UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

TÉCNICAS DE AGRUPAMENTO DE TEXTOS APLICADAS À COMPUTAÇÃO FORENSE

LUÍS FILIPE DA CRUZ NASSIF

ORIENTADOR: EDUARDO RAUL HRUSCHKA

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA
ÁREA DE CONCENTRAÇÃO INFORMÁTICA FORENSE E SEGURANÇA DA INFORMAÇÃO

PUBLICAÇÃO: PPGENE.DM - 094 A/11

BRASÍLIA / DF: 09/2011

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

TÉCNICAS DE AGRUPAMENTO DE TEXTOS APLICADAS À COMPUTAÇÃO FORENSE

LUÍS FILIPE DA CRUZ NASSIF

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE PROFISSIONAL EM INFORMÁTICA FORENSE E SEGURANÇA DA INFORMAÇÃO.

APROVADA POR:

EDUARDO RAUL HRUSCHKA, Doutor, USP
(ORIENTADOR)

ANDERSON CLAYTON ALVES NASCIMENTO, Doutor, UnB
(EXAMINADOR INTERNO)

NELSON FRANCISCO FAVILLA EBECKEN, Doutor, UFRJ
(EXAMINADOR EXTERNO)

DATA: BRASÍLIA/DF, 26 DE SETEMBRO DE 2011.

FICHA CATALOGRÁFICA

NASSIF, LUÍS FILIPE DA CRUZ
Técnicas de Agrupamento de Textos Aplicadas à Computação Forense [Distrito Federal] 2011.
xx, 71p., 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2011).

Dissertação de Mestrado – Universidade de Brasília, Faculdade de Tecnologia. Departamento de Engenharia Elétrica.

1. Agrupamento de dados 2. Mineração de textos
3. Computação Forense

I. ENE/FT/UnB. II. Título (Série)

REFERÊNCIA BIBLIOGRÁFICA

NASSIF, L. F. C. (2011). Técnicas de Agrupamento de Textos Aplicadas à Computação Forense. Dissertação de Mestrado, Publicação PPGENE.DM – 094 A/11, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 71p.

CESSÃO DE DIREITOS

NOME DO AUTOR: Luís Filipe da Cruz Nassif
TÍTULO DA DISSERTAÇÃO: Técnicas de Agrupamento de Textos Aplicadas à Computação Forense.
GRAU/ANO: Mestre/2011.

Luís Filipe da Cruz Nassif
Universidade de Brasília
Campus Universitário Darcy Ribeiro - Asa Norte
CEP 70910-900
Brasília/DF - Brasil

À minha amada esposa Lívia.

AGRADECIMENTOS

Agradeço ao Departamento Polícia Federal e à Universidade de Brasília, por desenvolverem e apoiarem o projeto do Mestrado Profissional em Engenharia Elétrica com Ênfase em Informática Forense e Segurança da Informação, no âmbito do qual esta pesquisa foi desenvolvida, e ao Ministério da Justiça, por fornecer os recursos financeiros necessários por meio do Programa Nacional de Segurança Pública com Cidadania – PRONASCI.

Agradeço ao Professor Eduardo Raul Hruschka por, inicialmente, ter aceitado orientar este projeto sem vislumbrar retorno financeiro, por toda a atenção e dedicação, que dificilmente receberia de outro orientador, por toda a orientação propriamente dita, que foi fundamental para que o trabalho alcançasse o êxito, e pela amizade, que tenho certeza que será duradoura.

Agradeço aos meus colegas de mestrado, principalmente a Mateus e a Oswaldo, pela ajuda com os trabalhos e principalmente pela amizade, que com certeza tornou o mestrado uma atividade mais prazerosa.

Finalmente, agradeço a minha amada e querida esposa Lívia, por todo o apoio, compreensão e dedicação, pois foram semanas que passamos distantes no primeiro ano do curso e muitas horas que deixamos de aproveitar para que eu pudesse fazer trabalhos e desenvolver esta pesquisa, e pelo seu amor, que me deu mais forças para continuar e terminar este trabalho.

RESUMO

TÉCNICAS DE AGRUPAMENTO DE TEXTOS APLICADAS À COMPUTAÇÃO FORENSE

Autor: Luís Filipe da Cruz Nassif
Orientador: Eduardo Raul Hruschka
Programa de Pós-graduação em Engenharia Elétrica
Brasília, setembro de 2011

Em análises periciais de computadores, usualmente são examinados centenas de milhares de arquivos. Grande parte dos dados desses arquivos é constituída por texto não estruturado, cuja análise por parte dos peritos é difícil de ser realizada. Nesse contexto, o uso de métodos automatizados de análise baseados na mineração de textos é de grande interesse. Particularmente, algoritmos de agrupamento podem facilitar a descoberta de conhecimentos novos e úteis nos textos sob análise. Este trabalho apresenta uma abordagem para aplicar agrupamento de documentos em análises periciais de computadores apreendidos durante investigações policiais. Para ilustrar tal abordagem, foi realizado um estudo comparativo de seis algoritmos de agrupamento de dados (*K-means, K-medoids, Single Link, Complete Link, Average Link* e CSPA) aplicados a cinco bases de dados textuais provenientes de investigações reais. Foram realizados experimentos utilizando-se diferentes combinações de parâmetros, totalizando dezoito instanciações diferentes dos algoritmos. Adicionalmente, dois índices de validade relativos (Silhueta e sua versão simplificada) foram utilizados para estimar automaticamente o número de grupos. Estudos relacionados encontrados na literatura se mostram significativamente mais limitados do que o estudo aqui apresentado, especialmente ao se considerar a variedade de algoritmos utilizados e a estimativa automática do número de grupos. Nesse contexto, o presente estudo poderá servir como ponto de partida para aqueles interessados em desenvolver pesquisas neste domínio de aplicação específico. Além disso, os experimentos realizados mostram que os algoritmos hierárquicos *Average Link* e *Complete Link* proporcionaram os melhores resultados. Os algoritmos particionais *K-means* e *K-medoids*, quando adequadamente inicializados, apresentaram resultados similares àqueles obtidos pelos algoritmos hierárquicos. Este estudo também apresenta e discute diversos resultados práticos mais específicos que podem ser úteis para pesquisadores e praticantes de análises forenses computacionais.

ABSTRACT

TEXT CLUSTERING TECHNIQUES APPLIED TO COMPUTER FORENSICS

Author: Luís Filipe da Cruz Nassif

Supervisor: Eduardo Raul Hruschka

Programa de Pós-graduação em Engenharia Elétrica

Brasília, september of 2011

In computer forensic analysis, hundreds of thousands of files are usually analyzed. Most of the data available in these files consists of unstructured text that are hard to be analyzed by human beings. In this context, the use of automated techniques, based on text mining, is of great relevance. In particular, clustering algorithms can help to find new, useful, and potentially actionable knowledge from text files. This work presents an approach that applies document clustering algorithms to forensic analysis of computers seized in police investigations. It was carried out a comparative study of six clustering algorithms – K-means, K-medoids, Single Link, Complete Link, Average Link and CSPA – when applied to five textual databases derived from real cases. A variety of experiments, using different combinations of parameter values, have been performed by running 18 different instantiations of the algorithms under study. In addition, two relative validity indexes for automatically estimating the number of groups – the Silhouette index and its simplified version – have been empirically assessed. To the best of our knowledge, studies of this nature, especially considering a variety of different clustering algorithms and the automatic estimation of the number of clusters, have not been reported in the literature about computer forensics. This study can thus serve as a starting point for researchers interested in developing further research in this particular application domain. In brief, the experiments performed on five real-world datasets show that the hierarchical algorithms known as Average Link and Complete Link provided the best performances. The partitional algorithms K-means and K-medoids, when appropriately initialized, have shown similar performances to those hierarchical algorithms. This study also presents and discusses several practical results for both researchers and practitioners of computer forensic analysis.

SUMÁRIO

RESUMO ..... **XI**

ABSTRACT ..... **XIII**

LISTA DE FIGURAS ..... **XVII**

LISTA DE TABELAS ..... **XIX**

1 INTRODUÇÃO **1**

1.1 Motivação 1

1.2 Trabalhos Relacionados 3

1.3 Objetivo 5

1.4 Organização do Trabalho 6

2 MINERAÇÃO DE TEXTOS **7**

2.1 Introdução 7

2.2 Pré-processamento 8

2.3 Redução da Dimensionalidade 10

2.4 Extração de Padrões 13

2.5 Validação e Interpretação 15

3 AGRUPAMENTO DE DADOS **17**

3.1 Introdução 17

3.2 Medidas de Proximidade 18

3.3 Algoritmos Particionais 20

3.4 Algoritmos Hierárquicos 23

3.5 Algoritmos de Consenso 26

3.6 Índices de Validação 28

3.6.1 Índices de Validação Externos 29

3.6.2 Índices de Validação Relativos 30

4 AVALIAÇÃO EXPERIMENTAL **34**

4.1 Bases de Dados 34

4.2 Ferramentas Utilizadas 35

4.3 Algoritmos e Parâmetros 36

4.4 Resultados 39

4.4.1 Base A *51*
4.4.2 Base B *53*
4.4.3 Base C *55*
4.4.4 Base D *57*
4.4.5 Base E *58*
4.5 CONSIDERAÇÕES FINAIS 60
5 CONCLUSÃO 63
5.1 PRINCIPAIS CONTRIBUIÇÕES 63
5.2 LIMITAÇÕES 64
5.3 TRABALHOS FUTUROS 65
REFERÊNCIAS 67

LISTA DE FIGURAS

Figura 2.1.1 - Processo de mineração de textos 8
Figura 2.2.1 - Matriz atributo-valor 10
Figura 3.1.1 - Diferentes grupos de acordo com diferentes medidas de proximidade 18
Figura 3.2.1 - Matriz de (dis)similaridades 20
Figura 3.3.1 - Partição dos dados em 3 grupos 21
Figura 3.4.1 - Dendrograma dos dados da Figura 3.3.1 obtido pelo algoritmo *SingleLink* 24
Figura 3.6.1.1 - Tabela de contingência 29
Figura 4.4.1 - Silhuetas dos algoritmos Kms100, Kms100* e AL100 42
Figura 4.4.2 - Silhueta e Silhueta Simplificada do algoritmo Kms100* 44
Figura 4.4.3 - ARI dos algoritmos NC e NC100 46
Figura 4.4.4 - Dendrograma obtido por AL100 na base A (K=23) 48
Figura 4.4.5 - Dendrograma obtido por AL100 na base B (K=49) 49
Figura 4.4.6 - Dendrograma obtido por AL100 na base C (K=40) 49
Figura 4.4.7 - Dendrograma obtido por AL100 na base D (K=38) 49
Figura 4.4.8 - Dendrograma obtido por AL100 na base E (K=51) 50

LISTA DE TABELAS

Tabela 4.1.1 - Características das bases de dados utilizadas 35
Tabela 4.3.1 - Resumo dos algoritmos e respectivos parâmetros 37
Tabela 4.4.1 - ARI para todas as bases, média e desvio padrão 40
Tabela 4.4.2 - JC para todas as bases, média e desvio padrão 40
Tabela 4.4.1.1 - Resultados para a base A (N=37 e K=23) 51
Tabela 4.4.2.1 - Resultados para a base B (N=111 e K=49) 53
Tabela 4.4.3.1 - Resultados para a base C (N=68 e K=40) 55
Tabela 4.4.4.1 - Resultados para a base D (N=74 e K=38) 57
Tabela 4.4.5.1 - Resultados para a base E (N=131 e K=51) 59

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Segundo Gantz et al. (2007) o volume de dados existentes no universo digital teria crescido de 161 hexabytes no ano de 2006 para 988 hexabytes no ano de 2010, cerca de 18 vezes a quantidade de informação presente em todos os livros já escritos até 2006, sendo que o volume de dados produzidos continua a crescer de forma exponencial. Dessa forma, a compreensão de tanta informação pelo homem fica prejudicada, tornando imprescindível o uso de técnicas automatizadas de análise e extração de conhecimento dos repositórios de dados. Essa grande quantidade de informação disponível em formato digital tem impacto negativo direto na área da Computação (ou Informática) Forense, que pode ser definida como uma área de conhecimento interdisciplinar que combina elementos da Ciência da Computação e do Direito para coletar e analisar dados de sistemas computacionais para produzir provas admissíveis judicialmente. Normalmente, durante as análises periciais de computadores, é comum encontrar centenas de milhares de arquivos por disco rígido. Organizar toda essa informação é essencial para viabilizar uma pesquisa, análise e utilização das informações disponíveis de forma rápida e eficaz - caso contrário, as informações podem ser subutilizadas.

Atualmente, diversas técnicas já são utilizadas para organizar as informações encontradas em análises forenses de computadores. Podem ser citadas, por exemplo, classificações dos arquivos baseadas em metadados como nome, tamanho, data e hora, localização e tipo dos arquivos, sendo esta última muito utilizada. Mesmo assim, é bastante comum que sejam encontrados entre centenas e dezenas de milhares de arquivos do mesmo tipo, como documentos de texto, imagens, correspondências eletrônicas, páginas de Internet, dentre outros. Assim, as categorias de classificação usuais acabam englobando muitos arquivos, o que não facilita significativamente o processo de perícia dos dados. Além disso, ainda mais importante que os metadados dos arquivos, são os próprios conteúdos dos arquivos, que normalmente são desconsiderados nas abordagens usuais de organização dos dados pelos aplicativos de análise. Frequentemente, dentro de uma mesma categoria de classificação, são encontrados arquivos com conteúdos completamente diferentes. Logo, essas técnicas de organização tradicionais de arquivos em análises forenses computacionais se mostram

ineficazes, tanto em relação à quantidade quanto em relação à qualidade das informações disponibilizadas. Nesse contexto, o uso de métodos de mineração de dados na Computação Forense, com ênfase no padrão de conteúdo dos arquivos, é promissor, pois pode contribuir para uma melhor organização dos dados por assunto e, consequentemente, para a descoberta de informações novas e úteis para as investigações. É importante ressaltar que dados de empresas e organizações são frequentemente alvo de exames periciais, sendo que a maior parte desses dados, cerca de 80%, são não estruturados (Gantz et al., 2007), formados principalmente por texto em linguagem natural. Assim, é de grande importância integrar aos exames periciais métodos provenientes da mineração de textos, a qual nada mais é que uma especialização do processo de mineração de dados, em que é necessário realizar um pré-processamento dos textos para estruturá-los em uma representação mais adequada de se trabalhar computacionalmente.

O processo de mineração de textos pode ser dividido em várias etapas. Uma delas é a extração de padrões, em que são descobertos padrões válidos e úteis nos textos. Essa etapa pode ser dividida, dentre outras, em classificação, agrupamento e sumarização de textos. Em tarefas de classificação, parte dos dados está previamente rotulada em classes. A partir desse conjunto de dados previamente rotulados (também conhecido como conjunto de treinamento), pode-se construir (induzir) um classificador, que essencialmente modela uma função que permite inferir os rótulos de classes para dados não presentes no conjunto de treinamento (i.e., dados ainda não rotulados). Conjuntos de dados rotulados usualmente não estão disponíveis para a realização de análises periciais. Mesmo assumindo que se disponha de uma base de dados rotulada, obtida a partir de perícias anteriores, há poucas chances de que as mesmas classes (possivelmente aprendidas anteriormente por um classificador, num processo de aprendizado supervisionado) continuem sendo válidas para novas amostras de dados, obtidas a partir de outros computadores e vinculados a processos de investigação diferentes. Mais precisamente, a probabilidade de que os dados seriam oriundos de diferentes populações seria alta, o que inviabilizaria a inferência estatística a partir de modelos existentes. Dessa forma, o uso de classificadores raramente é viável na prática pericial. Nesse contexto, métodos de aprendizado não supervisionado são de grande interesse. Mais especificamente, algoritmos de agrupamento de dados são usualmente utilizados em análises exploratórias de dados, quando há pouco ou nenhum conhecimento prévio sobre os dados que serão analisados, pois eles permitem a descoberta de padrões

válidos, úteis e desconhecidos nos dados. Com a utilização de algoritmos de agrupamento, documentos previamente desconhecidos, mas com mesmo padrão de conteúdo, poderiam ser alocados no mesmo grupo, dessa forma, facilitando a análise dos documentos presentes em computadores apreendidos. Nesse sentido, o perito poderia, após analisar alguns documentos de cada grupo, descartar grupos contendo documentos irrelevantes e concentrar a análise nos grupos contendo documentos relevantes. Em outras palavras, o agrupamento de documentos pode evitar o exame de todos os documentos individualmente e, no pior caso, o perito ainda poderia optar por analisá-los em sua totalidade. Assim, a adoção da abordagem pericial baseada em agrupamento de documentos pode melhorar a eficiência do processo de análise pericial de computadores apreendidos, conforme será discutido em maiores detalhes no presente trabalho.

1.2 TRABALHOS RELACIONADOS

Existem poucos trabalhos na literatura reportando o uso de técnicas de agrupamento de dados para Computação Forense. Essencialmente, a maioria dos trabalhos adota parte de algoritmos clássicos para agrupamento de dados - e.g., *Expectation-Maximization* (EM) para aprendizado não supervisionado de mistura de gaussianas, *K-means*, *Fuzzy C-means* e redes neurais do tipo SOM. Esses algoritmos possuem propriedades bem conhecidas e são amplamente usados em aplicações de mineração de dados. Mais especificamente, os algoritmos *K-means* e *Fuzzy C-means* podem ser vistos como casos particulares do EM. Algoritmos do tipo SOM, por sua vez, em geral apresentam bias indutivo semelhante ao *K-means*, sendo computacionalmente mais custosos, entretanto. Apesar de existirem métodos simples para se estimar o numero de grupos (ou equivalentemente para se realizar seleção de modelos ao se usar algoritmos do tipo EM), na prática forense, os algoritmos para agrupamento acima mencionados (EM, *K-means*, *Fuzzy C-means*, SOM) têm sido aplicados para um número de *clusters* pré-determinado pelo usuário. Evidentemente, tal parâmetro é de difícil escolha em situações práticas em que há pouco conhecimento sobre os dados. Nesse sentido, vale antecipar que o presente trabalho faz uso de métodos capazes de estimar o número de grupos a partir dos dados, tornando sua análise mais automática. Antes de apresentar os principais objetivos deste trabalho, cumpre abordar em maiores detalhes alguns trabalhos relacionados encontrados na literatura.

Fei et al. (2005) sugeriram o uso de redes neurais artificiais do tipo mapas auto-organizáveis (*Self Organizing Maps* - SOM) no agrupamento de arquivos, para auxiliar a tomada de decisão dos peritos e tornar o processo de análise mais eficiente. Foram realizados agrupamentos dos arquivos com base na sua data de criação, extensão de arquivo e hora de criação. Assim, foi exemplificado o uso de redes SOM para procurar padrões e visualizar similaridades nos dados. Entretanto, o conteúdo dos arquivos não foi considerado no estudo.

Beebe e Clark (2007) propuseram o uso de algoritmos de agrupamento para agrupar tematicamente os resultados de buscas por palavras chave, obtidos durante os exames periciais. A hipótese seria que o agrupamento dos resultados aumentaria a eficiência do processo de recuperação da informação, pois não seria necessário a revisão de todos os documentos encontrados pelo usuário. Nesse trabalho, os autores utilizaram redes neurais do tipo SOM. Dentre os dois estudos de casos reais realizados, um deles obteve resultados ruins, o que foi atribuído ao pequeno número de nós utilizado na rede neural.

Hadjidj et al. (2009) apresentaram um ambiente integrado de mineração de dados de e-mails para análises periciais, utilizando-se algoritmos de classificação e agrupamento por assunto e por autor. Posteriormente, Iqbal et al. (2010) propuseram a extração de características de estilos de escrita de um conjunto de *e-mails* anônimos para posterior agrupamento das mensagens por autor. Foram avaliadas várias combinações de características, como léxicas, sintáticas, estruturais e específicas de domínio, e três algoritmos de agrupamento: *K-means, Bisecting K-means* e *Expectation Maximization*. Os autores concluíram que os algoritmos de agrupamento são eficazes para separar as mensagens e extrair estilos de escrita únicos por autor. Também relataram que os algoritmos são sensíveis ao número de autores e à quantidade de mensagens por autor. Decherchi et al. (2009) também estudaram a aplicação de agrupamento de dados de e-mails para fins periciais. Os autores utilizaram uma base de dados pública de e-mails da empresa Eron e aplicaram uma variante do algoritmo *K-means* baseada em funções do tipo *Kernel*. Posteriormente, analisaram os resultados subjetivamente e concluíram que os resultados são interessantes e úteis do ponto de vista de uma investigação, pois proporcionam uma visualização geral dos dados, separados por assunto, sem precisar analisar individualmente o conteúdo de todos os arquivos.

Stoffel, Cotofrei e Han (2010) descreveram um método para inferir regras de associação a partir de dados forenses, supostamente fáceis de serem compreendidas por policiais e outros especialistas de domínio. O método é baseado no algoritmo de agrupamento probabilístico *Fuzzy C-Means* (FCM). De acordo com os autores, foram obtidas regras de associação muito boas, apesar de ser bastante difícil de gerar um significado semântico intuitivo para algumas das relações de pertinência, o que pode prejudicar a compreensão das regras por parte dos especialistas de domínio.

Conforme já mencionado, todos os trabalhos encontrados na literatura assumem que o usuário determina, *a priori*, o número de grupos (*clusters*) a serem obtidos. Essa limitação prática pode ser contornada pelo uso de esquemas de algoritmos que basicamente aplicam determinado algoritmo de agrupamento de dados (e.g., K-medias) múltiplas vezes, variando-se a quantidade de grupos, e, a partir do conjunto de partições obtidas, escolhem, de acordo com algum critério numérico, a melhor partição dentre aquelas obtidas (Vendramin et al., 2010). O presente trabalho faz uso de tais métodos que permitem estimar o número de grupos a partir dos dados, dessa forma, facilitando o trabalho do perito, que, na maior parte das situações, dificilmente saberia estimar, *a priori*, o numero de grupos presente em determinada base de dados. Além disso, surpreendentemente, a literatura de Computação Forense não faz menção ao uso de algoritmos clássicos de agrupamento hierárquicos. O presente estudo considera tais algoritmos clássicos, bem como avanços recentes na área de agrupamento de dados – tais como o uso de partições de consenso. Levando-se em conta essas considerações, a próxima seção apresenta os principais objetivos deste trabalho.

