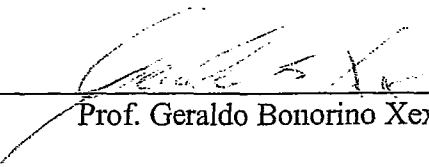
AGRUPAMENTO SEMI-SUPERVISIONADO DE DOCUMENTOS XML

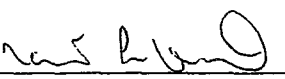
Eduardo Bezerra da Silva

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

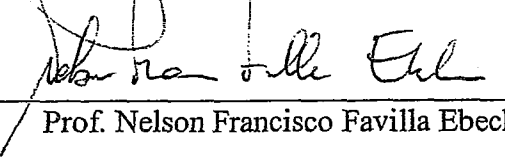


Prof. Geraldo Bonorino Xexéo, D.Sc.

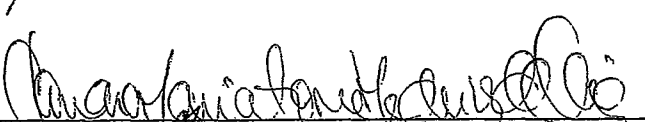


Profª. Marta Lima de Queirós Mattoso, D.Sc.

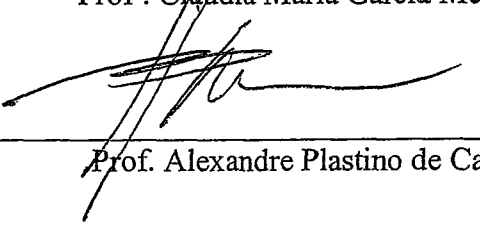
Prof. Jano Moreira de Souza, Ph.D.



Prof. Nelson Francisco Favilla Ebecken, Ph.D.



Profª. Claudia Maria Garcia Medeiros de Oliveira, Ph.D.



Prof. Alexandre Plastino de Carvalho, D.Sc.

RIO DE JANEIRO, RJ - BRASIL

JANEIRO DE 2006

SILVA, EDUARDO BEZERRA DA

Agrupamento Semi-Supervisionado de Documentos XML, [Rio de Janeiro] 2006

XIII, 115 p. 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia de Sistemas e Computação, 2006)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Análise de grupos; Mineração de dados textuais

I. COPPE/UFRJ II. Título (série)

Dedicatória

O ovo é a alma da galinha. A galinha desajeitada. O ovo certo. A galinha assustada. O ovo certo. Como um projétil parado. Pois ovo é ovo no espaço. Ovo sobre azul. Eu te amo, ovo. Eu te amo como uma coisa nem sequer sabe que ama outra coisa.

O Ovo e a Galinha, Clarice Lispector

Dedicado à memória de minha mãe e melhor amiga, Maria Bezerra da Silva

Agradecimentos

Me considero um privilegiado por ter tido dois orientadores tão competentes e, ao mesmo tempo, tão diferentes. Durante todo tempo, eles estavam sempre dispostos a me ajudar com o melhor que tinham. Sou muito agradecido a ambos pela paciência, pelo incentivo e por acreditarem em mim. Levarei para o resto de minha vida as lições de competência e empenho no trabalho científico da profa. Marta. Ao prof. Xexéo, meu eterno orientador (exceto por dez minutos...), obrigado por toda a ajuda e pelas perguntas intrigantes, que sempre me impulsionaram a realizar um trabalho de melhor qualidade.

Devo agradecimentos também aos demais professores da COPPE. Em particular, aos professores Jano de Souza, Gerson Zaverucha, Nelson Ebecken, Cláudio Esperança, Adilson Xavier e Inês Dutra que, de uma forma ou de outra, ajudaram no desenvolvimento desta tese. Às meninas da secretaria, Cláudia, Solange, Suely e Lúcia, obrigado pela competência e por toda a ajuda. Um obrigado especial à Patrícia e à Carol, secretárias da linha de Bancos de Dados, por todo auxílio prestado e pela amizade.

Agradeço ao CNPq e, em última análise, ao povo do Brasil, por patrocinarem o desenvolvimento desta tese, através de incentivo financeiro.

Aos meus amigos de tantos anos, Flavio Tavares, Jorge Soares, Alexandre Lima, Ricardo Choren e Gustavo Hajdu, agradeço pelos bons momentos e por compartilharem sua amizade comigo.

O desenvolvimento de uma tese é um trabalho que requer certa dose de reclusão do autor. Devo agradecimentos, e principalmente desculpas, a todos os meus amigos e à minha família, por estar ausente durante tanto tempo. Aos meus irmãos, Helton, Emanuel, Edmar e José Maria, obrigado por me compreenderem e sempre estarem ao meu lado. Pai, nunca terei como recompensar toda a dedicação e ajuda que o senhor me prestou.

Aline, você sabe o quanto me ajudou durante todo esse tempo. Seu incentivo nos momentos difíceis, e amor em todos os momentos, foram fundamentais para que eu chegasse ao final desse trabalho.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

AGRUPAMENTO SEMI-SUPERVISIONADO DE DOCUMENTOS XML

Eduardo Bezerra da Silva

Janeiro/2006

Orientadores: Geraldo Bonorino Xexéo
Marta Lima de Queirós Mattoso

Programa: Engenharia de Sistemas e Computação

Na medida em que provedores de informações podem expressá-las de maneiras mais complexas e contextualizadas, documentos semi-estruturados estão se tornando importantes em áreas como bibliotecas digitais, comércio eletrônico e aplicações Web. Uma característica dessas coleções de documentos é a existência de alguma estrutura ou metadado. Nesse cenário, métodos de agrupamento que possam aproveitar essa informação estrutural para melhor organizar tais coleções são altamente relevantes. Contudo, documentos semi-estruturados apresentam novos desafios para métodos de agrupamento, pois não é claro como informação estrutural pode ser utilizada para melhorar os resultados do agrupamento. Por outro lado, recentemente tem havido um crescente interesse na tarefa de agrupamento semi-supervisionado, na qual uma pequena quantidade de conhecimento prévio é fornecida para guiar o algoritmo para um melhor resultado. Um tipo particular de semi-supervisão são restrições (fornecidas pelo usuário) sobre objetos a serem agrupados. Cada restrição define se dois objetos devem permanecer no mesmo ou em diferentes grupos. Nesta tese, consideramos o problema de agrupamento baseado em restrições sobre documentos semi-estruturados. Consideramos a existência de uma forma particular de informação estrutural: documentos textuais em formato XML. Definimos e estendemos métodos para melhorar a qualidade dos resultados de agrupamento através do uso de informação estrutural para guiar o algoritmo. Resultados experimentais em coleções de documentos do mundo real mostram que nossa abordagem é efetiva.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

SEMI-SUPERVISED CLUSTERING OF XML DOCUMENTS

Eduardo Bezerra da Silva

January/2006

Advisors: Geraldo Bonorino Xexéo

Marta Lima de Queirós Mattoso

Department: Computing and Systems Engineering

As providers of information can express it in more contextualized and complex ways, semi-structured documents are becoming important in many areas, e.g., in digital libraries, e-commerce or Web applications. A particular characteristic of such document collections is the existence of some structure or metadata along with the data. In this scenario, clustering methods that can take advantage of such structural information to better organize such collections are highly relevant. Semi-structured documents pose new challenges to clustering, however, since it is not clear how this structural information can be used to improve the quality of the generated clustering models. On the other hand, recently there has a growing interest in the semi-supervised clustering task, in which a little amount of prior knowledge is provided to guide the algorithm to a better clustering model. A particular type of semi-supervision is in the form of user-provided constraints defined over pairs of objects, where each pair informs if its objects must be in the same or in different clusters. In this thesis, we consider the problem of constrained clustering in documents that present some form of structural information. We consider the existence of a particular form of information to be clustered: textual documents that present a logical structure represented in XML format. We define and extend methods to improve the quality of clustering results by using such structural information to guide the execution of the constrained clustering algorithm. Experimental results on real-world document collections show the effectiveness of our approach.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

...we came in?

Pink Floyd, The Wall (1979)

1.1 Introdução

A análise de grupos (ou agrupamento de dados) pode ser definida como o problema de agrupar um conjunto de objetos em dois ou mais segmentos (ou grupos), com base em alguma medida de proximidade ou de semelhança entre esses pontos. Dá-se o nome de *modelo de agrupamento*, ou simplesmente *agrupamento*, ao conjunto de grupos formados a partir dos objetos de uma coleção. Um bom agrupamento exibe a característica de que objetos associados ao mesmo grupo são bastante similares, ao mesmo tempo em que objetos associados a grupos diferentes exibem uma baixa similaridade. Aplicações diretas da análise de grupos incluem segmentação de clientes ou de produtos, agrupamento de genes em um experimento de micro-array [19], organização dos resultados de uma consulta enviada a um mecanismo de busca da WEB [72], etc. Além de possuir aplicações diretas, a análise de grupos também pode ser vista como uma etapa fundamental na análise de dados, onde os dados são agrupados para um posterior processamento. Embora de fácil definição, o problema da análise de grupos é de difícil solução. Uma das principais dificuldades relacionadas

à análise de grupos é definir corretamente o que significa *similaridade* entre os objetos envolvidos. Esse problema tem atraído a atenção de diversas comunidades da área científica, entre elas a ciência da computação, por aproximadamente quarenta anos. Conseqüentemente, existe uma quantidade imensa de soluções propostas na literatura para problemas de agrupamento.

Um algoritmo típico de agrupamento se encaixa na categoria de *aprendizado não-supervisionado*. Isso significa que o algoritmo não utiliza outra informação que não os próprios objetos para gerar o modelo de agrupamento. Sem perda de generalidade, considere uma classe particular de objetos, a dos documentos textuais que representam artigos científicos. Quando aplicados a esses objetos, algoritmos de agrupamento tradicionais não consideram a existência de restrições e relacionamentos na organização subjacente desses documentos, tais como o fato de um autor ter grande parcela de suas publicações em co-autoria com outro autor, ou o fato de dois ou mais descritores (palavras chave) utilizados para anotar documentos da coleção terem grande correlação entre si.

No entanto, nos últimos anos, uma parcela significativa de propostas de métodos para agrupamento sugere a incorporação de informação externa ou conhecimento prévio (*prior knowledge*) ao processo. Abordagens desse tipo são chamadas de *agrupamento semi-supervisionado*. Trabalhos nessa área tentam responder à seguinte pergunta: de que forma o resultado de um algoritmo de agrupamento pode ser melhorado através do uso de informação externa (na forma de restrições e relacionamentos entre os objetos a serem agrupados)? A busca de respostas a essa questão tem ocasionado o surgimento de diversos trabalhos na literatura que se encaixam na categoria de *métodos de agrupamento semi-supervisionado* [42, 45, 70, 18, 9, 68, 2, 8].

A abordagem geral dos trabalhos em agrupamento semi-supervisionado é utilizar informação fornecida pelo usuário acerca dos objetos a serem agrupados para guiar o processo de agrupamento. Um exemplo de informação externa considerada nesses trabalhos são restrições da forma $G(o_i) = G(o_j)$ ou $G(o_i) \neq G(o_j)$, denotando respectivamente que dois objetos o_i e o_j devem pertencer ao mesmo grupo (embora a restrição não identifique *qual* é o grupo), e que dois objetos devem per-

tencer a grupos diferentes. Dessa forma, o algoritmo recebe um conjunto não vazio de exemplos que correspondem a restrições entre pares de objetos (o_i, o_j). Através dessas restrições sobre os objetos fornecidas como exemplos, o método de agrupamento semi-supervisionado pode, por exemplo, alterar a forma com que calcula a similaridade entre os demais objetos da coleção, ou modificar a expressão da função objetivo do algoritmo. A conseqüência disso é que o resultado final da configuração de grupos gerada é alterado.

Note que a abordagem de agrupamento semi-supervisionado é perfeitamente aplicável a objetos que correspondem a *documentos textuais*: a um algoritmo de agrupamento semi-supervisionado podem ser fornecidos um conjunto $\mathcal{D}$ de documentos e um conjunto $\mathcal{R}$ de restrições, onde cada elemento de $\mathcal{R}$ é da forma $G(d_i) = G(d_j)$ ou da forma $G(d_i) \neq G(d_j)$, d_i e d_j, $i \neq j$, elementos de $\mathcal{D}$. Consideramos nessa tese a classe particular de objetos referente a documentos textuais.

1.2 Motivação

Os trabalhos na área de agrupamento semi-supervisionado têm mostrado resultados de sucesso, o que justifica o intenso interesse por essa área de pesquisa [1, 5, 3, 6, 7, 4, 67, 68, 18, 9, 64, 27, 28, 45]. A vantagem dessa abordagem em relação à estratégia convencional (puramente não supervisionada) diz respeito à possibilidade de o resultado do agrupamento refletir as necessidades do usuário em relação à formação dos grupos. Através do conhecimento prévio sobre os documentos fornecido pelo usuário, o algoritmo de agrupamento semi-supervisionado pode direcionar a produção dos grupos de tal forma que o resultado reflita esse conhecimento prévio. Os principais algoritmos de agrupamento semi-supervisionado atualmente existentes correspondem a extensões do algoritmo K-Means clássico [50]. Três dos principais algoritmos que adotam essa estratégia são *PCKMeans*, o *MPCKMeans* [6] e o *COP-KMeans* [68]. Resultados experimentais obtidos com esses algoritmos mostraram que a qualidade dos grupos formados aumenta em função da quantidade de informação externa fornecida pelo usuário. No entanto, nesses trabalhos

sobre agrupamento semi-supervisionado, há alguns problemas em aberto, que são discutidos a seguir [12].

1. Uma limitação que identificamos nos atuais algoritmos de agrupamento semi-supervisionado diz respeito à rigidez na definição de restrições. Esses algoritmos consideram que as restrições podem ser definidas apenas em relação aos objetos que devem ser agrupados. No entanto, se supomos a existência de metadados sobre esses objetos, como é o caso para documentos em formato XML (*Extensible Markup Language*), é desejável que haja a possibilidade de o usuário definir restrições também em relação a esses metadados. Essa flexibilidade potencialmente aumenta a força representativa do usuário com respeito à natureza dos grupos que ele deseja que o algoritmo forme.

2. Outra dificuldade encontrada na tarefa de agrupamento semi-supervisionado diz respeito à *quantidade* de restrições que o usuário deve fornecer. Métodos de agrupamento semi-supervisionado atualmente existentes supõem que o usuário deve fornecer uma quantidade significativa de informação externa para que o algoritmo consiga alcançar resultados satisfatórios em comparação à versão não-supervisionada. Há diversos domínios em que essa suposição se aplica, particularmente em coleções onde o número de características (atributos) de cada objeto é pequeno. Entretanto, em domínios de aplicação complexos como o agrupamento de documentos, é necessário que o usuário forneça uma quantidade de restrições que pode chegar à casa das centenas. Em situações práticas, o usuário não tem como fornecer (ou não quer fornecer) mais do que algumas dezenas de restrições entre objetos. Portanto, um primeiro problema em aberto na área de agrupamento semi-supervisionado é definir estratégias para diminuir a quantidade de informação externa fornecida pelo usuário e, ao mesmo tempo, manter a qualidade do agrupamento gerado.

3. Outro problema com os algoritmos de agrupamento semi-supervisionado atualmente existentes é que eles consideram que as restrições entre os objetos a serem agrupados têm igual importância. Novamente, isso não reflete a re-

alidade. Um sistema que implemente um algoritmo de agrupamento semi-supervisionado deve permitir ao usuário a definição dos graus de importância de cada restrição que ele fornece, no pior caso; no melhor caso, o sistema pode auxiliar o usuário nessa definição, ou mesmo sugerir graus de importância para as restrições fornecidas.

Como agravante, a proporção de coleções de documentos aos quais está associada alguma informação estrutural ou metadado, particularmente no formato XML [14], está crescendo rapidamente em diversas áreas de aplicação. Na WEB, por exemplo, a idéia da *WEB Semântica*, de que a Internet possa ser "navegada" automaticamente por agentes inteligentes, pressupõe que os dados nesse ambiente estejam representados em XML. Outro domínio de aplicação onde documentos com alguma estrutura são cada vez mais dominantes é o da *Bioinformática*. Bancos de dados imensos de seqüências de moléculas (proteínas e aminoácidos), assim como repositórios de coleções de artigos dessa área, contêm anotações textuais ou informação estrutural representadas através da utilização do formato XML. Não raro, essas coleções também apresentam metadados associados na forma de ontologias ou de taxonomias. Além disso, diversas coleções pré-existentes de documentos textuais estão sendo convertidas para XML [20]. Se a tarefa de agrupamento tiver que ser utilizada para organizar repositórios de documentos XML, a possibilidade de aproveitamento de informação adicional na forma de metadado para guiar ou restringir o processo de agrupamento não pode ser desprezada. Sendo assim, o problema de incorporar informação proveniente de metadados no processo de agrupamento é altamente relevante.

1.3 Contribuições

A nossa proposta de solução para os problemas levantados anteriormente considera que os objetos a serem agrupados são documentos em formato XML. Também tomamos como ponto de partida os algoritmos de agrupamento semi-supervisionado baseados no *K-Means* e mencionados anteriormente. Esta tese se encaixa então

no contexto de métodos semi-supervisionados de agrupamento sobre documentos XML, onde abordamos o seguinte problema: de que forma o usuário pode aproveitar a existência de algum metadado ou informação estrutural sobre a coleção de documentos a serem agrupados para definir restrições a serem consideradas durante o agrupamento desses documentos? Esta tese apresenta as seguintes contribuições:

1. Com o objetivo de aumentar a flexibilidade e força representativa do usuário, estendemos a possibilidade deste definir restrições sobre os documentos a serem agrupados. Fazemos isso através da definição de uma representação de documentos XML que permite que o usuário faça referência a informações estruturais desses documentos. O resultado é que o usuário pode definir restrições sobre elementos dos metadados da coleção, além de poder defini-las sobre identificadores de documentos (como já é feito nos métodos atualmente existentes).

2. Definimos uma forma de utilizar informação estrutural sobre a coleção de documentos XML para inferir um conjunto $\mathcal{S}$ de restrições a partir de um conjunto $\mathcal{U}$ de restrições fornecido pelo usuário. Em vez de passar $\mathcal{U}$ como entrada para o algoritmo de agrupamento, passamos o conjunto inferido $\mathcal{S}$. Como conseqüência, ocorre uma diminuição da sobrecarga sobre este usuário, em relação à quantidade de restrições que ele deve definir para que o algoritmo alcance uma qualidade satisfatória. Como nosso procedimento de inferência de restrições tem o potencial de gerar inconsistências, definimos também algoritmos para eliminação das mesmas.

3. Também utilizamos essa informação sobre a estrutura dos documentos para sugerir ao usuário graus de importância para as restrições a serem utilizadas durante o algoritmo de agrupamento.

Para validar nossa proposta, implementamos um protótipo de sistema de agrupamento semi-supervisionado utilizando a linguagem Java. Nesse protótipo, incorporamos os diversos algoritmos propostos nesta tese. Com a utilização desse protótipo,

executamos experimentos utilizando diversos subconjuntos da coleção OHSUMED [37]. De acordo com esses experimentos, nossa solução para inferência de um conjuntos de restrições $\mathcal{S}$ a partir do conjunto $\mathcal{U}$ de restrições definidas pelo usuário consegue atingir um fator de expansão significativo, de tal forma que a cardinalidade de $\mathcal{S}$ é, em média, 7 vezes maior que a cardinalidade de $\mathcal{U}$.

Constatamos também uma diminuição consistente na quantidade de iterações necessária para a convergência do nosso algoritmo de agrupamento semi-supervisionado, o *MAC-KMeans*, conforme fornecemos maiores quantidades de restrições como entrada. Como conseqüência, o tempo total de execução do algoritmo tende a permanecer constante, conforme quantidade maiores de restrições são fornecidas como entrada (até uma quantidade limite de restrições), a despeito de nosso algoritmo ter uma complexidade de tempo igual a $O(n^2)$.

1.4 Organização dos Capítulos

Esta tese está organizada como segue. O capítulo 2 apresenta uma visão geral da tarefa de agrupamento de documentos. O capítulo 3 apresenta aspectos relacionados a coleções de documentos textuais representadas em XML; este capítulo também apresenta uma descrição dos trabalhos relacionados aos desta tese. O nosso método de agrupamento semi-supervisionado de documentos é apresentado no capítulo 4. No capítulo 5, são apresentados resultados experimentais para validação do nosso método. O capítulo 6 encerra esta tese, com a apresentação de comentários conclusivos e indicação de perspectivas futuras de continuidade do trabalho.

Capítulo 2

A Tarefa de Agrupamento

One might wonder why anyone is interested in such an unpromising problem and whether or not it is even possible in principle to learn anything of value from unlabelled samples.

Duda, Hart & Stork: Pattern Classification and Scene Analysis (1995)

2.1 Introdução

A tarefa de agrupamento consiste em encontrar grupos de objetos em um espaço de atributos pré-definido. Por definição, não se sabe a priori qual é o grupo a que cada objeto pertence. Por essa razão, a tarefa de agrupamento tradicional é um problema de aprendizado não-supervisionado [24]: a sua aplicação não pressupõe a existência de rótulos (ou classes) associados aos objetos. Nessa tarefa, o objetivo é criar um número finito de partições de uma coleção de dados, tais que objetos dentro de uma mesma partição sejam maximamente similares de acordo com alguma métrica. Geralmente esta métrica é definida de tal forma que as similaridades dentro de um grupo são maximizadas e as similaridades entre grupos são minimizadas. Mais formalmente, a tarefa de agrupamento busca otimizar alguma *função objetivo* (implícita ou explícita) que é inerente aos dados sendo tratados.

Modelos de agrupamento existem em diferentes formas, mas, usualmente, os

padrões existem matematicamente como uma função de probabilidade que associa cada item no conjunto de objetos a algum grupo. Esse capítulo fornece uma visão geral da tarefa de agrupamento. Na Seção 2.2, descrevemos algumas medidas utilizadas para o cálculo da similaridade entre objetos de uma coleção. Na Seção 2.3, descrevemos e categorizamos os principais métodos de agrupamentos existentes. Finalmente, na Seção 2.4 descrevemos estratégias que podem ser utilizadas para medir a qualidade de um algoritmo de agrupamento. Antes de passar às próximas seções, entretanto, considere a notação apresentada a seguir, que utilizamos no restante deste capítulo.

- $\mathcal{X}$: conjunto de objetos a serem processados (agrupados) pelo algoritmo de agrupamento.
- k: quantidade de grupos identificada pelo algoritmo de agrupamento (ou fornecida como parâmetro de entrada).
- G: conjunto de grupos identificados por um algoritmo de agrupamento. Chamamos G de uma *k-partição* de $\mathcal{X}$. Note que $k = |G|$.
- $\{G_1, G_1, \ldots, G_k\}$: os k grupos de objetos identificados por um algoritmo de agrupamento. Ou seja $G = \{G_1, G_1, \ldots, G_k\}$.
- x_i: cada um dos $|\mathcal{X}|$ objetos sobre os quais o algoritmo de agrupamento é executado.

2.2 Medidas para Cálculo de Similaridade

A qualidade do resultado de um algoritmo de agrupamento depende, dentre outros fatores, de ser selecionada uma boa medida para calcular a similaridade (ou *distância*, se interpretarmos do ponto de vista geométrico) entre os objetos a serem agrupados. Nesta seção, são apresentadas algumas das diversas medidas utilizadas para calcular a similaridade entre dois objetos. Para uma descrição de outras medi-

das, o leitor é direcionado à referência [43]. Ou então às referências [15, 58] para o caso particular em que os objetos são documentos.

Normalmente as medidas de distância utilizadas no agrupamento de objetos correspondem a *métricas*, ou seja, essas medidas são positivas, simétricas e obedecem à desigualdade triangular [22, 13]. Podemos dividir essas medidas em dois tipos, de acordo com a natureza dos valores dos atributos dos objetos: *medidas de similaridade para dados numéricos* e *medidas de similaridade para dados categorizados*. As seções 2.2.1 e 2.2.2 descrevem esses dois tipos de medidas.

2.2.1 Medidas de Similaridade para Dados Numéricos

Há diversas medidas propostas para determinar a similaridade (ou distância) entre dois objetos quando as características desses objetos possuem domínio contínuo. De uma forma geral, essas medidas consideram cada objeto x como um vetor no espaço n-dimensional, onde n corresponde à quantidade de características de cada objeto x. De acordo com essa interpretação, medidas geométricas de distâncias podem ser utilizadas para determinar a similaridade entre os objetos a serem a grupados. O restante desta seção descreve medidas para cálculo de similaridade relevantes para esta tese: *distâncias de Minkowski*, *similaridade por co-seno* e *distâncias de Mahalanobis*.

Distâncias de Minkowski. As denominadas distâncias de Minkowski são medidas bastante utilizadas em problemas geométricos. A forma geral dessa família de distâncias é a seguinte:

$$L_p(x_a, x_b) = (\sum_{i=1}^{d} |x_{ia} - x_{ib}|^p)^{\frac{1}{p}} \tag{2.1}$$

Para o valor de p = 1, tem-se a *distância Manhattan*. Para o valor de $p = 2$, tem-se a *distância Euclidiana*.

De uma forma geral, medidas baseadas nas distâncias de Minkowski não se aplicam adequadamente a algumas aplicações da tarefa de agrupamento, particular-

mente quando a dimensionalidade[1]Denomina-se dimensionalidade de um item de dado à quantidade de atributos que são utilizados para descrever esse item.) dos dados é grande. Essa deficiência das medidas baseadas em distância é discutida em [60]. Por outro lado, essa característica (grande dimensionalidade) é intrínseca em dados textuais. De fato, estudos empíricos na área de Recuperação da Informação revelaram que *medidas direcionais* (ou seja, medidas baseadas somente na direção dos vetores que representam documentos) são mais adequadas para dados textuais. (Por essa razão, dados textuais são comumente chamado de *dados direcionais*.) Uma dessas medidas é a similaridade por co-seno, descrita a seguir.

Similaridade por co-seno. Para dados textuais, há um padrão de fato para medidas de similaridade entre dois documentos: a similaridade por co-seno [58]. O cálculo dessa medida produz o co-seno do ângulo entre dois vetores. A similaridade por co-seno é dada por:

$$S_{\cos}(x_a, x_b) = \frac{x_a^T x_b}{||x_a||_2 \times ||x_b||_2} \tag{2.2}$$

Uma propriedade interessante da similaridade por co-seno é a de que ela não depende do comprimento dos vetores envolvidos. Isso permite que dois documentos que tenham a mesma composição de termos, mas com quantidade diferentes, sejam tratados de forma equivalente. Em domínios de aplicação onde dados direcionais são típicos, como a bioinformática [30] e a filtragem colaborativa [62], a similaridade por co-seno se mostra adequada.

Distâncias de Mahalanobis. A distância de Mahalanobis entre dois vetores x_1 e x_2 (em um espaço n-dimensional) é dada pela seguinte expressão:

$$M(x_1, x_2) = \sqrt{(x_1 - x_2)^t Cov^{-1}(x_1 - x_2)} = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} \sigma_{ij}(x_{1i} - x_{2i})(x_{1i} - x_{2i})} \tag{2.3}$$

Na Equação 2.3, Cov é a matriz de variância-covariância de x_1 e x_2. Note que a distância de Mahalanobis é equivalente à distância Euclidiana se a matriz de co-

[1](

variância é a matriz identidade I. Note também que, quando Cov é uma matriz diagonal, a distância de Mahalanobis pode ser vista como uma generalização da distância Euclidiana. Essa generalização permite considerar correlações entre as características dos objetos. Note ainda que, da mesma forma que o parâmetro p (Equação 2.1) parametriza uma família de Distâncias de Minkowski, a matriz Cov parametriza uma família de distâncias de Mahalanobis. Métricas parametrizáveis como a distância de Mahalanobis são freqüentemente utilizadas em trabalhos de agrupamento semi-supervisionado baseados em distâncias (conforme descrito na seção 3.3.3).

2.2.2 Medidas de Similaridade para Dados Categorizados

Além das medidas de similaridade para objetos cujas características possuem valores contínuos, existem também diversas medidas apropriadas para dados *categorizados*. Um característica categorizada de um determinado objeto pode assumir um de um conjunto finito de valores.

Métricas para dados categorizados são normalmente utilizadas na tarefa de *agrupamento conceitual*. Nessas métricas, em vez da interpretação dos objetos como vetores no espaço m-dimensional, a interpretação utilizada é similaridade e dissimilaridade entre as características dos objetos.

Há dois tipos de características categorizadas: *nominais* e *ordinais*. Em caractetrísticas nominais, não há sentido em se falar da ordenação dos valores. Uma característica nominal (ou simbólica) é aquela em que seus valores não estão associados a uma ordem total. Por exemplo, uma variável representando *cor* pode ter valores tais como *verde*, *azul*, *marrom*, *preto* e *branco*. Para quaisquer desses valores, não faz sentido a aplicação de um operador relacional que não sejam os operadores de igualdade e de desigualdade. Já em caractetrísticas ordinais, uma ordenação dos valores é possível. Como exemplo, considere a Tabela 2.1, que apresenta um conjunto de objetos com características de valores categorizados. Nessa tabela, a característica *Grau de instrução* é ordinal.

A Tabela 2.2 resume as principais medidas para tipos de dados categorizados.

Tabela 2.1: Objetos com características de valores categorizados

Estatura	**Sexo**	**Grau de instrução**
alta	*F*	*nível-fundamental*
alta	*M*	*nível-médio*
baixa	*M*	*nível-médio*
alta	*F*	*nível-médio*
mediana	*M*	*graduação*
baixa	*F*	*mestrado*

São elas *Jaccard*, *Overlap*, *Dice* e *Co-seno*. Nas expressões correspondentes a cada uma dessas métricas, V_1 e V_2 são conjuntos de valores de características de dois objetos, e $Card(V)$ corresponde à cardinalidade do conjunto V.

Tabela 2.2: Métricas de similaridade para valores categorizados

Jaccard	$Card(V_1 \cap V_2)/Card(V_1 \cup V_2)$
Sobreposição (Overlap)	$Card(V_1 \cap V_2)/\min(Card(V_1), Card(V_2))$
Co-seno	$Card(V_1 \cap V_2)/\sqrt{Card(V_1) \times Card(V_1)}$
Dice	$(2 \times Card(V_1 \cap V_2))/(Card(V_1) + Card(V_2))$

2.3 Métodos de Agrupamento

Há uma vasta quantidade de algoritmos de agrupamento propostos na literatura. Essa seção descreve alguns possíveis critérios de classificação que podem ser aplicados sobre essas propostas:

- *Estratégia de agrupamento*: de que forma o algoritmo de agrupamento interpreta os objetos com o objetivo de formar grupos a partir desses objetos?
- *Natureza da pertinência de objetos em grupos*: objetos podem pertencer a mais de um grupo simultaneamente?

- *Estrutura do agrupamento*: de que maneira um algoritmo de agrupamento processa a coleção de objetos $\mathcal{X}$ para identificar um conjunto de grupos $\{G_i\}$? Qual é a estrutura dos grupos G_i gerados pelo algoritmo de agrupamento?

Além de descrever os aspectos enumerados acima, esta seção também descreve alguns exemplos de algoritmos de agrupamento existentes.

2.3.1 Estratégia de Agrupamento

Quando se leva em consideração a estratégia utilizada por um algoritmo de agrupamento para gerar os grupos, pode-se classificar esses algoritmos em *geradores* ou *discriminativos*. Métodos *geradores* (ou baseados em modelos) consideram que os objetos a serem agrupados são gerados a partir de uma mistura de distribuições de probabilidades, onde cada componente da mistura corresponde a um dos grupos [10, 57]. Métodos *discriminativos* (ou baseados em similaridade) calculam a distância ou similaridade entre todos os possíveis pares de pontos e agrupam os pares similares entre si [63, 40].

De uma forma geral, algoritmos geradores são mais escaláveis que algoritmos discriminativos. Os primeiros, em geral, apresentam complexidade $O(kn)$, onde k corresponde ao número de grupos gerados, e n corresponde à quantidade de objetos. Já os algoritmos discriminativos apresentam complexidade $O(n^2)$, por conta de terem que calcular a similaridade entre todos os possíveis pares de objetos.

