UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

EDUARDO NUNES BORGES

Um Método para Deduplicação de Metadados Bibliográficos baseado no Empilhamento de Classificadores

Tese apresentada como requisito parcial
para a obtenção do grau de
Doutor em Ciência da Computação

Profa. Dra. Renata Galante
Orientador

Porto Alegre, dezembro de 2013

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Borges, Eduardo Nunes

Um Método para Deduplicação de Metadados Bibliográficos baseado no Empilhamento de Classificadores / Eduardo Nunes Borges. – Porto Alegre: PPGC da UFRGS, 2013.

88 f.: il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2013. Orientador: Renata Galante.

1. Deduplicação. 2. Casamento aproximado. 3. Similaridade. 4. Aprendizado supervisionado. 5. Empilhamento de classificadores. I. Galante, Renata. II. Título.

“Voando sem instrumentos
Ao sabor do vento
Se depender de mim
Eu vou até o fim
Eu não vim até aqui
Pra desistir agora”
— HUMBERTO GESSINGER

AGRADECIMENTOS

Inicialmente eu agradeço a minha mãe Ione Nunes pelos 28 anos de incentivo e investimento na minha formação.

Aos meus colegas de mestrado e doutorado. Eles foram ótimos companheiros de pesquisa e de viagens. Em especial, a minha grande amiga Giseli Lopes, pelas discussões, contribuições científicas, ajuda no planejamento de experimentos e na formatação de artigos usando LaTeX.

Aos meus amigos Marcos Nunes, Otávio Acosta e Giseli Lopes que gentilmente me hospedaram em Porto Alegre após eu retornar para Rio Grande.

Aos meus amigos Rodrigo de Bem e Karina Machado que assumiram disciplinas sob minha responsabilidade na FURG para que eu pudesse me dedicar com exclusividade ao doutorado. Agradecimentos especiais a minha excelente colega de sala Cristina Meinhardt pela revisão de artigos.

Aos colegas e professores do grupo de pesquisa em banco de dados da Universidade Federal de Minas Gerais, cuja cooperação permitiu a publicação de muitos dos resultados desta tese.

Aos meus professores, em especial a Renata Galante e Karin Becker, pela orientação, confiança e experiência compartilhada.

Por fim, agradeço ao Instituto de Informática da UFRGS pela infra-estrutura disponibilizada, ao Centro de Ciências Computacionais da FURG pela concessão do meu afastamento, e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro durante os primeiros anos do curso.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

ACM	Association for Computing Machinery
BDBComp	Biblioteca Digital Brasileira de Computação
DBLP	Digital Bibliography & Library Project
IDF	Inverse Document Frequency
IEEE	Institute of Electrical and Electronics Engineers
INCT	Instituto Nacional de Ciência e Tecnologia
MARLIN	Multiply Adaptative Record Linkage with Induction
SGBD	Sistema Gerenciador de Bancos de Dados
SQL	Structured Query Language
SVM	Support Vector Machine
TF $\times$ IDF	Term Frequency $\times$ Inverse Document Frequency
UFMG	Universidade Federal de Minas Gerais
URL	Uniform Resource Locator
XML	Extensible Markup Language

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Metadados bibliográficos duplicados são registros que correspondem a referências bibliográficas semanticamente equivalentes, ou seja, que descrevem a mesma publicação. Identificar metadados bibliográficos duplicados em uma ou mais bibliotecas digitais é uma tarefa essencial para garantir a qualidade de alguns serviços como busca, navegação e recomendação de conteúdo. Embora diversos padrões de metadados tenham sido propostos, eles não resolvem totalmente os problemas de interoperabilidade porque mesmo que exista um mapeamento entre diferentes esquemas de metadados, podem existir variações na representação do conteúdo. Grande parte dos trabalhos propostos para identificar duplicatas aplica uma ou mais funções sobre o conteúdo de determinados campos no intuito de captar a similaridade entre os registros. Entretanto, é necessário escolher um limiar que defina se dois registros são suficientemente similares para serem considerados semanticamente equivalentes ou duplicados. Trabalhos mais recentes tratam a deduplicação de registros como um problema de classificação de dados, em que um modelo preditivo é treinado para estimar a que objeto do mundo real um registro faz referência. O objetivo principal desta tese é o desenvolvimento de um método efetivo e automático para identificar metadados bibliográficos duplicados, combinando o aprendizado de múltiplos classificadores supervisionados, sem a necessidade de intervenção humana na definição de limiares de similaridade. Sobre o conjunto de treinamento são aplicadas funções de similaridade desenvolvidas especificamente para o contexto de bibliotecas digitais e com baixo custo computacional. Os escores produzidos pelas funções são utilizados para treinar múltiplos modelos de classificação heterogêneos, ou seja, a partir de algoritmos de diversos tipos: baseados em árvores, regras, redes neurais artificiais e probabilísticos. Os classificadores aprendidos são combinados através da estratégia de empilhamento visando potencializar o resultado da deduplicação a partir do conhecimento heterogêneo adquirido individualmente pelos algoritmo de aprendizagem. O modelo de classificação final é aplicado aos pares candidatos ao casamento retornados por uma estratégia de blocagem de dois níveis bastante eficiente. A solução proposta é baseada na hipótese de que o empilhamento de classificadores supervisionados pode aumentar a qualidade da deduplicação quando comparado a outras estratégias de combinação. A avaliação experimental mostra que a hipótese foi confirmada quando o método proposto é comparado com a escolha do melhor classificador e com o voto da maioria. Ainda são analisados o impacto da diversidade dos classificadores no resultado do empilhamento e os casos de falha do método proposto.

Palavras-chave: Deduplicação, casamento aproximado, similaridade, aprendizado supervisionado, empilhamento de classificadores.

A Method for Bibliographic Metadata Deduplication based on Stacked Generalization

ABSTRACT

Duplicated bibliographic metadata are semantically equivalent records, i.e., references that describe the same publication. Identifying duplicated bibliographic metadata in one or more digital libraries is an essential task to ensure the quality of some services such as search, navigation, and content recommendation. Although many metadata standards have been proposed, they do not completely solve interoperability problems because even if there is a mapping between different metadata schemas, there may be variations in the content representation. Most of work proposed to identify duplicated records uses one or more functions on some fields in order to capture the similarity between the records. However, we need to choose a threshold that defines whether two records are sufficiently similar to be considered semantically equivalent or duplicated. Recent studies deal with record deduplication as a data classification problem, in which a predictive model is trained to estimate the real-world object to which a record refers. The main goal of this thesis is the development of an effective and automatic method to identify duplicated bibliographic metadata, combining multiple supervised classifiers, without any human intervention in the setting of similarity thresholds. We have applied on the training set cheap similarity functions specifically designed for the context of digital libraries. The scores returned by these functions are used to train multiple and heterogeneous classification models, i.e., using learning algorithms based on trees, rules, artificial neural networks and probabilistic models. The learned classifiers are combined by stacked generalization strategy to improve the deduplication result through heterogeneous knowledge acquired by each learning algorithm. The final model is applied to pairs of records that are candidate to matching. These pairs are defined by an efficient two phase blocking strategy. The proposed solution is based on the hypothesis that stacking supervised classifiers can improve the quality of deduplication when compared to other combination strategies. The experimental evaluation shows that the hypothesis has been confirmed by comparing the proposed method to selecting the best classifier or the majority vote technique. We also have analyzed the impact of classifiers diversity on the stacking results and the cases for which the proposed method fails.

Keywords: deduplication, approximate matching, similariry, supervised learning, stacked generalization.

1 INTRODUÇÃO

Bibliotecas digitais são sistemas de informação complexos construídos para atender as necessidades de informação de comunidades específicas (GONÇALVES et al., 2004). Elas são compostas por coleções de objetos digitais ricos, possivelmente multimídia, e oferecem serviços como busca, navegação e recomendação de conteúdo. Esses serviços permitem que os membros de uma comunidade alvo possam acessar e recuperar os objetos digitais de interesse (FOX et al., 1995; GONÇALVES et al., 2004).

Geralmente, as coleções de objetos digitais são descritas por um conjunto de registros de metadados que compõem um catálogo cuja função é descrever, organizar e especificar como esses objetos podem ser recuperados e manipulados. Além disso, o catálogo é responsável por controlar as permissões de acesso dos usuários. Com o objetivo de facilitar a interoperabilidade entre bibliotecas digitais diferentes e outros sistemas similares, os registros frequentemente seguem um padrão de metadados. Este padrão especifica, entre outras coisas, um conjunto de campos e a semântica de cada um para descrever um objeto digital. O *Dublin Core* (DCMI USAGE BOARD, 2012), por exemplo, é um padrão de metadados descritivos utilizado para representar e armazenar informação sobre publicações científicas e páginas *web*.

Embora muito úteis, estes padrões podem ser heterogêneos tanto no conteúdo quanto na estrutura, portanto eles não resolvem completamente todos os problemas de interoperabilidade dos sistemas de informação. Também não há um consenso entre todos as bibliotecas digitais em termos de um único padrão de fato. Mesmo se tal consenso existisse, diferenças na forma como alguns metadados são preenchidos, sem mencionar possíveis erros neste processo de aquisição das informações (digitação, ortografia, extração automática), permitem a existência de vários registros diferentes que descrevem o mesmo objeto digital. Os registros duplicados afetam diretamente a qualidade dos principais serviços das bibliotecas digitais como a busca e a navegação.

Considere o exemplo da Figura 1.1 que apresenta fragmentos de registros de metadados extraídos de três bibliotecas digitais distintas: BDBComp[1], DBLP[2] e IEEE Xplore[3]. Todos os registros fazem referência ao mesmo objeto digital. O campo *source* nos metadados extraídos da BDBComp (linha 3) corresponde ao campo *booktitle* da DBLP (linha 6). A estrutura dos metadados é diferente, mas ambos os campos referem-se à mesma propriedade: ao veículo de publicação de um artigo científico. Além disso, o autor do artigo, descrito pelos campos *creator* e *author*, varia na forma como é representado em cada fonte de dados (linhas 2, 5 e 8). Os valores do campo *title* também diferem na palavra "Remote" (linhas 1, 4 e 7).

[1] `http://www.lbd.dcc.ufmg.br/bdbcomp`. Data do acesso: 8/11/2013.

[2] `http://www.informatik.uni-trier.de/~ley/db`. Data do acesso: 8/11/2013.

[3] `http://ieeexplore.ieee.org`. Data do acesso: 8/11/2013.

BDBComp

```
1 <title>A Computer Vision Framework for Remote Eye Gaze Tracking</title>
2 <creator>Carlos H. Morimoto</creator>
3 <source>sibgrapi2003</source>
```

DBLP

```
4 <title>A Computer Vision Framework for Eye Gaze Tracking</title>
5 <author>Carlos Hitoshi Morimoto</author>
6 <booktitle>SIBGRAPI</booktitle>
```

IEEE Xplore

```
7 <title>A computer vision framework for eye gaze tracking</title>
8 <author>Morimoto, C.H.</author>
9 <pages>406</pages>
```

Figura 1.1: Heterogeneidade dos metadados.

Além dos problemas apresentados, algumas bibliotecas digitais armazenam a referência bibliográfica completa, sem discriminar os campos de metadados. Essas referências podem estar formatadas usando diferentes estilos como APA, Chicago, IEEE, Harvard e Vancouver (NEVILLE, 2010). Considere o exemplo da Figura 1.2 que apresenta três referências bibliográficas para o mesmo artigo científico oriundas de fontes distintas. Além dos estilos diferentes, elas apresentam outros problemas como omissão dos campos números de página (2 e 3), volume e número da revista (2). Também foram omitidos coautores (1 e 2). Na segunda referência, foi utilizado um acrônimo para representar o veículo de publicação.

1 Elmagarmid, A. et al., 2007. Duplicate record detection: a survey. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 19, No. 1, pp. 1-16.

2 Ahmed Elmagarmid and Vassilios Verykios. Duplicate Record Detection: A Survey, IEEE TKDE, 2007.

3 Elmagarmid, A.K.; Ipeirotis, P.G.; Verykios, V.S. Duplicate record detection. *Knowlege and Data Engineering*, IEEE Trans., 19(1), 2007.

Figura 1.2: Múltiplas referências bibliográficas para a mesma produção científica.

A tarefa de identificar registros duplicados que se referem a mesma entidade do mundo real é denominada deduplicação (CARVALHO et al., 2006). Esta tarefa pode ser bastante difícil devido principalmente a problemas mencionados anteriormente sintetizados a seguir:

- estrutura dos metadados distinta;
- variação na representação do conteúdo;
- diferentes estilos de formatação;
- omissão de determinados campos;
- uso de acrônimos;

- omissão de coautores.

Nos últimos anos, diversos métodos foram propostos para a deduplicação de registros. Muitos desses trabalhos focam na deduplicação no contexto da integração de dados relacionais (WHANG; GARCIA-MOLINA, 2013; DAL BIANCO et al., 2013; KIM; LEE, 2012; CARVALHO et al., 2012; WANG; LI; FENG, 2012; FREITAS et al., 2010; DORNELES et al., 2009; ZHAO; RAM, 2008). Poucas abordagens automáticas foram desenvolvidas especificamente para registros de metadados bibliográficos (TREERATPITUK; GILES, 2009; LAWRENCE; GILES; BOLLACKER, 1999).

No âmbito das bibliotecas digitais, a deduplicação é frequentemente baseada na semântica dos campos de metadados. Por exemplo, campos que especificam a autoria de um registro bibliográfico estão entre os mais discriminativos e, portanto, esta informação pode ser usada como uma forte evidência de similaridade no processo de deduplicação. Podem existir várias referências com títulos similares, mas se os autores não tiverem nomes similares, então provavelmente os registros referem-se a publicações diferentes. Por exemplo, tanto Baeza-Yates e Ribeiro-Neto (2011) quanto Manning, Raghavan e Schutze (2008) publicaram livros com títulos similares: *Modern Information Retrieval* e *Introduction to Information Retrieval*.

Outro problema frequente nas bibliotecas digitais é a variação na representação dos nomes de autores nas referências e citações bibliográficas. Essas variações incluem abreviações, inversões na ordem dos nomes e omissão de sufixos como Jr (LEY, 2002). Alguns trabalhos focam na desambiguação dos nomes de autores, principalmente a partir da informação de coautoria (TORVIK; SMALHEISER, 2009; HUANG; ERTEKIN; GILES, 2006; ON et al., 2005) ou extraídas da Web (PEREIRA et al., 2009).

Grande parte dos métodos propostos para identificação de duplicatas utiliza o conceito de medida de similaridade (DORNELES; GONÇALVES; MELLO, 2011). Esta medida é calculada através de uma função de similaridade ou de distância. Seja R o conjunto de todos os valores do tipo registro a ser comparado e $\mathbb{R}_1$ o conjunto de números reais no intervalo $[0, 1]$, a função de similaridade $f : \{R \times R\} \rightarrow \mathbb{R}_1$ recebe como parâmetros os registros $a, b \in R$ e retorna um escore de similaridade $s \in \mathbb{R}_1$. Já as funções de distância podem retornar valores de qualquer tipo e com qualquer distribuição. Portanto, é bastante comum normalizar ou mapear o conjunto imagem dessas funções para o conjunto $\mathbb{R}_1$.

Muitas funções de similaridade textual podem ser utilizadas para a tarefa de deduplicação (COHEN; RAVIKUMAR; FIENBERG, 2003; NAVARRO, 2001). Estas funções podem ser divididas em dois grupos de acordo com a granularidade dos objetos comparados: caracteres individuais ou palavras (*tokens*). *Levenshtein*, *Jaro*, *Jaro-Winkler*, *n-grams* e *Soundex* são exemplos de funções do primeiro grupo. *Jaccard*, *MongeElkan* e *Soft* TF $\times$ IDF comparam cadeias de caracteres separadas por espaços em branco. Os pacotes de código aberto SecondString[4] e SimMetrics[5] implementam estas e muitas outras funções.

Usualmente, algoritmos de deduplicação combinam os valores retornados por quaisquer funções de similaridade aplicadas a quaisquer atributos gerando um escore de similaridade entre os registros. Se este escore exceder um limiar de similaridade predefinido, os registros são considerados suficientemente similares para representar o mesmo objeto do mundo real. Entretanto, os escores de similaridade dependem do conteúdo ou valores comparados, da imagem das funções de similaridade envolvidas e do algoritmo de casamento aproximado. Portanto, a escolha de limiares de similaridade efetivos não é uma

[4] `http://secondstring.sourceforge.net`. Data do acesso: 8/11/2013.

[5] `http://sourceforge.net/projects/simmetrics/`. Data do acesso: 8/11/2013.

tarefa trivial. Nos últimos anos, algumas soluções para escolher valores de limiares apropriados foram propostas (ZHAO, 2008; STASIU; HEUSER; SILVA, 2005), porém este é um problema complexo ainda em aberto e está fora do escopo desta tese.

Para evitar o ônus da definição de valores adequados aos limiares de similaridade, muitos trabalhos têm proposto o uso de alguma técnica de aprendizado de máquina em particular, tais como: árvores de decisão (VERYKIOS; ELMAGARMID; HOUSTIS, 2000; TEJADA; KNOBLOCK; MINTON, 2001), *Support Vector Machines* (SVM) (BILENKO; MOONEY, 2003; CHEN; KALASHNIKOV; MEHROTRA, 2009; DAL BIANCO et al., 2013), regressão logística (ZHAO; RAM, 2008) ou programação genética (CARVALHO et al., 2012; FREITAS et al., 2010). Cada técnica tende a funcionar melhor em um caso particular, com um determinado conjunto de funções de similaridade e para um determinado conjunto de dados. Não há uma única técnica que seja sempre superior às demais.

Esta tese apresenta um método efetivo e automático para identificar metadados bibliográficos duplicados, baseado em heurísticas e em aprendizado de máquina. Um conjunto de funções de similaridade desenvolvidas especialmente para o domínio das bibliotecas digitais identifica variações na representação dos nomes dos autores utilizando um algoritmo com complexidade computacional linear. Os escores retornados pelas funções de similaridade são agregados utilizando múltiplos algoritmos de classificação supervisionados. Um modelo de classificação final é gerado usando a estratégia de empilhamento (WOLPERT, 1992), que consiste em um método para combinar múltiplos classificadores a partir de algoritmos de aprendizagem diferentes aplicados sobre um único conjunto de dados. Dessa forma, a qualidade da deduplicação é potencializada porque é considerado o conhecimento adquirido pelos classificadores combinados.

A aplicação de classificadores supervisionados remove a intervenção humana necessária para definição de limiares de similaridade e a combinação de múltiplos classificadores evita o forte acoplamento a uma determinada técnica de aprendizado. Assim, o método proposto pode ser aplicado nas mais diversas situações, aproveitando o conhecimento específico de cada classificador para identificar mais precisamente determinados registros de metadados bibliográficos duplicados.

A solução proposta é baseada na hipótese de que o empilhamento de classificadores supervisionados pode aumentar a qualidade da deduplicação quando comparado à escolha do melhor classificador ou a estratégias de combinação como o voto da maioria. Portanto, a tese visa esclarecer o ganho real do empilhamento frente às outras abordagens citadas.

Em suma, as principais contribuições desta tese são:

- a avaliação de um conjunto de heurísticas para identificação de metadados bibliográficos duplicados;

- a especificação de funções de similaridade desenvolvidas especialmente para o domínio das bibliotecas digitais, capazes de identificar variações na representação dos nomes dos autores utilizando um algoritmo com complexidade computacional linear;

- um método efetivo para combinar os escores retornados pelas funções de similaridade propostas utilizando algoritmos de classificação supervisionados, removendo a intervenção humana necessária para definir limiares de similaridade e aumentando a qualidade da identificação de metadados bibliográficos duplicados;

- uma estratégia de seleção aleatória de exemplos de treinamento baseada no cálculo amostral e em características do conjunto de dados;
- uma abordagem para empilhar classificadores supervisionados visando deduplicar metadados bibliográficos;
- a análise do ganho proporcionado pelo empilhamento em relação ao melhor classificador e a outra estratégia de combinação de classificadores (voto da maioria);
- uma análise dos casos de falha do método proposto;
- uma análise do impacto da diversidade dos classificadores no resultado do empilhamento.

O restante do texto está organizado da seguinte forma:

- No Capítulo 2, é apresentada uma revisão bibliográfica que aborda os assuntos permeados por esta tese. Este capítulo inclui conceitos sobre descoberta de conhecimento, classificação supervisionada, estratégias de combinação e medidas de diversidade de classificadores.
- O Capítulo 3 discute os trabalhos relacionados e posiciona a contribuição da tese na literatura atual. São abordados diferentes métodos de deduplicação de registros (instâncias) organizados em relação à dependência da definição de limiares de similaridade. Trabalhos que combinam diferentes deduplicadores com o objetivo de aumentar a qualidade do resultado são destacados no final do capítulo. Ainda são evidenciadas as particularidades da deduplicação de registros bibliográficos que não são tratadas pelos trabalhos apresentados.
- No Capítulo 4, é apresentado o método proposto nesta tese para identificar metadados bibliográficos duplicados a partir do empilhamento de classificadores. São especificadas em detalhe as fases que compõem o método: pré-processamento, seleção de exemplos de treinamento, modelagem, empilhamento de classificadores, blocagem e deduplicação.
- Uma série de experimentos que avaliam o método proposto são relatados no Capítulo 5. Esses experimentos envolvem quatro conjuntos de dados reais extraídos de bases de referências bibliográficas públicas. Múltiplas estratégias de empilhamento são avaliadas em função da qualidade da deduplicação e em relação à diversidade dos modelos de classificação. Os resultados são comparados com a qualidade do melhor classificador da fase de modelagem e com a estratégia de combinação voto da maioria.
- Por fim, no Capítulo 6, são apresentadas as considerações finais. As principais contribuições desta tese e os resultados obtidos no decorrer do doutorado são sintetizados e associados à produção bibliográfica resultante. Também são discutidos alguns pontos interessantes que podem ser explorados como trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Esse capítulo apresenta os principais conceitos relacionados ao aprendizado supervisionado de classificadores. Estes conceitos são necessários para o entendimento do método de deduplicação apresentado nesta tese. O método proposto identifica pares de registros de metadados bibliográficos duplicados com base no aprendizado do resultado das funções de similaridade propostas. O conhecimento aprendido pelos diversos modelos de classificação é combinado para gerar uma predição final com maior qualidade.

A Seção 2.1 apresenta a definição de descoberta de conhecimento, destacando a tarefa da classificação na etapa de mineração de dados. As funções de similaridade propostas são combinadas utilizando os classificadores apresentados na Seção 2.2. Na Seção 2.3, são apresentadas duas estratégias de combinação de classificadores utilizadas na avaliação experimental do método proposto. Uma série de medidas que avaliam a diversidade dos classificadores combinados é especificada na Seção 2.4. Por fim, são discutidas algumas considerações que relacionam os assuntos estudados com o objetivo desta tese.

2.1 Descoberta de conhecimento e classificação de dados

Descoberta de conhecimento em bases de dados (*Knowledge Discovery in Databases* - KDD) é um processo não trivial de identificar padrões potencialmente úteis e compreensíveis em meio às observações presentes em uma base de dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). Geralmente, esses padrões são extraídos de relacionamentos implícitos entre os dados analisados. Como resultado final, os padrões encontrados devem gerar conhecimento inteligível e imediatamente utilizável para o apoio às decisões.

A descoberta de conhecimento é dividia em cinco fases distintas (HAN; KAMBER, 2006) (Figura 2.1):

- seleção de dados - escolha do conjunto de dados contendo todas as possíveis variáveis (atributos) e observações (registros) que farão parte da análise. Esta fase pode ser bem complexa, uma vez que os dados podem ser extraídos de fontes distintas e heterogêneas (bancos de dados, *data warehouses*, planilhas, textos, páginas *Web*, etc.) e ainda podem possuir os mais diversos formatos.

- pré-processamento - limpeza e normalização dos dados. Inclui outras tarefas como remoção de ruído (*outliers*), tratamento de registros incompletos e remoção de redundância.

- transformação dos dados - adequação dos dados em relação à técnica e algoritmo de mineração a serem utilizados. Esta fase inclui a escolha da representação dos dados e a redução de dimensionalidade (número de atributos).

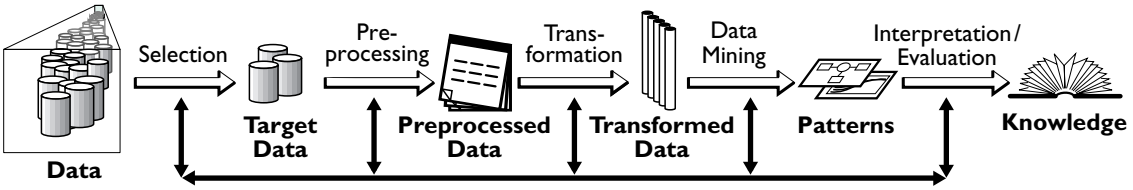

Figura 2.1: Visão geral do processo de descoberta do conhecimento em bases de dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

- mineração - análise automática dos dados em busca de padrões utilizando algoritmos de mineração de dados.

- interpretação de resultados - fase final responsável pela geração de conhecimento baseada nos padrões encontrados.

Caso não sejam encontrados resultados relevantes em quaisquer fases, o processo deve retornar a uma das fases anteriores para que os parâmetros necessários sejam devidamente ajustados.

A mineração de dados é a principal fase do processo. Ela pode ser definida como a análise automática dos dados em busca de padrões, que é apoiada por algoritmos de aprendizado de máquina, estatísticas e técnicas de visualização. Dependendo do objetivo da mineração, diversas técnicas podem ser utilizadas (TAN; STEINBACH; KUMAR, 2005):

- associação - definição de regras do tipo $A \rightarrow B$, em que A e B são elementos que co-ocorrem em diferentes observações da bases de dados. Uma das aplicações clássicas da associação consiste na descoberta de produtos que são comprados juntos. A análise dos resultados pode determinar que ações devem ser tomadas para incrementar a venda de B, que produtos são afetados pelo produto A e que promoções podem incluir A para incrementar a venda de B.

- descoberta de padrões sequenciais - definição de regras do tipo $(A)(C) \rightarrow B$ onde os pares AB e CB são elementos que co-ocorrem em diferentes observações da bases de dados, sendo que CB ocorre após AB. Portanto, é possível descobrir que um cliente que comprou o produto A e logo depois comprou o produto C, possivelmente comprará o produto B.

- agrupamento (*clustering*) - classificação não supervisionada de registros em grupos. É realizado com base na similaridade entre os registros. Deve-se maximizar a similaridade intragrupo e minimizar a similaridade intergrupo. Exemplos de métodos de agrupamento são: particionamento, hierárquico e incremental.

- classificação - método supervisionado que determina um modelo para um determinado atributo classe que é função dos valores dos outros atributos. Pode ser utilizado para predizer a que classe de dados uma nova instância pertence.

- regressão - predição do valor de uma variável contínua baseado no valor de outras variáveis, considerando um modelo de dependência linear ou não linear.

- detecção de desvios - identificação de desvios significativos em relação ao comportamento normal dos dados. Pode ser utilizado na detecção de fraudes, gargalos de um sistema, etc.

Uma das técnicas de mineração de dados amplamente utilizada é a classificação de dados. A classificação consiste no processo de encontrar, através de aprendizado de máquina supervisionado, um modelo ou função que descreva diferentes classes de dados (HAN; KAMBER, 2006). O aprendizado é supervisionado porque um agente externo rotula os exemplos de treinamento e age como um supervisor do processo de aprendizagem, fornecendo as saídas esperadas (classes de dados). O objetivo da classificação é rotular, automaticamente, novas instâncias da base de dados com uma determinada classe aplicando o modelo ou função aprendidos. Este modelo é baseado no valor dos atributos das instâncias de treinamento. Após a classificação, os dados de teste estarão categorizados em classes.

2.2 Classificadores supervisionados

A literatura científica descreve um largo conjunto de algoritmos de classificação (MITCHELL, 1997). Estes algoritmos podem ser organizados em diferentes tipos, de acordo com as características que utilizam no aprendizado. Cada tipo é mais indicado para um determinado conjunto de dados.

