uma esfera, e um dos possíveis vetores que está contido no conjunto dessa variedade (tangente à esfera).

[Edelman, Arias e Smith \(1999\)](#) definem o gradiente em um ponto  $Y$ , com a restrição  $Y^T Y = I$ , de uma função  $\mathcal{J}$  definida em uma Variedade Stiefel no espaço euclidiano como:

$$\tilde{\nabla}_Y \mathcal{J} = \nabla_Y \mathcal{J} - Y(\nabla_Y \mathcal{J})^T Y \quad (15)$$

onde  $\nabla_Y \mathcal{J}$  é o gradiente da função  $\mathcal{J}$  para todos os elementos da matriz  $Y$ .

Dada a equação 15, [Yoo e Choi \(2010\)](#) propõem os seguintes cálculos dos gradientes em Variedades Stiefel para  $\mathcal{F}_2$ , com  $U$  no conjunto  $\{U^T U = I\}$ , e com  $V$  no conjunto  $\{V^T V = I\}$ :

$$\begin{aligned}\tilde{\nabla}_U \mathcal{F}_2 &= \nabla_U \mathcal{F}_2 - U(\nabla_U \mathcal{F}_2)^T U = USV^T X^T U - XVS^T \\ \tilde{\nabla}_V \mathcal{F}_2 &= \nabla_V \mathcal{F}_2 - V(\nabla_V \mathcal{F}_2)^T V = VS^T U^T XV - X^T US\end{aligned}$$

Restando apenas o cálculo do gradiente para  $S$ , que como não há restrições, será igual à atualização do algoritmo para solução do BVD:

$$\tilde{\nabla}_S \mathcal{F}_2 = \nabla_S \mathcal{F}_2 = U^T USV^T V - U^T X V$$

É possível, assim, usar a estratégia da equação 14 e os gradientes calculados, para propor uma solução para o problema de ONMTF, alternativas às atualizações

das equações 11, 12 e 13, através da atualização multiplicativa. Essas atualizações são apresentadas nas equações 16a, 16b e 16c.

$$U^{(t+1)} \leftarrow U^{(t)} \odot \frac{XV^{(t)}S^{(t)T}}{U^{(t)}S^{(t)V^{(t)T}X^TU^{(t)}}} \quad (16a)$$

$$V^{(t+1)} \leftarrow V^{(t)} \odot \frac{X^TU^{(t+1)}S^{(t)}}{V^{(t)}S^{(t)T}U^{(t+1)T}XV^{(t)}} \quad (16b)$$

$$S^{(t+1)} \leftarrow S^{(t)} \odot \frac{U^{(t+1)T}XV^{(t+1)}}{U^{(t+1)T}U^{(t+1)}S^{(t)V^{(t+1)T}V^{(t+1)}}} \quad (16c)$$

### 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*) foi proposto por [Wang et al. \(2011\)](#), com os seguintes argumentos contra o uso prático dos problemas até agora propostos para encontrar cogrupos: os problemas de fatoração em três matrizes propostos até o momento, envolviam soluções algorítmicas iterativas, com multiplicações de matrizes intensas em cada etapa; e pelo fato dos algoritmos propostos realizarem coagrupamento relaxado, necessitando de pós-processamento ([como é feito o pós-processamento? deve-se falar disso.](#)) para a extração dos grupos, o que implica em inúmeras soluções para o agrupamento.

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

$$\begin{aligned} \mathcal{F}_3(F, S, G) &= \min_{F, S, G} \|X - FSG^T\|_F^2 \\ &\quad F \in \Psi^{n \times k}, \\ &\quad G \in \Psi^{m \times l} \end{aligned} \quad (17)$$

onde  $X \in \mathbb{R}^{n \times m}$ ,  $S \in \mathbb{R}^{k \times l}$ ,  $\Psi = \{0, 1\}$ . ([Acho que a restrição binária não é a única.](#)  
[Temos que a soma de linhas ou colunas deve somar 1.](#))

Assim como nos outros algoritmos,  $F$  é uma matriz indicadora dos grupos de linhas,  $G$  uma matriz indicadora dos grupos de colunas, e  $S$  contém os fatores que conectam um grupo de linhas aos grupos de colunas, e, um grupo de colunas aos grupos de linhas.

Como não há restrições em  $\mathcal{F}_3$  (não há restrições em  $S$ ?), é possível encontrar uma regra de atualização para  $S$ , e portanto, minimização de  $\mathcal{F}_3$ :

$$\begin{aligned}
 \nabla_S \mathcal{L} &= F^T X G - F^T F S G^T G = 0 \\
 \implies F^T F S G^T G &= F^T X G \\
 \implies (F^T F)^{-1} F^T F S G^T G (G^T G)^{-1} &= (F^T F)^{-1} F^T X G (G^T G)^{-1} \\
 \therefore S &= (F^T F)^{-1} F^T X G (G^T G)^{-1} \tag{18}
 \end{aligned}$$

Sendo assim, é possível resolver os subproblema para atualizar  $F$  e  $G$ . Primeiramente, fixando  $S$  e  $G$  e resolvendo o problema 3 para  $F$  de forma iterativa, verificando para cada linha de  $X$ , qual o protótipo mais o aproxima (semelhante ao algoritmo de agrupamento *K-means*), da mesma forma, é possível fixar  $S$  e  $F$  para alcançar uma solução iterativa para  $G$ , assim como mostra o algoritmo 3. Considere os índices  $i = \{1, \dots, n\}$ ,  $j = \{1, \dots, m\}$ ,  $p = p' = \{1, \dots, k\}$ , e  $q = q' = \{1, \dots, l\}$ , o contador de iterações  $t$ , e  $U^{(t)}$ ,  $S^{(t+1)}$  e  $V^{(t+1)}$ , as matrizes  $U$ ,  $S$  e  $V$ , na iteração  $t$ , respectivamente.

---

### Algoritmo 3 Algoritmo FNMTF

---

```

1: function FNMTF( $X, maxIter$ )
2:   Inicialize:  $F^{(0)} \geq 0$  ( $\in \{0, 1\}$  e somatório igual a 1),  $S^{(0)} \geq 0, G^{(0)} \geq 0$  e  $t \leftarrow 0$ .
3:   while (não convergiu) ou ( $t \leq maxIter$ ) do
4:
5:      $S^{(t+1)} \leftarrow (F^{(t)^T} F^{(t)})^{-1} F^{(t)^T} X G^{(t)} (G^{(t)^T} G^{(t)})^{-1}$  (19)
6:
7:      $\tilde{g}_{pj} \leftarrow (S^{(t+1)} G^{(t)^T})_{pj}, \forall p, j$ 
8:
9:      $(F^{(t+1)})_{ip} \leftarrow \begin{cases} 1 & p = \arg \min_{p' \in \{1, \dots, k\}} \|x_{i \cdot} - \tilde{g}_{p'}\|^2 \\ 0 & \text{caso contrário} \end{cases} \quad \forall i, p$  (20)
10:
11:     $\tilde{f}_{iq} \leftarrow (F^{(t+1)} S^{(t+1)})_{iq}, \forall i, q$ 
12:
13:     $(G^{(t+1)})_{jq} \leftarrow \begin{cases} 1 & q = \arg \min_{q' \in \{1, \dots, l\}} \|x_{\cdot j} - \tilde{f}_{\cdot q'}\|^2 \\ 0 & \text{caso contrário} \end{cases} \quad \forall j, q$  (21)
14:
15:     $t \leftarrow t + 1$ 
16:  end while
17:  return  $F^{(t+1)}, S^{(t+1)}, G^{(t+1)}$ 
18: end function