2.3.2 Natureza da Pertinência de Objetos em Grupos

Outro critério de classificação diz respeito ao modo pelo qual um algoritmo de agrupamento define o grau de pertinência de um objeto em $\mathcal{X}$ em um ou mais grupos de k. De acordo com esse critério, há os seguintes tipos de agrupamento [46, 38]: *agrupamento sobreposto* e *agrupamento estrito*.

Agrupamento sobreposto (*fuzzy clustering*). Esta família de algoritmos de agrupamento utiliza técnicas da Teoria de Conjuntos Difusos (*Fuzzy Set Theory*) para agrupar os objetos. A principal característica desses algoritmos é que eles

consideram que um objeto pode pertencer a mais de um grupo simultaneamente. Um exemplo de trabalho que propõe um algoritmo de agrupamento sobreposto é o *Fuzzy k-Means* [16].

Agrupamento estrito (*hard clustering*). Esta família de algoritmos considera que cada objeto deve pertencer somente a um dos grupos, ou seja $G_i \cap G_j = \emptyset, i \neq j$. A maioria dos algoritmos de agrupamento existentes gera grupos sem sobreposição. Nesta tese, também consideramos apenas o agrupamento estrito de objetos.

2.3.3 Estrutura do Agrupamento

Quando se considera a estrutura do agrupamento resultante da aplicação de um algoritmo de agrupamento qualquer, existem os seguintes métodos de agrupamento: *partitivos* e *hierárquicos*. Vamos agora descrever esses métodos e, em cada um deles, dar exemplos de algoritmos representativos.

Métodos Partitivos Um algoritmo partitivo gera k grupos mutuamente exclusivos de objetos. Usualmente, métodos partitivos consideram que o número de grupos a serem identificados é um parâmetro de entrada do algoritmo. Além disso, é utilizado o conceito de *centro*, μ_i de um grupo G_i, que corresponde a um vetor representativo de todos os vetores correspondentes aos objetos associados a G_i. A tarefa do algoritmo partitivo é então otimizar (maximizar ou minimizar) uma *função-objetivo* que envolve cálculos de distância entre os objetos e seus respectivos centros. Como há a noção de *centro* de um grupo, normalmente métodos de agrupamento partitivo consideram que a coleção $\mathcal{X}$ a ser agrupada provém de uma distribuição de probabilidades gaussiana subjacente. Os dois maiores representantes da família de métodos partitivos são o K-Means [50, 24] e o *k-medoids* [41]. A diferença entre esses dois algoritmos é que o K-Means considera como centro de um grupo o seu *centróide*, enquanto que o k-medoids considera como centro de um grupo o objeto (da coleção de entrada) que se encontra mais próximo do centro. Há também propostas de algoritmos partitivos adequados para dados categorizados. Dois exemplos são o *k-prototypes* e o *k-mode* [39], ambos variantes do K-Means.

Vamos agora descrever mais detalhadamente o algoritmo K-Means clássico, pois esse é o algoritmo que serve de base para a definição de nossa proposta. O K-Means é um algoritmo guloso no qual o objetivo principal é definir k centros (um para cada grupo) através da otimização local de uma função objetivo. Esse algoritmo considera como centro de um grupo o seu *centróide*. O centróide de um grupo é definido como o vetor soma de todos os vetores correspondentes aos objetos associados a este grupo. A tarefa do algoritmo K-Means é minimizar uma *função-objetivo* correspondente à distância total entre os objetos e os centróides dos grupos aos quais esses objetos foram associados. Ou seja, o algoritmo acha um mínimo local para o problema de minimizar a *função de distorção dos mínimos quadrados*, cuja fórmula é apresentada a seguir.

$$MSE = \frac{1}{|\mathcal{X}|} \sum_{i=1}^{k} \sum_{x \in G_i} ||x - \mu_i||^2 \tag{2.4}$$

Na Equação 2.4, $|\mathcal{X}|$ é a cardinalidade do conjunto de objetos, k é a quantidade de grupos e μ_i é o centróide do grupo G_i. Essa função é também comumente chamada de *erro médio quadrado* (*mean squared error*) ou *dispersão* [41].

O processo de otimização (local) do K-Means pode ser dividido em dois passos realizados iterativamente até que algum critério de convergência se verifique. O primeiro passo corresponde a uma *associação* de objetos a grupos (cada grupo é interpretado como um modelo de alguma função de densidade de probabilidades). O segundo passo corresponde a uma *atualização* dos grupos (modelos das funções de densidades de probabilidades) tomando como base as associações da iteração atual: o K-Means associa o objeto x ao grupo G_i, cujo centróide tem distância mínima para x. Esse processo continua até que se atinja algum critério de convergência. (Por essa razão, métodos partitivos são também chamados de *métodos de refinamento iterativo*.). A Figura 2.1 apresenta o pseudo-código do algoritmo K-Means em sua forma clássica [50].

Uma questão importante no K-Means é como determinar o número k de grupos a serem gerados. Usualmente, isso é feito através da execução do algoritmo diversas

Algoritmo: *K-Means básico.*

Entrada: uma coleção $\mathcal{X}$ de objetos; valor de k

Resultado: um agrupamento (ou k-partição) $\{G_i\}_{i=1}^{k}$ de $\mathcal{X}$

Passos:

1. *Inicialização*:
   - 1a. Selecionar aleatoriamente k pontos como os centróides iniciais
   - 1b. Associar cada ponto ao centróide mais próximo
2. *Iteração*:
   - 2a. Calcular o centróide de cada grupo: $\mu_i = \sum_{x \in G_i} x$
   - 2b. Associar cada ponto ao centróide mais próximo: $\sum_{x_i \in G} \|x_i - \mu_i\|^2$
3. Parar assim que os centróides se estabilizem.

Figura 2.1: Algoritmo K-Means básico

vezes, testando diferentes valores de k e selecionando o valor que gerou um agrupamento de maior qualidade com relação a algum critério de avaliação (sobre critérios de avaliação da qualidade de um agrupamento, vide seção 2.4).

Os principais problemas encontrados por métodos de agrupamento partitivo são: (1) o número de grupos a serem identificados deve ser conhecido a priori; (2) é difícil para esses métodos identificar grupos com grandes variações de tamanho. Geralmente um grupo naturalmente muito grande (comparativamente aos demais) tende a ser "quebrado"; (3) a presença de ruído (*outliers*) nos dados de entrada pode alterar significativamente o cálculo do centro de um grupo; esses métodos são mais adequados quando os objetos a serem agrupados tendem a formar grupos que apresentam uma estrutura *esférica*. Esse aspecto dos algoritmos partitivos tem como origem o fato de eles utilizarem um único objeto como representativo de cada grupo (normalmente esse objeto corresponde ao centróide do grupo, ou ao objeto mais próximo do centro). Apesar desses problemas, métodos de agrupamento partitivo

são bastante populares e utilizados em aplicações práticas.

Métodos Hierárquicos Em um algoritmo hierárquico, os grupos resultantes formam uma hierarquia (árvore) comumente chamada de *dendrograma*. A forma básica de um dendrograma é apresentada na Figura 2.2.

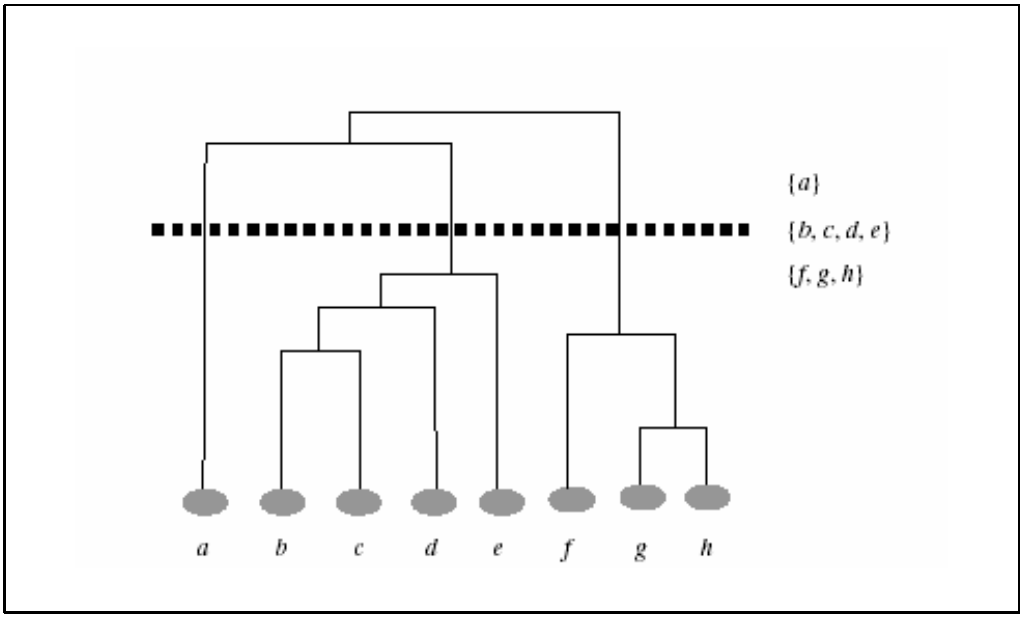


Figura 2.2: Forma básica de um dendrograma, estrutura gerada por algoritmos que formam agrupamentos hierárquicos de objetos.

Algoritmos hierárquicos podem ser subdivididos em dois tipos: *divisivos* (*top-down*) e *aglomerativos* (*bottom-up*). Algoritmos divisivos começam considerando a coleção de objetos como um único grupo. A seguir, dividem (particionam) os grupos iterativamente. Já os algoritmos aglomerativos começam por considerar cada objeto da coleção como sendo um grupo. A seguir, iterativamente, unem os grupos menores em grupos cada vez maiores, até que um único grupo que contém todos os objetos seja formado.

Um dos algoritmos hierárquicos propostos na literatura é o CURE [35]. Em vez de representar cada grupo através de um único ponto (como fazem os métodos partitivos), o CURE utiliza um número constante de pontos para representar. A seleção desses representantes é feita de tal forma que eles estejam o mais espalhados possível dentro do grupo. Uma vez selecionados, esses representantes são movidos

por um fator α ($0 \leq \alpha \leq 1$) em direção ao centróide do grupo. Grupos são unidos de forma iterativa com base nas suas similaridades. A similaridade entre dois grupos é medida pelo método de ligação simples, considerando somente os representantes desses dois grupos. Com uma escolha apropriada do parâmetro α, o CURE pode amenizar os problemas encontrados nos métodos partitivos (preferência por grupos esféricos e de tamanho uniforme e suscetibilidade a ruído).

Assim como nos outros métodos, a grande maioria das propostas de algoritmos hierárquicos presentes na literatura é adequada somente para dados contínuos. Uma exceção é o ROCK [36], algoritmo de agrupamento hierárquico apropriado para dados categorizados.

Uma desvantagem dos algoritmos de agrupamento que geram grupos hierárquicos é a sua alta complexidade computacional. A complexidade desses algoritmos é $O(n^2 \log n)$, onde n é o número de objetos. Por conta dessa complexidade, esses algoritmos se tornam computacionalmente proibitivos para problemas de agrupamento em que milhões de objetos estão envolvidos. Uma segunda desvantagem é que, uma vez formado um grupo (seja por aglomeração ou por divisão), este não pode ser desfeito em iterações posteriores do algoritmo.

2.4 Avaliação da Qualidade de um Agrupamento

Uma questão importante é como avaliar a qualidade dos grupos resultantes. A *avaliação de agrupamento* é uma área de pesquisa na qual se estudam maneiras de quantificar a *qualidade* (ou *validade*) de uma configuração de agrupamento [38]. Medidas para a avaliação da qualidade de um agrupamento podem ser baseadas em *medidas internas* ou em *medidas externas* [41]. Nas próximas duas seções, descrevemos cada uma dessas famílias de medidas de avaliação de qualidade.

2.4.1 Medidas Internas de Avaliação

A principal característica de uma medida interna é que seu cálculo é baseado única e exclusivamente no conjunto de objetos $\mathcal{X}$. Todo algoritmo de agrupamento possui

uma *função objetivo* que o guia através do espaço de possíveis soluções de agrupamento. Esta função é utilizada para avaliar soluções candidatas e indicar o quão boas (ou ruins) são essas soluções. O algoritmo tenta então otimizar (maximizar ou minimizar) o valor dessa função. Medidas internas de avaliação se baseiam no valor da função objetivo para avaliar a qualidade do resultado do agrupamento. De fato, freqüentemente, a mesma medida utilizada para definir a função-objetivo que o algoritmo busca otimizar é também utilizada como medida interna de avaliação. Neste contexto, um exemplo de medida interna de avaliação da qualidade de um agrupamento é a *Soma de Erros Médios Quadrados*, apresentada na Equação 2.4.

2.4.2 Medidas Externas de Avaliação

Medidas externas se baseiam na existência de alguma informação externa associada aos objetos recém-agrupados. A forma mais comum desse tipo de informação são categorias, $C = \{c_i | i = 1, 2, 3, ..., n\}$. A cada objeto em $\mathcal{X}$ estão associadas uma ou mais categorias c_i. Considera-se que esta informação não é utilizada durante a execução do algoritmo de agrupamento, mas sim apenas após essa execução. A qualidade do agrupamento é medida pelo grau em que o algoritmo conseguiu reconstituir a distribuição de categorias previamente existente. Para averiguar esse grau, é montada uma *matriz de proximidade* para comparar os resultados do agrupamento com os rótulos externos (categorias) atribuídos a cada objeto. Em uma situação ideal, o algoritmo de agrupamento criaria $|C|$ grupos, e cada grupo conteria somente objetos da mesma categoria. Exemplos de medidas externas são: *medida-F*, *informação mútua normalizada*, *entropia* e *pureza*. Descrevemos essas medidas nos próximos parágrafos.

Medida-F (*F-Measure*). Essa medida se baseia em dois conceitos da área de Recuperação de Informação: *precisão* (*precision*) e *abrangência* (*recall*). A precisão é a proporção de objetos relevantes no total de objetos recuperados. A abrangência é a proporção de objetos relevantes recuperados, em relação a todos os objetos relevantes. Quanto aplicada à avaliação do resultado de um agrupamento, uma forma particular dessa medida é utilizada, a Medida-F par-a-par (*pairwise F-measure*).

Chamemos essa medida de MFPP. Para calcular MFPP, precisamos primeiramente calcular a precisão P e a abrangência R, através das expressões a seguir.

$$P = \frac{Quantidade\ de\ pares\ corretamente\ alocados\ no\ mesmo\ grupo}{Quantidade\ total\ de\ pares\ alocados\ no\ mesmo\ grupo} \tag{2.5}$$

e

$$R = \frac{Quantidade\ de\ pares\ corretamente\ alocados\ no\ mesmo\ grupo}{Quantidade\ total\ de\ pares\ realmente\ no\ mesmo\ grupo} \tag{2.6}$$

A MFPP para o agrupamento de objetos é então dada por:

$$MFPP = \frac{2 \times P \times R}{P + R} \tag{2.7}$$

A medida-F varia no intervalo [0, 1] e quanto mais alto o valor dessa, melhor é a qualidade do agrupamento gerado. Para uma discussão mais detalhada dessa medida, veja [65].

Informação Mútua Normalizada (*Normalized Mutual Information*, NMI). Essa medida avalia a qualidade de um agrupamento através da utilização de uma informação externa, a saber, rótulos associados a cada ponto do conjunto de dados. A *NMI* mede o quanto o agrupamento em particular conseguiu reconstruir a informação dos rótulos associados a cada objeto [26, 60]. Se G e G' são dois conjuntos de grupos, então a informação mútua entre G e G' é dada por:

$$MI(G, G') = \sum_{G_i \in G, G_j \in G'} p(G_i, G_j) \times \log_2 \frac{p(G_i, G_j)}{p(G_i) \times p(G_j)} \tag{2.8}$$

onde, $p(G_i)$ e $p(G_j)$ são as probabilidades de um documento selecionado aleatoriamente da coleção pertencer aos grupos g_i e g_j, respectivamente; a expressão $p(G_i, G_j)$ é a probabilidade mútua de que este mesmo documento pertença tanto a G_i quanto G_j [51]. O valor de $MI(G, G')$ varia na faixa de zero até $\max(H(G), H(G'))$, onde $H(\mathcal{G})$ é a entropia do conjunto de grupos $\mathcal{G}$ (como definida por Shannon [61]). Para obter um valor que varie entre zero e um, defini-se a *informação mútua nor-*

malizada da seguinte forma:

$$NMI(G, G') = \frac{MI(G, G')}{\max(H(G), H(G'))} \tag{2.9}$$

Quanto mais o agrupamento resultante combina com os rótulos de categorias associadas aos objetos, mais próximo de 1 é o valor de *NMI*. Esse valor é próximo de 0 para um agrupamento aleatório dos objetos. Em [60], mostra-se que a medida *NMI* é melhor que as medidas de pureza e de entropia, sendo que essas últimas apresentam uma tendência (*bias*) para privilegiar soluções de agrupamento que apresentam um maior número de grupos.

Entropia (*Entropy*). Para cada grupo G_i, a entropia $E(G_i)$ desse grupo é definida pela expressão a seguir:

$$E(G_i) = -\sum_{i=1}^{k} (\frac{|C_i \cap G_i|}{|G_i|}) \log (\frac{|C_i \cap G_i|}{|G_i|}) \tag{2.10}$$

Na Equação 2.10, $|C_i \cap G_i|$ corresponde à quantidade de elementos do grupo G_i que estão associados à categoria C_i. A entropia de um grupo é uma medida da desordem no interior desse grupo. Sendo assim, quanto menor o valor da entropia de um grupo, melhor é a sua qualidade. O melhor valor dessa medida é 0. De fato, uma solução de agrupamento perfeita teria cada grupo contendo objetos de uma e somente uma classe. Essa medida foi originalmente proposta em [61] e utilizada para validação de algoritmos de agrupamento em [52].

Pureza (*Purity*). A pureza $P(G_i)$ de um grupo de objetos G_i é dada pela seguinte expressão:

$$P(G_i) = \frac{1}{|G_i|} \max_j(n_i^j) \tag{2.11}$$

Na Equação 2.11, n_i^j é a quantidade de objetos da j-ésima categoria que foi associada ao grupo G_i. O valor de $P(G_i)$ pode ser interpretado como a fração da quantidade de objetos em G_i que estão associados à classe majoritária dentro deste grupo. A medida de pureza para o agrupamento como um todo $\{G_i\}$ é obtida como uma soma ponderada das purezas de cada grupo G_i. Sendo $|\mathcal{X}|$ a quantidade total

de objetos a serem agrupados, a pureza $P(\{G_i\})$ de uma solução de agrupamento obtida a partir desses objetos pode então ser calculada por:

$$P(\{G_i\}) = \sum_{i=1}^{k} \frac{|G_i|}{|\mathcal{X}|} P(G_i) \tag{2.12}$$

Capítulo 3

Documentos XML e Agrupamento Semi-supervisionado

It is my ambition to say in ten sentences what others say in a whole book.

Friedrich Nietzsche (1844 - 1900)

3.1 Introdução

Quando os objetos a serem agrupados são dados textuais na forma de documentos, a análise de grupos recebe o nome de *agrupamento de documentos*. A tarefa de *agrupamento de documentos* tem o objetivo de estruturar coleções de documentos e viabilizar formas eficientes de navegação e busca desses documentos. Exemplos práticos de mecanismos de busca que utilizam agrupamento de documentos são os sistemas ClusterMed[1], Vivisimo[2], Metacrawler[3] e Northern Light[4]. Esses sistemas agrupam automaticamente os resultados de uma pesquisa em categorias.

Como o escopo dessa tese é o agrupamento de documentos textuais em formato XML, neste capítulo, consideramos o problema do processamento de documentos

[1]http://clustermed.info/
[2]http://www.vivisimo.com
[3]http://www.metacrawler.com
[4]http://www.northernlight.com

textuais que estejam neste formato (Seção 3.2). Também descrevemos a tarefa de agrupamento semi-supervisionado (Seção 3.3) e os trabalhos relacionados aos desta tese (Seção 3.4).

3.2 Documentos XML Centrados em Texto

Documentos XML podem ser divididos em duas categorias: *documentos centrados em dados* e *documentos centrados em texto*. Documentos XML centrados em dados são aqueles nos quais XML é utilizado para compartilhamento de dados. Exemplos incluem encomendas de produtos, registros de pacientes e dados científicos. A estrutura física de tais documentos é normalmente rígida. Um caso especial de documento XML centrado em dados é uma página WEB dinâmica, que apresenta um catálogo online de produtos ou uma lista de endereços. Esses documentos correspondem a conjuntos de dados regulares, ou seja, que apresentam uma estrutura bem definida, muitas vezes isomórfica a um esquema de banco de dados. Documentos XML centrados em texto, por outro lado, são aqueles documentos nos quais os valores de alguns elementos estão em um formato textual, tais como manuais, páginas WEB estáticas e artigos. Esses documentos são caracterizados por uma estrutura mais irregular e conteúdo misto (isto é, alguns elementos contêm texto, e outros elementos contêm dados relacionais).

Como uma restrição de escopo, nesta tese consideramos somente a categoria de documentos XML centrados em texto, visto que documentos XML centrados em dados são na verdade derivados de dados estruturados (como por exemplo, de um banco de dados relacional). Esta seção dá uma visão geral das principais operações que são tradicionalmente realizadas sobre uma coleção de documentos textuais como parte de seu pré-processamento.

Dá-se o nome de *pré-processamento de documentos* ao conjunto de atividades de transformação realizadas sobre uma coleção com o objetivo de adequar a mesma à posterior aplicação de um ou mais algoritmos de análise. As atividades que tradicionalmente compõem o pré-processamento de uma coleção de documentos são des-

critas nesta seção. Antes, como ressalva, é importante notar que há outras técnicas de pré-processamento além das aqui descritas. Essas técnicas são normalmente utilizadas em trabalhos da área de Processamento de Linguagem Natural (*Natural Language Processing*, NLP). Um exemplo de técnica normalmente utilizada nessa área é o etiquetamento POS (*Part of Speech Tagging*), na qual cada termo de um documento é analisado para identificar sua classe morfológica (substantivo, verbo, advérbio, etc.). Técnicas dessa natureza se encaixam no grupo de técnicas chamado de *análise sintática profunda* (*deep parsing*), e não são consideradas em nosso presente trabalho.

3.2.1 Eliminação de Palavras Funcionais

Inicialmente, um aspecto importante no pré-processamento de uma coleção de documentos é considerar a existência de palavras que têm pouca força representativa. Essas palavras são denominadas *palavras funcionais* (tradução para *stop words*). Exemplos de palavras funcionais são artigos, preposições, pronomes, etc. Em situações práticas, o que se faz é desconsiderar essas palavras funcionais durante o processamento dos documentos. Como as características dos documentos de uma coleção correspondem aos termos que ocorrem na mesma, a eliminação de palavras funcionais faz com que a dimensionalidade dessa coleção diminua.

A remoção de palavras funcionais também é justificada pela denominada *Lei de Zipf*, que estabelece que a freqüência de uma palavra (em um documento ou na coleção de documentos como um todo) é inversamente proporcional à posição da mesma em uma lista ordenada pela freqüência de ocorrência das palavras [15]. Quer dizer, a Lei de Zipf prevê que a freqüência da i-ésima palavra mais freqüente é $1/i^{\theta}$, onde θ é uma constante empírica que depende da coleção de documentos. Para coleções de documentos típicas, o valor de θ está entre 1.5 e 2.0.

3.2.2 Aplicação de Lematização

Outra atividade normalmente realizada para redução da quantidade de termos utilizados para representar uma coleção de documentos é o mapeamento das palavras que ocorrem na coleção para as suas correspondentes raízes morfológicas. Essa técnica é conhecida como lematização (tradução para *stemming*). Por exemplo, os termos *artérias* e *arterial* podem ser mapeados para seu lema comum, *artéria*. Da mesma forma que a remoção de palavras funcionais, o lematização faz com que a dimensionalidade do espaço diminua; isso porque dois ou mais termos podem ser mapeados para um único lema.

Tendo em vista que a técnica de lematização é dependente da linguagem na qual estão escritos os documentos da coleção, os diversos algoritmos que implementam a técnica de lematização também são dependentes desse aspecto; dentre esses algoritmos, dois dos mais conhecidos são de Porter [31, 53] e de Lovins [49]. Mais recentemente, linguagens para a definição de *stemmers* têm sido definidas. A linguagem mais conhecida é a *Snowball*[5], cujo autor é Martin Porter. No pré-processamento das coleções de documentos utilizadas nesta tese, utilizamos a Snowball (vide Seção 5.4).

Um aspecto negativo da aplicação de lematização é o fato de que alguns termos podem ser erroneamente mapeados para uma mesma raiz, o que resulta em uma matriz de documentos por termos que não reflete corretamente a coleção de documentos original. No entanto, a redução de dimensionalidade obtida com a aplicação de lematização compensa esse aspecto negativo.

3.2.3 Modelagem dos Documentos

Outra decisão que é tomada no pré-processamento de uma coleção de documentos é a escolha da forma de modelar as características dos mesmos. Em outra palavras, que representação deve ser utilizada para estruturar essa coleção de documentos? Existem diversos modelos de representação propostos na literatura. Entre os prin-

[5]http://snowball.tartarus.org/

cipais modelos de representação de documentos textuais, podemos citar o modelo probabilístico [55], os modelos booleanos clássico e estendido [48] e o modelo de espaço vetorial [56]. O modelo probabilístico, como o próprio nome deixa transparecer, interpreta cada documento como um evento em um espaço amostral. O modelo booleano é baseado na Teoria dos Conjuntos e interpreta cada documento como um conjunto de termos. Por fim, o modelo de espaço vetorial interpreta cada documento utilizando conceitos e técnicas da álgebra Linear e da Geometria Espacial. Esse último modelo é o mais utilizado na prática e é o que adotamos nesta tese. Por esta razão, o restante dessa seção considera este modelo em maiores detalhes.

O modelo de espaço vetorial (*vector space model*, VSM) é uma forma de representação e modelagem de documentos que é freqüentemente utilizada no processamento de dados textuais. Nesse modelo, documentos são interpretados como objetos geométricos, mais especificamente como vetores em um espaço m-dimensional. As coordenadas de um vetor representante de um documento são obtidas a partir do conteúdo do documento.

O conteúdo de um documento é obtido através do seu processamento e conseqüente obtenção do conjunto T de termos (palavras) que nele ocorrem. A quantidade de ocorrências de um termo, tanto em cada documento, quanto na coleção como um todo, é utilizada para obter os valores das coordenadas dos vetores representativos dos documentos da coleção. Dessa forma, essa coleção pode ser vista como uma matriz $n \times m$, na qual cada coluna corresponde a um documento. As colunas dessa matriz correspondem à importância (ou relevância) de cada termo da coleção com relação a cada documento. Essa matriz é conhecida como *matriz de termos por documentos*. Cada coluna dessa matriz recebe o nome de *bolsa de palavras* (*bag of words*). Dá-se o nome de *dimensionalidade* de uma coleção de documentos à cardinalidade do conjunto T.

A Figura 3.1 apresenta a forma esquemática da matriz de termos por documentos. Os valores numéricos w_{ij} correspondentes aos elementos dessa matriz são normalmente chamados de *pesos*. Os valores desses pesos são usualmente definidos a partir da freqüência dos *termos*, em cada documento e na coleção como um todo.

$$T \times D = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nm} \end{pmatrix}$$

Figura 3.1: Forma esquemática de uma matriz de termos por documentos. Cada coluna representa um documento da coleção e recebe o nome de *bolsa de palavras* (*bag of words*).

Medida TF/IDF

Existem diversos procedimentos que podem ser utilizados para o cálculo dos pesos de uma bolsa de palavras [59]. A maneria mais simples é utilizar a freqüência de um termo t_j em um documento d_i, ou seja, a quantidade de vezes que t_j ocorre em d_i. Outros procedimentos mais sofisticados consideram algum fator para ponderar o peso w_{ij}. O procedimento mais conhecido que segue essa estratégia corresponde à medida TF/IDF (*term frequency/inverse document frequency*), que descrevemos a seguir.

Denotamos por w_{ij} o valor da medida TF/IDF de um termo t_j em um documento d_i. São considerados dois fatores no cálculo de w_{ij}. O primeiro é a freqüência do termo t_j com relação a d_i, conforme descrito anteriormente. O segundo fator considerado na medida TF/IDF é a freqüência documental do termo, que corresponde à quantidade de documentos na coleção nos quais o termo t_j ocorre. A expressão matemática correspondente à medida TF/IDF e que corresponde à importância representativa do j-ésimo termo em relação ao i-ésimo documento é dada pela Equação 3.1. Nessa expressão, N corresponde à quantidade de documentos na coleção, df_{ij} é a quantidade de vezes que o termo t_j acontece no documento d_i, e df_j é a quantidade de vezes que o termo t_j acontece na coleção de documentos como um todo.

$$w_{ij} = df_{ij} \times \log \frac{N}{df_j} \tag{3.1}$$

A idéia subjacente à medida TF/IDF é a de que termos que ocorrem em documentos de uma coleção não têm igual força discriminatória para a caracterização desses documentos. De fato, note que a medida TF/IDF aumenta em função da

quantidade de ocorrências do termo no documento. Isso traduz o conceito intuitivo de que, quanto mais um termo ocorre em um documento, maior é o indício de que este termo seja representativo do mesmo (no entanto, veja a descrição sobre palavras funcionais). Além disso, a expressão acima também reflete outro conceito intuitivo, a saber, o fato de que quanto mais um termo ocorre na coleção como um todo, menor é o poder representativo deste termo com relação a um documento específico dessa coleção. De fato, considerando a situação extrema em que um termo ocorre em todos os documento da coleção, o segundo fator da Equação 3.1 se torna igual a zero.

Note que existem diversas variantes propostas na literatura para o cálculo do fator IDF na medida TF/IDF. A variante que utilizamos em nosso experimentos (e que é apresentada na Equação 3.1) é a mesma utilizada pelo sistema SMART [56].

Normalização

Outro aspecto importante a considerar durante o pré-processamento (principalmente quando o modelo de espaço vetorial é utilizado) é se há necessidade de *normalizar* os vetores que representam os documentos. Isso porque documentos grandes (ou seja, que contêm muitos termos) correspondem a vetores que possuem muitas coordenadas diferentes de zero. Isso faz com que o comprimento (módulo) desses vetores se torne grande, quando comparados aos demais documentos, o que pode resultar em distorções no cálculo de medidas de similaridades entre os documentos.

De fato, já foi demonstrado experimentalmente que a normalização dos vetores ajuda a reduzir a tendência (*bias*) que favorece documentos que têm maior comprimento [25]. Por essa razão, uma prática normalmente utilizada é alterar o comprimento dos vetores representantes dos documentos de tal forma que todos tenham comprimento unitário. Para isso, basta dividir (cada coordenada de) cada vetor $\vec{d}$ representante de um documento pela norma desse vetor, conforme mostra a Equação 3.2.

$$\vec{d}_{normalizado} = \frac{\vec{d}}{|\vec{d}|} \tag{3.2}$$

3.2.4 Redução da Dimensionalidade da Coleção

A grande dimensionalidade do espaço no qual documentos se encontram se deve ao fato do modelo de espaço vetorial considerar cada documento como um vetor em um espaço m-dimensional, onde m corresponde à quantidade de termos que ocorrem na coleção. É comum encontrar coleções de documentos contendo dezenas de milhares de termos, o que leva a uma representação de vetores em um espaço de dimensionalidade muito grande.

Vetores representativos de documentos também têm a característica intrínseca de serem esparsos (apresentarem muitas coordenadas com valor igual a zero). Isso se deve ao fato de que apenas uma pequena quantidade de termos da coleção como um todo ocorre em um determinado documento. Os demais termos da coleção que não ocorrem nesse documento têm no vetor correspondente o valor de coordenada igual a zero.

Ocorre que, mesmo após a remoção das palavras de pouco poder discriminatório (palavras funcionais) e redução de termos as suas raízes morfológicas (lematização), a quantidade de termos distintos que permanecem na coleção pode ainda ser muito grande para que a coleção seja tratável computacionalmente. De fato, a quantidade de termos que ocorrem em uma coleção de documentos pode chegar à casa das dezenas de milhares. Isso torna necessário e adequado eliminar o máximo possível de termos, de tal forma que o algoritmo de agrupamento utilizado trabalhe em uma espaço de dimensionalidade menor que a original. Por conta disso, outra atividade que é realizada durante o pré-processamento de documentos corresponde à aplicação de *técnicas para redução de dimensionalidade*.

