UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

Filipe Roberto Silva

DEFINIÇÃO E AVALIAÇÃO DE MÉTODOS PARA DETERMINAÇÃO DE SIMILARIDADE ENTRE TABELAS NA WEB

Florianópolis

2015

Filipe Roberto Silva

DEFINIÇÃO E AVALIAÇÃO DE MÉTODOS PARA DETERMINAÇÃO DE SIMILARIDADE ENTRE TABELAS NA WEB

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Ciência da Computação.
Ronaldo dos Santos Mello
Universidade Federal de Santa Catarina: Prof. Dr.

Florianópolis

2015

Ficha de identificação da obra elaborada pelo autor através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

A ficha de identificação é elaborada pelo próprio autor

Maiores informações em:
http://portalbu.ufsc.br/ficha

Filipe Roberto Silva

DEFINIÇÃO E AVALIAÇÃO DE MÉTODOS PARA DETERMINAÇÃO DE SIMILARIDADE ENTRE TABELAS NA WEB

Esta Dissertação foi julgada aprovada para a obtenção do Título de "Mestre em Ciência da Computação", e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina.

Florianópolis, 11 de Setembro 2015.

Prof. Ronaldo dos Santos Mello, Dr.
Coordenador do Programa

Banca Examinadora:

Prof. Ronaldo dos Santos Mello, Dr.
Orientador
Universidade Federal de Santa Catarina

Prof. Denio Duarte, Dr.
Universidade Federal da Fronteira Sul

Prof. Frank Augusto Siqueira, Dr.
Universidade Federal de Santa Catarina

Prof.ª Carina Friedrich Dorneles, Dr.ª
Universidade Federal de Santa Catarina

Este trabalho é dedicado aos meus amigos
e aos meus queridos pais.

AGRADECIMENTOS

Agradeço ao meu orientador Ronaldo pela disposição e atenção durante a realização deste trabalho. Aos colegas de laboratório pela ajuda e pela amizade. Aos meus pais pelo incentivo, apoio e paciência, e a Deus.

RESUMO

A Web é uma grande fonte de dados. Grandes quantidades de dados são inseridos diariamente e muitos desses dados estão na forma de tabelas HTML. Alguns trabalhos têm proposto formas de extrair e integrar o conteúdo dessas tabelas para torná-los mais acessíveis para o consumo humano. Porém, essa é uma tarefa complexa e um problema ainda em aberto visto que tabelas Web não possuem um padrão de representação. Além disso, o uso de sinônimos e abreviações torna difícil a comparação dos conteúdos dessas tabelas. Assim sendo, este trabalho propõe uma nova abordagem para determinar a similaridade entre tabelas Web capaz de lidar com suas diferentes estruturas e termos sinônimos. Trabalhos relacionados não lidam, ao mesmo tempo, com essas duas problemáticas. Experimentos realizados mostram que a abordagem é promissora.

Palavras-chave: Tabelas Web, Similaridade, Comparação

ABSTRACT

The Web is a huge information source. Large amounts of data are published daily and great part of them is available as HTML tables. Some works have proposed approaches to extract and integrate Web tables' content in order to make it more accessible for human consumption. However, this is a complex task and still an open issue given that Web tables do not have a unique representation pattern. Besides, the use of synonyms and abbreviations become hard the comparison of tables' content. Given that, we propose a new approach to determine similarity between Web tables which is able to deal with distinct structures and synonym terms. Related works do not deal, at the same time, with both problematics. Experimental evaluations had shown that the approach is promising.

Keywords: Web Tables, Matching, Similarity

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A Web é uma vasta fonte de informação. Muitos dados são publicados diariamente e boa parte deles está disponível na forma de tabelas HTML. Já em 2008, Cafarella et al. (2008) mostraram que, de 14.1 bilhões de tabelas extraídas da Web, 154 milhões continham dados relacionais de alta qualidade, ou seja, tabelas com dados relevantes em um domínio do conhecimento e que estão no formato relacional. Balakrishnan et al. (2015) estimam que atualmente existam dezenas de bilhões dessas tabelas somente na língua inglesa.

As tabelas na Web consideradas neste trabalho são caracterizadas pela *tag* `<table>` em páginas HTML e, como qualquer outra tabela, podem conter dados e cabeçalhos. Alguns trabalhos têm estudado formas de extrair e integrar informações presentes nessas tabelas (EMBLEY et al., 2011; MERGEN; FREIRE; HEUSER, 2010; VENETIS et al., 2011; LAI, 2013; WANG et al., 2012; FAN et al., 2014). Nesse processo de recuperação de informação sobre tabelas na Web, um método que compare essas tabelas e mostre quais são similares entre si, pode ser de grande auxílio. Esse método poderia ser utilizado, por exemplo, para agrupar tabelas por conteúdo, ou até mesmo indexar essas tabelas, o que tornaria um processo de busca por tabelas muito mais eficiente. Além disso, sabendo-se quais tabelas são similares entre si, seria possível integrar seus dados com muito mais confiabilidade.

A identificação de tabelas similares é bastante relevante para diversas áreas de pesquisa, como integração de dados na Web, preenchimento de tabelas incompletas utilizando dados de tabelas similares e busca por tabelas semelhantes na Web. Por exemplo, considere duas tabelas de filmes, uma contendo os atores que trabalharam em filmes e outra contendo dados de gêneros de filmes e suas classificações. Caso houvesse um suporte para o reconhecimento de tabelas similares, seria possível, por exemplo, constatar que ambas as tabelas possuem dados sobre os mesmos filmes e unificar os dados contidos nelas. Entretanto, pela falta de esquema, padrões e metadados na Web, esta é uma tarefa complexa e um tema ainda em aberto na literatura.

Venetis et al. (2011) descrevem um sistema que visa recuperar a semântica das tabelas na Web acrescentando nelas anotações. O objetivo principal desse sistema é contribuir com as buscas por tabelas na Web. Porém, esse trabalho não trata diretamente da comparação de tabelas. Fan et al. (2014) por sua vez, buscam recuperar o esquema das tabelas na Web e com isso descobrir quais tabelas possuem o mesmo es-

quema. Porém, esse trabalho considera somente tabelas horizontais, no estilo relacional, enquanto que tabelas na Web podem ser construídas de diversas formas. Mais detalhes sobre estruturas de tabelas na Web encontram-se na Seção 2.2.1 do Capítulo 2. Pawlik e Augsten (2011) apresentam um algoritmo de comparação de árvores que poderia ser adaptado para comparações de tabelas HTML. Entretanto, esse trabalho não leva em consideração termos sinônimos existentes nas tabelas.

Crestan e Pantel (2011) estimam que 88% das tabelas HTML são utilizadas para definir a estrutura de páginas Web e não possuem dados úteis para um processo de recuperação de informação. Por isso, nesse processo, tabelas desse tipo teriam que ser filtradas, visto que não possuem dados relevantes.

Além dessas tabelas ditas de *layout*, tabelas de dados não possuem um padrão único de construção. Este é um dos principais problemas para a recuperação correta das suas informações, visto que é necessário detectar onde e como os dados são apresentados.

Aliado a isso, existe a questão do uso de sinônimos e abreviações. Por exemplo, considere duas tabelas de descrição de carros, sendo que uma possui a coluna *make* e a outra *manufacture*. Ambas podem ter o mesmo esquema, porém com termos distintos. Isso torna difícil a comparação por similaridade entre os conteúdos das tabelas, visto que é preciso informar, de alguma maneira, ao sistema que realiza a comparação de tabelas por similaridade, que aqueles termos significam a mesma coisa.

Assim sendo, este trabalho possui como objetivo geral a criação de um método para determinação de similaridade entre tabelas na Web capaz de lidar com suas estruturas heterogêneas e termos sinônimos.

Para chegar a esse objetivo geral, foram utilizados os seguintes objetivos específicos:

- Criação de uma forma de padronizar os tipos de tabelas na Web;
- Criação de métodos de comparação de tabelas na Web;
- Avaliação experimental dos métodos propostos;
- Identificação dos melhores métodos e configurações de comparação de tabelas na Web.

Trabalhos relacionados não lidam, ao mesmo tempo, com todas as representações existentes para tabelas na Web e com o tratamento de sinônimos. Essa é a principal contribuição desta dissertação. A intenção é definir uma estratégia simples e efetiva para comparar tabelas

Web, tomando como base métricas de similaridade para dados textuais conhecidas na literatura e sua adaptação para este propósito.