```

---

## 4 Algoritmos propostos

### 4.1 Fatoração Tripla de Matrizes Não-negativas Sobrepostas

**Problema 4** (Problema de Fatoração Tripla de Matrizes Não-negativas Sobrepostas).

$$\begin{aligned} \mathcal{F}_4(U, S, V_{(1)}, \dots, V_{(k)}) &= \min_{U, S, V_{(1)}, \dots, V_{(k)}} \left\| X - U \sum_{p=1}^k I_{(p)} S V_{(p)}^T \right\|_F^2 \\ &\text{suj. a} \quad U \geq 0, \\ &\quad S \geq 0, \\ &\quad V_{(p)} \geq 0, \quad \forall p \end{aligned} \tag{22}$$

sendo os índices  $p = \{1, \dots, k\}$ , o conjunto de matrizes  $\{V_{(1)}, \dots, V_{(k)}\}$ , em que cada  $V_{(p)} \in \mathbb{R}_+^{m \times l}$ , uma matriz seletora  $I_{(p)}$  que contém 1 na posição  $I_{p,p}$  da sua diagonal e 0 no resto,  $U \in \mathbb{R}_+^{n \times k}$ ,  $S \in \mathbb{R}_+^{k \times l}$ , e  $\|\cdot\|_F$  a norma de Frobenius para matrizes.

Expandindo  $\mathcal{F}_4$ , para tornar o cálculo do gradiente mais simples, é possível obter a seguinte equação:

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

Para o cálculo do gradiente, é usada a mesma estratégia presente em [Yoo e Choi \(2010\)](#):

$$\nabla \mathcal{F}_4 = [\nabla \mathcal{F}_4]^+ - [\nabla \mathcal{F}_4]^-$$

Considere as seguintes igualdades para o cálculo dos gradientes, sendo  $A, Q, B$  e  $C$  matrizes de quaisquer dimensões que as tornem possíveis de serem multiplicadas:

$$\begin{aligned} \nabla_Q \operatorname{tr}(AQB) &= A^T B^T \\ \nabla_{Q^T} \operatorname{tr}(AQB) &= BA \end{aligned} \tag{23}$$

$$\begin{aligned} \nabla_Q \operatorname{tr}(AQBQ^T C) &= BQ^T CA + B^T Q^T A^T C^T \\ \nabla_{Q^T} \operatorname{tr}(AQBQ^T C) &= A^T C^T QB^T + CAQB \end{aligned} \tag{24}$$

Para o cálculo de  $\nabla_U \mathcal{F}_4$ , considere as partes positiva e negativa,  $[\nabla_U \mathcal{F}_4]^+ - [\nabla_U \mathcal{F}_4]^-$ . Usando a igualdade da equação 23, com  $A = X^T$ ,  $B = \sum_{p=1}^k I_{(p)} S V_{(p)}^T$  e  $Q = U$ , é possível obter  $\mathcal{F}_4]^-$ .

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

Para o cálculo de  $[\nabla_U \mathcal{F}_4]^+$ , é utilizado a igualdade da equação 24, com  $A = \sum_{p=1}^k V_{(p)} S^T I_{(p)}$ ,  $B = I$ ,  $Q = U^T$  e  $C = \sum_{p'=1}^k I_{(p')} S V_{(p')}^T$ .

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

Usando a igualdade da equação 23 para todas as partes da soma de  $p = 1, \dots, k$ , com  $A = X^T U I_{(p)}$ ,  $Q = S$ , e  $B = V_{(p)}^T$ , é possível obter  $[\nabla_S \mathcal{F}_4]^-$ :

$$\begin{aligned} [\nabla_S \mathcal{F}_4]^- &= -2 \nabla_S \left( \text{tr}[X^T U I_{(1)} S V_{(1)}^T] + \dots + \text{tr}[X^T U I_{(k)} S V_{(k)}^T] \right) \\ &= -2 \left( I_{(1)} U^T X V_{(1)} + \dots + I_{(k)} U^T X V_{(k)} \right) \\ &= -2 \sum_{p=1}^k I_p U^T X V_{(p)} \end{aligned}$$

Para o cálculo de  $[\nabla_S \mathcal{F}_4]^+$ , é utilizado a igualdade da equação 24 para todas as partes da soma de  $p = p' = 1, \dots, k$ , com  $A = V_{(p)}$ ,  $Q = S^T$ ,  $B = I_{(p)} U^T U I_{(p')}$  e  $C = V_{(p')}^T$ .

$$\begin{aligned} [\nabla_S \mathcal{F}_4]^+ &= \nabla_S \left( \text{tr}(X^T X) + \text{tr}[V_{(1)} S^T I_{(1)} U^T U I_{(1)} S V_{(1)}^T] + \dots + \text{tr}[V_{(1)} S^T I_{(1)} U^T U I_{(k)} S V_{(k)}^T] \right. \\ &\quad \left. + \dots + \text{tr}[V_{(k)} S^T I_{(k)} U^T U I_{(1)} S V_{(1)}^T] + \dots + \text{tr}[V_{(k)} S^T I_{(k)} U^T U I_{(k)} S V_{(k)}^T] \right) \\ &= (I_{(1)} U^T U I_{(1)} S V_{(1)}^T V_{(1)} + I_{(1)} U^T U I_{(1)} S V_{(1)}^T V_{(1)}) \\ &\quad + \dots + (I_{(1)} U^T U I_{(k)} S V_{(k)}^T V_{(1)} + I_{(k)} U^T U I_{(1)} S V_{(1)}^T V_{(k)}) \\ &\quad + \dots + (I_{(k)} U^T U I_{(1)} S V_{(1)}^T V_{(k)} + I_{(1)} U^T U I_{(k)} S V_{(k)}^T V_{(1)}) \\ &\quad + \dots + (I_{(k)} U^T U I_{(k)} S V_{(k)}^T V_{(k)} + I_{(k)} U^T U I_{(k)} S V_{(k)}^T V_{(k)}) \\ &= 2 \sum_{p=1}^k \sum_{p'=1}^k I_{(p)} U^T U I_{(p')} S V_{(p')}^T V_{(p)} \end{aligned}$$