1.3 OBJETIVO

Este trabalho tem como principal objetivo analisar e comparar os desempenhos de algoritmos de agrupamento de dados no contexto da Computação Forense – particularmente para analisar os conteúdos de arquivos de computadores apreendidos. Os algoritmos são avaliados quanto aos seus desempenhos no agrupamento de bases textuais obtidas a partir de cinco investigações reais. São comparados os algoritmos particionais K-medias (MacQueen, 1967) e K-medoides (Kaufman e Rousseeuw, 1990; Theodoridis e

Koutroumbas, 2006), os hierárquicos *Single Link* (Florek et al., 1951), *Complete Link* (Sorensen, 1948) e *Average Link* (Sokal e Michener, 1958), e o algoritmo de consenso entre partições CSPA (Strehl e Ghosh, 2002), incluindo algumas variações de parâmetros, totalizando dezoito instanciações diferentes dos algoritmos. Além disso, este trabalho também estuda técnicas para estimar o número de grupos automaticamente, dado que este é um parâmetro crítico de vários algoritmos e é normalmente desconhecido *a priori*. Nesta dissertação, foram avaliados dois índices de validade relativos utilizados para estimar o número de grupos: a Silhueta (Rousseeuw, 1987) e a Silhueta Simplificada (Hruschka et al., 2006). Não foram encontrados na literatura sobre Computação Forense estudos sobre algoritmos de agrupamento hierárquicos e de consenso entre partições, nem sobre a estimativa automática do número de grupos, o que é realizado neste trabalho.

1.4 ORGANIZAÇÃO DO TRABALHO

O restante deste trabalho está dividido conforme descrito a seguir. No capítulo 2, é apresentado o processo de mineração de textos, sendo descritas as suas principais etapas: pré-processamento, redução da dimensionalidade, extração de padrões, validação e interpretação dos resultados. No capítulo 3, descreve-se o processo de agrupamento de dados, incluindo os conceitos usados nesta dissertação relacionados a medidas de proximidade, algoritmos particionais, hierárquicos e de consenso, estimativa do número de grupos e índices de validação. No capítulo 4, são apresentados o método de avaliação, as bases de dados e ferramentas utilizadas, bem como são descritos os experimentos realizados e expostas as discussões dos resultados. Finalmente, no capítulo 5, são resumidas as principais contribuições deste trabalho, suas limitações e trabalhos futuros promissores.

2 MINERAÇÃO DE TEXTOS

2.1 INTRODUÇÃO

Atualmente, o volume de dados produzidos e armazenados nas organizações usualmente impossibilita sua análise sem o uso de técnicas automatizadas. A mineração de dados, de acordo com Fayyad et al. (1996), "é o processo não trivial de descoberta de padrões válidos, novos, úteis e compreensíveis nos dados". O processo de mineração de textos nada mais é que uma especialização da mineração de dados, é uma área interdisciplinar que reúne processamento de linguagem natural, aprendizado de máquina, visualização da informação. Ebecken et al. (2003) definem a mineração de textos como "um conjunto de técnicas e processos que descobrem conhecimento inovador nos textos". Entretanto, a mineração de textos, ao contrário da mineração de dados, lida com dados intrinsecamente não estruturados, sendo necessário realizar um pré-processamento antes de representá-los em uma forma mais fácil de trabalhar computacionalmente.

Os métodos de mineração de textos podem ser classificados em duas categorias: linguísticos e estatísticos (Srivastava e Sahami, 2009). Os métodos linguísticos se baseiam em técnicas de processamento da linguagem natural. Eles procuram construir uma representação semântica que reflita o significado, a estrutura intrínseca e as relações de causalidade presentes no texto. Essa abordagem tem o potencial de prover uma representação mais detalhada do conhecimento e que seja, ao mesmo tempo, compreensível computacionalmente. Entretanto, essa propriedade desejável se mostra, na prática, complexa e de difícil implementação, pois depende de uma ampla experiência do mundo como um todo para produzir conhecimento contextualmente válido. De maneira diametralmente oposta, os métodos estatísticos não levam em consideração o significado semântico do texto. Esses métodos se baseiam numa representação matemática do texto, sendo a mais comum a chamada "*bag-of-words*", na qual cada documento é representado por um vetor de frequências de cada palavra no texto. Entretanto, um ponto fundamental dessa representação é que as relações semânticas e de significado são perdidas juntamente com a ordenação das palavras. Apesar disso, diversas pesquisas têm mostrado que essa representação pode prover resultados extremamente satisfatórios em variadas aplicações, sendo ainda a representação mais favorecida (Srivastava e Sahami, 2009). Por isso, este

trabalho também adota os métodos estatísticos para mineração de textos.

Analogamente ao processo de mineração de dados, o processo de mineração de textos pode ser dividido em várias etapas: pré-processamento, redução da dimensionalidade, extração de padrões, validação e interpretação dos resultados. Trata-se de um processo interativo e iterativo, não sequencial, em que as etapas podem ser executadas mais de uma vez dependendo dos resultados obtidos pelas etapas seguintes, conforme ilustrado pela Figura 2.1.1. Nas próximas seções, serão descritas cada uma das etapas do processo de mineração de textos.

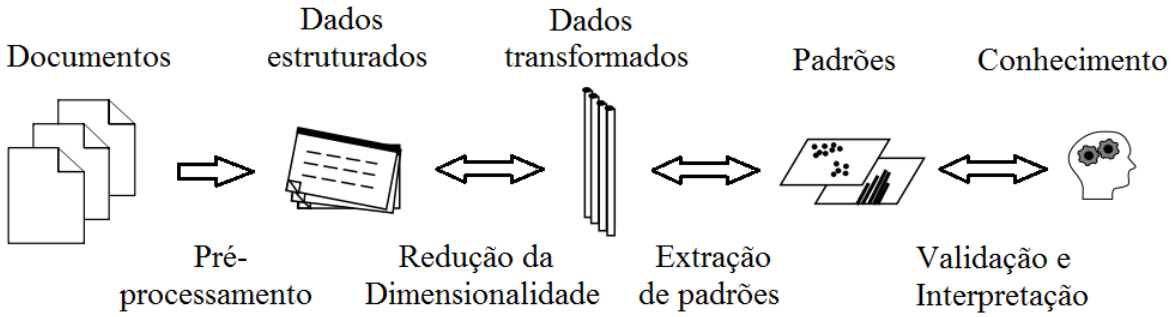

Figura 2.1.1 - Processo de mineração de textos

2.2 PRÉ-PROCESSAMENTO

O objetivo principal da etapa de pré-processamento é representar os textos em uma forma estruturada, mais adequada de ser manipulada pelos algoritmos de extração de padrões. Esta etapa normalmente diferencia os processos de mineração de textos e de mineração de dados, já que nesta última os dados já se encontram estruturados.

Essencialmente, o pré-processamento tem por objetivo identificar nos documentos as suas palavras, ou atributos, mais importantes, ou seja, que capturam melhor as ideias do texto. Para isso, primeiramente é realizado um processo de identificação das palavras, conhecido como *tokenization.*. A abordagem mais simples é considerar como atributos as sequências de letras dentro de um intervalo de comprimento (e.g., entre 4 a 25 letras, conforme utilizado neste trabalho), sendo os outros caracteres considerados separadores dos atributos. Dependendo do domínio da aplicação, podem ser utilizadas expressões regulares para definir os atributos.

Posteriormente, é realizada a eliminação de *stopwords*, que são palavras que normalmente não apresentam significado semântico útil. Dentre elas, podem ser citados artigos, pronomes, preposições e interjeições. Neste trabalho, foram adotadas *stopwords* para os idiomas Português e Inglês. Também podem ser incluídas palavras irrelevantes no domínio. Por exemplo, neste estudo, em que são considerados documentos de texto com extensão "doc", foram incluídas como *stopwords* sequencias de caracteres presentes nos cabeçalhos internos dos documentos, como "dttm", "created", "revised", "print", "microsoft" e "word".

Em seguida, é realizada a identificação e unificação de atributos que possuam o mesmo significado semântico. No caso de variações morfológicas, pode ser feita a redução das palavras aos seus radicais (*stemming*) ou ao seu lema (lematização), que consiste em reunir as diferentes variações de uma palavra sob uma única forma, como num dicionário. Neste trabalho, foi utilizado o processo de *stemming*. Também podem ser utilizados dicionários de sinônimos, chamados de *thesaurus,* em que as palavras são mapeadas para apenas um dos seus sinônimos.

Finalmente, após a geração dos atributos, no método estatístico para a mineração de textos, os documentos são estruturados matematicamente utilizando o modelo de espaço vetorial (Salton e Buckley, 1987), também chamado de abordagem "*bag-of-words*". Assim, os documentos são representados por vetores que registram as ocorrências de cada palavra no texto. Consequentemente, a informação de ordenação das palavras no documento é perdida, juntamente com as relações semânticas e de causalidade. Apesar disso, essa abordagem tem se mostrado eficiente em diversas aplicações, conforme já abordado na introdução deste capítulo.

Mais especificamente, o conjunto dos documentos é representado por uma matriz de pares atributo-valor, em que as linhas representam os documentos e as colunas representam os termos, conforme ilustrado na Figura 2.2.1, na qual *N* representa o número de documentos na coleção, *M* o número de atributos, d_i representa o *i*-ésimo documento, t_j representa o *j*-ésimo termo, ou atributo, e f_{ij} é um valor associado ao *j*-ésimo termo no *i*-ésimo documento.

$$\begin{array}{c} \\ d_1 \\ d_2 \\ \vdots \\ d_N \end{array} \begin{array}{c} \begin{array}{cccc} t_1 & t_2 & \cdots & t_M \end{array} \\ \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1M} \\ f_{21} & f_{22} & \cdots & f_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ f_{N1} & f_{N2} & \cdots & f_{NM} \end{bmatrix} \end{array}$$

Figura 2.2.1 - Matriz atributo-valor

De acordo com Salton e Buckley (1987), o valor de f_{ij} é composto por três componentes:

a) um referente à presença do termo no documento;
b) um referente à distribuição do termo pela coleção de documentos;
c) um referente à normalização, para que documentos de tamanhos diferentes que tratem do mesmo assunto tenham representações similares.

Há várias formas de se calcular os valores de f_{ij}. Elas podem ser divididas em binárias ou baseadas em frequência. Dentre as medidas binárias, pode-se citar a *Document Frequency* (DF) (Salton et al., 1975), na qual é atribuído a f_{ij} o valor 1 se o *j*-ésimo termo está presente no *i*-ésimo documento e zero se não estiver presente. As medidas baseadas em frequência, mais utilizadas, contabilizam o número de vezes que o termo está presente no documento. Dentre elas, podem ser citadas a *Term Frequency* (TF) (van Rijsbergen, 1979), em que é considerada a frequência absoluta do termo no documento, e a *Term Frequency Inverse Document Frequency* (TF-IDF) (Salton e Buckley, 1987), em que a TF é combinada com o inverso da DF, favorecendo termos com alta frequência de documento e que apresentem uma distribuição não uniforme ao longo da coleção. Neste trabalho, é adotada a TF por ser uma medida bastante utilizada na prática, de acordo com a qual o valor de f_{ij} é dado por:

$$f_{ij} = \frac{n_{ij}}{N_i} \tag{2.2.1}$$

Onde n_{ij} é o número de vezes que o *j*-ésimo termo aparece no *i*-ésimo documento e N_i é o número de termos presentes no *i*-ésimo documento da coleção.

2.3 REDUÇÃO DA DIMENSIONALIDADE

No processo de mineração de textos, como cada palavra presente na coleção de documentos representa um atributo, o número de atributos ultrapassa o número de documentos em mais de uma ordem de grandeza (Forman, 2003), gerando matrizes atributo-valor de alta dimensionalidade. Além disso, documentos tematicamente

relacionados têm frequências de atributos similares e não nulas apenas em um subconjunto dos atributos, o qual varia para cada grupo temático de documentos, gerando matrizes atributo-valor muito esparsas (Kriegel et al., 2009).

Geralmente, o desempenho dos algoritmos de reconhecimento de padrões é muito prejudicado com dados esparsos e de alta dimensionalidade, dificuldade conhecida como "maldição da dimensionalidade". Essa expressão foi cunhada por Bellman (1961) para descrever o fato que mais dimensões resultam em mais combinações e inviabilizam uma abordagem de completa enumeração das possibilidades, pois a tabularização e visualização dos dados se torna crescentemente difícil ou mesmo inviável. Adicionalmente, a grande dimensionalidade provoca um alto custo computacional, tornando a execução dos algoritmos muito lenta e até inviável em vários casos.

Por outro lado, conceitos como proximidade, distância ou vizinhança se tornam menos significativos com o aumento da dimensionalidade (Beyer et al., 1999) (Hinneburg et al., 2000). Resumidamente, esses artigos estabelecem que a distância relativa entre o vizinho mais distante e o vizinho mais próximo convergem para zero com o aumento da dimensionalidade *d*:

$$\lim_{d\to\infty} \frac{d_{max}-d_{min}}{d_{min}} \to 0 \qquad (2.3.1)$$

ou seja, a diferença de distância relativa entre o vizinho mais próximo e o mais distante se torna menos discriminante em espaços de alta dimensionalidade. Esse é um problema ainda mais sério do que a simples degradação de performance dos algoritmos. Nesse cenário, é fundamental a aplicação de técnicas de redução da dimensionalidade, ou do número de atributos, para melhorar a eficácia e eficiência dos algoritmos de reconhecimento de padrões. O objetivo é utilizar apenas os atributos mais relevantes para representar os documentos no domínio do problema. Basicamente, as técnicas de redução do número de atributos podem ser divididas em extração de atributos e seleção de atributos.

As técnicas de extração de atributos se caracterizam por gerar novos atributos a partir de uma combinação dos atributos já existentes. Normalmente, as redundâncias entre os atributos são eliminadas, resultando em um conjunto de novos atributos menor que o

original. Dentre essa técnicas, uma das mais conhecida é a *Principal Component Analisys* (PCA) (Jolliffe, 2002), que realiza uma combinação linear de atributos, gerando novos atributos independentes que representam as direções de maior variância dos dados, em ordem decrescente. Também pode ser citada a *Latent Semantic Indexing* (LSI) (Deerwester et al., 1988), que utiliza uma técnica da álgebra linear de fatoração de matrizes, conhecida como *singular value decomposition* (SVD). Entretanto, uma desvantagem dessas técnicas é que o conjunto de atributos gerado, por ser uma combinação linear dos atributos originais, é de difícil interpretação.

As técnicas de seleção de atributos escolhem um subconjunto dos atributos originais. Diferentemente do que acontece na extração de atributos, escolhe-se um subconjunto dos atributos originais sem transformá-los, dessa forma, mantendo seus significados e favorecendo a interpretação do modelo como um todo. Entretanto, o número de combinações de subconjuntos possíveis é muito elevado, especialmente para bases de dados de alta dimensionalidade, como as bases textuais. Mais formalmente, considerando um espaço *d*-dimensional, o número possível de subconjuntos diferentes de atributos a serem avaliados é dado por (Kriegel et al., 2009):

$$\sum_{k=1}^{d} \binom{d}{k} = 2^d - 1 \qquad (2.3.2)$$

Claramente, algoritmos de busca exaustiva são computacionalmente inviáveis na maioria das aplicações práticas, tornando-se comum a adoção de métodos de otimização heurísticos ou aproximados.

Técnicas de seleção de atributos podem ser classificadas em supervisionadas e não supervisionadas (Liu e Yu, 2005). As técnicas supervisionadas levam em consideração o rótulo de classe do conjunto de treinamento e, portanto, não são aplicáveis aos problemas abordados nesta dissertação. No contexto de aprendizado não supervisionado (*clustering*), pode-se lançar mão de técnicas tradicionais tais como *Ranking* pela Frequência de Termos (RTF) (Rijsbergen, 1979), em que são selecionados os atributos com maior somatório de suas frequências ao longo da coleção, e o *Ranking* pela TFIDF (Salton e Buckley, 1987), na qual são selecionados os atributos com maior somatório da TFIDF ao longo da coleção. Neste trabalho, foi utilizada uma técnica simples, mas eficaz, para selecionar atributos que se baseia na Variância do Termo (TV) (Liu et al., 2005), na qual são selecionados atributos com maior variância da frequência do termo ao longo da coleção de documentos. Dessa

forma, a medida não é prejudicada por atributos não discriminantes que têm alta frequência em todos os documentos da coleção. Mais especificamente, a TV pode ser calculada pela equação:

$$TV_j = \sum_{i=1}^{N} \left(f_{ij} - \bar{f_j}\right)^2 \quad (2.3.3)$$

onde f_{ij} é a frequência do *j*-ésimo atributo no *i*-ésimo documento e $\bar{f_j}$ é a média das frequências do *j*-ésimo atributo pelos documentos. Essa medida obteve bom desempenho no trabalho de Liu et al. (2005). Além disso, a TV foi comparada a outras 9 medidas no trabalho de Nogueira (2009), obtendo desempenho superior na avaliação não supervisionada utilizada, apesar de a superioridade não ter sido estatisticamente significante. Pela sua simplicidade, ela foi escolhida para ser utilizada neste trabalho.

2.4 EXTRAÇÃO DE PADRÕES

A próxima etapa do processo de mineração de textos envolve a extração de padrões, a qual pode ser categorizada como uma tarefa preditiva ou descritiva. Tarefas preditivas têm o objetivo de prever o valor de uma ou mais variáveis de objetos da base, considerando-se que já são conhecidos tais valores em outros objetos. Esse conjunto de objetos, cujos valores das variáveis são conhecidos previamente, é chamado de conjunto de treinamento. Algoritmos que utilizam um conjunto de treinamento dos dados são conhecidos como algoritmos de aprendizado de máquina supervisionados (Bishop, 2006). As tarefas preditivas podem ser divididas em regressão e classificação.

A regressão se refere à predição de uma ou mais variáveis com valores contínuos. Normalmente se procura uma função que modele os dados com o menor erro possível. A classificação se refere à predição de uma variável em um conjunto finito de valores discretos possíveis, os quais são chamados classes ou categorias. Dentre as tarefas preditivas, a mais comum na mineração de textos é a classificação, cujo objetivo principal é a categorização automática de documentos de texto. Por exemplo, pode-se ter um conjunto de documentos de texto de uma investigação já categorizados em classes como procurações, extratos bancários, comprovantes de transferência bancária e relatórios de fluxo de caixa. A partir disso, deseja-se saber a qual dessas classes pertence cada um dos documentos encontrados em um computador apreendido, sob a premissa de que pertence necessariamente a alguma delas.

Diferentemente das tarefas preditivas, nas tarefas descritivas os dados não possuem um rótulo de classe conhecido previamente. Logo, nas tarefas descritivas os algoritmos são ditos de aprendizado de máquina não supervisionado, pois não se dispõe de um conjunto de treinamento com rótulos de classe conhecidos *a priori* para as variáveis dependentes. As tarefas descritivas, na mineração de textos, podem ser divididas em aprendizado de regras de associação, agrupamento de textos e sumarização de textos.

O aprendizado de regras de associação permite descobrir relações lógicas entre atributos correlacionados em grandes bases de documentos (Agrawal et al., 1993). As relações lógicas inferidas são relações da forma $X \rightarrow Y$, em que X e Y são subconjuntos disjuntos de atributos da coleção de documentos, chamados de antecedente e consequente da regra, respectivamente. Como a quantidade de atributos em bases textuais é muito grande, o número de regras de associação obtidas também tende a ser grande. Logo, as regras de associação devem ser trabalhadas durante o processo interativo de mineração, para que possa ser extraído um conhecimento compreensível.

O agrupamento de dados, também conhecido como *clustering*, tem o objetivo de induzir grupos de dados, de tal forma que os objetos pertencentes ao mesmo grupo sejam mais semelhantes entre si do que os objetos pertencentes a grupos distintos (Everitt, 2001). Os algoritmos de agrupamento procuram encontrar uma estrutura desconhecida nos dados. Sob outra perspectiva, tais algoritmos buscam organizar os dados de uma forma conveniente e válida (Jain e Dubes, 1988). Algoritmos de agrupamento são normalmente utilizados em análises exploratórias de dados, quando se dispõe de pouco ou nenhum conhecimento sobre eles, como é o caso dos dados analisados em exames periciais. Nessas aplicações, bases de treinamento com rótulos de classes (para documentos típicos) usualmente não estão disponíveis, até mesmo porque não se conhecem, *a priori*, as classes de documentos que poderiam ser encontradas. Nesse contexto, algoritmos para aprendizado não supervisionado são de grande interesse. O próximo capítulo abordará em maiores detalhes os métodos de agrupamento de dados.

Por último, a sumarização automática de textos objetiva reduzir um documento textual ou uma coleção de documentos a um conjunto pequeno de palavras, frases ou parágrafos que

mantenham as ideias principais do texto. Pode ser dividida em métodos de extração e de abstração. Os de extração procuram selecionar um subconjunto das palavras ou frases existentes no texto original para formar o resumo. Diferentemente, os métodos de abstração constroem uma representação semântica interna e, a partir dela, usam técnicas de geração de linguagem natural para criar um resumo próximo daquilo que um humano produziria. As técnicas de abstração podem gerar palavras não presentes no texto original. Como os métodos de abstração são relativamente recentes, seu desempenho atualmente é inferior àqueles apresentados pelos métodos de extração.

2.5 VALIDAÇÃO E INTERPRETAÇÃO

Após a extração dos padrões, eles devem ser avaliados e interpretados pelo especialista de domínio no contexto do problema, a fim de verificar se os padrões produzidos são válidos e úteis ao objetivo final do processo. Na validação de tarefas preditivas de mineração, pode-se, por exemplo, verificar a acurácia de um classificador na rotulação de um conjunto de dados de teste, cujas classes dos objetos são conhecidas previamente. No caso de tarefas descritivas, como em agrupamento de dados, a validação dos padrões produzidos é mais difícil, pois os padrões são novos, e as tarefas em si são subjetivas, sendo mais complicada a avaliação com medidas objetivas. Do ponto de vista científico, entretanto, avaliações objetivas podem ser realizadas, lançando-se mão de experimentos controlados e de índices de validação externos, conforme será abordado em maiores detalhes nos capítulos seguintes. Na prática, a avaliação subjetiva por parte de um especialista de domínio é fundamental. Além disso, a quantidade de padrões produzidos pode ser muito grande, especialmente no processo de mineração de textos, podendo ser necessária uma filtragem dos padrões para que sejam apresentados apenas aqueles mais interessantes.