Técnicas para redução de dimensionalidade se baseiam na suposição de que cada documento é representado como um conjunto de características, onde cada característica de um documento está associada a um valor numérico que indica a importância daquela característica para o documento. Um exemplo de representação normalmente utilizado nessas técnicas é o modelo de espaço vetorial, descrito na seção 3.2.3.

De acordo com a natureza da operação realizada sobre a coleção de documen-

tos, técnicas para redução de dimensionalidade podem ser divididas em dois tipos: *seleção de termos* e *extração de termos*.

A seleção de termos tem o objetivo de selecionar os termos mais representativos da coleção e que, por conta disso, devem ser utilizados como características dos documentos durante a execução do algoritmo de agrupamento. Através da aplicação de alguma técnica de seleção sobre o conjunto original de termos T_o, obtém-se o conjunto de termos T_f, de tal forma que $|T_o| \gg |T_f|$ e $|T_o| \supset |T_f|$. Uma técnica de seleção de termos bastaste utilizada na prática é definir dois valores inteiros positivos, δ_{inf} e δ_{sup}, e eliminar todos os termos cuja freqüência (considerando a coleção de documentos como um todo) seja menor que δ_{inf} e maior que δ_{sup}. Esses valores são chamados de *pontos de corte*.

A extração de termos tem o objetivo de obter um conjunto T_f a partir de T_o, o conjunto original de termos da coleção. No entanto, diferentemente da seleção de termos, o conjunto de termos obtido a partir de uma técnica de extração de termos não é um subconjunto do conjunto de termos que existem naturalmente na coleção. Duas técnicas conhecidas de extração de termos são a *Indexação Semântica Latente* (LSI, Latent Semantic Indexing) e o *agrupamento de termos*. A estratégia geral dessas técnicas é formar termos a partir da combinação (linear ou não) dos termos originais da coleção.

Independente do tipo de técnica utilizada (a extração ou seleção de termos), a redução de dimensionalidade normalmente resulta no aumento da eficiência do processamento da coleção e na diminuição do risco de o algoritmo se ajustar demasiadamente aos documentos utilizados para geração do modelo de agrupamento.

3.3 Agrupamento Semi-Supervisionado

O objetivo dessa Seção é definir a tarefa de agrupamento semi-supervisionado, escopo principal desta tese. Antes disso, descrevemos a distinção que há entre duas tarefas relacionadas à tarefa de agrupamento semi-supervisionado: *classificação* e *agrupamento*.

3.3.1 Aprendizado: Classificação vs. Agrupamento

As tarefas de *classificação* e de *agrupamento* são dois problemas comumente encontrados em Mineração de Dados. Essas duas tarefas têm como objetivo mapear um conjunto de objetos (documentos, imagens, registros em uma tabela, etc.) em um conjunto de *categorias*. No entanto, a abordagem em cada uma dessas duas tarefas é diferente.

Em um algoritmo de classificação típico, a entrada é fornecida como uma função de mapeamento $f : T \rightarrow C$ que mapeia um objeto x_i pertencente a um conjunto de treinamento T em uma ou mais classes (ou categorias) pré-definidas de um conjunto finito C. Esta função implícita é apresentada ao algoritmo de classificação na forma de um conjunto de objetos $< x_i, c_{i1}, c_{i2}, \ldots, c_{ik} >$ onde x_i é o i-ésimo objeto no conjunto de treinamento, c_{i1} .. c_{ik} são as classes correspondentes, e k é um inteiro não-negativo. Um algoritmo de classificação aprende um padrão de classificação que o permite associar objetos a zero ou mais das classes representadas por c_{i1} .. c_{ik}. Usualmente, o modelo resultante da aplicação do algoritmo é validado em um conjunto de objetos que não foi utilizado na fase de aprendizado (treinamento), com o objetivo de validar esse modelo e averiguar sua capacidade preditiva sobre dados não utilizados durante sua geração. A tarefa de classificação é então uma tarefa que se encaixa na categoria de *aprendizado supervisionado*.

Pelo descrito acima em relação a algoritmos de classificação, pode-se entender a diferença principal entre as tarefas de classificação e agrupamento. Nesta última, os objetos utilizados no aprendizado não possuem rótulos associados. Um algoritmo de agrupamento clássico deve derivar um padrão de classificação somente a partir dos objetos $< x_i >$. A tarefa de agrupamento é então uma tarefa que se encaixa na categoria de *aprendizado não-supervisionado*.

No entanto, recentemente tem havido crescente interesse na tarefa de *agrupamento semi-supervisionado*, uma tarefa que se encontra a meio caminho entre as tarefas de classificação e de agrupamento. A idéia básica do agrupamento semi-supervisionado é adicionar certo grau de supervisão a algoritmos de agrupamento, que tradicionalmente são algoritmos que se encaixam na categoria de não-supervisionados.

A informação externa (supervisão) normalmente é apresentada ao algoritmo na forma de exemplos ou de restrições a serem aplicadas durante o processo de agrupamento.

3.3.2 Vantagens do Agrupamento Semi-Supervisionado

A razão para aplicar agrupamento semi-supervisionado é permitir que o processo de formação dos grupos de objetos se beneficie de informação externa acerca da organização subjacente a esses dados.

Uma vantagem prática do agrupamento semi-supervisionado é que essa abordagem reflete a noção do usuário do algoritmo acerca do tipo de agrupamento desejado. Por exemplo, considere uma coleção na qual cada documento possui a mesma estrutura do documento da Figura 4.1. Nessa hipotética coleção, haveria diversas características que o algoritmo de agrupamento poderia utilizar para separar os artigos em grupos: por autoria do artigo, por título, por filiação, etc. O fato é que não se pode dizer que somente um desses possíveis agrupamentos é o correto. O usuário poderia estar interessado em formar grupos de acordo com um conjunto não-vazio C de características específicas, o que significaria que C teria mais relevância (ou peso) que as demais características de cada artigo, para o tipo de agrupamento desejado.

Outro exemplo prático desse benefício se apresenta quando do processamento de um conjunto de objetos que possui *diversos* (mais de um) agrupamentos possíveis. Por exemplo, ao examinar uma coleção de documentos acerca de uma determinada doença, o usuário pode estar interessado em um de vários tipos de agrupamento possíveis: por reincidência da doença, por tipo de tratamento utilizado, etc. Métodos de agrupamento não-supervisionado podem apenas produzir um possível agrupamento, que pode mesmo não corresponder ao tipo de agrupamento desejado pelo usuário. Diferentemente, com o uso da abordagem de agrupamento semi-supervisionado, o usuário pode guiar a produção dos grupos de acordo com a sua *necessidade* no momento.

Do ponto de vista de uso prático, a semi-supervisão seria fornecida ao sistema por um usuário. Por exemplo, o sistema poderia apresentar ao usuário pares de objetos,

e o usuário por sua vez indicaria se os objetos (de cada par apresentado) devem ficar em um mesmo grupo, ou se devem ser postos em grupos diferentes. Após um número pré-determinado de indicações de pares, o sistema poderia começar a execução do algoritmo de agrupamento semi-supervisionado. Pode-se argumentar que este processo sobrecarrega o usuário, por conta de se esperar deste usuário informação para guiar o algoritmo. No entanto, isso é justificável levando em consideração a grande complexidade da tarefa de agrupamento e o fato de que pode haver diversas soluções de agrupamento igualmente adequadas, em função da necessidade do usuário.

3.3.3 Abordagens para Agrupamento Semi-Supervisionado

De acordo com [33], algoritmos para agrupamento semi-supervisionado podem ser categorizados em três tipos: *algoritmos baseados em aprendizado de métricas*, *algoritmos baseados em restrições* e *algoritmos baseados em rótulos*. Vamos agora descrever cada uma dessas abordagem, para estabelecer o contexto de nossa proposta de tese.

A primeira envolve a modificação da métrica normalmente utilizada pelo algoritmo de agrupamento para que a métrica modificada reflita eventuais relacionamentos entre os objetos de maneira mais efetiva. A principal motivação para o agrupamento semi-supervisionado baseado em aprendizado de métricas é o fato de que os objetos a serem agrupados podem residir em um espaço tal que os grupos formados não estejam suficientemente separáveis. Assim, quando a métrica é modificada, espera-se que objetos próximos (distantes) no espaço de acordo com a métrica original fiquem mais próximos (distantes) com a utilização da métrica modificada. Em outras palavras, os objetivos com a modificação da métrica são dois: minimizar a distância entre pontos de um mesmo grupo e maximizar a distância entre pontos pertencentes a grupos diferentes.

O processo de modificação da métrica é feito através de uma atividade de treinamento. Nessa atividade, a expressão matemática correspondente à métrica original é parametrizada por uma matriz A positivo-definida. Dado um conjunto de pontos rotulados, os valores a_{ij} dos elementos dessa matriz são obtidos através da aplicação

de uma fase de treinamento.

Como exemplo, considere a métrica Euclidiana. A expressão que mede a distância entre dois documentos é dada pela Equação 3.3. Note que essa expressão envolve a multiplicação dos vetores representantes dos documentos por uma matriz A. Em um algoritmo de agrupamento semi-supervisionado baseado em aprendizado de um métrica, o objetivo da fase de treinamento é definir os elementos (pesos) dessa matriz A.

$$\|di - dj\| = \sqrt{(d_i - d_j)^T A (d_i - d_j)} \tag{3.3}$$

Algoritmos de agrupamento semi-supervisionado baseados em aprendizado de distância, em geral, utilizam como base algum algoritmo pré-existente para agrupamento. No entanto, o algoritmo é modificado de tal forma que a medida de distância usualmente utilizada por ele é modificada com o objetivo de obedecer aos rótulos ou restrições existentes nos dados de entrada [42, 45, 70, 18, 9]. Conforme descrito na seção 2.4.1, a distância parametrizada de Mahalanobis pode ser treinada através da adaptação dos valores dos elementos da matriz de co-variância. Dos trabalhos mencionados acima, [9] e [70] treinam uma métrica de Mahalanobis através da técnica de otimização convexa.

A segunda abordagem para agrupamento semi-supervisionado é aquela na qual são utilizadas restrições entre objetos para influenciar no posicionamento desses objetos nos grupos formados pelo algoritmo. Algoritmos nessa abordagem têm como um dos componentes de entrada *restrições relacionais* que servem de guia para que esses algoritmos alcancem um melhor resultado com respeito à qualidade dos grupos gerados. Dois tipos de restrições comumente utilizados em trabalhos de agrupamento semi-supervisionado são **restrições positivas** (*must-links constraints*) e **restrições negativas** (*cannot-links constraints*). O primeiro tipo de restrição define que dois objetos devem obrigatoriamente estar no mesmo grupo. Já o segundo tipo de restrição serve para definir que dois objetos não podem pertencer ao mesmo grupo.

A terceira abordagem de agrupamento semi-supervisionado é a que considera que

uma pequena fração dos objetos a serem agrupados estão rotulados com o grupo ao qual devem pertencer. Um dos trabalhos da literatura que utiliza esta abordagem é o de Demiriz et al. [23]. Nesse trabalho, a função objetivo do algoritmo de agrupamento é modificada para recompensar situações em que os rótulos dos objetos (fornecidos como entrada) são obedecidas (ou seja, situações em que objetos de mesmo rótulo permanecem em um mesmo grupo).

A supervisão sob a forma de restrições é mais realística do que supervisão na forma de rótulos. Isso porque o usuário pode mais facilmente informar se, dados dois objetos, estes objetos devem ou não pertencer ao mesmo grupo. Na supervisão por rótulos, por outro lado, é menos provável a existência de rótulos anteriormente à execução do algoritmo de agrupamento, mesmo para uma pequena quantidade de documentos na coleção. Além disso, a abordagem que usa restrições é mais genérica que a abordagem que usa rótulos, pois a partir de um conjunto de objetos rotulados, pode-se derivar restrições positivas e negativas sobre eles. Nessa tese, consideramos apenas a supervisão por meio de restrições.

3.4 Trabalhos Correlatos

Nessa Seção, são descritos os trabalhos relacionados à nossa pesquisa. Dividimos essa descrição de acordo com os aspectos que exploramos em nossa tese.

3.4.1 Algoritmos para Agrupamento Semi-Supervisionado

Em [42] é proposta uma abordagem para agrupamento semi-supervisionado de documentos. A idéia básica dessa abordagem é variar a medida de distância utilizada através da atribuição de pesos diferentes às características dos objetos utilizados pelo algoritmo de agrupamento. Sendo assim, a medida de similaridade a ser utilizada pelo algoritmo em uma determinada coleção de documentos é adaptável. O "aprendizado" dessa medida de similaridade é feito através do prévio processamento de um conjunto de documentos, cada um deles rotulado como relevante ou irrelevante.

Alguns trabalhos exploram o fato de que nem todas as coordenadas de um espaço

de características de objetos têm igual importância. Em [71], os autores demonstram que, através de uma técnica de aprendizado denominada *programação semi-definida*, pode-se definir uma medida de similaridade (vide seção 2.2) mais adequada para o conjunto de objetos a ser agrupado. Essa métrica é produzida em uma fase anterior ao agrupamento. Para o aprendizado da métrica, são utilizadas restrições na forma de exemplos de pares de objetos similares e dissimilares entre si. A métrica aprendida é tal que respeita a essas restrições. Os autores conseguem obter uma melhoria significativa na qualidade dos grupos gerados com a utilização da medida aprendida. Contudo a complexidade computacional do método é $O(d^6)$, onde d é a dimensionalidade do espaço de objetos, o que reduz sobremaneira a sua aplicabilidade prática.

Em [6, 7], o algoritmo *MPC-KMeans* é apresentado. Neste algoritmo, um modelo probabilístico denominado HMRF (*Hidden Markov Random Field*) é utilizado. O modelo proposto é genérico o suficiente para permitir o uso de diferentes medidas de distância, tais como divergência-KL, divergência-I, distância de Mahalanobis e similaridade por co-seno. Assim como em [68], a técnica HMRF também considera restrições na forma de ligações positivas e negativas. Em [64], os autores apresentam um trabalho semelhante ao de [6, 7], considerando o problema de *filtragem de documentos*.

Uma limitação dos métodos de agrupamento semi-supervisionado baseados em aprendizado de métricas diz respeito ao tempo necessário para treinar a métrica de distância. Em situações práticas, onde o algoritmo de agrupamento deve ser utilizado para organizar os resultados de um mecanismo de busca *online* com o objetivo de apresentar esses grupos para o usuário desse mecanismo, o tempo necessário para treinamento da métrica torna o uso desses algoritmos inviável. A situação se agrava ainda mais quando os objetos que devem ser agrupados são documentos, que naturalmente possuem uma dimensionalidade muito alta. Por essa razão, em nossa tese, não utilizamos a abordagem de aprendizado de métricas. Em vez disso, utilizamos a abordagem de agrupamento semi-supervisionado baseado em restrições. Algoritmos representativos dessa abordagem são descritos a seguir.

Um algoritmo representativo da categoria de agrupamento semi-supervisionado baseado em restrições é o *COP-KMeans*, apresentado em [68]. Esse algoritmo corresponde a uma variante do K-Means e incorpora conhecimento prévio (*background knowledge*) no processo de agrupamento. Nesse algoritmo, os autores consideram dois tipos de restrições: positivas e negativas. No passo de inicialização, o *COP-KMeans* seleciona as sementes de forma aleatória. Mas, conforme cada semente s é selecionada, as restrições positivas e negativas em que esta semente eventualmente participe são consideradas e obedecidas pelo algoritmo. Isto é, todos os pontos p_i tais que exista uma restrição positiva da forma ML(s, p_i) são alocados no mesmo grupo de s. Dessa forma, esses pontos p_i não podem posteriormente ser selecionados como centros de novos grupos. Após a inicialização, o *COP-KMeans* executa iterativamente os dois passos a seguir:

- Associação de cada ponto ao grupo G tal que o centróide de G seja mais próximo e nenhuma restrição (positiva ou negativa) seja violada.
- Atualização do centróide de cada grupo G_i: $\mu_i = \sum_{x \in G_i} x$

Em [5], o algoritmo *PC-KMeans* é apresentado. Esse algoritmo também é uma extensão do K-Means que considera restrições positivas e negativas. No entanto, diferentemente do *COP-KMeans*, o *PC-KMeans* utiliza essas restrições na fase de inicialização do algoritmo de agrupamento, o que se justifica pelo fato de algoritmos partitivos (como o K-Means) serem altamente dependentes da fase de produção dos centróides iniciais.

Assim como *COP-KMeans* e o *PC-KMeans*, nosso algoritmo, o *MAC-KMeans*, também é uma extensão do K-Means clássico. Além disso, também consideramos restrições positivas e negativas. A diferença é que, nesta tese, essas restrições podem ser definidas sobre um nível de abstração adicional: além de ter a possibilidade de definir restrições sobre identificadores de documentos (assim como acontece no *COP-KMeans* e no *PC-KMeans*), nosso método também permite que o usuário defina restrições a nível de informação estrutural dos documentos XML. Isso permite que o usuário tenha maior flexibilidade e facilidade na definição das restrições que refletem

sua necessidade em relação à natureza dos grupos que ele deseja que o algoritmo gere (vide Seção 4.2).

Outro aspecto relevante para esta tese diz respeito à inicialização das *sementes* do algoritmo de agrupamento. No *PC-KMeans*, a inicialização das sementes é realizada através de um procedimento em dois passos. Em primeiro lugar, o conjunto de restrições fornecido é *aumentado*, através da consideração de que esse conjunto é consistente e de que as restrições positivas formam *classes de equivalência*. Dado um conjunto $\mathcal{R}$ de restrições definidas sobre documentos, o *PC-KMeans* realiza o procedimento a seguir para aumentar $\mathcal{R}$. (Considere nesta descrição que, dados dois documentos d_i e d_j, $ML(d_i, d_j)$ corresponde a uma restrição positiva, e que $CL(d_i, d_j)$ corresponde a uma restrição negativa entre esses documentos.)

1. Adicione todas as restrições positivas em $\mathcal{R}$ ao seu fechamento, *Fechamento($\mathcal{R}$)*.

2. Se as restrições ML(d_i, d_j) e ML(d_j, d_k) pertencem a $\mathcal{R}$, então adicione a restrição ML(d_i, d_k) ao *Fechamento($\mathcal{R}$)*.

3. Se as restrições ML(d_i, d_j) e CL(d_j, d_k) pertencem a $\mathcal{R}$, então adicione a restrição CL(d_i, d_k) ao *Fechamento($\mathcal{R}$)*.

4. Repita os passos 2 e 3, até que não haja mais restrições a serem adicionadas a *Fechamento($\mathcal{R}$)*

A Figura 3.2 ilustra as regras que são aplicadas pelo *PC-KMeans* para obter o *fechamento* de um conjunto de restrições.

Para exemplificar o passo de aumento, considere o conjunto $\mathcal{M}$ da Figura 3.3. As seguintes partições podem ser obtidas a partir de $\mathcal{M}$: $P_0 = \{A, C, D, E\}$ e $P_1 = \{G, J, Z\}$. Além disso, ainda de acordo com a Figura 3.3, o item F está ligado por restrições negativas tanto a elementos de P_0, quanto a elementos de P_1. Isso leva à criação de outra partição, $P_2 = \{F\}$.

Após o aumento do conjunto de restrições através da geração de seu fechamento transitivo, o *PC-KMeans* forma *partições* de documentos, através da utilização dos documentos que participam de restrições positivas. Os documentos de cada classe de

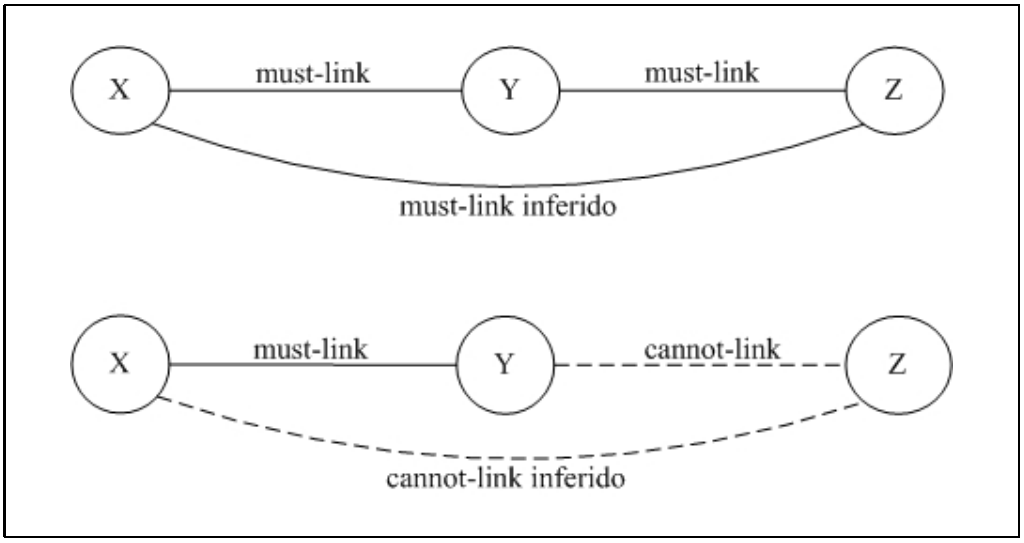


Figura 3.2: Inferência de restrições positivas e negativas a partir de restrições pré-existentes

equivalência formam então uma partição. Finalmente, essas partições são utilizadas para inicializar as sementes do algoritmo: em vez de selecionar as sementes de forma aleatória (como ocorre no K-Means clássico), o *PC-KMeans* calcula os centróide de cada partição e os utiliza como centróides iniciais dos grupos a serem gerados.

Note que o processo descrito acima assume que há *consistência* entre as restrições. Em nossa tese, não adotamos essa hipótese. Isso porque o algoritmo que definimos para inferir restrições sobre documentos a partir de restrições sobre metadados gera inconsistências, inerentemente. Para lidar com esse problema, definimos algoritmos para eliminar inconsistências do conjunto de restrições inferidas, antes de passá-las ao nosso algoritmo de agrupamento semi-supervisionado.

3.4.2 Novas Formas para Definição de Restrições

Nesta tese, definimos uma representação para documentos XML que permite ao usuário a definição de restrições positivas e negativas não somente ao nível de documentos, mas também ao nível de atributos existentes nos metadados da coleção. Essa possibildidade dá mais flexibilidade ao usuário na definição de restrições.

Um trabalho correlato nesse contexto de definição de restrições é [27], no qual Davidson e Ravi definem dois novos tipos de restrições, ϵ e δ. A restrição ϵ (ϵ-

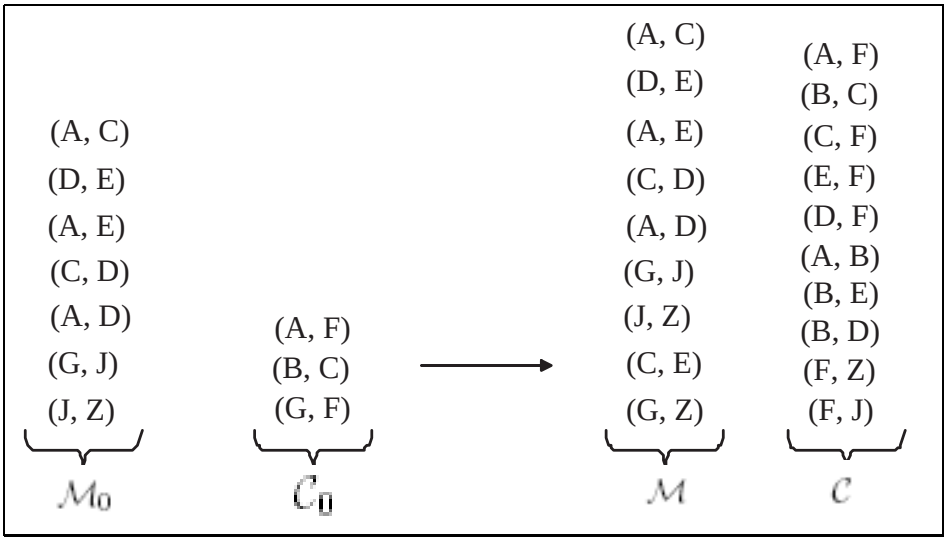


Figura 3.3: Exemplo de aumento do conjunto de restrições fornecido pelo usuário.

constraint) indica que cada objeto x em um grupo deve obrigatoriamente ter outro objeto y pertencente ao mesmo grupo, de tal forma que a distância entre x e y é no máximo ϵ. A restrição δ (δ-constraint) indica que todo objeto em um grupo deve obrigatoriamente estar a uma distância de no mínimo δ de objeto pertencentes a outro grupo. Ambas as restrições podem ser utilizadas para especificar conhecimento prévio em relação à distância mínimo entre grupos/objetos. Desta forma, essas restrições permitem que o usuário defina restrições ao nível dos grupos que devem ser formados pelo algoritmo.

Nesta tese, também definimos novas formas de o usuário definir restrições, assim como em [27]. Contudo, até onde vai nosso conhecimento, nossa abordagem para possibilitar a definição de restrições através do uso de informação estrutural em formato XML é inédita. Além disso, definimos uma abordagem de transformação de restrições ao nível de metadados em restrições ao nível de documentos. Isso possibilita o acoplamento de nossa abordagem a algoritmos que apenas trabalhem com restrições ao nível de documentos.

3.4.3 Definição de Pesos para Restrições

Diversos trabalhos em agrupamento semi-supervisionado supõem que todas as restrições têm igual importância. Normalmente, essas restrições são chamadas de *res-*

trições estritas (*hard constraints*). No entanto, assim como nesta tese, alguns trabalhos consideram a incorporação de graus de importância a cada restrição. Desta forma, a cada restrição é associado um valor que define a *força* dessa restrição, ou seja, o quão confiável é esta restrição. Chamamos estas restrições com pesos associados de *restrições estendidas* (*soft constraints*). Dessa forma, uma restrição estendida é da forma (o_i, o_j, w_{ij}), onde w_{ij} é o valor do peso da restrição definida entre os objetos o_i e o_j. Esta Seção analisa trabalhos correlatos em relação a este aspecto de nossa tese.

Em [47], Kleinberg e Tardos definem um modelo de restrições que também contempla restrições estendidas. Em sua formulação, eles permitem a definição de restrições ao nível de objetos. Cada restrição indica que dois itens devem possuir o mesmo rótulo (ser posicionadas no mesmo grupo), de acordo com o *peso* da restrição. Ou seja, apenas restrições positivas são consideradas. Eles incorporam a restrição na função objetivo do algoritmo de agrupamento através do cálculo de uma penalidade proporcional aos pesos das restrições violadas. Essa mesma abordagem de modificar a função objetivo do algoritmo é seguida nesta tese. No entanto, consideramos tanto restrições negativas quanto positivas (em vez de considerar apenas restrições positivas). Além disso, a estratégia utilizada em [47] para agrupamento com incorporação das restrições positivas usa uma técnica de programação linear, o que a torna cara computacionalmente.

Um exemplo de algoritmo que apenas considera restrições estritas é o *COP-KMeans* (descrito na Seção 3.4.1). Quando o *COP-KMeans* detecta alguma inconsistência, ele simplesmente termina sua execução e reporta que não pode ser produzido um agrupamento. Em [66], Wagstaff propõe uma extensão ao *COP-KMeans* que resolve o problema da falta de robustez desse algoritmo. Essa extensão é denominada *SCOP-KMeans* (*Soft COP-KMeans*). Nesse trabalho, o valor de cada peso varia entre -1 e $+1$, tanto para restrições positivas, quanto para restrições negativas. Outro algoritmo proposto na literatura que resolve o problema da falta de robustez do *COP-KMeans* é o *PC-KMeans* [5]. Esse algoritmo também utiliza restrições positivas e negativas. No entanto, sua função objetivo é modificada para

penalizar eventuais violações das restrições definidas. Uma característica comum do *SCOP-KMeans* e do *PC-KMeans* é que eles assumes que o conjunto de restrições a eles fornecido é *consistente*.

O que caracteriza um conjunto inconsistente de restrições é a existência (explícita ou implícita) de um ou mais pares de restrições (r_1, r_2), onde $r_1 = ML(o_i, o_j)$ (restrição positiva) e $r_2 = CL(o_i, o_j)$ (restrição negativa). Em situações práticas, e por vários motivos, o usuário pode fornecer ao sistema um conjunto de restrições *inconsistente*. Tomando isso como motivação, outro aspecto que exploramos nesta tese é o de usar pesos para restrições positivas e negativas com o objetivo de tratar eventuais inconsistências. Nosso algoritmo de agrupamento semi-supervisionado, o *MAC-KMeans*, também assume que o conjunto de restrições de entrada é consistente. No entanto, propomos algoritmos para eliminar inconsistências das restrições antes de passá-las a nosso algoritmo.

Uma característica comum de [47], [5] e [66] é que esses trabalhos assumem que o usuário é responsável por fornecer pesos para cada restrição definida. Nesta tese, aproveitamos a mesma formulação para restrições estendidas definida nesses trabalhos (ou seja, cada restrição está associada a um peso). Entretanto, definimos uma estratégia para extrair os pesos das restrições a partir dos metadados da coleção de documentos a ser agrupada, em vez de assumir que esta informação é fornecida pelo usuário (como acontece em [66] e [5]).

Capítulo 4

Agrupamento Semi-Supervisionado de Documentos XML com Utilização de Informação Estrutural

Andarão dois juntos se não houver um acordo entre eles?

Amós (3:3)

4.1 Introdução

Através da análise dos trabalhos correlatos (vide Seção 3.4), podemos concluir que há um interesse crescente nos últimos anos por técnicas que incorporem algum tipo de informação externa (ou conhecimento prévio) ao processo de agrupamento de documentos, com o objetivo de melhorar a qualidade dos resultados. No entanto, com o melhor de nosso conhecimento, constatamos que nenhum desses trabalhos explora a existência de informação estrutural na forma de XML sobre os objetos a serem agrupados. A proporção de dados no formato XML está crescendo rapidamente em diversas áreas de aplicação. Na WEB, por exemplo, a idéia da *WEB*

Semântica, de que a Internet possa ser *navegada* automaticamente por agentes inteligentes pressupõe que os dados nesse ambiente estejam representados em XML. Outro domínio de aplicação onde dados em XML são cada vez mais dominantes é o da *Bioinformática*. Bancos de dados imensos de seqüências de moléculas (proteínas e aminoácidos), assim como repositórios de coleções de artigos dessa área, contêm anotações textuais ou informação estrutural representadas através da utilização do formato XML. Se a tarefa de agrupamento tiver que ser utilizada para organizar esses repositórios, o aproveitamento de informação adicional para guiar o processo de agrupamento pode ser bastante útil.

Da análise dos fatos descritos acima, é que surgiu a motivação para esta tese, na qual abordamos o seguinte problema: de que forma o usuário de um algoritmo de agrupamento pode aproveitar a existência de informação estrutural sobre uma coleção de documentos XML para definir restrições a serem consideradas durante o agrupamento desses documentos? Neste capítulo, apresentamos nossa proposta de agrupamento semi-supervisionado de documentos XML através do uso de informação estrutural existente sobre eles. Nossa abordagem de agrupamento é *semi-supervisionada* no sentido de que, durante o processo de agrupamento, são utilizadas outras informações, além dos próprios objetos a serem agrupados.

A estrutura deste capítulo é descrita a seguir. Com o objetivo de viabilizar o uso de informação estrutural associada aos documentos a serem agrupados, definimos na Seção 4.2 uma forma de representação *dual* para documentos XML, de tal forma a permitir que um documento seja representado tanto por um conjunto de características encontradas em seus metadados, quanto por características retiradas de seu próprio conteúdo. Na Seção 4.3, definimos nossa abordagem para explorar a existência de informação estrutural sobre uma coleção de documentos com o objetivo de guiar o processo de agrupamento dessa coleção. Mais especificamente, consideramos a existência de um esquema XML sobre a coleção de documentos algo que pode ser utilizado como informação útil para guiar o processo de agrupamento em aspectos diversos. Finalmente, com o objetivo de dar uma visão sistêmica de nossa proposta, a Seção 4.4 descreve a arquitetura geral de um *sistema de agrupa-*

mento semi-supervisionado de documentos, ou seja, de um sistema que implemente os procedimentos relativos a agrupamento semi-supervisionado descritos na Seção 4.3. Neste capítulo, quando usamos os termos *sistema* e *usuário*, estamos fazemos menção ao sistema cuja arquitetura é apresentada na Seção 4.4 e a seus usuários, respectivamente.