Usando a igualdade da equação 23, com  $A = X^T U I_{(p)} S$ ,  $Q = V_{(p)}^T$ , e  $B = I$ ,  $\forall p$ , é possível obter  $[\nabla_{V_{(p)}} \mathcal{F}_4]^-$ :

$$\begin{aligned} [\nabla_{V_{(p)}} \mathcal{F}_4]^- &= -2 \nabla_{V_{(p)}} \left( \text{tr}[X^T U I_{(1)} S V_{(1)}^T] + \dots + \text{tr}[X^T U I_{(k)} S V_{(k)}^T] \right) \\ &= -2 X^T U \sum_{p'=1}^k I_{(p')} S \end{aligned}$$

Fixando a derivação para  $[\nabla_{V_{(p)}} \mathcal{F}_4]^+$  é possível notar 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 24, com  $A = I$ ,  $Q = V_{(p)}$ ,  $B = S^T I_{(p)} U^T U I_{(p')} S$  e  $C = I$ ,  $\forall p, p'$ :

$$\begin{aligned} [\nabla_{V_{(p)}} \mathcal{F}_4]_{p=p'}^+ &= \nabla_{V_{(p)}} \left( \text{tr}[V_{(1)} S^T I_{(1)} U^T U I_{(1)} S V_{(1)}^T] + \dots + \text{tr}[V_{(k)} S^T I_{(k)} U^T U I_{(k)} S V_{(k)}^T] \right) \\ &= \nabla_{V_{(p)}} \left( \text{tr}[V_{(p)} S^T I_{(p)} U^T U I_{(p)} S V_{(p)}^T] \right) \\ &= 2 V_{(p)} S^T I_{(p)} U^T U I_{(p)} S \end{aligned}$$

Para o caso em que  $p \neq p'$ , é usada a igualdade da equação 23 em duas outras situações, a qual  $V_{(p)}$  esta localizado à esquerda, então,  $Q = V_{(p)}$ ,  $A = I$  e  $B = S^T I_{(p)} U^T U I_{(p')}$ ; e quando  $V_{(p)}$  esta localizado à direita, então,  $Q = V_{(p)}^T$ ,  $A = V_{(p')} S^T I_{(p')} U^T U I_{(p)} S V_{(p)}^T$ ,  $\forall p, p'$ :

$$\begin{aligned} [\nabla_{V_{(p)}} \mathcal{F}_4]_{p \neq p'}^+ &= \nabla_{V_{(p)}} \left( \text{tr} [V_{(1)} S^T I_{(1)} U^T U I_{(2)} S V_{(2)}^T] + \dots + \text{tr} [V_{(1)} S^T I_{(1)} U^T U I_{(k)} S V_{(k)}^T] \right. \\ &\quad \left. + \dots + \text{tr} [V_{(k)} S^T I_{(k)} U^T U I_{(1)} S V_{(1)}^T] + \dots + \text{tr} [V_{(k)} S^T I_{(k)} U^T U I_{(k-1)} S V_{(k-1)}^T] \right) \\ &= \sum_{p' \neq p} \left[ \nabla_{V_{(p)}} \left( \text{tr} [V_{(p)} S^T I_{(p)} U^T U I_{(p')} S V_{(p')}^T] \right) \right. \\ &\quad \left. + \nabla_{V_{(p)}} \left( \text{tr} [V_{(p')} S^T I_{(p')} U^T U I_{(p)} S V_{(p)}^T] \right) \right] \\ &= \sum_{p' \neq p} 2(V_{(p')} S^T I_{(p')} U^T U I_{(p)} S) \end{aligned}$$

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

$$\begin{aligned} [\nabla_{V_{(p)}} \mathcal{F}_4]_{p \neq p'}^+ &= [\nabla_{V_{(p)}} \mathcal{F}_4]_{p=p'}^+ + [\nabla_{V_{(p)}} \mathcal{F}_4]_{p \neq p'}^+ \\ &= 2(V_{(p)} S^T I_{(p)} U^T U I_{(p)} S + \sum_{p' \neq p} V_{(p')} S^T I_{(p')} U^T U I_{(p)} S) \end{aligned}$$

O resultado final dos gradientes para  $U, S, V_{(p)}, \forall, p \in \{1, \dots, k\}$  são apresentados nas equações 25, 26 e 27, respectivamente.

$$[\nabla_U \mathcal{F}_4] = 2(-X \sum_{p=1}^k V_{(p)} S^T I_{(p)} + U \sum_{p=1}^k \sum_{p'=1}^k I_{(p)} S V_{(p)}^T V_{(p')} S^T I_{(p')}) \quad (25)$$

$$[\nabla_S \mathcal{F}_4] = 2(-\sum_{p=1}^k I_p U^T X V_{(p)} + \sum_{p=1}^k \sum_{p'=1}^k I_{(p)} U^T U I_{(p')} S V_{(p')}^T V_{(p)}) \quad (26)$$

$$[\nabla_{V_{(p)}} \mathcal{F}_4] = 2(-X^T U \sum_{p'=1}^k I_{(p')} S + V_{(p)} S^T I_{(p)} U^T U I_{(p)} S + \sum_{p' \neq p} V_{(p')} S^T I_{(p')} U^T U I_{(p)} S) \quad (27)$$

Sendo assim, as regras de atualização para as matrizes  $U, S, V_{(p)}, \forall, p \in \{1, \dots, k\}$  são mostradas nas equações 28, 30 e 29, apresentadas no algoritmo 4. Note que essa é a regra de atualização multiplicativa, adotando a mesma estratégia que em (YOO; CHOI, 2010) (mencionada no Capítulo 3 deste trabalho), que divide a parte negativa do gradiente pela parte positiva.

---

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

---

```

1: function OvNMTF( $X, maxIter$ )
2:   Inicialize:  $U^{(0)} \geq 0, S^{(0)} \geq 0, V_{(0)}^{(0)} \geq 0, \dots, V_{(k)}^{(0)} \geq 0$  e  $t \leftarrow 0$ .
3:   while (não convergiu) ou ( $t \leq maxIter$ ) do
4:

$$U^{(t+1)} \leftarrow U^{(t)} \odot \sum_{p=1}^k \frac{XV_{(p)}^{(t)}S^{(t)T}I_{(p)}}{\sum_{p'=1}^k U^{(t)}I_{(p)}S^{(t)}V_{(p)}^{(t)T}V_{(p')}^{(t)}S^{(t)T}I_{(p')}} \quad (28)$$

5:     for  $p \leftarrow 1, k$  do
6:

$$V_{(p)}^{(t+1)} \leftarrow V_{(p)}^{(t)} \odot \frac{\sum_{p'=1}^k X^T U^{(t+1)T} I_{(p)} S^{(t)}}{V_{(p)}^{(t)} S^{(t)T} I_{(p)} U^{(t+1)T} U^{(t+1)} I_{(p)} S^{(t)} + \sum_{p' \neq p} V_{(p')}^{(t)} S^{(t)T} I_{(p')} U^{(t+1)T} U^{(t+1)} I_{(p)} S^{(t)}} \quad (29)$$

7:   end for
8:

$$S^{(t+1)} \leftarrow S^{(t)} \odot \sum_{p=1}^l \frac{I_{(p)} U^{(t+1)T} X V_{(p)}^{(t+1)T}}{\sum_{p'=1}^l I_{(p)} U^{(t+1)T} U^{(t+1)} I_{(p')} S^{(t)} V_{(p')}^{(t+1)T} V_{(p)}^{(t+1)}} \quad (30)$$

9:    $t \leftarrow t + 1$ 
10:  end while
11:  return  $U^{(t+1)}, S^{(t+1)}, V_{(1)}^{(t+1)}, \dots, V_{(k)}^{(t+1)}$ 
12: end function