No caso da interpretação dos resultados, deve ser verificado se os padrões produzidos são úteis e compreensíveis. Nesse sentido, técnicas de visualização dos dados são muito úteis para a etapa de validação e interpretação dos resultados, podendo fornecer uma representação visual dos dados mais intuitiva e facilitando a compreensão dos padrões produzidos. Algumas técnicas de extração de padrões já fornecem uma representação visual dos dados. Por exemplo, os algoritmos de agrupamento hierárquico têm como resultado final uma representação dos dados em árvore, na forma de um dendrograma.

Caso o processo de validação e interpretação identifique que os padrões produzidos não são válidos, úteis ou compreensíveis, deve-se retornar para alguma das etapas anteriores do processo de mineração, a fim de aperfeiçoá-lo. A escolha de outros atributos durante a redução da dimensionalidade dos dados ou a execução do algoritmo de extração de padrões com outros parâmetros são etapas comumente repetidas durante o processo. Ou seja, o ciclo é executado iterativa e interativamente, até que os resultados sejam válidos, compreensíveis e úteis na produção do conhecimento final.

Esta dissertação procurou abordar a avaliação dos resultados obtidos via algoritmos de agrupamento tanto de maneira prática, contribuindo, dessa forma, para com a literatura especializada em Computação Forense, quanto estatística, nesse último caso, contribuindo para a literatura especializada no uso de diferentes algoritmos de agrupamento para aplicações específicas. Vale destacar que essas duas formas de avaliação são complementares e interdependentes, conforme ficará evidente ao serem apresentados os resultados experimentais obtidos.

3 AGRUPAMENTO DE DADOS

3.1 INTRODUÇÃO

Organizar dados em grupos é uma das formas mais fundamentais de conhecimento. O agrupamento de dados, ou *clustering*, tem o objetivo de induzir grupos nos dados, de forma que objetos pertencentes ao mesmo grupo sejam mais similares entre si do que objetos pertencentes a grupos diferentes. De acordo com Jain e Dubes (1988), os algoritmos de agrupamento procuram encontrar uma estrutura desconhecida nos dados, uma forma conveniente e válida de organizar os dados. Everitt (2001) apresenta algumas definições para agrupamento de dados:

Def. 1: "Um grupo (cluster) é um conjunto de entidades semelhantes, e entidades pertencentes a diferentes grupos não são semelhantes."

Def. 2: "Um grupo é uma aglomeração de pontos no espaço tal que a distância entre quaisquer dois pontos no grupo é menor do que a distância entre qualquer ponto no grupo e qualquer ponto fora deste."

Def. 3: "Grupos podem ser descritos como regiões conectadas de um espaço multidimensional contendo uma densidade de pontos relativamente alta, separada de outras tais regiões por uma região contendo uma densidade relativamente baixa de pontos."

Conforme se pode observar, as definições acima podem ser capturadas matematicamente de diferentes formas, por exemplo, usando-se diferentes conceitos de similaridade. Essa é uma das razões da existência de uma grande variedade de algoritmos de agrupamento de dados. Diferentemente do que ocorre em tarefas de classificação de dados, o agrupamento de dados não necessita de um subconjunto de treinamento dos dados rotulados, que apresentem categorias de classe previamente conhecidas, sendo considerada uma tarefa de aprendizado não supervisionado. Assim, o agrupamento de dados é utilizado em análises exploratórias de dados, quando há pouco ou nenhum conhecimento prévio sobre a estrutura dos dados, como é o caso dos arquivos analisados em exames periciais. Nessas aplicações, normalmente não se dispõe de conjuntos de treinamento com rótulos de classe conhecidos, até porque não se conhecem previamente quais classes de documentos podem ser encontradas nos computadores. Mesmo que se construísse uma base de treinamento a partir de perícias anteriores, dificilmente as mesmas classes continuariam válidas para outras

amostras de dados, provenientes de outros computadores apreendidos, no curso de investigações diferentes. Nesse contexto, algoritmos de agrupamento de dados são mais adequados do que algoritmos de classificação.

Vale observar que o agrupamento de dados é uma tarefa inerentemente subjetiva. Por exemplo, pessoas diferentes podem fazer agrupamentos diferentes com o mesmo conjunto de objetos, de acordo com o conceito de similaridade utilizado. A título de ilustração, quais seriam os grupos de objetos presentes na figura 3.1.1? Por um lado, pode-se considerar que existam grupos de objetos pretos, cinzas ou brancos, se a medida de similaridade considerada for a cor, por exemplo. Por outro lado, pode-se observar grupos de quadrados, triângulos ou círculos, se a medida utilizada for a forma geométrica. Ainda, pode-se categorizar os grupos de acordo com suas distâncias relativas em objetos à esquerda, ao centro ou à direita. Assim, não há um agrupamento (conjunto de grupos, também denominado de partição) certo ou errado, pois o conceito de grupo depende da medida de similaridade utilizada e do objetivo que se pretende alcançar no processo de agrupamento de dados.

Figura 3.1.1 - Diferentes grupos de acordo com diferentes medidas de proximidade

3.2 MEDIDAS DE PROXIMIDADE

Uma medida de proximidade pode tanto se referir a uma similaridade quanto a uma dissimilaridade, ou distância (Jain e Dubes, 1988). Uma medida de similaridade mede o quanto dois objetos são semelhantes (ou, equivalentemente, estão próximos entre si), enquanto uma medida de dissimilaridade mede o quanto dois objetos são diferentes (ou estão distantes entre si). Caso a similaridade ou a dissimilaridade estiverem normalizadas no intervalo [0,1] uma pode ser obtida subtraindo-se a outra do valor 1. Conforme exemplificado anteriormente, a escolha de uma medida de similaridade ou de dissimilaridade adequada é fundamental em problemas de agrupamento, tendo grande impacto no resultado final do processo. Neste trabalho, serão consideradas medidas de

dissimilaridade para dados contínuos. Assim, é desejável que uma medida de dissimilaridade entre dois objetos $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ apresente as seguintes propriedades (Xu e Wunsch, 2009) $\forall \boldsymbol{x}_i, \boldsymbol{x}_j$:

1. $d(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$ (não negatividade)
2. $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = d(\boldsymbol{x}_j, \boldsymbol{x}_i)$ (simetria)

Além disso, a medida de dissimilaridade é chamada de métrica, caso apresente as seguintes propriedades adicionais $\forall \boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k$:

3. $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0 \leftrightarrow \boldsymbol{x}_i = \boldsymbol{x}_j$ (reflexividade)
4. $d(\boldsymbol{x}_i, \boldsymbol{x}_k) \leq d(\boldsymbol{x}_i, \boldsymbol{x}_j) + d(\boldsymbol{x}_j, \boldsymbol{x}_k)$ (desigualdade triangular)

Existem muitas medidas de distância reportadas na literatura. Considerando que os dados, também denominados de objetos ou padrões, sejam representados por vetores $\boldsymbol{x}_i = [\boldsymbol{x}_{i1} \ldots \boldsymbol{x}_{im}]$ num espaço *m*-dimensional, uma medida de dissimilaridade bastante utilizada na prática é a distância euclidiana:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_{k=1}^{m} (x_{ik} - x_{jk})^2} \tag{3.2.1}$$

a qual é invariante sob transformações lineares do espaço e tende a formar grupos hiperesféricos (Duda et al., 2001). A distância euclidiana é um caso particular da distância de Minkowski (para $p = 2$):

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt[p]{\sum_{k=1}^{m} (x_{ik} - x_{jk})^p} \tag{3.2.2}$$

a partir da qual podem ser derivadas outras medidas de distância, como a distância de Manhattan ($p = 1$) e a distância suprema ($p \to \infty$). Em processos de mineração de textos, é comum utilizar o cosseno como medida de similaridade entre documentos (Baeza-Yates e Ribeiro-Neto, 1999), a qual realiza uma normalização implícita dos dados em função do tamanho dos documentos. Por isso, ela foi adotada neste trabalho. Além disso, o cosseno é invariante sob rotação do espaço e tende a formar grupos hiper-cônicos. Assim, a distância entre dois documentos é definida por:

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 - \cos(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 - \frac{\boldsymbol{x}_i \cdot \boldsymbol{x}_j^T}{\|\boldsymbol{x}_i\| \|\boldsymbol{x}_j\|} \tag{3.2.3}$$

Além da distância baseada no cosseno, outra medida de distância utilizada neste trabalho é

a distância de Levenshtein (1966), também chamada de distância de edição, utilizada para computar a diferença ou distância entre duas sequências de caracteres (*strings*). A distância de Levenshtein é definida como a quantidade mínima de edições necessárias para transformar uma *string* na outra, sendo que as operações de edição possíveis são: inserção, deleção ou substituição de um caractere. Neste trabalho, foi utilizada uma versão normalizada da distância de Levenshtein no intervalo [0,1], dividindo o resultado pelo valor máximo de distância possível, ou seja, pela quantidade de caracteres da maior *string* envolvida no cálculo.

Há casos em que os objetos não podem ser mapeados num espaço vetorial devido a peculiaridades dos seus atributos, como sequências de caracteres, por exemplo, mas ainda é possível obter uma medida de dissimilaridade entre os objetos. Nesses casos, os dados, ao invés de serem representados por uma matriz atributo-valor, são descritos por uma matriz de (dis)similaridade:

$$\begin{array}{c} \\ \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_n \end{array} \begin{array}{c} \begin{array}{cccc} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_m \end{array} \\ \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1m} \\ d_{21} & d_{22} & \cdots & d_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nm} \end{bmatrix} \end{array}$$

Figura 3.2.1 - Matriz de (dis)similaridades

onde o valor d_{ij} representa a distância ou similaridade entre os objetos $\boldsymbol{x}_i$ e $\boldsymbol{x}_j$. Além disso, vários algoritmos de agrupamento não necessitam de uma representação vetorial dos objetos, mas apenas dos valores de distância entre eles, recebendo como entrada uma matriz de (dis)similaridades (Jain e Dubes, 1988) (Tan et al., 2006) (Xu e Wunsch II, 2009).

3.3 ALGORITMOS PARTICIONAIS

Seja o conjunto de dados definido por $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2 \ldots, \boldsymbol{x}_N\}$. Uma partição rígida dos dados é um conjunto $\boldsymbol{P} = \{\boldsymbol{C}_1, \boldsymbol{C}_2 \ldots \boldsymbol{C}_K\}$ tal que $\boldsymbol{C}_1 \cup \boldsymbol{C}_2 \cup \cdots \cup \boldsymbol{C}_K = \boldsymbol{X}$, $\boldsymbol{C}_i \neq \emptyset$ e $\boldsymbol{C}_i \cap \boldsymbol{C}_j = \emptyset$ para $i \neq j$. Ou seja, uma partição dos dados é uma divisão dos dados em *K* subconjuntos disjuntos não vazios. Na Figura 3.3.1 é apresentado um exemplo de um particionamento de dados bidimensionais em três grupos.

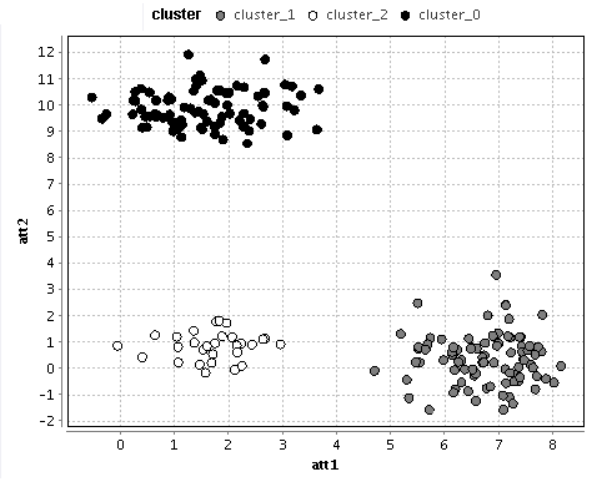

Figura 3.3.1 - Partição dos dados em 3 grupos

Os algoritmos de agrupamento particionais têm o objetivo de gerar uma partição rígida dos dados, segundo algum critério previamente definido. Sob o ponto de vista de otimização combinatória, o número de partições (*NP*) possíveis de organizar *N* objetos em *K* grupos é dado por Liu (1968):

$$NP(N,K) = \frac{1}{K!}\sum_{i=0}^{K}(-1)^i \binom{K}{i}(K-i)^N \tag{3.3.1}$$

Por exemplo, há 34.105 partições possíveis de 10 objetos em 4 grupos e esse número aumenta para 2.436.684.974.110.751 se consideradas partições de 25 objetos em 5 grupos. Ocorre que o número de possibilidades aumenta aproximadamente na razão $K^N/K!$. Assim, é inviável computacionalmente realizar uma busca exaustiva por todas as possibilidades para encontrar uma solução ótima global para o problema. Por isso, os algoritmos de agrupamento normalmente procuram otimizar uma função objetivo de acordo com alguma heurística, como subida de encosta. Uma das funções objetivo mais comuns é a soma do erro quadrático (*SEQ*) intra-grupos, dada por:

$$SEQ = \sum_{i=1}^{K}\sum_{\boldsymbol{x}\in \boldsymbol{C}_i} d(\boldsymbol{x}, \overline{\boldsymbol{x}}_i)^2 \tag{3.3.2}$$

Onde $\overline{\boldsymbol{x}}_i$ é o centroide do *i*-ésimo grupo, definido por:

$$\overline{\boldsymbol{x}}_i = \frac{1}{n_i}\sum_{\boldsymbol{x}\in \boldsymbol{C}_i} \boldsymbol{x} \tag{3.3.3}$$

Onde n_i é o número de objetos pertencentes ao grupo $\boldsymbol{C}_i$. O algoritmo das K-médias (MacQueen, 1967) procura minimizar a função objetivo na equação 3.3.2, segundo um procedimento iterativo de escalada de encosta, descrito a seguir:

1. escolher aleatoriamente *K* centroides iniciais dentre os dados;

2. associar cada objeto ao centroide mais próximo, formando *K* grupos;
3. atualizar os centroides, segundo a equação 3.3.3;
4. repetir os passos 2 e 3 até convergir, isto é, até que os grupos não mudem ou até que seja atingido um número máximo de iterações.

Esse algoritmo é amplamente utilizado na prática devido à sua simplicidade e eficiência computacional, de complexidade O(*NMK*), e, por isso, é indicado para grandes bases de dados (Wu e Kumar, 2009). Entretanto, o algoritmo das K-médias é muito sensível à inicialização dos centroides iniciais e usualmente converge para soluções que representam mínimos locais da função objetivo. Para tentar minimizar esse problema, uma estratégia comum é executar o algoritmo várias vezes, com inicializações diferentes dos centroides, sendo escolhida a configuração final que tenha menor valor de *SEQ* em (3.3.2). Outra estratégia, que pode ser usada concomitantemente, é escolher os centroides iniciais de acordo com alguma heurística. No trabalho de Peterson et al. (2010), são comparadas onze técnicas diferentes de inicialização dos centroides. Uma delas, proposta por Mirkin (2005), tem o objetivo é inicializar os centroides em objetos distantes, da seguinte forma:

1. Selecionar aleatoriamente um protótipo inicial;
2. Para cada objeto restante $\boldsymbol{x}_i$, definir d_i como a distância para o protótipo mais próximo;
3. Definir como novo protótipo o objeto com maior valor de d_i;
4. Repetir os passos 2 e 3 até que sejam escolhidos *K* protótipos iniciais.

Essa técnica foi utilizada neste trabalho para tentar minimizar o problema dos mínimos locais. Além disso, ela privilegia a separação de *outliers*, que são objetos muito diferentes dos demais, e torna o algoritmo com várias inicializações mais estável, pois há menos possibilidades de inicializações diferentes. Entretanto, o algoritmo foi adaptado neste trabalho, sendo o passo 1 alterado para "escolher aleatoriamente um objeto como primeiro centroide". Assim, o custo da inicialização deixa de ser quadrático e se torna linear com o número de objetos (para $K \ll N$), tornando essa técnica mais adequada para inicializar algoritmos lineares com o número de objetos, como o K-medias.

Outro algoritmo, muito semelhante ao K-médias, é o K-medoides (Kaufman e Rousseeuw, 1990). Ele pode ser usado quando se dispõe de uma matriz de dissimilaridades entre os

objetos, não havendo necessidade de representá-los num espaço vetorial. Por exemplo, esse algoritmo foi utilizado, neste trabalho, em agrupamentos baseados na distância de Levenshtein entre os nomes dos arquivos, os quais não podem ser representados como vetores facilmente como o conteúdo dos arquivos. Outro uso do K-medoides, neste trabalho, consiste no agrupamento de consenso entre partições, que é realizado a partir de uma matriz de dissimilaridades, conforme será abordado na seção 3.5. Entretanto, ao invés de computar centroides, esse algoritmo utiliza medoides, que são objetos representativos dos grupos e que apresentam menor distância média em relação aos objetos dos seus respectivos grupos. O algoritmo K-medoides pode ser descrito pelos seguintes passos:

1. escolher aleatoriamente *K* medoides iniciais dentre os dados;
2. associar cada objeto ao medoide mais próximo, formando *K* grupos;
3. para cada grupo, escolher como medoide o objeto que minimize a soma dos erros quadráticos (equação 3.3.2, utilizando o medoide ao invés do centroide);
4. repetir os passos 2 e 3 até que os grupos não mudem ou até que seja atingido um número máximo de iterações.

Porém, esse algoritmo possui complexidade computacional no mínimo quadrática em relação ao número de objetos, pois é necessário computar uma matriz de dissimilaridades. Entretanto, além de não precisar computar centroides, uma vantagem em relação ao K-médias é que ele é menos suscetível à presença de *outliers* (Kaufman e Rousseeuw, 1990). Além disso, o K-medoides também pode convergir para mínimos locais. Por isso, a adaptação da técnica de inicialização de Mirkin (2005) também foi utilizada com o K-medoides, neste estudo.

3.4 ALGORITMOS HIERÁRQUICOS

Os algoritmos hierárquicos organizam os dados em uma estrutura hierárquica, que pode ser interpretada como uma sequência de partições rígidas aninhadas (Jain e Dubes, 1988). Uma partição $\boldsymbol{P}_1$ está aninhada em uma partição $\boldsymbol{P}_2$ se todos os grupos de $\boldsymbol{P}_1$ são subconjuntos dos grupos de $\boldsymbol{P}_2$, ou seja, os grupos de $\boldsymbol{P}_2$ são formados pela união de grupos de $\boldsymbol{P}_1$. Assim, o resultado final de um algoritmo hierárquico pode ser descrito por uma árvore binária, chamada de dendrograma, conforme ilustrado pela Figura 3.4.1. Em

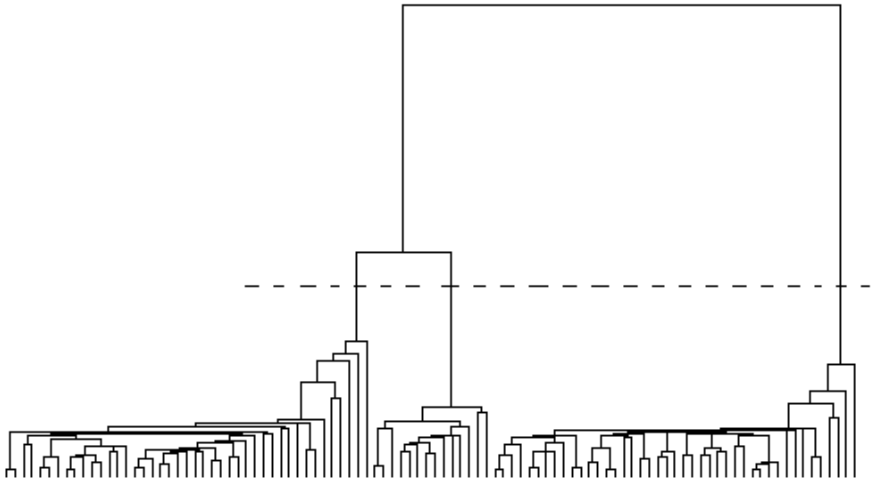

Figura 3.4.1 - Dendrograma dos dados da Figura 3.3.1 obtido pelo algoritmo *SingleLink*

um dendrograma, o nó raiz representa o conjunto completo de dados, enquanto as folhas representam cada objeto individual. Um nó interno representa um grupo formado pela união de dois grupos, unitários ou não. A altura de um nó interno é proporcional à distância entre os dois grupos que ele une. Essa representação visual provê uma descrição muito informativa da estrutura dos dados, podendo ser utilizada na visualização de dados de alta dimensionalidade, como texto, e interpretação dos padrões descobertos.

Os algoritmos hierárquicos podem ser classificados em divisivos e aglomerativos (Xu e Wunsch II, 2009). Os divisivos começam com um único grupo contendo todos os objetos, o qual é progressivamente dividido em grupos menores, até que todos os objetos sejam posicionados em N grupos unitários. Um algoritmo divisivo pode ser construído pela aplicação recursiva de um algoritmo particional. Os algoritmos aglomerativos executam o processo inverso, começando com N grupos unitários que são sucessivamente unidos até que todos os objetos estejam contidos em um único grupo. Este trabalho aborda algoritmos aglomerativos, por serem mais utilizados em geral.

Um algoritmo aglomerativo pode ser descrito genericamente pelos seguintes passos (adaptados de Xu e Wunsch II, 2005):

1. Seja $\widetilde{N}$ o número de grupos atual. Comece com $\widetilde{N} = N$ e calcule a respectiva a matriz de distâncias;

2. Seja $\boldsymbol{C}_i, 1 \leq i \leq \widetilde{N}$, um grupo qualquer. Faça $\boldsymbol{C}_i = \boldsymbol{C}_i \cup \boldsymbol{C}_j$ tal que $d(\boldsymbol{C}_i, \boldsymbol{C}_j) = d_{min}(\boldsymbol{C}_k, \boldsymbol{C}_l)\ \forall\ k, l \in \{1..\widetilde{N}\}, k \neq l$, e descarte o grupo $\boldsymbol{C}_j$;
3. Faça $\widetilde{N} = \widetilde{N} - 1$ e atualize $d(\boldsymbol{C}_i, \boldsymbol{C}_j)\ \forall\ j \in \{1..\widetilde{N}\}$ na matriz de distâncias;
4. Repita os passos 2 e 3 até obter $\widetilde{N} = 1$.