RIPPER (COHEN, 1995) é um algoritmo baseado em regras. Os atributos de entrada A_j e seus valores v_j são combinados por operadores relacionais op e usados em expressões condicionais para formar um conjunto de regras $r_i : (A_1\ op\ v_1) \wedge (A_2\ op\ v_2) \wedge ... \wedge (A_k\ op\ v_k) \rightarrow y_i$, em que y_i é a classe predita pela regra r_i. As regras são induzidas sequencialmente e para uma classe de cada vez. A última regra classifica todas as instâncias remanescentes. O algoritmo é dividido em duas etapas: crescimento e poda de regras. Na primeira, os atributos são combinados utilizando a medida de avaliação FOIL's *information gain* (QUINLAN; CAMERON-JONES, 1993). Já na segunda etapa, a poda de regras depende da avaliação realizada a partir da medida *minimum description length* proposta pelo autor.

Alguns algoritmos, tais como *CART* (BREIMAN et al., 1984), *ID3* (QUINLAN, 1986) e *C4.5* (QUINLAN, 1993) utilizam árvores de decisão para classificar registros. Uma árvore de decisão é composta por nós intermediários que representam atributos. As arestas definem um conjunto de valores que cada atributo pode assumir. Os nós folha indicam a classe de dados utilizada para rotular uma instância. As regras de classificação são extraídas a partir de todos os possíveis caminhos entre o nó raiz e as folhas. A Figura 2.2 apresenta um exemplo de árvore de decisão aprendida a partir de um conjunto de obser-

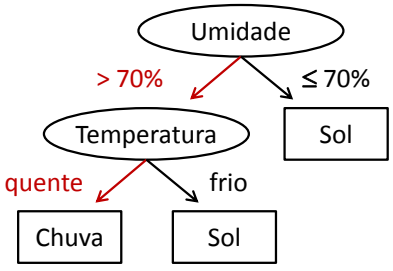

Figura 2.2: Exemplo de árvore de decisão.

vações meteorológicas, que rotula com a classe $Chuva$ uma observação com temperatura quente e umidade maior que 70%.

Um terceiro tipo de classificador baseia-se em redes neurais artificiais. *Multilayer Perceptron* (*MLP*) (HAYKIN, 2007) é uma rede que pode conter, além das camadas de entrada e saída do *perceptron*, camadas de nós intermediários denominadas camadas ocultas. A Figura 2.3 mostra um exemplo de rede destacando cada camada. *MLP* implementa o algoritmo *back propagation* (HECHT-NIELSEN, 1989) para atualizar os pesos da rede. Entre as adaptações do algoritmo original, *Voted Perceptron* (FREUND; SCHAPIRE, 1998) utiliza uma estratégia de eleição ponderada para combinar múltiplas predições.

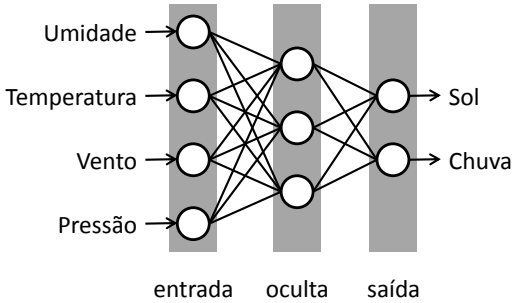

Figura 2.3: Exemplo de rede neural artificial multicamada.

Entre outros algoritmos baseados em funções matemáticas, destacam-se os *Support Vector Machines* (*SVM*) (BOSER; GUYON; VAPNIK, 1992). *SVM* é um algoritmo de classificação binária que traça um hiperplano ótimo que maximiza a margem de separação entre duas classes de dados. A etapa principal do algoritmo é descobrir os vetores de suporte que são as instâncias equidistantes do hiperplano. A Figura 2.4 apresenta um exemplo com um espaço de dados bidimensional destacando os vetores de suporte.

Quando os dados não são linearmente separáveis, o espaço de entrada é transformado aplicando uma função de núcleo que eleva o número de dimensões até que seja encontrado um espaço passível de separação linear. *Sequential Minimal Optimization* (*SMO*) (PLATT, 1999) é uma variação do *SVM* otimizada que utiliza uma quantidade de memória linear em relação ao tamanho do conjunto de treinamento.

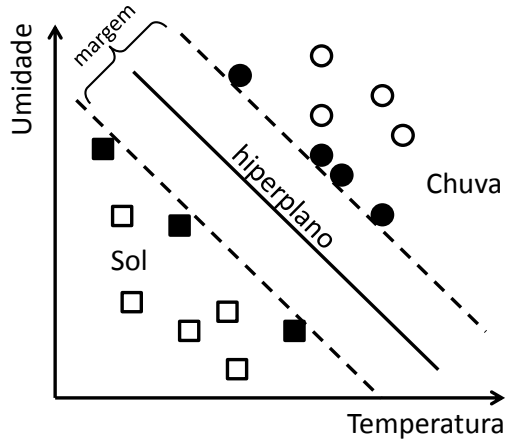

Figura 2.4: Exemplo de hiperplano e os vetores de suporte para classificação de dados linearmente separáveis.

Entre os modelos probabilísticos, destacam-se os algoritmos *Naïve Bayes* (JOHN; LANGLEY, 1995) e *Bayes Net* (COOPER; HERSKOVITS, 1992), ambos baseados no teorema de Bayes. O primeiro é dito ingênuo por não assumir relação de dependência entre os atributos de entrada. *Naïve Bayes* computa a probabilidade $P(c|r)$ de um registro r pertencer a uma determinada classe c a partir da probabilidade *a priori* $P(c)$ de um registro ser desta classe e das probabilidades condicionais $P(v_k|c)$ de cada valor v_k de atributo ocorrer em um registro da mesma classe. O objetivo do algoritmo é encontrar a melhor classe para um registro maximizando a probabilidade *a posteriori*. Já *Bayes Net* é um classificador que permite relacionar pares de atributos condicionalmente dependentes. O modelo de classificação é uma rede representada por um grafo acíclico direcionado em que os nós são os atributos de entrada e os arcos são as dependências entre eles. O algoritmo calcula uma tabela de probabilidades associando cada nó com seus vizinhos incidentes.

2.3 Combinando classificadores

Classificadores que implementam algoritmos diferentes potencialmente oferecem informação complementar sobre os padrões a serem classificados. Portanto, a combinação das predições realizadas por estes algoritmos pode aumentar a qualidade do processo de classificação. Esta seção apresenta as estratégias para combinar múltiplos classificadores utilizadas nesta tese.

2.3.1 Voto da maioria

Para a maioria dos problemas com várias soluções, a eleição (*voting*) pode ser utilizada para melhorar a confiabilidade e a acurácia dos sistemas. Na sua forma mais simples, essa estratégia trata os sistemas como caixas pretas, sem necessitar de informação adicional para sua implementação. A eleição é vantajosa especialmente quando utilizada para combinar classificadores comerciais de prateleira (*commercial off-the-shelf*[1]).

Conhecido na literatura científica como *plurality* (LIN et al., 2003), o voto da maioria é uma estratégia de eleição simples que conta o número de votos recebidos de cada classificador para cada classe de dados. A classe que receber o maior número de votos, ou seja, da maioria dos classificadores, é selecionada como rótulo.

Seja $L_i|1 \leq i \leq N$ um conjunto de N algoritmos de aprendizado e S um conjunto de dados formado por n instâncias $s_j = (x_j, y_j)|1 \leq j \leq n$, em que x_j é um vetor de características e y_j é o rótulo da classe de dados, cada classificador $C_i = L_i(S)$ é construído usando o mesmo conjunto de dados de treinamento. Para cada instância s_j, os classificadores aprendidos são utilizados para gerar predições $\hat{y}_j^i = C_i(x_j)$. O voto da maioria é a classe predita com maior número de votos recebidos por todos os classificadores.

A Tabela 2.1 apresenta um exemplo de instância com o vetor de características $x = (31, 91, 13, 1008)$ rotulado com a classe $y = Chuva$ que representa uma condição atmosférica num determinado momento.

Tabela 2.1: Instância representando uma condição atmosférica.

Temperatura (°C)	**Umidade (%)**	**Vento (km/h)**	**Pressão (Pa)**	**Tempo**
31	91	13	1008	Chuva

[1] *Software* pré-compilado adquirido ou licenciado como alternativa ao desenvolvimento por completo.

Considerando cinco classificadores construídos a partir do conjunto de dados atmosféricos e as seguintes predições $\hat{y}^1 = Chuva$, $\hat{y}^2 = Chuva$, $\hat{y}^3 = Encoberto$, $\hat{y}^4 = Chuva$, $\hat{y}^5 = Sol$, a estratégia de combinação voto da maioria rotula a instância apresentada na Tabela 2.1 com a classe $Chuva$ porque recebeu três votos enquanto as classes $Encoberto$ e Sol receberam apenas um voto cada.

Majority voting (KITTLER et al., 1998) é uma variação dessa estratégia de combinação a qual exige que pelo menos metade dos classificadores envolvidos decidam sobre a mesma classe de dados. Considerando as mesmas predições apresentadas no parágrafo anterior, essa variação do voto da maioria rotula a instância com a classe $Chuva$ porque recebeu três (60%) dos cinco possíveis votos. Quando existem apenas duas classes de dados, *majority voting* é equivalente à estratégia *plurality*.

2.3.2 Empilhamento

O empilhamento, conhecido na literatura como *stacked generalization* (TING; WITTEN, 1999, 1997; WOLPERT, 1992) ou simplesmente *stacking* (DZEROSKI; ZENKO, 2004), consiste em um método para combinar múltiplos classificadores. Estes classificadores são gerados a partir de algoritmos de aprendizagem diferentes aplicados sobre um único conjunto de dados.

Seja $L_i|1 \leq i \leq N$ um conjunto de N algoritmos de aprendizado e S um conjunto de dados formado por n instâncias $s_j = (x_j, y_j)|1 \leq j \leq n$, em que x_j é um vetor de características e y_j é o rótulo da classe de dados, um conjunto de classificadores $C_i = L_i(S)$ é gerado na primeira fase do empilhamento. Na segunda fase, um classificador de metanível é treinado combinando as saídas dos classificadores de nível básico.

Para gerar um conjunto de treinamento para o aprendizado do classificador de metanível é aplicado o procedimento de validação cruzada ou *leave-one-out*. Para cada instância s_j, os classificadores aprendidos são utilizados para gerar predições $\hat{y}^i_j = C_i(x_j)$. O conjunto de dados do metanível consiste de n instâncias $(\hat{y}^i_j, y_j))|1 \leq i \leq N \wedge 1 \leq j \leq n$, em que o vetor de características $\hat{y}^i_j$ é formado pelos rótulos das classes preditas pelos N classificadores de nível básico, mantendo-se a mesma classe original de dados y_j.

A Tabela 2.2 apresenta uma instância do conjunto de treinamento do metanível usando três classificadores de nível básico $\{a, b, c\}$ construídos a partir dos dados originais. O vetor de características $x = (Encoberto, Chuva, Encoberto)$ corresponde às predições feitas pelos classificadores para a instância original apresentada na Tabela 2.1.

Tabela 2.2: Instância do conjunto de treinamento do metanível usando três classificadores de nível básico.

Previsão a	**Previsão** b	**Previsão** c	**Tempo**
Encoberto	Chuva	Encoberto	Chuva

Por fim, o classificador do metanível é treinado a partir do conhecimento adquirido pelos classificadores de nível básico e é finalmente usado para inferir a classe das instâncias do conjunto de teste.

2.3.2.1 *Empilhando distribuições de probabilidade*

TING; WITTEN (1999) alteram o método de empilhamento original empilhando distribuições de probabilidade das predições dos classificadores. Ao invés dos rótulos das classes preditas, o vetor de características do metanível é formado pela confiança dos classificadores de nível básico.

Seja K o número de classes de dados originais distintas, o conjunto de dados do metanível consiste de n instâncias $(p^{C_i}(c_k|x_j), y_j)|1 \leq i \leq N \wedge 1 \leq j \leq n \wedge 1 \leq k \leq K$, em que o vetor de características $p^{C_i}(c_k|x_j)$ é formado pelas probabilidades preditas para cada classes de dados c_k usando cada classificador de nível básico C_i. É adicionado como último atributo a classe original de dados y_j.

A Tabela 2.3 apresenta uma instância do conjunto de treinamento do metanível usando dois classificadores de nível básico $\{a, b\}$ construídos a partir dos dados originais. O vetor de características $x = (0.20, 0.75, 0.05, 0.95, 0.05, 0.00)$ corresponde às confianças nas predições feitas pelos classificadores, para cada classe, sobre a instância original apresentada na Tabela 2.1.

Tabela 2.3: Instância do conjunto de treinamento do metanível usando as confianças de dois classificadores de nível básico.

Confiança a			**Confiança** b			**Tempo**
Chuva	**Encoberto**	**Sol**	**Chuva**	**Encoberto**	**Sol**	
0,20	0,75	0,05	0,95	0,05	0,00	Chuva

Os autores demonstram que esta estratégia é bastante eficaz quando utiliza no metanível uma adaptação de regressão linear denominada *Multi-response Linear Regression* (*MLR*). O algoritmo divide o problema de classificação em diversos problemas de regressão, um para cada classe.

2.3.2.2 *Estendendo os atributos do metanível*

DZEROSKI; ZENKO (2004) propõem uma extensão do método de empilhamento apresentado anteriormente. Além da distribuição de probabilidades de cada classe, são gerados outros dois tipos de atributos.

Seja x_j o vetor de características da instância j. Para todo classificador $C_i|1 \leq i \leq N$ e para toda classe $c_k|1 \leq k \leq K$, é definido um atributo como a distribuição de probabilidade da classe multiplicada pela probabilidade máxima considerando todas as classes, conforme Equação 2.1.

$$P_{C_i}(x_j) = p^{C_i}(c_k|x_j) \times \max_{k=1}^{K}(p^{C_i}(c_k|x_j)) \tag{2.1}$$

O segundo tipo de atributo é a entropia das distribuições de probabilidade para cada classificador C_i, calculada conforme a Equação 2.2. Quanto maior a entropia maior a incerteza na predição.

$$E_{C_i}(x_j) = -\sum_{k=1}^{K} p^{C_i}(c_k|x_j)\ log_2\ p^{C_i}(c_k|x_j) \tag{2.2}$$

Estes novos atributos são aplicados a todas as instâncias $x_j|1 \leq j \leq n$ para compor o conjunto de treinamento utilizado no aprendizado do metanível. Os autores mostram que os atributos propostos são representativos e melhoram a qualidade da classificação.

A Tabela 2.4 apresenta uma instância do conjunto de treinamento do metanível em que o vetor de características $x = (0,20,\ 0,75,\ 0,05,\ 0,15,\ 0,56,\ 0,04,\ 0,99,\ 0,95,\ 0,05,\ 0,00,\ 0,90,\ 0,05,\ 0,00,\ 0,27)$ corresponde aos atributos originais apresentados na Tabela 2.3 seguidos dos novos atributos propostos, ordenados por classificador (C).

Tabela 2.4: Instância do conjunto de treinamento do metanível usando novos atributos derivados das confianças de dois classificadores de nível básico.

C	**Confiança**			**Prod. Conf. Máx.**			**Entropia**	**Tempo**
	Chuva	**Encoberto**	**Sol**	**Chuva**	**Encoberto**	**Sol**		
a	0,20	0,75	0,05	0,15	0,56	0,04	0,99	Chuva
b	0,95	0,05	0,00	0,90	0,05	0,00	0,27	

2.4 Medidas de diversidade

A diversidade das predições é uma questão chave na combinação de classificadores. KUNCHEVA; WHITAKER (2003) definem uma série de medidas de diversidade e as relacionam com a qualidade de um sistema de classificação. Essas medidas são baseadas na concordância ou discordância dos classificadores envolvidos.

Seja n o número de instâncias avaliadas por um par de classificadores C_i e C_j e M uma matriz de relacionamento entre eles contendo a quantidade de instâncias em que cada classificador acerta (1) e/ou erra (0) a predição da classe de dados (Tabela 2.5). Por exemplo, n^{01} representa o número de instâncias que C_i erra e C_j acerta. A diagonal principal mostra a quantidade de instâncias igualmente rotuladas por ambos os classificadores. Já a diagonal secundária apresenta o número de registros em que os classificadores discordam na classe predita. O somatório de todas as células é o total de instâncias avaliadas pelos classificadores analisados.

Tabela 2.5: Matriz de relacionamento entre dois classificadores C_i e C_j.

	Acertos C_j	Erros C_j
Acertos C_i	n^{11}	n^{10}
Erros C_i	n^{01}	n^{00}
$n = n^{11} + n^{10} + n^{01} + n^{00}$		

2.4.1 Falha dupla df

A falha dupla df é definida pela Equação 2.3 e representa a proporção de erros cometidos pelos dois classificadores C_i, C_j simultaneamente em relação ao total de instâncias. df varia no intervalo fechado [0,1] e é inversamente proporcional à diversidade entre os classificadores.

$$df_{i,j} = \frac{n^{00}}{n^{00} + n^{01} + n^{10} + n^{11}} \tag{2.3}$$

2.4.2 Medida de discordância Dis

A medida de discordância Dis é definida pela Equação 2.4 e representa a razão entre o número de instâncias em que os classificadores C_i, C_j discordam e o total de instâncias. Dis varia no intervalo fechado [0,1] e é diretamente proporcional à diversidade entre os classificadores.

$$Dis_{i,j} = \frac{n^{01} + n^{10}}{n^{00} + n^{01} + n^{10} + n^{11}} \tag{2.4}$$

2.4.3 Estatística Q

A estatística Q é uma medida de diversidade definida pela Equação 2.5 que opera sobre a saída de um par de classificadores C_i, C_j. Q varia no intervalo fechado [-1,1] e é inversamente proporcional à diversidade entre os classificadores. Para classificadores que reconhecem os mesmos objetos, Q assume valor positivo. $Q = 0$ para classificadores estatisticamente independentes e negativo quando acertam as predições de instâncias distintas.

$$Q_{i,j} = \frac{n^{11}n^{00} - n^{01}n^{10}}{n^{11}n^{00} + n^{01}n^{10}} \tag{2.5}$$

2.4.4 Coeficiente de correlação ρ

O coeficiente de correlação ρ é definido pela Equação 2.6. Assim como Q, ρ opera sobre a saída de um par de classificadores C_i, C_j e retorna um valor real que varia no intervalo fechado [-1,1]. ρ também é inversamente proporcional à diversidade entre os classificadores e tem sempre o mesmo sinal que ρ, sendo $|\rho| \leq |Q|$.

$$\rho_{i,j} = \frac{n^{11}n^{00} - n^{01}n^{10}}{\sqrt{(n^{11} + n^{10})(n^{01} + n^{00})(n^{11} + n^{01})(n^{10} + n^{00})}} \tag{2.6}$$

2.4.5 Concordância entre avaliadores k

A concordância entre avaliadores k é definida pela Equação 2.7. k também opera sobre a saída de um par de classificadores C_i, C_j e retorna um valor no intervalo [-1,1]. k é inversamente proporcional à diversidade entre os classificadores.

$$k_{i,j} = \frac{2(n^{11}n^{00} - n^{01}n^{10})}{(n^{11} + n^{10})(n^{01} + n^{00})(n^{11} + n^{01})(n^{10} + n^{00})} \tag{2.7}$$

2.4.6 Entropia E

A entropia E opera sobre a saída de um conjunto de N classificadores $L = \{C_1, C_2, \ldots, C_N\}$ e é definida pela Equação 2.8,

$$E_L = \frac{1}{n}\sum_{i=1}^{n} \frac{\min[l(x_i), N - l(x_i)]}{N - \lceil N/2 \rceil} \tag{2.8}$$

em que n é o número de instâncias e $l(x_i)$ é o número de classificadores que rotularam corretamente a instância x_i. E varia no intervalo [0,1] e é diretamente proporcional à diversidade entre os classificadores.

2.4.7 Variância Kohavi-Wolpert KW

A variância Kohavi-Wolpert KW opera sobre a saída de um conjunto de N classificadores $L = \{C_1, C_2, \ldots, C_N\}$. KW diverge da medida de discordância Dis por um coeficiente conforme a Equação 2.9.

$$KW_L = \frac{N-1}{2N} Dis \tag{2.9}$$

KW varia no intervalo [0,1/2] e é diretamente proporcional à diversidade entre os classificadores.

2.4.8 Exemplo

Para exemplificar o cálculo das medidas de diversidade apresentadas, considere as Tabelas 2.6 e 2.7, as quais apresentam matrizes de relacionamento entre classificadores com baixa e alta diversidade respectivamente. Os valores resultantes das medidas que operam sobre pares de classificadores estão apresentados na Tabela 2.8. Essas medidas poderiam ser utilizadas para um conjunto de classificadores considerando a média dos valores entre todas as combinações de pares.

Tabela 2.6: Exemplo de baixa diversidade entre classificadores.

	Acertos $C4.5$	Erros $C4.5$
Acertos SMO	70	5
Erros SMO	15	10

$n = 100$

Tabela 2.7: Exemplo de alta diversidade entre classificadores.

	Acertos $RIPPER$	Erros $RIPPER$
Acertos MLP	25	50
Erros MLP	20	5

$n = 100$

Tabela 2.8: Medidas de diversidade calculadas para os exemplos apresentados.

Diversidade	df	Q	k	ρ	Dis
Alta	0,050	-0,778	-0,402	-0,406	0,700
Baixa	0,100	0,806	0,397	0,404	0,200

2.5 Considerações finais

O problema da deduplicação de registros pode ser modelado como um problema de classificação de dados. Dado um conjunto de registros que referenciam objetos do mundo real, dois registros possuem a mesma classe de dados se referenciam o mesmo objeto. O objetivo da classificação é estimar um modelo preditivo que descreva cada classe de dados distinta, ou seja, que estime a que objeto do mundo real um registro faz referência.

O aprendizado supervisionado pode ser usado tanto para modelagem descritiva de registros de metadados bibliográficos duplicados quanto para modelagem preditiva (BORGES, 2009). A primeira determina que características definem um conjunto de referências para uma mesma publicação. Por exemplo, referências com mesmo ano de publicação e título muito similares têm grande probabilidade de descrever o mesmo conteúdo. A segunda identifica referências bibliográficas que potencialmente descrevem a mesma publicação. Essa predição é feita com base nas características aprendidas a partir de um conjunto de treinamento.

Esta tese especifica um método automático para identificar metadados bibliográficos duplicados que combina o aprendizado de múltiplos classificadores supervisionados. Os classificadores aprendidos são combinados utilizando a estratégia de empilhamento visando potencializar o resultado da deduplicação a partir do conhecimento heterogêneo adquirido individualmente pelos algoritmos de aprendizagem. Quanto maior a diversidade entre os algoritmos combinados, maior a probabilidade de acerto do modelo de classificação final. O método proposto é avaliado por uma série de experimentos em que os resultados são comparados com os obtidos pela escolha do melhor classificador e pelo voto da maioria.

As métricas de avaliação de modelos de classificação são aplicadas diretamente como medida de qualidade de processo de deduplicação. Assim como a grande maioria dos trabalhos relacionados abordados no próximo capítulo, esta tese utiliza as métricas descritas para comparar os resultados alcançados com métodos estado da arte. Ainda é analisado o impacto da diversidade dos classificadores na qualidade da deduplicação.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta uma série de trabalhos relacionados a esta tese. São abordados diferentes métodos de deduplicação de registros (instâncias). Métodos específicos para identificar similaridades no esquema de estruturas complexas não fazem parte do escopo deste estudo porque metadados bibliográficos são, na grande maioria das vezes, organizados em uma estrutura plana.

A Seção 3.1 discute uma seleção de trabalhos dependentes da definição de limiares de similaridade. Esta seleção cobre diversos aspectos da tarefa de deduplicação: formalização do problema, funções de similaridade, algoritmos probabilísticos, modelo espacial vetorial de recuperação de informações, comparação de *rankings* e o mapeamento de escores. A Seção 3.2 apresenta um conjunto de trabalhos baseados em aprendizado de máquina que realizam o casamento dos registros duplicados sem a necessidade da definição de limiares. A Seção 3.3 destaca dois trabalhos que combinam outros deduplicadores com o objetivo de aumentar a qualidade do resultado. Por fim, a Seção 3.4 sintetiza e compara as principais características dos trabalhos analisados. Ainda são evidenciadas as particularidades da deduplicação de registros bibliográficos que não são tratadas pelos trabalhos apresentados.

3.1 Trabalhos dependentes da definição de limiares de similaridade

Os trabalhos apresentados nesta seção propõem métodos de deduplicação diferentes, mas dependentes da interação do usuário na definição de pelo menos um limiar.

3.1.1 FELLEGI; SUNTER (1969)

Um dos trabalhos pioneiros na área foi proposto por FELLEGI; SUNTER (1969). Os autores formalizaram uma solução para o problema de reconhecer registros duplicados utilizando-se de um modelo matemático.

Uma função de similaridade, denominada *linkage rule*, é aplicada a um par de registros e o valor retornado é comparado com os dois limiares definidos anteriormente. Se o valor exceder os limiares, o par comparado é considerado uma réplica. Se o valor for inferior, os registros são considerados distintos.

A Figura 3.1 apresenta um exemplo de histograma com a distribuição de pares em relação ao escore de similaridade retornado por uma determinada função. É necessária a intervenção humana para julgar a identificação de réplicas quando o valor retornado pela função estiver na região crítica entre os dois limiares, representada pela área central da figura. Neste caso, os registros comparados não contêm informação representativa com evidências suficientes para determinar se o par representa registros duplicados.

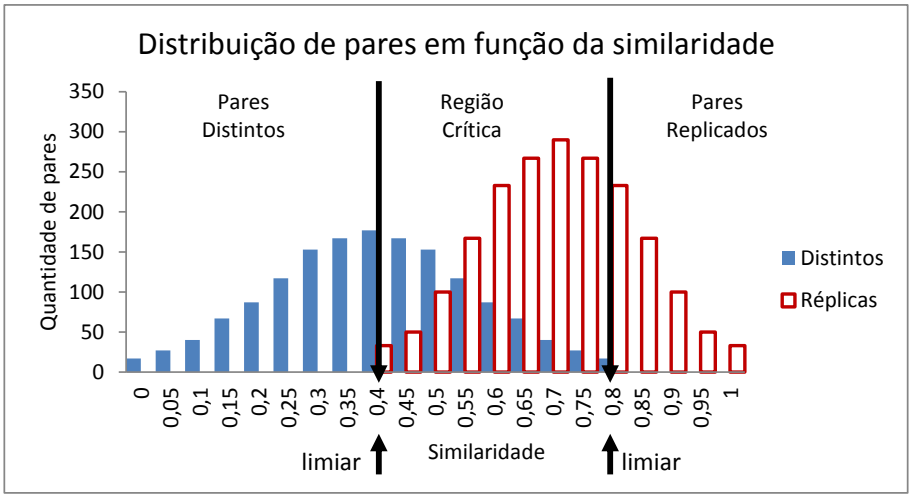

Figura 3.1: Histograma evidenciando a região crítica na distribuição de pares em função da similaridade.

3.1.2 CiteSeer (LAWRENCE; GILES; BOLLACKER, 1999)

A identificação de referências bibliográficas duplicadas é explicitamente discutida por LAWRENCE; GILES; BOLLACKER (1999). Os autores relatam as técnicas utilizadas no desenvolvimento do sistema CiteSeer, cujo principal objetivo é construir um índice de citação autônomo, sem a intervenção do usuário para catalogar informações e ligar os objetos referenciados. As principais técnicas apresentadas para o casamento aproximado de referências provenientes de múltiplas fontes de dados baseiam-se em métricas como a distância de edição (LEVENSHTEIN, 1966), frequência de palavras, casamento de frases e extração de campos como autores e título.

Uma nova versão do sistema, denominada CiteSeerX[1], foi proposta como uma biblioteca digital de literatura científica. O sistema inclui um motor de busca com foco principalmente nas áreas de Ciência da Computação e da Informação. Dentre as funcionalidades atualmente implementadas destacam-se a extração automática de metadados descritivos de todos os documentos indexados e a desambiguação de nomes de autores (TREERATPITUK; GILES, 2009).

3.1.3 CHAUDHURI et al. (2003)

CHAUDHURI et al. (2003) propõem um algoritmo probabilístico para recuperar os K registros mais similares a um registro de entrada, de acordo com uma função de casamento aproximado *fuzzy*. Essa função considera o peso das palavras contidas nos registros usando a métrica *Inverse Document Frequency* (IDF) (BAEZA-YATES; RIBEIRO-NETO, 2011) do modelo espacial vetorial de recuperação de informações.