```

---

## 4.2 Fatoração Binária Tripla de Matrizes Não-negativas Sobrepostas

**Problema 5** (Problema de Fatoração Binária Tripla de Matrizes Não-negativas Sobrepostas).

$$\begin{aligned}
\mathcal{F}_5(F, S, G_{(1)}, \dots, G_{(k)}) &= \min_{F, S, G_{(1)}, \dots, G_{(k)}} \left\| X - F \sum_{p=1}^k I_{(p)} S G_{(p)}^T \right\|_F^2 \\
&\text{subj. a} & F &\in \Psi^{n \times k}, \\
&& S &\geq 0, \\
&& G_{(p)} &\in \Psi^{m \times l}, \quad \forall p
\end{aligned} \quad (31)$$

sendo  $\Psi = \{0, 1\}$ , os índices  $p = \{1, \dots, k\}$ , o conjunto de matrizes  $\mathcal{G} = \{G_{(1)}, \dots, G_{(k)}\}$ , uma matriz seletora  $I_{(p)}$  que contém 1 na posição  $I_{p,p}$  da sua diagonal e 0 no resto,  $S \in \mathbb{R}_+^{k \times l}$ ,  $X \in \mathbb{R}_+^{n \times m}$ , e  $\|\cdot\|_F$  a norma de Frobenius para matrizes.

---

**Algoritmo 5** Algoritmo BinOvNMTF
 

---

1: **function** BINOVNMTF( $X, maxIter$ )  
 2:   **Inicialize:**  $F^{(0)} \geq 0, S^{(0)} \geq 0, G_{(0)}^{(0)} \geq 0, \dots, G_{(k)}^{(0)} \geq 0$  e  $t \leftarrow 0$ .  
 3:   **while** (não convergiu) ou ( $t \leq maxIter$ ) **do**  
 4:     (esses 4 somatórios estão corretos? Note que  $C_p$  e  $W_p$  nunca é atualizado.)  

$$(S^{(t+1)})_{pq} \leftarrow \frac{1}{|\{\mathbf{x}_{i \cdot} \in \mathcal{C}_p\}| |\{\mathbf{x}_{\cdot j} \in \mathcal{W}_q\}|} \sum_{i=1}^n \sum_{j=1}^m \sum_{\mathbf{x}_{i \cdot} \in \mathcal{C}_p} \sum_{\mathbf{x}_{\cdot j} \in \mathcal{W}_q} x_{ij} \quad (32)$$
  
 5:     **for**  $p \leftarrow 1, k$  **do**  
 6:       (melhor falar  $\tilde{\mathbf{f}} = \mathbf{f} \times S$ , parece que abaixo está faltando a multiplicação por  $S$ )  

$$\tilde{\mathbf{F}} \leftarrow \begin{bmatrix} & & \vdots & \\ \cdots & \mathbf{f}_{i \cdot}^{(t)}, \forall \mathbf{x}_{i \cdot} \in \mathcal{C}_p & \cdots & \\ & & \vdots & \end{bmatrix} S^{(t+1)}$$
  
 7:       (faz sentido essa atualização? não basta utilizar o  $X$ ?, isto é, não precisa o  $\tilde{X}$ .)  

$$\tilde{\mathbf{X}} \leftarrow \begin{bmatrix} & & \vdots & \\ \cdots & \mathbf{x}_{i \cdot}, \forall \mathbf{x}_{i \cdot} \in \mathcal{C}_p & \cdots & \\ & & \vdots & \end{bmatrix}$$
  
 8:  

$$(G_{(p)}^{(t+1)})_{jq} \leftarrow \begin{cases} 1 & q = \arg \min_{q' \in \{1, \dots, l\}} \left\| \tilde{\mathbf{x}}_{\cdot j} - \tilde{\mathbf{f}}_{\cdot q'} \right\|^2 \\ 0 & \text{caso contrário} \end{cases} \quad \forall j, q \quad (33)$$
  
 9:     **end for**  
 10:  

$$\tilde{\mathbf{g}}_{p \cdot} \leftarrow (\mathbf{s}_{p \cdot}^{(t+1)} G_{(p)}^{(t)^T})_{pj}, \forall p$$
  
 11:  

$$(F^{(t+1)})_{ip} \leftarrow \begin{cases} 1 & p = \arg \min_{p' \in \{1, \dots, k\}} \left\| \mathbf{x}_{i \cdot} - \tilde{\mathbf{g}}_{p' \cdot} \right\|^2 \\ 0 & \text{caso contrário} \end{cases} \quad \forall i, p \quad (34)$$
  
 12:      $t \leftarrow t + 1$   
 13:     **end while**  
 14:     **return**  $F^{(t+1)}, S^{(t+1)}, G_{(1)}^{(t+1)}, \dots, G_{(k)}^{(t+1)}$   
 15: **end function**


---

## 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*.

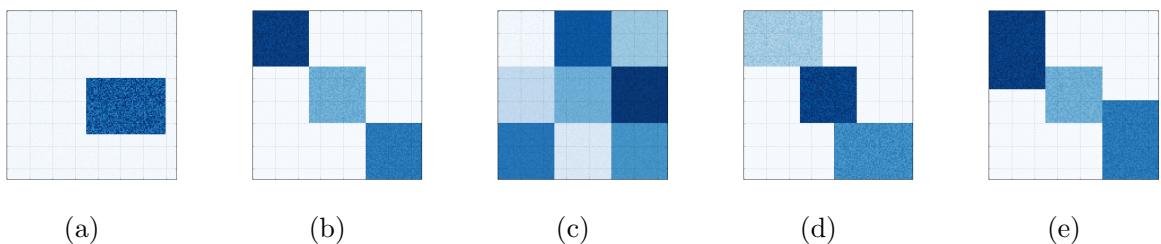
Para proporcionar uma visão organizada das capacidades e limitações dos algoritmos, primeiramente são apresentados os experimentos e os resultados obtidos com as bases de dados sintéticas. Tais resultados são apresentados em termos de qualidade de *reconstrução*, a qual é analisada com apoio de visualização 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 xxxx*. Então, a qualidade de *particionamento e de coagrupamento* é avaliada 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 5 são apresentadas as visualizações gráficas de cada uma das estruturas de cogrupos em estudo.