4.2 Representação Dual para Documentos XML

Nesta Seção, definimos nossa representação dual para um documento XML. Esta representação é importante em virtude de que precisamos consultar tanto a estrutura quanto o conteúdo dos documentos XML na coleção a ser agrupada. Nesta tese, consideramos uma definição simplificada para um documento XML, a saber: um documento XML é uma árvore rotulada $Arv(V, E)$, onde V é um conjunto de nós e $E \subseteq V \times V$ é um conjunto de arestas. Cada nó pode conter zero ou mais nós como filhos. Arestas em E são usadas para conectar um elemento a seus subelementos. Não consideramos *atributos* de um nó.

Um documento XML típico que consideramos é apresentado na Figura 4.1. Este documento é dito *centrado em texto* devido ao fato de que alguns de seus elementos têm conteúdo textual (e.g., *ArticleTitle* e *Abstract*). Por outro lado, este documento também possui elementos descritivos (e.g., *AuthorList* e *MeshHeadingList*).

A abordagem convencional para o pré-processamento de documentos textuais aplica as técnicas descritas na Seção 3.2 (remoção de *stop words*, *stemming* e redução de dimensionalidade), e desconsidera quaisquer informações estruturais sobre os documentos processados. Visto que nosso principal objetivo é utilizar informação estrutural no formato XML associada aos documentos para melhorar os resultados do agrupamento, nesta tese adotamos uma abordagem diferente para indexação de documentos: consideramos que um documento XML centrado em texto é pré-processado de acordo com duas perspectivas. A seguir, passamos à descrição dessas duas perspectivas.

Considere uma coleção de documentos XML centrados em texto. Considere

```
<MedlineCitation>
  <PMID>8651511</PMID>
  <ArticleTitle>
    The morphology of valves and valve-like structures in the canine and
    feline thoracic duct.
  </ArticleTitle>
  <Abstract>
    <AbstractText>
      The microanatomy and ultrastructure of the feline and canine thoracic
      duct and afferent lymphatics were studied by scanning and ... (truncado)
    </AbstractText>
  </Abstract>
  <AuthorList>
    <Author>Bannykh S. S.</Author>
    <Author>Mironov A. A.</Author>
    <Author>Bannykh G. G.</Author>
    <Author>Mironov A. A. Jr.</Author>
  </AuthorList>
  <MeshHeadingList>
    <MeshHeading>Animals</MeshHeading>
    <MeshHeading>Cats</MeshHeading>
    <MeshHeading>Dogs</MeshHeading>
    <MeshHeading>Endothelium</MeshHeading>
    <MeshHeading>Thoracic Duct</MeshHeading>
    <MeshHeading>Microscopy, Electron</MeshHeading>
  </MeshHeadingList>
  <PublicationTypeList>
    <PublicationType>Journal</PublicationType>
  </PublicationTypeList>
</MedlineCitation>
```

Figura 4.1: Um documento XML centrado em texto

também que $\mathcal{A}$ é o conjunto de todas as subárvores XML obtidas a partir da árvore correspondente ao esquema XML dessa coleção. Além disso, considere dois subconjuntos de $\mathcal{A}$, que denotamos por $\mathcal{A}_{vsm}$ e $\mathcal{A}_{desc}$. Esses subconjuntos correspondem a perspectivas complementares de um documento que permitem explorar e consultar sua estrutura e conteúdo. Por sua importância para nossa tese, esses subconjuntos são descritos separadamente nas subseções que seguem. A Figura 4.2, que apresenta quatro subárvores extraídas da estrutura do documento XML apresentado na Figura 4.1, é utilizada como exemplo em ambas as subseções seguintes.

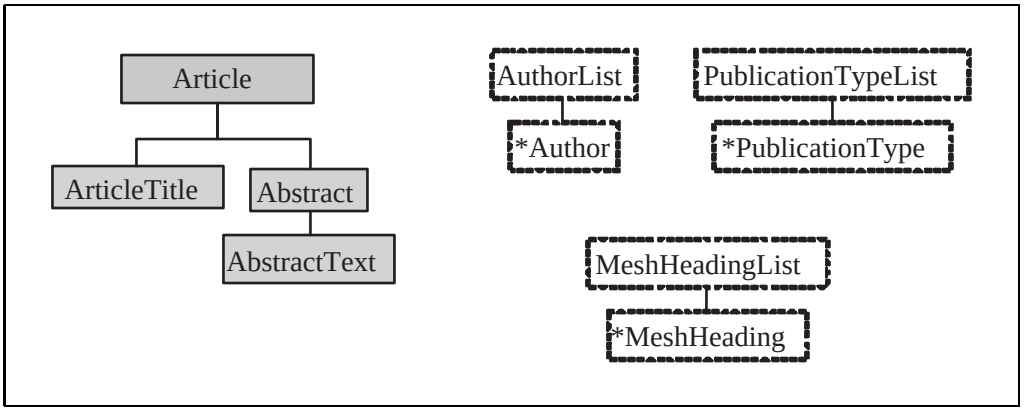


Figura 4.2: Quatro subárvores correspondentes aos conjuntos $\mathcal{A}_{vsm}$ e $\mathcal{A}_{desc}$ e extraídas a partir de uma única árvore correspondente à estrutura de documentos XML em uma coleção.

4.2.1 Documentos como Bolsas de Palavras ($\mathcal{A}_{vsm}$)

A primeira perspectiva, que considera os elementos de tipo texto do documento, aplica as mesmas técnicas descritas na Seção 3.2. Assim, seguindo a metodologia de sistemas da área de Recuperação de Informação [58], utilizamos o modelo de espaço vetorial (vide Seção 3.2.3) para representar nós de tipo texto de um documento XML. Nesse modelo, um subconjunto dos elementos tipo texto é representado como um vetor no espaço m-dimensional. Denominamos esse conjunto de elementos como $\mathcal{A}_{vsm}$, e damos sua definição a seguir.

Definição 4.1 *$\mathcal{A}_{vsm}$ é o conjunto de subárvores de $\mathcal{A}$ cujo conteúdo é utilizado para representar os documentos como uma bolsa de palavras. Esta primeira perspectiva representa um documento XML como $d_i = (w_{i1}, w_{i2}, \ldots, w_{im})$, onde w_{ij} é o grau de importância do j-ésimo termo em relação a $\mathcal{A}_{vsm}$.*

A representação obtida através do pré-processamento dos elementos de $\mathcal{A}_{vsm}$ corresponde a uma bolsa de palavras na qual um documento d_i é visto como um vetor em um espaço m-dimensional. Cada uma das m dimensões desse espaço corresponde a um termo que ocorre no conteúdo de pelo menos um dos elementos de $\mathcal{A}_{vsm}$. Note que uma medida de similaridade nesse espaço (como as medidas de Minkowski ou a medida de co-seno, descritas na Seção 2.2.1) pode ser utilizada para calcular a proximidade de d_i em relação a outros documentos (que também são interpretados

como vetores). Essa é a representação geométrica de d_i que é normalmente utilizada em algoritmos clássicos de agrupamento de documentos.

É importante notar que o valor de m corresponde à quantidade de termos que ocorrem internamente a um ou mais dos elementos do conjunto $\mathcal{A}_{vsm}$. Ou seja, em vez de serem considerados os termos que ocorrem em *todos* os elementos de d_i, apenas os termos que ocorrem em $\mathcal{A}_{vsm}$ são considerados. Isso resulta na representação de um documento $d_i = (w_{i1}, w_{i2}, \ldots, w_{im})$, onde o peso w_{ij} representa o grau de importância do j-ésimo termo em relação a $\mathcal{A}_{vsm}$. Há diversas maneiras de calcular o valor de w_{ij}. Em nossos experimentos (capítulo 5), utilizamos a medida TF/IDF como grau de importância (vide Seção 3.2.3).

Como exemplo, considere a Figura 4.2, que ilustra elementos de $\mathcal{A}$. A primeira subárvore (em negrito, à direita) é uma subárvore de $\mathcal{A}_{vsm}$, $\mathcal{A}_{vsm} \subset \mathcal{A}$. Desta subárvore, podemos extrair uma bolsa de palavras para representar os documentos de acordo com a perspectiva geométrica.

4.2.2 Documentos como Tuplas de Características ($\mathcal{A}_{desc}$)

A segunda perspectiva que consideramos no pré-processamento de uma coleção de documentos XML é relativa à consideração do subconjunto de $\mathcal{A}$ formado pelos elementos XML cujos subelementos possuem conteúdo retirado de um mesmo domínio de valores categorizados. Denominamos esse conjunto de elementos como $\mathcal{A}_{desc}$, e damos sua definição a seguir.

Definição 4.2 *$\mathcal{A}_{desc}$ é o conjunto de subárvores de $\mathcal{A}$ cujo conteúdo é utilizado para representar os documentos como uma bolsa de características correspondentes a atributos categorizados.*

Esta segunda perspectiva de um documento XML obtida através do processamento dos elementos contidos em $\mathcal{A}_{desc}$ permite uma outra interpretação desse documento, a saber, como um objeto composto de um conjunto de atributos (características) *categorizados* (vide Seção 2.2.2). Ocorre que, considerando essa interpretação, também podemos aplicar sobre dois documentos quaisquer d_i e d_j da

coleção uma medida de similaridade entre suas características. Para isso, podemos utilizar as medidas de similaridade para atributos categorizados definidas na Seção 2.2.2. Por exemplo, esses modelos nos permitem calcular o grau de interseção das listas de autores (ou das listas de descritores) de d_i e d_j. Note que essa representação permite calcular o quão similares são dois documentos d_i e d_j em relação aos valores categorizados dos atributos multivalorados extraídos do conjunto $\mathcal{A}_{desc}$.

Para exemplificar, considere novamente a Figura 4.2. As subárvores apresentadas à direita dessa figura (com retângulos pontilhados) representam elementos de $\mathcal{A}_{desc}$. Note que os elementos de $\mathcal{A}_{desc}$ podem ser interpretados como atributos categorizados multivalorados, sendo que os valores de cada atributo podem ser obtidos a partir de seus subelementos. Por exemplo, a subárvore cuja raiz é *AuthorList*, é equivalente a um atributo categorizado multivalorado, e o domínio deste atributo é o conjunto de todos os nomes de autores.

Vamos agora formalizar a interpretação de um documento XML como um conjunto de atributos categorizados multivalorados. Inicialmente, considere a existência de um operador Ψ que, quando aplicado a $\mathcal{A}_{desc}$, resulta em uma *seqüência* de subárvores de $\mathcal{A}_{desc}$. Além disso, cada subárvore dessa seqüência é rotulada com um número inteiro sequencial. Isto posto, dada uma coleção de documentos XML, representamos os atributos de cada documento d_i pela tupla $T_{d_i} = (att_1, att_2, ..., att_{|\mathcal{A}_{desc}|})$. Cada atributo att_k é um par $[f_1(k), f_2(d_i, k)]$, onde a função $f_1(\cdot)$ retorna o rótulo do k-ésimo elemento na seqüência gerada por $\Psi(\mathcal{A}_{desc})$, e a função $f_2(\cdot, \cdot)$ retorna o conjunto V_{ik} de valores correspondentes ao k-ésimo elemento de $\Psi(\mathcal{A}_{desc})$ que ocorre no documento d_i ($V_{ik} \subset Dom(att_k)$).

Para exemplificar a construção da tupla T_{d_i} correspondente a um documento d_i da coleção, considere novamente o documento exibido na Figura 4.1. A Figura 4.3 apresenta a tupla T_{d_i} referente a este documento, quando os elementos de $\mathcal{A}$ cujas raízes são os elementos XML de rótulos *AuthorList*, *PublicationTypeList* e *MeshHeadingList* são consideradas como elementos do conjunto $\mathcal{A}_{desc}$. Note que cada componente de T_{d_i} corresponde a um par (att_k, V_{ik}), onde att_k é o atributo multivalorado corresponde à raiz da subárvore correspondente na seqüência $\Psi(\mathcal{A}_{desc})$, e V_{ik} é o

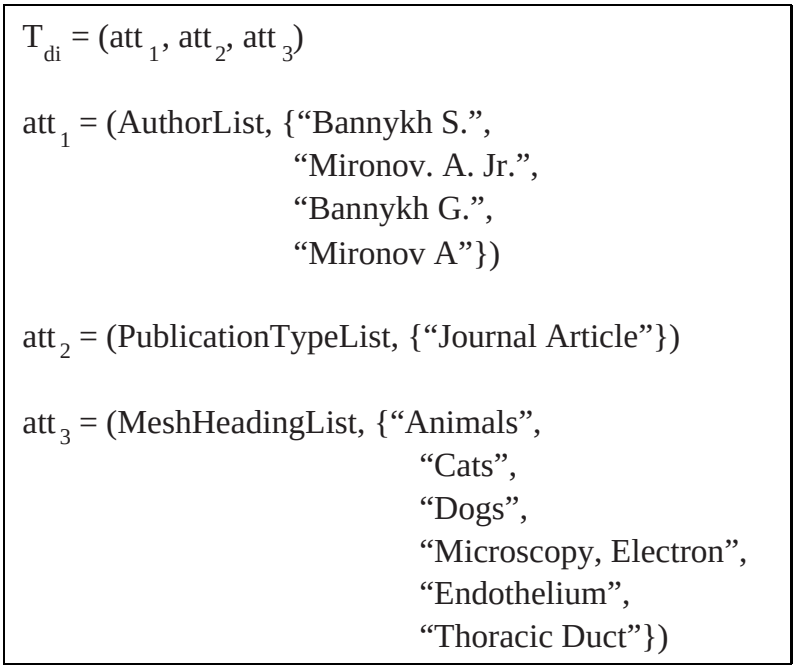


Figura 4.3: Tupla de atributos categorizados multivalorados extraída a partir de um documento XML

conjunto de valores do atributo multivalorado att_k que são usados para *anotar* d_i. Note também que, quando consideramos como d_i o documento XML apresentado na Figura 4.1, e se att_k =*MeshHeadingList*, então V_{ik} = {“Animals”, “Cats”, “Dogs”, “Endothelium”, “Thoracic Duct”, “Microscopy, Electron”}.

4.2.3 Novas Possibilidades para Definição de Restrições

A representação dual proposta nesta Seção resulta na possibilidade de interpretar e manipular cada documento centrado em texto de duas formas. Primeiramente, um documento da coleção pode ser manipulado como um vetor de termos, da mesma forma que é definido no modelo de espaço vetorial (vide Seção 3.2.3). Em segundo lugar, esse mesmo documento pode ser manipulado como um conjunto de características (atributos) multivaloradas, sendo que essas características são extraídas a partir da informação estrutural existente no esquema da coleção de documentos XML. Sendo assim, temos que:

$$d_i = \langle \mathcal{A}_{vsm}, \mathcal{A}_{desc} \rangle \tag{4.1}$$

Agora, considere novamente as definições de restrições positivas e negativas dadas na Seção 3.3.3. Note que essas restrições somente podem ser aplicadas a identificadores de documentos da coleção. Ou seja, quando utilizados em $\mathrm{ML}(d_i, d_j)$ ou $\mathrm{CL}(d_i, d_j)$, d_j e d_i são identificadores do i-ésimo e do j-ésimo documentos da coleção. Denominamos esse tipo de restrição de *restrições doc-doc*.

Por outro lado, se adotamos a representação correspondente à Equação 4.1, surgem novas possibilidades de definições de restrições além das atualmente possíveis entre identificadores de documentos. Em particular, todos os atributos do conjunto $\mathcal{A}_{desc}$ podem ser utilizados para definições de restrições. A seguir, enumeramos e damos exemplos dos cenários possíveis de definições de restrições positivas e negativas, uma vez que a perspectiva dual correspondente à Equação 4.1 é adotada para cada documento XML da coleção.

1. *Restrições doc-doc*: a possibilidade de especificar restrições positivas e negativas sobre um par de documentos permanece em nossa representação dual. Por exemplo, o usuário pode definir que dois documentos cujos identificadores são d_3 e d_{14} devem permanecer em grupos diferentes: $\mathrm{CL}(d_3, d_{14})$.

2. *Restrições meta-meta*: o usuário pode também definir restrições somente ao nível de metadados, com atributos derivados do conjunto $\mathcal{A}_{desc}$. Por exemplo, considere um atributo att_k extraído de $\mathcal{A}_{desc}$, onde v_a e v_b são dois dos valores em $\mathrm{Dom}(att_k)$. O usuário pode agora definir que documentos anotados com o valor v_a e documentos anotados com o valor v_b devem permanecer no mesmo grupo, isto é, $\mathrm{ML}(v_a, v_b)$.

3. *Restrições doc-meta*: agora o usuário pode definir restrições no nível de metadados (ou nível estrutural) e combiná-las com restrições ao nível de documentos. Por exemplo, dados dois autores identificados por a_1 e a_{10}, e os documentos d_2 e d_8, o usuário pode definir que $\mathrm{ML}(d_2, d_8) \wedge \mathrm{ML}(a_1, a_{10})$.

Desejamos enfatizar que a representação que adotamos é relevante para diversos domínios de aplicação que trabalham com documentos textuais semi-estruturados

em geral, e com documentos XML centrados em texto em particular. Por exemplo, em bibliotecas digitais, é comum encontrar informações descritivas associadas às obras (artigos, livros, etc.) catalogadas. Autores, palavras-chave de um conjunto predefinido, e conceitos de uma taxonomia são exemplos de informação descritiva que freqüentemente estão associadas aos documentos em tais repositórios. Esta informação descritiva é representada em nossa abordagem pelo conjunto $\mathcal{A}_{desc}$.

Em adição às novas possibilidades de definição de restrições, outro aspecto que é relevante para nossa tese é o de que há um mapeamento entre cada atributo categorizado multivalorado att_k (extraído de $\mathcal{A}_{desc}$) e o conjunto $\{d_i\}$ de documentos que são anotados com valores de att_k. Denotamos por $Docs(v, k)$ o conjunto de documentos associados ao valor v de um atributo att_k. Inversamente, podemos também obter a lista de valores do atributo att_k que é usada para anotar um documento d_i. Denotamos por V_{ik} o conjunto de valores de att_k que são usados para anotar o documento d_i. Por sua importância para nossa proposta de tese, definimos esses conjuntos a seguir. Este mapeamento em duas vias representado por $Docs(v, k)$ e V_{ik} é explorado na próxima Seção.

Definição 4.3 *$Docs(v, k)$. Dados um atributo multivalorado att_k e um valor $v \in Dom(att_k)$, definimos $Docs(v, k)$ como o conjunto cujos elementos são todos os documentos anotados com o valor v.*

Definição 4.4 *V_{ik}. Dados um documento d_i e um atributo multivalorado att_k, definimos V_{ik} como o conjunto de valores do atributo att_k utilizados para anotar d_i.*

4.3 Uso de Metadados XML em Agrupamento Semi-Supervisionado

A representação dual de documentos XML descrita na Seção 4.2 abre novas possibilidades para a definição de restrições. Essa representação também torna a informação estrutural sobre os documentos disponível para a execução do próprio algoritmo de agrupamento. Com isso, em vez de somente trabalhar sobre o conteúdo textual dos

documentos XML centrados em texto, um algoritmo de agrupamento pode também consultar os metadados associados à coleção. Exploramos esse aspecto nas seções seguintes, cujos assuntos são resumidos a seguir. Na Seção 4.3.1, descrevemos como um conjunto de restrições definidas (ao nível de metadados) pelo usuário pode ser usado para inferir restrições ao nível de documentos. Conforme detalhado mais adiante, esse procedimento de inferência de restrições tem o pontencial de gerar informações inconsistentes. Por conta disso, propomos alguns algoritmos para eliminar restrições inconsistentes na Seção 4.3.2. Finalmente, definimos nosso algoritmo para agrupamento semi-supervisionado baseado em restrições, o *MAC-KMeans*, na Seção 4.3.3.

4.3.1 Expansão de Restrições Definidas pelo Usuário

Conforme descrito na Seção 1.2, uma dificuldade prática na tarefa de agrupamento semi-supervisionado é que, em domínios de aplicação complexos, o usuário deve obrigatoriamente fornecer uma quantidade relativamente grande de supervisão para que o algoritmo possa alcançar uma boa qualidade no agrupamento. (Na verdade, esse é um problema da tarefa de aprendizado semi-supervisionado como um todo.) Em [6] e em [7], são apresentadas curvas de aprendizado nas quais os resultados apresentam execuções dos respectivos algoritmos de agrupamento semi-supervisionado correspondentes a 500 e até 1.000 restrições. Embora os resultados obtidos nesses trabalhos tenham sido bem melhores (do ponto de vista dos agrupamentos gerados) quando comparados aos produzidos pela versão não-supervisionada do algoritmo, não é razoável assumir que o usuário esteja disposto a fornecer tamanha quantidade de pares de restrições. A situação se agrava quando pensamos em um cenário de aplicação no qual um mecanismo de busca de informações utiliza um algoritmo de agrupamento para organizar os documentos relevantes a uma consulta antes de apresentá-los ao usuário.

O objetivo desta Seção é definir um procedimento que, a partir de um conjunto $\mathcal{U}$ de restrições *meta-meta* definidas pelo usuário, pode *inferir* $\mathcal{S}$ a partir de $\mathcal{U}$, onde $\mathcal{S}$ é o conjunto de restrições ao nível de documentos correspondentes às restrições

Algoritmo: *InferirRestriçõesDocDoc*

Entrada: conjunto $\mathcal{U} = \mathcal{U}_{ML} \cup \mathcal{U}_{CL}$ de restrições meta-meta definidas pelo usuario

Resultado: conjunto $\mathcal{S}$ de restrições doc-doc inferidas

Passos:
1. $\mathcal{S} \leftarrow \emptyset$

2. **Para cada** restrição $r(v_a, v_b) \in \mathcal{U}$, **faça**
    2.1. **Se** r.tipoLigação = POSITIVA, **então**
        2.1.1. $\mathcal{S} \leftarrow \mathcal{S} \cup \{ML(d_i, d_j) \mid d_i, d_j \in Docs(v_a, k), d_i \neq d_j\}$
        2.1.2. $\mathcal{S} \leftarrow \mathcal{S} \cup \{ML(d_i, d_j) \mid d_i, d_j \in Docs(v_b, k), d_i \neq d_j\}$
        2.1.3. $\mathcal{S} \leftarrow \mathcal{S} \cup \{ML(d_i, d_j) \mid d_i \in Docs(v_a, k), d_j \in Docs(v_b, k)\}$
    2.2. **Se** r.tipoLigação = NEGATIVA, **então**
        2.2.1. $V_1 \leftarrow Docs(v_a, k) - Docs(v_b, k)$
        2.2.2. $V_2 \leftarrow Docs(v_b, k) - Docs(v_a, k)$
        2.2.3. $\mathcal{S} \leftarrow \mathcal{S} \cup \{CL(d_i, d_j) \mid d_i \in V_1, d_j \in V_2, d_i \neq d_j\}$

Figura 4.4: Algoritmo *InferirRestriçõesDocDoc*: infere restrições ao nível de documentos a partir das restrições ao nível de metadados definidas pelo usuário.

em $\mathcal{U}$. Além disso, dependendo do atributo selecionado como base para definição, os conjuntos $\mathcal{U}$ e $\mathcal{S}$ são tais que $Card(\mathcal{U}) \ll Card(\mathcal{S})$. O procedimento de inferência de restrições *doc-doc* a partir de restrições *meta-meta* é o que chamados de *inferência de restrições*. O algoritmo que implementa esse procedimento, *InferirRestriçõesDocDoc*, é apresentado na Figura 4.4. Para iniciar a descrição desse algoritmo, observe que $\mathcal{U}$ possui dois subconjuntos disjuntos, $\mathcal{U}_{ML}$ e $\mathcal{U}_{CL}$, que correspondem a restrições *meta-meta* positivas e negativas, respectivamente.

Primeiramente, descrevemos a parte do algoritmo *InferirRestriçõesDocDoc* relativa à expansão de restrições positivas (vide passo 2.1). Note que essa parte envolve dois tipos de inferência, que chamamos de *interna* e *cruzada*. Seja descrever esses tipos de inferência. Considere, para isso, que o usuário definiu uma restrição sobre o par (v_a, v_b), $v_a, v_b \in Dom(att_k)$. Na inferência interna, cada um dos dois valores do atributo att_k é considerado separadamente: dado um valor v, são inferidas res-

trições *doc-doc* correspondentes através dos elementos do conjunto $Docs(v, k)$. Já na inferência cruzada, os dois valores do atributo att_k envolvidos na restrição são usados. Neste caso, as restrições *doc-doc* inferidas estão sempre contidas no conjunto $\{(d_i, d_j) \mid d_i \in Docs(v_a, k), d_j \in Docs(v_b, k)\}$. A partir de $\mathcal{U}_{ML}$, é possível inferir restrições ao nível de documentos através dos conjuntos $Docs(v, k)$, onde v é algum valor que ocorre em pelo menos uma das restrições que são elementos de $\mathcal{U}_{ML}$. Essa geração pode ser descrita da seguinte maneira. Dada uma restrição $ML(v_a, v_b) \in \mathcal{U}_{ML}$, o algoritmo primeiramente gera restrições *internas* correspondentes sobre os elementos dos conjuntos $Docs(v_a, k)$ e $Docs(v_b, k)$, separadamente. Em seguida, o algoritmo gera restrições *externas*, desta vez considerando os conjuntos $Docs(v_a, k)$ e $Docs(v_b, k)$ simultaneamente.

Agora vamos descrever a parte do algoritmo *InferirRestriçõesDocDoc* relativa à inferência de restrições negativas (vide passo 2.2). Há dois aspectos importantes com respeito a essa inferência. Em primeiro lugar, não há sentido em inferir restrições internas para acrescentar ao conjunto $\mathcal{S}$. Outro aspecto é que nem todos os pares do conjunto $\{(d_i, d_j) \mid d_i \in Docs(v_a, k), d_j \in Docs(v_b, k)\}$ são considerados. Os pares que pertencem à interSeção $Docs(v_a, k) \cap Docs(v_b, k)$ não são usados na inferência. Isso porque sua utilização resultaria na geração de restrições inconsistentes pelo algoritmo *InferirRestriçõesDocDoc*. Para exemplificar esse segundo aspecto, a Figura 4.5 apresenta um diagrama de *Venn* com dois conjuntos de documentos (parte esquerda da figura) e as restrições negativas inferidas pelo algoritmo a partir desses conjuntos (parte direita da figura). Note que os documentos d_5 e d_7 (pertencentes à interseção dos conjuntos $Docs(v_a, k) = \{d_1, d_2, d_5, d_7\}$ e $Docs(v_b, k) = \{d_3, d_4, d_5, d_7\}$) não são usados na expansão.

Desejamos agora dar uma idéia acerca da quantidade de restrições *doc-doc* que podem ser inferidas a partir de um conjunto de restrições *meta-meta* definido sobre um atributo att_k. Fazemos isso com o objetivo de enfatizar que, dependendo do atributo selecionado para definição do conjunto $\mathcal{U}$ de restrições *meta-meta*, a quantidade de restrições *doc-doc* inferidas pode ser bem maior que a cardinalidade de $\mathcal{U}$. Sejam v_a e v_b dois valores no domínio de att_k. Seja também $p = |Docs(v_a, k)|$

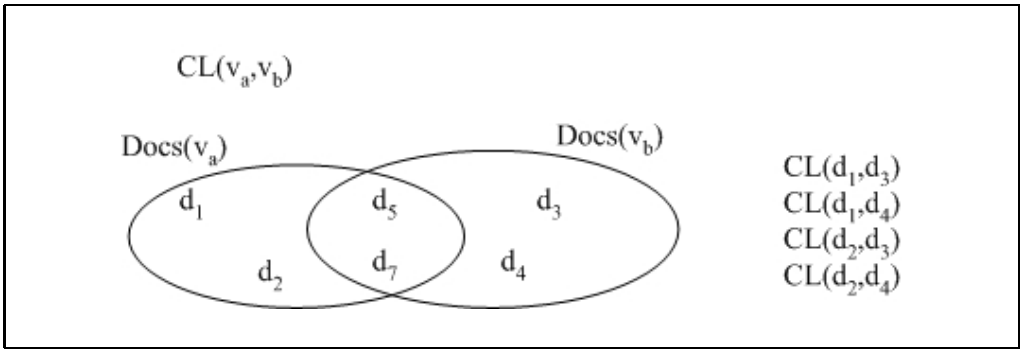


Figura 4.5: Exemplo de restrições negativas inferidas em uma iteração do algoritmo *InferirRestriçõesDocDoc*

e $q = |Docs(v_b, k)|$. No caso de uma restrição *meta-meta* ML(v_a, v_b), a quantidade mínima de restrições inferidas é obtida quando $Docs(v_a, k) \subseteq Docs(v_b, k)$ ou quando $Docs(v_a, k) \supseteq Docs(v_b, k)$. Por outro lado, a quantidade máxima de restrições inferidas é obtida quando $Docs(v_a, k) \cap Docs(v_b, k)$ é igual ao conjunto vazio. Portanto, os valores N^{ML}_{upper} e N^{ML}_{lower} dos limites superior e inferior podem ser obtidos pela Equação 4.2 e pela Equação 4.3, respectivamente. Na Equação 4.2, o primeiro termo do lado direito corresponde às restrições cruzadas, enquanto que o segundo e terceiro termos correspondem às restrições internas. Já na Equação 4.3, somente são consideradas as restrições internas, visto que não há possibilidade de geração de restrições cruzadas.

$$N^{ML}_{upper} = p \times q + \begin{pmatrix} p \\ 2 \end{pmatrix} + \begin{pmatrix} q \\ 2 \end{pmatrix} \tag{4.2}$$

$$N^{ML}_{lower} = \max(\begin{pmatrix} p \\ 2 \end{pmatrix}, \begin{pmatrix} q \\ 2 \end{pmatrix}) \tag{4.3}$$

No caso de uma restrição meta-meta CL$(v_a,\ v_b)$, não há sentido em derivar restrições internas ao nível de documentos, a partir de v_a (ou de v_b) unicamente; o sistema pode somente derivar restrições cruzadas ao nível de documentos. Dessa forma, os valores N^{CL}_{upper} e N^{CL}_{lower} dos limites superior e inferior podem ser obtidos pelas Equações 4.4 and 4.5, respectivamente.

$$N_{upper}^{CL} = p \times q \tag{4.4}$$

$$N_{lower}^{CL} = 0 \tag{4.5}$$

Um aspecto importante do algoritmo *InferirRestriçõesDocDoc* é que ele infere restrições sem levar em consideração o fato de que o conjunto $\mathcal{S}$ resultante pode ser *inconsistente*. Para definir o conceito de *conjunto inconsistente de restrições*, precisamos primeiramente descrever uma propriedade da inferência de restrições *doc-doc* positivas: dada uma restrição $ML(v_a, v_b)$, *InferirRestriçõesDocDoc* gera restrições *doc-doc* positivas da forma $ML(d_i, d_j)$, onde $(d_i, d_j) \in Docs(v_a, k)XDocs(v_b, k)$ e $d_i \neq d_j$. De acordo com essa propriedade, podemos definir a *vizinhança positiva* gerada por uma restrição *meta-meta*, o que fazemos a seguir.

Definição 4.5 ***Vizinhança positiva gerada a partir de uma restrição meta-meta***. *Seja $r = ML(v_a, v_b)$ uma restrição meta-meta. Seja $\mathcal{R}_{ML}^{ab}$ o conjunto de restrições positivas inferido a partir de r. Chamamos de* ***vizinhança positiva*** *o conjunto de documentos que participam de ao menos uma restrição do conjunto $\mathcal{R}_{ML}^{ab}$. Denotamos esse conjunto por $\mathcal{V}(r)$.*

De acordo com a definição 4.5, temos uma quantidade de vizinhanças igual à quantidade de restrições *meta-meta* positivas definidas pelo usuário. Podemos agora definir o conceito de conjunto inconsistente de restrições.