Por similaridade entre tabelas, entende-se aqui como tabelas que tratam de um mesmo domínio do conhecimento e que compartilham algumas propriedades. As tabelas 1 e 2, por exemplo, possuem dados diferentes e a maioria dos seus atributos de cabeçalho são diferentes. Mesmo assim, ambas tratam de filmes de 2014 e possuem os atributos *Genre* e *Name* (*Title*) em comum. Isso as torna similares.

Tabela 1: Filmes de Janeiro de 2014

Title	**Studio**	**Genre**	**Directors**
The Lego Movie	Warner Bros.	Action, comedy	Phil Lord, Christopher Miller
Barefoot	Roadside Attractions	Romantic, comedy, drama	Andrew Fleming
The Monuments Men	Columbia Pictures / 20th Century Fox	Drama, war	George Clooney

Fonte: Wikitables (http://downey-n1.cs.northwestern.edu/public/)

Tabela 2: Filmes de Fevereiro de 2014

Name	**Distributor**	**Genre**
Jamesy Boy	Phase 4 Films	Crime, drama
Paranormal Activity: The Marked Ones	Paramount Pictures	Horror
Dumbbells	GoDigital	Comedy

Fonte: Wikitables (http://downey-n1.cs.northwestern.edu/public/)

Portanto, a proposta deste trabalho visa comparar tabelas na Web tratando suas diferentes estruturas, além de detectar o uso de termos sinônimos entre tabelas. Vários métodos de comparação e configurações diferentes para estes métodos foram testados para se alcançar resultados com melhor qualidade. Essas diferentes configurações se deram, por exemplo, através da modificação das funções de similaridade, dos pesos, bem como das características utilizadas nas comparações entre tabelas. A intenção com isso foi avaliar diversas métricas de similaridade, que podem ser selecionadas conforme o domínio dos dados, assim como o que considerar, em termos da estrutura das tabelas, na comparação. Uma avaliação experimental demonstrou que a proposta

é promissora quando comparada com alguns trabalhos relacionados e também com métodos que não levam em conta estruturas heterogêneas das tabelas envolvidas na comparação.

A seguir, no Capítulo 2 é apresentada uma fundamentação teórica sobre tabelas na Web e funções de similaridade. O Capítulo 3 apresenta os trabalhos relacionados. Os Capítulos 4 e 5 detalham a proposta e apresentam a avaliação experimental, respectivamente. Por fim, o Capítulo 6 é dedicado à conclusão.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 SIMILARIDADE E DISTÂNCIA

Uma melhor compreensão da abordagem para determinação de similaridade entre tabelas na Web requer um entendimento dos conceitos de similaridade e distância. Considerando a definição de entidade dada em Definição 2.1.1, tem-se que, segundo Chen, Ma e Zhang (2007), similaridade e distância são medidas amplamente utilizadas em diversos tópicos de pesquisa para comparar entidades.

Definição 2.1.1 *(Entidade) Uma entidade é uma abstração de algo do mundo real, podendo ter suas características descritas e modeladas para o armazenamento em um banco de dados, como por exemplo, uma sequência de DNA ou um filme.*

Segundo Chen, Ma e Zhang (2009), uma função de distância mede a proximidade entre duas entidades, sendo que quanto mais distantes (maior o valor de distância), mais distintas são as entidades e, quanto mais próximo de zero o valor de distância, mais semelhantes são as entidades, conforme a Definição 2.1.2. Neste caso, o intervalo de valores da função é $[0, \infty)$.

Definição 2.1.2 *(Função de Distância) Dadas duas entidades x e y e uma função de distância d tal que $d(x, y) \geq 0$, d será igual a zero se e psomente se $x = y$, sendo d maior que zero, caso contrário.*

Uma função de similaridade, por sua vez, mede a similaridade entre duas entidades e é o inverso da função de distância, conforme a Definição 2.1.3. Chen, Ma e Zhang (2007) apresenta a definição de métrica de similaridade normalizada, que é a utilizada neste trabalho. Ela aplica um intervalo limite para os valores de similaridade, ou seja, dadas duas entidades x e y, uma função de similaridade s é tal que $0 \leq s(x, y) \leq 1$, sendo que $s(x, y) = 1$ se e somente se $x = y$. Neste caso, o intervalo de valores da função é $[0, 1]$.

Definição 2.1.3 *(Função de Similaridade) Dadas duas entidades x e y, uma função de similaridade s é tal que $0 \leq s(x, y) \leq 1$, sendo que $s(x, y) = 1$ se e somente se $x = y$.*

Este trabalho utiliza principalmente métricas de similaridade. Porém, algumas vezes é necessário converter valores de distância para

similaridade. Para tanto, é utilizada a conversão simples, para distâncias não normalizadas, apresentada pela Equação 2.1.

$$s(x, y) = \frac{1}{d(x, y) + 1} \tag{2.1}$$

Apesar desses conceitos, é importante ressaltar que a definição de similaridade é relativa. Dizer que duas entidades são semelhantes ou não depende da opinião do observador. Por exemplo, é possível dizer que duas imagens são semelhantes baseando-se em suas cores. Porém, de um outro ponto de vista, elas podem ser diferentes por suas formas, ou pelo que representam. Por isso, na comparação entre entidades, é necessário especificar os parâmetros de comparação e o que define a similaridade entre elas. A similaridade entre tabelas considerada neste trabalho, como comentado anteriormente, refere-se à similaridade entre os domínios das tabelas. Esses domínios são identificados principalmente a partir dos dados e dos cabeçalhos presentes nas tabelas. Mais detalhes são apresentados nas Seções 4.3.2 e 4.3.3 do Capítulo 4.

2.2 TABELAS NA WEB

As tabelas na Web consideradas nesta dissertação são somente as tabelas delimitadas pela *tag* `<table>` em uma página HTML. Seguindo as convenções propostas por Lautert, Scheidt e Dorneles (2013), tem-se que o índice x denota as linhas de um tabela e o índice y as suas colunas. Assim sendo, conceitos relacionados a tabelas na Web, relevantes para este trabalho, são definidos a seguir.

Definição 2.2.1 *(Célula). Uma célula c_{xy} é uma intersecção entre uma linha e uma coluna de uma tabela, tal que $c_{xy} = h|d$, onde h é um cabeçalho (header) e d um dado (data).*

Um exemplo para a definição 2.2.1, baseado em Lautert, Scheidt e Dorneles (2013), é mostrado na Tabela 3.

Definição 2.2.2 *(Tabela na Web). Uma tabela t na Web é definida como $t = \{c_{xy} | x > 0 \wedge y > 0 \wedge \forall c_{xy}(c_{xy} \in C)\}$, sendo C o conjunto de células consecutivas da tabela e c_{xy} uma célula de C.*

Pela definição 2.2.2, tem-se que uma tabela é um conjunto de células consecutivas.

Tabela 3: Conteúdo de uma Tabela.

c_{11}	c_{12}	...	c_{1y}
c_{21}	c_{22}	...	c_{2y}
⋮	⋮		⋮
c_{x1}	c_{x2}	...	c_{xy}

Fonte: Lautert, Scheidt e Dorneles (2013)

Definição 2.2.3 *(Cabeçalho). Um cabeçalho h em uma tabela na Web t é um conteúdo de uma célula que descreve uma propriedade de uma entidade representada por t.*

A Figura 1 por exemplo apresenta uma tabela com diversos cabeçalhos como "Marca", "GPU", "Modelo", "Memória" dentre outros, que descrevem propriedades de placas de vídeo, o qual é a entidade representada pela tabela.