Figura 5 – 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.



Fonte: Lucas Fernandes Brunialti, 2016

Para compreender a representação gráfica apresentada na figura 5 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 5 são **matrizes de valores reais positivos** geradas artificialmente. Primeiramente, uma matriz de tamanho  $150 \times 150$  é preenchida com valores ponto flutuante, gerados aleatoriamente a partir de uma distribuição uniforme

$unif(0, 0, 1, 0)$ , no intervalo  $[0, 0, 1, 0]$ <sup>1</sup>. 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 pertencentes 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 5, alguns detalhes precisam ser observados. A depender do objetivo da análise de coagruamento, na figura 5a, 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 5a 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  $[140 .. 149]$ .

---

<sup>1</sup> 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 principal interesse e oito cogrupos não destacados na figura;
- (e): três cogrupos de principal interesse e oito cogrupos não destacados na figura.

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

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

---

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

<sup>3</sup> Para experimentos com *fuzzy k-means* foi usado a implementação do algoritmo *fuzzy k-means* da biblioteca *scikit-fuzzy* da linguagem Python.

<sup>4</sup> 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 análises 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)
<i>k-means</i>	ok	ok	ok	ok*	
<i>fuzzy-k-means</i>	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

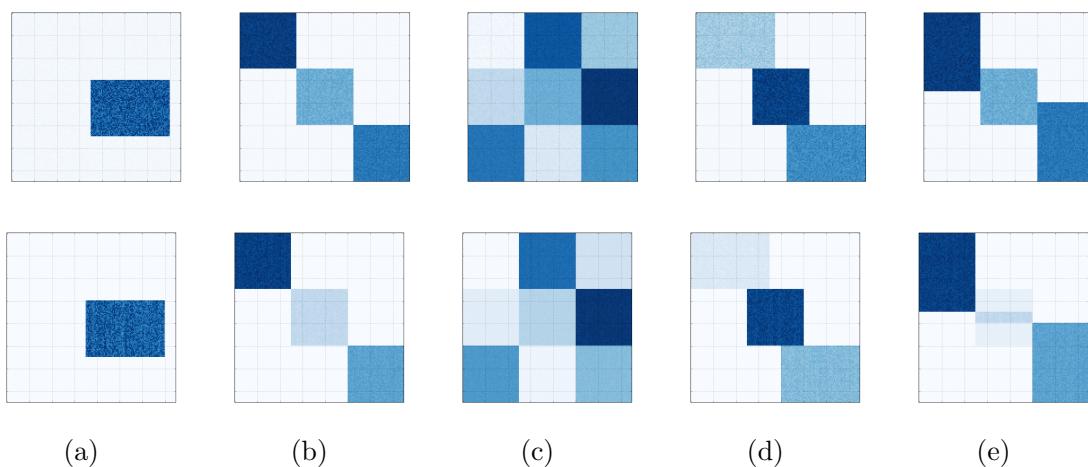
### 5.1.2 Reconstrução a partir dos resultados do algoritmo *k-means*

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 ocorre 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 7. 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 6 – 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*.



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.

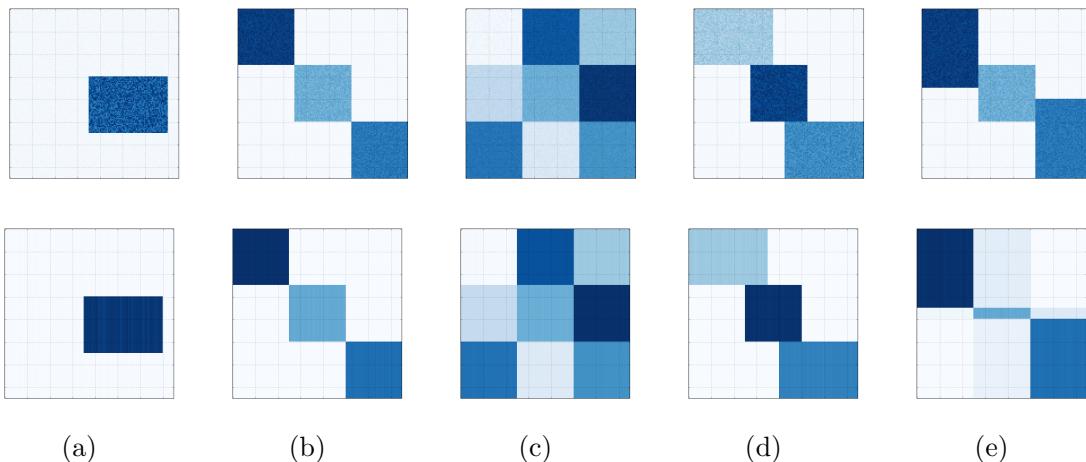
O modelo não permite a descoberta de grupos com interseção de linhas (um dado não pode pertencer a mais de um grupo), e portanto não permite uma boa representação para a reconstrução no caso da base de dados (e). Esse resultado já era esperado devido à natureza da solução apresentada pelo *k-means*. **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 7 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 7 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *fuzzy k-means*.



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.**

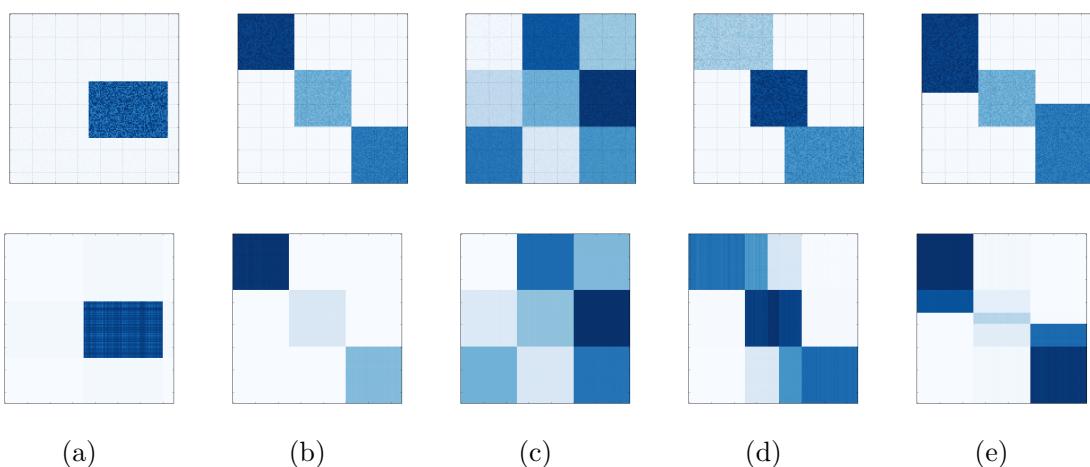
lulu 2792448c20174b8a33ff41ecb8de077f0d58b4c5

#### 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 reconstruções pode ser visualmente observada na figura 8. A reconstrução foi obtida a partir da multiplicação das matrizes fatoradas, ou seja,  $USV^T$ , conforme explicado no capítulo 3.