A solução foi proposta para comparar registros no processo de carga de *data warehouses*. O parâmetro K pode ser substituído por um limiar de similaridade definido pelo usuário. Neste caso, é realizada uma consulta por abrangência (CHáVEZ et al., 2001) recuperando todos os itens com similaridade suficiente em relação ao registro de entrada.

Para reduzir o número de pares a ser comparado é utilizada blocagem baseada em *q-grams*. A técnica proposta é extensível, pois possibilita o uso de funções de pesos específicas para certos domínios ao invés dos pesos gerados pela métrica IDF.

[1] `http://citeseerx.ist.psu.edu/`. Data do acesso: 8/11/2013.

3.1.4 CARVALHO; SILVA (2003)

CARVALHO; SILVA (2003) também utilizam o modelo vetorial para calcular a similaridade entre objetos provenientes de múltiplas fontes de dados. Este método pode ser utilizado para deduplicar objetos com estruturas complexas, como documentos XML. É assumido que conjuntos de atributos equivalentes semanticamente são previamente identificados por uma determinada abordagem de casamento de esquemas (RAHM; BERNSTEIN, 2001).

São propostas e avaliadas quatro estratégias distintas para atribuir pesos e combinar escores de similaridade para cada atributo:

1. cada vetor é composto por todos os termos da coleção. É utilizado o modelo vetorial tradicional para o cálculo da similaridade;

2. os vetores são compostos apenas pelos termos semanticamente equivalentes em ambos os objetos. É utilizado o modelo vetorial para o cálculo da similaridade;

3. um vetor é construído para cada atributo semanticamente equivalente em ambos os objetos. A similaridade é calculada a partir do somatório das similaridades (modelo vetorial) entre cada par de vetores. O peso $\phi = 1$ é utilizado para cada atributo, ou seja, a importância de cada atributo para o cálculo da similaridade é a mesma;

4. a mesma estratégia do item anterior com a exceção de que o peso de cada atributo pode assumir um valor $\phi \geq 0$.

Os experimentos realizados pelos autores indicam que a estratégia 4 obtém melhores resultados, mas a proposta não determina automaticamente o valor dos pesos ϕ associados a cada atributo. Esta tarefa fica por conta do usuário.

3.1.5 PROM (DOAN et al., 2003)

O Sistema *PROfile-based Object Matching* (PROM) (DOAN et al., 2003) oferece uma solução para o casamento aproximado de registros heterogêneos que explora os atributos não compartilhados pelos pares comparados. A chave para aumentar a acurácia do processo de identificação de réplicas adotada pelo sistema PROM é verificar a coerência dos valores desses atributos combinados. Por exemplo, embora os registros $r_1 = (nome : Maria\ Silva, rua : Av.\ Ipiranga, salario : R\$\ 30.000)$ e $r_2 = (nome : Maria\ Silva, idade : 6\ anos, rua : Av.\ Ipiranga)$ compartilhem 66% dos atributos com valores idênticos, a combinação dos atributos não compartilhados indica que Maria Silva ganha 30 mil reais mensais com apenas 6 anos de idade. Apesar de possível, é improvável que a pessoa descrita pelos dois registros seja a mesma.

O processo de casamento começa calculando a similaridade entre os atributos compartilhados pelo par de registros. Se a similaridade é baixa, os registros são ditos distintos. Embora qualquer função de similaridade possa ser utilizada no início do processo de casamento, os autores não deixam claro como definir um limiar de similaridade.

Em seguida, o sistema verifica a coerência da combinação dos atributos não compartilhados pelo par de registros similares usando módulos que aplicam diferentes criadores de perfis. Um criador de perfil contém conhecimento sobre um determinado conceito e é usado para decidir se alguma restrição do conceito é violada pelo par. Por exemplo, a data de uma crítica de cinema não poderia ser menor que a data de lançamento do filme criticado. Os criadores de perfis podem ser construídos manualmente ou a partir do uso de

alguma técnica de mineração de dados aplicada sobre um conjunto de treinamento. Por exemplo, para o domínio de filmes, poderiam ser aprendidas regras de associação a partir dos dados do Internet Movie Database[2] (IMDb). Perfis podem descartar diretamente um par de registros ou atribuir uma determinada confiança na identificação de duplicatas. As confianças geradas pelos diversos perfis são combinadas utilizando-se de uma média ponderada, com pesos definidos manualmente pelo usuário.

3.1.6 GUHA et al. (2004)

A deduplicação é abordada por GUHA et al. (2004) na combinação de *rankings* de similaridade. *Rankings* individuais são gerados para cada atributo de uma relação R, de acordo com uma consulta Q. Cada *ranking* individual é composto por um conjunto de tuplas formadas por um único atributo e são ordenados em função da similaridade com o atributo correspondente da consulta Q.

É proposta uma função de fusão (*merging function*), denominada *footrule distance*, que combina os escores de cada atributo de uma tupla da relação R considerando também suas posições em cada um dos *rankings*. São selecionados os $top - k$ (CHAUDHURI; GRAVANO, 1999) registros mais relevantes. O objetivo da proposta é derivar um *ranking* final "ótimo" de tamanho k, definido pelo usuário.

3.1.7 DORNELES et al. (2009, 2004)

DORNELES et al. (2004) formalizam um conjunto de métricas de similaridade capazes de lidar com coleções de valores que ocorrem em documentos XML. Os autores definem dois tipos de métricas, um para valores atômicos (*Metrics for Atomic Values* – MAV) e outro para valores complexos (*Metrics for Complex Values* – MCV). MAV são dependentes do domínio de aplicação enquanto MCV são definidas de acordo com as características dos elementos que compõem a estrutura do valor complexo.

Qualquer função de similaridade entre *strings* pode ser utilizada como uma MAV para calcular o escore de similaridade de nodos atômicos. Já para calcular o valor de similaridade entre dois nodos complexos, são combinados diversos valores já calculados para os nodos filho destes nodos. Se os nodos filho são atômicos, MAV são combinadas. Se os nodos filho também são nodos complexos, uma MCV é usada recursivamente. São definidas métricas para comparar tuplas, subtuplas, listas ordenadas, sublistas, conjuntos e subconjuntos de valores atômicos.

Em um trabalho posterior (DORNELES et al., 2009), os autores propõem que, ao invés de atribuir um limiar de similaridade baseado nos escores retornados pelas funções de similaridade, o usuário possa especificar a precisão esperada no processo de deduplicação. Este método mapeia os escores de similaridade em valores de precisão usando um conjunto de treinamento. A escolha da precisão esperada é uma tarefa mais fácil para o especialista, entretanto a identificação de réplicas ainda requer a intervenção humana na definição da precisão.

[2] `http://www.imdb.com`. Data do acesso: 8/11/2013.

3.2 Trabalhos independentes da definição de limiares de similaridade

Outros trabalhos têm proposto métodos para solucionar o problema da deduplicação baseados em técnicas de aprendizado de máquina, principalmente utilizando aprendizado supervisionado. Estes métodos estimam os escores de similaridade e realizam o casamento dos registros duplicados sem a necessidade da definição de limiares.

3.2.1 VERYKIOS; ELMAGARMID; HOUSTIS (2000)

VERYKIOS; ELMAGARMID; HOUSTIS (2000) propõem uma metodologia para deduplicar registros baseada no aprendizado supervisionado que utiliza árvores de decisão. Inicialmente, os dados são pré-processados para lidar com inconsistências como valores nulos, valores em branco e *outliers*. A segunda fase é responsável pela geração de pares de registros candidatos ao casamento. É empregada uma estratégia de blocagem denominada *sorted neighborhood approach* (HERNÁNDEZ; STOLFO, 1998) que ordena os registros de acordo com os valores de um determinado atributo e constrói uma janela deslizante de tamanho constante sobre a lista ordenada. A cada iteração da janela, todos os registros que a compõem são organizados par a par e incluídos na lista de candidatos.

Na terceira fase é utilizado um algoritmo de agrupamento para rotular os pares como réplicas, possivelmente réplicas ou registros distintos. Então, um modelo de classificação baseado em árvores de decisão é construído sobre os pares rotulados. Por fim, as regras geradas pela árvore são podadas e reduzidas a um pequeno conjunto com alta confiança, melhorando a acurácia da predição. O conjunto final de regras é utilizado para identificar os registros duplicados no conjunto de teste.

O estudo de caso reportado pelos autores mostra a usabilidade da metodologia proposta combinando *AutoClass* (CHEESEMAN et al., 1988) como algoritmo de agrupamento e *ID3* (QUINLAN, 1986) como classificador baseado em árvores.

3.2.2 *Active Atlas* (TEJADA; KNOBLOCK; MINTON, 2001)

O sistema *Active Atlas* (TEJADA; KNOBLOCK; MINTON, 2001) realiza o mapeamento entre registros oriundos de diferentes fontes de dados. As regras de mapeamentos de atributos são especificadas com base em um processo de treinamento que utiliza árvores de decisão e aprendizado ativo (COHN; ATLAS; LADNER, 1994). Os atributos são comparados utilizando funções que identificam *substrings*, acrônimos e abreviações. O resultado das funções é combinado por uma modificação da métrica TF $\times$ IDF que pondera as dimensões vetoriais (BAEZA-YATES; RIBEIRO-NETO, 2011).

O usuário inicia o processo rotulando como réplicas um pequeno conjunto de exemplos que são usados para gerar um comitê de árvores de decisão utilizando o algoritmo *C4.5* (QUINLAN, 1993). Os exemplos são escolhidos aleatoriamente para compor conjuntos disjuntos de treinamento utilizados em cada árvore, as quais votam nos melhores candidatos ao mapeamento. O exemplo em que o comitê apresenta discordância máxima é apresentado ao usuário para ser rotulado manualmente. O usuário, então, avalia esse exemplo e rotula os registros como réplicas ou não.

O processo se repete até que o comitê convirja para a mesma árvore de decisão ou um determinado número de exemplos seja rotulado. Ao final do aprendizado, o voto da maioria (Seção 2.3.1) é utilizado para rotular o restante das instâncias.

3.2.3 *Constrained Cascade Generalization* (CCG) (ZHAO; RAM, 2008)

Métodos de deduplicação que aprendem regras de casamento aproximado utilizando árvores de decisão (VERYKIOS; ELMAGARMID; HOUSTIS, 2000; TEJADA; KNOBLOCK; MINTON, 2001) sofrem uma restrição nos limites de decisão imposta pelas árvores. Estas regras são definidas por áreas no espaço de características delimitadas por hiperplanos ortogonais aos eixos. A Figura 3.2 apresenta um exemplo com o classificador *C4.5*, evidenciando os limites de decisão em um espaço formado pelos atributos título e autor. Além disso, são destacados com circunferências os erros de classificação impostos pela restrição da ortogonalidade dos planos.

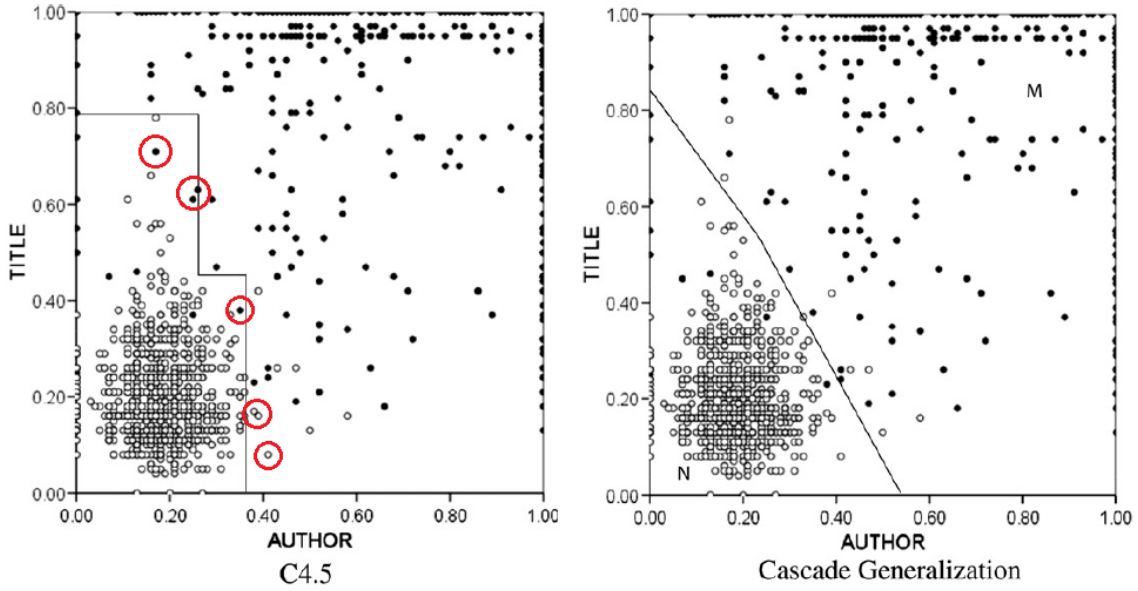

Figura 3.2: Limites de decisão considerando dois atributos gerados pelo classificador *C4.5* e pelo método *constrained cascade generalization*, evidenciando erros de classificação solucionados pelo método. Adaptado de (ZHAO, 2008).

ZHAO; RAM (2004) propõem um método denominado *Constrained Cascade Generalization* (CCG) que combina outras técnicas de classificação de forma a aliviar o viés representacional das árvores de decisão. Esse método potencialmente melhora a acurácia da classificação, como mostra o exemplo da Figura 3.2 que evita determinados erros de classificação impostos pela ortogonalidade das regras geradas pelo *C4.5*.

Em trabalho posterior (ZHAO; RAM, 2008), os autores propõem o uso deste método de classificação para identificar casamentos entre entidades oriundas de fontes heterogêneas de dados. Os autores avaliam empiricamente o método a partir de uma série de experimentos utilizando dados reais de três diferentes domínios. Os dados foram extraídos de um sistema de reservas de voos, dos catálogos de duas livrarias online e de uma base de dados de bens departamentais. Os resultados da avaliação mostram que o método proposto combinando o algoritmo *C4.5* com regressão logística (HOSMER; LEMESHOW, 2000) fornece um pequeno acréscimo, mas estatisticamente significativo, na acurácia do casamento aproximado quando comparado aos dois algoritmos isolados.

3.2.4 COHEN; RICHMAN (2002)

COHEN; RICHMAN (2002) propõem uma técnica escalável e adaptativa para agrupar registros duplicados. A ideia básica é utilizar uma métrica de comparação com baixo custo computacional para agrupar registros em grupos sobrepostos, ou seja, com alguns elementos em comum, denominados *canopies* (MCCALLUM; NIGAM; UNGAR, 2000). Dessa forma, os pares candidatos ao casamento são drasticamente reduzidos utilizando uma estratégia de blocagem baseada na métrica TF×IDF (BAEZA-YATES; RIBEIRO-NETO, 2011) como *canopy* para múltiplas funções de similaridade de *strings*.

Os registros candidatos ao casamento são organizados como vértices de um grafo em que as arestas representam o fator de confiança na predição do par de vértices ser duplicado, o qual é estimado por um classificador baseado na entropia máxima (NIGAM; LAFFERTY; MCCALLUM, 1999). Os grupos são gerados a partir de um algoritmo guloso que considera inicialmente cada vértice um grupo único e a cada iteração funde os grupos mais próximos até que restem apenas um determinado número de grupos passado como parâmetro. Todos os elementos do mesmo grupo são identificados como réplicas de um mesmo registro.

3.2.5 MARLIN (BILENKO; MOONEY, 2003)

O sistema MARLIN (BILENKO; MOONEY, 2003) explora um *framework* para identificar registros duplicados usando métricas de similaridade de *strings* adaptativas. Estas métricas adaptam-se a cada atributo dos registros, de acordo com o domínio de valores.

Os autores definem duas métricas de similaridade textual. A primeira utiliza o algoritmo *Expectation-Maximization* (EM) para estimar os parâmetros de um modelo baseado na distância de edição. A segunda métrica usa o algoritmo de classificação *Support Vector Machines* – SVM (BOSER; GUYON; VAPNIK, 1992) aplicado à representação vetorial dos atributos.

Depois de estimadas para cada atributo, as funções de similaridade são aplicadas sobre os pares candidatos ao casamento, usando blocagem baseada na função *Jaccard* (COHEN; RAVIKUMAR; FIENBERG, 2003). Então, os escores calculados são combinados por outro classificador SVM que finalmente identifica os registros duplicados.

3.2.6 *Genetic Programming* (GP) (CARVALHO et al., 2012, 2008, 2006)

CARVALHO et al. (2006) aplicam programação genética (*Genetic Programming – GP)* para gerar automaticamente funções de similaridade que identifiquem registros duplicados em um determinado repositório. As funções são representadas por uma expressão matemática que combina e pondera os atributos que formam um registro.

Uma expressão é implementada como uma árvore binária cujos nós folha são atributos e os nós intermediários são operações matemáticas. A cada geração de indivíduos, um novo conjunto de árvores é gerado a partir do cruzamento das árvores da geração anterior. Além de herdar características, um fator de mutação garante um determinado nível de diversidade na população. Os experimentos realizados pelos autores mostram que as funções geradas são mais efetivas para identificar duplicatas que o método estatístico proposto por FELLEGI; SUNTER (1969). Além disso, elas utilizam um conjunto menor de atributos para comparação.

A abordagem baseada em programação genética é estendida em trabalho posterior (CARVALHO et al., 2008) de forma a ser independente do método de FELLEGI; SUNTER (1969). As funções de similaridade geradas são baseadas na combinação de múl-

tiplos pares *atributo / função de similaridade*, as quais são computacionalmente menos exigentes que as propostas por outras abordagens, uma vez que potencialmente usam menos evidência. Os experimentos mostram um acréscimo de 6.5% na qualidade da deduplicação quando comparado ao método baseado em SVM utilizado pelo sistema MARLIN (BILENKO; MOONEY, 2003).

Um terceiro artigo relata um estudo que descreve as principais propriedades da programação genética e aborda detalhes sobre a parametrização do algoritmo (CARVALHO et al., 2008). Os experimentos apresentados mostram que a escolha de alguns parâmetros pode melhorar o resultado da deduplicação em até 30%.

Recentemente, CARVALHO et al. (2012) generalizam os resultados do método proposto mostrando que programação genética também é capaz de encontrar funções de deduplicação efetivas, mesmo quando as funções de similaridade adequadas para cada atributo não são conhecidas de antemão. Esta característica é extremamente útil para o usuário porque ele não precisa se preocupar com a seleção de funções e análise de domínio dos atributos. Além disso, os autores demonstram que o método proposto adapta a função de deduplicação de acordo com os limiares de similaridade escolhidos para classificar um par como réplica ou registros distintos. Essa característica libera o usuário do ônus causado pela sintonia fina desses limiares.

3.2.7 *Active* GP (FREITAS et al., 2010)

A maioria dos algoritmos de programação genética lidam com o problema de classificação seguindo uma abordagem supervisionada, em que todas as instâncias do conjunto de treinamento estão rotuladas com uma determinada classe de dados. FREITAS et al. (2010) propõem um método de deduplicação baseado em programação genética semi-supervisionada, apropriado para os casos em que o esforço humano para rotular a base de treinamento é muito alto. Este método utiliza aprendizagem ativa (COHN; ATLAS; LADNER, 1994), na qual um comitê de funções de deduplicação multiatributo vota para classificar pares como réplicas ou registros distintos.

O método inicia por uma etapa de blocagem em que são selecionados os pares candidatos ao casamento. É aplicada uma função de similaridade sobre esses pares, que são ordenados em relação ao escore retornado pela função. São selecionados como amostra os k primeiros e últimos pares do *ranking*, que são enviados para avaliação manual. Essas instância rotuladas pelo usuário compõem o conjunto inicial de treinamento.

A cada geração, é estimado um conjunto de funções de deduplicação (indivíduos). Essas funções são formadas por combinações de múltiplos pares *atributo / função de similaridade* e implementadas como árvores, da mesma forma que no trabalho proposto por CARVALHO et al. (2008). Cada função estimada avalia as instâncias de treinamento e as melhores funções são escolhidas para compor um comitê baseado na estratégia de combinação voto da maioria (*majority voting*) (Seção 2.3.1), responsável por rotular todas as instâncias, previamente rotuladas ou não. Quando o número de votos não é suficiente para predizer a classe do par de registros, o usuário é questionado para solucionar o conflito. O processo continua com uma etapa de reforço, em que o resultado do aprendizado ativo é considerado.

O resultado dos experimentos apresentados mostram que, quando comparado ao método proposto por CARVALHO et al. (2006), a qualidade da deduplicação se mantém, reduzindo consideravelmente o número de instâncias de treinamento rotuladas.

3.2.8 FS-Dedup (DAL BIANCO et al., 2013)

O desempenho e a qualidade do processo de deduplicação de grandes volumes de dados dependem de usuários especialistas para configurar as fases mais importantes no processo: a estratégia de blocagem e o algoritmo de classificação. DAL BIANCO et al. (2013) propõem um arcabouço chamado *FS-Dedup* que auxilia a sintonia do processo com um esforço reduzido do usuário.

FS-Dedup explora algoritmos de deduplicação baseada em assinatura. Esses algoritmos são caracterizados pela alta eficiência e escalabilidade em grandes conjuntos de dados mas exigem muita intervenção de um usuário especialista para configurar diversos parâmetros. A abordagem proposta exige que o usuário rotule apenas um pequeno conjunto de pares de registros para sintonizar os parâmetros automaticamente.

A partir dos pares rotulados é identificada a região crítica onde pares duplicados e distintos retornam valores de similaridade muito próximos. Os pares desta região são utilizados para configurar modelos de classificação baseados no algoritmo *SVM* e na função *n-gram*.

Os experimentos realizados sobre conjuntos de dados reais e sintéticos contendo milhões de registros mostram que a proposta atinge a qualidade máxima da técnica de deduplicação baseada em assinatura com um esforço manual do usuário bastante reduzido. Além disso, *FS-Dedup* exige até 7 vezes menos pares rotulados no conjunto inicial do que a técnica de deduplicação baseada em aprendizado ativo proposta por BELLARE et al. (2012).

3.3 Combinando deduplicadores

Esta seção destaca alguns trabalhos relacionados que combinam diferentes métodos para identificar registros duplicados com o objetivo de aumentar a qualidade do processo de deduplicação.

3.3.1 ZHAO; RAM (2005)

ZHAO; RAM (2005) aplicam vários algoritmos de classificação para determinar se dois registros provenientes de fontes de dados distintas representam a mesma entidade do mundo real. Os autores reportam resultados de experimentos que demonstram melhoras significativas na classificação quando estes classificadores são combinados utilizando diversas estratégias.

Sistemas de classificação homogêneos, ou seja, aqueles que implementam o mesmo algoritmo de aprendizado, são compostos utilizando três estratégias diferentes: um comitê de validação cruzada, *bootstrap aggregating – bagging* (BREIMAN, 1996) e *boosting* (FREUND; SCHAPIRE, 1996). Essas estratégias manipulam os exemplos de treinamento com o objetivo de melhorar a acurácia do sistema de classificação.

Na validação cruzada, as instâncias de treinamento são divididas em 10 conjuntos disjuntos denominados *folds*. Um classificador é construído para cada subconjunto distinto formado por 9 *folds*. Já a estratégia *bagging* consiste em gerar múltiplas versões de um mesmo classificador a partir de diferentes amostras. São selecionadas aleatoriamente e com reposição n exemplos para formar cada novo conjunto de treinamento, em que n é a quantidade de instâncias do conjunto original. Os classificadores são treinados sobre cada novo conjunto. Tanto na validação cruzada quanto na estratégia *bagging*, as predições dos classificadores aprendidos são combinadas pelo voto da maioria. Já na estratégia

boosting, a decisão final de classificação é dada por um esquema de votação em que os classificadores são ponderados de acordo com sua acurácia. Cada exemplo de treinamento possui um peso. Os classificadores são treinados sequencialmente de modo que a cada iteração um novo classificador tenta rotular corretamente os erros de classificação dos classificadores prévios, dando um peso maior a estas instâncias.

Para combinar classificadores heterogêneos, ou seja, aqueles que implementam algoritmos de aprendizado diferentes, são utilizadas as estratégias de aprendizado em cascata (*cascading*) (GAMA; BRAZDIL, 2000) e empilhamento (*stacking*) (WOLPERT, 1992), detalhado na Seção 2.3.2. Essas estratégias mantêm o mesmo conjunto de instâncias por classificador, manipulando o vetor de características.

No aprendizado em cascata, as predições de um determinado classificador em cada classe de dados são utilizadas como novas características artificialmente agregadas aos dados originais. Esse novo conjunto de treinamento é utilizado por outro classificador num processo incremental em que o conhecimento adquirido pelos classificadores de níveis anteriores é utilizado no aprendizado do nível atual. Essa estratégia difere do empilhamento porque os classificadores são agregados a cada nível e a ordem em que são aplicados modifica o resultado do sistema de classificação. No empilhamento, todos os classificadores são estimados sobre os mesmos dados de entrada e apenas um algoritmo é utilizado no segundo nível para combinar o resultado dos classificadores.

Os autores apresentam os resultados da identificação de registros duplicados em um provedor de aplicações para a indústria aérea. Cada companhia mantém um banco de dados individual que descreve as reservas de passagens. O objetivo do experimento reportado foi identificar passageiros replicados em uma única companhia aérea ou entre diversas companhias. Foram utilizados como classificadores base os algoritmos:

- *1-rule* (HOLTE, 1993);
- regressão logística (HOSMER; LEMESHOW, 2004);
- classificação por regressão logística linear (LANDWEHR; HALL; FRANK, 2005);
- *C4.5* (QUINLAN, 1993);
- *Naïve Bayes* (JOHN; LANGLEY, 1995);
- *Multilayer Perceptron* (HAYKIN, 2007);
- *Decision Table* (KOHAVI, 1995);
- *k-NN* (AHA; KIBLER; ALBERT, 1991).

Esses classificadores foram aplicados sobre um espaço de características formado pelo resultado de um conjunto de funções aplicadas sobre cada par de registros. Por exemplo, o primeiro e último nomes dos passageiros foram comparados utilizando uma composição de distância de edição (LEVENSHTEIN, 1966), busca por *substring* e a função *Soundex* (ZOBEL; DART, 1996). Outros atributos como cidade, estado, números de telefone, pontos de embarque e desembarque foram comparados apenas por igualdade.

Entre os resultados apresentados, a cascata de regressão logística e *C4.5* atingiu a melhor acurácia (99,807%), sendo estatisticamente superior aos resultados apresentados individualmente pelos dois classificadores. A estratégia *bagging* melhorou significativamente apenas o algoritmo *C4.5*, obtendo resultados equivalentes para todos os outros

algoritmos. Já a estratégia *boosting*, melhorou os resultados de alguns classificadores e piorou os de outros. Por fim, a estratégia de empilhamento melhorou significativamente os resultados da classificação quando comparada a 5 dos 7 classificadores combinados.

3.3.2 CHEN; KALASHNIKOV; MEHROTRA (2009)

CHEN; KALASHNIKOV; MEHROTRA (2009) propõem um *framework* para combinar múltiplos sistemas de deduplicação utilizando duas abordagens distintas baseadas em aprendizado supervisionado que aprendem um mapeamento entre as decisões de agrupamento dos sistemas e características do contexto de cada deduplicador.

A deduplicação é definida formalmente como um problema de agrupamento. Dado um conjunto de referências para objetos do mundo real, o objetivo é agrupar as referências que descrevem o mesmo objeto em grupos não sobrepostos. Cada grupo resultante deve representar um objeto precisa e completamente. Todos as referências que pertencem a um determinado grupo são denominadas correferências. O agrupamento é representado por um grafo de referências conectadas por relacionamentos de correferência. Dessa forma, um algoritmo de deduplicação tem como objetivo particionar as arestas deste grafo.

Os sistemas de deduplicação combinados são vistos como caixas pretas, ou seja, os detalhes internos dos algoritmos não são conhecidos. Cada sistema fornece um conjunto de grupos diferente, os quais são combinados gerando um único agrupamento final. A estratégia de blocagem utilizada forma blocos com todos os vértices em que pelo menos um sistema de deduplicação tenha decidido que os registros referenciam o mesmo objeto.