Figura 1: Guia de Compra Placa de Vídeo

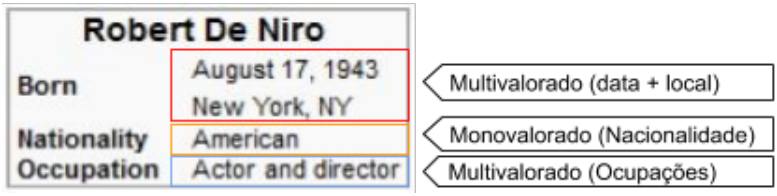
Marca	GPU	Modelo	Velocidade do GPU	Memória	Taxa de Transf	Preço
PowerColor	AMD	HD3450	600MHz	256 MB/64bits	8GB/s	R$170
XFX	AMD	HD4350	600MHz	512 MB/64bits	8GB/s	R$220
HIS Hightech	AMD	HD4650	600MHz	512 MB/128bits	22,4GB/s	R$250
XFX	NVIDIA	8400GS	450MHz	512 MB/64bits	6,4GB/s	R$177
XFX	NVIDIA	8600GT	540MHz	512 MB/128bits	12,8GB/s	R$221
Zotac	NVIDIA	9400GT	550MHz	512 MB/128bits	12,8GB/s	R$240

Fonte: Tecmundo (http://www.tecmundo.com.br/video-game-e-jogos/1820-guia-de-compra-placa-de-video.htm)

Definição 2.2.4 *(Dado). Um dado em uma tabela na Web é um conteúdo de uma célula que não é cabeçalho, sendo definido como d = mono|multi|t, onde mono é um conteúdo monovalorado, multi é um conteúdo multivalorado e t é uma tabela na Web.*

Definição 2.2.5 *(Tipos de Dados). Dados de uma tabela na Web podem ser de três tipos: monovalorados, multivalorados e tabelas, como apresentado na Definição 2.2.4, sendo monovalorado um dado que possui somente um valor em uma célula, multivalorado um dado que possui*

Figura 2: Tipos de dados



Fonte: Lautert, Scheidt e Dorneles (2013)

múltiplos valores em uma célula e tabela um dado que mantém uma tabela na Web em uma célula.

Exemplos de dados monovalorados e multivalorados estão destacados na Figura 2. A Figura 3 apresenta exemplos de dados do tipo tabela, sendo que existe uma tabela externa que possui duas grandes células. Cada uma dessas células, por sua vez, possui tabelas internas. Essas tabelas internas são consideradas dados do tipo tabela, pois estão contidas em células de outra tabela.

2.2.1 Taxonomia

Além dos conceitos já definidos, as tabelas na Web possuem outra particularidade que é a falta de padrão em sua forma de construção. Por isso Lautert, Scheidt e Dorneles (2013) sugerem uma taxonomia para tabelas na Web. O trabalho utiliza uma classificação primária proposta por Crestan e Pantel (2011) apresentada na Figura 4a. Essa classificação é mutuamente exclusiva e se divide entre tabelas de dados relacionais e tabelas de *layout*. Tabelas de *layout* são divididas em:

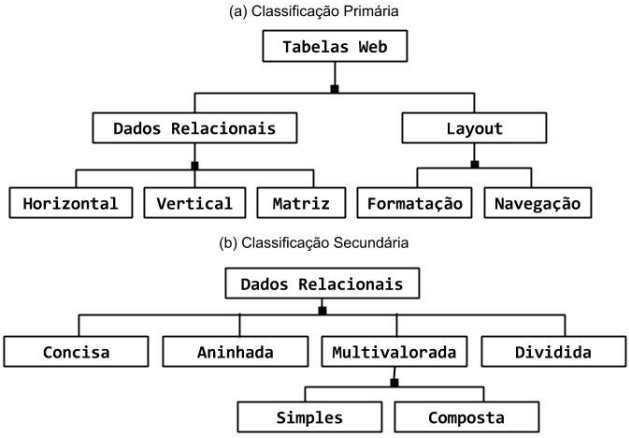
- **Navegação**: tabelas utilizadas para navegação em um sítio na Web, como por exemplo, categorias de produtos disponíveis;
- **Formatação**: tabelas utilizadas para organizar visualmente os elementos de uma página.

Tabelas relacionais, por sua vez, são divididas em:

Figura 3: Dados do tipo tabela

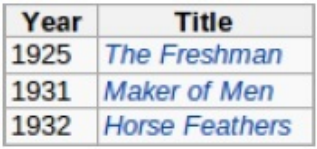
Fonte: Wikipedia: Campeonato Brasileiro de Futebol

Figura 4: Classificação dos tipos de tabelas na Web.



Fonte: Lautert, Scheidt e Dorneles (2013)

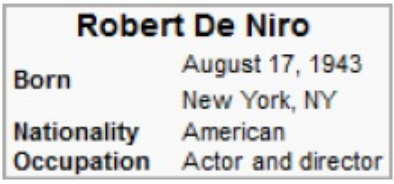
Figura 5: Tabela horizontal



Year	Title
1925	*The Freshman*
1931	*Maker of Men*
1932	*Horse Feathers*

Fonte: Lautert, Scheidt e Dorneles (2013)

Figura 6: Tabela vertical



Robert De Niro	
Born	August 17, 1943 New York, NY
Nationality	American
Occupation	Actor and director

Fonte: Lautert, Scheidt e Dorneles (2013)

- **Vertical**: tabelas cujos cabeçalhos estão dispostos na direção vertical;
- **Horizontal**: tabelas cujos cabeçalhos estão dispostos na direção horizontal;
- **Matriz**: tabelas que possuem cabeçalhos tanto na direção vertical quanto na direção horizontal e no cruzamento entre os dois encontram-se os dados. Elas são utilizadas para correlacionar duas propriedades, por exemplo, número de acidentes por mês para cada estado.

Exemplos de tabela horizontal e vertical são apresentados nas Figuras 5 e 6, respectivamente. A Figura 7 mostra um exemplo de tabela do tipo Matriz.

Além dessa classificação primária, Lautert, Scheidt e Dorneles (2013) propõem uma classificação secundária, não mutualmente exclusiva, que melhor detalha as tabelas relacionais. Essa classificação pode ser vista na Figura 4b e define os seguintes grupos:

- **Concisa**: tabelas que possuem células mescladas;

Figura 7: Exemplo de tabela do tipo matriz.

Cause	1980s	1990s	2000s
Pilot Error	26	27	30
Weather	14	10	8
Mechanical Failure	20	18	24

Fonte: Lautert, Scheidt e Dorneles (2013)

- **Aninhada**: tabelas que possuem tabelas Web internas;
- **Dividida**: tabelas que, por questões de espaço, são divididas horizontal ou verticalmente, sendo suas partes posicionadas lado a lado ou uma abaixo (ou acima) da outra;
- **Multivalorada Simples**: tabelas com múltiplos valores de um mesmo domínio em uma célula. As Tabelas 1 e 2, por exemplo, podem apresentar diversos valores referentes a gêneros de filmes na coluna *Genre*;
- **Multivalorada Composta**: tabelas com múltiplos valores de domínios diferentes em uma célula. A tabela da Figura 6 por exemplo, possui um campo *Born* onde existem dados de data e local. Tabelas com células contendo uma frase ou um texto completo também se enquadram nessa classificação.

Esta dissertação toma como base essa taxonomia para identificar os principais tipos de tabelas e características que destacam cada tipo, visando a construção dos métodos de comparações de tabelas. Maiores detalhes podem ser encontrados no Capítulo 4.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta uma breve descrição dos trabalhos mais recentes relacionados à problemática de análise de tabelas na Web. Ela é dividida em três subseções que classificam estes trabalhos quanto ao seu objetivo: identificação de estrutura, identificação de semântica e comparação de tabelas. Este levantamento do estado da arte foi alvo de um trabalho anterior (SILVA; MELLO, 2014).

3.1 IDENTIFICAÇÃO DE ESTRUTURA

Esta seção descreve trabalhos que analisam a estrutura de tabelas na Web, identificando cabeçalhos, dados e disposição das mesmas.

Lautert, Scheidt e Dorneles (2013) além da taxonomia apresentada na Seção 2.2.1, propõem o *WTClassifier*, uma abordagem baseada em uma rede neural supervisionada para classificar as tabelas. Esse classificador utiliza o *framework Neuroph* (SEVARAC et al., 2008) e é treinado a partir de padrões definidos para cada tipo de tabela. Estes padrões são especificados a partir de características como posição de tabelas internas, listas ordenadas ou não ordenadas dentro das tabelas, pontuações, dentre outras características. O classificador é alimentado com essas características extraídas de tabelas reais e, como resultado, o sistema cria um neurônio para cada categoria de tabela. A partir disso, classificam-se as tabelas de acordo com a taxonomia apresentada na Seção 2.2.1.