Figura 8 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *ONMTF*.



Fonte: Lucas Fernandes Brunialti, 2016

Analizando 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.

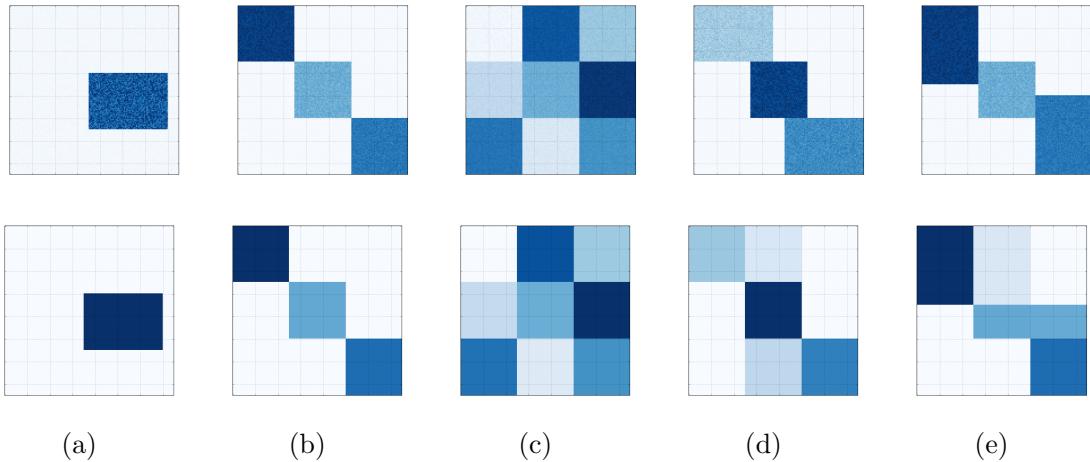
A coloração diferenciada daquela esperada (segundo a matriz original) é resultante do estado da matriz  $V$ , que contém as relações de associação de linhas e/ou colunas aos cogrupos. Para o caso (d), por exemplo, é possível deduzir, a partir da análise visual, que há valores de magnitude diferentes estabelecendo a associação das colunas  $\mathbf{X} \times \mathbf{X}$ ,  $\mathbf{Y} \times \mathbf{Y}$  e  $\mathbf{Z} \times \mathbf{Z}$ , ao grupo de linhas  $\mathbf{X} \times \mathbf{X}$ . 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  $\mathbf{X} \times \mathbf{X}$ .

### 5.1.5 Reconstrução a partir dos resultados do algoritmo ***FNMTF***

Para execução dos experimentos com o FNMTF a seguinte parametrização foi estabelecida: número máximo de iterações (300) ou a diferença do erro de minimização em duas iterações consecutivas ( $\leq 1,0 * 10^{-4}$ ), 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 9. A reconstrução foi obtida a partir da multiplicação das matrizes fatoradas, ou seja,  $FSG^T$ , 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 *FNMTF*.



Fonte: Lucas Fernandes Brunialti, 2016

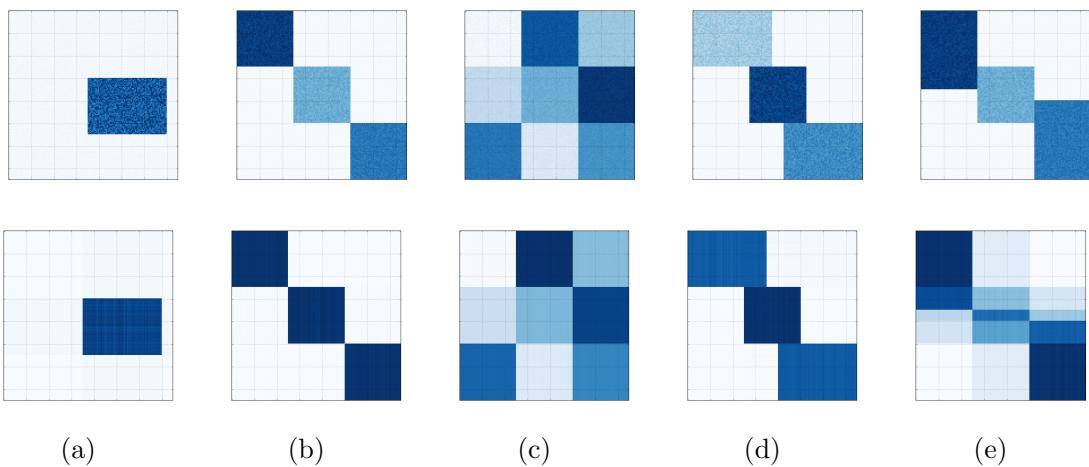
Este caso é semelhante ao anterior (algoritmo *ONMTF*). O algoritmo permitiu a reconstrução com êxito dos casos (a), (b) e (c), falhando na reconstrução dos casos (d) e (e), nos quais há intersecção de colunas ou linhas nos cogrupos de interesse. No entanto, nesse caso o algoritmo restringe a associação 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 10. 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 10 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *OvNMTF*.



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 cogerupos com intersecção de colunas, e portanto é capaz de resolver o caso (d).

Como o algoritmo não é capaz de lidar com cogerupos 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.

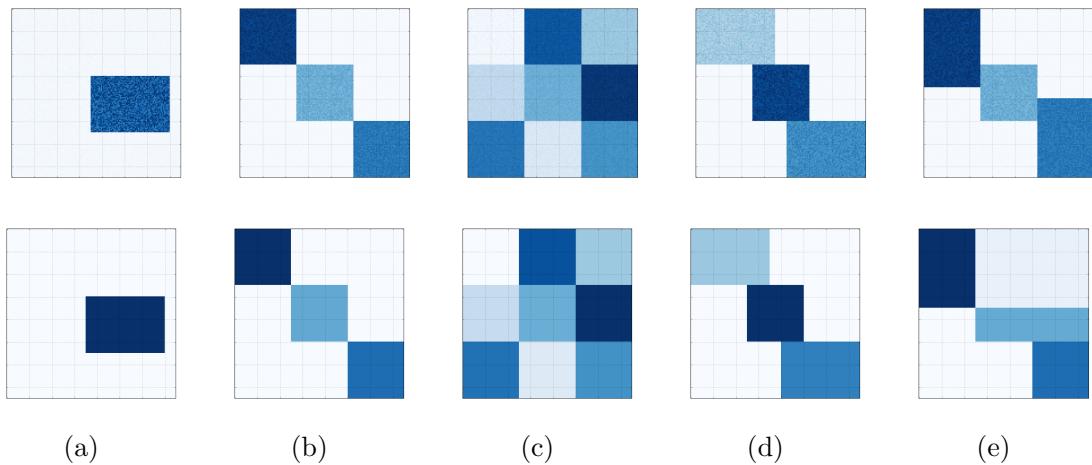
### 5.1.7 Reconstrução a partir dos resultados do algoritmo *BinOvNMTF*

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 (e), 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 11. 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 11 – As primeiras cinco matrizes são as matrizes originais, as demais são suas respectivas reconstruções, realizadas a partir dos resultados obtidos com o algoritmo *BinOvNMTF*.