Entretanto, ainda é necessário definir o conceito de distância entre dois grupos $d(\boldsymbol{C}_i, \boldsymbol{C}_j)$. Diversos algoritmos hierárquicos aglomerativos se diferenciam por essa definição de distância. A seguir são apresentados alguns algoritmos e a sua respectiva definição de $d(\boldsymbol{C}_i, \boldsymbol{C}_j)$:

1. *Single Link* (Florek et al., 1951):

$$d(\boldsymbol{C}_i, \boldsymbol{C}_j) = d_{min}(\boldsymbol{x}, \boldsymbol{y}) \mid \boldsymbol{x} \in \boldsymbol{C}_i, \boldsymbol{y} \in \boldsymbol{C}_j \tag{3.4.1}$$

   ou seja, a distância entre dois grupos é a distância mínima entre dois objetos, pertencentes a grupos diferentes. Esse algoritmo induz grupos contíguos, baseados nas adjacências entre os objetos, podendo assumir formas variadas. Entretanto é sensível a *outliers* e grupos pouco separados, podendo provocar encadeamento ou mistura entre os grupos (Everitt et al., 2001);

2. *Complete Link* (Sorensen, 1948):

$$d(\boldsymbol{C}_i, \boldsymbol{C}_j) = d_{max}(\boldsymbol{x}, \boldsymbol{y}) \mid \boldsymbol{x} \in \boldsymbol{C}_i, \boldsymbol{y} \in \boldsymbol{C}_j \tag{3.4.2}$$

   ou seja, a distância entre dois grupos é a distância máxima entre dois objetos, pertencentes a grupos diferentes. Essa distância induz grupos de formas globulares. Ela é menos sensível a presença de *outliers*, porém tende a dividir grupos de diâmetros similares (Tan et al., 2006).

3. *Average Link* (Sokal e Michener, 1958):

$$d(\boldsymbol{C}_i, \boldsymbol{C}_j) = |\boldsymbol{C}_i|^{-1}|\boldsymbol{C}_j|^{-1} \sum_{\boldsymbol{x} \in \boldsymbol{C}_i, \boldsymbol{y} \in \boldsymbol{C}_j} d(\boldsymbol{x}, \boldsymbol{y}) \tag{3.4.3}$$

   ou seja, a distância entre dois grupos é a distância média entre pares de objetos, pertencentes a grupos diferentes. Essa distância é um meio termo das duas distâncias anteriores. Também induz grupos hiperesféricos, é menos sensível à presença de *outliers* e tende a formar grupos com diâmetros similares (Tan et al., 2006).

Há outras formas de definir a distância entre dois grupos, como a distância ente os centroides dos grupos ou o aumento do erro quadrático após a união dos grupos (Ward,

1963). Os algoritmos hierárquicos têm complexidade computacional no mínimo quadrática com relação ao número de objetos, pois é necessário computar uma matriz de dissimilaridades $N \times N$. Entretanto, apresentam algumas vantagens, pois o resultado do agrupamento é independente da ordem de apresentação dos objetos, conseguem lidar com a presença de *outliers*, fornecem uma visualização final intuitiva e de fácil interpretação pelo usuário (porém subjetiva) e não é necessário fixar o número de grupos previamente. O número de grupos pode ser definido pelo usuário pela inspeção visual do dendrograma ou com o auxílio de métodos numéricos, através de um corte horizontal do dendrograma. Por exemplo, na Figura 3.4.1, observa-se uma linha tracejada horizontal que corta o dendrograma em três pontos, extraindo uma das partições aninhadas, constituída por três grupos. Neste trabalho, os métodos hierárquicos, após sua execução, sofrem um corte horizontal, através de métodos numéricos abordados na seção 3.6, sendo então extraída uma das partições aninhadas para ser avaliada posteriormente.

3.5 ALGORITMOS DE CONSENSO

Além dos algoritmos apresentados nas seções anteriores, existem centenas de outros reportados na literatura. Strehl e Ghosh (2002) introduziram o problema de combinar múltiplas partições dos dados em uma única partição de consenso, sem ter que necessariamente acessar os atributos originais dos objetos, mas apenas os rótulos fornecidos pelas partições existentes, descrevendo um arcabouço para reutilização de conhecimento. A combinação de várias partições pode fornecer resultados com maior acurácia e robustez. O conjunto de partições existentes é denominado um *ensemble* (ou agregação) de partições. Essas partições podem ser obtidas por diversas formas, como pela execução de vários algoritmos de agrupamento, pela execução de um único algoritmo de agrupamento com parâmetros de inicialização diferentes - como número de grupos ou posições iniciais de centroides - pela utilização de diferentes atributos para representar os objetos ou pela alteração da ordem de apresentação dos objetos em algoritmos de agrupamento sequenciais. No trabalho de Strehl e Ghosh (2002), são propostos três algoritmos de agrupamento de consenso entre partições diferentes: *Cluster-based Similarity Partitioning Algorithm* (CSPA), *HyperGraph Partitioning Algorithm* (HGPA) e *Meta-CLustering Algorithm* (MCLA). Dentre eles, CSPA e MCLA obtiveram as melhores performances nos experimentos realizados naquele estudo, apresentando respectivamente

complexidades computacionais de $O(N^2KR)$ e $O(NK^2R^2)$ segundo os autores, onde *N, K* e *R* são respectivamente o número de objetos, grupos e partições. Quanto mais o número de grupos *K* se aproxima de *N* – caso das bases de dados utilizadas neste trabalho – tanto mais estes algoritmos tendem a apresentar a mesma complexidade computacional em relação ao número de objetos, e, como o algoritmo CSPA apresenta uma abordagem teórica mais simples, ele foi escolhido para ser utilizado nos experimentos deste trabalho.

O algoritmo CSPA é baseado em um mecanismo de votação para combinar o ensemble de partições, as quais podem apresentar diferentes números de grupos. Define-se então uma medida de similaridade entre os objetos, que essencialmente reflete a proporção com que dois objetos estão no mesmo grupo no conjunto de partições disponíveis, dando origem a uma matriz de similaridade denominada matriz de coassociação (Fred e Jain, 2005). Basicamente, a presença de dois objetos em um mesmo grupo, em cada partição do ensemble, é computada como um voto de similaridade, ou coassociação, entre eles. Assim, cada elemento s_{ij} da matriz de similaridade pode ser definido como uma fração ponderada das partições do ensemble em que os objetos $\boldsymbol{x}_i$ e $\boldsymbol{x}_j$ estão presentes no mesmo grupo, ou seja:

$$s_{ij} = \sum_{l=1}^{P} w_l \cdot a_l(i,j) / \sum_{l=1}^{P} w_l \qquad (3.5.1)$$

onde $a_l(i,j) = \begin{cases} 0, g_l(i) \neq g_l(j) \\ 1, g_l(i) = g_l(j) \end{cases}$ sendo *P* o número de partições no ensemble, w_l o peso da *l*-ésima partição do ensemble e $g_l(i)$ o rótulo do grupo que contém o objeto $\boldsymbol{x}_i$ na *l*-ésima partição do ensemble. Depois de computada a matriz de coassociação, pode ser utilizado qualquer algoritmo de agrupamento que trabalhe a partir de uma matriz de (dis)similaridade para obter o particionamento final de consenso.

Neste trabalho, o algoritmo CSPA é utilizado com ensembles de partições gerados de duas formas diferentes. Na primeira delas, o algoritmo das K-médias é executado diversas vezes, utilizando subconjuntos de atributos diferentes em cada execução, fornecendo partições diversificadas. Assim, cada partição do ensemble é gerada a partir de atributos diferentes dos dados, com o objetivo de obter partições independentes e variadas que retratem diferentes características dos documentos. Nesse ensemble, todas as partições têm o mesmo peso no agrupamento de consenso. O outro tipo de ensemble utilizado é composto por apenas duas partições dos dados. Uma delas é obtida pelo algoritmo K-

medoides, a partir das dissimilaridades entre as nomenclaturas dos arquivos, utilizando a distância de Levenshtein (ver seção 3.2). A outra partição é obtida pelo algoritmo das K-medias, a partir das dissimilaridades entre os conteúdos dos arquivos, utilizando a distância baseada no cosseno (equação 3.2.3). A hipótese subjacente à esta abordagem é que a adição de informação sobre a nomenclatura dos arquivos pode melhorar o resultado do agrupamento. Entretanto, conforme se pode observar na equação 3.5.1, o peso que cada partição possui no ensemble é uma variável livre que pode ser otimizada. No caso particular abordado nesta dissertação, pode-se obter diferentes combinações de valores de pesos para a partição baseada no nome e para a partição baseada no conteúdo dos arquivos. Por isso, o peso de cada partição é variado de zero a um, com incrementos de um décimo, sendo que os pesos devem somar uma unidade, com o objetivo de investigar se há algum intervalo de pesos que melhore o resultado do agrupamento das bases de dados utilizadas nos experimentos. Assim, utilizando um ensemble com apenas duas partições, a equação 3.5.1 é redefinida para:

$$s_{ij} = w_1 \cdot a_1(i,j) + (1 - w_1) \cdot a_2(i,j) \tag{3.5.2}$$

Onde w_1 é o peso que a primeira partição do ensemble recebe no cálculo da matriz de coassociação.

3.6 ÍNDICES DE VALIDAÇÃO

A validação de um agrupamento de dados se refere ao procedimento que avalia os resultados de um agrupamento de forma quantitativa e objetiva. Há índices de validação para estruturas hierárquicas, particionais, grupos individuais, dentre outros, porém aqui serão abordados apenas índices de validação de agrupamentos particionais, pois estes são o resultado final dos experimentos realizados neste trabalho. Os índices de validação podem ser classificados em externos, internos e relativos (Jain e Dubes, 1988). Os índices externos medem o quão bem um agrupamento obtido se adequa a um agrupamento de referência, já conhecido anteriormente, dependendo de informação prévia sobre as categorias dos dados, no caso de partições. Os índices internos (e.g., SEQ na equação 3.3.2) medem a qualidade de um agrupamento utilizando características internas dos próprios dados, sem o uso de informações externas. Os índices relativos medem a qualidade relativa entre dois agrupamentos diferentes, servindo para compará-los. A fim de escolher qual partição se ajusta melhor aos dados, a avaliação experimental realizada neste trabalho se baseia em

índices externos e relativos, os quais serão brevemente revisados nas seções 3.6.1 e 3.6.2, respectivamente.

3.6.1 Índices de Validação Externos

Os índices externos, conforme já exposto, medem o grau de correspondência entre o agrupamento obtido e o agrupamento de referência, conhecido previamente. Segundo Hubert e Arabie (1985), os índices externos podem ser definidos a partir de uma tabela de contingência. Seja a partição obtida por um algoritmo $\boldsymbol{C} = \{\boldsymbol{C}_1, \boldsymbol{C}_2, \dots, \boldsymbol{C}_K\}$ e seja a partição de referência dos dados $\boldsymbol{R} = \{\boldsymbol{R}_1, \boldsymbol{R}_2, \dots, \boldsymbol{R}_R\}$. A tabela de contingência obtida a partir dessas duas partições está representada na Figura 3.6.1. Nessa tabela, n_{ij} representa o número de objetos comuns aos grupos $\boldsymbol{C}_i$ e $\boldsymbol{R}_j$, n_i^C o número de objetos presentes no grupo $\boldsymbol{C}_i$ e n_j^R o número de objetos presentes no grupo $\boldsymbol{R}_j$.

$$\begin{array}{c} \\ \boldsymbol{C}_1 \\ \boldsymbol{C}_2 \\ \vdots \\ \boldsymbol{C}_K \\ \\ \end{array} \begin{array}{c} \begin{array}{cccc} \boldsymbol{R}_1 & \boldsymbol{R}_2 & \cdots & \boldsymbol{R}_R \end{array} \\ \begin{bmatrix} n_{11} & n_{12} & \cdots & n_{1R} \\ n_{21} & n_{22} & \cdots & n_{2R} \\ \vdots & \vdots & \ddots & \vdots \\ n_{K1} & n_{K2} & \cdots & n_{KR} \end{bmatrix} \\ \begin{array}{cccc} n_1^R & n_2^R & \cdots & n_R^R \end{array} \end{array} \begin{array}{c} \\ n_1^C \\ n_2^C \\ \vdots \\ n_K^C \\ N \end{array}$$

Figura 3.6.1.1 - Tabela de contingência

Utilizando essa tabela, podem ser definidas as seguintes variáveis:

- *a*: quantidade de pares de objetos pertencentes ao mesmo grupo em $\boldsymbol{C}$ e à mesma categoria em $\boldsymbol{R}$:

$$a = \frac{1}{2}\left[\sum_{i=1}^{K}\sum_{j=1}^{R} n_{ij}^2 - N\right] \quad (3.6.1.1)$$

- *b*: quantidade de pares de objetos pertencentes ao mesmo grupo em $\boldsymbol{C}$ e a categorias diferentes em $\boldsymbol{R}$:

$$b = \frac{1}{2}\left[\sum_{j=1}^{R}\left(n_j^R\right)^2 - \sum_{i=1}^{K}\sum_{j=1}^{R} n_{ij}^2\right] \quad (3.6.1.2)$$

- *c*: quantidade de pares de objetos pertencentes a grupos diferentes em $\boldsymbol{C}$ e à mesma categoria em $\boldsymbol{R}$:

$$c = \frac{1}{2}\left[\sum_{i=1}^{K}\left(n_i^C\right)^2 - \sum_{i=1}^{K}\sum_{j=1}^{R} n_{ij}^2\right] \quad (3.6.1.3)$$

- d: quantidade de pares de objetos pertencentes a grupos diferentes em $\boldsymbol{C}$ e a categorias diferentes em $\boldsymbol{R}$:

$$d = \binom{N}{2} - a - b - c = \frac{N(N+1)}{2} - \frac{1}{2}\left[\sum_{i=1}^{K}\left(n_i^C\right)^2 + \sum_{j=1}^{R}\left(n_j^R\right)^2\right] \quad (3.6.1.4)$$

Logo, os valores de *a* e *d* expressam uma concordância entre a partição obtida e a partição de referência, enquanto que os valores *b* e *c* podem ser considerados discordâncias. Assim, define-se o *Rand Index* (RI) (Rand, 1971) pela equação:

$$RI = \frac{a+d}{a+b+c+d} \quad (3.6.1.5)$$

Entretanto, um dos problemas desse índice externo é o fato de ele não ser corrigido para aleatoriedade, isto é, o seu valor esperado não é zero ao comparar partições aleatórias. Assim, Hubert e Arabie (1985) derivaram o *Ajusted Rand Index* (ARI), ajustado para corrigir tal situação:

$$ARI = \left(a - \frac{(a+c)(a+b)}{a+b+c+d}\right) / \left(\frac{(a+c)+(a+b)}{2} - \frac{(a+c)(a+b)}{a+b+c+d}\right) \quad (3.6.1.6)$$

Outra interpretação possível das variáveis é o valor de *a* ser considerado como a quantidade de acertos, *b* e *c* como a quantidade de erros e *d* como um valor neutro, podendo ser descartado. Além disso, o valor de *d* tende a dominar as outras variáveis, principalmente quando o número de grupos é grande. Para minimizar esse problema, o *Jaccard Coefficient* (JC) (Jaccard, 1908) é definido como:

$$JC = \frac{a}{a+b+c} \quad (3.6.1.7)$$

Note que o valor desses três índices externos, RI, ARI e JC, está compreendido no intervalo [0,1], sendo obtido o valor zero quando as duas partições são completamente discordantes e sendo obtido o valor 1 quando as duas partições são exatamente iguais.

3.6.2 Índices de Validação Relativos

Índices de validação relativos comparam dois agrupamentos diferentes segundo algum critério objetivo. Eles são muito utilizados para estimar o número de grupos nos dados (Milligan e Cooper, 1985) – especialmente para algoritmos de agrupamento particionais. Para isso, um método muito comum é executar o algoritmo de agrupamento diversas vezes, para valores progressivamente maiores de *K,* sendo escolhido o valor de *K* que corresponde ao particionamento com maior (alternativamente menor para alguns índices) valor do índice de validação relativo utilizado. Esse método de estimativa do número de grupos também foi utilizado neste estudo. Existem diversos índices de validade relativos reportados na literatura, sendo alguns dos mais populares o *Variance Ratio Criterion*

(VRC, Calinski e Harabasz, 1974), o Davies-Bouldin (1979) e a Silhueta (Rousseeuw, 1987). Um estudo recente com experimentos comparativos realizados com 40 índices relativos, em 1.080 bases de dados, utilizando 962.928 partições, sugere que a silhueta fornece resultados muito bons (Vendramin et al., 2010). Por isso, este trabalho utiliza nos experimentos a silhueta para estimar o número de grupos nos dados. Assim como vários outros índices relativos, a Silhueta se baseia em conceitos geométricos de compactação e separação entre os grupos, conforme descrito a seguir. Seja a_i a distância média de um objeto $\boldsymbol{x}_i$ a todos os outros objetos pertencentes ao seu grupo $\boldsymbol{C}_p$. Seja d_i^q a distância média do objeto $\boldsymbol{x}_i$ a todos os objetos do grupo $\boldsymbol{C}_q, q \neq p$. E seja $b_i = \min(d_i^q)$, $q \in \{1, \dots, K\}, q \neq p$, ou seja, b_i é a distância média de $\boldsymbol{x}_i$ a todos os objetos do grupo vizinho mais próximo. Então, a silhueta individual do objeto $\boldsymbol{x}_i$ é definida pela equação:

$$s_i = (b_i - a_i)/\max(a_i, b_i) \tag{3.6.2.1}$$

A partir da definição acima, está claro que $1 \leq s_i \leq 1$. Quanto mais próximo s_i estiver do valor 1, melhor terá sido a classificação do objeto $\boldsymbol{x}_i$ ao seu respectivo grupo $\boldsymbol{C}_p$. Quanto mais próximo s_i estiver do valor -1, melhor teria sido a classificação do objeto $\boldsymbol{x}_i$ ao seu grupo vizinho mais próximo. Um valor de s_i próximo de zero indica que o objeto $\boldsymbol{x}_i$ está na borda do seu grupo com o grupo vizinho mais próximo. Além disso, o valor de s_i é arbitrariamente definido como zero caso o objeto $\boldsymbol{x}_i$ seja um *singleton*, isto é, caso seja o único objeto do seu grupo (Kaufman e Rousseeuw, 1990). Definir s_i como zero para os *singletons* impede que a silhueta, definida por

$$SWC = \frac{1}{N}\sum_{i=1}^{N} s_i \tag{3.6.2.2}$$

eleja a solução trivial *K* = *N* como o número de grupos mais adequado para o conjunto de dados. Para calcular a silhueta, é necessário computar as distâncias entre todos os pares de objetos. Assim, ela tem complexidade quadrática em relação ao número de objetos. Para superar tal limitação computacional em grandes bases de dados, pode ser utilizada uma versão da silhueta simplificada (Vendramin et al., 2010), em que as distâncias médias do objeto $\boldsymbol{x}_i$ ao seu grupo, a_i, e ao grupo vizinho mais próximo, b_i, podem ser redefinidas como distâncias ao centroide do seu grupo e ao centroide do grupo vizinho mais próximo, respectivamente. Dessa forma, a complexidade passa a ser linear com o número de objetos e a eficácia da silhueta simplificada em estimar o número de grupos continua competitiva com a eficácia da silhueta tradicional, conforme experimentos de Vendramin et al. (2010). Nesse sentido, a silhueta simplificada foi também utilizada nesta dissertação para estimar o número de grupos com os algoritmos de agrupamento de complexidade linear em relação

ao número de objetos, como o K-medias.

Vale observar que o método de estimativa do número de grupos com índices relativos pela avaliação de partições com diferentes valores de K, anteriormente descrito, foi adaptado para utilização com o algoritmo CSPA de consenso entre partições baseadas na nomenclatura dos arquivos e partições baseadas no conteúdo dos arquivos. Caso o maior valor de silhueta tenha sido obtido por mais de uma partição, o método de estimativa do número de grupos normalmente escolhe a partição com menor valor de K. Por exemplo, seja a partição $\boldsymbol{R} = \{\{\boldsymbol{a}, \boldsymbol{b}\},\{\boldsymbol{c}\},\{\boldsymbol{d}\}\}$ com peso 1 no algoritmo de consenso, composta por 4 objetos distribuídos em 3 grupos. Na matriz de coassociação, as distâncias entre esses objetos serão $d(\boldsymbol{a}, \boldsymbol{b}) = 0$ e $d(\boldsymbol{a}, \boldsymbol{c}) = d(\boldsymbol{a}, \boldsymbol{d}) = d(\boldsymbol{b}, \boldsymbol{c}) = d(\boldsymbol{b}, \boldsymbol{d}) = d(\boldsymbol{c}, \boldsymbol{d}) = 1$. Ainda, sejam as partições $\boldsymbol{P} = \{\{\boldsymbol{a}, \boldsymbol{b}\},\{\boldsymbol{c}, \boldsymbol{d}\}\}$ e $\boldsymbol{Q} = \{\{\boldsymbol{a}, \boldsymbol{b}\},\{\boldsymbol{c}\},\{\boldsymbol{d}\}\}$, ambas obtidas pelo algoritmo de agrupamento para $K = 2$ e $K = 3$, respectivamente. A silhueta atribuirá o valor 0,5 para ambas as partições $\boldsymbol{P}$ e $\boldsymbol{Q}$ obtidas e, como normalmente é escolhida a partição com menor valor de K, será escolhida a partição $\boldsymbol{P}$, que tem 2 grupos. Em outras palavras, a silhueta tende a escolher partições em que os *singletons* tenham sigo agrupados. Entretanto, a partição escolhida $\boldsymbol{P}$ é diferente da partição $\boldsymbol{R}$ original com peso 1 no agrupamento de consenso, gerando uma inconsistência, pois é esperado que o particionamento final de consenso seja igual à partição com peso 1. Assim, no caso do algoritmo de consenso entre nomenclatura e conteúdo dos arquivos, o método de estimativa do número de grupos foi adaptado para escolher arbitrariamente partições com maiores valores de K - caso o valor máximo de silhueta tenha sido obtido por mais de uma partição - o que corrige a inconsistência exemplificada.

Finalmente, como as bases de dados utilizadas nos experimentos possuem grande quantidade de *singletons* (*outliers*), em alguns algoritmos foi adotada uma estratégia mais explícita para identificar e separar esses objetos. Essa estratégia consiste em, após a execução do algoritmo de agrupamento para vários valores de K, remover os *singletons* presentes na partição final escolhida pela silhueta e executar novamente o algoritmo de agrupamento com os objetos restantes, para vários valores de K. Posteriormente, caso a silhueta escolha uma nova partição que ainda contenha *singletons*, estes são removidos e o procedimento é repetido recursivamente até ser obtida uma partição sem a presença de *outliers*. Ao final, os *outliers* identificados e removidos pela aplicação recursiva da silhueta

são novamente adicionados à partição final, também como *singletons*. Dessa forma, esse método privilegia a identificação e separação de *singletons*, o que pode apresentar resultados interessantes em bases de dados com grande quantidade de grupos unitários, conforme será investigado no capítulo experimental.