Definição 4.6 ***Conjunto inconsistente de restrições doc-doc***. *Considere os conjuntos de restrições $\mathcal{U}$ e $\mathcal{S}$, onde $\mathcal{S} = \mathcal{S}_{ML} \cup \mathcal{S}_{CL}$ é obtido a partir da aplicação do algoritmo InferirRestriçõesDocDoc sobre $\mathcal{U}$. O conjunto $\mathcal{S}$ é* ***inconsistente*** *se existe ao menos um par de documentos d_i e d_j tais que $d_i, d_j \in \mathcal{V}(r)$, para alguma resrição $r \in \mathcal{U}$, e $CL(d_i, d_j) \in \mathcal{S}_{CL}$.*

Note que a definição 4.6 não considera restrições triviais da forma $CL(d_i, d_i)$, visto que o algoritmo *InferirRestriçõesDocDoc*, por construção, não permite a geração desse tipo de inconsistência.

O fato de o conjunto $\mathcal{S}$ ser inconsistente pode ser analisado graficamente se interpretarmos $\mathcal{S}$ como um grafo não-direcionado. Nesse grafo, os vértices são os documentos que são referenciados por pelo menos uma restrição em $\mathcal{S}$. Além disso, existe uma aresta entre os documentos d_i e d_j se e somente se existe uma restrição (positiva ou negativa) pertencente a $\mathcal{S}$ envolvendo d_i e d_j. Nessa interpretação, o conjunto de restrições *doc-doc* inferido a partir de uma restrição $r = ML(v_a, v_b)$ corresponde a um *grafo completo*, onde os vértices são documentos em $\mathcal{V}(r)$, e onde as arestas correspondem às restrições positivas inferidas a partir de r pelo algoritmo *InferirRestriçõesDocDoc*.

Para exemplificar, a Figura 4.6 apresenta um grafo não-direcionado que representa restrições inferidas a partir do conjunto $\mathcal{U} = \{ML(v_a, v_b),\ CL(v_b, v_c)\}$. Os conjuntos $Docs(v_a, k)$, $Docs(v_b, k)$ e $Docs(v_c, k)$ são também apresentados. No grafo, arestas sólidas representam restrições *doc-doc* positivas; arestas pontilhadas representam restrições *doc-doc* negativas. O conjunto $\mathcal{S}$ formado a partir de $\mathcal{U}$ é representado pelo grafo. Note, por exemplo, que esse grafo contém uma aresta correspondente a uma restrição negativa envolvendo o par de documentos (d_1, d_3), e que ambos pertencem a uma vizinhança positiva. Portanto, de acordo com a Definição 4.6, o conjunto de restrições correspondente ao grafo da Figura 4.6 é inconsistente. É importante notar também nesse exemplo que o documento d_5 não está envolvido em inconsistência alguma.

Considere o subconjunto $\mathcal{S}_{ML} \in \mathcal{S}$, composto das restrições positivas inferidas por *InferirRestriçõesDocDoc*. Neste ponto, é importante enfatizarmos uma diferença fundamental entre nossa abordagem de agrupamento semi-supervisionado e as de outros trabalhos, em relação a $\mathcal{S}_{ML}$. Nesses trabalhos, a propriedade do *fechamento transitivo* do conjunto $\mathcal{S}_{ML}$ é considerada. Ou seja, considera-se que $\mathcal{S}_{ML}$ forma uma *relação de equivalência* sobre os documentos, e que, por conseqüência, o conjunto de restrições pode ser aumentado pela geração do seu fechamento transitivo. (Descrevemos isso na Seção 3.4.1.) A vantagem da aplicação dessa operação é que a cardinalidade do conjunto de restrições fornecidas ao algoritmo aumenta. Essa aplicação é válida, desde que o conjunto $\mathcal{S}$ seja consistente, o que é considerado em

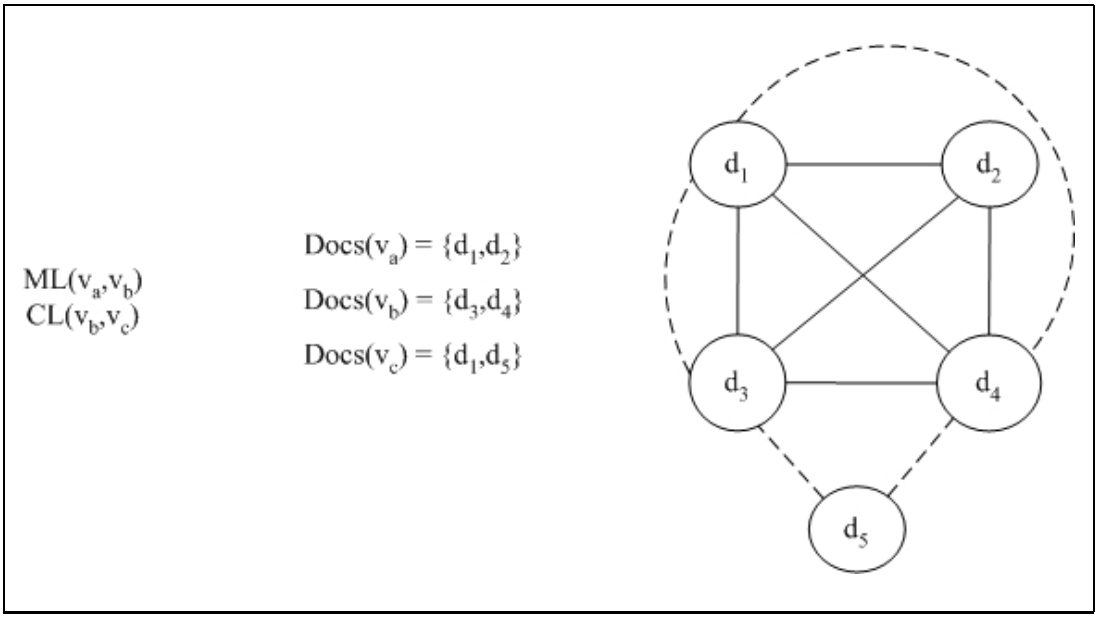


Figura 4.6: Exemplo de inconsistência entre restrições. Nesta Figura, considere por simplicidade que $Docs(v)$ equivale a $Docs(v, k)$, onde k é o índice do atributo multi-valorado em questão.

trabalhos correlatos [68, 1]. Nesta tese, decidimos por *não* aplicar a seqüência de passos acima, nem ao conjunto $\mathcal{U}$, nem ao conjunto $\mathcal{S}$. As razões para essa decisão são descritas a seguir.

Em primeiro lugar, pode haver inconsistências no conjunto de restrições *meta-meta* de entrada, fornecido pelo usuário. Que dizer, em situações práticas, e por vários motivos, o usuário pode fornecer ao sistema um conjunto de restrições *inconsistente*. Como agravante, o próprio procedimento de inferência de restrições *doc-doc* realizado por *InferirRestriçõesDocDoc* também gera inconsistências inerentemente (vide Figura 4.6). Se aplicarmos a operação de fechamento sobre um conjunto inconsistente, certamente obtemos como resultado um conjunto com uma quantidade maior de inconsistências.

Outra razão para não aplicarmos a operação de fechamento diz respeito às restrições *doc-doc* positivas geradas a partir de uma restrição *meta-meta*. Considere que $ML(v_a, v_b)$ e de $ML(v_c, v_d)$ são duas restrições *meta-meta* fornecidas pelo usuário. Já sabemos que os valores v_a, v_b, v_c e v_d estão associados aos conjuntos $Docs(v_a, k)$, $Docs(v_b, k)$, $Docs(v_c, k)$ e $Docs(v_d, k)$, respectivamente (vide Definição 4.3). Consi-

dere também que $\mathcal{R}^{ab}_{ML}$ e $\mathcal{R}^{cd}_{ML}$ são os conjuntos de restrições positivas inferidas a partir de $ML(v_a, v_b)$ e de $ML(v_c, v_d)$, respectivamente. Agora, considere a Figura 4.7, que esquematiza a situação que queremos descrever. Nessa figura, as "nuvens" representam os conjuntos de documentos referenciados por $\mathcal{R}^{ab}_{ML}$ e $\mathcal{R}^{cd}_{ML}$. Conforme esquematizado nessa figura, basta que $\mathcal{R}^{ab}_{ML}$ e $\mathcal{R}^{cd}_{ML}$ possuam apenas um documento d_i em comum para que a operação de fechamento transforme esses dois conjuntos em um único. O conjunto resultante, por sua vez, seria unido a todo conjunto com o qual possuísse algum documento em comum, e assim por diante. Em um caso extremo, todas as restrições positivas poderiam acabar por serem unidas em um único conjunto. Além disso, se houver duas restrições, uma positiva e outra negativa, nas quais participa um documento d_j (que não seja referenciado nos conjuntos $\mathcal{R}^{ab}_{ML}$ e $\mathcal{R}^{cd}_{ML}$), o fechamento sobre $\mathcal{R}^{ab}_{ML}$ e $\mathcal{R}^{cd}_{ML}$ resultaria na criação de novas inconsistências relativas a d_j. Isso porque seriam criadas restrições positivas entre d_j e os documentos referenciados por $\mathcal{R}^{ab}_{ML}$ (pois, conforme a Figura 4.7, d_j participa de uma restrição positiva com algum documento referenciado em $\mathcal{R}^{ab}_{ML}$).

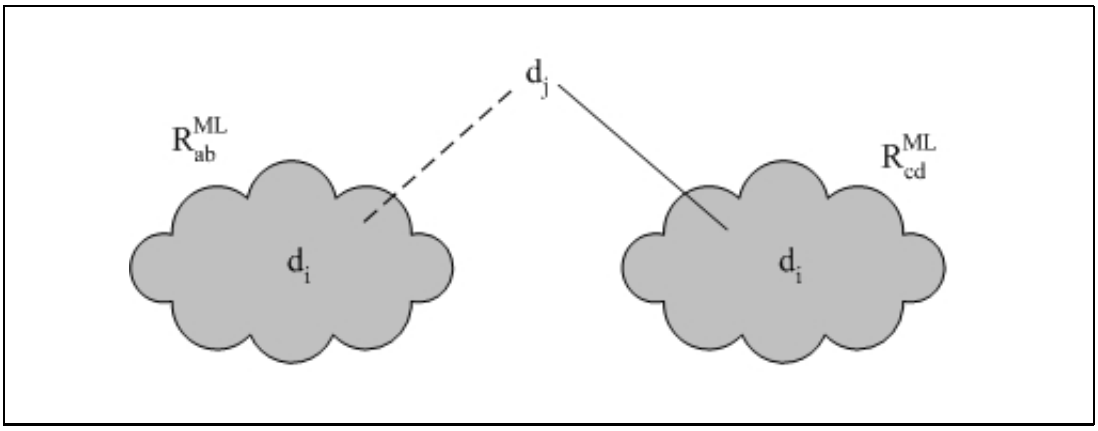


Figura 4.7: Esquema da situação em que novas inconsistências são geradas, se for aplicado o fechamento sobre dois conjuntos de restrições positivas.

A nossa decisão por não utilizar a operação de fechamento evita a criação de novas inconsistências. No entanto, a geração de inconsistências é inerente ao procedimento de inferência representado pelo algoritmo *InferirRestriçõesDocDoc*, que processa cada restrição no conjunto $\mathcal{U}$ de forma separada. Por outro lado, quanto mais inconsistente for o conjunto de restrições utilizado como entrada para o al-

goritmo de agrupamento semi-supervisionado, menor é o aumento da qualidade do modelo de agrupamento. Precisamos, portanto, de algum tratamento que possa ser aplicado ao conjunto de restrições inferido com o objetivo de minimizar o impacto das inconsistências sobre o resultado do agrupamento. Deixamos a descrição desse tratamento para a próxima Seção.

4.3.2 Tratamento de Restrições Inconsistentes

Conforme descrito na Seção 3.4.3, alguns trabalhos em agrupamento semi-supervisionado supõem que existe consistência entre as restrições. Como não assumimos esta premissa, nesta Seção descrevemos nossa abordagem para tratamento de um conjunto inconsistente de restrições. Uma primeira solução que pode ser dada é definir um algoritmo para *eliminar* as inconsistências do conjunto $\mathcal{S}$ inferido a partir de $\mathcal{U}$. A proposição a seguir serve como ponto de partida para elaboração desse algoritmo.

> **Proposição (Condição necessária para existência de inconsistência)**. *Seja o conjunto* $\mathcal{S} = \mathcal{S}_{ML} \cap \mathcal{S}_{CL}$ *de restrições* doc-doc *positivas e negativas. Sejam os conjuntos* $\mathcal{D}_{ML}$ *e* $\mathcal{D}_{CL}$ *de documentos que são referenciados em* $\mathcal{S}_{ML}$ *e* $\mathcal{S}_{CL}$*, respectivamente. Seja ainda o conjunto* $\mathcal{D}_{MLCL} = \mathcal{D}_{ML} \cap \mathcal{D}_{CL}$*. Se* $\mathcal{S}$ *é inconsistente, então pelo menos uma de suas restrições referencia algum documento em* $\mathcal{D}_{MLCL}$*.*

A proposição acima declara que, em um dado conjunto $\mathcal{S}$ de restrições *doc-doc*, restrições inconsistentes sempre referenciam[1] documentos pertencentes a $\mathcal{D}_{MLCL}$, interseção dos conjuntos de documentos referenciados por restrições tanto positivas, quanto negativas, pertencentes a $\mathcal{S}$. No entanto, note que essa proposição não declara que *todos* os documentos em $\mathcal{D}_{MLCL}$ estão envolvidos em inconsistências.

[1]Dizemos que uma restrição r *referencia* o par de objetos o_i e o_j quando esses dois objetos são utilizados na definição do par não-ordenado associado a r. Ou seja, $r = ML(o_i, o_j)$ ou $r = CL(o_i, o_j)$.

Desta forma, um possível algoritmo para eliminar inconsistências de $\mathcal{S}$ consiste em percorrer os elementos de $\mathcal{D}_{MLCL}$ e, para cada um deles, verificar se estão envolvidos em alguma inconsistência. Se este for o caso, o algoritmo remove a restrição correspondente. Este algoritmo, que chamamos de *EI1*, é apresentado na Figura 4.8.

Algoritmo: *EI1*

Entrada: conjunto de restrições *doc-doc* $\mathcal{S} = \mathcal{S}_{ML} \cup \mathcal{S}_{CL}$
conjunto de restrições *meta-meta* positivas $\mathcal{U}_{ML}$
Resultado: conjunto de restrições $\mathcal{S}_c$, tal que $\mathcal{S}_c \subseteq \mathcal{S}$ e $\mathcal{S}_c$ não possui inconsistências.

Passos:
1. $\mathcal{S}_c \leftarrow \mathcal{S}$
2. $\mathcal{D}_{MLCL} \leftarrow$ conjunto de documentos $\{d_i\}$, onde cada d_i é referenciado por ao menos uma restrição positiva r_1 e ao menos uma restrição negativa r_2, onde $r_1, r_2 \in \mathcal{S}$
3. $\mathcal{R}_{MLCL} \leftarrow$ conjunto de restrições que referenciam ao menos um documento em $\mathcal{D}_{MLCL}$
4. $\{\mathcal{V}_k\}_{k=1..|\mathcal{U}_{ML}|} \leftarrow$ VizinhançasPositivas($\mathcal{U}_{ML}$)
5. **Para cada** restrição $r(d_i, d_j) \in \mathcal{R}_{MLCL}$, **faça**
   Se r.tipoLigação = NEGATIVA **e** $\exists \mathcal{V}_k$ tal que $\{d_i, d_j\} \subset \mathcal{V}_k$, **então**
   $\mathcal{S}_c \leftarrow \mathcal{S}_c - \{r\}$
6. **Retorne** $\mathcal{S}_c$

Figura 4.8: Algoritmo *EI1*: elimina restrições inconsistentes no conjunto de restrições de entrada.

Os passos 1, 2 e 3 do algoritmo *EI1* são auto-explicativos. Já os passos 4 e 5 merecem uma explicação mais detalhada. O passo 4 considera que existe uma função chamada *VizinhançasPositivas*. Dado o conjunto $\mathcal{S}_{ML}$ de restrições *meta-meta* positivas, considere que $m = |\mathcal{S}_{ML}|$. Essa função gera m conjuntos (não necessariamente disjuntos) de documentos. Cada conjunto gerado por esta função corresponde aos documentos em $Docs(v_a, k) \cup Docs(v_b, k)$, onde v_a e v_b são tais que $ML(v_a, v_b) \in \mathcal{S}_{ML}$. O passo 5 corresponde a um laço que percorre cada restrição $r(d_i, d_j)$ que referencia ao menos um documento no conjunto $\mathcal{D}_{MLCL}$. Se r for

uma restrição negativa, e se tanto d_i quanto d_j pertencem a algum dos conjuntos gerados pela função *VizinhançasPositivas*, r é removida do conjunto de restrições consistentes.

Note que o *EI1* tem como característica eliminar restrições negativas em favorecimento das restrições positivas. Para entender isso, note que, uma vez que detecta uma inconsistência em alguma vizinhança positiva, o algoritmo sempre remove a restrição negativa participante dessa inconsistência. Uma desvantagem relativa a essa característica do *EI1* é que ele pode estar removendo justamente a restrição negativa que é útil para a necessidade do usuário, e conseqüentemente a restrição positiva indesejável acaba por permanecer no conjunto de restrições inferidas.

O algoritmo *EI2*, que apresentamos na Figura 4.9, é similar ao *EI1*. A diferença é que, quando o *EI2* identifica um par de documentos d_i e d_j em uma vizinhança positiva para o qual existem as restrições $r1 = ML(d_i, d_j)$ e $r2 = CL(d_i, d_j)$, tanto $r1$, quanto $r2$ são eliminadas. Assim, diferente do algoritmo *EI1*, *EI2* não favorece nenhum dos dois tipos de restrição.

Os algoritmos *EI1* e *EI2* se baseiam apenas na informação estrutural geradora de inconsistências para decidir pela eliminação de restrições. Em particular, o *EI2* considera o par de restrições indistintamente, e decide por remover as duas restrições. Entretanto, se houver alguma forma de definir um grau de importância ou "peso" para cada restrição, a restrição de maior peso poderia permanecer, e a de menor peso seria eliminada. Desejamos, então, propor uma terceira abordagem para tratamento de inconsistências que considere a eliminação de restrições com base no conceito de *peso* de uma restrição, o que definimos a seguir.

Para cada restrição que o usuário fornece ao sistema, ele/ela o faz com algum grau de certeza. Chamamos esse grau de certeza de *peso* da restrição. Uma possibilidade para definir o peso de cada restrição é fazer com que o sistema consulte o usuário sobre este valor. Esta é a solução adotada em [66] (vide Seção 3.4.3). Esta solução, no entanto, diminui a autonomia do sistema. Em vez disso, nossa terceira abordagem para tratamento de inconsistências consiste em atribuir ao sistema a responsabilidade de gerar um peso para cada uma das restrições *doc-doc* inferidas.

Algoritmo: *EI2*

Entrada: conjunto de restrições *doc-doc* $\mathcal{S} = \mathcal{S}_{ML} \cup \mathcal{S}_{CL}$
conjunto de restrições *meta-meta* positivas $\mathcal{U}_{ML}$
Resultado: conjunto de restrições $\mathcal{S}_c$, tal que $\mathcal{S}_c \subseteq \mathcal{S}$ e $\mathcal{S}_c$ não possui inconsistências.

Passos:
1. $\mathcal{S}_c \leftarrow \mathcal{S}$
2. $\mathcal{D}_{MLCL} \leftarrow$ conjunto de documentos $\{d_i\}$, onde cada d_i é referenciado por ao menos uma restrição positiva r_1 e ao menos uma restrição negativa r_2, onde $r_1, r_2 \in \mathcal{S}$
3. $\mathcal{R}_{MLCL} \leftarrow$ conjunto de restrições que referenciam ao menos um documento em $\mathcal{D}_{MLCL}$
4. $\{\mathcal{V}_k\}_{k=1..|\mathcal{U}_{ML}|} \leftarrow$ VizinhançasPositivas($\mathcal{U}_{ML}$)
5. **Para cada** restrição $r1(d_i, d_j) \in \mathcal{R}_{MLCL}$, **faça**
   Se $r1$.tipoLigação = NEGATIVA **e** $\exists \mathcal{V}_k$ tal que $\{d_i, d_j\} \subset \mathcal{V}_k$, **então**
      5.1. $\mathcal{S}_c \leftarrow \mathcal{S}_c - \{r1\}$
      5.2. Para cada $r2$ tal que $r2$.tipoLigação = POSITIVA e $d_i, d_j \in \mathcal{V}_k$, **faça**
      $\mathcal{S}_c \leftarrow \mathcal{S}_c - \{r2\}$
6. **Retorne** $\mathcal{S}_c$

Figura 4.9: Algoritmo *EI2*: elimina restrições inconsistentes no conjunto de restrições de entrada.

Mais uma vez, procuramos aproveitar informação estrutural associada à coleção de documentos XML a serem agrupados para definir esses pesos. Uma vez que o sistema define esses pesos, ele pode se basear nestes para decidir quais restrições deve eliminar com o objetivo de obter um conjunto consistente. A seguir, descrevemos nossa estratégia para derivação de pesos para as restrições.

Após a aplicação do algoritmo *InferirRestriçõesDocDoc*, o sistema possui dois conjuntos de restrições, $\mathcal{S}_{ML}$ e $\mathcal{S}_{CL}$ (vide Seção 4.3.1). Sem perda de generalidade, considere uma dessas restrições, a que faz referência aos documentos d_i e d_j. De acordo com nossa representação dual de documentos XML descrita na Seção 4.2, há um mapeamento entre d_i e os valores do atributo $att_k \in T_{d_i}, \forall k$ (o mesmo vale para d_j). Portanto, o sistema pode obter os dois conjuntos de valores de att_k, V_{ik} e V_{jk}, que ocorrem nos documentos d_i e d_j, respectivamente (i.e., $V_{ik}, V_{jk} \subset Dom(att_k)$).

Uma hipótese que adotamos é a seguinte: dados dois conjuntos de valores V_{ik} e V_{jk} contidos no domínio do atributo att_k, a similaridade (ou dissimilaridade) entre V_{ik} e V_{jk} tem correlação com o peso da restrição definida sobre o par (d_i, d_j), independentemente de esta restrição ser positiva ou negativa.

Para detalhar essa hipótese, considere que há uma forma de calcular a similaridade (ou dissimilaridade) entre os conjuntos V_{ik} e V_{jk}. Dada uma restrição positiva sobre o par (d_i, d_j), consideramos que quanto mais alta a similaridade entre V_{ik} e V_{jk}, mais alto deve ser o custo de eliminar essa restrição. De forma análoga, se existe uma restrição negativa definida sobre o par (d_i, d_j), quanto mais alta a dissimilaridade entre V_{ik} and V_{jk}, tanto mais alto deve ser a força dessa restrição.

Para exemplificar e ao mesmo tempo dar uma base intuitiva para nossa hipótese, considere o caso específico da coleção MEDLINE, uma coleção de artigos na área biomédica, disponível e consultável em http://www.pubmed.com (vide também a Seção 5.3 e a Figura 4.1). Considere também que um subconjunto dessa coleção está para ser processado por um algoritmo de agrupamento semi-supervisionado. Nessa situação, se dois ou mais documentos ligados por restrições positivas têm listas similares de autores, assumimos esse fato como indício de que essa restrição foi definida por conta dessa similaridade entre as listas de autores. Além disso, se esses mesmos documentos compartilham uma boa proporção de sua lista de descritores, isso é outro indício de que a causa de o usuário ter definido a restrição positiva entre os documentos foi o fato de os documentos terem afinidade em seus descritores. Um raciocínio análogo pode ser feito em relação ao conceito de dissimilaridade.

Conforme descrito nos parágrafos anteriores, assumimos a hipótese de que podemos derivar o peso de uma restrição *doc-doc* envolvendo os documentos d_i e d_j em função da similaridade (ou dissimilaridade) entre os valores dos atributos multivalorados correspondentes a d_i e a d_j. A questão que se segue é a de como calcular a similaridade (ou dissimilaridade) entre duas tuplas T_{d_i} e T_{d_j}. De acordo com a Seção 4.2.2, temos uma tupla T_{d_i} para cada documento d_i da coleção. Lembre-se de que cada componente dessas tuplas são conjuntos de valores no domínio de um atributo categorizado multivalorado. Portanto, várias medidas para cálculo de similaridade

Algoritmo: *GerarPML*

Entrada: $\mathcal{S}_{ML}$

Resultado: uma matriz $PML = [w_{ij}]$ de pesos das restrições positivas

Passos:
1. $w_{ij} \leftarrow 0, \forall i \forall j$
2. **Para cada** par não ordenado $(d_i, d_j) \in \mathcal{S}_{ML}$, **faça**
    Para cada atributo att_k, **faça**
        2.1. $V_{ik} \leftarrow$ conjunto de valores att_k utilizados para anotar d_i
        2.2. $V_{jk} \leftarrow$ conjunto de valores att_k utilizados para anotar d_j
        2.3. $w_{ij} \leftarrow w_{ij} + Sim(V_{ik}, V_{jk})$
        2.4. $w_{ji} \leftarrow w_{ij}$

Figura 4.10: Algoritmo *GerarPML*: gera a matriz PML de pesos de restrições positivas

entre atributos categorizados podem ser aplicadas para obter as similaridades (ou dissimilaridades) entre as tuplas T_{d_i} e T_{d_j}. (Algumas dessas medidas são descritas na Seção 2.2.2.) Desta forma, dada uma restrição positiva (negativa) definida sobre o par de documentos (d_i, d_j), definimos o *peso* w_{ij} ($\overline{w}_{ij}$) dessa restrição como o grau de similaridade (dissimilaridade) entre as tuplas T_{d_i} e T_{d_j}. Ou seja:

$$w_{ij} = Sim(T_{d_i}, T_{d_j}) \tag{4.6}$$

e

$$\overline{w}_{ij} = Dissim(T_{d_i}, T_{d_j}) \tag{4.7}$$

Se considerarmos todos os elementos dos conjuntos $\mathcal{S}_{ML}$ e $\mathcal{S}_{CL}$, duas matrizes simétricas $PML = [w_{ij}]$ e $PCL = [\overline{w}_{ij}]$ podem ser formadas. O elemento w_{ij} corresponde ao peso da restrição positiva definida entre os documentos d_i e d_j (onde $1 \leq i, j \leq |\mathcal{S}_{ML}|$). De forma análoga, o elemento $\overline{w}_{ij}$ corresponde ao peso da restrição negativa definida entre os documentos d_i e d_j (onde $1 \leq i, j \leq |\mathcal{S}_{CL}|$). Os algoritmos *GerarPML* (Figura 4.10) e *GerarPCL* (Figura 4.11) podem ser utilizados

Algoritmo: *GerarPCL*

Entrada: $\mathcal{S}_{CL}$

Resultado: uma matriz $PCL = [\overline{w}_{ij}]$ de pesos das restrições negativas

Passos:
1. $\overline{w}_{ij} \leftarrow 0, \forall i \forall j$
2. **Para cada** par não ordenado $(d_i, d_j) \in \mathcal{S}_{CL}$, **faça**
    Para cada atributo att_k, **faça**
        2.1. $V_{ik} \leftarrow$ conjunto de valores att_k utilizados para anotar d_i
        2.2. $V_{jk} \leftarrow$ conjunto de valores att_k utilizados para anotar d_j
        2.3. $\overline{w}_{ij} \leftarrow \overline{w}_{ij} + Dissim(V_{ik}, V_{jk})$
        2.4. $\overline{w}_{ji} \leftarrow \overline{w}_{ij}$

Figura 4.11: Algoritmo *GerarPCL*: gera a matriz WCL de pesos associados a restrições negativas

para gerar as matrizes *PML* e *PCL*, respectivamente.

Após a geração das matrizes de pesos, o sistema tem informações suficientes para eliminar restrições inconsistentes, através do procedimento descrito a seguir. Considere a Figura 4.12, na qual recorremos novamente à interpretação de um conjunto de restrições como um grafo não-direcionado. A Figura 4.12a ilustra de forma esquemática uma situação geradora de inconsistência em um conjunto de restrições, na qual dois documentos, d_i e d_j, satisfazem *simultaneamente* a duas condições: (1) participam de uma restrição negativa e (2) pertencem a alguma vizinhança positiva. Note que há ciclos envolvendo a aresta correspondente à restrição negativa entre os documentos d_i e d_j. A Figura 4.12b fornece um exemplo dessa situação para uma vizinhança composta de quatro documentos, d_1, d_2, d_3 e d_4.

Uma solução possível para eliminar a inconsistência que está esquematizada na Figura 4.12 consiste em realizar uma de três ações possíveis, que descrevemos a seguir: (1) eliminar a restrição negativa entre d_i e d_j; (2) eliminar as restrições positivas nas quais d_i participa; (3) eliminar as restrições positivas nas quais d_j participa. A realização de uma dessas três ações elimina a inconsistência em questão. A questão principal é justamente saber qual das três ações tomar. Para isso, definimos

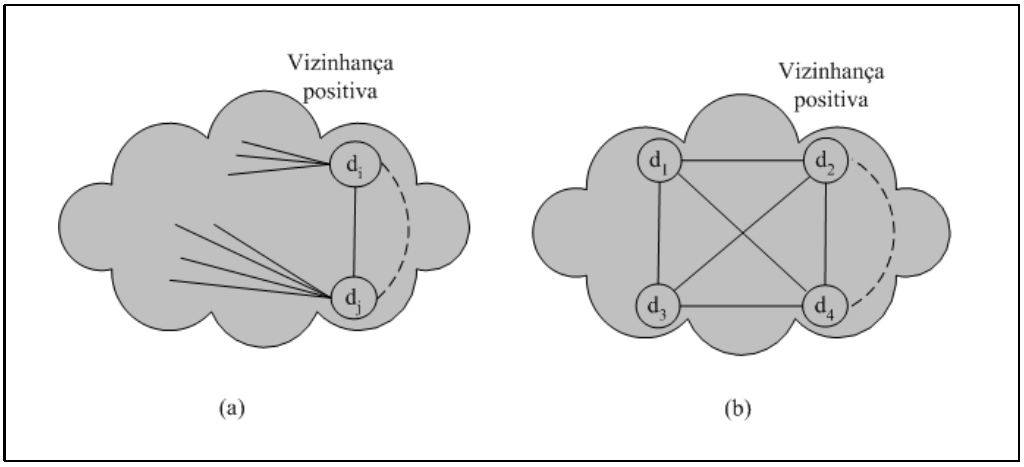


Figura 4.12: (a) Situação em que os documentos d_i e d_j participam de uma inconsistência, pois satisfazem às seguintes condições: participam de uma restrição negativa e pertencem a uma vizinhança positiva. (b) Exemplo dessa situação para uma vizinhança de quatro documentos.

o problema de eliminação de restrições como um problema de *programação linear inteira* (PLI). Para a definição deste problema, considere as variáveis x_{cl}, x^i_{ml} e x^j_{ml} e seus possíveis valores, descritos a seguir.

- x_{cl} é igual a 1 se a restrição negativa deve permanecer; 0 em caso contrário.
- x^i_{ml} é igual a 1 se as restrições positivas nas quais d_i participa devem permanecer; 0 em caso contrário.
- x^j_{ml} é igual a 1 se as restrições positivas nas quais d_j participa devem permanecer; 0 em caso contrário.

Note que as três variáveis acima são inteiras. Uma vez definidas essas três variáveis, podemos definir os dois componentes do problema de PLI: a função objetivo a ser maximizada e as restrições impostas sobre as variáveis x_{cl}, x^i_{ml} e x^j_{ml}. Essa definição é apresentada na Figura 4.13.