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, ele 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 par 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 cogrupo 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)
<i>k-means</i>	l	1,0	1,0	1,0	1,0	<b>0,78</b>
	c	–	–	–	–	–
<i>fuzzy-k-means</i>	l	1,0	1,0	1,0	1,0	<b>0,78</b>
	c	–	–	–	–	–
ONMNTF	l	1,0	1,0	1,0	1,0	<b>0,78</b>
	c	1,0	1,0	1,0	<b>0,78</b>	1,0
FNMTF	l	1,0	1,0	1,0	1,0	<b>0,53</b>
	c	1,0	1,0	1,0	<b>0,78</b>	1,0
OvNMTF	l	1,0	1,0	1,0	1,0	<b>0,78</b>
	c	1,0	1,0	1,0	<b>1,0</b>	1,0
BinOvNMTF	l	1,0	1,0	1,0	1,0	<b>0,53</b>
	c	1,0	1,0	1,0	<b>1,0</b>	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**<sup>5</sup> 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**<sup>6</sup> 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*<sup>7</sup>, 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

<sup>5</sup> <http://qwone.com/~jason/20Newsgroups/>

<sup>6</sup> <http://robotics.stanford.edu/~gal/data.html>

<sup>7</sup> <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<sup>8</sup>. 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
<i>20Newsgroup</i>	12.998	18.221	20	
<i>NIPS14-17</i>	17.583	420	9	
<i>IG</i>	19.563	4.593	13	
<i>IG toy</i>	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:

---

<sup>8</sup> <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  $D = \{d_1, \dots, d_i, \dots d_n\}$ , e um conjunto de  $m$  termos  $\{t_1, \dots, t_j, \dots t_m\}$ , 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  $\mathbf{v}_{d_i} = [tf(t_1, d_i), \dots, tf(t_j, d_i), \dots, tf(t_m, d_i)]$ . Contudo, Salton, Wong e Yang (1975) mostram a partir de experimentação em diversos conjuntos de dados textuais, que o uso da relação conhecida como Frequênciade Termos-Frequênciade 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 35](#), 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.

$$\begin{aligned} tfidf(t_j, d_i) &= tf(t_j, d_i) \cdot \left( \log_2 \frac{n}{df(t_j)+1} \right) \\ tfidf(t_j, d_i) &= tf(t_j, d_i) \cdot IDF(t_j) \end{aligned} \quad (35)$$

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  $tf\_idf(t_j, d_i)$ , obtém-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	$l$	<i>Rand Index</i>
<i>k-means</i>	TF	-	0.0000
<i>k-means</i>	TF norm	-	0.3350
<i>k-means</i>	TF-IDF	-	0.0000
<i>k-means</i>	TF-IDF norm	-	0.2262
<i>fuzzy k-means</i>	TF	-	0.0000
<i>fuzzy k-means</i>	TF norm	-	0.1757
<i>fuzzy k-means</i>	TF-IDF	-	0.0000
<i>fuzzy k-means</i>	TF-IDF norm	-	0.3832
<i>ONMTF</i>	TF	19	0.1883
<i>ONMTF</i>	TF norm	13	0.2023
<i>ONMTF</i>	TF-IDF	13	0.1814*
<i>ONMTF</i>	TF-IDF norm	7	0.2037
<i>FNMTF</i>	TF	-	0.0000
<i>FNMTF</i>	TF norm	19	0.2678
<i>FNMTF</i>	TF-IDF	-	0.0000
<i>FNMTF</i>	TF-IDF norm	16	0.2480
<i>OvNMTF</i>	TF	-	0.0000
<i>OvNMTF</i>	TF norm	-	0.0000
<i>OvNMTF</i>	TF-IDF	-	0.0000
<i>OvNMTF</i>	TF-IDF norm	-	0.0000
<i>BinOvNMTF</i>	TF	16	0.4415
<i>BinOvNMTF</i>	TF norm	7	0.4081*
<i>BinOvNMTF</i>	TF-IDF	7	0.3681*
<i>BinOvNMTF</i>	TF-IDF norm	10	0.3261*

#### 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	$l$	<i>Rand Index</i>
<i>k-means</i>	TF	-	0.0002
<i>k-means</i>	TF norm	-	0.6496
<i>k-means</i>	TF-IDF	-	0.0004
<i>k-means</i>	TF-IDF norm	-	0.3381
<i>fuzzy k-means</i>	TF	-	0.0044
<i>fuzzy k-means</i>	TF norm	-	0.0043
<i>fuzzy k-means</i>	TF-IDF	-	0.0413
<i>fuzzy k-means</i>	TF-IDF norm	-	0.0778
<i>ONMTF</i>	TF	6	0.3910
<i>ONMTF</i>	TF norm	5	0.7485
<i>ONMTF</i>	TF-IDF	5	0.4099
<i>ONMTF</i>	TF-IDF norm	3	0.7098
<i>FNMTF</i>	TF	2	0.0383
<i>FNMTF</i>	TF norm	3	0.3810
<i>FNMTF</i>	TF-IDF	5	0.0387
<i>FNMTF</i>	TF-IDF norm	5	0.2966
<i>OvNMTF</i>	TF	-	0.0000
<i>OvNMTF</i>	TF norm	-	0.0000
<i>OvNMTF</i>	TF-IDF	-	0.0000
<i>OvNMTF</i>	TF-IDF norm	-	0.0000
<i>BinOvNMTF</i>	TF	3	0.0509
<i>BinOvNMTF</i>	TF norm	5	0.7197
<i>BinOvNMTF</i>	TF-IDF	5	0.2263
<i>BinOvNMTF</i>	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<sup>9</sup> 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 existirem 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

---

<sup>9</sup> <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 está presente em cada um dos coclusters com um fator associado, assim como cada palavra está 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 12 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 12, note que as palavras do corpo da notícia foram coloridas de acordo com as cores dos coclusters de palavras às 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.