4 AVALIAÇÃO EXPERIMENTAL

O método de avaliação experimental utilizado neste trabalho se baseia nos índices de validação externos ARI (Hubert e Arabie, 1985) e JC (Jaccard, 1908), revisados na seção 3.6.1, os quais comparam os resultados obtidos pelos algoritmos de agrupamento particionais com partições de referência. Do ponto de vista científico, o uso de partições de referência para avaliar algoritmos de agrupamento de dados é considerado o procedimento mais principiado. Nesse caso, as partições de referência são normalmente obtidas a partir de dados gerados sinteticamente, de acordo com alguma distribuição de probabilidades. Do ponto de vista prático, tais partições de referência são usualmente empregadas para se escolher um determinado algoritmo de agrupamento que seja mais apropriado para determinada aplicação, ou para calibrar seus parâmetros. Nesse caso, uma partição de referência é construída por um especialista de domínio e reflete as expectativas que ele tem sobre os grupos que deveriam ser encontrados numa determinada amostra da base.

Este capítulo está organizado da seguinte forma: a seção 4.1 descreve as bases de dados utilizadas nos experimentos, a seção 4.2 indica os aplicativos utilizados e as implementações realizadas para viabilizar a avaliação experimental, a seção 4.3 resume os algoritmos avaliados e seus respectivos parâmetros de inicialização e a seção 4.4 apresenta os resultados dos experimentos e as discussões realizadas.

4.1 BASES DE DADOS

Nos experimentos deste estudo, foram utilizadas cinco bases de dados obtidas a partir de casos reais de investigação diversificados. Como os exames periciais normalmente são realizados separadamente para cada disco rígido, cada base de dados foi obtida a partir de um disco rígido diferente. Para cada um dos cinco discos rígidos, foram selecionados todos os documentos com extensão “doc”, “docx” e “odt”. Em seguida, foram excluídos os documentos duplicados em cada base, com código de integridade *Message Digest 5* (MD5 - Rivest, 1992) repetido. Posteriormente, o texto plano dos documentos foi extraído para que pudessem ser processados pela ferramenta de mineração de textos.

Os índices externos de validação ARI e JC, utilizados na avaliação dos resultados,

pressupõem a existência de uma partição de referência para cada base de dados. Entretanto, tais partições de referência usualmente não estão disponíveis em casos reais, as quais foram então construídas manualmente por um especialista de domínio, através da inspeção individual do conteúdo de cada documento. As bases de dados contêm quantidades diversificadas de documentos (*N*), número de grupos (*K*), números de atributos (*D*), número de grupos unitários ou *singletons* (*S*) e número de documentos por grupo (#), conforme apresentado na Tabela 4.1.1.

Tabela 4.1.1 - Características das bases de dados utilizadas

Base	*N*	*K*	*D*	*S*	# Maior Grupo
A	37	23	1744	12	3
B	111	49	7894	28	12
C	68	40	2699	24	8
D	74	38	5095	26	17
E	131	51	4861	31	44

4.2 FERRAMENTAS UTILIZADAS

Para viabilizar a realização dos experimentos deste trabalho foi necessária a utilização de duas ferramentas diferentes, bem como realizadas algumas implementações. A primeira ferramenta foi utilizada durante a etapa de construção das cinco bases de dados, para extrair o conteúdo textual dos arquivos de extensão “doc”, “docx” e “odt”, contidos nos cinco discos rígidos selecionados. Para isso, foi implementado um *script* de extração de texto utilizando o aplicativo pericial EnCase Forensic, versão 6.18.0.59. O segundo aplicativo utilizado foi o *Rapid Miner 5* (licença AGPL 3), que fornece um pacote de mineração de dados implementado em Java, contendo diversos operadores para utilização nas várias etapas do processo de mineração, como pré-processamento, redução de atributos, extração de padrões e validação. Foram utilizadas suas implementações base dos algoritmos K-medias, K-medoides, *Average Link*, *Single Link* e *Complete Link*. Entretanto, por não estarem disponíveis nesse pacote, foi necessário implementar o algoritmo de consenso entre partições CSPA (Strehl e Ghosh, 2002), a inicialização de protótipos em objetos distantes de Mirkin (2005), a distância de Levenshtein normalizada entre sequencias de caracteres, os índices de validação relativos entre partições silhueta e

silhueta simplificada, o processo de remoção de *outliers* baseado na aplicação recursiva da silhueta e os índices de validação externos Rand ajustado e Jaccard. Além disso, como foram coletados dados de tempo de execução dos algoritmos, é necessário descrever o *hardware* utilizado. Os experimentos foram executados em um computador com processador Intel Core 2 Duo P9400 2.4 Ghz, 4 Gigabytes de memória RAM e disco rígido SATA Seagate, modelo ST9250410ASG.

4.3 ALGORITMOS E PARÂMETROS

Conforme exposto anteriormente, este trabalho tem como principal objetivo uma comparação entre algoritmos de agrupamento de dados aplicados à Informática Forense. Foram avaliados algoritmos particionais, hierárquicos e de consenso, com diferentes parâmetros de inicialização. Também foram avaliadas técnicas de estimativa automática do número de grupos. A seguir são enumerados os algoritmos avaliados e respectivas variações de parâmetros:

a) K-médias, a partir do conteúdo, com os 100 atributos de maior variância da TF e inicializações aleatórias (Kms100);
b) K-médias, a partir do conteúdo, com todos os atributos e inicializações aleatórias (Kms);
c) K-médias, a partir do conteúdo, com os 100 atributos de maior variância da TF, inicializações aleatórias e aplicação recursiva da silhueta simplificada (Kms100S);
d) K-médias, a partir do conteúdo, com todos os atributos, inicializações aleatórias e aplicação recursiva da silhueta simplificada (KmsS);
e) K-médias, a partir do conteúdo, com os 100 atributos de maior variância da TF e inicializações em objetos distantes (Kms100*);
f) K-médias, a partir do conteúdo, com os 100 atributos de maior variância da TF, inicializações em objetos distantes e silhueta tradicional (KmsT100*);
g) K-medoides, a partir do conteúdo, com os 100 atributos de maior variância da TF e inicializações aleatórias (Kmd100);
h) K-medoides, a partir do conteúdo, com os 100 atributos de maior variância da TF e inicializações em objetos distantes (Kmd100*);
i) K-medoides, utilizando a distância de *Levenshtein* normalizada entre os

nomes dos documentos (KmdLev);

j) K-medoides, utilizando a distância de *Levenshtein* normalizada entre os nomes dos documentos e aplicação recursiva da silhueta (KmdLevS);
k) *Average Link*, a partir do conteúdo, com os 100 atributos de maior variância da TF (AL100);
l) *Complete Link*, a partir do conteúdo, com os 100 atributos de maior variância da TF (CL100);
m) *Single Link*, a partir do conteúdo, com os 100 atributos de maior variância da TF (SL100);
n) Consenso entre algoritmos KmdLev e Kms100, com variação do peso de cada um no cálculo da matriz de coassociação (NC100);
o) Consenso entre algoritmos KmdLev e Kms, com variação do peso de cada um no cálculo da matriz de coassociação (NC);
p) Consenso entre 100 partições geradas pelo K-médias, a partir do conteúdo, selecionando 100 atributos aleatórios para cada partição (E100);
q) Consenso entre partições geradas pelo K-médias, a partir do conteúdo, selecionando conjuntos disjuntos de 100 atributos aleatórios para cada partição (E100Disj);
r) Consenso entre 10 partições geradas pelo K-médias, a partir do conteúdo, selecionando conjuntos disjuntos de 10 atributos aleatórios a partir dos 100 atributos de maior variância da TF (E10Disj);

Na Tabela 4.3.1 é apresentado um resumo dos algoritmos e seus respectivos parâmetros de inicialização e instanciação, onde "Sil. Simp." e "Sil. Rec." são abreviaturas para Silhueta Simplificada e Silhueta Recursiva, respectivamente.

Tabela 4.3.1 - Resumo dos algoritmos e respectivos parâmetros

Sigla	Algoritmo	Atributos	Medida de Distância	Inicialização	Estimativa de *K*
Kms	K-médias	Conteúdo (todos)	Cosseno	Aleatória	Sil. Simp.
Kms100	K-médias	Conteúdo (100 > TV)	Cosseno	Aleatória	Sil. Simp.
Kms100*	K-médias	Conteúdo (100 > TV)	Cosseno	Mirkin	Sil. Simp.
KmsT100*	K-médias	Conteúdo (100 > TV)	Cosseno	Mirkin	Silhueta
KmsS	K-médias	Conteúdo (todos)	Cosseno	Aleatória	Sil. Rec.
Kms100S	K-médias	Conteúdo (100 > TV)	Cosseno	Aleatória	Sil. Rec.

Kmd100	K-medoides	Conteúdo (100 > TV)	Cosseno	Aleatória	Silhueta
Kmd100*	K-medoides	Conteúdo (100 > TV)	Cosseno	Mirkin	Silhueta
KmdLev	K-medoides	Nome	Levenshtein	Aleatória	Silhueta
KmdLevS	K-medoides	Nome	Levenshtein	Aleatória	Sil. Rec.
AL100	*AverageLink*	Conteúdo (100 > TV)	Cosseno	-	Silhueta
CL100	*CompleteLink*	Conteúdo (100 > TV)	Cosseno	-	Silhueta
SL100	*SingleLink*	Conteúdo (100 > TV)	Cosseno	-	Silhueta
NC	CSPA	Nome e Conteúdo (todos)	Coassociação	Aleatória	Sil. Simp.
NC100	CSPA	Nome e Conteúdo (100 > TV)	Coassociação	Aleatória	Sil. Simp.
E100	CSPA	Conteúdo (100 aleatórios)	Coassociação	Aleatória	Sil. Simp.
E100Disj	CSPA	Conteúdo (100 aleatórios disjuntos)	Coassociação	Aleatória	Sil. Simp.
E10Disj	CSPA	Conteúdo (10 aleatórios disjuntos dentre os 100 > TV)	Coassociação	Aleatória	Sil. Simp.

A seguir são apresentadas algumas observações importantes sobre os algoritmos:

- para os algoritmos de consenso entre nome e conteúdo, *K* foi estimado utilizando a silhueta tradicional, eventualmente priorizando partições com maior número de grupos, conforme descrito na seção 3.6.2;
- o algoritmo K-medoides foi executado com 100 inicializações diferentes para cada valor de *K (*para tentar minimizar a convergência para mínimos locais);
- o algoritmo K-medias foi executado apenas com 10 inicializações diferentes para cada valor de *K*, devido ao alto valor da constante de tempo ocasionado pelo cômputo de distâncias e de centroides. Caso fossem executadas 100 inicializações, como no K-medoides, os experimentos poderiam se tornar inviáveis devido ao maior tempo de execução, principalmente dos algoritmos com todos os atributos e de consenso entre partições;
- o algoritmo KmsT100*, apesar de ser linear com o número de objetos, utilizou a silhueta tradicional para estimar o número de grupos, para permitir uma comparação com os outros algoritmos que utilizaram a silhueta tradicional;
- para obter a partição resultante dos algoritmos de consenso, foi utilizado o K-medoides a partir da matriz de coassociação, também com 100

inicializações aleatórias;

- alguns algoritmos utilizaram a silhueta recursivamente para remover *outliers* de uma forma mais agressiva, conforme descrito na seção 3.6.2, sendo adicionada a letra "S" ao final da sigla desses algoritmos;
- o algoritmo de consenso entre partições E100Disj utilizou o maior número possível de partições geradas a partir de subconjuntos disjuntos de 100 atributos do conteúdo dos arquivos. Por exemplo, na base A, com 1744 atributos, foram geradas 17 partições e, na base B, com 7894 atributos, foram geradas 78 partições, a partir de subconjuntos disjuntos de 100 atributos.

4.4 RESULTADOS

De um modo geral, o algoritmo hierárquico *Average Link,* utilizando os 100 atributos de maior variância (AL100) e com estimativa do número de grupos pela silhueta tradicional, apresentou os melhores resultados, tanto em relação a uma maior média dos índices de validação externos ARI e JC nas cinco bases de dados, quanto em relação a uma maior consistência e estabilidade, devido ao um menor valor de desvio padrão de ARI e JC dentre todos algoritmos, conforme pode ser observado nas Tabelas 4.4.1 e 4.4.2 – nessas tabelas, foram reportados os resultados dos algoritmos de consenso entre nomenclatura e conteúdo dos arquivos correspondentes às partições cujo peso de conteúdo resultou nos maiores valores dos índices externos de validação. O algoritmo *Complete Link* (CL100), também hierárquico, apresentou desempenho muito similar ao *Average Link*, com alta média de ARI e JC e pequeno desvio padrão desses mesmos índices de validação. Entretanto, o algoritmo hierárquico *Single Link* (SL100) obteve um desempenho ruim relativamente aos outros dois algoritmos hierárquicos, principalmente nas bases A e B, apresentando menor média dos índices de validação e certa instabilidade, com um maior desvio padrão. Isso pode ter sido causado pela sensibilidade do *Single Link* à presença de *outliers,* que podem provocar um encadeamento de objetos durante o agrupamento, misturando grupos diferentes (Everitt et al, 2001).

Tabela 4.4.1 - ARI para todas as bases, média e desvio padrão

Algoritmo	Base A	Base B	Base C	Base D	Base E	Média	Desvio
AL100	0,94	0,83	0,89	0,99	0,90	0,91	0,06
CL100	0,94	0,76	0,89	0,98	0,90	0,89	0,08
KmsT100*	0,81	0,76	0,89	0,97	0,94	0,88	0,09
Kmd100*	0,81	0,76	0,89	0,96	0,93	0,87	0,08
SL100	0,54	0,63	0,90	0,98	0,88	0,79	0,19
NC100	0,66	0,64	0,78	0,74	0,72	0,71	0,06
Kms	0,61	0,60	0,69	0,79	0,84	0,71	0,11
NC	0,61	0,60	0,69	0,79	0,84	0,71	0,11
Kms100*	0,53	0,63	0,63	0,68	0,93	0,68	0,15
Kmd100	0,81	0,58	0,72	0,25	0,79	0,63	0,23
Kms100	0,64	0,64	0,78	0,29	0,72	0,62	0,19
KmsS	0,47	0,11	0,75	0,80	0,82	0,59	0,30
Kms100S	0,60	0,54	0,74	0,20	0,69	0,55	0,21
E10Disj	0,60	0,57	0,52	0,24	0,20	0,43	0,19
E100Disj	0,43	0,08	0,37	0,71	0,47	0,41	0,22
E100	0,61	0,10	0,29	0,76	0,08	0,37	0,31
KmdLevS	0,62	0,23	0,37	0,55	0,05	0,36	0,23
KmdLev	0,46	0,16	0,32	0,74	0,08	0,35	0,26

Tabela 4.4.2 - JC para todas as bases, média e desvio padrão

Algoritmo	Base A	Base B	Base C	Base D	Base E	Media	Desvio
AL100	0,88	0,72	0,81	0,97	0,83	0,84	0,09
CL100	0,88	0,62	0,81	0,97	0,83	0,82	0,13
KmsT100*	0,68	0,63	0,81	0,95	0,90	0,79	0,14
Kmd100*	0,68	0,63	0,81	0,94	0,89	0,79	0,13
SL100	0,38	0,47	0,82	0,97	0,81	0,69	0,25
Kms	0,45	0,44	0,54	0,67	0,76	0,57	0,14
NC	0,45	0,44	0,54	0,67	0,76	0,57	0,14
NC100	0,50	0,49	0,65	0,62	0,61	0,57	0,07
Kms100*	0,38	0,48	0,47	0,55	0,88	0,55	0,20
Kmd100	0,68	0,42	0,57	0,16	0,69	0,5	0,22
KmsS	0,32	0,09	0,61	0,69	0,73	0,49	0,28
Kms100	0,48	0,49	0,65	0,19	0,61	0,48	0,18
Kms100S	0,44	0,38	0,59	0,12	0,57	0,42	0,19
E10Disj	0,44	0,41	0,36	0,15	0,22	0,32	0,13
E100Disj	0,29	0,07	0,24	0,58	0,39	0,31	0,19
E100	0,45	0,08	0,19	0,64	0,05	0,28	0,25
KmdLev	0,31	0,10	0,20	0,62	0,05	0,25	0,23
KmdLevS	0,45	0,14	0,23	0,41	0,03	0,25	0,18

Merecem destaque os desempenhos dos algoritmos particionais K-medias (KmsT100*) e K-medoides (Kmd100*), utilizando os 100 atributos de maior variância, a silhueta tradicional para estimar o número de grupos e a inicialização de Mirkin (2005) em objetos distantes. Os desempenhos desses algoritmos foram comparáveis aos desempenhos dos melhores algoritmos hierárquicos, AL100 e CL100, apresentando médias altas de ARI e JC e baixos desvios padrão, relativamente. Vale observar que a complexidade computacional do K-medias é linear com o número de objetos, enquanto que a complexidade do K-medoides e dos algoritmos hierárquicos é no mínimo quadrática com o número de objetos. Além disso, ressalta-se que os desempenhos dos algoritmos Kms100* e Kmd100*, com inicialização em objetos distantes, foi bem melhor que suas as respectivas versões com inicialização em objetos aleatórios, Kms100 e Kmd100. Isso indica que a estratégia de inicialização de protótipos em objetos distantes entre si de Mirkin (2005) favorece a diminuição do problema dos mínimos locais do K-medias e do K-medoides, ao mesmo tempo em que contribui para a separação de *outliers*. Para ilustrar melhor como a inicialização dos protótipos influencia os resultados, foram gerados gráficos dos valores de Silhueta em função de *K* para os algoritmos Kms100, Kms100* e AL100, nas cinco bases, apresentados na Figura 4.4.1. Observa-se, na figura, que o algoritmo Kms100, com inicialização aleatória dos centroides, oscila muito, gerando instabilidade de resultados, e apresenta maior quantidade de máximos locais de Silhueta. Opostamente, o algoritmo Kms100*, com a inicialização em objetos distantes de Mirkin (2005) é mais estável e apresenta menor quantidade de máximos locais. Surpreendentemente, o algoritmo Kms100* apresenta uma curva muito parecida com a curva do algoritmo hierárquico AL100, principalmente para maiores valores de *K*. Esse fato pode ser justificado, em parte, porque ambos os algoritmos privilegiam a separação de *outliers*, favorecendo resultados similares. Assim, caso o número de grupos seja conhecido *a* priori, o algoritmo Kms100* (que é computacionalmente mais eficiente) pode ser utilizado para alcançar resultados similares aos que seriam obtidos pelo algoritmo AL100.

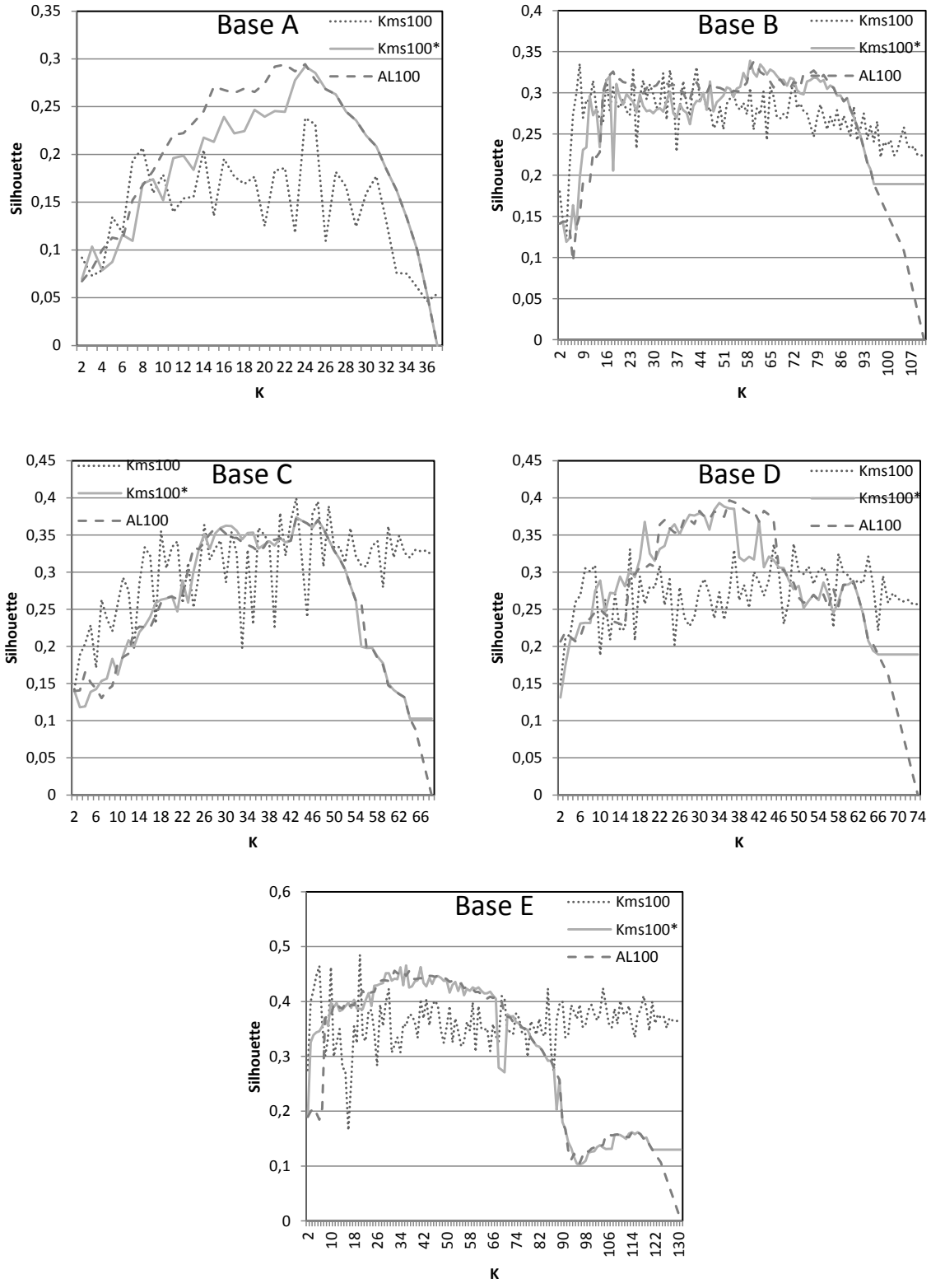

Figura 4.4.1 - Silhuetas dos algoritmos Kms100, Kms100* e AL100

Além disso, também se observa que o algoritmo das K-médias com a inicialização de Mirkin (2005) em objetos distantes e utilizando a silhueta simplificada para estimar o

número de grupos, Kms100*, obteve desempenho pouco melhor que a sua versão com inicialização em objetos aleatórios, Kms100, tanto na média dos índices de validação externos como no desvio padrão. Entretanto, o algoritmo Kms100*, o qual utiliza a silhueta simplificada para estimar *K*, obteve desempenho bastante inferior relativamente aos algoritmos KmsT100*, Kmd100* e aos algoritmos hierárquicos, que utilizam a silhueta tradicional para estimar o número de grupos. Essa observação é interessante e sugere que a silhueta simplificada está fornecendo uma estimativa de *K* de pior qualidade em relação à silhueta tradicional. Isso é justificável, pois, fundamentalmente, há perda de informação durante a estimativa do número de grupos por parte da silhueta simplificada, a qual é uma aproximação da tradicional, pois utiliza distâncias aos centroides dos grupos ao invés de distâncias médias aos objetos dos grupos. Conforme pode ser observado nas tabelas das próximas seções, em quatro das cinco bases de dados, a silhueta simplificada forneceu valores de *K* bem menores que a silhueta tradicional e que a partição de referência. Em uma análise mais detalhada, note que a distância média de um determinado objeto a todos os objetos de um grupo tende a ser maior do que a distância desse mesmo objeto ao centroide do grupo. Além disso, essa diferença entre distância média aos objetos e distância ao centroide tende a ser maior quando é considerado o grupo do próprio objeto e tende a ser menor quando considerados outros grupos. Por isso, quando utilizadas distâncias a centroides ao invés de distâncias médias na silhueta simplificada, o valor de *a* na Equação 3.6.2.1 tende a diminuir mais do que *b*, fazendo com que a diferença $b - a$ seja maior na silhueta simplificada. Além disso, quanto menores os grupos ou, equivalentemente, quanto maior a quantidade de grupos, a diferença entre a distância média aos objetos de um grupo e a distância ao centroide do grupo tende a diminuir, se igualando no caso de grupos unitários – *singletons*. Logo, na silhueta simplificada, a diferença $b - a$ tende a diminuir com o aumento do número de grupos, aproximando-se do valor da silhueta tradicional e se igualando no caso extremo $K=N$, quando cada grupo é formado por um único objeto, conforme pode ser observado na Figura 4.4.2. Em todos os gráficos, o valor de silhueta simplificada é maior no início e diminui progressivamente até se igualar ao valor de silhueta. Assim, na silhueta simplificada, a diferença $b - a$ tende a ser maior para valores mais baixos de *K*, favorecendo partições com menor número de grupos.