No problema apresentado na Figura 4.13, há três coeficientes, um para cada variável envolvida: c_1, c_2 e c_3. O coeficiente c_1 é igual ao valor $\overline{w}_{ij}$ retirado da matriz de pesos WCL, e corresponde ao peso da restrição negativa existente entre d_i e d_j. Já os coeficientes c_2 e c_3 correspondem às médias dos pesos das restrições positivas nas quais os documentos d_i e d_j participam, respectivamente. Os valores de c_2 e de

maximizar $Z = c_1 x_{cl} + c_2 x^i_{ml} + c_3 x^j_{ml}$

sujeito a

$$p = x_{cl} + x^i_{ml} + x^j_{ml}$$

e aos seguintes limites de variáveis

$$2 \leq p \leq 2$$
$$x_{cl} \in \{0, 1\}$$
$$x^i_{ml} \in \{0, 1\}$$
$$x^j_{ml} \in \{0, 1\}$$

Figura 4.13: Definição do problema de PLI associado ao algoritmo *EI3*.

c_3 são obtidos a partir dos pesos da matriz WML. Note que esses valores são fixos para os documentos d_i e d_j. O resultado da solução do problema acima é a definição dos valores das variáveis x_{cl}, x^i_{ml} e x^j_{ml}, e a conseqüente definição de que restrições devem ser eliminadas. Se este problema for resolvido para cada par de documentos envolvidos em uma restrição negativa e que pertençam a uma mesma vizinhança positiva, as inconsistências do conjunto são totalmente eliminadas. Resumimos esse procedimento de eliminação de inconsistência no algoritmo *EI3*, apresentado na Figura 4.14.

A vantagem do algoritmo *EI3* sobre o *EI1* (vide Figura 4.8) é que o primeiro não favorece as restrições positivas indistintamente. Em vez disso, dada uma restrição negativa r envolvida em uma inconsistência, considere que r é composta dos documentos d_i e d_j. Nesta situação, são calculados os pesos de r e de todas as restrições positivas nas quais d_i e d_j participam dentro da vizinhança positiva em questão. Uma vez calculados os pesos, eles são considerados na decisão de que restrições devem ser removidas para eliminação da inconsistência.

4.3.3 Definição do Algoritmo *MAC-KMeans*

Nesta Seção, definimos nosso algoritmo de agrupamento semi-supervisionado. Esse algoritmo, que chamamos de *MAC-KMeans*, é uma extensão do bem conhecido K-Means [50]. Essa extensão diz respeito a dois aspectos do K-Means: definição das

Algoritmo: *EI3*

Entrada: conjunto de restrições *doc-doc* $\mathcal{S} = \mathcal{S}_{ML} \cup \mathcal{S}_{CL}$.

Resultado: conjunto de restrições $\mathcal{S}_c$, tal que $\mathcal{S}_c \subseteq \mathcal{S}$ e $\mathcal{S}_c$ não possui inconsistências.

Passos:
1. $\mathcal{S}_c \leftarrow \mathcal{S}$
2. $\mathcal{D}_{MLCL} \leftarrow$ conjunto de documentos $\{d_i\}$, onde cada d_i é referenciado por ao menos uma restrição positiva r_1 e ao menos uma restrição negativa r_2, onde $r_1, r_2 \in \mathcal{S}$
3. $\mathcal{R}_{MLCL} \leftarrow$ conjunto de restrições que referenciam ao menos um documento em $\mathcal{D}_{MLCL}$
4. $\{\mathcal{V}_k\}_{k=1..|\mathcal{U}_{ML}|} \leftarrow$ VizinhançasPositivas($\mathcal{U}_{ML}$)
5. **Para cada** restrição $r(d_i, d_j) \in \mathcal{R}_{MLCL}$, **faça**
   Se r.tipoLigação = NEGATIVA **e** $\exists \mathcal{V}_k \in \mathcal{V}$ tal que $\{d_i, d_j\} \subset \mathcal{V}_k$, **então**
      5.1. Obter valores x_{cl}, x^i_{ml} e x^j_{ml}, solução do PLI da Figura 4.13
      5.2. **Se** $x_{cl} = 0$, **então**
           $\mathcal{S}_c \leftarrow \mathcal{S}_c - \{r\}$
      Senão se $x^i_{ml} = 0$, **então**
           $\mathcal{S}_c \leftarrow \mathcal{S}_c$ - {Restrições em V_k que referenciam d_i}
      Senão se $x^j_{ml} = 0$, **então**
           $\mathcal{S}_c \leftarrow \mathcal{S}_c$ - {Restrições em V_k que referenciam d_j}
6. Retorne $\mathcal{S}_c$

Figura 4.14: Algoritmo *EI3*: elimina restrições inconsistentes no conjunto de restrições de entrada.

sementes (centróides iniciais dos grupos) e definição da função objetivo. Ambos os aspectos levam em consideração a existência de restrições positivas e negativas. Descrevemos esses dois aspectos a seguir e, ao final desta Seção, apresentamos o pseudocódigo do *MAC-KMeans*.

Definição das Sementes (Inicialização)

Em um algoritmo de agrupamento partitivo como o K-Means, as *sementes* do agrupamento são os centróides iniciais dos k grupos a serem formados. A qualidade dos grupos formados por um algoritmo de agrupamento partitivo depende diretamente

da escolha de boas sementes na fase de inicialização. Uma *boa semente* para um grupo é aquela que reflete tanto quanto possível a distribuições dos pontos daquele grupo. Quanto melhores as sementes do agrupamento, maior é a probabilidade de o algoritmo escapar de um mínimo local correspondente a uma solução ruim. Além disso, a qualidade das sementes também influencia na quantidade de iterações necessárias para convergência do algoritmo. De fato, a importância da inicialização pode ser formalmente justificada através do uso dos limites de Chernoff-Hoeffding (*Chernoff-Hoeffding Bounds*). Dado um agrupamento de pontos em particular (com uma distribuição Gaussiana subjacente) cujo centróide é μ, e dada uma amostra aleatória de tamanho m desse agrupamento cujo centróide é $\bar{\mu}$, a expressão que define os limites de Chernoff-Hoeffding é a seguinte:

$$P((|\mu - \bar{\mu}|) \geq \delta) \leq e^{-\delta^2 \frac{m}{2}} \tag{4.8}$$

Na expressão acima, $\delta \in \Re_+$. Essa expressão pode ser interpretada da seguinte forma: a diferença entre o centróide verdadeiro e o centróide estimado do agrupamento cai exponencialmente conforme aumenta a quantidade de pontos utilizados na inicialização. Nesta Seção descrevemos nossa proposta para inicialização de sementes, através da utilização das vizinhanças positivas (vide Definição 4.5).

Conforme mencionamos na Seção 4.3.1, o sistema pode formar *vizinhanças* (vide Definição 4.5) de documentos, uma para cada restrição *meta-meta* do conjunto $\mathcal{U}_{ML}$ fornecido pelo usuário. Seja η a quantidade de vizinhanças assim formadas. Essas vizinhanças podem ser usadas na fase de definição das sementes do algoritmo de agrupamento, conforme descrevemos a seguir. Seja k a quantidade de grupos a ser formada pelo algoritmo. Por um lado, se $k \leq \eta$, o sistema pode selecionar as k vizinhanças de maior cardinalidade a partir das η vizinhanças originais. Para cada uma dessas k vizinhanças, seu centróide é calculado. Os k centróides assim formados são usados como sementes na inicialização do algoritmo. Por outro lado, se $k > \eta$, o sistema pode aproveitar as η vizinhanças para formar η centróides. Os restantes $k - \eta$ centróides necessários para a inicialização são selecionados aleatoriamente, a partir de perturbações aleatórias do centróide global da coleção de objetos a ser

agrupadas [32].

Definição da Função Objetivo

A função objetivo que utilizamos em nosso algoritmo é modificada para penalizar eventuais violações de restrições. Seja $\mathcal{D} = \{d_i\}$ um conjunto de documentos XML na representação dual (vide Seção 4.2). Além disso, considere que cada grupo a ser formado pelo algoritmo possui um rótulo associado $l \in L$ (onde $L = \{1, 2, \ldots, k\}$). Assuma também que $c : \mathcal{D} \rightarrow L$ é uma função que retorna o rótulo do grupo associado a um certo documento, e que μ_l é o centróide do grupo G_l. A função objetivo que usamos é definida pela Equação 4.9.

$$\begin{aligned} \mathcal{F} \;=\; & \sum_{d_i \in \mathcal{D}} cos(d_i, \mu_{c(d_i)}) \\ & - CustoTotalML \\ & - CustoTotalCL \end{aligned} \tag{4.9}$$

A primeira parte da função objetivo considera a interpretação geométrica para documentos e corresponde à medida de co-seno, definida na Seção 2.2.1 (a versão do K-Means que usa a medida de co-seno como métrica de distância é conhecida como *K-Means esférico* [25]). Nós utilizamos a medida de co-seno (em vez da distância euclideana) devido ao fato de que a primeira é mais adequada para dados esparsos em altas dimensões, como é o caso de documentos textuais.

A segunda e terceira partes da função objetivo correspondem aos custos de violação de restrições. Considere novamente os conjuntos $\mathcal{S}_{ML}$ e $\mathcal{S}_{CL}$ de restrições (positivas e positivas) resultantes do algoritmo *InferirRestriçõesDocDoc*. As Equações 4.10 e 4.11 fornecem os custos totais de violação de restrições positivas e negativas constituintes de $\mathcal{S}_{ML}$ e de $\mathcal{S}_{CL}$, respectivamente.

$$CustoTotalML = \sum_{(d_i, d_j) \in \mathcal{S}_{ML}} w_{ij} \times (1 - I(c(d_i), c(d_j))) \tag{4.10}$$

$$CustoTotalCL = \sum_{(d_i,d_j)\in\mathcal{S}_{CL}} \overline{w}_{ij} \times I(c(d_i), c(d_j)) \tag{4.11}$$

Nas Equações 4.10 e 4.11, $I(x, y)$ é uma função indicadora (isto é, I retorna 1 quando $x = y$, e I retorna 0 quando $x \neq y$). Os valores de w_{ij} e $\overline{w}_{ij}$ são os pesos das restrições positivas e negativas, respectivamente.

O *MAC-KMeans* é semelhante ao algoritmo *PC-KMeans*, definido em [4] (vide Seção 3.4.1). Há três diferenças, entretanto. Em primeiro lugar, em nossa tese, as restrições fornecidas ao algoritmo são resultado dos processos de inferência e de eliminação de inconsistências. Em segundo lugar, o cálculo do co-seno na função objetivo é relativo somente à bolsa de palavras formada pelos elementos em $\mathcal{A}_{vsm}$, em vez de ser relativo ao conteúdo completo de cada documento. Finalmente, o *MAC-KMeans* utiliza *vizinhanças* de documentos para inicialização das sementes, em vez de utilizar *partições*. (Com relação a esta última diferença, vide Seção 3.4.1.)

Pseudocódigo

Apresentamos agora nosso algoritmo para agrupamento semi-supervisionado de documentos, o *MAC-KMeans* (Metadata Aware Constrained K-Means). O pseudocódigo desse algoritmo é apresentado na Figura 4.15. Este algoritmo recebe como entrada um conjunto $\mathcal{D}$ de documentos, dois conjuntos $\mathcal{S}_{ML}$ e $\mathcal{S}_{CL}$ de restrições *doc-doc* positivas e negativas consistentes, e a quantidade k de grupos a ser identificada na coleção. Recebe também referências para as matrizes de pesos PML e PCL, cuja obtenção descrevemos na Seção 4.3.2. A saída deste algoritmo é uma k-partição do conjunto $\mathcal{D}$.

Conforme mencionamos anteriormente, o *MAC-KMeans* é uma extensão do K-Means esférico (que, por sua vez é uma extensão do K-Means clássico). A complexidade do algoritmo K-Means esférico é igual a $O(nkd)$, onde $n = |\mathcal{D}|$, d é a quantidade de termos em $\mathcal{D}$ e k é a quantidade de grupos a ser formada. Uma questão relevante é sobre a complexidade do nosso algoritmo *MAC-KMeans*. Sejam m e c as cardinalidades dos conjuntos $\mathcal{S}_{ML}$ e $\mathcal{S}_{CL}$, respectivamente. No passo do

Algoritmo: *MAC-KMeans*

Entrada: coleção $\mathcal{D}$ de documentos,
conjuntos de restrições *doc-doc* consistentes $\mathcal{S}_{ML}$ e $\mathcal{S}_{CL}$,
PML (matriz de pesos das restrições em $\mathcal{S}_{ML}$),
PCL (matriz de pesos das restrições em $\mathcal{S}_{CL}$),
quantidade de grupos k.

Resultado: uma k-partição de $\mathcal{D}$: $\{G_1, G_1, \ldots, G_k\}$

Passos:
1. *Inicialização*: forme k centróides iniciais μ_i
   utilizando as vizinhanças formadas com o conjunto $\mathcal{S}_{ML}$

2. Repita até que os centróides não mais se modifiquem:
   2.1. $\forall d_i \in \mathcal{D}$, associe d_i ao grupo G_l tal que $\mathcal{F}$ seja maximizada
   2.2. Atualize o centróide de cada grupo: $\mu_l = \frac{1}{|G_l|} \sum_{d_i \in G_l} d_i$

Figura 4.15: Algoritmo *MAC-KMeans*: extensão do K-Means para agrupamento semi-supervisionado de documentos XML

MAC-KMeans correspondentes à associação de documentos aos centróides, a complexidade é $O(m + c)$. Visto que a quantidade $m + c$ tem um limite superior igual a $n(n-1)/2$, a complexidade de uma iteração do algoritmo é $O(nkd + n^2)$. Isso faz com que o *MAC-KMeans* tenha uma complexidade assintótica igual a $O(n^2)$, que é *pior* que a complexidade do algoritmo K-Means esférico. No entanto, em nossos experimentos (vide capítulo 5), constatamos que a quantidade de iterações do *MAC-KMeans* é consistentemente menor que a do K-Means esférico, o que resulta em um tempo de execução *maior* para este último.

4.4 Arquitetura do Sistema de Agrupamento Semi-Supervisionado

Esta Seção apresenta uma visão sistêmica de nossa tese. Aqui descrevemos uma arquitetura para um sistema que implemente os procedimentos relativos a agrupa-

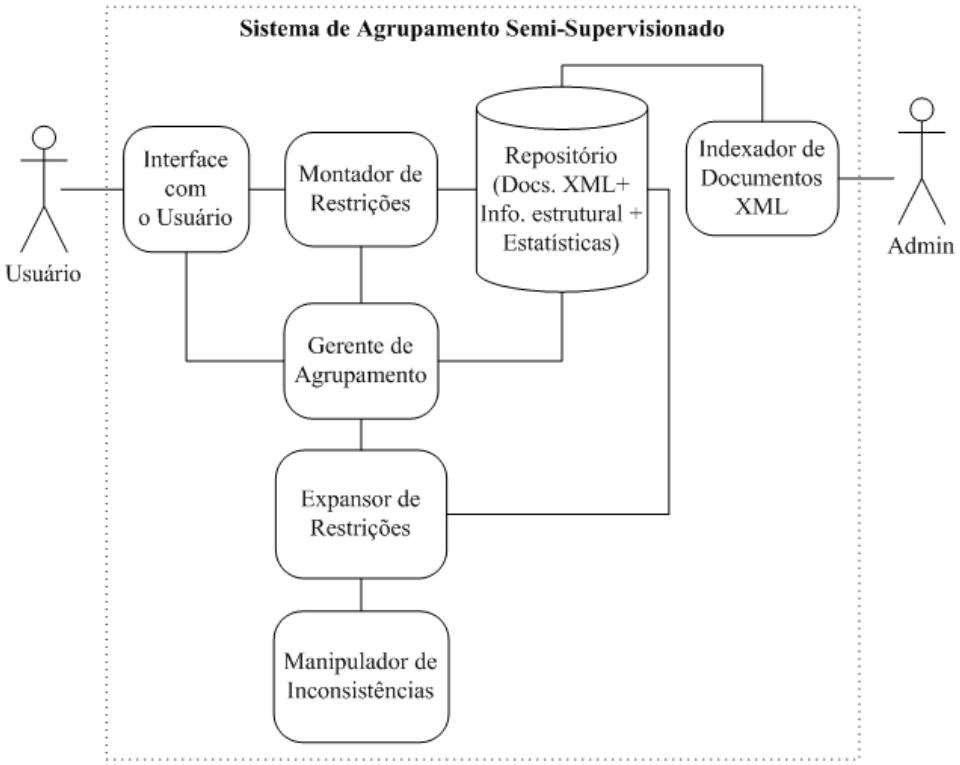


Figura 4.16: Arquitetura para um sistema de agrupamento semi-supervisionado de documentos XML

mento semi-supervisionado descritos neste capítulo. A Figura 4.16 apresenta esquematicamente os módulos desse sistema. Os parágrafos a seguir descrevem cada um desses módulos.

O módulo *Indexador de Documentos XML* (IDX) tem a responsabilidade de processar documentos neste formato para produzir uma representação dual dos mesmos. Note que a arquitetura pressupõe a existência de intervenção humana (desempenhando o papel de administrador do sistema) durante o processo de indexação. O usuário administrador deve definir quais são os elementos XML de uma coleção que devem ser utilizados como as subárvores correspondentes aos elementos dos conjuntos $\mathcal{A}_{vsm}$ e de $\mathcal{A}_{desc}$ (vide Seção 4.2). Após o usuário administrador fazer essa definição, o módulo indexador processa os documentos para extrair a representação dual. Após essa indexação, o repositório contém os documentos XML, além de conter resumos estatísticos sobre os valores dos atributos multivalorados correspondentes a $\mathcal{A}_{desc}$. Conforme descrito anteriormente neste capítulo, esses resumos estatísticos

são necessários para o cálculo dos pesos associados às restrições inferidas (vide Seção 4.3.2).

Durante uma sessão de uso do sistema, em que o usuário está realizando a análise exploratória dos dados (documentos), é possível que ele requisite ao sistema a execução do algoritmo de agrupamento diversas vezes. Em cada requisição, o usuário deve fornece um conjunto não vazio de restrições. O módulo *Montador de Restrições* (MR) corresponde à parte do sistema que tem a responsabilidade de selecionar pares de objetos do repositório, e apresentá-los ao usuário para que este rotule cada par como uma restrição positiva ou negativa. Note que o usuário pode desistir de fornecer restrições a qualquer momento durante o processo de apresentação de pares, e requisitar ao sistema a execução do algoritmo com as restrições fornecidas até o momento. Note ainda que, antes de iniciar a apresentação de pares ao usuário, o sistema precisa consultar esse usuário acerca de qual atributo (retirado de $\mathcal{A}_{desc}$) deve ser utilizado nessa definição. Os valores pertencentes ao domínio do atributo selecionado pelo usuário são usados pelo sistema para a montagem do pares. Por exemplo, considerando a estrutura da coleção à qual pertence o documento da Figura 4.1, esse presente módulo poderia apresentar ao usuário a lista de atributos disponíveis (selecionados pelo usuário administrador com a utilização do módulo IDX): *AuthorsList*, *MeshHeadingList* e *PublicationType*. Após o usuário selecionar um desses atributos, o módulo MR teria condições de iniciar a apresentação dos pares ao usuário, para que este defina o tipo de restrição existente em cada par. Consideramos também que, durante este processo de apresentação, o usuário pode responder que não sabe definir uma restrição sobre um ou mais dos pares apresentados. Neste caso, o sistema simplesmente ignora aquele par e seleciona outro para apresentação.

Uma vez que o usuário responde a um conjunto de consultas acerca do tipo de cada par de objetos apresentado, é o momento de o sistema inferir as restrições *doc-doc* (considerando que o usuário definiu uma ou mais restrições ao nível de metadados). O módulo *Expansor de Restrições* (ER) é o responsável por esta tarefa. A abordagem de inferência de restrições descrita na Seção 4.3.1 é implementada por

este módulo. O módulo ER também interage com o repositório para obter acesso aos resumos estatísticos associados aos metadados da coleção.

Na Seção 4.3.2, descrevemos nossa abordagem para tratamento de restrições inconsistentes, através da definição de três algoritmos alternativos. Esses algoritmos fazem parte do módulo que chamamos de *Manipulador de Inconsistências* (MI). Esse módulo define os pesos das restrições para aplicar um dos algoritmos de eliminação de inconsistências definidos na Seção 4.3.2. Alternativamente, o módulo MI pode apresentar ao usuário as inconsistências detectadas (juntamente com os pesos encontrados). Este, por sua vez, pode ratificar ou alterar essas restrições antes de executar o algoritmo de agrupamento.

O módulo *Gerente de Agrupamento*, como o próprio nome deixa transparecer, é responsável pela coordenação e seqüenciamento da execução dos demais módulos do sistema. Em particular, este módulo é responsável pela ativação do algoritmo *MAC-KMeans* uma vez que todas as informações necessárias para sua execução foram coletadas ou produzidas pelos demais módulos.

Capítulo 5

Resultados Experimentais

A computation is a temptation that should be resisted as long as possible.

J. P. Boyd

5.1 Introdução

Neste capítulo, descrevemos os experimentos que realizamos com o objetivo de validar nossa proposta de agrupamento semi-supervisionado baseado em informação estrutural. A estrutura deste capítulo é a seguinte. Na Seção 5.2, são descritos alguns detalhes de implementação de nossa proposta. A Seção 5.3 descreve a OHSUMED, coleção na qual baseamos nossos experimentos. Descrevemos nossa metodologia de avaliação na Seção 5.4. Finalmente, as Seções 5.5, 5.6 e 5.7 descrevem os experimentos realizados.

5.2 Detalhes de Implementação

A implementação do protótipo de sistema de agrupamento semi-supervisionado foi feita em Java.[1] Utilizamos diversas bibliotecas de código aberto nessa imple-

[1]Descrevemos uma proposta de arquitetura para um sistema de agrupamento semi-supervisionado na Seção 4.4.

mentação. Dentre essas bibliotecas, as mais importantes são o *WEKA*[2] e o *Lucene*[3].

O *WEKA* é um conjunto de ferramentas para realização de experimentos em mineração de dados. Como o código fonte está disponível, implementamos o *MAC-KMeans* aproveitando a infra-estrutura do *WEKA*. As facilidades dessa ferramenta para realização de experimentos foram também bastante úteis nesta tese.

O *Lucene* é uma biblioteca para indexação e consulta de documentos. Essa biblioteca é bastante utilizada em aplicações de recuperação de informações (*information retrieval*. Em nossa implementação, o *Lucene* foi útil por permitir a indexação de informações estruturais (isto é, do esquema) dos documentos XML que utilizamos. Mais especificamente, a implementação de nossa abordagem de representação dual de documentos XML (vide Seção 4.2) foi facilitada com o uso do *Lucene*.

5.3 Coleções de Documentos Utilizadas

As coleções *20-Newsgroups*[4] e *Reuters-21578*[5] são duas coleções clássicas utilizadas em diversos trabalhos, tanto na tarefa de classificação, quanto na tarefa de agrupamento. No entanto, essas coleções não são apropriadas para nossos experimentos, por conta de não possuírem metadados associados. Por conta disso, decidimos utilizar a coleção MEDLINE para realizar nossos experimentos.

O MEDLINE é o principal banco de dados do Pubmed [54]. Seu conteúdo consiste de artigos em pesquisa básica e em ciências médicas. Atualmente o MEDLINE contém mais de 15 milhões de entradas. Seu conteúdo cataloga obras (artigos, livros, etc.) de mais de 4.800 periódicos científicos em biomedicina. O MEDLINE não armazena o conteúdo completo dos artigos e livros, mas suas citações. Cada citação compreende diversas informações sobre a obra, tais como título, resumo, autores e fonte. Atualmente, o MEDLINE possui resumos para cerca de 75% dos artigos registrados. Desde 1996, possui uma interface de consulta via WEB. Este imenso

[2]http://www.cs.waikato.ac.nz/ml/weka/
[3]http://lucene.apache.org/
[4]http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html
[5]http://www.research.att.com/~lewis

banco de dados é atualizado semanalmente e é também disponibilizado para fins de pesquisa.

Em nossos experimentos, utilizamos uma subcoleção do MEDLINE denominada OHSUMED, disponível em ftp://medir.ohsu.edu/pub/ohsumed. A coleção OHSUMED foi desenvolvida por Willian Hersh e seu grupo na Universidade de Ciências Medicas de Oregon [37]. Essa coleção consiste de 348.566 citações de 270 conferências médicas entre os anos de 1987 e 1991.

Na OHSUMED, os campos disponíveis são título, resumo, descritores MeSH, autores, fonte, tipo de publicação. O documento XML apresentado na Figura 4.1 é um dos documentos desta coleção. O que nos motivou a utilizar essa coleção foi o fato de cada entrada ter sido manualmente indexada com pelo menos uma de 23 categorias de doenças. As categorias da OHSUMED estão listadas na Tabela 5.2. O fato de existirem essas categorias é importante, pois permitiu que utilizássemos uma medida externa de avaliação em relação aos agrupamentos gerados nos experimentos (vide Seção 2.4). É importante notar, entretanto, que somente consideramos os documentos da OHSUMED associados a uma única categoria. (Em outras palavras, não consideramos o caso de agrupamento sobreposto, no qual um documento pode estar associado a mais de um grupo.) Além disso, do total de 348.566 entradas na OHSUMED, 233.445 possuem resumo (*abstract*). Em nossos experimentos, apenas consideramos as entradas com resumo.

5.4 Metodologia de Avaliação

Nos experimentos descritos neste capítulo, utilizamos diversos subconjuntos da coleção OHSUMED. Para gerar esses subconjuntos, selecionamos os primeiros 20.000 documentos do ano de 1991. Cada documento dessa coleção está associado a uma ou mais das 23 categorias existentes na OHSUMED. Desses 20.000 documentos, selecionamos os que satisfazem aos dois critérios seguintes: (1) possuem resumo associado e (2) estão associados a uma e somente uma categoria. Ou seja, ignoramos os documentos sem resumo e os que estão associados a mais de uma categoria. Essa

filtragem feita sobre os primeiros 20.000 documentos do ano de 1991 resultou em uma coleção de 3.357 documentos. A partir dessa coleção filtrada, extraímos as coleções utilizadas em nossos experimentos. A primeira coleção, que chamamos de *ohsu-03a* foi gerada através da escolha dos documentos associados às categorias C19, C06 e C18. A segunda coleção, que chamamos de *ohsu-03b*, foi gerada através da seleção dos documentos associados às categorias C12, C15, e C05. Finalmente, uma terceira coleção foi gerada, *ohsu-04*. Nessa coleção, selecionamos documentos de 4 categorias: C12, C06, C18 e C19.

5.4.1 Geração da Representação Dual para os Documentos

Após a extração das coleções descritas acima, nós as processamos (com o uso do *Lucene*) para representá-las segundo nossa forma dual, descrita na Seção 4.2. A representação correspondente a $\mathcal{A}_{desc}$ foi gerada considerando apenas o elemento *MeshHeadingList* de cada documento. Já a representação correspondente a $\mathcal{A}_{vsm}$ foi gerada considerando o conteúdo textual dos elementos *ArticleTitle* e *AbstractText* de cada entrada (vide Figura 4.1). O procedimento utilizado para a representação correspondente a $\mathcal{A}_{vsm}$ é descrito nos passos a seguir.

1. Removemos palavras funcionais (vide Seção 3.2.1) de uma lista pré-definida.
2. Aplicamos o algoritmo *Snowball* (presente no *Lucene*) para reduzir os termos a sua forma comum. Conforme descrito na Seção 3.2.2, esse procedimento tem o objetivo de excluir palavras muito freqüentes, ou palavras muito raras.
3. Após isso, eliminamos os termos que apareciam menos de três vezes na sua respectiva coleção (subconjunto da OHSUMED).
4. Para cada termo restante, calculamos sua medida TF/IDF.
5. Cada vetor correspondente a um documento foi então normalizado.

A Tabela 5.1 mostra o nome, quantidade de documentos e quantidade de dimensões de cada subconjunto, após o pré-processamento descrito acima ter sido realizado. É importante notar que, embora os subconjuntos descritos acima possuam

Tabela 5.1: Subconjuntos da OHSUMED utilizados nos experimentos

Nome do subconjunto	*#documentos*	*#dimensões*
ohsu-03a	382	1866
ohsu-03b	291	1546
ohsu-04	581	2028

uma quantidade relativamente pequena de documentos, eles ainda são relevantes e importantes de serem estudados. De fato, situações em que o tamanho da coleção é bem menor que a quantidade de dimensões da mesma são freqüentes na prática, quando os objetos a serem agrupados são documentos textuais.

5.4.2 Geração Artificial de Restrições

Outro aspecto importante de nossa metodologia de avaliação diz respeito ao procedimento que utilizamos para gerar as restrições que foram utilizadas nos experimentos. Em uma situação real, o próprio usuário do algoritmo de agrupamento semi-supervisionado fornece informação acerca da naturera (positiva ou negativa) de um par de objetos (documentos ou valores de atributos) que o sistema apresenta. Em nossos experimentos, utilizamos restrições geradas artificialmente em dois níveis de abstração: restrições *doc-doc* e restrições *meta-meta*. Descrevemos o processo de geração de cada um desses dois conjuntos a seguir. Antes disso, note que, nos dois casos de geração artificial de restrições descritos abaixo, os rótulos das categorias são utilizados para gerar as restrições artificialmente, mas não são utilizados durante a execução do algoritmo.

Restrições *doc-doc*

Para cada restrição, selecionamos de forma aleatória dois documentos d_i e d_j da coleção em questão. Após isso, consultamos o rótulo de cada um desses dois documentos. (Esses rótulos correspondem às 23 categorias utilizadas na OHSUMED.) Se os rótulos são iguais, então criamos uma restrição positiva entre os documentos. Do contrário, se os rótulos são diferentes, criamos uma restrição negativa entre os d_i e d_j.

Note que este procedimento assegura que as restrições geradas são completamente consistentes com a distribuição subjacente às categorias da coleção OHSUMED.

Restrições *meta-meta*

Um complicador neste caso de geração de restrições *meta-meta* diz respeito ao fato de que dois descritores podem estar associados a listas de documentos de diferentes classes. (Note que isso não acontece no caso das restrições *doc-doc*, pois estamos trabalhando com documentos associados a uma única classe.) Portanto, não podemos utilizar o mesmo critério de decisão descrito acima para definir a natureza (positiva ou negativa) da restrição *meta-meta*. O que fazemos é adotar primeiramente selecionar do repositório todos os descritores associados a no mínimo 2, e a no máximo 5 documentos. A justificativa para a escolha do limite inferior é o fato de que descritores associados a apenas um documento têm pouco poder de expansão durante a aplicação do algoritmo *InferirRestriçõesDocDoc* (vide Figura 4.4).

A justificativa para a escolha do limite superior pode ser dada através de uma analogia com a medida TF/IDF, descrita na Seção 3.2.3. Essa medida associa valores baixos a termos que ocorrem em muitos documentos de uma coleção, por conta de esses termos terem pouca força discriminativa. De forma análoga, um descritor v para o qual $Docs(v, k)$ (vide Definição 4.3) é muito alto tem pouca capacidade de representar um conjunto coeso de documentos, seja qual for a necessidade de agrupamento do usuário. Isso é particularmente importante na OHSUMED, onde os descritores formam uma rede hierárquica, com descritores mais abrangentes que outros. (Por exemplo, o descritor *Adult* ocorre em 102 dos 291 documentos da coleção *ohsu-03b*.) Além disso, descritores associados a muitos documentos têm um maior potencial para gerarem inconsistências no processo de inferência de restrições. Por essas razões, decidimos por utilizar um limite superior igual a 5 como ponto de corte para os descritores que são utilizados na definição das restrições *meta-meta*.