Son e Park (2013) propõem um método de classificação de tabelas na Web entre relacionais e *layout*. O trabalho afirma que tabelas possuem informações de conteúdo e estruturais, porém, nem sempre é simples identificar as características estruturais das tabelas. Os autores afirmam ainda que existem dois tipos de informação estrutural. Uma delas consiste nas *tags* que constituem as tabelas e suas relações. O outro tipo consiste no contexto onde a tabela está inserida, ou seja, relações entre as *tags* internas e externas à tabela. No método proposto, essas características juntamente com as informações de conteúdo, são processadas separadamente por algoritmos de análise de padrões. Por fim, esses padrões são utilizados para treinar máquinas de vetores de suporte (SVM) que verificam quais características melhor definem uma tabela de dados ou de layout. A partir de um modelo treinado é possível utilizar SVM para classificar novas tabelas.

Lai (2013) propõe um método para extrair a estrutura de tabelas na Web e reorganizá-las para melhorar a acessibilidade de usuários com deficiência visual. O trabalho afirma que o modo mais comum para essas pessoas acessarem a Web é através de softwares que traduzem texto em fala. Entretanto, esses sistemas simplesmente falam o que está na tela de forma linear, o que dificulta o entendimento do conteúdo das tabelas. Devido a isso, o trabalho visa a extração da estrutura dessas tabelas para recuperar suas informações e melhor apresentá-las aos deficientes visuais.

Primeiramente, classifica-se as tabelas em *layout* e dados. Para isso, são verificadas similaridades das células das tabelas horizontalmente e verticalmente, o que é chamado no trabalho de *Hparallel* e *Vparallel*. Essa similaridade é determinada através de características visuais (como estilos CSS) e de texto utilizando funções de similaridade. Dependendo da similaridade dos dados, ou como eles se relacionam verticalmente e horizontalmente, é possível descobrir como os dados estão dispostos e se as tabelas são de dados ou de *layout*.

Essas células similares horizontalmente e verticalmente são comparadas com o total de colunas ou linhas e células totais. Dessa relação de células similares com os totais de células, descobre-se quais células são similares verticalmente ou horizontalmente (*Vparallel* ou *Hparallel*), e a partir da quantidade dessas células similares classifica-se a tabela em tabela de *layout* ou de dados. O método também busca por linhas ou colunas que possuam menor similaridade com o restante da tabela para identificar cabeçalhos ou rodapés.

Uma vez identificadas as estruturas das tabelas, é possível transformá-las em estruturas mais simples de serem interpretadas por sistemas de leitura para deficientes visuais. Lai (2013) explica que os sistemas de leitura interpretam essas tabelas lendo linha por linha ou coluna por coluna, dependendo da disposição da tabela, e associando o cabeçalho à célula que está sendo lida.

3.2 IDENTIFICAÇÃO DE SEMÂNTICA

Esta seção apresenta trabalhos cujo foco está em obter a semântica das tabelas na Web, como por exemplo, identificar o domínio dos dados e suas entidades.

O trabalho de Balakrishnan et al. (2015) visa criar um repositório com tabelas HTML de alta qualidade para uso por sistemas de busca por dados tabulares na Web, sistemas esses integrados a diversas

aplicações da Google. O sistema proposto realiza tanto a identificação da estrutura das tabelas, diferenciando tabelas de formatação de tabelas de dados, quanto a identificação de semântica, criando anotações para as colunas das tabelas.

Quanto à identificação de tabelas de qualidade, o trabalho utiliza, em um primeiro momento, heurísticas simples de identificação de tabelas irrelevantes, como tabelas muito pequenas, tabelas com formulários, calendários e etc. De acordo com os autores, estas heurísticas eliminam cerca de 90% das tabelas de baixa qualidade. A seguir, utiliza-se uma técnica de aprendizado de máquina para filtrar o restante das tabelas. O trabalho considera como tabelas de qualidade tanto tabelas horizontais quanto verticais. Assim sendo, o sistema é treinado utilizando características estruturais das tabelas, como número de linhas, número de colunas e fração de células vazias, bem como aspectos semânticos, como a relação da tabela com a página em que está inserida e conceitos associados às colunas. Após o treinamento é realizada a filtragem das tabelas de qualidade.

Além da identificação de tabelas de qualidade, o trabalho também propõe a identificação da semântica das tabelas. Primeiramente, é identificada a *coluna assunto* de cada tabela. Considerando um banco de dados relacional, essa coluna *assunto* seria como uma chave primária da tabela, porém sem a restrição de valores únicos. Na Figura 7, por exemplo, essa coluna seria *Cause*. Além da identificação dessa coluna, também são identificados os conceitos de cada coluna da tabela. Para isso, o trabalho utiliza a base de conhecimento *Google Knowledge Graph* e mapeia os valores contidos nas colunas a conceitos da base. Segundo os autores, a possibilidade de mapear as colunas de uma tabela a um ou mais conceitos já é um grande indicativo de que a tabela é de boa qualidade. A existência destes mapeamentos também é considerada no processo de filtragem de tabelas.

Venetis et al. (2011) descrevem um sistema que visa melhorar a questão da descoberta da semântica de tabelas na Web acrescentando a elas anotações. O objetivo principal desse sistema é contribuir com buscas na Web. A motivação para essa proposta é que os motores de busca da Web tratam tabelas como documentos de texto comuns. Porém, dados de tabelas poderiam ser melhor recuperados se fossem tratados de forma diferente, observando a semântica contida nas tabelas. Por exemplo, muitas vezes tabelas na Web não possuem cabeçalhos explícitos demostrando o assunto tratado na mesma. Com a recuperação da semântica dessas tabelas seria possível considerá-las no resultado de uma busca mesmo elas não contendo dados explicita-

mente relacionados à palavra-chave. O conhecimento da semântica das tabelas também permitiria a aplicação de operações de combinação de tabelas, como junções e uniões.

Assim, para recuperar a semântica das tabelas, o trabalho propõe a criação de duas bases de dados geradas automaticamente a partir de textos da Web: a base *isA*, com pares na forma $\{classe, instância\}$, é obtida utilizando basicamente padrões linguísticos. Essa base serve para relacionar entidades (instâncias) com seus conceitos (classes). A segunda base mantém relações em triplas na forma {*argumento1*, *predicado*, *argumento2*}. Ela é obtida utilizando o *TextRunner* (BANKO; ETZIONI, 2008) como ferramenta de extração de informação a partir de textos. Cada tripla dessa base é uma relação de uma entidade com a outra. Por exemplo, ao relacionar as entidades *Barack Obama* com *Estados Unidos*, seria possível criar a tripla {*Barack Obama*, *presidente*, *Estados Unidos*}.

Segundo o trabalho, uma coluna a é rotulada com uma classe k da base de dados *isA* se uma fração substancial das células na coluna a é rotulada com a classe k na base de dados *isA*. Por exemplo, supondo que as palavras *Jamesy Boy* e *Dumbbells* da Tabela 4 tenham sido classificadas pela base da dados *isA* com a classe *Filmes* e que 2/3 das células da *Coluna 1* (sem levar em conta o cabeçalho) foram rotuladas com a classe *Filmes*, é possível rotular aquela coluna com a classe *Filmes*.

De forma semelhante, a relação entre duas colunas a e b é rotulada com uma relação r se uma fração substancial de pares de valores de a e b ocorre, nas extrações, na forma (a,r,b) na base de dados de relações. Por exemplo, supondo que existam as triplas {*Jamesy Boy*, *distríbuido por*, *Phase 4 Films*} e {*Dumbbells*, *distríbuido por*, *GoDigital*} na base de dados de relações e que 2/3 dos valores da *Coluna 1* estão relacionados com os valores da *Coluna 2* com a relação *distríbuido por*, é possível dizer que as *Colunas 1* e *2* possuem a relação *distríbuido por*.