Para cada par de referências candidato ao casamento, é definido um vetor de características de contexto extraídas dos grafos de correferências. Elas são calculadas para cada sistema de deduplicação a ser combinado. A primeira característica é baseada no número de grupos formados em cada bloco. A segunda utiliza o grau de saída dos vértices que compõem o par. Esses valores são agregados e normalizados no intervalo $[0, 1]$.

A inovação do método proposto pelos autores é o aprendizado baseado nessas características. São propostas duas abordagens de aprendizado supervisionado. A primeira é baseada no treinamento de um classificador a partir das decisões e características de contexto e pode ser visto como uma extensão do método baseado em comitê proposto por ZHAO; RAM (2005). A segunda abordagem aprende, para cada deduplicador, o efeito do contexto na qualidade da deduplicação. Então, esse conhecimento e as decisões dos deduplicadores são usados para produzir um agrupamento final.

Os experimentos reportados adotam os algoritmos SMO (PLATT, 1999) e regressão logística (HOSMER; LEMESHOW, 2004) e comparam o método proposto com outras estratégias de agregação de agrupamentos em diferentes cenários e bases de dados. O método melhora significativamente os resultados em todos os experimentos apresentados. Além disso, mostra uma melhora consistente na medida que o número de deduplicadores combinados aumenta, enquanto os resultados de outras estratégias flutuam.

Quando comparada a proposta de ZHAO; RAM (2005), o *framework* tem a vantagem de não precisar conhecer nenhum detalhe dos algoritmos empregados nos sistemas de deduplicação tampouco das características que compõem os registros comparados.

3.3.3 WHANG; GARCIA-MOLINA (2013, 2012)

WHANG; GARCIA-MOLINA (2012) atacam o problema da deduplicação conjunta. Por exemplo, verificar que os registros compostos pelas tuplas (*name*: Knowlege and Data Engineering, *year*: 2012) e (*title*: IEEE TKDE, *volume*: 24, *number*: 1) representam o mesmo veículo de publicação pode ajudar na deduplicação de registros que representem

artigos científicos publicados neste periódico. Consequentemente, os artigos identificados como réplicas podem conter informação para auxiliar no casamento de outros tipos de registros como nomes de autores ou ainda retroalimentar a deduplicação dos veículos de publicação.

Os autores especificam um arcabouço modular e flexível em que múltiplos algoritmos de deduplicação podem ser sintonizados para tipos particulares de registros. São definidos planos de execução lógica destes algoritmos que respeitam determinadas restrições de recursos como quantidade de memória e número de núcleos de CPU. A abordagem proposta encontra o plano de execução mais eficiente dado uma quantidade limitada de recursos.

O comportamento e a escalabilidade da abordagem proposta são avaliados a partir de um conjunto de experimentos realizados sobre bases de dados sintéticas e reais. É utilizada a estratégia de blocagem vizinhança ordenada, a função de similaridade *Jaro* e o algoritmo de deduplicação *R-Swoosh* (BENJELLOUN et al., 2009).

Em um trabalho posterior (WHANG; GARCIA-MOLINA, 2013), os autores estendem a proposta adicionando técnicas de otimização para planos de execução eficientes, técnicas de treinamento para algoritmos de deduplicação conjunta e um conjunto de novos experimentos.

3.4 Análise dos trabalhos relacionados

Esta seção apresenta uma análise comparativa dos métodos de deduplicação de registros apresentados neste capítulo. Primeiramente, são apresentados os critérios utilizados para comparação. Em seguida, é feita a análise de cada critério, que é resumida na Tabela 3.1.

Os seguintes critérios foram utilizados para realizar a comparação dos trabalhos relacionados:

- limiar (Lim) - dependência da definição de limiares de similaridade pelo usuário;
- aprendizagem - tarefa de aprendizagem de máquina;
- blocagem - estratégia de blocagem utilizada para gerar os pares de registros candidatos ao casamento;
- casamento - algoritmo que opera sobre os escores gerados pelas funções de similaridade. Alguns trabalhos propõem novas métricas ou funções de composição, enquanto outros propõem o uso de algoritmos de mineração de dados;
- funções - funções de similaridade usadas para comparar os atributos de cada par de registros;
- T - indica se todos os atributos que compõem os registros são utilizados no processo de casamento. Quando não há valor, apenas os atributos em comum aos pares são comparados;
- área - indica em que contexto ou área de aplicação o trabalho é proposto;
- contribuição - cita a principal contribuição do trabalho.

Tabela 3.1: Comparativo entre os trabalhos relacionados.

Trabalho	Lim	Aprendizagem	Blocagem	Casamento	Funções	T	Área	Contribuição
FELLEGI; SUNTER (1969)	•			*linkage rule*			casamento de registros	formalização do problema
CiteSeer (LAWRENCE; GILES; BOLLACKER, 1999)	•			função de composição	distância de edição e frequência de palavras		bibliotecas digitais	índice de citações autônomo
CHAUDHURI et al. (2003)	•	agrupamento	*q-grams*	*fuzzy top-k*	IDF	•	*data cleaning*	modelo probabilístico
CARVALHO; SILVA (2003)	•			modelo vetorial ponderado	TF × IDF		deduplicação de XML	modelo vetorial
PROM (DOAN et al., 2003)	•	associação		média ponderada da confiança dos perfis	quaisquer	•	integração de dados	atributos heterogêneos
GUHA et al. (2004)	•			*ranking footrule distance*	quaisquer		*data cleaning*	combinação de *rankings*
DORNELES et al. (2004)	•			MVC	MVA	•	consultas aproximadas	funções de composição
DORNELES et al. (2009)	•	mapeamento		média dos escores árvores de decisão	*Jaccard*, *Jaro-Winkler*, *q-gram* e distância de edição		integração de dados	escore ajustado
VERYKIOS; ELMAGARMID; HOUSTIS (2000)		classificação	vizinhança ordenada	árvores de decisão	distância de edição		*data cleaning*	poda de regras
Active Atlas (TEJADA; KNOBLOCK; MINTON, 2001)		classificação	*string canopies*	comitê de árvores	TWIDF		integração de dados	aprendizado ativo
CCG (ZHAO; RAM, 2008)		classificação		cascata de árvore e regressão logística	*Soundex* e distância de edição		integração de dados	cascata de classificadores
COHEN; RICHMAN (2002)		agrupamento	TF × IDF	entropia máxima	*Jaccard*, *2-gram* e distância de edição		integração de dados	agrupamento hierárquico
MARLIN (BILENKO; MOONEY, 2003)		classificação	Jaccard	EM e SVM	distância de edição e TF × IDF		integração de dados	similaridade adaptativa
GP (CARVALHO et al., 2012, 2006)		classificação		programação genética (melhor função)	*n-gram*, *Jaccard*, *Soft* TF × IDF *Jaro-Winkler* e distância de edição	•	integração de dados	similaridade adaptativa
Active GP (FREITAS et al., 2010)		classificação		programação genética (comitê de funções)	*n-gram*, *Jaccard*, *Soft* TF × IDF, *Jaro-Winkler* e distância de edição	•	integração de dados	aprendizado ativo
FS-Dedup (DAL BIANCO et al., 2013)		mapeamento e classificação	filtragem de prefixo	SVM e *n-gram*	*Jaccard*	•	integração de dados	redução do esforço de rotulação
BORGES et al. (2011)	•		viz. ordenada e NAMEMATCH	METADATAMATCH	NAMEMATCH e distância de edição		bibliotecas digitais	casos de falha das funções
BORGES et al. (2011a,b)		classificação		múltiplos algoritmos	NAMEMATCH e distância de edição		bibliotecas digitais	avaliação de modelos
ZHAO; RAM (2005)		classificação		*bagging*, *boosting*, empilhamento e *cascading*	*substring*, Soundex e distância de edição		integração de dados	combinação de classificadores
CHEN; KALASHNIKOV; MEHROTRA (2009)		classificação	1 voto	SMO e regressão logística	NA	NA	integração de dados	combinação de deduplicadores
WHANG; GARCIA-MOLINA (2013, 2012)	•	agrupamento	viz. ordenada	plano de execução	Jaro e R-Swoosh		integração de dados	deduplicação conjunta
Tese		**classificação**	**viz. ordenada e NAMEMATCH**	**empilhamento**	**NAMEMATCH e distância de edição**		**bibliotecas digitais**	**combinação de classificadores**

Legenda: • = sim, NA = não aplicável

Os primeiros sete trabalhos apresentados na Tabela 3.1 são dependentes da definição de um ou mais limiares de similaridade. Embora alguns desses métodos de deduplicação utilizem aprendizagem de máquina no algoritmo de casamento, o usuário ainda necessita intervir no processo de deduplicação com um parâmetro que delimita e define quais pares de registros são considerados duplicatas. No sistema PROM, as regras de associação definem os perfis utilizados no casamento de registros. Mas estes perfis são aplicados apenas aos pares considerados suficientemente similares a partir de um limiar definido pelo usuário. CHAUDHURI et al. (2003) exigem o tamanho k do *ranking* de registros similares. DORNELES et al. (2009) precisam da precisão esperada no processo de deduplicação. Já os métodos baseados em programação genética (CARVALHO et al., 2012; FREITAS et al., 2010) bem como *FS-Dedup*, apesar de utilizarem limiares de similaridade, são capazes de adaptar as funções de casamento durante o aprendizado. Portanto, estes métodos são considerados independentes da definição de limiares, assim como o método proposto nesta tese.

Alguns trabalhos apresentam uma estratégia de blocagem para redução do número de pares candidatos ao casamento. VERYKIOS; ELMAGARMID; HOUSTIS (2000) bem como WHANG; GARCIA-MOLINA (2012) ordenam os registros de acordo com um atributo chave e comparam apenas aqueles que estiverem dentro de uma vizinhança determinada. DAL BIANCO et al. (2013) usam chaves formadas por fragmentos de atributos. Esta estratégia é conhecida na literatura como filtragem baseada em prefixo. Trabalhos recentes (KIM; LEE, 2012; WANG; LI; FENG, 2012) propõem alternativas às estratégias tradicionais de filtragem no contexto de junções e consultas por similaridade. CHAUDHURI et al. (2003) propõem o uso de *q-grams*, que é uma função de similaridade indexável. O sistema *Active Atlas* aplica funções de *string* mais baratas computacionalmente como *canopies* de funções mais caras. A mesma estratégia é utilizada por COHEN; RICHMAN (2002) e pelo sistema MARLIN utilizando as funções TF$\times$IDF e *Jaccard*, respectivamente. CHEN; KALASHNIKOV; MEHROTRA (2009) selecionam os pares de registros em que pelo menos um deduplicador que compõe o sistema os considera registros duplicados. Os demais trabalhos não relatam o uso de qualquer técnica de blocagem. Utilizando suposições simples sobre os metadados que representam o ano de publicação e a autoria de citações bibliográficas, esta tese adapta a estratégia de blocagem vizinhança ordenada e utiliza a função NAMEMATCH como *canopy* para a distância de edição. A estratégia proposta foi publicada no artigo (BORGES et al., 2011).

A maioria dos trabalhos analisados utiliza alguma técnica de aprendizagem de máquina para identificar registros duplicados. A tarefa de aprendizagem mais utilizada é a classificação, entretanto os tipos de classificadores variam consideravelmente. VERYKIOS; ELMAGARMID; HOUSTIS (2000) e TEJADA; KNOBLOCK; MINTON (2001) utilizam árvores de decisão. ZHAO (2008) propõe a cascata de classificadores baseados em árvore e regressão logística. SVM são utilizados pelo sistema MARLIN, *FS-Dedup* e por CHEN; KALASHNIKOV; MEHROTRA (2009). CARVALHO et al. (2012); FREITAS et al. (2010) são baseados em programação genética. Por fim, ZHAO; RAM (2005) combinam diversos outros classificadores a partir do uso das estratégias *bagging*, *boosting*, *cascading* e empilhamento.

Entre as funções de similaridade de *string* mais utilizadas pelos trabalhos relacionados estão *Jaccard* e as baseadas no índice TF $\times$ IDF e na proporção de *n-grams* em comum, principalmente porque possibilitam o pré-processamento e a indexação dos termos. Muitos também utilizam a *Jaro-Winkler*, distância de edição e *Soundex* que são efetivas para a maioria dos domínios. Esta tese propõe o algoritmo NAMEMATCH que

identifica variações na representação dos nomes dos autores utilizando a função INISIM. O sistema PROM e os métodos propostos por CHAUDHURI et al. (2003); DORNELES et al. (2004); CARVALHO et al. (2006); FREITAS et al. (2010); DAL BIANCO et al. (2013) aplicam as funções de similaridade sobre todos os atributos dos objetos comparados para realizar o casamento. Esta característica pode fazer com que objetos que diferem muito no número de atributos não sejam identificados como réplicas. Entretanto, esses métodos têm a vantagem de não depender do usuário ou de alguma técnica automatizada para identificar os mapeamentos entre diferentes esquemas.

Apesar dos trabalhos serem desenvolvidos com o mesmo objetivo final, os métodos são aplicados em diferentes contextos: *data cleaning*, deduplicação de documentos XML, integração de dados de fontes distintas e consultas aproximadas. A maioria dos trabalhos relacionados apresentam soluções gerais para a deduplicação de registros, mas não tratam especificamente da comparação de nomes próprios, os quais, como discutido na introdução desta tese, são evidência essencial para identificar registros bibliográficos. Somente o sistema CiteSeer (LAWRENCE; GILES; BOLLACKER, 1999) foi proposto para identificar metadados bibliográficos duplicados em bibliotecas digitais e utiliza uma abordagem específica para identificar variações nos nomes dos autores.

Identificar variações nos nomes dos autores presentes em citações bibliográficas pode ser considerado um subproblema da deduplicação. Algumas funções de similaridade foram propostas especificamente para comparar nomes próprios. Por exemplo, a função *Guth* (GUTH, 1976) suporta apenas pequenas variações de grafia. A função *Acronyms* (LIMA, 2002) suporta abreviações de nomes, mas é limitada porque utiliza apenas letras maiúsculas para produzir um acrônimo, sem tratar a omissão de alguns nomes. Por exemplo, o acrônimo de John A. B. C. Smith é JABCS enquanto o acrônimo de John A. Smith é JAS. Acrônimos distintos não seriam considerados variações do mesmo nome. Já a função *comparação por fragmentos* (ou *Fragments*) (COTA; GONÇALVES; LAENDER, 2007) suporta, além de abreviações, as inversões de nomes do meio, mas inversões do último nome são detectadas apenas na presença de um caractere que indique essa inversão, como é o caso da vírgula. Além disso, a complexidade do algoritmo é quadrática tanto em termos do número quanto do tamanho dos fragmentos de texto. *Jaro-Winkler* (WINKLER, 1990) é baseada no número e na ordem dos caracteres em comum e prioriza similaridades no início dos nomes. Esta característica permite identificar corretamente erros de grafia. Entretanto, esta função foi desenvolvida para ser aplicada individualmente sobre o primeiro e o último nomes, portanto não considera inversões ou abreviações. A função proposta *IniSim* é capaz de solucionar os problemas apresentados pelas demais funções usando um algoritmo com complexidade linear. Uma vantagem comum a todas estas funções de similaridade é que elas podem ser consideradas independentes de idioma. A Tabela 3.2 sumariza as características de cada função.

Tabela 3.2: Características das funções de similaridade entre nomes próprios.

Características	***Guth***	***Acronyms***	***Fragments***	***Jaro-Winkler***	**INISIM**
variações de grafia	limitada	•	•	•	•
abreviações		limitada	•		•
inversões de nomes			parcial		•
complexidade	linear	linear	quadrática	quadrática	linear
independência do idioma	•	•	•	•	•

KÖPCKE; RAHM (2010) apresentam um estudo analítico dos resultados publicados por um conjunto de *frameworks* para deduplicação de dados e os compara em função de uma série de características. Os autores concluem que as avaliações dos *frameworks* usam diversas metodologias e métricas tornando difícil a comparação de qualidade da deduplicação, eficiência computacional e efetividade da blocagem. Além disso, alguns testes de escalabilidade deixam a desejar porque utilizam bases de dados relativamente pequenas.

Em um segundo artigo, KÖPCKE; THOR; RAHM (2010a) apresentam uma avaliação de implementações de diferentes deduplicadores. Os experimentos realizados comparam o tempo de processamento e a qualidade da deduplicação de quatro bases com dados reais de referências bibliográficas extraídas de bibliotecas digitais e produtos de comércio eletrônico. Entre os deduplicadores que dependem da definição de limiares de similaridade, foram avaliados FellegiSunter (FELLEGI; SUNTER, 1969), PPJoin+ (XIAO et al., 2011) e um sistema comercial cuja licença não permite a divulgação. Também foram considerados os métodos baseados em *SVM* implementados pelos sistemas FEBRL (CHRISTEN, 2008a,b) e MARLIN. Apesar de serem independentes da definição de limiares, foi adotada uma estratégia que seleciona aleatoriamente entre 40 e 60% de pares não replicados com pelo menos 40% de similaridade entre si de acordo com a função TF $\times$ IDF. Dessa forma, a amostra de exemplos de treinamento mantém certo balanceamento e despreza pares obviamente não replicados. Esses parâmetros foram definidos com base nos resultados de uma série de outros experimentos relatados em (KÖPCKE; THOR; RAHM, 2010b).

Os autores apontam que os métodos baseados em SVM superam os demais na maioria das combinações de funções de similaridade e atributos. Enquanto a deduplicação de referências bibliográficas é resolvida com apenas 50 exemplos de treinamento, o resultado do casamento de produtos não é satisfatório, mesmo com 500 exemplos rotulados no treino. Além disso, não foi possível estimar os modelos de classificação sem a etapa de blocagem porque o tempo de execução seria proibitivo. A melhor escalabilidade foi observada utilizando PPJoin+ configurado com um único atributo e as funções *Cosine* ou *Jaccard*, que são indexáveis. Este método pode ser aplicado sobre todos os pares (produto cartesiano) na mesma escala de tempo dos pares candidatos selecionados pela estratégia de blocagem.

Neste capítulo, foram apresentados trabalhos relacionados a esta tese que especificam novos métodos de deduplicação de registros. Os trabalhos estudados combinam um conjunto de funções de similaridade de *string* de propósito geral em uma função de composição ou em um algoritmo de classificação. Essas funções não resolvem os problemas da variação na representação de nomes próprios, os quais são essenciais para identificação da autoria de citações bibliográficas. Além disso, possuem custo computacional elevado. Esta tese especifica funções de similaridade que identificam compatibilidades na autoria de objetos implementadas a partir de um algoritmo com complexidade linear.

A partir da análise realizada, é possível observar que a grande maioria dos trabalhos estudados aplica uma técnica de aprendizado de máquina em particular para identificar registros duplicados. O sucesso da técnica aplicada depende muito das características da base de dados. O método proposto combina diferentes técnicas de aprendizado para aprimorar a qualidade da deduplicação a partir do conhecimento heterogêneo aprendido pelo empilhamento de múltiplos algoritmos de classificação.

ZHAO; RAM (2005) também empilham classificadores, porém os experimentos conduzidos pelo autores foram executados sobre uma base de dados balanceada em relação à classe réplica ou registros distintos. Em cenários reais, dificilmente haveria este balanceamento e o custo dos falsos positivos tende a ser menor que o dos falsos negativos, principalmente porque a quantidade de registros duplicados é pequena.

Uma visão geral de diversos outros trabalhos sobre deduplicação pode ser encontrada no livro (CHRISTEN, 2012a) e nos *surveys* (DORNELES; GONÇALVES; MELLO, 2011; ELMAGARMID; IPEIROTIS; VERYKIOS, 2007). Outras estratégias de indexação específicas para deduplicação de grande volumes de dados são abordadas por CHRISTEN (2012b). Trabalhos recentes tratam do problema de valores nulos (*missing*) na deduplicação de registros (SARIYAR; BORG; POMMERENING, 2012).

4 MÉTODO PROPOSTO

Este capítulo apresenta em detalhe o método proposto para identificar metadados bibliográficos duplicados a partir do empilhamento de classificadores. A Seção 4.1 apresenta uma visão geral do método de deduplicação e enfatiza o objetivo da tese. Nas seções seguintes, cada etapa do método proposto é detalhada.

4.1 Visão Geral

O objetivo principal desta tese é o desenvolvimento de um método efetivo e automático para identificar metadados bibliográficos duplicados, combinando o aprendizado de múltiplos classificadores supervisionados. A Figura 4.1 apresenta uma visão geral das etapas que compõem o método proposto. A partir de uma ou mais fontes de dados nas quais se deseja identificar referências duplicadas para as mesmas produções bibliográficas, (1) é extraído um conjunto de metadados descritivos e pré-processados de forma a compor um repositório (Seção 4.2). Sobre os registros do repositório (2) é aplicado um determinado esquema de seleção de exemplos aleatória baseado no cálculo amostral e em características do conjunto de dados (Seção 4.3). Os pares de exemplos selecionados compõem o conjunto de treinamento sobre o qual (3) são aplicadas as funções de similaridade que operam sobre determinados atributos. Os escores produzidos pelas funções são utilizados para treinar múltiplos modelos de classificação heterogêneos, ou seja, a partir de algoritmos de diversos tipos: baseados em árvores, regras, redes neurais artificiais, probabilísticos, etc (Seção 4.4). As predições desses classificadores (4) são empilhadas compondo um segundo conjunto de treinamento sobre o qual é aprendido um outro modelo de classificação que efetivamente é aplicado sobre os pares candidatos ao casamento (Seção 4.5). Esses pares candidatos (5) são escolhidos utilizando-se de uma estratégia de blocagem de dois níveis (Seção 4.6). O resultado da deduplicação (6) é um conjunto de pares de registros considerados duplicados, ou seja, que referenciam a mesma produção bibliográfica (Seção 4.7).

A solução proposta é baseada na hipótese de que o empilhamento de classificadores supervisionados (Seção 2.3.2) pode aumentar a qualidade da deduplicação quando comparado a escolha do melhor classificador ou a estratégias de combinação como o voto da maioria (Seção 2.3.1). Esta tese também visa demonstrar o ganho real do empilhamento frente às outras abordagens citadas.

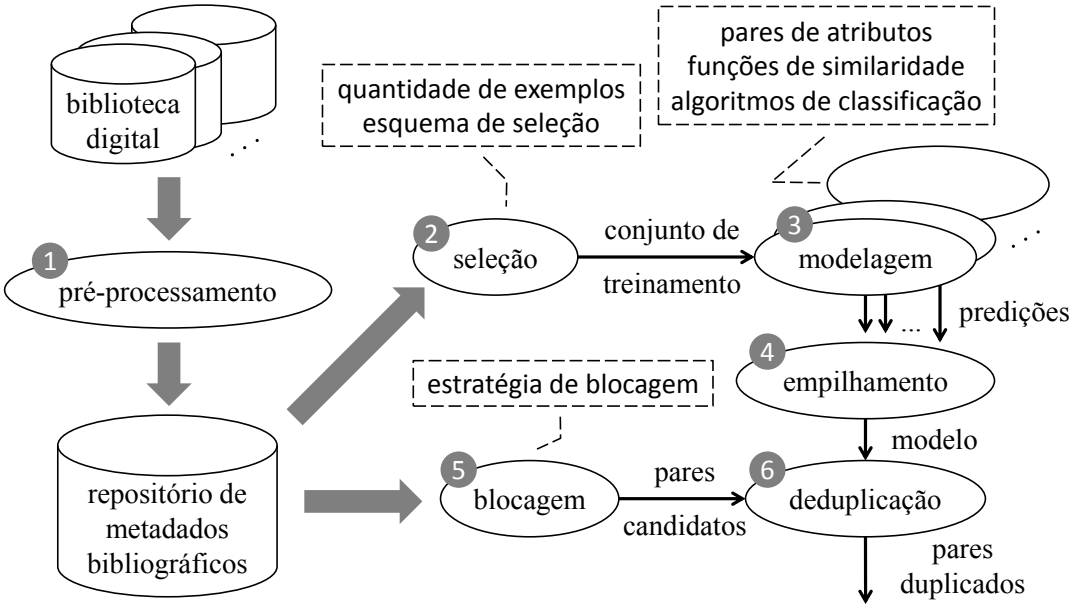

Figura 4.1: Visão geral do método de deduplicação proposto.

4.2 Pré-processamento

Padrões de metadados como Dublin Core (DCMI USAGE BOARD, 2012) e MARC 21 (LIBRARY OF CONGRESS, 2013) são representados por registros compostos por vários campos em uma estrutura plana. O método de deduplicação proposto seleciona de uma ou mais fontes os metadados existentes em diferentes tipos de publicação como livros, artigos de revista, artigos apresentados em conferências e páginas *web*. São extraídos apenas os campos que definem nomes dos autores, título e ano de publicação porque estes costumam estar presentes na grande maioria das referências bibliográficas. Além disso, eles são menos suscetíveis a ruído quando comparados a campos como veículo de publicação, números de página, entre outros. Por exemplo, tanto *TKDE* quanto *IEEE Transactions on Knowledge and Data Engineering* descrevem a mesma revista científica e uma das referências poderia omitir os números de página. Além dos campos mencionados anteriormente, é assumido que existe um campo *classe* que define a que publicação do mundo real o registro bibliográfico faz referência. Este campo é essencial para o treinamento dos classificadores supervisionados.

Sobre os registros extraídos, um conjunto de operações de pré-processamento é aplicado para limpar e normalizar o conteúdo dos metadados selecionados. A Tabela 4.1 apresenta uma lista de operações e os campos sobre os quais cada operação é aplicada. O primeiro passo é limpar todos os campos aplicando transformações de *string* usuais. O ano de publicação é transformado em um número inteiro válido no domínio. Os quatro dígitos mais à esquerda são extraídos de datas ou *timestamps* no padrão ISO, como no exemplo "**2011**-09-20 08:32:45". Além das transformações típicas nos nomes dos autores, são definidos caracteres delimitadores para serem usados nas funções de similaridade que comparam este campo. Por fim, são removidas as instâncias com ruído, como aquelas que não têm um ano de publicação válido ou pelo menos um autor. Todas as instâncias pré-processadas são agrupadas em um único repositório de metadados bibliográficos.

A maioria dos trabalhos descritos no Capítulo 3 contribuem com soluções para o problema de deduplicação de registros em geral, os quais não levam em consideração as

Tabela 4.1: Operações de pré-processamento.

Metadado	**Operações de Limpeza e Normalização**
título, ano de publicação, nomes de autores e classe	remover valores nulos remover aspas simples e duplas remover acentuação converter caracteres para letras minúsculas
ano de publicação	remover qualquer caractere não numérico se existir apenas 2 dígitos, adicionar "19" antes deles se existir mais de 4 dígitos, extrair os 4 mais a esquerda
nomes de autores	configurar o delimitador de autor (ex. ";") configurar o delimitador de inversão de nomes (ex. ",") remover números inserir espaço entre abreviaturas substituir espaços duplos por espaços simples remover espaços no início e fim remover identificadores

especificidades do domínio das bibliotecas digitais. Esta tese propõe um método para deduplicação de registros bibliográficos em que o conteúdo dos metadados que representam esses registros é comparado utilizando funções de distância ou similaridade especialmente desenvolvidas para este fim. As funções propostas, bem como os metadados descritivos selecionados, foram definidos com base nas seguintes suposições:

1. Referências duplicadas apresentam anos de publicação muito próximos. O mesmo ano ou apenas um ano de diferença é uma boa indicação porque esta diferença cobre, por exemplo, os casos de eventos que têm seus *proceedings* publicados no ano seguinte e os artigos aceitos em periódicos ainda não publicados.

2. Referências duplicadas compartilham a maioria dos autores. Devido a erros e problemas na aquisição dos dados, é bastante comum encontrar referências bibliográficas em que a lista de autores não está completa. Além disso, o método de deduplicação precisa suportar problemas de representação como variações de grafia, omissão dos nomes do meio, abreviações e inversão na ordem dos nomes.

3. Referências duplicadas apresentam títulos muito similares, mas se os autores não tiverem nomes similares, muito provavelmente elas são objetos do mundo real distintos.