Note que mesmo após a seleção dos descritores de acordo com o limite superior descritos acima, ainda há a possibilidade de geração de restrições inconsistentes. Para isso, basta que usuário defina uma restrição sobre algum descritor que esteja

associado a documentos em mais de uma categoria. É claro que, uma vez que o sistema solicita ao usuário que defina o tipo de restrição sobre um par de descritores (v_i, v_j), este usuário pode ficar em dúvida e decidir por não fazer a definição. Neste caso, o sistema simplesmente apresenta outro par para que o usuário faça a definição do tipo. Por outro lado, pode ser que o usuário decida por definir a restrição (seja ela positiva ou negativa) sobre (v_i, v_j), o que tem como conseqüência a inferência de restrições *doc-doc* incoerentes com a distribuição das categorias da OHSUMED. Para simular essa situação possível de acontecer na prática, utilizamos o conceito de *entropia* [61]. Essa medida caracteriza o grau de pureza de uma coleção de objetos. Aqui, os objetos são as categorias associadas ao conjunto $Docs(v, k)$, cujos elementos são os documentos referenciados pelo descritor v. Sendo assim, a expressão para a entropia é apresentada na Equação 5.1. Nessa Equação, $p(c_i)$ corresponde à proporção de documentos em $Docs(v, k)$ que pertencem à categoria cujo rótulo é c_i.

$$E(Docs(v, k)) = -\sum_{i=1}^{k} p(c_i) \log_2 p(c_i) \tag{5.1}$$

Se um descritor possui somente documentos de mesma categoria (isto é, $k = 1$ na Equação 5.1), a entropia é mínima (igual a zero). Por outro lado, se um descritor possui documentos de mais de uma categoria, a entropia é maior que zero, e aumenta conforme a quantidade de categorias aumenta. Para dar um exemplo, considere o descritor denominado *In Vitro*, existente na OHSUMED. Esse descritor possui 5 documentos associados, sendo que 4 pertencem à categoria $C05$ e 1 pertence à $C12$. O cálculo da entropia associada a este descritor é feito da seguinte maneira:

$$E(Docs(In\ Vitro)) = -(4/5) \times \log_2(4/5) - (1/5) \times \log_2(1/5) = 0.721 \tag{5.2}$$

Decidimos por utilizar o conceito de entropia para simular o comportamento do usuário em uma situação real, diante de um par de descritores para o qual este usuário não tem certeza de que tipo de restrição definir. Em nossos experimentos, utilizamos o valor 75% como ponto de corte: se um determinado descritor está as-

sociado a uma entropia maior que 75% do valor máximo possível para a entropia, consideramos que o usuário opta por não definir uma restrição sobre o par apresentado. Note que o valor 0% corresponde a um usuário super-especialista, que tem completo conhecimento da coleção de documentos que está em análise. Essa situação é equivalente ao usuário ter completo conhecimento da estrutura de classes subjacente à coleção de documentos. Por outro lado, o valor igual a 100% para o ponto de corte corresponde a um usuário aleatório, que diante de uma restrição para a qual não tem certeza do tipo, escolhe arbitrariamente (aleatoriamente) este tipo. Consideramos que essas duas situações não refletem a realidade e, ao escolhermos o valor 75% para o ponto de corte, estamos considerando um usuário com bom conhecimento sobre a coleção de documentos.

Outro aspecto importante relativo às restrições *meta-meta* diz respeito à eliminação de inconsistências. Já que escolhemos um valor maior 0% para o ponto de corte mencionado acima, o processo de inferência (vide Figura 4.4) inerentemente gera restrições inconsistentes. Em todos os experimentos que apresentamos neste capítulo com o *MAC-KMeans*, utilizamos previamente o algoritmo *EI3* (vide Figura 4.14) para eliminação de inconsistências.

5.4.3 Medidas de Avaliação Utilizadas

Todos os experimentos foram realizados em 20 rodadas independentes. Dessas 20 rodadas, calculamos a média. Os valores apresentados nos gráficos são relativos às médias sobre as 20 rodadas. Além disso, assumimos que a quantidade de grupos é uma variável conhecida em todos os experimentos (assim como fazem outros trabalhos que utilizam o K-Means ou alguma variante deste).

Nos experimentos que realizamos para comparar a qualidade dos agrupamentos gerados, utilizamos duas medidas de avaliação externa (Seção 2.4.2). Em nosso caso, a categorização externa corresponde às 23 classes utilizadas para rotular os documentos da coleção OHSUMED. A primeira medida que utilizamos é a *informação mútua normalizada* (vide Equação 2.9). Conforme descrito na Seção 2.4.2, essa medida varia entre 0 e 1. Quanto mais alto o seu valor, mais próximo está o modelo

de agrupamento $\mathcal{C}$ gerado pelo algoritmo da categorização externa $\mathcal{C}'$. A segunda medida de avaliação externa que utilizamos é a *medida-F*, definida pela Equação 2.7, que é comumente utilizada em experimentos que envolvem dados textuais.

Há duas justificativas para a nossa escolha por medidas *externas*, em vez utilizarmos uma medida *interna* de avaliação. Em primeiro lugar, mencionamos na Seção 2.4.1 uma forma freqüentemente utilizada para medir a qualidade de um agrupamento: comparar o valor final da função objetivo alcançado pelo algoritmo de agrupamento, o que corresponde ao uso de uma medida interna de avaliação. No entanto, na tarefa de agrupamento semi-supervisionado, as restrições muitas vezes fazem com que o agrupamento formado seja bastante diferente do que seria formado no caso completamente não-supervisionado. Sendo assim, uma solução de agrupamento que satisfaça as restrições pode parecer *pior* que uma solução que não o faça, se somente o valor da função objetivo for utilizado. Portanto, consideramos que a necessidade de agrupamento do usuário é equivalente à formada pela categorias da OHSUMED e utilizamos as duas medidas mencionadas acima para comparar os resultados obtidos com a distribuição dessas categorias.

5.5 Qualidade do Agrupamento

Nos experimentos desta Seção, nosso objetivo é avaliar o uso do conjunto de restrições resultante do algoritmo *InferirRestriçõesDocDoc* (definido na Seção 4.3.1)) e sua influência sobre a qualidade do agrupamento gerado pelo algoritmo semi-supervisionado. Nossa estratégia para realizar essa avaliação foi a de comparar a qualidade do agrupamento em dois cenários, descritos a seguir.

No primeiro cenário, o usuário fornece uma quantidade n de restrições *meta-meta*. Essas restrições são então passadas como entrada para o algoritmo *InferirRestriçõesDocDoc*. Após isso, a saída desse algoritmo é passada para o *MAC-KMeans*. Para os testes com a inferência de restrições, utilizamos como atributo multivalorado o elemento *MeshHeadingList* da coleção OHSUMED (vide Figura 4.1).

No segundo cenário, o usuário fornece a mesma quantidade n de restrições, só

Tabela 5.2: Categorias existentes na coleção OHSUMED. As quantidades de documentos exibidas na terceira coluna são relativas aos documentos rotulados com somente uma categoria.

Nome	**Id**	**Qtd. docs**
Bacterial Infections and Mycoses	C01	423
Virus Diseases	C02	158
Parasitic Diseases	C03	65
Neoplasms	C04	1160
Musculoskeletal Diseases	C05	282
Digestive System Diseases	C06	587
Stomatognathic Diseases	C07	100
Respiratory Tract Diseases	C08	469
Otorhinolaryngologic Diseases	C09	124
Nervous System Diseases	C10	619
Eye Diseases	C11	161
Urologic and Male Genital Diseases	C12	490
Female Genital Diseases and Pregnancy Complications	C13	280
Cardiovascular Diseases	C14	1245
Hemic and Lymphatic Diseases	C15	214
Neonatal Diseases and Abnormalities	C16	200
Skin and Connective Tissue Diseases	C17	294
Nutritional and Metabolic Diseases	C18	385
Endocrine Diseases	C19	191
Immunologic Diseases	C20	524
Disorders of Environmental Origin	C21	546
Animal Diseases	C22	92
Pathological Conditions, Signs and Symptoms	C23	1796

Tabela 5.3: Distribuição de categorias nos subconjuntos gerados

ohsu-03a		**ohsu-03b**		**ohsu-04**	
C19	40	C12	155	C12	155
C06	176	C15	54	C06	176
C18	166	C05	82	C18	166
				C19	40

que ao nível de documentos. Para este cenário, utilizamos o algoritmo *PC-KMeans* [5], proposto por Sugato Basu e cuja implementação foi gentilmente fornecida pelo autor.

Primeiramente, executamos o *PC-KMeans* através do fornecimento de quantidades crescentes de restrições ao nível de documentos. Após isso, executamos o *MAC-KMeans*, dessa vez fornecendo a ele quantidades crescentes de restrições *doc-doc* inferidas pelo *InferirRestriçõesDocDoc* (relativas a *MeshHeadingList*). Em ambos os casos, variamos as quantidades de restrições (*meta-meta* para o *MAC-KMeans*, *doc-doc* para o *PC-KMeans*) na faixa de 0 até 100, em intervalos de 10.

As curvas de aprendizado que apresentam valores comparativos da medida NMI para esses dois algoritmos são apresentadas na Figura 5.1 (para *ohsu03a*), na Figura 5.2 (para *ohsu03b*), e na Figura 5.3 (para *ohsu04*).

As Figuras 5.4, 5.5 e 5.6 apresentam resultados equivalentes aos das Figuras 5.1, 5.2 e 5.3, com a diferença de que a *medida-F* é utilizada, em vez da medida NMI. Essas figuras apenas confirmam a tendência de melhora na qualidade do agrupamento quando comparamos o uso de quantidades iguais de restrições *doc-doc* e *meta-meta*.

Através da comparação dos valores da NMI e da medida-F alcançados com o uso de restrições *doc-doc* fornecidas ao *PC-KMeans* e os valores correspondentes alcançados com o mesmo número de restrições ao *meta-meta* (descritores MeSH) passadas por inferência ao *MAC-KMeans*, podemos perceber que, em geral, com apenas 10 restrições ao nível de metadados, o *MAC-KMeans* alcança níveis da NMI e de Medida-F que não são alcançados pelo *PC-KMeans* com o uso de restrições ao nível de documentos, mesmo quando 100 restrições são fornecidas.

Note que, para uma quantidade de restrições igual a 0, tanto o *MAC-KMeans*,

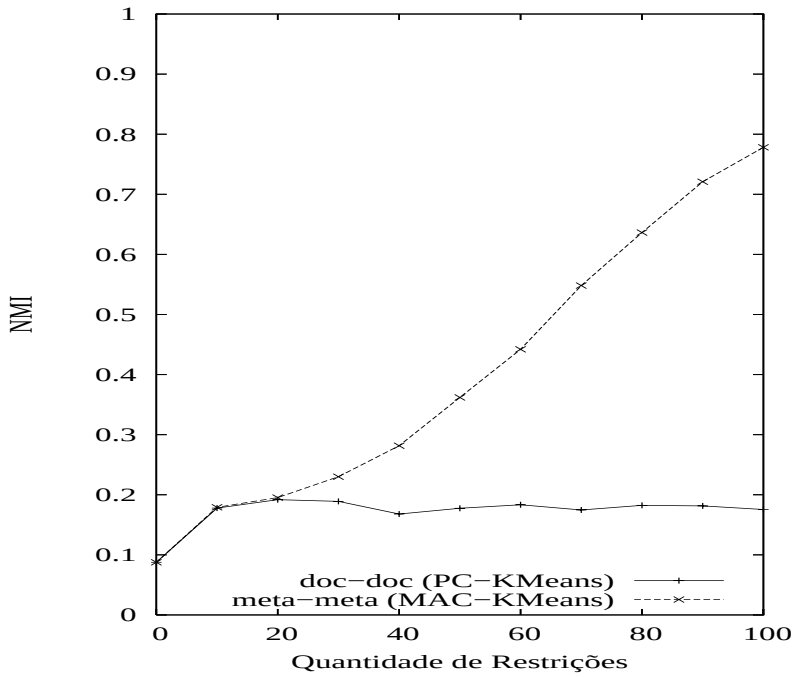


Figura 5.1: Comparação da medida NMI para a coleção *ohsu03a*, considerando diferentes valores de restrições ao nível de metadados geradas artificialmente

quanto o *PC-KMeans* são equivalentes ao K-Means esférico. Isso explica o fato de as curvas de aprendizado partirem sempre do mesmo valor de qualidade para uma quantidade de restrições igual a 0.

5.6 Quantidade de Restrições Inferidas

Para corroborar com os resultados obtidos para a medida NMI e a Medida-F, a Tabela 5.4 apresenta os valores médios das quantidades de restrições *doc-doc* inferidas a partir de uma certa quantidade de restrições *meta-meta*. A primeira coluna dessa tabela apresenta a quantidade de restrições *meta-meta* consideradas. A segunda, terceira e quarta colunas dessa tabela apresentam os resultados para a coleções *ohsu03a*, *ohsu03b* e *ohsu04*, respectivamente.

A análise dos dados apresentados na Tabela 5.4 nos permite concluir que, a grande quantidade de restrições *doc-doc* que é obtida através do algoritmo *Infe-*

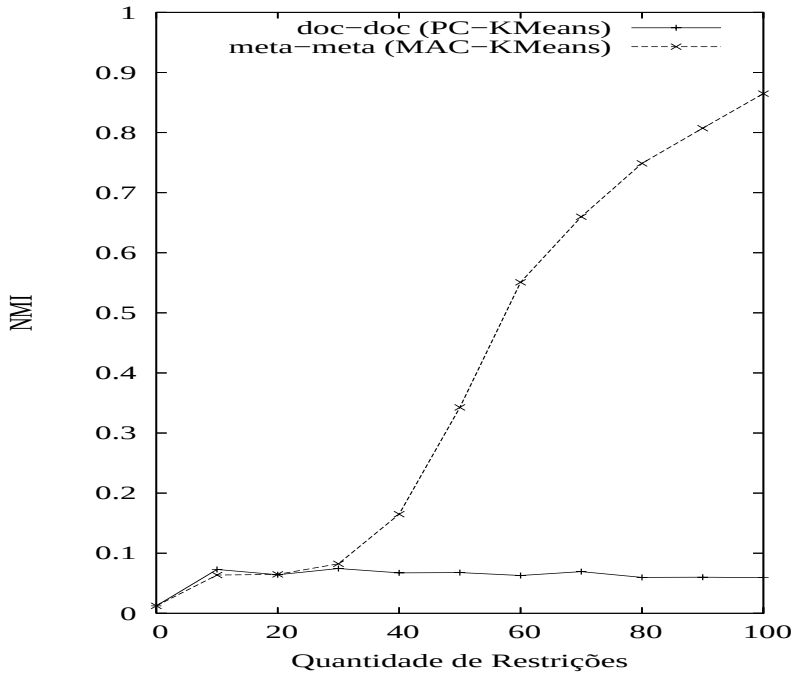


Figura 5.2: Comparação da medida NMI para a coleção *ohsu03b*, considerando diferentes valores de restrições ao nível de metadados geradas artificialmente

Tabela 5.4: Valores médios das quantidades de restrições *doc-doc* inferidas a partir de uma certa quantidade de restrições *meta-meta*. Dados relativos às coleções *ohsu03a*, *ohsu03b* e *ohsu04*.

Qtd. restrições	*ohsu03a*	*ohsu03b*	*ohsu04*
10	74,56	68,93	68,35
20	154,54	146,63	140,91
30	231,65	214,54	213,34
40	304,09	276,39	279,55
50	383,86	349,86	348,16
60	451,10	417,87	415,03
70	521,78	478,39	487,89
80	595,55	533,07	556,25
90	664,58	605,28	630,22
100	730,47	652,01	691,03

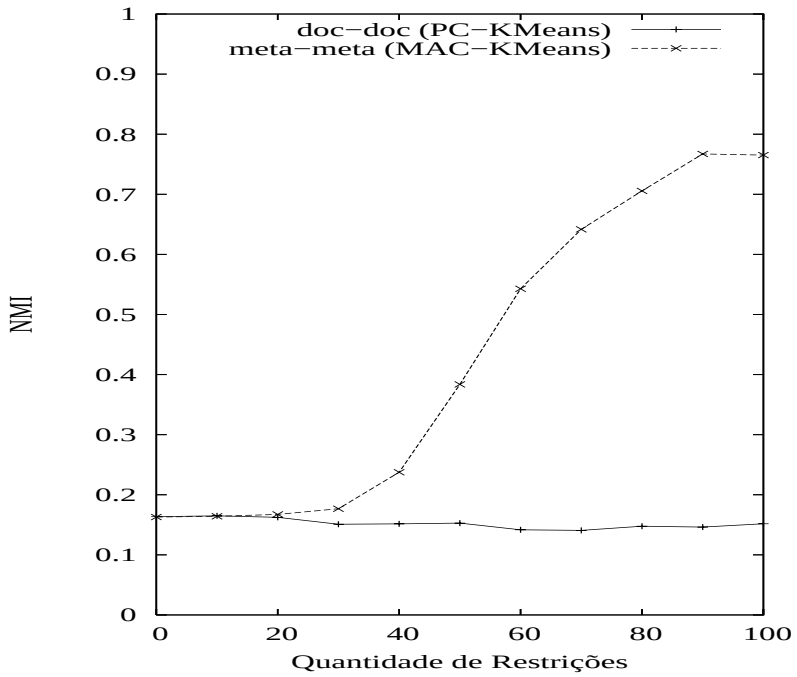


Figura 5.3: Comparação da medida NMI para a coleção *ohsu04*, considerando diferentes valores de restrições ao nível de metadados e de documentos

rirRestriçõesDocDoc é responsável pelo melhor qualidade do *MAC-KMeans*. De fato, essa grande quantidade de restrições inferidas é a razão da obtenção de valores melhores para a qualidade dos agrupamentos gerados. Por exemplo, a Tabela 5.4 permite perceber que, quando a quantidade de restrições ao nível de metadados é igual a 10, o sistema consegue inferir uma quantidade de restrições ao nível de documentos, em média, $7,45$ vezes maior, para a coleção *ohsu03a*. Valores semelhantes também são obtidos quando analisamos as duas outras coleções.

Os experimentos dessa Seção mostram que, para uma mesma quantidade de restrições fornecidas pelo usuário, a qualidade final do agrupamento é significativamente melhor se as mesmas forem definidas ao nível de metadados em vez de serem ao nível de documentos. Isso porque, com a aplicação do algoritmo *Inferir-RestriçõesDocDoc*, as restrições ao nível de metadados podem ser utilizadas para inferir restrições *doc-doc* por um fator bem maior que 1. Sendo assim, o algoritmo acaba por receber uma quantidade de restrições *doc-doc* maior do que acontece no

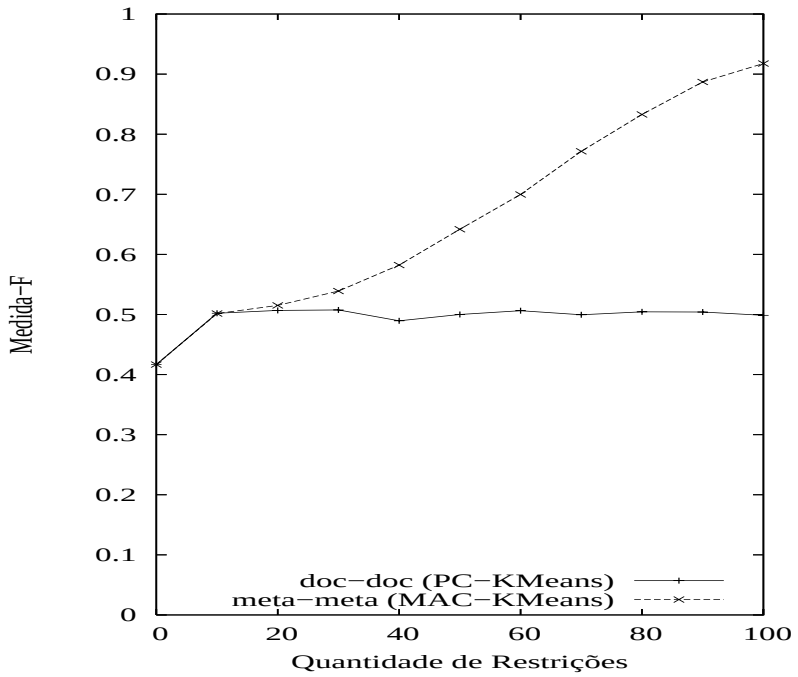


Figura 5.4: Comparação da medida-F para a coleção *ohsu03a*, considerando diferentes valores de restrições ao nível de metadados geradas artificialmente

caso em que o usuário as fornece diretamente ao sistema. O efeito prático disso é que o usuário pode definir *menos* restrições para alcançar os mesmos resultados alcançados no caso do fornecimento de restrições ao nível de metadados. Portanto, nossa estratégia de expansão de restrições tem o potencial de diminuir a sobrecarga sobre o usuário do sistema, sem diminuir a qualidade final do agrupamento.

5.7 Quantidade de Iterações até a Convergência

Conforme discutido na Seção 4.3.3, o *MAC-KMeans* apresenta uma complexidade assintótica igual a $O(n^2)$. O mesmo vale para o *PC-KMeans*, o que leva esses dois algoritmos a terem complexidade maior que a do K-Means esférico. Com relação a isso, um aspecto importante de nossa abordagem de agrupamento semi-supervisionado é relativo à quantidade de iterações necessárias para a convergência do algoritmo *MAC-KMeans*. As Figuras 5.7 (*ohsu-03a*), 5.8 (*ohsu-03b*) e 5.9 (*ohsu-*

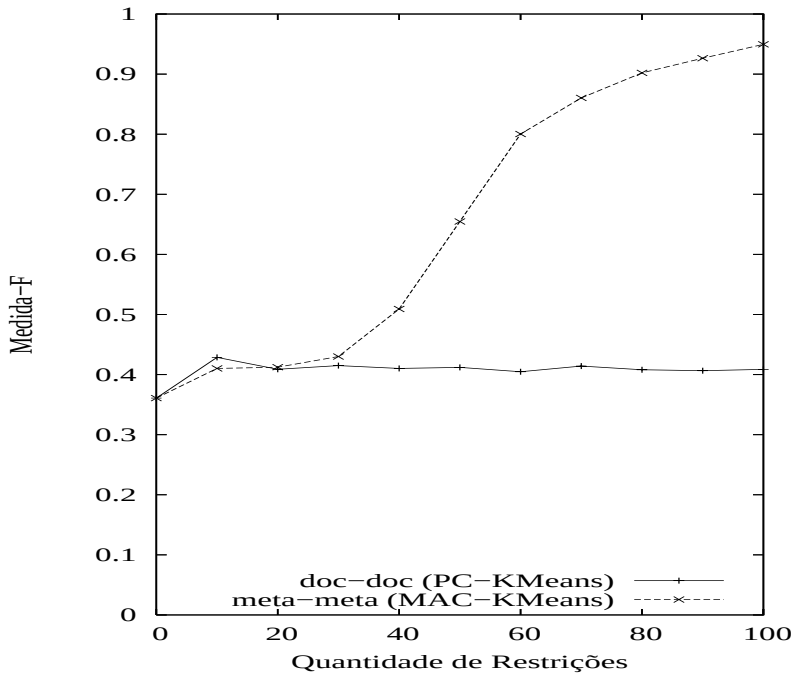


Figura 5.5: Comparação da medida-F para a coleção *ohsu03b*, considerando diferentes valores de restrições ao nível de metadados geradas artificialmente

04) mostram as quantidades de iterações (até a convergência) em função da quantidade de restrições fornecidas aos algoritmos *PC-KMeans* (*doc-doc*) e *MAC-KMeans* (*meta-meta*).

Através da análise das Figuras 5.7, 5.8 e 5.9, verificamos que o algoritmo *MAC-KMeans* necessita de uma quantidade de iterações para convergir consistentemente menor que o *PC-KMeans*. De fato, para ambos os algoritmo, a quantidade de iterações necessárias para convergência diminui, conforme mais restrições são fornecidas. A razão desse comportamento comum aos dois algoritmos é que as restrições fornecidas (ao *PC-KMeans* ou ao *MAC-KMeans*) permitem que estes gerem boas sementes para sua inicialização. Por sua vez, o fato de serem produzidas boas sementes leva aqueles algoritmos a *escaparem* mais facilmente de mínimos locais e a convergirem mais rapidamente.

No entanto, podemos notar, também pelas Figuras 5.7, 5.8 e 5.9, que a quantidade de iterações necessárias para convergência diminui *mais rápido* para o *MAC-*

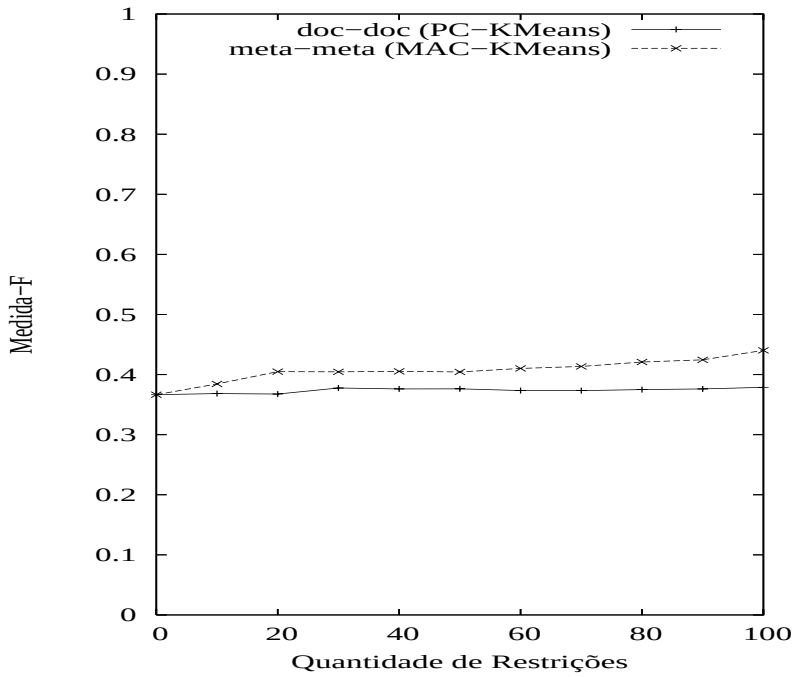


Figura 5.6: Comparação da medida-F para a coleção *ohsu04*, considerando diferentes valores de restrições ao nível de metadados e de documentos

KMeans do que para o *PC-KMeans*. Isso porque, fixada uma mesma quantidade de restrições fornecidas pelo usuário, o *MAC-KMeans* recebe como entrada uma quantidade de restrições bem maior que o *PC-KMeans* (por conta do processo de inferência de restrições), e isso faz com que a quantidade de iterações necessárias à convergência pelo primeiro seja menor do que a quantidade necessária pelo segundo.

É interessante ainda discutir sobre a influência da quantidade de restrições fornecidas no tempo total de execução de um agrupamento semi-supervisionado baseado em restrições. Já sabemos que quantidades maiores de restrições fazem com que a quantidade de iterações até a convergência diminua. Dois fatores são condicionantes desse tempo de execução. Mais que isso, a quantidade de iterações até a convergência diminui mais rápido para o *MAC-KMeans* do que para o *PC-KMeans*, pois o primeiro recebe uma quantidade de restrições bem maior que o primeiro (uma vez fixada uma quantidade de restrições de entrada para esses algoritmos). No entanto, não fica claro pela análise das Figuras 5.7, 5.8 e 5.9 qual é a relação entre

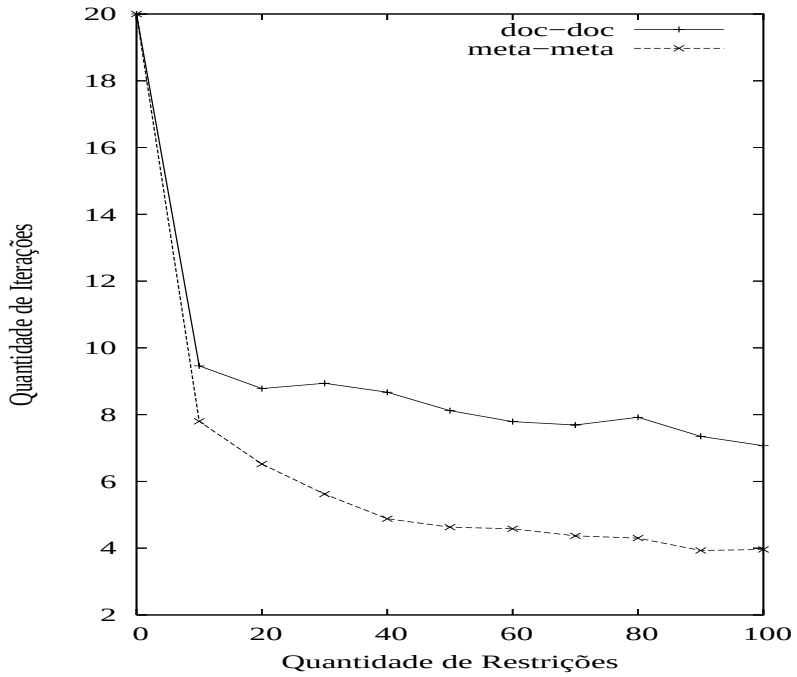


Figura 5.7: Quantidades de iterações até a convergência, em função da quantidade de restrições fornecidas ao algoritmo. Dados relativos à coleção *ohsu-03a*

o *tempo total de execução* do *MAC-KMeans* e do *PC-KMeans*, para uma mesma quantidade de restrições de entrada.

Para esclarecer a questão levantada acima, considere os dois fatores que influenciam no tempo total dos algoritmos *MAC-KMeans* e *PC-KMeans*. O primeiro fator corresponde à quantidade de restrições que ele deve considerar. O tempo de execução do algoritmo é *diretamente* proporcional à quantidade de restrições a serem consideradas. O segundo fator é a quantidade de iterações necessárias até a convergência. O tempo de execução do algoritmo é *inversamente* proporcional à quantidade de iterações necessárias para a convergência. Para comparar a influência desses dois fatores, plotamos os tempos totais de execução daqueles dois algoritmos. Essas plotagens são apresentadas nas Figuras 5.10, 5.11 e 5.12.

Uma primeira conclusão que podemos tirar dos dados das Figuras 5.10, 5.11 e 5.12 é que, na prática, a complexidade maior dos algoritmos de agrupamento semi-supervisionado baseados em restrições não resulta em um tempo maior de execução

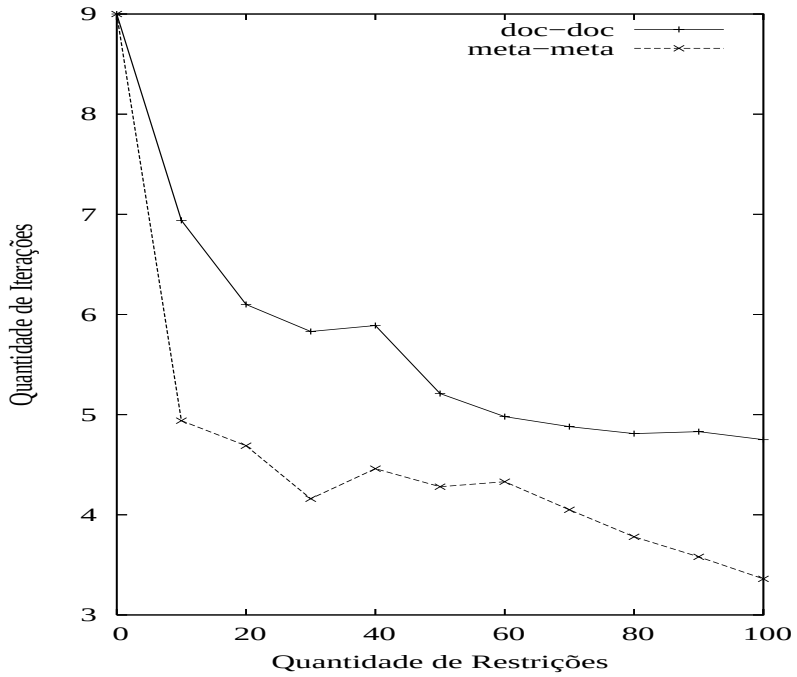


Figura 5.8: Quantidades de iterações até a convergência, em função da quantidade de restrições fornecidas ao algoritmo. Dados relativos à coleção *ohsu-03b*

em relação ao algoritmo não-supervisionado. De fato, note que nas Figuras 5.10, 5.11 e 5.12, o tempo de execução do K-Means esférico (algoritmo não-supervisionado) corresponde aos pontos das curva com quantidade de restrições igual a 0.

Outra conclusão que podemos tirar dos dados das Figuras 5.10, 5.11 e 5.12 é que os tempos de execução dos algoritmos *MAC-KMeans* e *PC-KMeans* são bastante similares. Isso se justifica pelo fato de que, embora o *MAC-KMeans* tenha uma quantidade maior de restrições para usar comparativamente ao *PC-KMeans* (o que aumento o tempo de execução do primeiro), essa maior quantidade de restrições leva a uma quantidade menor de restrições necessárias até a convergência. Isso faz com que os tempos de execução desses algoritmos permaneçam similares, conforme aumentamos a quantidade de restrições de entrada.