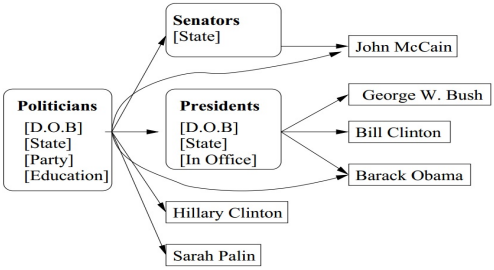
Wang et al. (2012) também propõem uma forma de recuperar a semântica das tabelas na Web. Ele afirma que a chave para entender o significado de tabelas é saber qual conceito melhor descreve as entidades e atributos contidos nas tabelas. Para encontrar esses conceitos, o trabalho utiliza uma base de conhecimento chamada Probase (WU et al., 2012). Esta base contém conceitos, atributos e entidades. A Figura 8 mostra um exemplo desta base. Ela possui conceitos, como 'Politicians', 'Presidents' e 'Senator', atributos como 'State', 'Party', 'D.O.B' e entidades como 'John McCain' e 'Bill Clinton'. As linhas e colunas

Tabela 4: Filmes de Fevereiro de 2014

Coluna 1	Coluna 2	Coluna 3
Jamesy Boy	Phase 4 Films	Crime, drama
Paranormal Activity: The Marked Ones	Paramount Pictures	Horror
Dumbbells	GoDigital	Comedy

Fonte: Wikitables (http://downey-n1.cs.northwestern.edu/public/)

Figura 8: Um fragmento da base de conhecimento Probase.



Fonte: Wang et al. (2012)

das tabelas são comparadas com essa base e a semântica das tabelas é recuperada.

Uma vez realizada a detecção da semântica das tabelas na Web, o trabalho propõe e desenvolve um sistema de busca semântica em tabelas. Esse sistema busca encontrar, em meio a tabelas na Web, um resultado que melhor se relacione semanticamente com as palavras-chave utilizadas pelo usuário. Por exemplo, a Figura 9 seria um resultado obtido a partir do termo de busca *Aniversário do Barack Obama*. Esse resultado viria a partir de tabelas na Web previamente anotadas com semântica.

Braunschweig et al. (2015) apresentam uma abordagem para recuperação de contexto de tabelas na Web. O objetivo do trabalho é enriquecer as tabelas com informações para melhorar o processo de indexação e busca por tabelas na Web. Diferentemente das abordagens anteriores, este trabalho busca enriquecer as tabelas HTML com informações do contexto onde elas estão inseridas, ou seja, da página HTML.

Figura 9: Aniversário do Barack Obama

President	Date of Birth
Barack Obama	Aug 4, 1961

Fonte: Wang et al. (2012)

O trabalho considera as seguintes fontes de contexto para enriquecemento das tabelas:

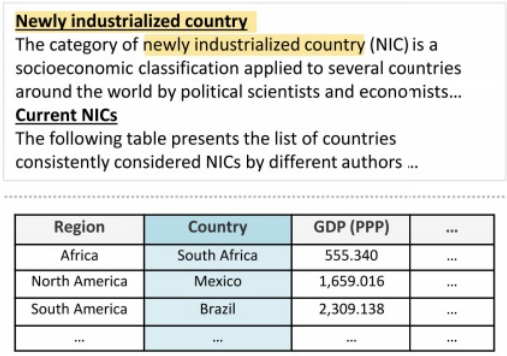
- **Título da tabela**: representado pela tag HTML `<caption>` existente em algumas tabelas na Web;
- **Títulos da página**: representados pelas tags `<h1>-<h6>`. O trabalho utiliza as mais próximas à tabela;
- **Texto ao redor**: representados pelos parágrafos (`<p>`) e listas (`<li>`) existentes ao redor da tabela;
- **Texto completo**: considera-se todo o texto da página Web.

Esses elementos de contexto são processados, sendo filtradas as ditas frases nominais, que são frases sem verbos e que irão enriquecer as tabelas com informações adicionais. Assim sendo, o trabalho busca encontrar frases nominais relacionadas direta ou indiretamente às tabelas, o que é chamado de *contexto diretamente relacionado* e *contexto indiretamente relacionado*.

O contexto diretamente relacionado são frases nominais que possuem diretamente palavras, frases, siglas ou abreviações existentes nas tabelas HTML. Elas são encontradas simplesmente através de comparações de strings ou, no caso das siglas, utiliza-se expressões regulares para encontrar um conjunto de palavras que corresponda à sigla. A Figura 10 por exemplo, apresenta contextos desse tipo, onde as frases em destaque no texto possuem diretamente palavras relacionadas com a coluna em destaque (*Country*). A Figura 11, por sua vez, possui em destaque no texto uma sigla com seu significado, o que pode ser utilizado para acrescentar semântica à coluna destacada na tabela (*HDI*).

O contexto indiretamente relacionado é obtido a partir de uma base de conhecimento. De forma semelhante aos trabalhos de recuperação de semântica já apresentados, o trabalho recupera os conceitos existentes nos dados das tabelas na Web e utiliza esses conceitos para

Figura 10: Contexto Diretamente Relacionado



Newly industrialized country

The category of newly industrialized country (NIC) is a socioeconomic classification applied to several countries around the world by political scientists and economists...

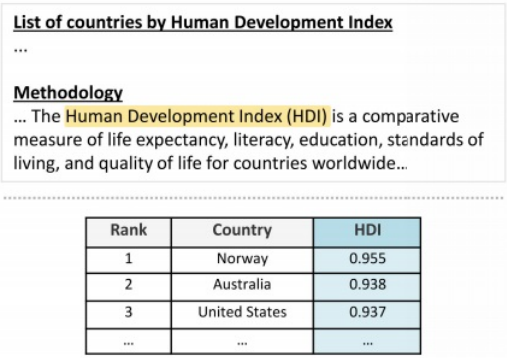
Current NICs

The following table presents the list of countries consistently considered NICs by different authors ...

Region	Country	GDP (PPP)	...
Africa	South Africa	555.340	...
North America	Mexico	1,659.016	...
South America	Brazil	2,309.138	...
...	...	...	...

Fonte: Braunschweig et al. (2015)

Figura 11: Contexto Diretamente Relacionado com Sigla



List of countries by Human Development Index

...

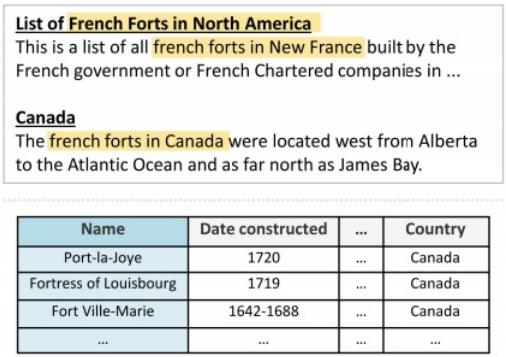
Methodology

... The Human Development Index (HDI) is a comparative measure of life expectancy, literacy, education, standards of living, and quality of life for countries worldwide...

Rank	Country	HDI
1	Norway	0.955
2	Australia	0.938
3	United States	0.937
...	...	...

Fonte: Braunschweig et al. (2015)

Figura 12: Contexto Indiretamente Relacionado



Fonte: Braunschweig et al. (2015)

encontrar o contexto indiretamente relacionado. Utilizando o exemplo da Figura 12, a proposta seria encontrar conceitos para a coluna destacada (*Name*), e a partir desses conceitos descobrir que o texto em destaque possui relação com a coluna. No caso da Figura 12, descobriu-se que a coluna *Name* possui nomes de fortes franceses na América do Norte.

O trabalho também considera a problemática dos termos sinônimos na comparação de strings. Para tanto, ele utiliza a base de dados léxica *WordNet* (MILLER, 1995) como forma de identificar termos sinônimos em meio ao contexto.

3.3 COMPARAÇÃO DE TABELAS

As propostas anteriores foram apresentadas por estarem associadas à manipulação de tabelas na Web. Esta subseção apresenta trabalhos cujo foco está na comparação de tabelas na Web, estando mais fortemente relacionados à proposta deste trabalho.

Fan et al. (2014) apresenta um sistema semi-supervisionado para encontrar conceitos em tabelas na Web. O objetivo do trabalho é descobrir esses conceitos e utilizá-los em comparações de tabelas na Web. Esse processo seria o mesmo que recuperar o esquema das tabelas e realizar um processo de *schema matching*.