4.3 Seleção

A estratégia de seleção de exemplos de treinamento proposta seleciona aleatoriamente uma amostra de pares de registros pré-processados com tamanho mínimo calculado a partir da estimativa da proporção populacional. Esta estratégia considera duas características do conjunto de dados que auxiliam na etapa de modelagem: conter pelo menos metade dos pares duplicados e manter certo balanceamento em relação à classe.

Para definir o número mínimo de pares necessários para o treinamento é realizado o cálculo amostral com base na estimativa da proporção populacional (TRIOLA et al.,

2013). É considerada uma população infinita, em que o tamanho da amostra (pares selecionados) deve ser muito menor do que 5% do tamanho da população (combinação de todos os registros). A Equação 4.1 define o tamanho mínimo n da amostra,

$$n = \frac{Z^2 \times p \times q}{E^2} \tag{4.1}$$

em que Z é o valor crítico que corresponde ao grau de confiança desejado, p é a proporção de indivíduos da classe de interesse, $q = 1 - p$ é a proporção de indivíduos de outras classes e E é a margem de erro, ou seja, a diferença máxima entre a proporção amostral e a verdadeira proporção populacional. A Tabela 4.2 apresenta os valores críticos associados a graus de confiança maiores que 90%.

Tabela 4.2: Valores críticos associados aos graus de confiança na amostra (TRIOLA et al., 2013).

Grau de Confiança (%)	**Valor Crítico Z**
90	1,645
91	1,695
92	1,751
93	1,811
94	1,881
95	1,960
96	2,054
97	2,170
98	2,328
99	2,575

Seja R um conjunto de registros bibliográficos pré-processados conforme o procedimento descrito na Seção 4.2, a função SUBSETOFPAIRS : $\{R\} \rightarrow R \times R$ é definida pelo algoritmo

SUBSETOFPAIRS(D, n)
1 $size_{Sample} \leftarrow$ MAX(COUNT$(D), n)$;
2 $Pairs \leftarrow C_2^D$;
3 $Pairs_{Dup} \leftarrow$ DUPLICATEDSELECT$(Pairs)$;
4 $Pairs_{\neg Dup} \leftarrow Pairs - Pairs_{Dup}$;
5 $Clear_{\neg Dup} =$ LOWSIMSELECT$(Pairs_{\neg Dup})$;
6 $Possible_{\neg Dup} = Pairs_{\neg Dup} - Clear_{\neg Dup}$;
7 $size_{Pairs_{Dup}} \leftarrow$ COUNT$(Pairs_{Dup})$;
8 $dup \leftarrow \frac{\text{MIN}(size_{Pairs_{Dup}}, size_{Sample})}{2}$;
9 $\neg dup \leftarrow size_{Sample} - dup$;
10 $Sample \leftarrow$ RANDOMSELECT$(Pairs_{Dup}, dup)$;
11 $Sample \leftarrow Sample \cup$ RANDOMSELECT$(Clear_{\neg Dup}, \frac{\neg dup}{2})$;
12 $Sample \leftarrow Sample \cup$ RANDOMSELECT$(Possible_{\neg Dup}, \frac{\neg dup}{2})$;
13 **return** $Sample$;

em que $D \subseteq R$ é um conjunto de registros bibliográficos pré-processados; n é o número mínimo de pares calculado anteriormente; $size_X$ é o tamanho do conjunto X, $Pairs$ contém os registros em D combinados dois a dois; $Pairs_{Dup}$ é o subconjunto dos pares duplicados, ou seja, que têm a mesma classe e referenciam a mesma publicação; $Pairs_{\neg Dup}$ são todos os pares distintos; $Clear_{\neg Dup}$ são os pares claramente distintos, ou seja, registros que apresentam baixa similaridade entre si; $Possible_{\neg Dup}$ são pares distintos mais difíceis de identificar porque possuem certa similaridade. Além disso, considere a função MAX retorna o maior valor entre os parâmetros, COUNT retorna o número de elementos em um conjunto, DUPLICATEDSELECT retorna os pares duplicados de um conjunto, LOWSIMSELECT retorna os pares com baixa similaridade, MIN retorna o menor valor entre os parâmetros e RANDOMSELECT retorna aleatoriamente uma determinada quantidade de elementos de um conjunto.

Esta estratégia de seleção tem como objetivo gerar uma amostra aleatória com uma quantidade de pares $size_{Sample}$ igual ao número de registros bibliográficos pré-processados D (linha 1) mas que respeite n. Além disso, ela deve respeitar certo balanceamento em relação à classe (par duplicado ou distinto). O algoritmo combina os registros em pares (linha 2) e os separa em dois conjuntos de acordo com a classe (linhas 3-4). Os pares distintos também são separados em dois conjuntos de acordo com a similaridade (linhas 5-6). As proporções entre os conjuntos são calculadas (linhas 7-9), selecionando-se metade dos pares duplicados $Pairs_{Dup}$ (linha 10) e o restante para completar o tamanho da amostra igualmente distribuído entre pares claramente distintos $Clear_{\neg Dup}$ (linha 11) e distintos com alguma similaridade $Possible_{\neg Dup}$ (linha 12). Note que a amostra fica igualmente balanceada em relação à classe se o tamanho da amostra é maior que o número de pares duplicados (linha 8).

No Capítulo 5, este algoritmo é empregado sobre todos os conjuntos de dados usados na avaliação do método proposto.

4.4 Modelagem

Os pares de referências selecionadas para compor o conjunto de treinamento geram novos registros com os metadados oriundos de cada referência. Sobre cada par, são aplicadas as funções de similaridade propostas e os escores retornados são adicionados como novos campos de cada registro. A Tabela 4.3 apresenta os novos campos e as funções correspondentes. São calculadas diferenças e similaridades entre os pares de anos de publicação, nomes de autores e títulos. Se as classes originais forem iguais, uma nova classe binária denominada *duplicated pair* é definida com o valor *1*. Caso contrário, o valor *0* é atribuído.

Os campos originais provenientes de cada registro (título, nomes de autores e ano de publicação) são removidos, restando apenas campos numéricos com diferentes distribuições de valores. O método proposto evita campos com cadeias de caracteres porque apenas sobre os escores retornados pelas funções serão usados para treinar os classificadores, permitindo o uso de vários tipos de algoritmos de classificação.

4.4.1 Comparando nomes de autores

Os nomes de autores são comparados utilizando-se duas funções propostas especificamente para este fim. Juntas, elas retornam a similaridade entre um par de registros bibliográficos baseada nas iniciais dos nomes. Estas funções foram propostas inicialmente

Tabela 4.3: Funções de distância ou similaridade aplicadas a cada par de referências.

Novo metadado	**Função de distância ou similaridade**
author number diff	diferença absoluta entre o número de autores
year diff	diferença absoluta entre os anos de publicação
author diff	diferença entre os autores baseada no produto entre *author sim* e o maior número de autores
author sim	similaridade entre os autores de acordo com a função NAMEMATCH (Seção 4.4.1)
title diff	distância de edição (LEVENSHTEIN, 1966) entre os títulos
title sim	similaridade entre os títulos baseada na distância *title diff* normalizada
duplicated pair	classe binária (sim/não) indicando referências duplicadas

em um artigo publicado nos anais do Simpósio Brasileiro de Banco de Dados (BORGES; GALANTE; GONÇALVES, 2008) e foram adaptadas para o método descrito nesta tese.

A função *Initials Similarity* (INISIM) identifica variações na representação dos nomes de um autor considerando erros de grafia, inversões, abreviações e omissões de nomes. Apenas as iniciais dos nomes do autor são comparadas possibilitando uma implementação eficiente desta função em um algoritmo linear.

Seja PN um conjunto de nomes próprios e $\mathbb{N}_1$ o conjunto de números naturais $\{0, 1\}$, INISIM : $\{PN \times PN\} \rightarrow \mathbb{N}_1$ calcula a distância entre as iniciais de dois nomes próprios, ou seja, nomes de autores. A função INISIM verifica se as iniciais dos nomes próprios $a, b \in PN$ potencialmente representam o nome de uma mesma pessoa. São realizadas comparações apenas entre a primeira, a segunda e a última inicial já que o primeiro e o último nome dos autores aparecem nestas posições frequentemente, mesmo se houver uma inversão de nomes. Quando as iniciais pertencem a um nome de autor compatível, ou seja, quando ambas as representações podem corresponder à mesma entidade do mundo real, a função INISIM retorna o escore de similaridade $s \in \mathbb{N}_1 | s = 1$. Caso contrário, é retornado $s = 0$. INISIM é definida pela Equação 4.2 como

$$\text{INISIM}(a, b) = \begin{cases} 1, & \text{se} \quad \begin{array}{l} (a_1 = b_1 \wedge a_m = b_n) \vee \\ (a_1 = b_2 \wedge a_m = b_1) \vee \\ (a_1 = b_1 \wedge a_2 = b_2) \vee \\ (a_1 = b_n \wedge a_2 = b_1) \end{array} \\ 0, & \text{caso contrário} \end{cases} \tag{4.2}$$

em que a, b são cadeias de caracteres formadas pelas iniciais dos nomes dos autores, a_i é o i-ésimo caractere de a, ou seja, a i-ésima inicial do nome do primeiro autor, b_i é o i-ésimo caractere de b, ou seja, a i-ésima inicial do nome do segundo autor, m e n são os tamanhos das cadeias a e b respectivamente.

Por exemplo, considere a função INITIALS que extrai as iniciais de um nome próprio.

INISIM (INITIALS (John Winston Lennon), INITIALS (Lenon, Jhon)) =
INISIM (JWL, LJ) = 1 ($a_1 = b_2 \ \wedge \ a_m = b_1$)

INISIM (INITIALS (John Winston), INITIALS (Lennon, John)) =
INISIM (JW, LJ) = 0

Perceba que INISIM lida com erros de grafia nos nomes próprios com exceção daqueles que ocorrem na primeira, segunda e última iniciais. Entretanto, estes erros dificilmente devem ocorrer. Por exemplo, nos conjuntos de dados experimentais utilizados nesta tese, não há nenhuma ocorrência de erros de grafia exatamente nestes caracteres. Outros trabalhos (CONVIS; GLICKMAN; ROSENBAUM, 1982) também confiam no fato de que geralmente as iniciais são a evidência mais confiável para casos de erro ortográfico.

A segunda função de similaridade proposta, denominada NAMEMATCH, compara conjuntos de nomes próprios. No caso desta tese, ela é aplicada especificamente para comparar nomes de autores associados a dois registros bibliográficos.

Seja Str um conjunto de cadeias de caracteres, $\mathbb{R}_1$ o conjunto de números reais no intervalo $[0, 1]$. NAMEMATCH : $\{Str \times Str\} \rightarrow \mathbb{R}_1$ é definida pelo algoritmo

```
NAMEMATCH(K, L)
1  m ← LENGTH(K);
2  n ← LENGTH(L);
3  for i ← 1 to m
4      do for j ← 1 to n
5             do if INISIM(K_i, L_j) = 1
6                   then counter ← counter + 1;
7                        L_j ← null;
8  return counter/MAX(m, n);
```

em que K, L são listas compostas pelas iniciais dos nomes dos autores dos registros bibliográficos que estão sendo comparados, $K_i \in Str$ é o i-ésimo elemento da lista K, ou seja, as iniciais do i-ésimo nome de autor do primeiro registro, $L_j \in Str$ é o j-ésimo elemento da lista L, ou seja, as iniciais do j-ésimo nome de autor do segundo registro. Além disso, considere a função LENGTH que retorna o tamanho de uma lista, a variável $counter$ que indica o número de casamentos encontrados pela função INISIM e a função MAX que retorna o tamanho da maior lista.

O algoritmo exige dois parâmetros que correspondem às listas de iniciais dos nomes dos autores de dois registros bibliográficos. Ambas as listas são percorridas (linhas 3-4) com o objetivo de encontrar casamentos entre os nomes de autores, os quais são identificados pela função INISIM. Quando ocorre um casamento entre dois nomes de autores (linha 5), a função atribui um valor nulo às iniciais do j-ésimo autor, evitando comparações futuras (linha 7). A variável $counter$ conta o número de casamentos encontrados (linha 6) que é dividido pelo tamanho da maior lista (linha 8). NAMEMATCH retorna o escore $s \in \mathbb{R}_1$ que representa a porcentagem de casamentos encontrados entre os nomes de autores dos registros bibliográficos comparados.

Por exemplo, considere a omissão de um autor no segundo parâmetro da função

NAMEMATCH ([AB, CD, EF] , [FE, BA])

Os valores 3 e 2 são atribuídos às variáveis m e n. A lista [AB, CD, EF] é processada e cada elemento é comparado a todos os elementos ainda não casados da lista [FE, BA]. A função INISIM retorna 1 para os pares (AB, BA) e (EF, FE). A variável $counter$ é incrementada para 2. A função retorna 67%.

Note que NAMEMATCH permite comparar registros bibliográficos com número diferente de autores. Portanto, mesmo que ocorram erros nas iniciais de algum autor, é bastante improvável que existam erros nas iniciais de vários autores de um registro bibliográfico ao mesmo tempo.

A função proposta visa maximizar a revocação, ou seja, recuperar o maior número de casamentos relevantes. Embora a função INISIM possa identificar incorretamente autores com as mesmas iniciais, é bastante improvável que para um determinado par de registros bibliográficos existam muitos autores com as mesmas iniciais, mas com nomes diferentes. Além disso, um par de registros raramente será classificado como duplicado apenas com base nesta evidência. Ainda são utilizados outros metadados como ano e título da publicação, elevando a precisão do processo de deduplicação.

Os resultados experimentais empíricos desta tese demonstram a efetividade destas suposições aplicadas às funções INISIM e NAMEMATCH. Além disso, os resultados relatados no artigo publicado no periódico *Information Processing & Management* (BORGES et al., 2011) mostram ganho de até 183% na qualidade da deduplicação quando INISIM é comparada às funções específicas para nomes próprios apresentadas na Tabela 3.2 (com exceção de *Jaro-Winkler*).

4.4.2 Modelos de Classificação

Os dados transformados de acordo com o procedimento descrito no início da Seção 4.4 são utilizados para compor o conjunto de treinamento do nível básico. A Tabela 4.4 apresenta um exemplo de instância deste conjunto. Os vetores de características são compostos pelos metadados apresentados na Tabela 4.3: *author number diff*, *year diff*, *author diff*, *author sim*, *title diff* e *title sim*. A classe de dados é definida pelo campo *duplicated pair*.

Tabela 4.4: Conjunto de treinamento.

author number diff	**year diff**	**author diff**	**author sim**	**title diff**	**title sim**	**duplicated pair**
1	0	2	0,6	4	0,87	yes

Os seguintes algoritmos foram utilizados para treinar modelos de classificação baseados em características distintas:

- *Multilayer Perceptron* (MLP) (HAYKIN, 2007) - rede neural artificial, baseado em função;
- *Voted Perceptron* (FREUND; SCHAPIRE, 1998) - baseado em função;
- *SMO* (PLATT, 1999) - variação do SVM (BOSER; GUYON; VAPNIK, 1992), baseado em função;

- *Naïve Bayes* (NB) (JOHN; LANGLEY, 1995) - baseado no teorema de Bayes;
- *Bayes Net* (COOPER; HERSKOVITS, 1992) - baseado no teorema de Bayes;
- *RIPPER* (COHEN, 1995) - baseado em regras;
- *C4.5* (QUINLAN, 1993) - baseado em árvores de decisão.

O uso de múltiplos classificadores heterogêneos potencialmente oferece informação complementar sobre o padrão dos registros bibliográficos duplicados. Idealmente, quanto mais heterogêneo o conjunto de algoritmos, maior a diversidade nas classes preditas pelos classificadores e, consequentemente, maior a capacidade do empilhamento em elevar a qualidade da deduplicação quando comparado ao melhor classificador do conjunto.

4.5 Empilhamento

Os classificadores aprendidos na etapa de modelagem são utilizados para gerar predições binárias (réplica ou não-réplica) para cada um dos pares de registros bibliográficos contidos no conjunto de teste. Mantendo a mesma classe de dados, foram definidos três conjuntos instâncias do metanível, seguindo abordagens distintas:

1. O vetor de características é composto pelo rótulo das classes preditas pelos modelos treinados. A Tabela 4.5 apresenta um exemplo considerando os algoritmos MLP, NB e C4.5.

Tabela 4.5: Empilhando rótulo das classes preditas.

MLP	**NB**	**...**	**C4.5**	**duplicated pair**
yes	no	...	yes	yes

2. O vetor de características é composto pelo valor das predições de cada um dos modelos treinados. A Tabela 4.6 apresenta o exemplo anterior adaptado de acordo com esta abordagem.

Tabela 4.6: Empilhando valores de predição.

MLP		**NB**		**...**	**C4.5**		**duplicated pair**
yes	**no**	**yes**	**no**		**yes**	**no**	
1	0	0,25	0,75	...	0,95	0,05	yes

Assim como proposto por TING; WITTEN (1999), os valores de predição variam no intervalo fechado $[0, 1]$ e dependem do algoritmo utilizado. Por exemplo, para o algoritmo *Naïve Bayes*, a predição corresponde à probabilidade *a posteriori* de um par de registros pertencer à mesma classe, ou seja, representarem metadados bibliográficos duplicados. Já para os algoritmos *RIPPER* e *C4.5*, a predição corresponde à precisão da regra ou do nó que classificou o par de registros. Nos algoritmos baseados em função, a predição binária é mapeada diretamente para o rótulo da classe a ser predita.

Tabela 4.7: Empilhando rótulos e predições.

MLP			NB			...	C4.5			duplicated pair
yes	**no**	**R**	**yes**	**no**	**R**		**yes**	**no**	**R**	
1	0	yes	0,25	0,75	no	...	0,95	0,05	yes	yes

3. O vetor de características é composto tanto pelo rótulo como pelo valor das predições de cada um dos modelos treinados. A Tabela 4.7 apresenta o exemplo anterior adaptado de acordo com esta abordagem.

Por fim, o classificador do metanível é treinado utilizando qualquer algoritmo de classificação a partir do conhecimento adquirido pelos classificadores de nível básico e é finalmente usado para inferir a classe das instâncias do conjunto de teste, ou seja, dos pares candidatos selecionados na etapa de blocagem.

Na avaliação experimental relatada no Capítulo 5, o método proposto foi exercitado utilizando os mesmos algoritmos apresentados na Seção 4.4.2 para treinamento do metanível. Os resultados foram avaliados utilizando-se a métrica de qualidade medida F (MANNING; RAGHAVAN; SCHUTZE, 2008), calculada em função da precisão e da revocação. Esta avaliação é interpretada como a qualidade do processo de deduplicação dos metadados bibliográficos.

4.6 Blocagem

A estratégia de blocagem proposta é dividida em dois níveis e está representada graficamente na Figura 4.2. Inicialmente, todos os registros armazenados no repositório de metadados bibliográficos são organizados em blocos com anos de publicação compatíveis. Cada conjunto de registros de um mesmo bloco é reorganizado em blocos menores no segundo nível, contendo apenas os registros com nomes de autores compatíveis.

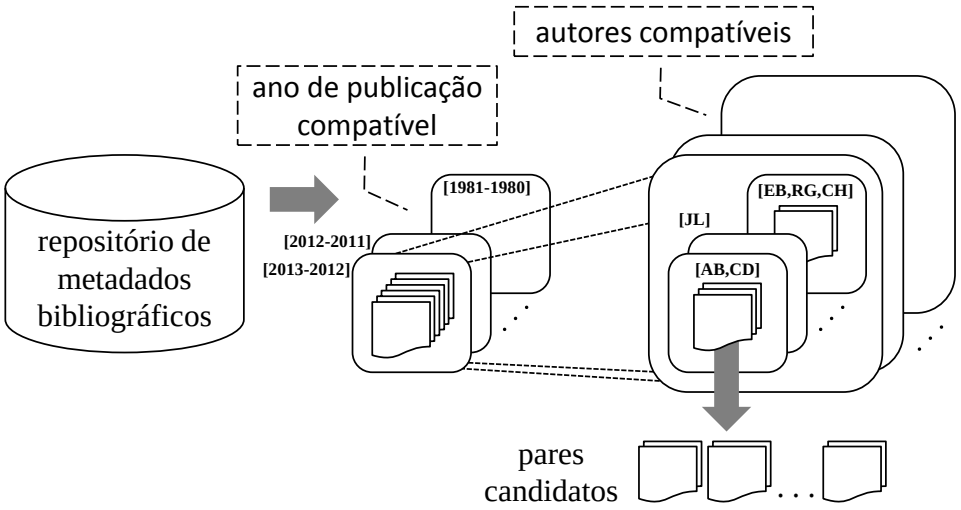

Figura 4.2: Estratégia de blocagem proposta.

A compatibilidade dos anos de publicação é calculada pelo mesmo ano ou por apenas um ano de diferença. Dessa forma, cada bloco inicial contém os registros publicados em um determinado intervalo de 2 anos. Por exemplo, seriam gerados os blocos 2013-2012, 2012-2011, 2011-2010, e assim sucessivamente. Esta estratégia é uma adaptação da vizinhança ordenada (*sorted neighborhood approach*) (HERNÁNDEZ; STOLFO, 1998) em que a janela deslizante não depende do número de registros mas do resultado de uma operação sobre o conteúdo de um campo. Assim, o tamanho de registros por janela pode variar. DRAISBACH et al. (2012) também propõem uma versão da vizinhança ordenada que aumenta ou diminui o tamanho da janela em função da similaridade.

A compatibilidade dos nomes de autores é calculada de acordo com a função NAMEMATCH. Os blocos menores são compostos pelos registros que contenham a maioria dos autores com as mesmas iniciais segundo a função INISIM.

A Tabela 4.8 apresenta 4 exemplos de registros. Aplicando estratégia de blocagem proposta são criados dois blocos iniciais: [2000-1999] contendo os registros {1,2,3} e [2001-2000] composto por {2,3,4}. Considerando apenas os registros contidos em [2000-1999], a segunda fase produz um único bloco [EB,RG] porque mais da metade dos autores de cada registro têm iniciais compatíveis. Os pares candidatos gerados são {(1,2), (2,3), (1,3)}. Analisando os registros do bloco [2001-2000], são produzidos dois blocos [EB,RG] = {2,3} e [EB] = {4}. Como o único par possível (2,3) já está no conjunto de candidatos, a blocagem retorna {(1,2),(2,3),(1,3)}.

A diferença entre os anos de publicação bem como as funções NAMEMATCH e INISIM funcionam como *canopies* da distância de edição que é aplicada no cálculo da similaridade entre os títulos. Esta estratégia é simples, porém bastante eficiente porque evita a operação de uma função de similaridade com complexidade quadrática em relação ao tamanho das *strings* comparadas. O processo ainda pode ser acelerado criando-se um índice invertido sobre as iniciais dos nomes dos autores, que facilita a aplicação da função NAMEMATCH apenas sobre os registros que contenham pelo menos um autor com inicial compatível.

Os experimentos apresentados no artigo publicado no periódico *Information Processing & Management* (BORGES et al., 2011) demonstram a efetividade da estratégia proposta. Para um conjunto de dados composto por 1985 registros provenientes das bibliotecas digitais BDBComp e DBLP, contendo 417 referências duplicadas, foram selecionados apenas 383 candidatos dos aproximadamente 1,5 milhões de pares possíveis (0,025%). Estes pares foram determinados a partir da combinação de registros com valores válidos para todos os campos apresentados na Tabela 4.3, ou seja, foram excluídos os *outliers*. Em um segundo conjunto de dados de referências replicadas contendo 2191 registros que incluem 29.611 pares duplicados, foram selecionados 31.623 candidatos dos aproximadamente 1,9 milhão de pares possíveis (1,68%).

Tabela 4.8: Exemplos de registros para blocagem.

	Ano	**Autores**	**Título**
1	1999	Eduardo Borges; Renata Galante; Carlos Heuser	Deduplicação de Metadados
2	2000	Gabriel Ramos; Edimundo Batista	Algoritmos de Classificação
3	2000	Renata Galante; Borges, Eduardo	Deduplicação de Dados
4	2001	Edimundo C. Batista	Algoritmo de classificação

Como não faz parte das contribuições desta tese, uma avaliação experimental mais detalhada desta etapa do método está prevista apenas como trabalho futuro. Embora a blocagem seja essencial para lidar com questões de escalabilidade, ela não é determinante na qualidade da deduplicação. Soluções recentes da literatura (CHRISTEN, 2012b), incluindo outras estratégias de blocagem (BAXTER; CHRISTEN; CHURCHES, 2003), podem ser usadas para substituir ou aprimorar a estratégia proposta. Além disso, a avaliação das funções de similaridade pode ser acelerada em ambientes distribuídos (DAL BIANCO; GALANTE; HEUSER, 2011) usando um modelo de programação paralela como o MapRaduce (DEAN; GHEMAWAT, 2008).

4.7 Deduplicação

Por fim, sobre os pares candidatos selecionados pela estratégia de blocagem são aplicadas as funções apresentadas na Tabela 4.3 seguindo o mesmo processo detalhado no início da Seção 4.4. Dessa forma, as instâncias geradas podem ser submetidas aos modelos de classificação do nível básico, cujas predições são usadas pelo classificador de metanível para efetivamente rotular o par como duplicado ou como registros distintos.

5 AVALIAÇÃO EXPERIMENTAL

Este capítulo descreve uma série de experimentos que avaliam o método proposto para identificar automaticamente metadados bibliográficos duplicados. O objetivo destes experimentos é medir a qualidade do resultado da deduplicação de registros e esclarecer o ganho real do empilhamento frente a escolha do melhor classificador e da estratégia de combinação voto da maioria.

A Seção 5.1 define as métricas de avaliação da qualidade da deduplicação. A Seção 5.2 mostra detalhe sobre os conjuntos de dados utilizados. As configurações do método proposto são apresentadas na Seção 5.3. Os experimentos foram organizados em dois grupos. No primeiro, apresentado na Seção 5.4, os exemplos de treinamento são selecionados aleatoriamente e os resultados são analisados em função da distribuição das predições dos classificadores de nível básico. Também são investigados os casos em que o método proposto falha na identificação de metadados bibliográficos duplicados. Por fim, a Seção 5.5 detalha os experimentos do segundo grupo. Os exemplos de treinamento são selecionados utilizando-se a estratégia proposta baseada nas características do conjunto de dados (Seção 4.3). Os resultados são analisados em função da diversidade dos classificadores envolvidos. Cada grupo de experimentos apresenta a qualidade da deduplicação utilizando múltiplas estratégias de empilhamento.

5.1 Métricas de avaliação

A deduplicação foi avaliada a partir de métricas que estimam a qualidade da classificação, ou seja, avaliam a capacidade de predição do modelo aprendido. Foram utilizadas as seguintes métricas para avaliação dos resultados:

- precisão - razão entre o número de instâncias classificadas corretamente e o número total de instâncias da classe predita.
- revocação - razão entre o número de instâncias classificadas corretamente e o número total de instâncias da classe real
- medida F balanceada (F1) - média harmônica entre a precisão e a revocação. Quanto mais próxima de 100%, maior a qualidade geral da deduplicação.
- teste T (*Student's t-test*) (STUDENT, 1908) - verifica se a diferença entre o desempenho de dois métodos de deduplicação é estatisticamente significativo. O teste T avalia se as médias de duas distribuições normais de valores são estatisticamente diferentes, apresentando bons resultados mesmo quando as distribuições não são perfeitamente normais. Foi utilizado o limiar de significância estatística $\alpha = 0,01$.

Quando o valor da probabilidade p bi-caudal calculado é menor que α, existe uma diferença significativa entre os desempenhos dos métodos analisados. O melhor desempenho é do método com a maior média de distribuição.

5.2 Conjuntos de dados

Foram utilizados quatro conjuntos de dados reais que consistem de referências bibliográficas para artigos científicos da área Ciência da Computação. As Seções 5.2.1 e 5.2.4 descrevem coleções de referências bibliográficas extraídas de mecanismos de busca na *Web*. Já as Seções 5.2.2 e 5.2.3 detalham coleções provenientes de bibliotecas digitais. Por fim, a Seção 5.2.5 sumariza características dos conjuntos de dados e apresenta estatísticas em relação ao número de registros.