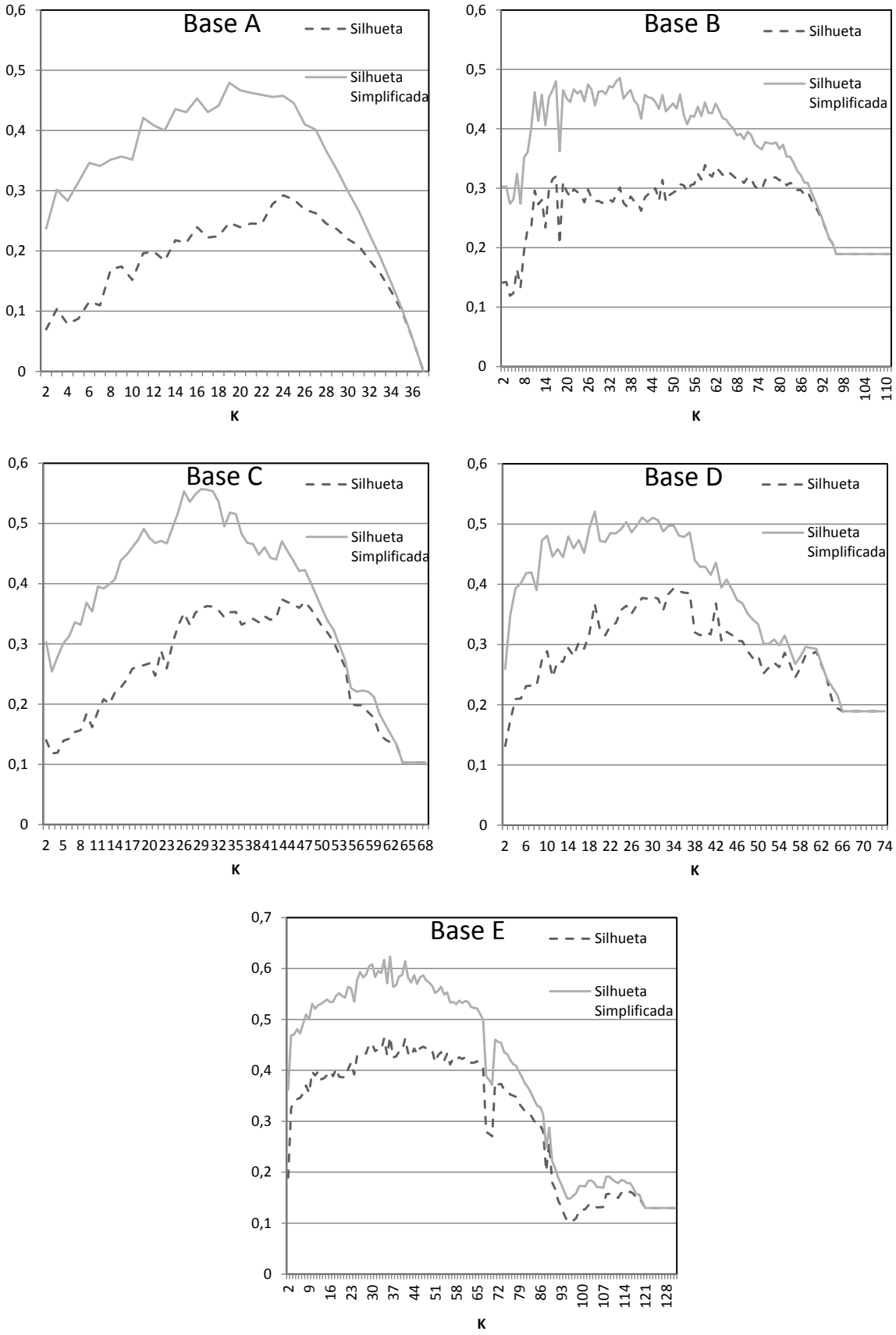

Figura 4.4.2 - Silhueta e Silhueta Simplificada do algoritmo Kms100*

Outra conclusão interessante é que a utilização apenas da nomenclatura como medida de similaridade não trouxe bons resultados no geral, de acordo com o desempenho médio dos algoritmos KmdLev e KmdLevS, os quais foram os piores de acordo com as Tabelas 4.4.1 e 4.4.2. Isso é compreensível, já que o nome dos arquivos provê menos informações sobre os arquivos do que o seu próprio conteúdo. Entretanto, uma exceção a esse comportamento geral foi o desempenho relativamente bom dos algoritmos KmdLev e KmdLevS na base de dados D, sendo melhor até que alguns algoritmos baseados no conteúdo dos arquivos, conforme pode ser observado nas Tabelas 4.4.1 e 4.4.2. Esse fato reforça a hipótese de que as informações de nomenclatura dos arquivos, apesar de mais pobres do que aquelas obtidas via conteúdo, podem ser úteis no processo de agrupamento se utilizadas conjuntamente com o conteúdo dos arquivos. Nesse sentido, a Figura 4.4.3 apresenta gráficos de ARI dos algoritmos de consenso entre partições baseadas na nomenclatura e no conteúdo dos arquivos, em função do peso da partição baseada no conteúdo no cálculo da matriz de coassociação. Nessa figura, os pesos da partição de conteúdo foram variados arbitrariamente em intervalos de 0,1 e as linhas foram traçadas continuamente para facilitar a visualização da tendência do algoritmo. Um resultado interessante que pode ser observado é a ocorrência de picos de ARI do algoritmo NC100 - de consenso entre nomenclatura e conteúdo dos arquivos com seleção dos 100 atributos de maior variância da TF - na base A, quando o peso da partição de conteúdo foi de 0,6 e 0,7, e na base C, quando o peso da partição de conteúdo foi de 0,6. Apesar de não ter acontecido com todas as bases de dados, esse resultado sugere que a mistura de informações sobre nomenclatura e conteúdo dos arquivos pode enriquecer o cálculo da similaridade entre os documentos, o que incentiva uma melhor investigação do uso da nomenclatura dos documentos em processos de agrupamento de arquivos. Nota-se também, na Figura 4.4.3, picos de ARI nas bases de dados B, C e E, quando o peso de conteúdo no agrupamento de consenso foi de uma unidade, o que significa que, nessas bases, o melhor resultado foi obtido quando utilizadas apenas informações de conteúdo dos documentos. Outro aspecto que merece comentários na Figura 4.4.3, foi o comportamento distinto dos algoritmos NC e NC100 na base de dados D. Observa-se que o algoritmo NC100 tem uma queda brusca de desempenho quando o peso da partição baseada no conteúdo aumenta. Isso pode ser explicado porque a partição de conteúdo em si apresenta um valor de ARI relativamente baixo, provavelmente ocasionado pela convergência do K-medias para um mínimo local ruim.

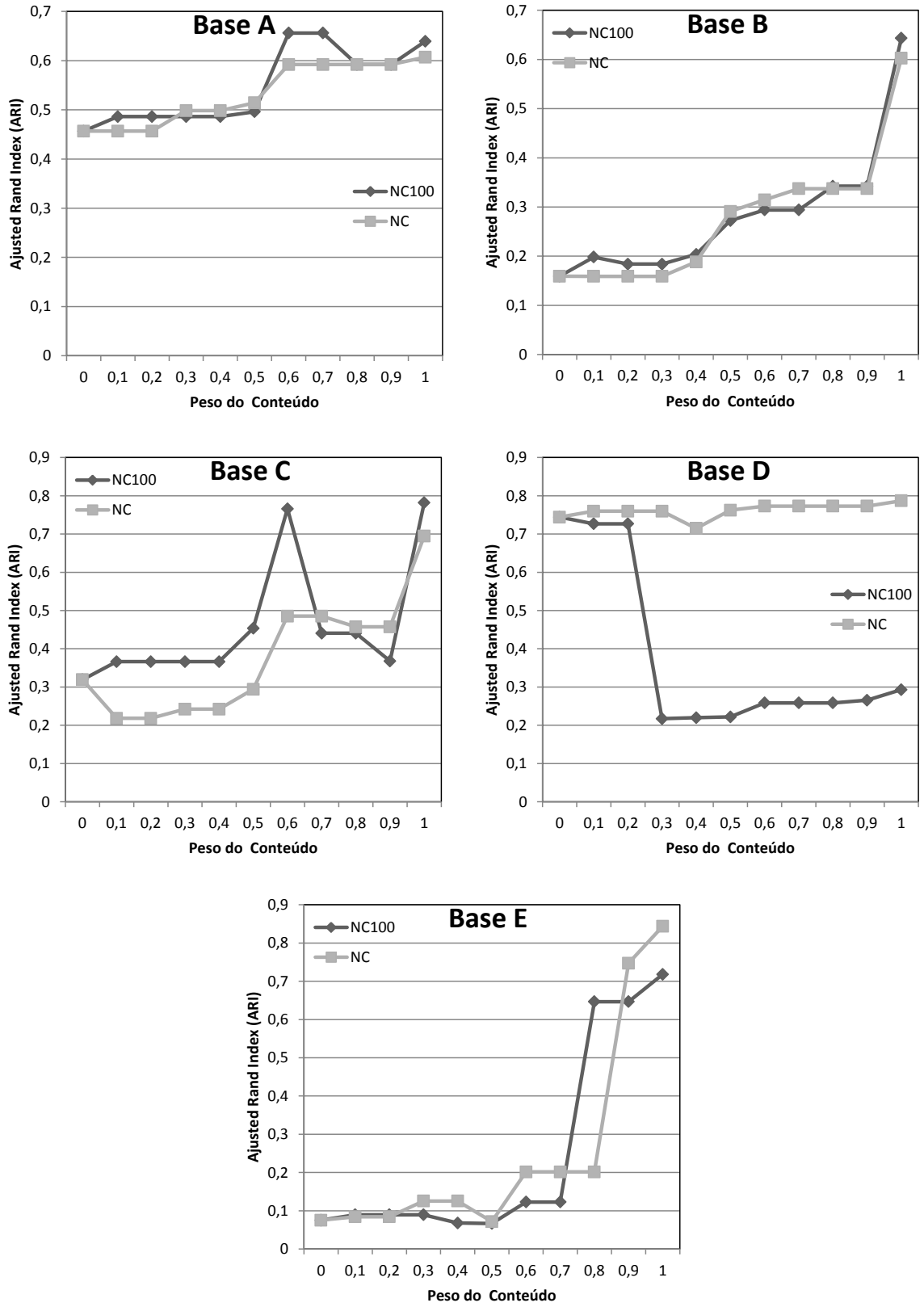

Figura 4.4.3 - ARI dos algoritmos NC e NC100

Os resultados relativamente ruins dos índices de validação externos dos algoritmos E100 e E100Disj merecem explicações, principalmente nas bases de dados com maior quantidade

de atributos, como a base B. Esses algoritmos obtém uma partição de consenso a partir de um conjunto de partições geradas por várias execuções do K-medias. No caso do E100, cada partição é gerada selecionando 100 atributos aleatórios para representar os documentos e, no caso do E100Disj, as partições são geradas a partir da seleção de subconjuntos disjuntos de 100 atributos aleatórios. Entretanto, devido à alta esparsidade da matriz atributo-valor, fato comum em bases de dados textuais, e devido à escolha aleatória dos atributos, a maioria dos documentos acaba sendo representada por vetores com quase todos os atributos nulos, sendo agrupados em um único grupo cujo centroide é um vetor nulo. Isso fornece uma falsa informação de similaridade entre esses documentos durante o cálculo da matriz de coassociação, o que acaba prejudicando o agrupamento final de consenso. Logo, a escolha de atributos aleatórios para gerar o conjunto de partições em algoritmos de agrupamento de consenso não é uma boa estratégia em bases textuais.

Outro aspecto que merece ser abordado é o desempenho dos algoritmos com a aplicação recursiva da silhueta para remover *outliers* de uma forma mais contundente, conforme procedimento detalhado na seção 3.6.2. De acordo com as Tabelas 4.4.1 e 4.4.2, os algoritmos KmsS e Kms100S apresentaram valores médios dos índices de validação externos relativamente inferiores às respectivas versões desses algoritmos sem aplicação recursiva da silhueta, Kms e Kms100, respectivamente. Já o algoritmo KmdLevS, baseado na similaridade entre as nomenclaturas dos documentos, apresentou desempenho médio dos índices externos similar à sua versão sem aplicação recursiva da silhueta, KmdLev. Assim, os resultados nas bases de dados utilizadas nos experimentos sugerem que a estratégia de remoção de *outliers* pela execução recursiva da silhueta não é adequada para problemas que envolvam agrupamento de bases de dados textuais na Informática Forense. Ainda de acordo com as Tabelas 4.4.1 e 4.4.2, em três das cinco bases de dados utilizadas nos experimentos, os algoritmos com seleção dos 100 atributos de maior variância da TF obtiveram melhores resultados que os algoritmos que utilizaram todos os atributos para representar os documentos. Esse resultado é interessante, pois sugere que a seleção de atributos com maior variância mantém a eficácia dos algoritmos de agrupamento comparável às suas respectivas versões com o uso de todos os atributos. Ao mesmo tempo, a eficiência computacional dos algoritmos é melhorada com a diminuição do tempo de execução dos mesmos, pois sua eficiência computacional normalmente é dependente do número de atributos.

Merece ser destacado que os algoritmos hierárquicos, além de terem apresentado bons resultados na avaliação via índices externos de validação, em geral, apresentam uma característica desejável, pois proporcionam uma visualização final do conjunto de dados intuitiva, na forma de um dendrograma. Essa representação visual provê uma descrição muito informativa da estrutura dos dados, podendo ser utilizada na visualização de dados de alta dimensionalidade, como os documentos textuais analisados durante os exames periciais, sendo útil para a descoberta de padrões durante a análise exploratória dos dados. A título de ilustração, nas Figuras 4.4.4 a 4.4.8 são apresentados os dendrogramas obtidos pelo algoritmo AL100, que obteve os melhores resultados em praticamente todas as bases de dados. Os dendrogramas foram cortados horizontalmente na altura correspondente ao número de grupos obtido pela Silhueta. É importante destacar que sub-árvores com pequena altura e grande largura representam grupos coesos – contendo documentos muito similares – e numerosos, respectivamente. Tais grupos são bons candidatos para uma análise inicial por parte do perito. Além disso, uma abordagem de análise possível por parte do perito é, após encontrar um grupo de documentos relevantes, analisar o grupo de documentos mais semelhante ao grupo encontrado, pois é possível que ele também seja importante. Isso pode ser realizado "subindo e descendo" o dendrograma, i.e., acessando o grupo cujo nó pai também seja pai do grupo de documentos relevantes encontrado anteriormente.

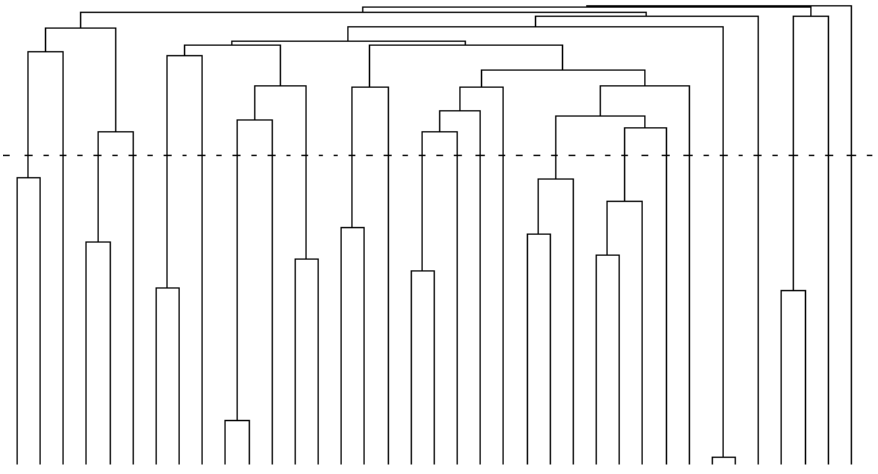

Figura 4.4.4 - Dendrograma obtido por AL100 na base A (K=23)

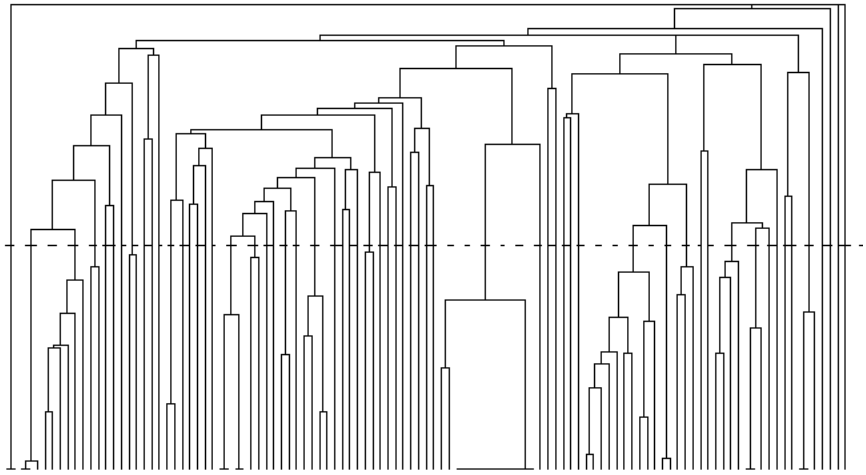

Figura 4.4.5 - Dendrograma obtido por AL100 na base B (K=49)

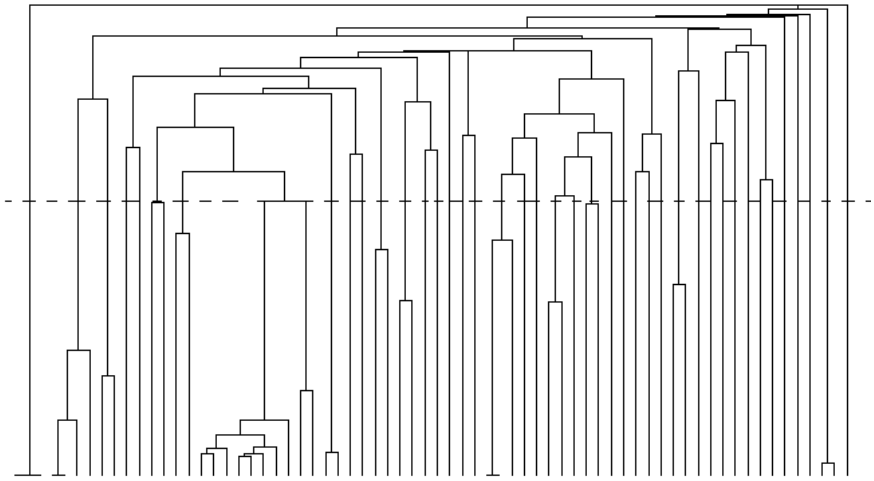

Figura 4.4.6 - Dendrograma obtido por AL100 na base C (K=40)

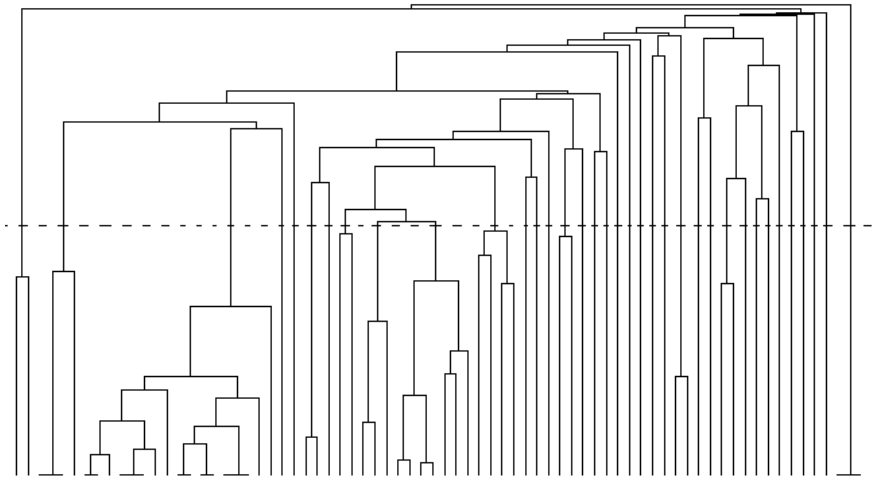

Figura 4.4.7 - Dendrograma obtido por AL100 na base D (K=38)