Para encontrar os conceitos de um grupo de tabelas, utiliza-se

a base de conhecimento *Freebase* (BOLLACKER et al., 2008), buscando vincular conceitos presentes na base a cada coluna dessas tabelas. O sistema cria um grafo bipartido relacionando as colunas da tabela aos conceitos da *Freebase*. As arestas dos grafos possuem pesos obtidos a partir de comparações entre os valores das colunas e instâncias dos conceitos. O sistema verifica o grau de dificuldade em se descobrir os conceitos de cada coluna a partir desses pesos. Colunas com pesos muito semelhantes recebem um grau de dificuldade maior. Em contra partida, colunas com pesos que se destacam do restante, por possuírem valores maiores, facilitam a determinação do conceito. Para os casos considerados difíceis, o usuário deve proceder a vinculação manualmente.

O trabalho sugere ainda que identificar os conceitos de algumas colunas pode ajudar a descobrir os conceitos de outras colunas. Para tanto, ele calcula o grau de influência de uma coluna sobre as outras. Ele explica que existem dois tipos de influências: *Intra-Table* e *Inter-Table*. *Intra-Table* é uma influência interna à tabela onde, se uma coluna A_i possui o conceito Z_x, outra coluna da mesma tabela A_j, tem grandes chances de possuir o conceito z_y caso Z_x e Z_y sejam relacionados. *Inter-Table*, por sua vez, é uma influência entre tabelas onde, se duas colunas de tabelas diferentes A_i e A_m estão relacionadas, A_m tem grandes chances de possuir o mesmo conceito de A_i. A relação entre as colunas, nesse caso, é obtida utilizando a métrica de similaridade *Cosine*.

Uma vez descobertos os conceitos de cada tabela, é possível então determinar diretamente quais tabelas são similares. Para isso, o trabalho simplesmente compara os conceitos das colunas das tabelas. Uma correspondência é gerada quando duas colunas possuem o mesmo conceito. O trabalho não dá detalhes quanto à quantidade necessária de correspondências para que duas tabelas sejam consideradas similares.

Para avaliar seus resultados, o trabalho utiliza ainda outros dois métodos mais simples de comparação de tabelas: o *HeaderSim* e o *InstanceSim*. Segundo Fan et al. (2014), o *HeaderSim* compara as tabelas utilizando a métrica de similaridade de string *Jaccard* para comparar somente os cabeçalhos das tabelas, não comparando seus dados. Em contrapartida, o *InstanceSim* compara somente os dados das tabelas.

O trabalho demonstra, em seus experimentos, que obteve resultados um pouco melhores que os baselines *HeaderSim* e *InstanceSim*. Entretanto, vale lembrar que o método proposto é semi-supervisionado, diferente dos baselines.

Pawlik e Augsten (2011) propõem o RTED, um algoritmo para calcular a distância de edição entre árvores. Essa distância é calculada pela sequência de operações de edição, com menor custo, que transforma uma árvore em outra. As seguintes operações são consideradas:

- excluir um nodo;
- inserir um nodo;
- renomear o rótulo de um nodo.

Cada operação recebe um custo e a distância de edição é a soma desses custos para transformar uma árvore em outra. Algoritmos desse tipo utilizam recursividade para dividir as árvores em subárvores. Assim, são calculadas as edições necessárias de subproblemas menores.

Entretanto, a complexidade desse algoritmo é bastante alta, chegando a ser exponencial em alguns casos. Alguns algoritmos otimizam essa recursividade e melhoram a complexidade. Contudo, eles possuem alta dependência do formato das árvores. Com isso, o trabalho propõe o RTED, um algoritmo de complexidade $O(n^3)$, que otimiza a recursividade ao dividir as árvores de forma independente do formato.

Esse algoritmo compara árvores rotuladas em geral e pode ser adaptado para comparar árvores HTML.

Sarma et al. (2012) propõem um *framework* de comparação de tabelas cujo principal objetivo é encontrar tabelas candidatas para operações de junção e de união. Para isso, ele parte dos seguintes princípios:

- Duas tabelas t_1 e t_2 são relacionadas se podem identificar uma tabela virtual t tal que t_1 e t_2 são resultados de duas consultas q_1 e q_2, respectivamente, sobre t;
- A tabela t deve ser coerente, ou seja, deve manter conteúdos relacionados. Neste sentido, não é coerente, por exemplo, imaginar uma tabela que contenha preços de chá na China e os vencedores da Maratona de Boston;
- As consultas q_1 e q_2 devem possuir estrutura similar em termos de seleções e projeções.

O trabalho considera dois tipos mais comuns de tabelas na Web que possuem relacionamento: *Entity Complement* e *Schema Complement*. Tabelas *Entity Complement* são obtidas a partir de diferentes seleções sobre um mesmo conjunto de atributos de uma tabela. É o

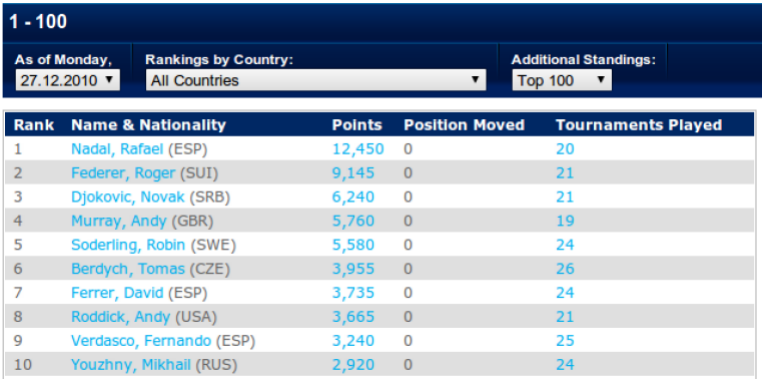
caso das Figuras 13 e 14, onde são selecionadas as colocações de 1 a 100 e de 101 a 200, respectivamente, a partir de uma tabela hipotética com as colocações de 1 a 200. Para identificar duas tabelas desse tipo, o trabalho segue os seguintes critérios:

- **Expansão de entidades**: Uma tabela deve adicionar novas entidades à outra;

- **Consistência de esquema**: As tabelas devem possuir esquemas similares;

- **Consistência entre as entidades**: para garantir a coerência da tabela virtual t.

Figura 13: Top 100 Tênis Masculino 2010 do ATP World Tour



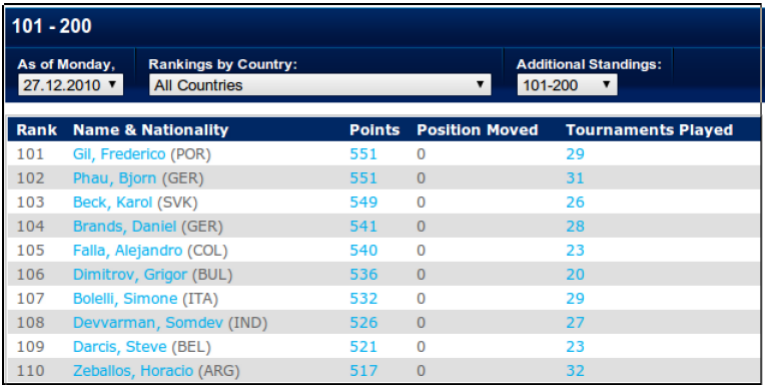
1 - 100

As of Monday, 27.12.2010 | Rankings by Country: All Countries | Additional Standings: Top 100

Rank	Name & Nationality	Points	Position Moved	Tournaments Played
1	Nadal, Rafael (ESP)	12,450	0	20
2	Federer, Roger (SUI)	9,145	0	21
3	Djokovic, Novak (SRB)	6,240	0	21
4	Murray, Andy (GBR)	5,760	0	19
5	Soderling, Robin (SWE)	5,580	0	24
6	Berdych, Tomas (CZE)	3,955	0	26
7	Ferrer, David (ESP)	3,735	0	24
8	Roddick, Andy (USA)	3,665	0	21
9	Verdasco, Fernando (ESP)	3,240	0	25
10	Youzhny, Mikhail (RUS)	2,920	0	24

Fonte: Sarma et al. (2012)

Figura 14: Colocações 100 - 200 Tênis Masculino 2010 do ATP World Tour



101 - 200

As of Monday, 27.12.2010 ▾ | Rankings by Country: All Countries ▾ | Additional Standings: 101-200 ▾