5.2.1 Cora

O primeiro conjunto de dados utilizado nos experimentos apresentados nesta tese foi extraído da coleção Cora, que é formada por um conjunto de referências para artigos científicos da Ciência da Computação obtidos a partir do uso do motor de busca *Cora Computer Science* (MCCALLUM et al., 2000). As referências foram segmentadas em metadados por um sistema de extração de informação, resultando em certo ruído entre os campos. Por exemplo, algumas datas de publicação foram capturadas em algum campo diferente de *ano de publicação*. Os dados brutos contam com 2191 registros (referências) distribuídos em 305 classes distintas (publicações reais). Esta coleção tem sido utilizada para avaliação experimental em diversos trabalhos relacionados à deduplicação (BORGES et al., 2011a,b; CARVALHO et al., 2008, 2006; BILENKO; MOONEY, 2003). A Tabela 5.1 apresenta um exemplo de registros duplicados neste conjunto de dados. Além das diferentes representações dos nomes de autores e do veículo de publicação, o segundo registro apresenta informação adicional sobre números de página e editora.

Tabela 5.1: Exemplo de registros duplicados no conjunto de dados Cora.

Campo	**Conteúdo**
title	Applying the weak learning framework to understand and improve C4.5.
author	Tom Dietterich, Michael Kearns, and Yishay Mansour.
year	1996.
venue	In Machine Learning: Proceedings of the Thirteenth International Conference,
title	Applying the weak learning framework to understand and improve C4.5.
author	Dietterich, T. G., Kearns, M., & Mansour, Y.
year	1996.
venue	In Proceedings of the 13th International Conference on Machine Learning,
other	(pp. 96-104). Morgan Kaufmann.

5.2.2 DBLP-BDBComp

O segundo conjunto de dados foi criado a partir de metadados bibliográficos extraídos das bibliotecas digitais BDBComp[1] e DBLP[2]. Quando os metadados foram coletados (01/03/2007), havia cerca de 4 mil referências para artigos científicos publicados no Brasil

[1] `http://www.lbd.dcc.ufmg.br/bdbcomp`. Data do acesso: 8/11/2013.
[2] `http://www.informatik.uni-trier.de/~ley/db`. Data do acesso: 8/11/2013.

Tabela 5.2: Conferências cobertas pelo conjunto de dados DBLP-BDBComp.

Conferência	Intervalo	BDBComp	DBLP	Área do Conhecimento	Abrangência
ERBD	2005	18		Bancos de Dados	Nacional
SBBD	2001-2005	**122**	**143**	Bancos de Dados	Nacional
WIDM	2001-2004		76	Bancos de Dados	Internacional
ER	2001-2004		214	Sistemas de Informação	Internacional
CAiSE	2001-2004		192	Sistemas de Informação	Internacional
SIBGRAPI	2001-2004	**242**	**274**	Computação Gráfica	Nacional
CGI	2001-2004		196	Computação Gráfica	Internacional
SVR	2001-2004	107		Realidade Virtual	Nacional
INTERACT	2003		215	Interação Humano-Computador	Internacional
SBIA	2002-2004	**93**	**93**	Inteligência Artificial	Nacional
Σ	2001-2005	582	1.403		

na BDBComp. Os registros foram coletados utilizando-se o protocolo OAI-PMH, no formato Dublin Core (DCMI USAGE BOARD, 2012). Na DBLP, havia mais de 800 mil referências para artigos científicos publicados em diversos países. Os metadados foram coletados no *site Web* da biblioteca digital no formato XML. Esta coleção foi utilizada para avaliação experimental em (BORGES et al., 2011) com o nome de *Libraries*.

Foram selecionados os metadados que descrevem os artigos científicos publicados em algumas conferências nacionais e internacionais. A Tabela 5.2 indica o número de referências bibliográficas em cada biblioteca digital para cada conferência, totalizando 1985 registros. Além disso, esta tabela apresenta o intervalo de anos de publicação selecionado. A última linha da tabela apresenta o total de registros que compõe o conjunto de dados denominado DBLP-BDBComp.

As conferências SBBD, SBIA e SIBGRAPI são indexadas pelas duas bibliotecas digitais, portanto existem registros bibliográficos duplicados. Foram identificados por um usuário especialista 417 pares duplicados provenientes destas três conferências. A Tabela 5.3 mostra um exemplo de registros duplicados presente neste conjunto de dados. Existem omissões de nomes do meio dos autores e uma pequena variação no veículo de publicação. Além disso, cada registro contém informação única como números de página, idioma, local e editora.

Tabela 5.3: Exemplo de registros duplicados no conjunto de dados DBLP-BDBComp.

Campo	Conteúdo
title	Mining Reliable Models of Associations in Dynamic Databases.
authors	Adriano Veloso, Wagner Meira Jr., Márcio de Carvalho
year	2002
booktitle	SBBD
pages	263-277
title	Mining reliable models of associations in dynamic databases.
creators	Adriano A. Veloso, Wagner Meira Jr., Márcio Luiz Bunte de Carvalho
date	2002
source	sbbd2002
language	por
coverage	Gramado, RS, Brasil
rights	Sociedade Brasileira de Computação

5.2.3 DBLP-ACM

O terceiro conjunto de dados consiste de metadados bibliográficos extraídos da biblioteca digital da ACM[3] e da DBLP. As referências selecionadas cobrem os mesmos conjuntos de conferências e revistas da Ciência da Computação, totalizando 4.910 registros para 2.687 publicações distintas. Esta coleção tem sido utilizada para avaliação de outros trabalhos relacionados à deduplicação (KÖPCKE; RAHM, 2010; KÖPCKE; THOR; RAHM, 2010a,b; THOR; RAHM, 2007). A Tabela 5.4 apresenta um exemplo de registros duplicados neste conjunto de dados. Entre os principais problemas destacam-se o deslocamento de algumas palavras do título e o acrônimo no veículo de publicação.

Tabela 5.4: Exemplo de registros duplicados no conjunto de dados DBLP-ACM.

Campo	**Conteúdo**
title	An open abstract-object storage system
authors	Stephen Blott, Lukas Relly, Hans-Jörg Schek
venue	International Conference on Management of Data
year	1996
url	263-277
title	An Open Storage System for Abstract Objects
authors	Lukas Relly, Stephen Blott, Hans-Jörg Schek
venue	SIGMOD Conference
year	1996

5.2.4 DBLP-Scholar

O último conjunto de dados consiste dos mesmos registros provenientes da DBLP contidos na coleção DBLP-ACM adicionados de referências extraídas do motor de busca Google Scholar[4]. Para obter estas referências, foram executadas diversas consultas por título e por veículo de publicação e os resultados adicionados ao conjunto de dados. O Scholar extrai automaticamente os metadados bibliográficos de documentos capturados na *Web*, gerando registros duplicados devido a problemas como heterogeneidade na representação da autoria, variações de grafia e erros de extração da informação. Foram identificados por um usuário especialista 2.517 pares duplicados entre os 66.879 registros. Esta coleção foi utilizada para avaliação dos mesmos trabalhos que DBLP-ACM. A Tabela 5.5 apresenta um exemplo de registros duplicados neste conjunto de dados. Os títulos diferem em muitas palavras. Mais da metade dos autores foram omitidos no primeiro registro. O veículo de publicação é representado como um acrônimo no segundo registro. Além disso, existem problemas na codificação de caracteres.

5.2.5 Estatísticas sobre os conjuntos de dados

A Tabela 5.6 apresenta, para cada conjunto de dados, o número de registros proveniente de cada fonte (biblioteca digital ou motor de busca), o total de registros contidos no conjunto, o número de referências distintas e o número de pares duplicados. O método de deduplicação proposto tem como objetivo identificar os elementos da última coluna, ou seja, detectar quando dois registros bibliográficos referenciam a mesma publicação real.

[3] `http://dl.acm.org/`. Data do acesso: 8/11/2013.

[4] `http://scholar.google.com`. Data do acesso: 8/11/2013.

Tabela 5.5: Exemplo de registros duplicados no conjunto de dados DBLP-Scholar.

Campo	Conteúdo
url	eSPpim_OX4QJ
title	The ClustRa Telecom Database
authors	S Hvasshovdâ?l
venue	
year	1995
url	conf/vldb/HvasshovdTBH95
title	The ClustRa Telecom Database: High Availability, High Throughput, and Real-Time Response
authors	S Hvasshovd, Ø Torbjørnsen, S Bratsberg, P Holager
venue	VLDB
year	1995

Tabela 5.6: Estatísticas sobre os conjuntos de dados utilizados na avaliação experimental.

Conjunto de dados	Registros			Referências distintas	Pares duplicados
	Fonte 1	Fonte 2	Total		
Cora	-	-	2.191	305	25.304
DBLP-BDBComp	1.403	582	1.985	1.570	417
DBLP-ACM	2.616	2.294	4.910	2.687	2.218
DBLP-Scholar	2.616	64.263	66.879	61.662	2.517

5.3 Configurações dos experimentos

Esta seção apresenta as principais configurações do método proposto utilizadas na avaliação experimental. A avaliação mede a qualidade da deduplicação de registros e verifica o ganho do empilhamento em relação à escolha do melhor classificador e ao voto da maioria.

O primeiro grupo de experimentos avalia as etapas de pré-processamento, modelagem, empilhamento e deduplicação (vide Figura 4.1), utilizando o conjunto de dados Cora. Os registros foram processados de acordo com o procedimento detalhado na Seção 4.2 e combinados em pares gerando aproximadamente 1,9 milhões de novas instâncias. As funções de distância e similaridade (Tabela 4.3) foram aplicadas a cada nova instância. Por fim, mantendo a mesma proporção de pares duplicados, foram selecionadas aleatoriamente 10% das instâncias cinco vezes para compor cinco amostras distintas.

Cada amostra tem 2.540 (1,3%) instâncias rotuladas como réplicas (*duplicated pair= yes*) e 191.013 (98,7%) como objetos do mundo real distintos (*duplicated pair = no*). O número de instâncias foi reduzido porque os algoritmos de classificação rodam inteiramente em memória e não têm complexidade linear.

A qualidade da deduplicação e o ganho do empilhamento são analisados em função da distribuição das predições dos classificadores de nível básico. Também são investigados os casos em que o método proposto falha na identificação de metadados bibliográficos duplicados.

O segundo grupo de experimentos avalia todas as etapas do método proposto com exceção da blocagem, utilizando todos os conjuntos de dados: Cora, DBLP-BDBComp, DBLP-ACM e DBLP-Scholar. Os exemplos de treinamento são selecionados utilizando-

se a estratégia proposta baseada nas características do conjunto de dados. O número mínimo de pares de cada amostra $n = 1842$ foi calculado considerando grau de confiança 99% e erro amostral de 3%. A Tabela 5.7 apresenta, para cada conjunto de dados, a quantidade de registros, o número de pares duplicados existentes e o número de pares selecionados aleatoriamente pela estratégia proposta em cada amostra distribuídos nas seguintes categorias: duplicados ($Pairs_{Dup}$), claramente distintos ($Clear_{\neg Dup}$) e distintos com alguma similaridade ($Possible_{\neg Dup}$).

Tabela 5.7: Estatísticas sobre as amostras de treinamento do nível básico.

Conjunto de dados	**Registros**	**Pares duplicados**	SUBSETOFPAIRS(**Registros**)		
			$Pairs_{Dup}$	$Clear_{\neg Dup}$	$Possible_{\neg Dup}$
Cora	2.191	25.304	1.096	548	548
DBLP-BDBComp	1.985	417	209	888	888
DBLP-ACM	4.910	2.218	1.109	1.901	1.901
DBLP-Scholar	66.879	2.517	1.259	32.810	32.810

Foram considerados claramente distintos os pares em que não há similaridade entre os autores (*author sim* $= 0$) ou cujos títulos apresentam similaridade menor que um determinado limiar (*title sim* $< 0,25$). Consequentemente, pares com alguma similaridade têm *author sim* > 0 e *title sim* $\geq 0,25$.

Os valores apresentados nas últimas três colunas foram determinados a partir da execução do algoritmo SUBSETOFPAIRS sobre cada conjunto de dados. Perceba que o número de pares de cada amostra (soma das últimas três colunas) é maior que o mínimo exigido pelo cálculo amostral e igual ao número de registros do conjunto de dados.

Para a modelagem dos classificadores de nível básico, foram selecionadas 3 amostras de treinamento para cada conjunto de dados. Os respectivos conjuntos de teste são formados por todas as combinações de pares que não foram selecionadas para o treinamento. A única exceção ocorre para o conjunto de dados DBLP-Scholar em que foram selecionados todos os pares duplicados remanescentes adicionados de aproximadamente 900 mil pares distintos. A Tabela 5.8 apresenta o número aproximado de registros e a porção de pares duplicados. A qualidade da deduplicação e o ganho do empilhamento são analisados em função da diversidade dos classificadores envolvidos.

Tabela 5.8: Estatísticas sobre os conjuntos de teste do nível básico.

Conjunto de dados	**Registros**	**Pares duplicados**
Cora	1,9 M	1.095
DBLP-BDBComp	1,9 M	208
DBLP-ACM	5,9 M	1.109
DBLP-Scholar	*0,9 k	1.258

* Reduzido devido ao tamanho do conjunto de dados

Nos dois grupos de experimentos, são apresentados resultados considerando múltiplas abordagens de empilhamento, em que o vetor de características é composto por:

- **rótulos** das classes preditas pelos modelos treinados;
- valores das **predições** de cada um dos modelos treinados;

- **rótulos e predições**.

Os experimentos relatados neste capítulo não fazem uso de qualquer técnica de blocagem, embora esteja descrita a aplicação de uma estratégia específica no método proposto. Esta decisão foi adotada para evitar qualquer distorção nos valores de revocação que poderiam ocorrer visto que nem todos os registros duplicados poderiam ficar organizados nos mesmos blocos. Nos trabalhos futuros, descritos no Capítulo 6, está prevista a experimentação usando a estratégia de blocagem descrita na Seção 4.6.

Todos os experimentos foram executados em um computador pessoal usando o Sistema Gerenciador de Bancos de Dados (SGBD) PostgreSQL[5] e a ferramenta de mineração de dados Weka[6] (WITTEN; FRANK; HALL, 2011).

5.4 Primeiro grupo de experimentos: seleção aleatória de exemplos

Os resultados da deduplicação dos modelos de nível básico estão sumarizados na Tabela 5.9 que apresenta, para cada algoritmo de classificação, a média e o desvio padrão da medida F balanceada (F1) considerando as cinco amostras do conjunto de dados Cora. Os parâmetros foram estabelecidos manualmente visando a criação de modelos de fácil interpretação e de modo a evitar superajuste (*overfitting*) em relação ao conjunto de dados. Os modelos foram avaliados utilizando validação cruzada 10-*fold*. Os resultados consideram apenas a classe de interesse (*duplicated pair = yes*) e estão ordenados de acordo com o melhor desempenho em F1.

Tabela 5.9: Qualidade dos classificadores de nível básico.

	Algoritmo	**Parâmetros**	**F1(%)**
1	*RIPPER*	uma fase de otimização	90,28 $\pm$ 0,53
2	*C4.5*	10 instâncias por folha 15% de confiança para poda	89,66 $\pm$ 0,64
3	*MLP*	5 nós na camada oculta 50 épocas	89,40 $\pm$ 0,14
4	*SMO*	*kernel* linear	88,30 $\pm$ 0,10
5	*BayesNet*	*default*	84,56 $\pm$ 0,57
6	*Voted Perceptron*	*default*	82,64 $\pm$ 0,24
7	*Naïve Bayes*	*default*	81,24 $\pm$ 0,39

Observando a Tabela 5.9, percebe-se que os resultados foram próximos considerando os quatro melhores algoritmos (linhas 1-4). O melhor resultado de 90,28 $\pm$ 0,53% foi obtido pelo algoritmo *RIPPER* (linha 1), que marcou precisão de 86,1 $\pm$ 1,2% e revocação igual a 95 $\pm$ 1,7%. O pior resultado de 81,24 $\pm$ 0,39% foi apresentado pelo algoritmo *Naïve Bayes* (linha 7), principalmente devido à baixa precisão menor que 70%. Valores de precisão baixos indicam que muitos falsos positivos foram retornados, o que degrada

[5] `http://www.postgresql.org`. Data do acesso: 8/11/2013.

[6] `http://www.cs.waikato.ac.nz/ml/weka`. Data do acesso: 8/11/2013.

a qualidade do processo de deduplicação. Todos os experimentos atingiram F1 maior que 80%, ou seja, os algoritmos de classificação testados combinados com as funções de similaridade propostas foram efetivos para identificar a maioria dos registros bibliográficos duplicados sem comprometer o resultado da deduplicação com um número excessivo de falsos positivos.

Para entender melhor como os classificadores utilizam os valores de distância ou similaridade para rotular um par de registros como referências duplicadas ou não, foram analisados os melhores modelos gerados neste grupo de experimentos. A Tabela 5.10 apresenta uma lista de regras induzidas pelo algoritmo *RIPPER* para uma determinada amostra. Cada linha mostra um predicado lógico que inclui certos metadados, o rótulo da classe predita (*duplicated pair*), o número de instâncias rotuladas pela regra e o número de erros de classificação. A primeira regra classifica a maioria dos pares duplicados usando apenas os campos *title sim* e *year diff*. Este comportamento mostra que esses atributos são os mais discriminatórios para a identificação de referências duplicadas. O limiar de 62% para a distância de edição normalizada é suficiente para classificar corretamente 88% das réplicas. Além dos atributos mencionados, as regras seguintes (linhas 2-6) usam a similaridade entre os autores. O limiar de 67% significa que 2/3 dos autores devem ser equivalentes. Essas regras também lidam com questões específicas nos anos de publicação e no comprimento dos títulos. A última regra classifica a maioria das instâncias como pares de referências distintas.

Tabela 5.10: Regras geradas pelo algoritmo *RIPPER*.

	Predicado	Classe predita	Rotulados	Erros
1	$\textit{title sim} \geq 0.62 \wedge \textit{year diff} \leq 0$	*yes*	2.485	258
2	$\textit{author sim} \geq 0.67 \wedge \textit{title sim} \geq 0.48 \wedge \textit{year diff} \leq 1 \wedge \textit{title diff} \geq 36$	*yes*	13	3
3	$\textit{author sim} \geq 0.67 \wedge \textit{title sim} \geq 0.72 \wedge \textit{title diff} = 4$	*yes*	11	2
4	$\textit{author sim} \geq 0.67 \wedge \textit{title sim} \geq 0.44 \wedge \textit{year diff} \leq 1 \wedge \textit{title diff} \leq 25$	*yes*	9	1
5	$\textit{author sim} \geq 0.67 \wedge \textit{title sim} \geq 0.74 \wedge \textit{year diff} \leq 2 \wedge \textit{title diff} \geq 16$	*yes*	5	1
6	$\textit{author sim} \geq 0.67 \wedge \textit{title sim} \geq 0.47 \wedge \textit{year diff} \leq 1 \wedge \textit{title diff} \geq 34$	*yes*	9	2
7	$\forall$ *instance*	*no*	191.021	275

A Figura 5.1 apresenta a árvore de decisão gerada pelo algoritmo *C4.5* para outra amostra. Os valores de classe *yes/no* são representados por *t/f* respectivamente. *title diff* foi o atributo mais discriminatório (raiz), seguido por *year diff* e *title sim*. *C4.5* foi capaz de identificar corretamente 84% das réplicas usando apenas a distância de edição entre os títulos e a diferença entre os anos de publicação. Apesar de *title diff* ser considerado mais importante que *title sim*, a similaridade é essencial para classificar corretamente a grande maioria dos pares distintos ($\textit{title diff} > 14 \wedge \textit{title sim} \leq 54\%$). A similaridade entre os autores foi utilizada no quarto nível da árvore em dois nós, identificando corretamente mais 86 réplicas. Ainda são deduplicados outros 212 pares em nós descendentes de *author sim*.

O artigo publicado nos *Proceedings of the IADIS International Conference WWW/ Internet* (BORGES et al., 2011b) apresenta características destes e de outros modelos de classificação gerados pelos algoritmos *Naïve Bayes*, *RIPPER* e *C4.5* para o mesmo conjunto de dados. Outros experimentos com pequenas alterações na configuração de parâmetros dos algoritmos de classificação estão descritos no artigo publicado nos *Proceedings of the International Conference of the Chilean Computer Science Society* (BORGES et al., 2011a). Estes experimentos incluem os resultados da deduplicação utilizando

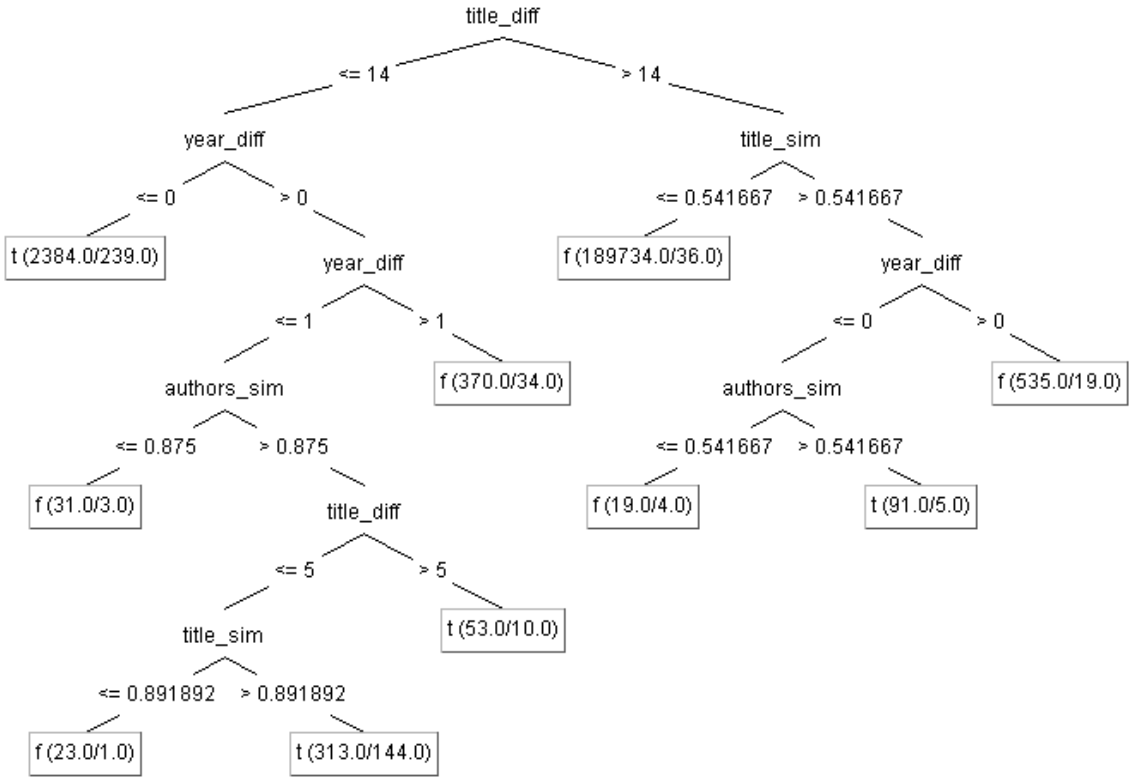

Figura 5.1: Árvore de decisão gerada pelo algoritmo *C4.5*.

um filtro de discretização (FAYYAD; IRANI, 1993) que transforma todos os atributos numéricos em nominais. Os algoritmos *MLP*, *Naïve Bayes* e *SMO* beneficiaram-se da discretização alcançando resultados em torno de 5% melhores. Entretanto, os algoritmos baseados em árvore ou regras apresentaram decréscimo na qualidade da deduplicação.

5.4.1 Avaliação da deduplicação a partir do empilhamento

Esta seção apresenta os resultados do método proposto e os compara com as estratégias voto da maioria e escolha do melhor classificador. É medida a qualidade da deduplicação empilhando tanto o rótulo da classe predita pelos classificadores de nível básico (*duplicated pair* = *yes* / *no*) quanto a distribuição de predições retornada pelos algoritmos, que varia no intervalo fechado $[0, 1]$.

A Tabela 5.11 apresenta média da medida F balanceada (F1), considerando as 5 amostras, para cada configuração do empilhamento. Em cada seção da tabela, os resultados estão ordenados de acordo com a F1. Os parâmetros utilizados em cada algoritmo foram os mesmos do nível básico.

Tabela 5.11: Combinando os classificadores com o empilhamento.

	Estratégia de combinação	Algoritmo	F1(%)
1		*SMO*	90,50 ± 0,19
2		*C4.5*	90,50 ± 0,26
3	Empilhamento	*RIPPER*	90,38 ± 0,28
4		*MLP*	90,35 ± 0,25
5	(rótulos)	*Voted Perceptron*	90,35 ± 0,13
6		*Naïve Bayes*	89,10 ± 0,16
7		*BayesNet*	89,05 ± 0,13
8		*Voted Perceptron*	90,50 ± 0,24
9		*SMO*	90,48 ± 0,24
10	Empilhamento	*RIPPER*	90,30 ± 0,19
11		*MLP*	90,00 ± 0,17
12	(predições)	*BayesNet*	87,33 ± 0,77
13		*Naïve Bayes*	80,94 ± 0,64
14		*C4.5*	76,40 ± 31,59
15		*SMO*	90,53 ± 0,21
16		*RIPPER*	90,52 ± 0,16
17	Empilhamento	*Voted Perceptron*	90,50 ± 0,24
18		*C4.5*	90,50 ± 0,22
19	(rótulos e predições)	*MLP*	90,20 ± 0,08
20		*BayesNet*	88,38 ± 0,45
21		*Naïve Bayes*	81,00 ± 1,17

Conjunto de dados Cora

A F1 média variou entre 76,40 ± 31,59% para o empilhamento configurado com o algoritmo *C4.5* usando as predições retornadas pelos classificadores (linha 14) e 90,53 ± 0,21% para o empilhamento configurado com *SMO* usando tanto os rótulos quanto as predições (linha 15). O alto desvio padrão apresentado pelo algoritmo C4.5 deve-se ao fato da distribuição de dados em uma das amostras não ter favorecido o aprendizado com os parâmetros utilizados.

A Tabela 5.12 compara o melhor resultado do empilhamento apresentando o ganho de qualidade em relação às demais estratégias de combinação de classificadores.

Tabela 5.12: Ganho do empilhamento em relação a outras estratégias de combinação de classificadores.

	Estratégia	Algoritmo	F1(%)	Ganho(%)
1	Empilhamento (rótulos e predições)	*SMO*	90,53 ± 0,21	
2	Melhor classificador	*RIPPER*	90,28 ± 0,53	0,28
3	Voto da maioria		90,17 ± 0,18	0,40

O ganho da melhor configuração do empilhamento (linha 1) quando comparada ao voto da maioria (linha 3) foi de 0,40%. O teste T realizado mostra que este ganho é relevante, pois o valor de p bicaudal $= 4,98m$ calculado é menor que o coeficiente de significância adotado $\alpha = 0,01$. Entretanto, o resultado pode ser considerado equivalente ao melhor classificador de nível básico (linha 2) porque o ganho de 0,28% não é significativo ($p = 0,06$).

5.4.2 Análise da distribuição das predições

Para entender melhor o comportamento dos algoritmos de classificação, foram analisadas as distribuições dos valores de predição na classe *duplicated pair = yes* para uma determinada amostra. A Figura 5.2 apresenta a distribuição das predições para os verdadeiros positivos. O histograma apresenta o número de instâncias classificadas corretamente como réplicas para cada intervalo de confiança da predição.

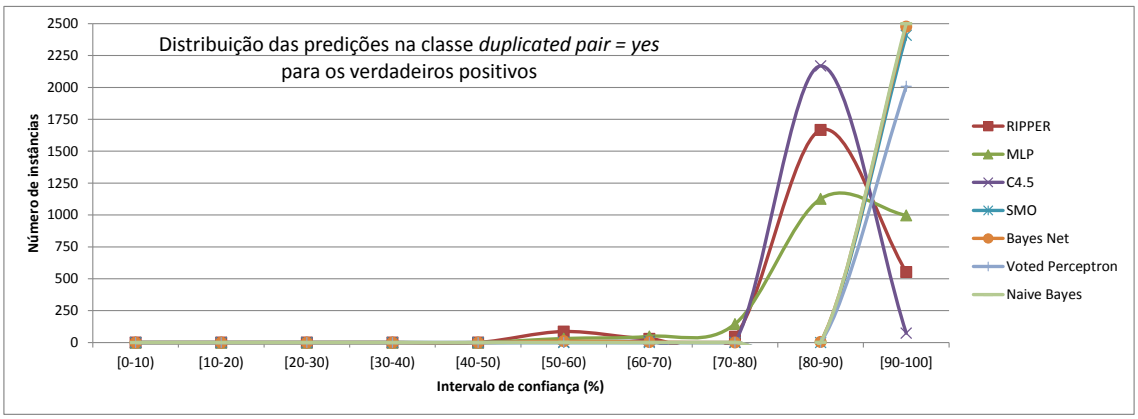

Figura 5.2: Distribuição de predições para os verdadeiros positivos.