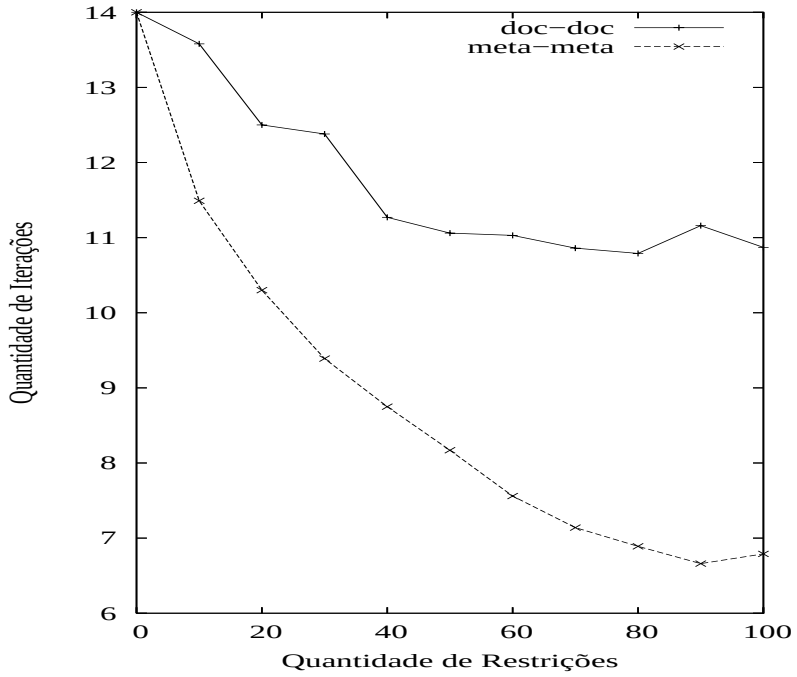


Figura 5.9: Quantidades de iterações até a convergência, em função da quantidade de restrições fornecidas ao algoritmo. Dados relativos à coleção *ohsu-04*

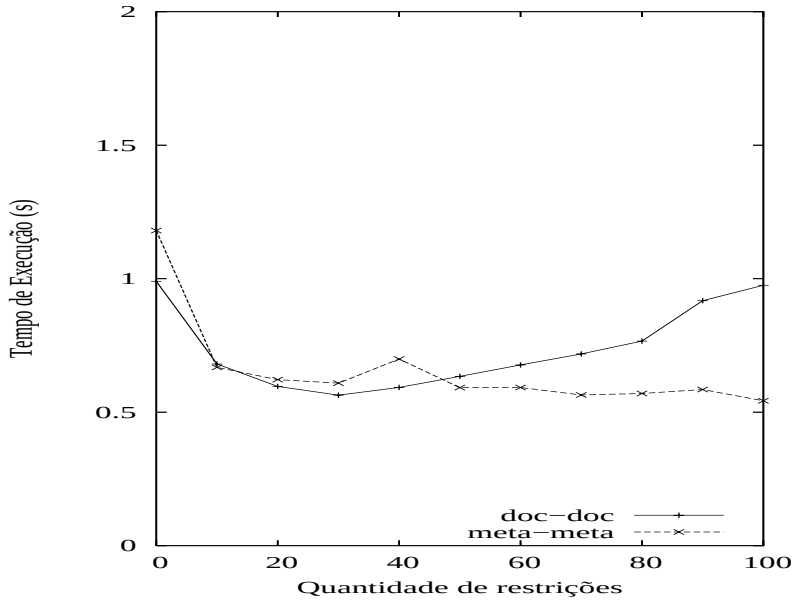


Figura 5.10: Tempos totais de execução dos algoritmos *MAC-KMeans* e *PC-KMeans*. Dados relativos à coleção *ohsu-03a*

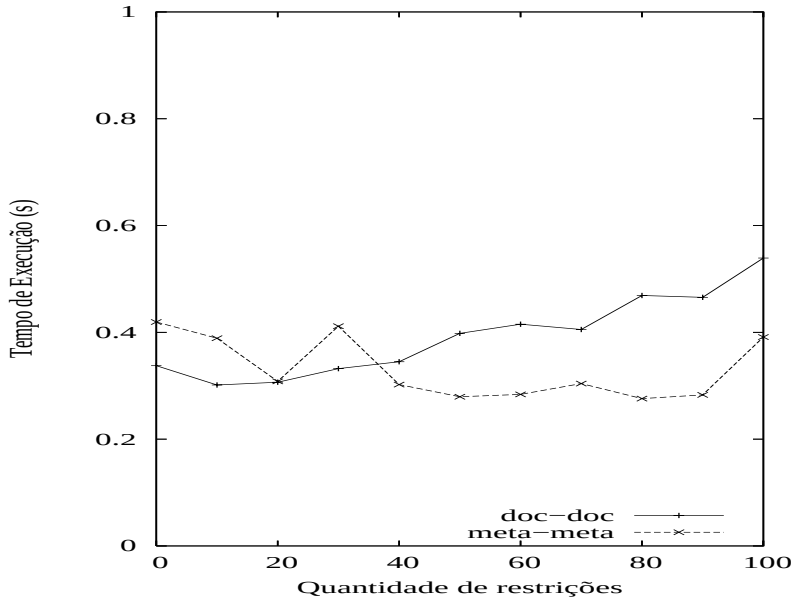


Figura 5.11: Tempos totais de execução dos algoritmos *MAC-KMeans* e *PC-KMeans*. Dados relativos à coleção *ohsu-03b*

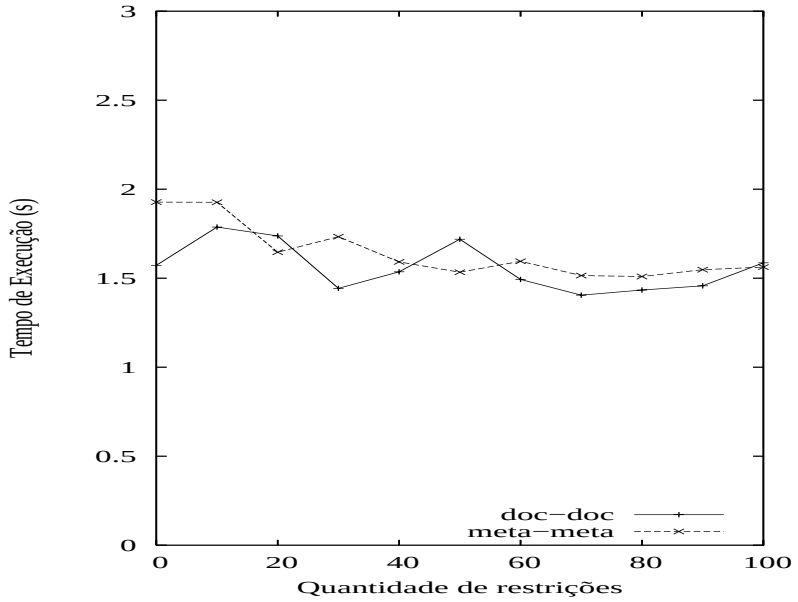


Figura 5.12: Tempos totais de execução dos algoritmos *MAC-KMeans* e *PC-KMeans*. Dados relativos à coleção *ohsu-04*

Capítulo 6

Conclusões

Isn't here...

Pink Floyd, The Wall (1979)

6.1 Análise das Contribuições

Apesar de sua longa história, a tarefa de agrupamento ainda possui muitos problemas em aberto. Nessa tarefa, diversos trabalhos da literatura têm mostrado que a incorporação de conhecimento prévio é bastante benéfica. Esta tese apresentou uma abordagem para aproveitar informação estrutural associada a documentos XML centrados em texto no agrupamento semi-supervisionado. Primeiramente, definimos nossa visão dual de um documento XML, na qual este pode ser interpretado e manipulado tanto através de seu conteúdo, quanto de sua estrutura lógica. Esta representação ocasionou o surgimento de novas possibilidades para o usuário definir restrições ao nível de metadados.

Após a definição de uma representação para documentos XML, utilizamos esta perspectiva dual para derivar um algoritmo de inferência de restrições *doc-doc* a partir das restrições *meta-meta* fornecidas pelo usuário. É importante notar que nossa abordagem para inferências de restrições a partir das fornecidas pelo usuário pode ser facilmente acoplada a sistemas que implementam algoritmos de agrupamento

semi-supervisionado que trabalham somente sobre restrições ao nível de documentos. Isto porque o produto final do processo de expansão é um conjunto de restrições *doc-doc*. Por essa razão, consideramos nossa abordagem uma extensão das abordagens propostas em trabalhos anteriores, que manipulam apenas restrições *doc-doc* (vide Seção 3.4). Conforme mencionamos no capítulo 4, a qualidade dos resultados gerados por um algoritmo de agrupamento partitivo depende muito da definição inicial dos centróides, ou seja, das sementes para inicialização do algoritmo. Além disso, de uma forma geral, quanto mais restrições são fornecidas a um algoritmo de agrupamento semi-supervisionado, melhor é a qualidade do resultado. Isso se explica pelo fato de o algoritmo utilizar as restrições para definir boas sementes para sua inicialização. Nossos experimentos realizados sobre subcoleções da OHSUMED mostraram que conseguimos alcançar taxas de inferência bastante altas. Nesses experimentos, as quantidades de restrições *doc-doc* resultantes da aplicação do algoritmo de inferência alcançaram valores na média 7 vezes maiores que a quantidade de restrições de entrada. Esse resultado é significativo, pois permite ao sistema gerar modelos de agrupamento de qualidade satisfatória a partir de uma quantidade relativamente pequena de restrições. Ou seja, a definição de restrições sobre os metadados permite que o usuário forneça menos restrições para o sistema (comparativamente ao caso em que o usuário define as restrições sobre os documentos diretamente), sem perda de qualidade no agrupamento.

6.2 Limitações e Trabalhos Futuros

Na Seção 4.3.2, definimos algoritmos para eliminar inconsistências que surgem durante o processo de inferência de restrições *doc-doc* a partir de restrições *meta-meta*. No entanto, os três algoritmos propostos (*EI1*, *EI2* e *EI3*) realizam a eliminação de inconsistências seguindo uma estratégia local. Queremos dizer com isso que esses algoritmos eliminam inconsistências para cada vizinhança positiva de forma separada uma da outra. Ao eliminar a inconsistência em uma determinada vizinhança, esses algoritmos podem estar eliminando inconsistências em outras vizinhanças. Como

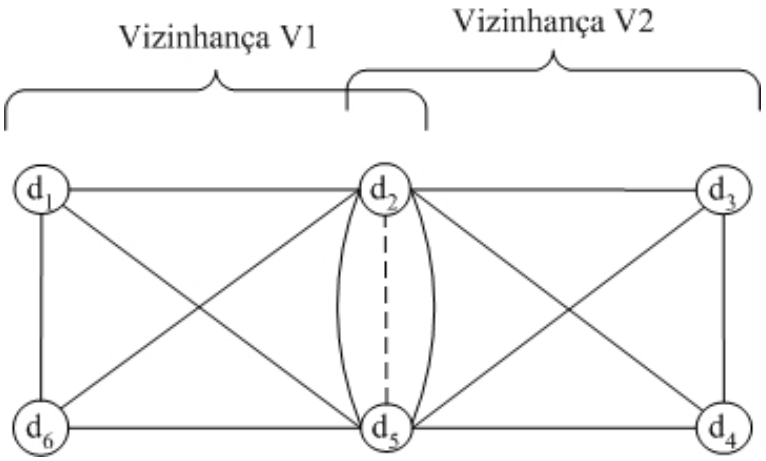


Figura 6.1: Grafo que ilustra o problema com a eliminação local (por vizinhança positiva) de inconsistências.

conseqüência, dependendo da ordem na qual as vizinhanças são processadas, o resultado final pode ser aquém do ótimo. A Figura 6.1 apresenta um exemplo dessa situação. Nessa figura, são desenhadas duas vizinhanças, V_1 e V_2. Note que, se o algoritmo eliminar a restrição negativa entre os documentos d_2 e d_5 quando estiver processando V_1, a inconsistência existente em V_2 é também eliminada. Esta tese deixa então a seguinte questão em aberto para ser tratada em um próximo trabalho: é possível definir um algoritmo que realize a eliminação seguindo uma estratégia menos localizada e que produza um resultado melhor?

Nossa abordagem descrita nesta tese para aproveitar a existência de metadados XML é restrita a somente uma coleção de documentos. Ou seja, não consideramos o caso em que a coleção a ser agrupada é composta de duas ou mais subcoleções de documentos XML, onde cada delas contém um esquema XML diferente da outra. No entanto, acreditamos que nosso trabalho pode ser estendido para trabalhar com coleções de documentos XML que possuam correspondências entre seus esquemas. Logicamente, deve ser definida uma forma de mapear elementos de um esquema para elementos de outro esquema. Isso com o objetivo de permitir que se possa calcular a similaridade semântica entre dois documentos de coleções diferentes. De fato, há diversos trabalhos que abordam o problema de mapeamento entre esquema XML e

entre ontologias. O sistema *UMLS Metathesaurus*[1], por exemplo, é um sistema cujo objetivo é unificar diversas ontologias na área médica.

Nosso foco nesta tese foi o caso das restrições *meta-meta* (que envolvem um único atributo) e a inferência de restrições *doc-doc* a partir das primeiras. Uma possibilidade não explorada nesta tese é o caso em que o usuário deseja definir restrições tanto no nível de documentos, quanto no de metadados. Além disso, outra situação que não exploramos são restrições sobre dois ou mais atributos. Fica para um próximo trabalho a análise dessas possibilidades e suas conseqüências sobre o processo de inferência.

O *MeSH* (*Medical Subjects Headings*) é uma taxonomia de conceitos estruturada de forma hierárquica e utilizada para anotar manualmente os documentos do *PUBMED*, o maior banco de dados de artigos em formato digital atualmente existente na área biomédica[2]. Uma das coleções existentes no *PUBMED* é a *MEDLINE*, da qual a OHSUMED é uma subcoleção. Nesta tese, utilizamos os descritores *MeSH* associados aos documentos da OHSUMED. Contudo, não consideramos a relação hierárquica existente entre os descritores. Além disso, os principais descritores de cada documento (isto é, os mais relevantes) são anotados como tal. Intuitivamente, tanto a estrutura hierárquica, quanto a informação sobre a relevância dos descritores podem ser exploradas para ajudar no processo de eliminação de inconsistências e de expansão de restrições. Por conta disso, também pretendemos investigar como a existência dessas características nos metadados pode ser aproveitada e incorporada à nossa presente proposta.

Um dos principais problemas em aberto em métodos de agrupamento partitivos é o fato de a grande maioria deles, o K-Means e suas variantes incluídas, considerarem que a quantidade de grupos a ser formada como informação de entrada. Esse problema é de importância prática, principalmente em um sistema interativo que agrupe os documentos resultados de um processo de busca antes de apresentá-los ao usuário. Nesse contexto, pretendemos também investigar como a existência de

[1]http://www.nlm.nih.gov/pubs/factsheets/umlsmeta.html
[2]Vide http://www.pubmed.com e http://www.mesh.com

restrições definidas sobre os objetos a serem agrupados poder ser usada para derivar limites inferiores e superiores para o valor de k (quantidade de grupos a ser formada).

Nesta tese, consideramos apenas o caso de agrupamento estrito (vide Seção 2.3.2). Ou seja, não é considerada a possibilidade de um documento pertencer a mais de um grupo. Na verdade, a maioria dos algoritmos de agrupamento semi-supervisionado existentes se baseiam na premissa de que cada objeto pertence a um e somente um grupo. No entanto, diversas coleções de dados reais possuem a característica de que seus elementos pertencem a mais de um grupo simultaneamente. Essa suposição dos algoritmos de agrupamento semi-supervisionado, o nosso incluído, limita sua utilização no processamento dessas coleções. Em um trabalho futuro, pretendemos avaliar de que maneira informação estrutural associada à coleção de documentos pode ser utilizada em algoritmos de agrupamento difusos. Já há trabalhos investigando esse problema. Dois deles são [17] e [34]. Como é bem sabido, no agrupamento sobreposto, o modelo de agrupamento resultante pode conter objetos que pertençam a mais de um grupo simultaneamente. O que esses trabalhos propõem é utilizar a técnica de *aprendizado ativo* para definir quais são os graus de pertinência dos objetos situados na fronteira entre dois ou mais grupos. Nossa intenção é aproveitar a existência de informação ao nível de metadados para derivar esses graus de pertinência. Nesse contexto, se constitui um problema em aberto a questão de como redefinir os conceitos de restrições positivas e negativas para contemplar o caso em que objetos pertençam a mais de um grupo.

Apesar da grande quantidade de algoritmos de agrupamento existente, o problema de atualização incremental dos grupos quando novos objetos estão disponíveis ainda não foi suficientemente explorado na literatura. Até onde vai nosso conhecimento, os únicos trabalhos que atacam esse problema são IncrementalDBSCAN [29] e IncOPTICS [44]. Quando os grupos foram formados considerando conhecimento externo (como é o caso da tarefa de agrupamento semi-supervisionado), definir como esses grupos podem ser atualizados não é uma tarefa trivial. É nosso interesse avaliar esse aspecto em um próximo trabalho.

Por fim, acreditamos que ainda há grandes oportunidades de pesquisa na exploração de metadados para a tarefa de agrupamento. Essa tese explorou uma pequena parte de todos os recursos existentes na estrutura de uma coleção de documentos XML. Além disso, com o desenvolvimento da WEB Semântica, o uso desse padrão de representação só tende a aumentar. Nesse contexto uma tecnologia particular que se mostra promissora é o *RDF* (*Resource Description Framework*) [21]. Seu crescente uso em bibliotecas digitais (vide, por exemplo, [69]) serve de motivação para explorarmos essa tecnologia na tarefa de agrupamento semi-supervisionado de documentos XML.

Bibliografia

[1] Sugato Basu. "Semi-supervised Clustering with Limited Background Knowledge". In: *Proceedings of the Ninth AAAI/SIGART Doctoral Consortium*, pp. 979–980, San Jose, CA, jul 2004.

[2] Mikhail Bilenko e Sugato Basu. "A Comparison of Inference Techniques for Semi-supervised Clustering with Hidden Markov Random Fields". In: *Proceedings of the ICML-2004 Workshop on Statistical Relational Learning and its Connections to Other Fields (SRL-2004)*, Banff, Canada, 2004.

[3] Sugato Basu, Arindam Banerjee e Raymond Mooney. "Semi-supervised Clustering by Seeding". In: *Proceedings of the 19th Intl. Conf. on Machine Learning (ICML-2002)*, pp. 19–26, Sydney, Australia, jul 2002.

[4] Sugato Basu, Mikhail Bilenko e Raymond J. Mooney. "Comparing and Unifying Search-Based and Similarity-Based Approaches to Semi-Supervised Clustering". In: *Proceedings of the ICML-2003*, pp. 42–49, Washington DC, Aug 2003.

[5] Sugato Basu, Arindam Banerjee e Raymond J. Mooney. "Active Semi-Supervision for Pairwise Constrained Clustering". In: *Proceedings of the SIAM International Conference on Data Mining (SDM-2004)*, pp. 333–344, Lake Buena Vista, FL, Apr 2004.

[6] Sugato Basu, Mikhail Bilenko e Raymond J. Mooney. "A Probabilistic Framework for Semi-Supervised Clustering". In: *Proceedings of the KDD-2004*, pp. 59–68, Seattle, WA, aug 2004.

[7] Mikhail Bilenko, Sugato Basu e Raymond J. Mooney. "Integrating Constraints and Metric Learning in Semi-Supervised Clustering". In: *Proceedings of the 21st International Conference on Machine Learning (ICML-2004)*, pp. 81–88, Banff, Canada, jul 2004.

[8] Abraham Bagherjeiran, Christoph F. Eick e Ricardo Vilalta:. *Adaptive Clustering: Better Representatives with Reinforcement Learning*. Technical Report UH-CS-05-06, University of Houston, Department of Computer Science, mar 2005.

[9] Aharon Bar-Hillel, Tomer Hertz, Noam Shental e Daphna Weinshall. "Learning distance functions using equivalence relations". In: *Proceedings of the Proc. of 20th International Conference on Machine Learning (ICML)*, pp. 11–18, Washington DC, aug 2003.

[10] J. Bilmes. *A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models*. Technical Report ICSI-TR-97-021, University of Berkeley, 1997.

[11] C. L. Blake e C. J. Merz. "UCI: repository of machine learning databases", 1998.

[12] Eduardo Bezerra, Marta Mattoso e Geraldo Xexéo. "On the Usage of Structural Information in Constrained Semi-Supervised Clustering of XML Documents". Florent Masseglia, Pascal Poncelet e Maguelonne Teisseire editores, In: *Data Mining Patterns: New Methods and Applications*. Idea Group Inc., to be published in 2006.

[13] T. BOZKAYA e M. OZSOYOGLU. "Distance-based indexing for high-dimensional metric spaces". In: *Proceedings of the Proc. ACM SIGMOD International Conference on Management of Data*, pp. 357–368, 1997.

[14] Tim Bray, Jean Paoli e C. M. Sperberg-McQueen. *Extensible Markup Language (XML) 1.0 (Second Edition)*. W3C Recommendation REC-xml-20001006,

World Wide Web Consortium (W3C), oct 2000.

http://www.w3.org/TR/2000/REC-xml-20001006/

[15] Ricardo A. Baeza-Yates e Berthier A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[16] Masahiro Okamoto Chinatsu Arima, Taizo Hanai. "Gene Expression Analysis Using Fuzzy K-Means Clustering". *Genome Informatics*, 14:334–335, 2003.

[17] Nicolas Cebron e Michael R. Berthold. "Mining of Cell Assay Images Using Active Semi-Supervised Clustering". In: *Proceedings of the IEEE International Conference on Data Mining, Workshop Computational Intelligence in Data Mining (ICDM 05)*, pp. 63–69, 2005.

[18] David Cohn, Rich Caruana e Andrew McCallum. *Semi-supervised clustering with user feedback*. Technical Report TR2003-1892, Cornell University, 2003.

[19] Darya Chudova, Christopher Hart, Eric Mjolsness e Padhraic Smyth. "Gene Expression Clustering with Functional Mixture Models". Sebastian Thrun, Lawrence Saul e Bernhard Schölkopf editores, In: *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[20] Wesley W. Chu. "Medical Digital Library to Support Scenario Specific Information Retrieval". In: *Proceedings of the Kyoto International Conference on Digital Libraries*, pp. 388, 2000.

citeseer.csail.mit.edu/chu00medical.html

[21] K. Selçuk Candan, Huan Liu e Reshma Suvarna. "Resource description framework: metadata and its applications". *SIGKDD Explor. Newsl.*, 3(1):6–19, 2001.

[22] E. Chavez, Gonzalo Navarro, Ricardo Baeza-Yates e J. L. Marronquin. "Searching in metric spaces". *ACM Computing Surveys*, 33(3):273–321, 2001.

[23] Ayhan Demiriz, Kristin P. Bennett e Mark J. Embrechts. "Semi-Supervised Clustering Using Genetic Algorithms". In: *Proceedings of the Intelligent Engineering Systems through Artificial Neural Networks*, pp. 809–814. ASME Press, 1999.

[24] R. O. Duda, P. E. Hart e D. G. Stork. *Pattern Classification*. Wiley, 2001.

[25] Inderjit S. Dhillon e Dharmendra S. Modha. "Concept Decompositions for Large Sparse Text Data Using Clustering". *Machine Learning*, 42(1/2):143–175, 2001.

[26] Byron E. Dom. *An Information-Theoretic External Cluster-Validity Measure*. Technical Report RJ 10219, IBM Research, may 2001.

[27] Ian Davidson e S. Ravi. "Clustering under Constraints: Feasibility Issues and the K-Means Algorithm". In: *Proceedings of the 5th SIAM Data Mining Conference*, 2005.

[28] Ian Davidson e S. Ravi. "Hierarchical Clustering with Constraints: Theory and Practice". In: *Proceedings of the Ninth European Conference on Principles and Practice of Knowledge Discovery in Databases (PAKDD-2005)*, Porto, Portugal, 2005.

[29] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Michael Wimmer e Xiaowei Xu. "Incremental Clustering for Mining in a Data Warehousing Environment". In: *Proceedings of the VLDB '98: 24rd International Conference on Very Large Data Bases*, pp. 323–333, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[30] Michael B. Eisen, Paul T. Spellman, Patrick O. Brown e David Botstein. "Cluster analysis and display of genome-wide expression patterns". *National Academy Science*, 95:14863–14868, 1998.

[31] Willian Frakes e Ricardo Baeza-Yates editores,. *Stemming algorithms*, pp. 131–160. Prentice Hall, New Jersey, 1992.

[32] U. Fayyad, C. Reina e P. Bradley. "Initialization of Iterative Refinement Clustering Algorithms". In: *Proceedings of the Knowledge Discovery and Data Mining*, pp. 194–198, 1998.

[33] N. Grira, M. Crucianu e N. Boujemaa. "Unsupervised and semi-supervised clustering: a brief survey". In: *Proceedings of the A Review of Machine Learning Techniques for Processing Multimedia Content, report of the MUSCLE European Network of Excellence (FP6)*, July 2004.

[34] Nizar Grira, Michel Crucianu e Nozha Boujemaa. "Active Semi-Supervised Clustering for Image Database Categorization". In: *Proceedings of the 7th ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR'05)*, Singapore, November 2005.

[35] Sudipto Guha, Rajeev Rastogi e Kyuseok Shim. "CURE: An Efficient Clustering Algorithm for Large Databases". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD-98)*, pp. 73–84, Seattle, WA, USA, 1998.

[36] Sudipto Guha, Rajeev Rastogi e Kyuseok Shim. "ROCK: A Robust Clustering Algorithm for Categorical Attributes". *Information System Journal*, 25(5):273–321, 2000.

[37] William Hersh, Christopher Buckley, T.J. Leone e David Hickman. "OHSUMED: an interactive retrieval evaluation and new large text collection for research". W. Bruce Croft e Cornelis J. Van Rijsbergen editores, In: *SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pp. 192–201, Dublin, IE, 1994. Springer Verlag, Heidelberg, DE.

[38] Maria Halkidi, Yannis Batistakis e Michalis Vazirgiannis. "On Clustering Validation Techniques". *J. Intell. Inf. Syst.*, 17(2-3):107–145, 2001.

[39] Zhexue Huang. “A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining.”. In: *Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, pp. 0–, 1997.

[40] Piotr Indyk. “A Sublinear Time Approximation Scheme for Clustering in Metric Spaces”. In: *Proceedings of the FOCS ’99: 40th Annual Symposium on Foundations of Computer Science*, pp. 154–159, Washington, DC, USA, 1999. IEEE Computer Society.

[41] Anil K. Jain e Richard C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, New Jersey, 1988.

[42] Han joon Kim e Sang goo Lee. “User-Feedback Driven Document Clustering Technique for Information Organization”. *IEICE Transactions on Information and Systems*, E85-D(6):1043–1048, 2002.

[43] A. K. Jain, M. N. Murty e P. J. Flynn. “Data clustering: a review”. *ACM Computing Surveys*, 31(3):264–323, 1999.

[44] Hans-Peter Kriegel, Peer Kröger e Irina Gotlibovich. “Incremental OPTICS: Efficient Computation of Updates in a Hierarchical Cluster Ordering.”. In: *Proceedings of the Proc. 5th International Conference on Data Warehousing and Knowledge Discovery (DaWaK’03)*, pp. 224–233, Prague, Czech Republic, 2003.

[45] Dan Klein, Sepandar D. Kamvar e Christopher D. Manning. “From Instance-level Constraints to Space-Level Constraints: Making the Most of Prior Knowledge in Data Clustering”. In: *Proceedings of the ICML-2002*, pp. 307–314, 2002.

[46] Michael Kearns, Yishay Mansour e Andrew Y. Ng. “An information-theoretic analysis of hard and soft assignment methods for clustering”. In: *Proceedings of the NATO Advanced Study Institute on Learning in graphical models*, pp. 495–520, Norwell, MA, USA, 1998. Kluwer Academic Publishers.

[47] Jon Kleinberg e Eva Tardos. "Approximation Algorithms for Classification Problems with Pairwise Relationships: Metric Labeling and Markov Random Fields". In: *Proceedings of the FOCS '99: 40th Annual Symposium on Foundations of Computer Science*, pp. 14, Washington, DC, USA, 1999. IEEE Computer Society.

[48] Joon Ho Lee. "Properties of extended Boolean models in information retrieval". In: *Proceedings of the SIGIR'94: 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 182–190, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[49] J. B. Lovins. "Development of a stemming algorithm". *Mechanical Translation and Computational Linguistics*, 11:22–31, 1968.

[50] J. B. MacQueen. "Some Methods for classification and analysis of multivariate observations". *L. M. LeCam and N. Neyman [eds.] fifth Berkeley symposium on mathematical statistics and probability*, 1:281–297, 1967.

[51] C. D. Manning e H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, MA, 1999.

[52] George Karypis Michael Steinbach e Vipin Kumar. "A comparison of document clustering techniques". In: *Proceedings of the KDD Workshop on Text Mining*, 2000.

[53] Martin F. Porter. "An algorithm for suffix stripping". *Program*, 14(3):130–137, 1980.

[54] PUBMED, 2005.

[55] S.E. Robertson e K. Sparck Jones. "Relevance Weighting of Search Terms". *Journal of the American Society for Information Science*, 27(3):129–146, 1976.

[56] J. Rocchio. *Relevance Feedback in Information Retrieval*. G. Salton (editor), The SMART Retrieval System: Experiments in Automatic Document Processing. Prentice–Hall, Inc., Englewood Cliffs, NJ, 1971.

[57] K. Rose. "Deterministic Annealing for Clustering, Compression, Classification, Regression, and Related Optimization Problems". *IEEE*, 86(11):2210–2239, November 1998.

[58] Gerard Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Addison-Wesley Publishing Company, Inc., New York, 1988.

[59] Gerald Salton e Chris Buckley. "Term-weighting approaches in automatic text retrieval". *Information Processing and Management*, 24(5):513–523, 1988.

[60] Alexander Strehl, Joydeep Ghosh e Raymond Mooney. "Impact of similarity measures on web-page clustering". In: *Proceedings of the 7th Natl. Conf on Artificial Intelligence: Workshop of AI for Web Search (AAAI 2000)*, pp. 58–64, 2000.

[61] C. E. Shannon. "A mathematical theory of communication". *Bell System Technical Journal*, 27:379 – 423, Jul 1948.

[62] Badrul Sarwar, George Karypis, Joseph Konstan e John Reidl. "Item-based collaborative filtering recommendation algorithms". In: *Proceedings of the Tenth International Conference on World Wide Web*, pp. 285–295. ACM Press, 2001.

[63] B. Scholkopf e A. Smola. *Learning with kernels*. MIT Press, 2002.

[64] N. Tang e V. Rao Vemuri. "User-interest-based Document Filtering via Semi-supervised Clustering". In: *Proceedings of the 15th International Symposium on Methodologies for Intelligent Systems (ISMIS 2005)*, 2005.

[65] C.J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

[66] Kiri Wagstaff. *Intelligent Clustering with Instance-level Constraints*. Ph.D. thesis, Cornell University, Aug 2002.

[67] Kiri Wagstaff e Claire Cardie. "Clustering with Instance-level Constraints". In: *Proceedings of the ICML'00: Seventeenth International Conference on Machine*

Learning, pp. 1103–1110, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[68] Kiri Wagstaff, Claire Cardie, Seth Rogers e Stefan Schröedl. "Constrained k-means clustering with background knowledge". In: *Proceedings of the Eighteenth International Conference on Machine Learning (ICML-2001)*, pp. 577–584, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[69] X. Wang e H.J. Hamilton. "Towards An Ontology-Based Spatial Clustering Framework". In: *Proceedings of the Eighteenth Canadian Artificial Intelligence Conference (AI 2005)*, pp. 205–216, Victoria, Canada, Maio 2005.

[70] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan e Stuart Russell. "Distance Metric Learning with Application to Clustering with Side-Information". S. Thrun S. Becker e K. Obermayer editores, In: *Advances in Neural Information Processing Systems 15*, pp. 505–512. MIT Press, Cambridge, MA, 2003.

[71] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan e Stuart Russell. "Distance Metric Learning with Application to Clustering with Side-Information". S. Thrun S. Becker e K. Obermayer editores, In: *Advances in Neural Information Processing Systems 15*, pp. 505–512. MIT Press, Cambridge, MA, 2003.

[72] Hua-Jun Zeng, Qi-Cai He, Zheng Chen, Wei-Ying Ma e Jinwen Ma. "Learning to cluster web search results". In: *Proceedings of the SIGIR '04: 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 210–217, New York, NY, USA, 2004. ACM Press.