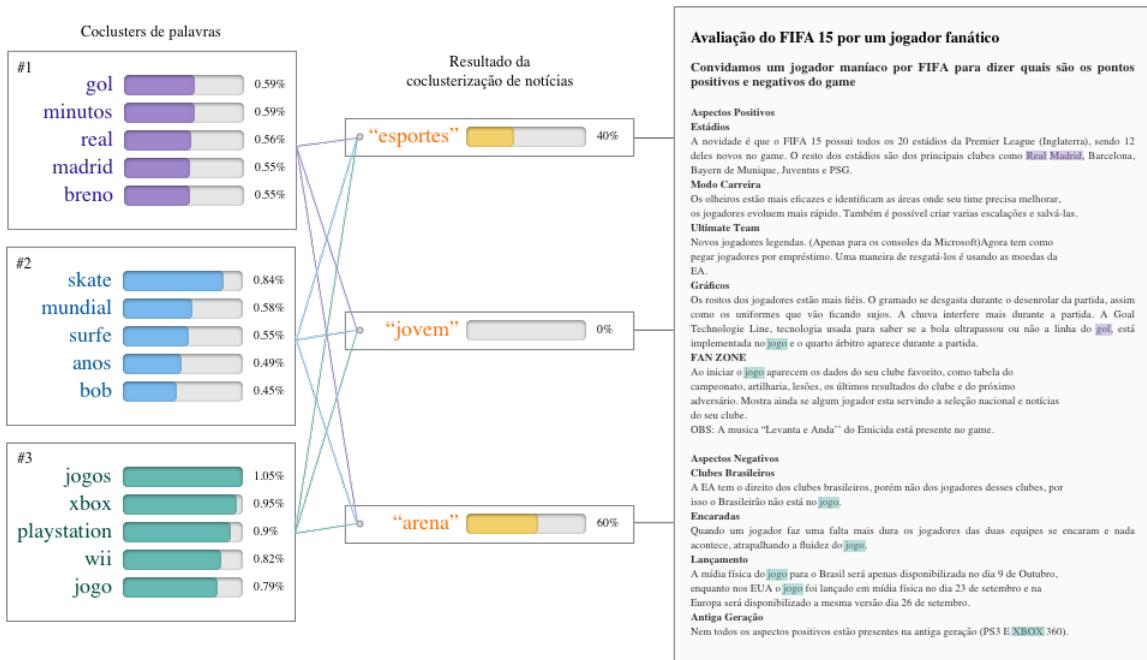


Figura 12 – Exemplo de uma notícia do canal arena e sua disposição quanto aos coclusters de notícias e palavras.

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	<b>0.9744</b>
Cocluster de notícias “esporte”	<b>0.9604</b>	0.0348	0.0048
Cocluster de notícias “jovem”	0.0444	<b>0.9324</b>	0.0232

Os grupos de palavras foram summarizados através da visualização em *word cloud*, em que o tamanho das palavras é definido pelo seu fator de pertinência ao cocluster correspondente. A Figura 13 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 13 – Visualização *word cloud* de palavras para cada cocluster de palavras gerados pelo algoritmo *ONMTF*.

A Figura 13 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.

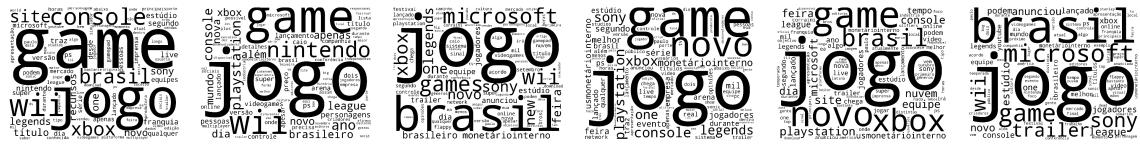
#### 5.2.4.2 Análise de dados utilizando *FNMTF*

#### 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	<b>0.2516</b>	0.0535	<b>0.3294</b>	0.1499	0.0691
Cocluster de notícias “esporte”	0.0949	0.0943	<b>0.2239</b>	0.1287	<b>0.2878</b>	0.1703
Cocluster de notícias “jovem”	0.0381	<b>0.2545</b>	<b>0.3324</b>	<b>0.2220</b>	0.0932	0.0597



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

Figura 14 – 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 15 – 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 16 – 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<sup>10</sup>

- 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 71.
- BAZARAA, M.; SHERALI, H.; SHETTY, C. *Nonlinear Programming: Theory and Algorithms*. [S.l.]: Wiley, 2006. ISBN 9780471787761. Citado na página 34.
- 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.
- DING, C. et al. Orthogonal nonnegative matrix tri-factorizations for clustering. In: *In SIGKDD*. [S.l.]: Press, 2006. p. 126–135. Citado 2 vezes nas páginas 34 e 35.
- EDELMAN, A.; ARIAS, T. A.; SMITH, S. T. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, v. 20, n. 2, p. 303–353, abr. 1999. ISSN 0895-4798. Disponível em: <<http://dx.doi.org/10.1137/S0895479895290954>>. Citado na página 38.
- 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 76.
- 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 61.
- 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 78.
- HAYKIN, S. *Neural Networks and Learning Machines (3rd Edition)*. 3. ed. [S.l.]: Prentice Hall, 2008. Hardcover. Citado na página 78.

<sup>10</sup> De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

HO, N.-D. *Nonnegative Matriz Factorization Algorithms and Applications*. Tese (Doutorado) — Université Catholique de Louvain, Louvain-la-Neuve, Belgique, 6 2008. Citado na página 19.

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 76.

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 71.

LEE, D. D.; SEUNG, H. S. Learning the parts of objects by nonnegative matrix factorization. *Nature*, v. 401, p. 788–791, 1999. Citado na página 20.

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.

LONG, B.; ZHANG, Z. M.; YU, P. S. *Co-clustering by block value decomposition*. [S.l.]: ACM Press, 2005. 635–640 p. Citado 3 vezes nas páginas 30, 31 e 34.

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 63, 71, 76 e 77.

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 46.

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 78.

MURPHY, K. P. *Machine Learning: A Probabilistic Perspective*. [S.l.]: The MIT Press, 2012. Citado na página 78.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 49.

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 58.

- 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 52.
- 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 63, 64, 76 e 77.
- 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 63, 76 e 77.
- 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 67.
- 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 na página 39.
- WEISS, S. M.; INDURKHYA, N.; ZHANG, T. *Fundamentals of predictive text mining*. London; New York: Springer-Verlag, 2010. Citado na página 77.
- 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 Information 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 Processing and Management*, v. 46, p. 559–570, 2010. Citado 8 vezes nas páginas [9](#), [20](#), [32](#), [36](#), [37](#), [38](#), [41](#) e [43](#).

## 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  $D = \{d_1, d_2, \dots, d_n\}$ , em que  $d_i$  representa um documento e  $n$  o número total de documentos, e um conjunto de termos  $\mathcal{T} = \{t_1, t_2, \dots, t_m\}$ , 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:  $\vec{vt}_{d_i} = (TF(t_1, d_i), TF(t_2, d_i), \dots, TF(t_m, d_i))$ . 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:

$$\begin{aligned} TF-IDF(t_j, d_i) &= TF(t_j, d_i) \cdot IDF(t_j) \\ TF-IDF(t_j, d_i) &= TF(t_j, d_i) \cdot \left( \log_2 \frac{n}{DF(t_j) + 1} \right) \end{aligned} \quad (36)$$

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*.