Rank	Name & Nationality	Points	Position Moved	Tournaments Played
101	Gil, Frederico (POR)	551	0	29
102	Phau, Bjorn (GER)	551	0	31
103	Beck, Karol (SVK)	549	0	26
104	Brands, Daniel (GER)	541	0	28
105	Falla, Alejandro (COL)	540	0	23
106	Dimitrov, Grigor (BUL)	536	0	20
107	Bolelli, Simone (ITA)	532	0	29
108	Devvarman, Somdev (IND)	526	0	27
109	Darcis, Steve (BEL)	521	0	23
110	Zeballos, Horacio (ARG)	517	0	32

Fonte: Sarma et al. (2012)

Figura 15: Top 100 Tênis Masculino 2010 do ESPN

2010 Men's Tennis ATP Rankings

Year: 2010 ▾
Type: ATP Rankings ▾

Men's Singles Rankings

RK	PLAYER	COUNTRY	MOVEMENT	POINTS
1	Rafael Nadal		0	12450
2	Roger Federer		0	9145
3	Novak Djokovic		0	6035
4	Andy Murray		1	5760
5	Robin Soderling		1	5580
6	Tomas Berdych		0	3955
7	David Ferrer		0	3735
8	Andy Roddick		0	3665
9	Fernando Verdasco		0	3240
10	Mikhail Youzhny		0	2920

Fonte: Sarma et al. (2012)

Para garantir esses critérios, o trabalho, primeiramente, utiliza o algoritmo de Venetis et al. (2011)(ver Seção 3.2) para encontrar as enti-

dades das tabelas. Após isso, são utilizadas três bases de conhecimento para rotular essas entidades: *Freebase*, *WebIsA* (VENETIS et al., 2011) e uma base própria. O grau de relacionamento entre as entidades distintas de duas tabelas é calculado pela quantidade de rótulos em comum. Com isso, verifica-se a possibilidade de expansão. A consistência de esquema é verificada a partir de cálculos de similaridade entre os atributos (utilizando o pacote de similaridade Java SecondString (COHEN; RAVIKUMAR; FIENBERG, 2003)), tipos de dados e valores (usando uma variante da métrica *Jaccard* (JACCARD, 1908)). Por fim, a consistência entre as tabelas é verificada a partir das entidades em comum.

O segundo tipo de tabelas considerado pelo trabalho, *Schema Complement*, diz respeito a tabelas que possuem o mesmo conjunto de entidades, porém, com atributos distintos, ainda que semanticamente relacionados. É o caso das Figuras 13 e 15 onde são selecionados os mesmos conjuntos de dados, contudo seus atributos não são totalmente idênticos. Para identificar tabelas *Schema Complement*, são considerados os seguintes fatores:

- **Cobertura do conjunto de entidades**: Uma tabela deve possuir a maioria das entidades da outra, senão todas;

- **Atributos adicionais são bem-vindos**: Uma tabela deve possuir atributos adicionais os quais não são descritos na outra.

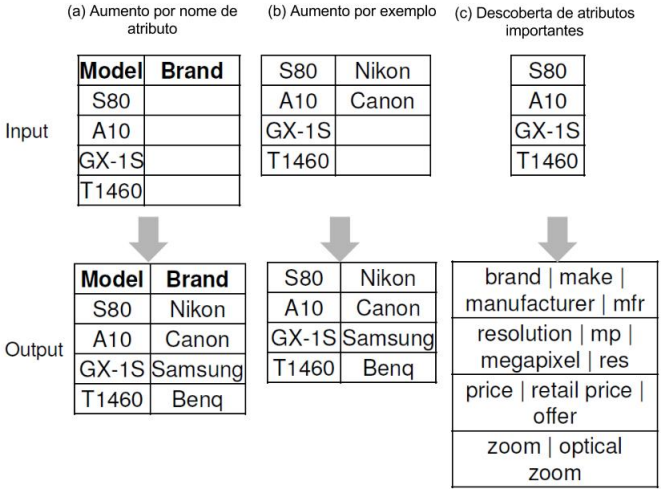
Nesse caso, a cobertura é garantida verificando-se a quantidade de entidades em comum entre as tabelas. Para verificar os atributos adicionais, utiliza-se os mesmos cálculos de similaridade vistos para as tabelas do tipo *Entity Complement* e empregados na verificação de consistência de esquema. Feito esse *schema matching*, são observados os atributos que não são comuns às duas tabelas.

Por fim, o trabalho gera escores para cada um desses tipos de tabelas e avalia os métodos com base em observações de usuários para verificar quais tabelas são mais aptas para operações de junção e união.

Yakout et al. (2012) focam na coleta de informações de entidades. O trabalho explica que, em um processo de coleta de informações, existem basicamente duas subtarefas: encontrar valores de atributos das entidades e encontrar os atributos relevantes dessas entidades. O trabalho propõe automatizar essas tarefas utilizando tabelas extraídas da Web. Esse processo é feito por três operações chave:

- **Aumento por nome de atributo**: Adicionar valores a entidades a partir de atributos conhecidos;

Figura 16: Operações chave para coleta de informações



Fonte: (YAKOUT et al., 2012)

- **Aumento por exemplo**: Adicionar valores a entidades a partir de exemplos conhecidos;

- **Descoberta de atributos importantes**: Descobrir atributos importantes a partir de valores de entidades conhecidos.

Essas três operações são ilustradas na Figura 16. A Figura 16(a), por exemplo, mostra um **aumento por nome de atributo**, onde a partir dos atributos conhecidos *Model* e *Brand*, acrescentam-se novos dados às entidades, no caso, para o atributo *Brand*. A Figura 16(b) mostra um **aumento por exemplo** onde, não existindo nomes de atributos (cabeçalhos), somente exemplos de dados, foram descobertos novos dados para as entidades da tabela. Finalmente, a Figura 16(c) demonstra a **descoberta de atributos importantes**, sendo que, a partir de valores de entidades contidas na tabela, foram descobertos atributos relevantes para aquele domínio do conhecimento.

O trabalho explica que para realizar essas operações é necessário realizar o *schema matching* das tabelas na Web. Porém, dado um grande número de tabelas, esse processo pode ser muito demorado. Por isso, é proposto um framework que pré-processa as tabelas na Web

realizando o *schema matching*, indexa essas tabelas utilizando um algoritmo baseado no *Page rank* (HAVELIWALA, 2002) e em tempo de busca, realiza as operações de coleta de informações.

Com relação ao processo de *matching* de tabelas na Web, o trabalho trata somente tabelas horizontais. Para tanto, utiliza um algoritmo de aprendizado de máquina treinado com as seguintes características das tabelas:

- **Similaridade do contexto**: utiliza as funções TF-IDF e *cosine* para comparar o texto existente ao redor da tabela na página HTML;
- **Tabela com contexto**: novamente utiliza as funções TF-IDF e *cosine* para comparar uma tabela com o texto ao redor de outra tabela para verificar se são similares;
- **URL**: compara as URLs das páginas HTML utilizando *cosine*;
- **Similaridade das tuplas**: compara os valores contidos nas tabelas, que no contexto deste trabalho são os dados das tabelas;
- **Nomes de atributos**: compara os cabeçalhos das tabelas;
- **Similaridade entre tabelas**: compara as tabelas como um conjunto genérico de palavras;
- **Largura das tabelas**: compara a largura das tabelas.

3.4 COMPARATIVO

A Tabela 5 mostra um comparativo dos trabalhos relacionados já inclusa a proposta desta dissertação. Como é possível observar, seis das 12 abordagens processa somente tabelas horizontais, e dentre os trabalhos que tratam de comparação de tabelas, somente uma abordagem (PAWLIK; AUGSTEN, 2011) processa qualquer tipo de tabela, apesar de essa abordagem comparar somente a estrutura das tabelas HTML. Quanto ao tratamento de sinônimos, as abordagens que utilizam bases de conhecimento acabam, por consequência disso, possuindo algum tratamento desse tipo, visto que as bases de conhecimento podem extrair conceitos de termos com mesmo significado, tornando possível identificar que são sinônimos.