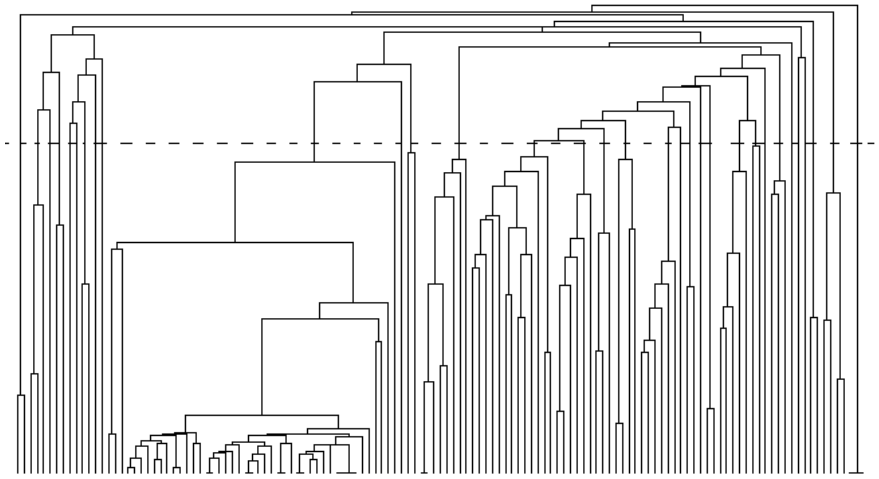

Figura 4.4.8 - Dendrograma obtido por AL100 na base E (K=51)

Finalmente, fazendo uma avaliação sob um ponto de vista prático, em todas as bases de dados utilizadas, o melhor algoritmo de agrupamento (geralmente o AL100), obteve grupos contendo ou documentos relevantes ou documentos irrelevantes no contexto de uma investigação, conforme será detalhado nas próximas seções. Dentre os relevantes, podem ser citados grupos de documentos com comprovantes de transferências bancárias internacionais, com recibos de pagamento, com boletos de operações cambiais, com relatórios diários de movimentação financeira e grupos contendo procurações. Dentre os grupos irrelevantes, podem-se citar grupos contendo documentos em branco, documentos corrompidos, modelos de etiquetas, modelos de exemplo de aplicativo, avisos de períodos de férias, dentre outros. Assim, o agrupamento de documentos se mostra eficaz em separar os documentos sob análise em grupos de documentos relevantes e grupos irrelevantes para a investigação. Nesse sentido, o perito poderia se concentrar inicialmente em analisar documentos representativos de cada grupo e, a partir dessa análise preliminar, eventualmente decidir pelo exame detalhado dos demais documentos de cada grupo em questão. Num cenário mais prático e realista, no qual especialistas de domínio (e.g., peritos criminais) são escassos e dispõem de tempo limitado, é razoável assumir que, após encontrar um documento relevante (e.g., em uma busca por palavras-chave), o perito poderia priorizar a análise dos outros documentos pertencentes ao grupo do documento encontrado, pois é provável que tais documentos também sejam relevantes para a investigação. Assim, o agrupamento de documentos se apresenta muito útil em análises periciais de computadores, ajudando os peritos a conduzir os exames de uma forma mais eficiente, focando nos documentos mais relevantes, sem precisar inspecionar todos os

documentos individualmente.

Nas próximas seções, serão apresentados alguns resultados específicos para cada base de dados, como número de grupos estimado por cada algoritmo, tempo de execução dos algoritmos e descrições de alguns grupos de documentos obtidos. Finalmente, na última seção deste capítulo, serão realizadas considerações finais sobre os resultados experimentais e apresentadas algumas recomendações práticas a partir dos resultados obtidos.

4.4.1 Base A

Na Tabela 4.4.1.1, são apresentados os valores de ARI, JC, quantidade de grupos e tempo de execução obtido por cada algoritmo. Os tempos de execução incluem o tempo de obtenção do dendrograma para os algoritmos hierárquicos e a estimativa do número de grupos pelas silhuetas. Para os algoritmos de consenso entre nomenclatura e conteúdo dos documentos, são apresentados os resultados correspondentes à partição de consenso cujo peso de conteúdo resultou nos maiores valores dos índices de validação externos.

Tabela 4.4.1.1 - Resultados para a base A (N=37 e K=23)

Algoritmo	ARI	JC	K	Tempo (h:m:s)
AL100	0,94	0,88	24	00:00:00
CL100	0,94	0,88	24	00:00:00
Kmd100	0,81	0,68	24	00:00:13
Kmd100*	0,81	0,68	24	00:00:15
KmsT100*	0,81	0,68	24	00:00:07
NC100	0,66	0,50	18	00:04:50
Kms100	0,64	0,48	19	00:00:06
KmdLevS	0,62	0,45	24	00:00:53
E100	0,61	0,45	20	00:13:49
Kms	0,61	0,45	18	00:01:10
NC	0,61	0,45	18	00:04:58
E10Disj	0,60	0,44	19	00:00:21
Kms100S	0,60	0,44	24	00:00:11
SL100	0,54	0,38	19	00:00:00
Kms100*	0,53	0,38	19	00:00:10
KmsS	0,47	0,32	25	00:06:33
KmdLev	0,46	0,31	23	00:00:38
E100Disj	0,43	0,29	14	00:06:53

Conforme se pode observar na Tabela 4.4.1.1, os algoritmos com melhor desempenho na base A foram o AL100 e CL100, ambos hierárquicos. O algoritmo SL100, também hierárquico, obteve um desempenho relativamente ruim, possivelmente por ser sensível à presença *outliers*. A partição obtida por AL100 e CL100, que foi a mesma para ambos os algoritmos, diferiu da partição de referência em apenas um documento, que foi separado do seu grupo original e formou um novo grupo unitário. Foram obtidos 13 *singletons*, constituídos por documentos relevantes e irrelevantes, como *spam* e uma solicitação de transferência bancária, por exemplo. Os outros 11 grupos (não unitários) obtidos são descritos a seguir:

G1) 02 documentos de licença e ajuda de software de foto e imagem;
G2) 02 orçamentos de manutenção de veículo;
G3) 02 boletos de Documento de Arrecadação de Receitas Federais (DARF);
G4) 02 propostas de renovação de seguros;
G5) 02 relatórios de serviço de manutenção em embarcação;
G6) 02 propostas de venda de condicionador de ar;
G7) 03 recibos de pagamento;
G8) 02 panfletos sobre fragmentadoras de papel;
G9) 02 modelos de etiquetas da empresa;
G10) 03 modelos de envelope da empresa;
G11) 02 documentos provenientes de *SPAM*;

Observa-se que, do ponto de vista pericial, foram encontrados tanto grupos potencialmente relevantes, contendo documentos como boletos de DARF (G3) e recibos de pagamentos (G7), como também grupos irrelevantes, tais como modelos de etiqueta (G9) e de envelope (G10). Isso mostra a utilidade dos algoritmos de agrupamento em separar documentos relevantes de irrelevantes em bases sobre as quais há pouca informação, auxiliando as análises do especialista de domínio.

Os algoritmos Kmd100, Kmd100* e KmsT100*, particionais, obtiveram um bom desempenho relativamente aos melhores algoritmos (AL100 e CL100). Em termos absolutos, o resultado daqueles parece ser bem inferior ao destes, mas, na prática, as partições obtidas diferem pouco entre esses algoritmos – apenas no agrupamento de dois

documentos. Entretanto, o número de objetos na base A é pequeno e, consequentemente, uma pequena diferença entre a partição obtida e a de referência causa grande diminuição de ARI e JC. Além disso, apesar de o algoritmo Kms100 ser muito similar a Kmd100, seu resultado foi bem inferior. Provavelmente, isso ocorreu devido à menor quantidade de inicializações do Kms100, dez vezes menor, o qual deve ter convergido para um mínimo local relativamente ruim da função objetivo.

4.4.2 Base B

Na Tabela 4.4.2.1, são apresentados os valores de ARI, JC, quantidade de grupos e tempo de execução obtido por cada algoritmo para a base B. Para os algoritmos de consenso entre nomenclatura e conteúdo dos documentos, são apresentados os resultados correspondentes à partição de consenso cujo peso de conteúdo resultou nos maiores valores dos índices de validação externos.

Tabela 4.4.2.1 - Resultados para a base B (N=111 e K=49)

Algoritmo	ARI	JC	K	Tempo (h:m:s)
AL100	0,83	0,72	60	00:00:03
KmsT100*	0,76	0,63	62	00:02:59
Kmd100*	0,76	0,63	62	00:04:14
CL100	0,76	0,62	63	00:00:03
Kms100	0,64	0,49	37	00:01:33
NC100	0,64	0,49	37	01:00:33
Kms100*	0,63	0,48	33	00:03:42
SL100	0,63	0,47	77	00:00:03
Kms	0,60	0,44	39	02:05:37
NC	0,60	0,44	39	02:44:11
Kmd100	0,58	0,42	30	00:04:39
E10Disj	0,57	0,41	60	00:07:19
Kms100S	0,54	0,38	71	00:06:37
KmdLevS	0,23	0,14	73	00:08:31
KmdLev	0,16	0,10	58	00:04:26
KmsS	0,11	0,09	63	14:15:26
E100	0,10	0,08	38	02:49:11
E100Disj	0,08	0,07	16	02:26:03

Na base B, o algoritmo com melhor desempenho dos índices externos de validação novamente foi o AL100, hierárquico. Posteriormente, os melhores desempenhos foram dos

algoritmos KmsT100* e Kmd100*, particionais com inicialização dos protótipos em objetos distantes, e CL100, também hierárquico. Novamente o algoritmo SL100 foi inferior aos demais algoritmos hierárquicos, possivelmente por encadeamento provocado pelos *outliers*. Na partição resultante do algoritmo AL100, foram obtidos 42 *singletons* constituídos por documentos irrelevantes do ponto de vista pericial, como letras de músicas e principalmente trabalhos escolares. Os outros grupos, não unitários, obtidos por esse algoritmo são descritos a seguir:

G1) 02 documentos corrompidos;
G2) 03 questionários respondidos sobre educação e relação professor/aluno;
G3) 02 comentários sobre entrevistas com professores de escolas públicas e particulares;
G4) 06 relatórios de estágio em educação infantil;
G5) 04 documentos vazios, com mesmo nome de autor nos metadados;
G6) 02 documentos sobre psicodrama;
G7) 02 capas de trabalhos escolares;
G8) 02 orçamentos, 01 relatório de despesas e 01 pedido de serviço de empreiteira;
G9) 04 documentos sobre mulheres no mercado de trabalho;
G10) 03 entrevistas sobre mulheres no mercado de trabalho;
G11) 01 documento com despesas diárias e 01 capa de trabalho escolar;
G12) 03 documentos sobre o Índice de Gini;
G13) 02 currículos;
G14) 10 documentos sobre comunicação organizacional;
G15) 02 listas de referências bibliográficas;
G16) 03 documentos sobre comunicação organizacional;
G17) 12 documentos vazios e 01 modelo de referência bibliográfica com mesmo autor nos metadados;
G18) 02 documentos sobre entidade assistencial para cegos;

Nota-se que, nessa base, a maioria dos documentos é relacionada a trabalhos escolares, sem relevância sob o ponto de vista de uma investigação. Essa é uma situação muito comum, pois não há conhecimento prévio sobre os dados que serão analisados, tampouco

garantia de que os discos rígidos apreendidos contêm informações importantes, o que só será descoberto após o exame pericial. Entretanto, o grupo G8 foi o único a reunir documentos de possível relevância, como orçamentos, relatórios de despesas e pedidos de serviço. Isso exemplifica mais uma vez a utilidade dos algoritmos de agrupamentos de dados em separar dados relevantes e irrelevantes, o que auxilia o perito a descartar documentos sem importância e focar naqueles que realmente têm relevância.

4.4.3 Base C

Na Tabela 4.4.3.1, são apresentados os valores de ARI, JC, quantidade de grupos e tempos de execução obtidos por cada algoritmo. Seguindo os procedimentos adotados para avaliar os resultados nas bases A e B, para os algoritmos de consenso entre nomenclatura e conteúdo dos documentos são apresentados os resultados correspondentes à partição de consenso cujo peso de conteúdo resultou nos maiores valores dos índices de validação externos.

Tabela 4.4.3.1 - Resultados para a base C (N=68 e K=40)

Algoritmo	ARI	JC	K	Tempo (h:m:s)
SL100	0,90	0,82	47	00:00:01
AL100	0,89	0,81	43	00:00:01
CL100	0,89	0,81	43	00:00:01
KmsT100*	0,89	0,81	43	00:00:40
Kmd100*	0,89	0,81	43	00:00:56
Kms100	0,78	0,65	28	00:00:21
NC100	0,78	0,65	28	00:12:23
KmsS	0,75	0,61	40	00:28:00
Kms100S	0,74	0,59	35	00:01:08
Kmd100	0,72	0,57	30	00:00:55
Kms	0,69	0,54	32	00:09:23
NC	0,69	0,54	32	00:20:38
Kms100*	0,63	0,47	28	00:00:50
E10Disj	0,52	0,36	24	00:02:33
E100Disj	0,37	0,24	21	00:15:35
KmdLevS	0,37	0,23	43	00:01:42
KmdLev	0,32	0,20	38	00:00:54
E100	0,29	0,19	19	00:24:22

Na base C, o algoritmo com melhor desempenho foi o SL100, hierárquico, seguido pelos algoritmos AL100 e CL100, também hierárquicos, e KmdT100* e Kmd100*, particionais

com inicialização em objetos distantes. Como a partição obtida pelos algoritmos AL100, CL100, KmsT100* e Kmd100*, idêntica para todos eles, obteve valores dos índices de validação externos praticamente iguais à partição do algoritmo SL100 e como o número de grupos daquela partição foi mais próximo ao da partição de referência, a seguir serão descritos os grupos da partição obtida pelos algoritmos AL100, CL100, KmsT100* e Kmd100*. Nessa partição, foram obtidos 28 grupos *singletons*, constituídos por documentos relevantes e irrelevantes, como relatórios diários de movimentação financeira e receitas de alimentos, respectivamente. Os 15 grupos não unitários podem ser descritos da seguinte forma:

G1) 03 documentos em branco;
G2) 04 permissões para operação bancária (saque e apresentação de cheque);
G3) 02 documentos sobre salário maternidade;
G4) 02 relações de alimentos com respectivas quantidades;
G5) 01 aviso sobre aperfeiçoamento das operações cambiais e 01 lista de documentos para atualização de dados cadastrais;
G6) 02 documentos de revisão de boletas de operações cambiais;
G7) 01 modelo de ficha cadastral de corretora de valores e 01 modelo de contrato com a corretora;
G8) 01 estatuto de clube de investimento e 01 termo de adesão ao clube;
G9) 02 modelos de registro de movimentação em espécie maior ou igual a R$100.000,00;
G10) 08 boletos (comprovantes) de operações cambiais de compra ou venda de valores;
G11) 02 avisos da corretora de câmbio sobre horário de funcionamento;
G12) 03 modelos de etiquetas da corretora de câmbio;
G13) 01 aviso sobre horário de expediente e 01 recibo de cheque;
G14) 02 relatórios de movimentos diários de compra e venda de câmbio;
G15) 02 documentos de exemplo de aplicativo de escritório;

Nessa base, novamente se observa a presença de grupos de documentos relevantes do ponto de vista pericial, como de boletos de operações cambiais (G10), relatórios de movimentações cambiais diárias (G14) e permissões para operações bancárias (G2).

Também há grupos irrelevantes no contexto de uma investigação, como modelos de etiquetas (G12) e avisos sobre horário de expediente (G13). Também se percebe a formações de grupos relativamente ruins, reunindo documentos tematicamente diferentes, como o grupo G13, o qual contém um aviso sobre horário de expediente e um recibo de cheque, o que exemplifica que o resultado da execução de um algoritmo de agrupamento nem sempre provê grupos coesos, ilustrando a dificuldade de encontrar soluções globais para problemas de agrupamento de dados.

4.4.4 Base D

A Tabela 4.4.4.1 sumariza os resultados obtidos para a base de dados D. O algoritmo com melhor desempenho dos índices externos novamente foi o AL100, o qual obteve valores excepcionais de ARI=0,99 e JC=0,97. Os outros algoritmos hierárquicos, SL100 e CL100, também obtiveram desempenhos muito similares. Em seguida, os algoritmos particionais KmsT100* e Kmd100*, com inicializações em objetos distantes, também obtiveram desempenhos muito bons relativamente aos algoritmos hierárquicos. A partição obtida pelo

Tabela 4.4.4.1 - Resultados para a base D (N=74 e K=38)

Algoritmo	ARI	JC	K	Tempo (h:m:s)
AL100	0,99	0,97	36	00:00:02
SL100	0,98	0,97	41	00:00:02
CL100	0,98	0,97	35	00:00:02
KmsT100*	0,97	0,95	34	00:00:54
Kmd100*	0,96	0,94	32	00:01:20
KmsS	0,80	0,69	21	01:18:42
Kms	0,79	0,67	16	00:22:29
NC	0,79	0,67	16	00:37:42
E100	0,76	0,64	20	00:57:47
KmdLev	0,74	0,62	25	00:01:12
E100Disj	0,71	0,58	16	00:34:52
Kms100*	0,68	0,55	21	00:01:08
KmdLevS	0,55	0,41	30	00:02:32
Kms100	0,29	0,19	36	00:00:33
NC100	0,74	0,62	25	00:16:38
Kmd100	0,25	0,16	43	00:01:21
E10Disj	0,24	0,15	48	00:02:38
Kms100S	0,20	0,12	53	00:02:00

algoritmo AL100 diferiu da partição de referência apenas nos grupos G7 e G12, descritos

adiante, cada um sendo uma união de dois grupos diferentes da partição de referência. Esse algoritmo obteve 24 *singletons*, constituídos por documentos relevantes e irrelevantes como, por exemplo, um recibo de pagamento e documentos corrompidos, respectivamente. Os 12 grupos não unitários obtidos pelo algoritmo AL100 são apresentados a seguir:

G1) 02 documentos de aviso de aplicativo de escritório;
G2) 04 planos de chamada de horários de empresa de segurança;
G3) 02 documentos corrompidos;
G4) 02 solicitações de serviço de empresa de apoio a eventos;
G5) 17 cronogramas de convocação para evento de segurança com o timbre "confidencial";
G6) 03 documentos em branco;
G7) 01 declaração de realização de serviço de segurança e 01 formulário de aceitação de proposta de serviço;
G8) 02 procurações;
G9) 02 relatórios de ocorrências;
G10) 03 documentos de confirmação de prestação de serviço de segurança em evento;
G11) 07 propostas de prestação de serviço de segurança;
G12) 02 planejamentos estratégicos operacionais e 02 relatórios de acompanhamento de evento;

Novamente foram obtidos grupos aparentemente relevantes do ponto de vista pericial, contendo documentos classificados como confidenciais (G5), procurações (G8), dentre outros. Também foram encontrados grupos sem relevância no contexto de um exame pericial típico, contendo documentos em branco (G6), corrompidos (G3), exemplos de aviso de aplicativo (G1), etc.

4.4.5 Base E

A Tabela 4.4.5.1 resume os resultados obtidos para a base de dados E. Diferentemente das demais, os algoritmos que obtiveram os melhores desempenhos dos índices externos de validação foram os algoritmos particionais com inicialização dos protótipos em objetos distantes: KmsT100*, Kmd100* e Kms100*. Em seguida, os algoritmos hierárquicos

AL100, CL100 e SL100 obtiveram desempenhos relativamente muito bons. O algoritmo com melhor desempenho nessa base, KmsT100*, obteve 12 grupos *singletons*, todos formados por documentos irrelevantes, como mensagens de *spam* e trabalhos de curso de inglês, por exemplo. Os 22 grupos não unitários obtidos por esse algoritmo são descritos a seguir:

Tabela 4.4.5.1 - Resultados para a base E (N=131 e K=51)

Algoritmo	ARI	JC	K	Tempo (h:m:s)
KmsT100*	0,94	0,90	34	00:04:53
Kmd100*	0,93	0,89	34	00:06:56
Kms100*	0,93	0,88	36	00:06:01
CL100	0,90	0,83	30	00:00:03
AL100	0,90	0,83	32	00:00:03
SL100	0,88	0,81	56	00:00:03
Kms	0,84	0,76	20	01:55:47
NC	0,84	0,76	20	03:33:30
KmsS	0,82	0,73	34	12:31:36
Kmd100	0,79	0,69	16	00:08:14
Kms100	0,72	0,61	20	00:02:43
NC100	0,72	0,61	20	01:49:14
Kms100S	0,69	0,57	21	00:10:02
E100Disj	0,47	0,39	8	02:30:16
E10Disj	0,20	0,22	2	00:12:14
E100	0,08	0,05	84	05:00:50
KmdLev	0,08	0,05	71	00:07:51
KmdLevS	0,05	0,03	103	00:14:25

G1) 02 documentos de exemplo de aplicativo de escritório;

G2) 03 documentos em branco;

G3) 03 currículos;

G4) 04 comprovantes de transferências bancárias internacionais;

G5) 02 grades de disciplinas de curso MBA;

G6) 01 documento com artigos de lei relacionada ao INSS e 01 piada na forma de aviso proveniente do Judiciário;

G7) 44 contratos de locação de imóvel;

G8) 07 documentos com aulas de inglês e 01 documento com dados de proprietários de imóvel;

G9) 07 cartas (sendo 05 para o mesmo destinatário) e 01 receita de alimento;

G10) 01 documento sobre aplicação financeira, 01 com dados e valores de bens

para imposto de renda e 01 em branco;

G11) 06 procurações e 01 descrição de serviço de marcenaria;

G12) 02 documentos com fórmulas de medicamentos;

G13) 02 solicitações de correntista para o banco Banespa;

G14) 03 documentos com aulas de inglês;

G15) 05 documentos com aulas de inglês, 02 recibos, 01 relação de convidados e 01 pedido de cancelamento de contrato;

G16) 01 modelo de vistoria em imóvel e 01 relação de dados de imóveis;

G17) 02 cartas para o mesmo destinatário sobre seu benefício do INSS;

G18) 02 aulas de inglês sobre preposições;

G19) 05 documentos com aulas de inglês;

G20) 02 documentos em inglês sobre vocabulário similar;

G21) 02 formulários de declaração final de espólio;

G22) 02 documentos com dados e valor de reserva de voo Frankfurt – Brasília.