Os pares duplicados preditos corretamente por todos os classificadores de nível básico retornaram valores de predições variando entre 80 e 100% para a grande maioria das instâncias. Entretanto, idealmente, a maioria dessas instâncias deveria estar no maior intervalo de confiança [90-100%], como é caso dos algoritmos *Bayes Net* e *Naïve Bayes*. Os algoritmos *SMO* e *Voted Perceptron* retornam uma predição binária, portanto todas as instâncias classificadas acabam no maior intervalo de confiança.

A distribuição das predições para os falsos positivos, ou seja, para os pares de registros incorretamente classificados como réplicas, é apresentada na Figura 5.3. Quanto maior a confiança do classificador, menor deveria ser a estimativa de predição para os falsos positivos. Idealmente, as instâncias deveriam estar distribuídas no menor intervalo de confiança [50-60%), como é o caso do algoritmo *RIPPER*. Considerando apenas os falsos positivos, os piores desempenhos foram dos algoritmos baseados no teorema de Bayes porque apresentaram muitos falsos positivos e distribuídos em sua grande maioria no intervalo [90-100%].

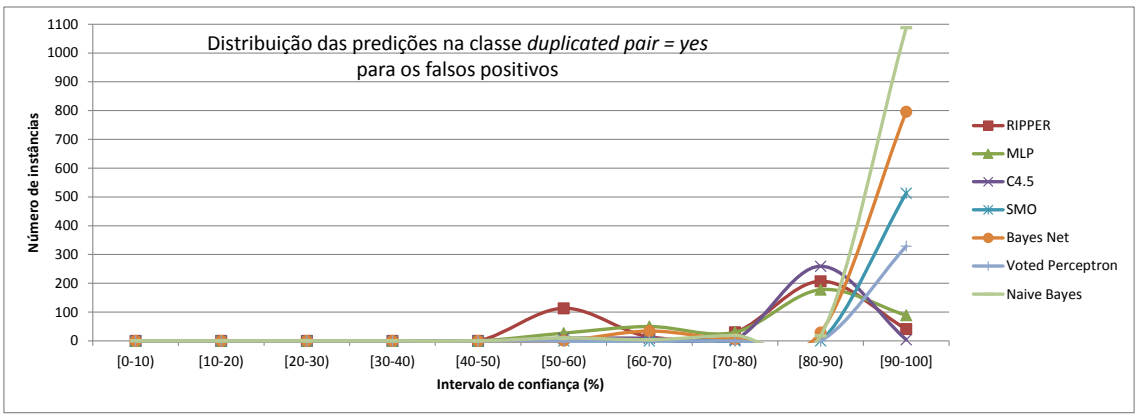

Figura 5.3: Distribuição de predições para os falsos positivos.

A partir da análise das distribuições de predição, chegou-se a conclusão de que os algoritmos mais confiáveis são aqueles que apresentaram muitos verdadeiros positivos com predições distribuídas nos dois últimos intervalos [80-100%] e poucos falsos positivos, concentrados preferencialmente nos intervalos de confiança menores que 80%. Portanto, os algoritmos mais confiáveis foram o *RIPPER* seguido do *C4.5*.

5.4.3 Análise dos casos de falha

Com o objetivo de entender melhor os resultados apresentados, foram analisados os casos de falha do método proposto. Considere *oráculo* a combinação de classificadores em que pelo menos um dos algoritmos classifica corretamente um par de referências bibliográficas. Todos os pares em que nenhum dos classificadores acerta a classe de dados podem ser considerados impossíveis de acertar. Esta análise pode ser bastante útil para desenvolvedores de novos métodos para identificação de metadados bibliográficos porque os casos de falha mostram as dificuldades de realizar o casamento de alguns campos.

Utilizando a mesma amostra em que foram analisadas as distribuições das predições (Seção 5.4.2), a qual contém mais de 193 mil pares de registros bibliográficos incluindo 2.540 pares duplicados, a Tabela 5.13 apresenta a quantidade de falsos positivos e negativos para cada caso de falha do oráculo. Falsos positivos são pares distintos detectados como duplicados por todo o conjunto de classificadores. Falsos negativos representam registros duplicados identificados como distintos. Os problemas analisados são classificados nos seguintes casos:

- Falha na segmentação de metadados - as funções retornam valores equivocados porque operam sobre valores de campos deslocados durante o processo de segmentação de metadados. Por exemplo, o título "*D.P., and Warmuth, On-line prediction and conversion strategies. In Proceedings of the First Euro-COLT Workshop.*" contém informação adicional sobre um dos autores e o veículo de publicação.

- Omissão do primeiro ou último nome - a função INISIM considera a primeira e a última inicial dos nomes dos autores. Por exemplo, "*David Haussler, Michael Kearns, and Robert Schapire.*" e "*D. Haussler, M. Kearns, R.*" não foram identificados como mesma autoria porque o último nome do terceiro autor foi omitido em um dos registros.

- Omissão de palavras no título ou títulos diferentes - a função LEVENSHTEIN não identifica corretamente a similaridade entre os títulos. Por exemplo, "*Learning problem solving heuristics by experimentation.*" e "*Learning by experimentation: Acquiring and refining problem-solving heuristics.*" diferem em diversas palavras.

- Erro no gabarito de réplicas - devido à rotulação manual das correspondências entre os registros. Por exemplo, *webber1994* e *weber1994* deveriam representar a mesma classe de dados.

- Conferência *versus* LNCS - os campos *venue* e *other* podem conter informação relevante que ajude a distinguir referências para *Lecture Notes in Computer Science*. Por exemplo, "*In Proceedings of the Second European Conference on Computational Learning Theory, pages 23-37. Springer-Verlag,*" e "*Lecture Notes in Artificial Intelligence, In Vitanyi, P. (Ed.), Vol. 904. Berlin, Germany: Springer-Verlag*".

Tabela 5.13: Casos de falha do oráculo.

Casos de falha	**Falsos positivos**	**Falsos negativos**
Falha na segmentação de metadados		**6**
Omissão do primeiro ou último nome		4
Omissão de palavras no título ou títulos diferentes		11
Erro no gabarito de réplicas	5	**2**
Conferência *versus* LNCS	3	
Conferência *versus* periódico	2	
Não publicado, publicação futura e relatório técnico	**170**	1
Total	180	12

- Conferência *versus* periódico - os campos *venue* e *other* podem conter informação relevante que ajude a distinguir artigos publicados em anais de conferências de extensões submetidas para periódicos. Por exemplo, "*Journal of Computer and System Sciences, To appear. An extended abstract appeared in Computational Learning Theory: Second European Conference, EuroCOLT '95, pages 23-37, Springer-Verlag*".

- Não publicado, publicação futura e relatório técnico - os campos *venue* e *other* podem conter informação relevante que ajude a distinguir artigos publicados, não publicados ou relatórios técnicos com os mesmos títulos e autores. Por exemplo, "*Technical Report CMU-CS-90-100, School of Computer Science, Carnegie-Mellon University, Pitts-burgh,PA*" e "*Advances in Neural Information Processing Systems in Touretzky, D., (ed.), 2, pp. 524-532, San Francisco: Morgan Kaufmann*".

Dos 180 falsos positivos analisados, a grande maioria dos casos (170) são referências para relatórios técnicos confundidas com referências para artigos científicos publicados, não publicados ou aceitos para publicação futura. Ainda ocorreram 5 erros no gabarito das réplicas, fruto de variações na grafia da classe de dados. Os casos menos frequentes são 3 confusões entre artigos publicados em uma determinada conferência e a citação para LNCS e 2 casos de artigos publicados em conferência estendidos para periódico.

Quando aplicadas as funções sobre os metadados pré-processados, a similaridade entre as iniciais dos autores foi 100% para 178 casos. Além disso, o valor da distância de edição entre os títulos é bastante baixo variando de 0 a 6 caracteres. As similaridades entre os títulos resultaram em valores que variam de 89 a 100%. Consequentemente, todas as predições retornadas pelos algoritmos de classificação são maiores que 71,70%, classificando incorretamente os pares analisados como duplicados.

Na metade dos 12 casos de falsos negativos, os problemas ocorrem devido a erros na segmentação dos campos de metadados. Parte dos campos *author* ou *venue* aparecem em *title*. Este problema teve como consequência a omissão do primeiro ou último nome de um ou mais autores e/ou as diferenças entre os títulos. Ainda ocorreram 2 erros no gabarito de réplicas e 1 caso de artigo não publicado. Nos 3 casos remanescentes, os títulos simplesmente são muito diferentes sem um motivo aparente.

O valor da distância de edição entre os títulos é bastante alto variando de 36 a 92 caracteres. As similaridades relativas foram de 7 a 49%. Esses registros apresentam problemas nos títulos que desvirtuam muito da média e os modelos acabam os desconsiderando. Consequentemente, a grande maioria das predições dos classificadores tende a zero, com

exceção de três instâncias avaliadas pelo algoritmo *BayesNet* que marcam 30,5 $\pm$ 4,9%. Mesmo assim, os 12 pares são classificados incorretamente como distintos.

Como o método proposto não utiliza os campos *venue* e *other* dos metadados bibliográficos, não é possível treinar os modelos de classificação para considerar casos de falha de artigos não publicados, publicações futuras e relatórios técnicos (170/180 dos falsos positivos). Entretanto, esses casos podem ser facilmente identificados em um passo de pré-processamento. Esta ação irá beneficiar todos os classificadores de nível básico e, muito provavelmente, não implicará um ganho do empilhamento frente ao melhor classificador ou ao voto da maioria. Quanto aos falsos negativos, 8/12 são realmente muito difíceis de acertar porque envolvem problemas de segmentação de campos e erros no gabarito. Por fim, esta análise mostra que as funções propostas para identificação da similaridades entre os autores são bastante efetivas visto que não houve casos de falha na identificação da autoria.

5.5 Segundo grupo de experimentos: seleção de exemplos baseada nas características do conjunto de dados

Os resultados da deduplicação dos modelos de nível básico estão sumarizados na Tabela 5.14 que apresenta, para cada algoritmo de classificação, a média e o desvio padrão da medida F balanceada (F1) considerando as três amostras de cada conjunto de dados. Os parâmetros foram os mesmos utilizados nos experimentos do primeiro grupo (Seção 5.4). Os modelos foram avaliados utilizando os conjuntos de teste do metanível, permitindo calcular o ganho de qualidade do empilhamento. Os resultados consideram apenas a classe de interesse (*duplicated pair = yes*) e estão ordenados de acordo com o algoritmo de classificação. Os melhores resultados estão destacados em negrito.

Observando a Tabela 5.14, percebe-se que para cada conjunto de dados, o melhor resultado foi obtido por um algoritmo de classificação diferente: *Voted Perceptron* marcou $F1 = 86,03 \pm 0,40$ na base Cora (linha 7); *C4.5* atingiu $F1 = 97,07 \pm 1,11$ na DBLP-BDBComp (linha 2); *SMO* obteve $F1 = 76,93 \pm 1,11$ na DBLP-ACM (linha 6); e *Naïve Bayes* marcou $F1 = 97,00 \pm 0,30$ na DBLP-Scholar (linha 4). Este comportamento apoia a hipótese desta tese que não há um classificador específico que funcione bem em todos os casos e que o empilhamento pode melhorar a qualidade da deduplicação utilizando informação heterogênea dos modelos aprendidos.

Tabela 5.14: F1(%) dos classificadores de nível básico.

	Algoritmo	**Cora**	**DBLP-BDBComp**	**DBLP-ACM**	**DBLP-Scholar**
1	*BayesNet*	79,87 $\pm$ 0,32	95,70 $\pm$ 0,26	62,90 $\pm$ 6,09	96,73 $\pm$ 0,67
2	*C4.5*	78,93 $\pm$ 4,45	**97,07 $\pm$ 1,11**	65,37 $\pm$ 13,64	90,73 $\pm$ 6,69
3	*MLP*	84,20 $\pm$ 2,00	84,57 $\pm$ 16,95	58,50 $\pm$ 15,52	96,93 $\pm$ 0,23
4	*Naïve Bayes*	84,47 $\pm$ 2,64	90,30 $\pm$ 0,79	49,30 $\pm$ 20,52	**97,00 $\pm$ 0,30**
5	*RIPPER*	81,43 $\pm$ 5,19	70,13 $\pm$ 40,63	49,40 $\pm$ 26,94	92,33 $\pm$ 3,15
6	*SMO*	82,83 $\pm$ 1,63	95,40 $\pm$ 0,90	**76,93 $\pm$ 1,11**	96,23 $\pm$ 0,57
7	*Voted Perceptron*	**86,03 $\pm$ 0,40**	22,53 $\pm$ 8,27	33,80 $\pm$ 4,44	88,47 $\pm$ 0,21

5.5.1 Avaliação da deduplicação a partir do empilhamento

Esta seção apresenta os resultados do método proposto e os compara com as estratégias voto da maioria e escolha do melhor classificador. É medida a qualidade da deduplicação empilhando o rótulo da classe predita pelos classificadores de nível básico, a predição retornada pelos algoritmos e a combinação de ambas as abordagens.

A Tabela 5.15 apresenta estatísticas sobre as amostras que compõem o conjunto de treinamento do metanível. Os campos apresentados são os mesmos da Tabela 5.7. A distribuição dos pares nas três categorias segue o mesmo algoritmo. Entretanto, são excluídos do conjunto de pares duplicados inicial os pares selecionados no treinamento do nível básico, ou seja, são considerados apenas os pares de teste rotulados pelos classificadores aprendidos.

Tabela 5.15: Estatísticas sobre as amostras de treinamento do metanível.

Conjunto de dados	**Registros**	**Pares duplicados**	SUBSETOFPAIRS(**Registros**)		
			$Pairs_{Dup}$	$Clear_{\neg Dup}$	$Possible_{\neg Dup}$
Cora	2.191	25.304	548	822	822
DBLP-BDBComp	1.985	417	104	940	940
DBLP-ACM	4.910	2.218	555	2.178	2.178
DBLP-Scholar	66.879	2.517	629	19.175	*19.175

* Limite existente no conjunto de dados

A Tabela 5.16 apresenta os melhores resultados da combinação dos classificadores considerando as múltiplas abordagens do empilhamento (rótulos, predições ou ambos). A métrica de avaliação e os parâmetros utilizados foram os mesmos relatados anteriormente. Para cada conjunto de dados, o melhor resultado do empilhamento é comparado às estratégias melhor classificador e voto da maioria. O ganho do empilhamento sobre estas estratégias é relatado na última coluna.

Para o conjunto de dados Cora, *Voted Perceptron* foi o melhor classificador de nível básico e atingiu $F1 = 86,03 \pm 0,40$ (linha 2). O empilhamento das predições retornadas pelo algoritmo *SMO* resultou em $F1 = 87,17 \pm 1,24$ (linha 1). Este valor representa um ganho de 1,32% em relação ao melhor classificador. Quando comparado ao resultado do voto da maioria ($F1 = 86,65 \pm 1,17$ - linha 3), o ganho foi de apenas 0,6%. O teste T realizado sobre as 2.191 observações (registros) mostra que estes ganhos são significativos, pois os valores de p bicaudal calculados ($1,18\mu$ e $11,7p$) são menores que o coeficiente de significância adotado $\alpha = 0,01$.

O maior ganho do empilhamento em relação às demais estratégias ocorreu no conjunto de dados DBLP-ACM. *SMO* foi o melhor classificador de nível básico e atingiu $F1 = 76,93 \pm 1,11$ (linha 8). O empilhamento das predições retornadas pelo algoritmo *MLP* resultou em $F1 = 81,97 \pm 1,42$ (linha 7), que representa ganho de 6,54%. Quando comparado ao voto da maioria ($F1 = 78,32 \pm 2,29$ - linha 9), o ganho foi de 4,66%. Este acréscimo de qualidade na deduplicação é bastante significativo (4.910 observações, $p \ll \alpha$), confirmando a hipótese apresentada nesta tese.

Não foi possível observar diferença de qualidade significativa entre o empilhamento e as demais estratégias nos demais conjuntos de dados. Os testes T realizados sobre as 1.985 observações para DBLP-BDBComp e 66.879 para DBLP-Scholar mostraram que $p \geq \alpha$. Entretanto, note que a F1 para os métodos *baselines* são excelentes, próximos de 97% (linhas 5-6,11-12), e dificilmente podem ser melhorados.

Tabela 5.16: Ganho do empilhamento em relação a outras estratégias de combinação de classificadores.

	Conjunto de dados	Estratégia	Algoritmo	F1(%)	Ganho (%)
1		**Empilhamento (predições)**	***SMO***	**87,17 ± 1,24**	
2	Cora	Melhor classificador	*Voted Perceptron*	86,03 ± 0,40	**1,32**
3		Voto da maioria		86,65 ± 1,17	0,60
4		Empilhamento (rótulos)	*MLP*	96,30 ± 0,90	
5	DBLP-BDBComp	Melhor classificador	*C4.5*	97,07 ± 1,11	-0,79
6		Voto da maioria		96,30 ± 1,51	0,00
7		**Empilhamento (predições)**	***MLP***	**81,97 ± 1,42**	
8	DBLP-ACM	Melhor classificador	*SMO*	76,93 ± 1,11	**6,54**
9		Voto da maioria		78,32 ± 2,29	**4,66**
10		Empilhamento (rótulos)	*BayesNet*	96,97 ± 0,50	
11	DBLP-Scholar	Melhor classificador	*Naïve Bayes*	97,00 ± 0,30	-0,03
12		Voto da maioria		96,97 ± 0,60	-0,01

Tabela 5.17: Medidas de diversidade para cada conjunto de dados.

	Conjunto de dados	$df_{avg}(m)$	$Q_{avg}(\%)$	$k_{avg}(\%)$	$\rho_{avg}(\%)$	$Dis_{avg}(m)$	$E(m)$	$KW(m)$
1	Cora	3,449 ± 0,321	99,85 ± 0,04	67,47 ± 3,14	68,39 ± 2,75	**3,37 ± 0,45**	**4,83 ± 0,59**	**1,44 ± 0,192**
2	DBLP-BDBComp	**0,004 ± 0,002**	99,89 ± 0,05	30,70 ± 0,53	34,92 ± 0,58	0,14 ± 0,02	0,16 ± 0,02	0,06 ± 0,008
3	DBLP-ACM	0,033 ± 0,006	**99,79 ± 0,06**	**25,87 ± 5,62**	**28,85 ± 5,66**	0,29 ± 0,02	0,36 ± 0,02	0,12 ± 0,009
4	DBLP-Scholar	0,030 ± 0,003	99,98 ± 0,01	42,76 ± 3,86	45,67 ± 3,22	0,11 ± 0,03	0,14 ± 0,03	0,05 ± 0,001

5.5.2 Análise do impacto da diversidade dos classificadores

A Tabela 5.17 apresenta, para cada conjunto de dados, a média e desvio padrão das medidas de diversidade apresentadas na seção 2.4 calculadas para todo conjunto de classificadores considerando todas as instâncias dos conjuntos de teste. Os melhores resultados estão destacados em negrito.

O melhor valor da falha dupla $df = 0,004 \pm 0,002m$ refere-se ao conjunto de dados DBLP-BDBComp. Entretanto, não houve ganho ou perda significativa na qualidade da deduplicação para este conjunto de dados. Este comportamento mostra que df não é uma medida de diversidade adequada para a análise do ganho em F1 porque mede apenas o quanto dois classificadores erram juntos, sem levar em consideração o quanto acertam juntos. O valor significativamente maior para o conjunto de dados Cora ($df = 3,449 \pm 0,321m$) ocorreu porque existem muitos casos em que nenhum classificador é capaz de identificar um determinado par duplicado. A análise dos casos de falha apresentada na Seção 5.4.3 mostra alguns exemplos de falsos negativos e os problemas na representação das referências bibliográficas que impossibilitam a deduplicação.

A medida de discordância Dis variou de $0,11 \pm 0,03$ até $3,37 \pm 0,45m$. Apesar de Cora apresentar a maior concordância em erros (df), também apresentou maior discordância nos acertos, indicando maior diversidade dos classificadores aprendidos. KW variou de $0,05 \pm 0,001$ até $1,44 \pm 0,192$, mas apresentou o mesmo comportamento, visto que é definido por Dis multiplicado por uma constante calculada em função do número de classificadores do conjunto.

A estatística Q variou de $99,79 \pm 0,06$ a $99,98 \pm 0,01\%$. Estes valores próximos de 1 indicam que os classificadores acertam juntos a grande maioria dos objetos, considerando os pares distintos. DBLP-ACM apresentou o melhor valor desta estatística, do coeficiente de correlação $\rho = 28,85 \pm 5,66\%$ e da concordância entre avaliadores $k = 25,87 \pm 5,62\%$. Estes valores foram até 61,7% menores dos que os calculados para os demais conjuntos de dados, sugerindo uma relação direta de k e ρ com o ganho de qualidade do empilhamento.

Em suma, os melhores resultados de 6 das 7 medidas de diversidade ocorreram justamente para os conjuntos de dados em que o empilhamento aumentou a qualidade da deduplicação.

6 CONCLUSÕES

Neste capítulo, são apresentadas as considerações finais. Os resultados alcançados e as contribuições da tese são sumarizados e associados à produção bibliográfica resultante. Por fim, são discutidos alguns trabalhos futuros.

Esta tese apresentou um método efetivo e automático para identificar metadados bibliográficos duplicados baseado em um conjunto de funções de similaridade e no empilhamento de classificadores supervisionados. Dentre as contribuições da pesquisa realizada durante o doutorado, destacam-se:

1. Uma ampla revisão bibliográfica sobre os assuntos permeados por esta tese e comparativos entre os principais trabalhos relacionados.

Esta revisão incluiu o aprendizado de máquina supervisionado, estratégias de combinação de classificadores e medidas que avaliam a diversidade dos classificadores combinados. Como resultado da pesquisa bibliográfica inicialmente desenvolvida, foi publicado o relatório técnico

> BORGES, E. N. **Um estudo sobre algoritmos de classificação aplicados à deduplicação**. 2009. Trabalho Individual III, TI-1340 — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

Os trabalhos estudados incluem diversos métodos de deduplicação de registros, os quais são classificados segundo a dependência da definição de limiares de similaridade, a técnica de aprendizagem de máquina utilizada, a estratégia de blocagem proposta, as funções de similaridade envolvidas e a área de aplicação.

2. A especificação de um método não supervisionado para a deduplicação de metadados bibliográficos, baseada em heurísticas e em um conjunto de funções de similaridade desenvolvidas especialmente para o domínio das bibliotecas digitais. O método identifica variações na representação dos nomes dos autores, tipicamente encontradas em metadados bibliográficos, utilizando um algoritmo com complexidade linear.

3. A análise dos casos de falha das funções de deduplicação propostas e de um conjunto de funções *baselines*. Esta análise é importante para o desenvolvimento de novas abordagens porque mostra as dificuldades no casamento aproximado de alguns campos como nomes de autores.

4. Uma estratégia de blocagem em dois níveis baseada em uma adaptação da vizinhança ordenada e nas funções propostas como *canopies* da distância de edição.

As contribuições citadas nos itens 2 a 4 estão detalhadas no artigo

> BORGES, E. N. et al. An unsupervised heuristic-based approach for bibliographic metadata deduplication. **Information Processing and Management**, [S.l.], v.47, n.5, p.706–718, 2011.

O método proposto reduz drasticamente o número de comparações que utilizam algoritmos de casamento de *strings* através de uma estratégia de blocagem de dois níveis bastante eficiente, mas é sensível à definição de limiares de similaridade como a diferença mínima entre os anos de publicação, a porcentagem mínima de casamentos entre os autores e a distância entre os títulos. O processo de deduplicação começa verificando se os anos de publicação do par de objetos digitais estão dentro do intervalo definido. Somente objetos cujo valor absoluto da diferença entre seus anos de publicação seja menor ou igual ao limiar previamente estabelecido terão seus autores comparados. Esta estratégia é usada para reduzir significativamente o número de comparações de *strings* adicionais, que têm um custo computacional maior. Então, as iniciais dos nomes dos autores são extraídas e comparadas. Somente os objetos que atingirem o limiar de casamento de autores terão seus títulos comparados pela função de distância de edição.

A cooperação com o grupo de pesquisa em banco de dados da Universidade Federal de Minas Gerais (UFMG) propiciou que estas contribuições fossem utilizadas no desenvolvimento do Portal Brasileiro de Ciência e Tecnologia - Ciência Brasil[1] - para identificação dos relacionamentos de coautoria nas redes de colaboração dos Institutos Nacionais de Ciência e Tecnologia (INCT) do país. A arquitetura do portal e as soluções adotadas para deduplicação, entre outras tarefas, estão detalhadas no artigo

> LAENDER, A. H. F. et al. CiênciaBrasil – The Brazilian Portal of Science and Technology. In: SEMINÁRIO INTEGRADO DE SOFTWARE E HARDWARE - SEMISH, 38. **Anais...** [S.l.: s.n.], 2011. p.221–228.

5. Um método efetivo para combinar os escores retornados pelas funções de similaridade previamente desenvolvidas utilizando algoritmos de classificação supervisionados.

O método proposto remove a intervenção humana porque elimina a necessidade de definir limiares de similaridade para as funções. Os experimentos realizados mostram que o método baseado em classificação, utilizando o algoritmo *C4.5*, supera em até 11% a qualidade da deduplicação alcançada pelo método não supervisionado (item 2). Esta contribuição está detalhada no artigo

> BORGES, E. N. et al. An Automatic Approach for Duplicate Bibliographic Metadata Identification Using Classification. In: INTERNATIONAL CONFERENCE OF THE CHILEAN COMPUTER SCIENCE SOCIETY, SCCC, 30. **Proceedings...** [S.l.: s.n.], 2011. p.47–53.

[1] `http://www.pbct.inweb.org.br/pbct`. Data do acesso: 8/11/2013.

6. A análise das características dos modelos de classificação gerados pelos algoritmos *Naïve Bayes*, *RIPPER* e *C4.5*.

Quando estas características são comparadas com as suposições definidas no método não supervisionado, é possível observar que a diferença no ano de publicação é realmente um atributo importante. Entretanto, o mais discriminatório é, de fato, o título de uma referência bibliográfica. Esta contribuição está detalhada no artigo

> BORGES, E. N. et al. A Classification-based Approach for Bibliographic Metadata Deduplication. In: IADIS INTERNATIONAL CONFERENCE WWW/INTERNET. **Proceedings...** [S.l.: s.n.], 2011. p.221–228.

7. uma estratégia de seleção aleatória de exemplos de treinamento baseadas no cálculo amostral e em características do conjunto de dados.

Na especificação do método de deduplicação foi proposta uma estratégia de seleção de exemplos que reduz consideravelmente o tamanho do conjunto de treinamento e mantém as principais características necessárias para o aprendizado efetivo dos modelos de classificação. Os experimentos realizados mostraram que é possível identificar metadados bibliográficos duplicados com até 97% de F1 usando esta estratégia em dois conjuntos de dados.

8. Uma abordagem para empilhar classificadores supervisionados visando deduplicar metadados bibliográficos.

9. A análise do ganho proporcionado pelo empilhamento em relação ao melhor classificador e a outra estratégia de combinação de classificadores (voto da maioria).

Empilhando algoritmos de diversos tipos (baseados em árvores, regras, redes neurais artificiais e probabilísticos), é possível evitar o forte acoplamento a uma determinada técnica de aprendizado. Assim, o método proposto pode ser aplicado nas mais diversas situações, aproveitando o conhecimento heterogêneo dos classificadores empilhados para identificar mais precisamente determinados registros de metadados bibliográficos duplicados.