Conforme descrição acima, novamente foram encontrados grupos interessantes do ponto de vista de um exame pericial, como um grupo contendo muitos contratos de locação (G7), o que sugere que o investigado pode estar utilizando imóveis para lavar dinheiro e outro grupo contendo comprovantes de transferências bancárias internacionais (G4), dentre outros. Também foram obtidos grupos irrelevantes, contendo documentos em branco (G2), documentos de exemplos de aplicativo (G1), etc. Isso mostra mais uma vez a potencialidade dos algoritmos de agrupamento de dados em auxiliar análises exploratórias de dados em exames periciais.

4.5 CONSIDERAÇÕES FINAIS

Neste capítulo, foram apresentados os resultados e as discussões dos experimentos realizados. A avaliação experimental foi baseada na utilização dos índices externos de validação ARI (Hubert e Arabie, 1985) e JC (Jaccard, 1908), os quais refletem a correspondência entre as partições obtidas pelos algoritmos de agrupamento e as partições de referência dos dados, conhecidas previamente. Foram utilizadas cinco bases de dados diferentes nos experimentos. Tais bases de dados foram obtidas por meio da extração de documentos de texto contidos em discos rígidos de computadores apreendidos em

investigações reais. Dentre os algoritmos avaliados estão algoritmos particionais, hierárquicos e de consenso entre partições, com utilização de diferentes parâmetros, totalizando 18 instanciações diferentes dos algoritmos. Também foram avaliados métodos automáticos de estimativa do número de grupos dos dados, que é um parâmetro crítico de vários algoritmos e normalmente desconhecido *a priori*.

De um modo geral, os algoritmos hierárquicos *Average Link* e *Complete Link*, utilizando os 100 atributos de maior variância da TF e estimando o número de grupos com a silhueta, obtiveram os maiores valores dos índices de validação, bem como uma maior estabilidade dos resultados (menor desvio padrão dos índices de validação). Apesar de esses algoritmos apresentarem complexidade computacional no mínimo quadrática com o número de objetos da base, suas constantes de tempo de execução são bastante pequenas. Por isso, esses algoritmos apresentaram os menores tempos de execução nas bases de dados avaliadas, da ordem de poucos segundos, sendo indicados para aplicações práticas em bases de dados de tamanhos similares. Além disso, a propriedade de separação de *outliers* desses algoritmos se mostrou bastante adequada para utilização nas bases de dados utilizadas. Adicionalmente, o resultado desses algoritmos fornece uma visualização hierárquica dos grupos de objetos (dendrograma) muito útil para a exploração dos padrões, principalmente quando o número de grupos não é conhecido previamente.

Além disso, os algoritmos particionais K-medias e K-medoides, utilizando os 100 atributos de maior variância da TF, estimando o número de grupos com a silhueta e utilizando a inicialização em objetos distantes de Mirkin (2005), apresentaram desempenhos tão bons quanto os melhores algoritmos hierárquicos. Entretanto, esses algoritmos particionais apresentaram constantes de tempo muito elevadas, principalmente o K-medias, devido a muitos cálculos de distâncias e de centroides num espaço de grande dimensionalidade. Apesar disso, o algoritmo K-medias possui complexidade computacional linear em relação ao número de objetos, sendo indicado para aplicações práticas em bases de dados com grande número de objetos, como conjuntos de arquivos de e-mail ou de páginas de Internet, os quais normalmente são encontrados em quantidades entre milhares e dezenas de milhares de arquivos em exames periciais. A utilização da inicialização em objetos distantes de Mirkin (2005) foi fundamental para tornar o desempenho do algoritmo K-medias comparável ao desempenho dos algoritmos hierárquicos, pois proporcionou uma

separação de *outliers* similar à destes algoritmos. Além disso, a estimativa do número de grupos do K-medias pela silhueta simplificada não proporcionou bons resultados, na medida em que ela induz quantidades menores de grupos que a silhueta, enquanto as bases de dados avaliadas apresentam grandes quantidades de grupos.

5 CONCLUSÃO

5.1 PRINCIPAIS CONTRIBUIÇÕES

A principal contribuição deste trabalho foi uma comparação objetiva de diversos algoritmos de agrupamento de dados aplicados à Computação Forense, mais especificamente em bases de dados textuais. Foram avaliados os algoritmos particionais K-medias (MacQueen, 1967) e K-medoides (Kaufman e Rousseeuw, 1990), os hierárquicos *Single Link* (Florek et al., 1951), *Complete Link* (Sorensen, 1948) e *Average Link* (Sokal e Michener, 1958), e o algoritmo de consenso entre partições CSPA (Strehl e Ghosh, 2002). Foram realizadas algumas variações de parâmetros, totalizando 18 instanciações diferentes dos algoritmos. Além disso, no presente trabalho também foram comparadas duas técnicas para estimar o número de grupos automaticamente, a Silhueta (Rousseeuw, 1987) e a Silhueta Simplificada (Hruschka et al., 2006), dado que o número de grupos é um parâmetro crítico de vários algoritmos e é normalmente desconhecido *a priori*. Até onde foi possível investigar, não foram reportados na literatura sobre Computação Forense estudos sobre algoritmos de agrupamento hierárquicos e de consenso, nem sobre técnicas para a estimativa automática do número de grupos, o que foi realizado neste estudo.

Nos experimentos realizados, de modo geral, os algoritmos hierárquicos *Average Link* e *Complete Link* apresentaram os melhores resultados, obtendo os maiores valores dos índices de validação externos utilizados, bem como uma maior estabilidade de resultados. Apesar de possuírem complexidade computacional quadrática com relação ao número de objetos, apresentaram constante de tempo de execução bastante pequenas, sendo indicados para bases de dados pequenas, de poucas centenas de objetos. Além disso, os algoritmos particionais K-medias e K-medoides foram menos afetados pelo problema da convergência para mínimos locais com a utilização da inicialização de protótipos em objetos distantes de Mirkin (2005), apresentando resultados tão bons quanto os melhores algoritmos hierárquicos. Apesar de apresentar uma constante de tempo de execução elevada devido a muitos cálculos de distância entre objetos num espaço de alta dimensionalidade, o algoritmo K-medias é bastante escalável por possuir complexidade computacional linear em relação ao número de objetos, sendo indicado para grandes bases de dados.

Além disso, alguns resultados sugerem que a utilização da nomenclatura dos documentos em conjunto com informações de conteúdo, mediante o uso de algoritmos de consenso entre partições, pode enriquecer o processo de agrupamento de arquivos. Em relação aos índices de validação relativos utilizados para estimar o número de grupos, a Silhueta apresentou melhor desempenho que a Silhueta Simplificada. Esse resultado pode ser justificado porque a Silhueta Simplificada fundamentalmente utiliza menos informação, além de usualmente induzir uma menor quantidade de grupos que a Silhueta, o que foi prejudicial nas bases de dados utilizadas, que apresentam grande quantidade de grupos. Entretanto, a Silhueta Simplificada, por ser mais eficiente computacionalmente que a Silhueta, pode se mostrar mais adequada para utilização em bases de dados mais numerosas.

Finalmente, foi possível observar nos experimentos a capacidade dos algoritmos de agrupamento de identificar e separar grupos de documentos relevantes de grupos de documentos irrelevantes, sob o ponto de vista pericial. Assim, foi evidenciado como os algoritmos de agrupamento podem contribuir com a descoberta de conhecimentos novos e úteis durante os exames periciais, que são essencialmente análises exploratórias de dados. Ao mesmo tempo, os algoritmos de agrupamento têm o potencial de tornar o processo pericial mais eficiente, na medida em que facilitam a localização de informações importantes sem que seja necessário examinar individualmente todos os documentos sob análise.

Algumas dessas contribuições foram apresentadas em artigos aceitos para publicação em congressos científicos das áreas de Inteligências Artificial e Computação Forense, nos trabalhos de Nassif e Hruschka (2011a, 2011b, 2011c).

5.2 LIMITAÇÕES

O estudo realizado neste trabalho possui algumas limitações, principalmente decorrentes do escopo definido inicialmente para a pesquisa. Uma das limitações se refere à utilização de apenas documentos de texto com extensões “doc”, “docx” e “odt” nas bases de dados utilizadas nos experimentos. Usualmente, são encontrados e analisados diversos outros tipos de arquivos com conteúdo textual em exames periciais de computadores, como

mensagens eletrônicas (*e-mails*), documentos de hipertexto de Internet, planilhas eletrônicas, *logs* de aplicativos, arquivos do tipo "pdf", históricos de conversas instantâneas, dentre outros. Entretanto, a inclusão de muitos tipos de arquivos, principalmente mensagens eletrônicas e documentos de hipertexto, que normalmente estão presentes em quantidades da ordem de dezenas de milhares, poderia inviabilizar os experimentos, pois, devido ao método de avaliação dos algoritmos pela utilização de índices externos de validação, seria necessário construir manualmente partições de referência para grandes quantidades de arquivos através da inspeção individual dos seus conteúdos, já que não se dispõe de tais partições normalmente.

Outra limitação deste trabalho é referente à utilização de um algoritmo de *stemming* apenas para palavras da língua portuguesa, durante a etapa de pré-processamento dos documentos. Embora o idioma da maioria dos documentos textuais presentes nas bases de dados seja o Português, há alguns documentos escritos na língua inglesa e alemã. É provável que a não utilização de algoritmos de *stemming* específicos para palavras nesses idiomas tenha diminuído o desempenho dos algoritmos. O fato de documentos em idiomas diversos coexistirem num mesmo computador sob análise é muito comum de ocorrer durante os exames periciais. Por exemplo, manuais de programas e documentos de hipertexto de Internet escritos em Inglês são muito encontrados em computadores de usuários cuja língua nativa seja o Português. O tratamento dessa diversidade de idiomas dos documentos é importante e deve ser abordada em uma aplicação prática, mas foge ao escopo e objetivo deste trabalho.

5.3 TRABALHOS FUTUROS

Como trabalhos futuros, podem ser abordadas as limitações presentes neste trabalho. Por exemplo, podem ser realizados experimentos com bases de dados constituídas por outros tipos de arquivos, como mensagens de correio eletrônico ou documentos de hipertexto da Internet, para avaliar se os algoritmos apresentam o mesmo desempenho relativo com bases de dados compostas por tipos de arquivos diferentes. Outro aspecto que pode ser abordado é a realização de novos experimentos com a incorporação de mais algoritmos de *stemming*, aplicados de acordo com o idioma utilizado em cada documento, o que permitiria investigar se o desempenho dos algoritmos de agrupamento é sensível ao

processo de *stemming* realizado durante a etapa de pré-processamento dos textos.

Também merecem ser estudados métodos para rotular os grupos obtidos. A atribuição de rótulos aos grupos é interessante, pois permitiria ao perito, especialista no domínio da aplicação, identificar com maior rapidez as ideias e os assuntos presentes em cada grupo, mesmo antes de analisar o conteúdo dos documentos pertencentes ao *cluster.* Além disso, a atribuição de rótulos aos grupos é uma etapa importante para a construção de classificadores, que podem ser interessantes no contexto de uma grande investigação com vasta quantidade de computadores apreendidos, especialmente se os grupos obtidos nas primeiras análises demonstrarem ser de grande utilidade.

Outro trabalho futuro promissor se refere à avaliação de algoritmos que obtenham partições sobrepostas ou *fuzzy,* como o *Fuzzy C-Means* e *Expectation-Maximization* para modelos baseados em misturas de distribuições de probabilidades gaussianas. Muitas vezes, não se pode atribuir cada documento da coleção a apenas um grupo, pois ele pode abordar diversos assuntos diferentes, o que pode prejudicar o desempenho dos algoritmos de partições rígidas. Nesses casos, é melhor atribuir a cada documento probabilidades de pertinência a cada grupo, o que é obtido pelos algoritmos de agrupamento de partições sobrepostas.

Finalmente, seria interessante avaliar outros índices de validade relativos para estimar o número de grupos para algoritmos com complexidade computacional linear em relação ao número de objetos, especialmente índices cuja complexidade também seja linear, como o *Variance Ratio Criterion* (VRC, Calinski e Harabasz, 1974), já que o desempenho da Silhueta Simplificada apresentou um desempenho relativamente inferior na estimativa do número de grupos para o K-medias.

REFERÊNCIAS

Agrawal, R., Imielinski, T., Swami, A. (1993). *Mining Association Rules* Between Sets of Items in Large Databases, SIGMOD Conference 1993, pp. 207-216.

Baeza-Yates, R., Ribeiro-Neto, B. (1999). Modern Information Retrieval. Addison Wesley e ACM Press.

Beebe, N. L., Clark, J. G. (2007). Digital forensic text string searching: Improving information retrieval effectiveness by thematically clustering search results. Digital Investigation, Volume 4, Supplement 1, pp. 49 a 54. Ed. Elsevier.

Bellman, R. (1961). Adaptive Controll Processes. A Guided Tour. Princeton University Press.

Beyer, K., Goldstein, J., Ramakrishnan, R. e Shaft., U. (1999). When is "nearest neighbor" meaningful? In Proceedings of the 7th International Conference on Database Theory (ICDT), Jerusalem, Israel.

Bishop, C. M. (2006). Pattern Recognition and Machine Learning, Capítulo 1, pp. 3. Editora Springer.

Calinski, R. B., Harabasz, J. (1974). A dentrite method for cluster analysis. Comm. In statistics, 3: pp. 1-27.

Davies, D. L., Bouldin, D. W. (1979). A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1: pp.224-227.

Decherchi, S., Tacconi, S., Redi, J., Leoncini, A., Sangiacomo, F., Zunino, R. (2009). Text Clustering for Digital Forensics Analysis. Computational Intelligence in Security for Information Systems, Advances in Soft Computing, Vol. 63, pp. 29-36. Ed. Springer.

Deerwester, S., et al. (1988). Improving Information Retrieval with Latent Semantic Indexing, Proceedings of the 51st Annual Meeting of the American Society for Information Science 25, pp. 36-40.

Duda, R., Hart, P., e Stork, D. (2001). Pattern Classification, 2nd edition. John Wiley & Sons. New York, NY.

Ebecken, N. F. F., Lopes, M. C. S., e Costa, M. C. A. (2003). Mineração de textos. Em Rezende, S. O., editor, Sistemas Inteligentes: Fundamentos e Aplicações, capítulo 13, pp. 337-364. Manole, 1 edição.

Everitt, B. S., Landau, S., e Leese, M. (2001). Cluster Analysis. Arnold, Quarta edição.

Fayyad, U., Piatetsky-Shapiro, G., Amith, S., Smyth, P. (1996). From data mining to

knowledge discovery: an overview. Press, A., editor, Advances in Knowlegdge Discovery and Data Mining, pp. 1-34.

Fei, B.K.L., Eloff, J.H.P., Venter, H.S. e Oliver, M.S. (2005). Exploring Forensic Data with Self-Organizing Maps. IFIP International Conference on Digital Forensics 2005, Orlando, Florida. Advances in Digital Forensics, pp. 113 a 123. Ed. Springer.

Florek, K., Lukaskzewicz, J., Perkal, J. e Zubrycki, S. (1951). Sur la liason et la division des points d'un ensemble fini. Colloquium Mathematicae, 2, pp.282-285.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classication. The Journal of Machine Learning Research, 3: pp. 1289-1305.

Fred, A. L. N., e Jain, A. K. (2005). Combining Multiple Clusterings Using Evidence Accumulation. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, n. 6., pp. 835-850.

Gantz, J. F., Reinsel, D., Chute, C., Schlichting, W., McArthur, J., Minton, S., Xheneti, I., Toncheva, A., e Manfrediz, A. (2007). The expanding digital universe: A forecast of worldwide information growth through 2010. External Publication of IDC (Analyse the Future) Information and Data, p.p. 1 a 21.

Hadjidj R., Debbabi, M., Lounis, H., Iqbal, F., Szporer, A., Benredjem, D. (2009). Towards an integrated e-mail forensic analysis framework. Digital Investigation, Volume 5, Issues 3-4, pp. 124 a 137. Ed. Elsevier.

Hinneburg, A., Aggarwal,C. C., e Keim, D. A. (2000). *What is the nearest neighbor in high dimensional spaces*? In Proceedings of the 26th International Conference on Very Large DataBases (VLDB).

Hruschka, E. R., Campello, R. J. G. B., e de Castro, L. N. (2006). *Evolving clusters in gene-expression data*. Information Sciences, 176:1898-1927.

Hubert, L., e Arabie, P. (1985). *Comparing partitions. Journal of Classification*, v. 2, pp. 193-218.

Iqbal, F., Binsalleeh, H., Fung, B. C. M., Debbabi, M. (2010). Mining writeprints from anonymous e-mails for forensic investigation. Digital Investigation, Volume 7, Issues 1-2, pp. 56 a 64. Ed. Elsevier.

Jaccard, P. (1908). *Nouvelles recherches sur la distribution florale. Bulletin de la Société Vaudoise de Sciences Naturelles*, 44: pp. 223-370.

Jain, A. K., Dubes, R. C. (1988). Algorithms for Clustering Data. Editora Prentice-Hall.

Jolliffe, I. T. (2002). Principal Component Analysis. Springer, Segunda edição.

Kaufman, L, e Rousseeuw, P. (1990). *Finding Groups in Data: An introduction to cluster analysis*. Wiley-Interscience.

Kriegel, H.P., Kroger, P., e Zimek, A. (2009). *Clustering High-Dimensional Data: A Survey on Subspace Clustering, Pattern-Based Clustering and Correlation Clustering*. ACM Transactions on Knowledge Discovery from Data, Vol. 3, No. 1.

Kullback, S., Leibler, R.A. (1951). On Information and Sufficiency. *Annals of Mathematical Statistics* 22 (1): pp. 79–86.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady* 10: pp. 707–10.

Liu, C. L. (1968). Introduction to Combinatorial Mathematics (Computer Science Series). Mcgraw-Hill College.

Liu, H. e Yu, L. (2005). *Toward integrating feature selection algorithms for classication and clustering*. IEEE Transactions on Knowledge and Data Engineering, 17(4): pp. 491-502.

Liu, L., Kang, J., Yu, J., e Wang, Z. (2005). A comparative study on unsupervised feature selection methods for text clustering. Em NLP-KE'05: Proceedings of 2005 IEEE International Conference on Natural Language Processing and Knowledge Engineering, pp. 597-601.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press. pp. 281–297

Mirkin, B. (2005). Clustering for data mining: A data recovery approach. London: Chapman and Hall.

Milligan, G. W., e Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. Psychometrika, 50(2): pp. 159-179.

Nassif, L. F. C., e Hruschka, E. R. (2011a). Document Clustering for Forensic Computing: An Approach for Improving Computer Inspection (aceito). The Tenth International Conference on Machine Learning and Applications (ICMLA'2011), IEEE, Honolulu.

Nassif, L. F. C., e Hruschka, E. R. (2011b). Agrupamento de Documentos Aplicado à Computação Forense: Uma Abordagem para Aperfeiçoar Análises Periciais de Computadores (aceito). X Congresso Brasileiro de Inteligência Computacional (CBIC'2011), Fortaleza.

Nassif, L. F. C., e Hruschka, E. R. (2011c). Computação Forense via Agrupamento Hierárquico de Documentos (aceito). The Sixth International Conference on Forensic Computer Science (ICoFCS'2011), Florianópolis.

Nogueira, B. M. (2009). Avaliação de métodos não-supervisionados de seleção de atributos para Mineração de Textos. Dissertação de Mestrado. Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo. São Carlos.

Peterson, A. D., Ghosh, A. P., e Maitra, R. (2010). A systematic evaluation of different methods for initializing the K-means clustering algorithm. IEEE Transactions on Knowledge and Data Engineering. Preprint.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association, pp. 846-850.

Rijsbergen, C. J. V. (1979). Information Retrieval. Editora Butterworth.

Rivest, R. L. (1992). The MD5 Message-Digest Algorithm. Request for Comments (RFC) 1321. Networking Working Group.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20: pp. 53–65.

Salton, G. e Buckley, C. (1987). Term weighting approaches in automatic text retrieval. Relatório técnico, Ithaca, NY, EUA.

Salton, G., Yang, C. S., e Yu, C. T. (1975). A theory of term importance in automatic text analysis. Journal of the American Association Science.

Sokal, R. e Michener, C. (1958). A statistical method for evaluating systematic relationships. *University of Kansas, Science Bulletin* 38: pp. 1409-1438.

Sorensen, T. (1948). A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. *Biologiske Skrifter* 5: pp. 1–34.

Srivastava, A. N., e Sahami, M. (2009). Text Mining: Classification, Clustering, and Applications. Chapman & Hall e CRC Press. Data Mining and Knowledge Discovery Series.

Stoffel, K., Cotofrei, P., Han, D. (2010). Fuzzy Methods for Forensic Data Analysis. IEEE International Conference of Soft Computing and Pattern Recognition 2010, pp. 23 a 28.

Strehl, A. e Ghosh, J. (2002). Cluster Ensembles – A Knowledge Reuse Framework for Combining Multiple Partitions. Journal of Machine Learning Research 3 (2002), pp.

583-617.

Tan, P., Steinbach, M., e Kumar, V. (2006). Introduction to Data Mining. Addison-Wesley. pp. 515-526.

Theodoridis, S. e Koutroumbas, K. (2006). *Pattern Recognition, Terceira edição,* pp. 635. Elsevier.

Vendramin, L., Campello, R. J. G. B., Hruschka, E. R. (2010). Relative clustering validity criteria: A comparative overview. Statistical Analysis and Data Mining, v. 3, pp. 209-235.

Walter, C. (2005). "Kryder's Law". *Scientific American*. *Nature Publishing Group*.

Ward, J. H. (1963). Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, vol. 58, pp. 236–244.

Wu, X., Kumar, V. (2009). The Top Ten Algorithms in Data Mining. Chapman & Hall e CRC Press. Data Mining and Knowledge Discovery Series.

Xu, R., Wunsch II, D. C. (2005). Survey of Clustering Algorithms. IEEE Transactions on Neural Networks, vol. 16, n. 3, pp. 645-678.

Xu, R., Wunsch II, D. C. (2009). Clustering. Wiley / IEEE Press.