A hipótese de que o empilhamento de classificadores supervisionados pode aumentar a qualidade da deduplicação quando comparado à escolha do melhor classificador ou ao voto da maioria foi verificada na metade dos conjunto de dados avaliados. O resultado do empilhamento para os outros conjuntos de dados pode ser considerado um empate técnico com as estratégias comparadas visto que não houve perda ou ganho estatisticamente significativo. Além disso, estes resultados já eram muito bons ($F1 = 97\%$) e difíceis de serem superados.

10. A análise do impacto da diversidade dos classificadores no resultado do empilhamento.

11. A análise dos casos de falha do método proposto.

Estas análises permitiram entender melhor o comportamento do método proposto nesta tese. Observando as medidas calculadas, percebeu-se que existe uma relação direta entre o ganho de qualidade do empilhamento e a diversidade dos classificadores envolvidos. Identificando os casos de falha, verificou-se que referências para relatórios técnicos

são confundidas com referências para artigos científicos publicados, não publicados ou aceitos para publicação futura. Esses casos podem ser facilmente identificados em um passo de pré-processamento que opere sobre os metadados que descrevem o veículo de publicação. Os casos mais complicados são aqueles em que ocorrem problemas devido a erros na segmentação dos campos.

Ao longo do desenvolvimento deste trabalho, alguns aspectos do método de deduplicação proposto foram identificados como passíveis de melhorias ou extensões. É possível refinar os experimentos realizados usando diferentes estratégias de empilhamento. TING; WITTEN (1999) afirmam que o algoritmo *Multi-response Linear Regression* (MLR) obtém melhores resultados como classificador do metanível do empilhamento. O uso deste algoritmo combinado com os atributos propostos por DZEROSKI; ZENKO (2004), descritos na Seção 2.3.2, pode aumentar significativamente a qualidade da deduplicação.

Um segunda extensão do método insere uma etapa de *boosting* sobre cada algoritmo de nível básico do empilhamento, conforme demonstrado na Figura 6.1. O algoritmo de *boosting* manipula os exemplos de treinamento com o objetivo de melhorar a acurácia do sistema de classificação. A partir do conjunto de treinamento inicial e um algoritmo de classificação, o algoritmo de *boosting* treina classificadores sequencialmente, de modo que a cada iteração um novo classificador tenta rotular corretamente os erros dos classificadores prévios dando maior peso ω às instâncias rotuladas incorretamente. A acurácia de cada classificador é usada para calcular a importância α na composição de um votador ponderado. A predição final de cada *boosting* é usada para compor os exemplos de treinamento no metanível do empilhamento.

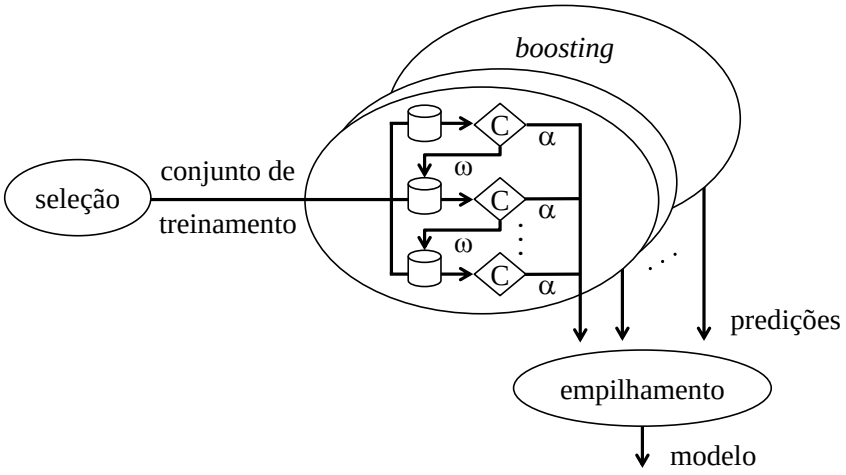

Figura 6.1: Combinação de *boosting* e empilhamento.

Ainda é necessário avaliar as estratégias propostas para seleção de exemplos de treinamento e blocagem. A redução na quantidade de pares candidatos é essencial para questões de escalabilidade, principalmente para algoritmos de aprendizado baseados em função que possuem alta complexidade computacional. Os resultados da qualidade da deduplicação apresentados nesta tese poderão ser comparados aos obtidos futuramente com a etapa de blocagem, possibilitando avaliar o ganho de desempenho frente à perda de qualidade no processo de deduplicação.

Por fim, analisar a distribuição das medidas de diversidade pode revelar comportamentos não demonstrados pela média dos valores apresentada na Seção 5.5.2. Seria interessante plotar dispersões tanto da qualidade da deduplicação quanto das predições dos

classificadores em função de cada medida de diversidade. Este estudo pode revelar uma série de relações entre as medidas estudadas e gerar novas conclusões sobre o impacto da diversidade dos classificadores no empilhamento.

As contribuições apresentadas nos itens 7 a 11 e os resultados de alguns dos trabalhos futuros serão submetidos para um periódico internacional no início de 2014. Além disso, será desenvolvido um projeto de cooperação com o Centro de Ciências Computacionais da FURG para dar continuidade à pesquisa apresentada nesta tese.

REFERÊNCIAS

AHA, D.; KIBLER, D.; ALBERT, M. Instance-based learning algorithms. **Machine Learning**, [S.l.], v.6, n.1, p.37–66, 1991.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Modern Information Retrieval**: the concepts and technology behind search. 2nd.ed. [S.l.]: ACM Press / Addison-Wesley, 2011.

BAXTER, R.; CHRISTEN, P.; CHURCHES, T. A Comparison of Fast Blocking Methods for Record Linkage. In: ACM SIGKDD WORKSHOP DATA CLEANING, RECORD LINKAGE, AND OBJECT CONSOLIDATION. **Proceedings...** [S.l.: s.n.], 2003. p.25–27.

BELLARE, K. et al. Active Sampling for Entity Matching. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, New York. **Proceedings...** ACM, 2012. p.1131–1139. (KDD '12).

BENJELLOUN, O. et al. Swoosh: a generic approach to entity resolution. **The VLDB Journal**, Secaucus, NJ, USA, v.18, n.1, p.255–276, 2009.

BILENKO, M.; MOONEY, R. J. Adaptive duplicate detection using learnable string similarity measures. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING. **Proceedings...** [S.l.: s.n.], 2003. p.39–48.

BORGES, E. N. **Um estudo sobre algoritmos de classificação aplicados à deduplicação**. 2009. Trabalho Individual III, TI-1340 — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

BORGES, E. N. et al. An unsupervised heuristic-based approach for bibliographic metadata deduplication. **Information Processing and Management**, [S.l.], v.47, n.5, p.706–718, 2011.

BORGES, E. N. et al. An Automatic Approach for Duplicate Bibliographic Metadata Identification Using Classification. In: INTERNATIONAL CONFERENCE OF THE CHILEAN COMPUTER SCIENCE SOCIETY, SCCC, 30. **Proceedings...** [S.l.: s.n.], 2011. p.47–53.

BORGES, E. N. et al. A Classification-based Approach for Bibliographic Metadata Deduplication. In: IADIS INTERNATIONAL CONFERENCE WWW/INTERNET. **Proceedings...** [S.l.: s.n.], 2011. p.221–228.

BORGES, E. N.; GALANTE, R. M.; GONÇALVES, M. A. Uma abordagem efetiva e eficiente para deduplicação de metadados bibliográficos de objetos digitais. In: BRAZILIAN SYMPOSIUM ON DATABASES, Porto Alegre. **Proceedings...** Sociedade Brasileira de Computação, 2008. p.76–90.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: COMPUTATIONAL LEARNING THEORY. **Proceedings...** [S.l.: s.n.], 1992. p.144–152.

BREIMAN, L. Bagging Predictors. **Machine Learning**, [S.l.], v.24, n.2, p.123–140, 1996.

BREIMAN, L. et al. **Classification and Regression Trees**. Monterey, CA: Wadsworth and Brooks, 1984.

CARVALHO, J. C. P.; SILVA, A. S. da. Finding similar identities among objects from multiple web sources. In: ACM INTERNATIONAL WORKSHOP ON WEB INFORMATION AND DATA MANAGEMENT. **Proceedings...** [S.l.: s.n.], 2003. p.90–93.

CARVALHO, M. G. de et al. Learning to deduplicate. In: ACM IEEE JOINT CONFERENCE ON DIGITAL LIBRARIES. **Proceedings...** [S.l.: s.n.], 2006. p.41–50.

CARVALHO, M. G. de et al. Replica identification using genetic programming. In: ACM SYMPOSIUM ON APPLIED COMPUTING. **Proceedings...** [S.l.: s.n.], 2008. p.1801–1806.

CARVALHO, M. G. de et al. The impact of parameter setup on a genetic programming approach to record deduplication. In: BRAZILIAN SYMPOSIUM ON DATABASES, Porto Alegre, Brazil. **Proceedings...** Sociedade Brasileira de Computação, 2008. p.91–105.

CARVALHO, M. G. de et al. A Genetic Programming Approach to Record Deduplication. **IEEE Transactions on Knowledge and Data Engineering**, Piscataway, NJ, USA, v.24, n.3, p.399–412, 2012.

CHAUDHURI, S. et al. Robust and efficient fuzzy match for online data cleaning. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA CONFERENCE. **Proceedings...** [S.l.: s.n.], 2003. p.313–324.

CHAUDHURI, S.; GRAVANO, L. Evaluating Top-k Selection Queries. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, San Francisco, CA. **Proceedings...** Morgan Kaufmann Publishers, 1999. p.397–410.

CHEESEMAN, P. et al. AutoClass: a bayesian classification system. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, San Francisco. **Proceedings...** Morgan Kaufmann Publishers, 1988. p.54–64.

CHEN, Z.; KALASHNIKOV, D. V.; MEHROTRA, S. Exploiting context analysis for combining multiple entity resolution systems. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA CONFERENCE, New York, NY, USA. **Proceedings...** ACM, 2009. p.207–218.

CHRISTEN, P. Febrl -: an open source data cleaning, deduplication and record linkage system with a graphical user interface. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, New York, NY, USA. **Proceedings...** ACM, 2008. p.1065–1068.

CHRISTEN, P. Febrl: a freely available record linkage system with a graphical user interface. In: AUSTRALASIAN WORKSHOP ON HEALTH DATA AND KNOWLEDGE MANAGEMENT, Darlinghurst, Australia, Australia. **Proceedings...** Australian Computer Society: Inc., 2008. p.17–25.

CHRISTEN, P. **Data Matching**: concepts and techniques for record linkage, entity resolution, and duplicate detection. [S.l.]: Springer Publishing Company, Incorporated, 2012.

CHRISTEN, P. A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication. **IEEE Transactions on Knowledge and Data Engineering**, [S.l.], v.24, n.9, p.1537–1555, 2012.

CHáVEZ, E. et al. Searching in metric spaces. **ACM Comput. Surv.**, New York, NY, USA, v.33, n.3, p.273–321, 2001.

COHEN, W. W. Fast Effective Rule Induction. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING. **Proceedings...** [S.l.: s.n.], 1995. p.115–123.

COHEN, W. W.; RAVIKUMAR, P.; FIENBERG, S. E. A Comparison of String Distance Metrics for Name-Matching Tasks. In: IJCAI WORKSHOP ON INFORMATION INTEGRATION. **Proceedings...** [S.l.: s.n.], 2003. p.73–78.

COHEN, W. W.; RICHMAN, J. Learning to match and cluster large high-dimensional data sets for data integration. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING. **Proceedings...** [S.l.: s.n.], 2002. p.475–480.

COHN, D.; ATLAS, L.; LADNER, R. Improving Generalization with Active Learning. **Mach. Learn.**, Hingham, MA, USA, v.15, n.2, p.201–221, 1994.

CONVIS, D. B.; GLICKMAN, D.; ROSENBAUM, W. S. **Alpha content match prescan method for automatic spelling error correction**. 1982. n.4328561, U.S. Patent.

COOPER, G. F.; HERSKOVITS, E. A Bayesian Method for the Induction of Probabilistic Networks from Data. **Machine Learning**, Hingham, MA, USA, v.9, n.4, p.309–347, 1992.

COTA, R. G.; GONÇALVES, M. A.; LAENDER, A. H. F. A Heuristic-based Hierarchical Clustering Method for Author Name Disambiguation in Digital Libraries. In: BRAZILIAN SYMPOSIUM ON DATABASES, SBBD, 21., João Pessoa, PB, Brazil. **Proceedings...** SBC, 2007. p.20–34.

DAL BIANCO, G. et al. Tuning large scale deduplication with reduced effort. In: INTERNATIONAL CONFERENCE ON SCIENTIFIC AND STATISTICAL DATABASE MANAGEMENT, New York. **Proceedings...** ACM, 2013. p.18:1–18:12. (SSDBM).

DAL BIANCO, G.; GALANTE, R.; HEUSER, C. A. A fast approach for parallel deduplication on multicore processors. In: ACM SYMPOSIUM ON APPLIED COMPUTING. **Proceedings...** [S.l.: s.n.], 2011. p.1027–1032.

DCMI USAGE BOARD. **DCMI Metadata Terms**. Available at http://dublincore.org/documents/dcmi-terms/. Access date: November 8th, 2013.

DEAN, J.; GHEMAWAT, S. MapReduce: simplified data processing on large clusters. **Communications of the ACM**, [S.l.], v.51, n.1, p.107–113, 2008.

DOAN, A. et al. Profile-based object matching for information integration. **IEEE Intelligent Systems**, [S.l.], v.18, n.5, p.54–59, 2003.

DORNELES, C. F. et al. Measuring similarity between collection of values. In: ANNUAL ACM INTERNATIONAL WORKSHOP ON WEB INFORMATION AND DATA MANAGEMENT, WIDM, 6. **Proceedings...** New York: ACM, 2004. p.56–63.

DORNELES, C. F. et al. A strategy for allowing meaningful and comparable scores in approximate matching. **Information Systems**, [S.l.], v.34, n.8, p.673–689, 2009.

DORNELES, C. F.; GONÇALVES, R.; MELLO, R. S. Approximate data instance matching: a survey. **Knowledge and Information Systems**, [S.l.], v.27, p.1–21, 2011.

DRAISBACH, U. et al. Adaptive Windows for Duplicate Detection. In: IEEE INTERNATIONAL CONFERENCE ON DATA ENGINEERING. **Proceedings...** [S.l.: s.n.], 2012. p.1073–1083.

DZEROSKI, S.; ZENKO, B. Is Combining Classifiers with Stacking Better than Selecting the Best One? **Mach. Learn.**, [S.l.], v.54, n.3, p.255–273, 2004.

ELMAGARMID, A.; IPEIROTIS, P.; VERYKIOS, V. Duplicate Record Detection: a survey. **IEEE Transactions on Knowledge and Data Engineering**, [S.l.], v.19, n.1, p.1–16, 2007.

FAYYAD, U. M.; IRANI, K. B. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE. **Proceedings...** [S.l.: s.n.], 1993. p.1022–1029.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. The KDD process for extracting useful knowledge from volumes of data. **Communications of the ACM**, [S.l.], v.39, n.11, p.27–34, 1996.

FELLEGI, I. P.; SUNTER, A. B. A Theory for Record Linkage. **Journal of the American Statistical Association**, [S.l.], v.64, n.328, p.1183–1210, December 1969.

FOX, E. A. et al. Digital libraries. **Communications of the ACM**, New York, NY, USA, v.38, n.4, p.22–28, 1995.

FREITAS, J. de et al. Active Learning Genetic programming for record deduplication. In: IEEE CONGRESS ON EVOLUTIONARY COMPUTATION. **Proceedings...** [S.l.: s.n.], 2010. p.1–8.

FREUND, Y.; SCHAPIRE, R. E. Experiments with a New Boosting Algorithm. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING. **Proceedings...** [S.l.: s.n.], 1996. p.148–156.

FREUND, Y.; SCHAPIRE, R. E. Large margin classification using the perceptron algorithm. In: ANNUAL CONFERENCE ON COMPUTATIONAL LEARNING THEORY, 11., New York, NY, USA. **Proceedings...** ACM, 1998. p.209–217. (COLT' 98).

GAMA, J.; BRAZDIL, P. Cascade Generalization. **Machine Learning**, [S.l.], v.41, n.3, p.315–343, 2000.

GONÇALVES, M. A. et al. Streams, structures, spaces, scenarios, societies (5s): a formal model for digital libraries. **ACM Transactions on Information Systems**, [S.l.], v.22, n.2, p.270–312, 2004.

GUHA, S. et al. Merging the results of approximate match operations. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES. **Proceedings...** [S.l.: s.n.], 2004. p.636–647.

GUTH, G. J. A. Surname Spellings and Computerized Record Linkage. **Historical Methods Newsletter**, [S.l.], v.10, n.1, p.10–19, December 1976.

HAN, J.; KAMBER, M. **Data Mining**: concepts and techniques. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006.

HAYKIN, S. **Neural Networks**: a comprehensive foundation. Upper Saddle River, USA: Prentice-Hall, Inc., 2007.

HECHT-NIELSEN, R. Theory of the backpropagation neural network. In: INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS. **Proceedings...** [S.l.: s.n.], 1989. p.593–605.

HERNÁNDEZ, M. A.; STOLFO, S. J. Real-world Data is Dirty: data cleansing and the merge/purge problem. **Data Min. Knowl. Discov.**, Hingham, MA, USA, v.2, n.1, p.9–37, 1998.

HOLTE, R. Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. **Machine Learning**, [S.l.], v.11, n.1, p.63–90, 1993.

HOSMER, D.; LEMESHOW, S. **Applied Logistic Regression**. [S.l.]: John Wiley & Sons Inc., 2000.

HOSMER, D. W.; LEMESHOW, S. **Applied logistic regression**. [S.l.]: Wiley-Interscience, 2004. v.354.

HUANG, J.; ERTEKIN, S.; GILES, C. Efficient Name Disambiguation for Large-Scale Databases. In: KNOWLEDGE DISCOVERY IN DATABASES: PKDD 2006. **Anais...** Springer Berlin Heidelberg, 2006. p.536–544. (Lecture Notes in Computer Science, v.4213).

JOHN, G.; LANGLEY, P. Estimating Continuous Distributions in Bayesian Classifiers. In: CONFERENCE IN UNCERTAINTY IN ARTIFICIAL INTELLIGENCE. **Proceedings...** [S.l.: s.n.], 1995. p.338–345.

KIM, J.; LEE, H. Efficient Exact Similarity Searches Using Multiple Token Orderings. In: IEEE INTERNATIONAL CONFERENCE ON DATA ENGINEERING. **Proceedings...** [S.l.: s.n.], 2012. p.822–833.

KITTLER, J. et al. On combining classifiers. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [S.l.], v.20, n.3, p.226–239, 1998.

KOHAVI, R. The power of decision tables. In: MACHINE LEARNING: ECML-95. **Anais...** Springer Berlin Heidelberg, 1995. p.174–189. (Lecture Notes in Computer Science, v.912).

KÖPCKE, H.; RAHM, E. Frameworks for entity matching: a comparison. **Data & Knowledge Engineeering**, [S.l.], v.69, n.2, p.197–210, 2010.

KÖPCKE, H.; THOR, A.; RAHM, E. Evaluation of entity resolution approaches on real-world match problems. **The VLDB Journal**, [S.l.], v.3, n.1-2, p.484–493, 2010.

KÖPCKE, H.; THOR, A.; RAHM, E. Learning-Based Approaches for Matching Web Data Entities. **IEEE Internet Computing**, [S.l.], v.14, n.4, p.23–31, 2010.

KUNCHEVA, L. I.; WHITAKER, C. J. Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy. **Machine Learning**, Hingham, MA, USA, v.51, n.2, p.181–207, 2003.

LANDWEHR, N.; HALL, M.; FRANK, E. Logistic Model Trees. **Mach. Learn.**, Hingham, MA, USA, v.59, n.1-2, p.161–205, 2005.

LAWRENCE, S.; GILES, C. L.; BOLLACKER, K. Digital Libraries and Autonomous Citation Indexing. **IEEE Computer**, [S.l.], v.32, n.6, p.67–71, 1999.

LEVENSHTEIN, V. I. Binary codes capable of correcting deletions, insertions and reversals. **Soviet Physics Doklady**, [S.l.], v.10, n.8, p.707–710, 1966.

LEY, M. The DBLP Computer Science Bibliography: evolution, research issues, perspectives. In: INTERNATIONAL STRING PROCESSING AND INFORMATION RETRIEVAL SYMPOSIUM. **Proceedings...** Springer, 2002. p.481–486. (Lecture Notes in Computer Science, v.2476).

LIBRARY OF CONGRESS. **MARC 21 Format for Bibliographic Data**. Available at http://www.loc.gov/marc/bibliographic/. Access date: November 8th, 2013.

LIMA, A. E. N. **Pesquisa de similaridade em XML**. Monografia de Graduação em Ciência da Computação, Instituto de Informática, UFRGS, Porto Alegre.

LIN, X. et al. Performance analysis of pattern classifier combination by plurality voting. **Pattern Recognition Letters**, [S.l.], v.24, n.12, p.1959–1969, 2003.

MANNING, C. D.; RAGHAVAN, P.; SCHUTZE, H. **Introduction to Information Retrieval**. [S.l.]: Cambridge University Press, 2008.

MCCALLUM, A. K. et al. Automating the Construction of Internet Portals with Machine Learning. **Information Retrieval**, [S.l.], v.3, n.2, p.127–163, 2000.

MCCALLUM, A.; NIGAM, K.; UNGAR, L. H. Efficient clustering of high-dimensional data sets with application to reference matching. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, New York, NY, USA. **Proceedings...** ACM, 2000. p.169–178.

MITCHELL, T. **Machine Learning**. [S.l.]: McGraw Hill, 1997.

NAVARRO, G. A guided tour to approximate string matching. **ACM Comput. Surv.**, New York, NY, USA, v.33, n.1, p.31–88, Mar. 2001.

NEVILLE, C. **The Complete Guide to Referencing and Avoiding Plagiarism**. [S.l.]: McGraw-Hill, 2010.

NIGAM, K.; LAFFERTY, J.; MCCALLUM, A. Using Maximum Entropy for Text Classification. In: IJCAI WORKSHOP ON INFORMATION FILTERING. **Proceedings...** [S.l.: s.n.], 1999. p.61–67.

ON, B.-W. et al. Comparative study of name disambiguation problem using a scalable blocking-based framework. In: ACM/IEEE-CS JOINT CONFERENCE ON DIGITAL LIBRARIES, 5., New York, NY, USA. **Proceedings...** ACM, 2005. p.344–353.

PEREIRA, D. A. et al. Using web information for author name disambiguation. In: ACM/IEEE-CS JOINT CONFERENCE ON DIGITAL LIBRARIES, 9., New York, NY, USA. **Proceedings...** ACM, 2009. p.49–58.

PLATT, J. C. Using analytic QP and sparseness to speed training of support vector machines. In: CONFERENCE ON ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS. **Proceedings...** [S.l.: s.n.], 1999. p.557–563.

QUINLAN, J.; CAMERON-JONES, R. **FOIL**: a midterm report. [S.l.]: Springer Berlin Heidelberg, 1993. 1–20p. (Lecture Notes in Computer Science, v.667).

QUINLAN, J. R. Induction of Decision Trees. **Machine Learning**, [S.l.], v.1, n.1, p.81–106, 1986.

QUINLAN, J. R. **C4.5**: programs for machine learning. San Francisco, USA: Morgan Kaufmann Publishers Inc., 1993.

RAHM, E.; BERNSTEIN, P. A. A survey of approaches to automatic schema matching. **The VLDB Journal**, Secaucus, NJ, USA, v.10, n.4, p.334–350, 2001.

SARIYAR, M.; BORG, A.; POMMERENING, K. Missing values in deduplication of electronic patient data. **Journal of the American Medical Informatics Association**, [S.l.], v.19, n.e1, p.e76–e82, 2012.

STASIU, R. K.; HEUSER, C. A.; SILVA, R. da. Estimating Recall and Precision for Vague Queries in Databases. In: INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION SYSTEMS ENGINEERING, CAISE, 17. **Proceedings...** [S.l.: s.n.], 2005. p.187–200.

STUDENT. The probable error of a mean. **Biometrika**, [S.l.], v.6, n.1, p.1–25, 1908.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining**. [S.l.]: Addison-Wesley, 2005.

TEJADA, S.; KNOBLOCK, C. A.; MINTON, S. Learning object identification rules for information integration. **Information Systems**, [S.l.], v.26, n.8, p.607–633, 2001.

THOR, A.; RAHM, E. MOMA-A mapping-based object matching system. In: BIENNIAL CONFERENCE ON INNOVATIVE DATA SYSTEMS RESEARCH. **Proceedings...** [S.l.: s.n.], 2007. p.247–258.

TING, K. M.; WITTEN, I. H. Stacked generalization: when does it work? In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICAL INTELLIGENCE. **Proceedings...** Morgan Kaufmann Publishers Inc., 1997. p.866–871. (IJCAI'97).

TING, K. M.; WITTEN, I. H. Issues in Stacked Generalization. **Journal of Artificial Intelligence Research**, [S.l.], v.10, p.271–289, 1999.

TORVIK, V. I.; SMALHEISER, N. R. Author name disambiguation in MEDLINE. **ACM Trans. Knowl. Discov. Data**, New York, NY, USA, v.3, n.3, p.11:1–11:29, 2009.

TREERATPITUK, P.; GILES, C. L. Disambiguating authors in academic publications using random forests. In: ACM/IEEE-CS JOINT CONFERENCE ON DIGITAL LIBRARIES, New York, NY, USA. **Proceedings...** ACM, 2009. p.39–48.

TRIOLA, M. F. et al. **Introdução à Estatística**: atualização da tecnologia. Rio de Janeiro: LTC, 2013. v.11.

VERYKIOS, V. S.; ELMAGARMID, A. K.; HOUSTIS, E. N. Automating the approximate record-matching process. **Information Sciences**, [S.l.], v.126, n.1?4, p.83–98, 2000.

WANG, J.; LI, G.; FENG, J. Can We Beat the Prefix Filtering?: an adaptive framework for similarity join and search. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA CONFERENCE, New York. **Proceedings...** ACM, 2012. p.85–96. (SIGMOD '12).

WHANG, S.; GARCIA-MOLINA, H. Joint Entity Resolution. In: IEEE INTERNATIONAL CONFERENCE ON DATA ENGINEERING. **Proceedings...** [S.l.: s.n.], 2012. p.294–305.

WHANG, S.; GARCIA-MOLINA, H. Joint entity resolution on multiple datasets. **The VLDB Journal**, [S.l.], v.22, n.6, p.773–795, 2013.

WINKLER, W. E. String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. In: SECTION ON SURVEY RESEARCH METHODS, AMERICAN STATISTICAL ASSOCIATION. **Proceedings...** [S.l.: s.n.], 1990. p.354–359.

WITTEN, I. H.; FRANK, E.; HALL, M. A. **Data Mining**: practical machine learning tools and techniques. [S.l.]: Morgan Kaufmann, 2011.

WOLPERT, D. H. Stacked generalization. **Neural Networks**, [S.l.], v.5, n.2, p.241–259, 1992.

XIAO, C. et al. Efficient similarity joins for near-duplicate detection. **ACM Trans. Database Syst.**, New York, NY, USA, v.36, n.3, p.15:1–15:41, 2011.

ZHAO, H. Instance weighting versus threshold adjusting for cost-sensitive classification. **Knowledge and Information Systems**, [S.l.], v.15, p.321–334, 2008.

ZHAO, H.; RAM, S. Constrained cascade generalization of decision trees. **IEEE Transactions on Knowledge and Data Engineering**, [S.l.], v.16, n.6, p.727–739, 2004.

ZHAO, H.; RAM, S. Entity identification for heterogeneous database integration - a multiple classifier system approach and empirical evaluation. **Information Systems**, [S.l.], v.30, n.2, p.119 – 132, 2005.

ZHAO, H.; RAM, S. Entity matching across heterogeneous data sources: an approach based on constrained cascade generalization. **Data & Knowledge Engineeering**, [S.l.], v.66, n.3, p.368 – 381, 2008.

ZOBEL, J.; DART, P. Phonetic string matching: lessons from information retrieval. In: ACM SIGIR CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL, New York. **Proceedings...** ACM, 1996. p.166–172.