Baseado nas características e limitações desses trabalhos, esta dissertação propõe a criação e avaliação de alguns métodos de comparação de tabelas Web com suportes a tratamento de sinônimos e que

Tabela 5: Comparativo entre os Trabalhos Relacionados

Trabalho	Abordagem	Tipo de Tabelas Processadas	Tratamento de Sinônimos	Estratégia
(LAUTERT; SCHEIDT; DORNELES, 2013)	Estrutura	Todos os tipos considerados pela taxonomia	Não se aplica	Rede neural
Son e Park (2013)	Estrutura	Dados e Layout	Não se aplica	Análise de padrões
Lai (2013)	Estrutura	Horizontais, Verticais, Matriciais	Não se aplica	Funções de similaridade
Balakrishnan et al. (2015)	Identificação de semântica	Horizontais, Verticais	Sim	Base de conhecimento e aprendizado de máquina
Wang et al. (2012)	Identificação de semântica	Horizontais	Sim	Base de conhecimento (Probase)
Venetis et al. (2011)	Identificação de semântica	Horizontais	Sim	Base de conhecimento (própria)
Braunschweig et al. (2015)	Identificação de semântica	Horizontais	Sim	Base de conhecimento
Fan et al. (2014)	Comparação de tabelas	Horizontais	Sim	Base de conhecimento e funções de similaridade
Sarma et al. (2012)	Comparação de tabelas	Horizontais	Sim	Base de conhecimento e funções de similaridade
Pawlik e Augsten (2011)	Comparação de tabelas	Qualquer tipo	Não	TED
Yakout et al. (2012)	Comparação de tabelas	Horizontais	Não	Aprendizado de máquina e funções de similaridade
Proposta	Comparação de tabelas	Qualquer tipo	Sim	Base de conhecimento e funções de similaridade

considerem as estruturas heterogêneas das tabelas Web, além de comparar os métodos com esses suportes com métodos mais simples de comparação. Nenhum trabalho relacionado trata conjuntamente das duas problemáticas de tratamento de sinônimos e estruturas heterogêneas das tabelas.

Os métodos propostos nesta dissertação consideram a taxonomia recente de tabelas na Web, apresentada em Lautert, Scheidt e Dorneles (2013), para fins de identificação de estruturas heterogêneas de tabelas e uso na comparação. Nenhum trabalho relacionado considera essa taxonomia, que é a mais abrangente a respeito na literatura atual.

4 PROPOSTA

A proposta dessa dissertação é definir e avaliar uma série de métodos para determinação de similaridade entre tabelas na Web. Esses métodos foram criados sobre uma arquitetura base, detalhada a seguir, que pode ser adaptada modificando a forma de comparação, métricas e bases de conhecimento, visando uma melhor adequação ao domínio de dados considerado. O objetivo desses métodos é comparar tabelas HTML levando em conta os diferentes formatos de construção dessas tabelas e o uso de palavras sinônimas. Essas são as principais contribuições desta dissertação.

4.1 PREMISSAS

Este trabalho considera, como tabelas similares, duas ou mais tabelas que possuam um mesmo domínio do conhecimento e que compartilhem algumas propriedades, conforme a Definição 4.1.1. Portanto, não é necessário que elas possuam exatamente os mesmos dados. Um exemplo disso pode ser visto nas Tabelas 6 e 7. Apesar de não possuírem exatamente os mesmos filmes nem o mesmo esquema, ambas tratam de filmes e compartilham os atributos *Genre* e *Title* (*Name*).

Definição 4.1.1 *(Tabelas Similares). Duas tabelas na Web* t_A *e* t_B *que contenham informações de um mesmo domínio do conhecimento e que* $t_A.H \cap t_B.H \neq \emptyset$*, podem ser consideradas similares mesmo que* $t_A.D \cap t_B.D = \emptyset$*, onde* H *e* D *são os conjuntos de cabeçalhos e dados de uma tabela na Web respectivamente, conforme Definição 4.3.2. Sendo que* $t_A.H \cap t_B.H$ *inclui termos sinônimos.*

Tabela 6: Filmes de Janeiro de 2014

Title	**Studio**	**Genre**	**Directors**
The Lego Movie	Warner Bros.	Action, comedy	Phil Lord, Christopher Miller
Barefoot	Roadside Attractions	Romantic, comedy, drama	Andrew Fleming
The Monuments Men	Columbia Pictures / 20th Century Fox	Drama, war	George Clooney

Fonte: Wikitables (http://downey-n1.cs.northwestern.edu/public/)

Tabela 7: Filmes de Fevereiro de 2014

Name	Distributor	Genre
Jamesy Boy	Phase 4 Films	Crime, drama
Paranormal Activity: The Marked Ones	Paramount Pictures	Horror
Dumbbells	GoDigital	Comedy

Fonte: Wikitables (http://downey-n1.cs.northwestern.edu/public/)

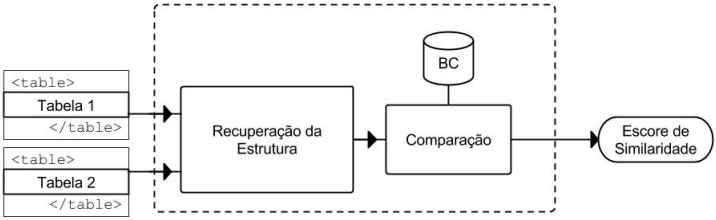
Uma forma de descobrir se duas tabelas pertencem a um mesmo domínio é comparando os seus esquemas. Sarma et al. (2012), Wang et al. (2012) e Lai (2013) utilizam os cabeçalhos de tabelas na Web em seus processos de *schema matching*. Portanto, um primeiro passo para se comparar tabelas é comparar os seus cabeçalhos.

Juntamente com os cabeçalhos, tabelas também possuem dados, conforme as Definições 2.2.1, 2.2.2 e 2.2.4 do Capítulo 2. Esses dados podem ser um componente importante no processo de comparação de similaridade entre tabelas na Web, sendo considerados por alguns trabalhos relacionados (WANG et al., 2012; YAKOUT et al., 2012). Assim sendo, dados também foram incluídos nos métodos de comparação de tabelas na Web propostos.

4.2 ARQUITETURA

A Figura 17 apresenta a arquitetura base utilizada para a criação e avaliação dos métodos propostos. Todos os métodos de comparação propostos foram criados considerando esse formato. Conforme comentado anteriormente, essa arquitetura permite a configuração de alguns parâmetros, alguns deles são: a forma de comparação de tabelas, a função de similaridade de string utilizada na comparação, a base de conhecimento, quais características serão comparadas (cabeçalhos, dados, estrutura, contexto e etc), a forma de detecção dessas características (ver Seção 3.1 sobre formas de detecção de estrutura), os pesos utilizados na comparação de cada característica e o *threshold* aplicado sobre o escore de similaridade. As subseções a seguir detalham melhor cada componente dessa arquitetura.

Figura 17: Arquitetura do Base para os Métodos



4.2.1 Entrada

A arquitetura possui como entrada duas tabelas na Web. Essas tabelas são recebidas no formato HTML, supondo a existência de método prévio de identificação e extração de tabelas presentes em páginas Web. Essa extração geralmente é feita a partir de um tipo de programa chamado *Web Crawler* (SREEJA; CHAUDHARI, 2014), o qual percorre os sites a partir de seus links e recupera informações de interesse, no caso, tabelas HTML. Porém, essa extração não é o foco dessa dissertação e por isso considera-se que as tabelas já foram previamente extraídas.

4.2.2 Recuperação da Estrutura

A seguir, é feito um processamento para recuperar a estrutura dessas tabelas, distinguindo cabeçalhos, dados e tipos de tabelas. Como comentado anteriormente, essa proposta difere de outras abordagens de comparação de tabelas pois trata diferentes disposições de tabelas, como horizontais, verticais e matriciais (LAUTERT; SCHEIDT; DORNELES, 2013). Esse tratamento é feito através de uma padronização de estruturas diferentes de tabelas durante a recuperação da estrutura. Portanto, é nesse momento que se utiliza a taxonomia para diferenciar os principais tipos de tabelas. Mais detalhes sobre essa padronização são descritos na Seção 4.3.2.

4.2.3 Comparação

Após a detecção da estrutura, ocorre a comparação das tabelas. Nesse momento o método utiliza-se de funções de similaridade de string e outras técnicas detalhadas a seguir, para comparar as principais características das tabelas. É durante a comparação que também é feita a detecção de termos sinônimos. Isso é feito através do suporte de uma base de conhecimento, garantindo que o sistema encontre tabelas similares baseado no significado das palavras e não apenas na sua grafia. Essa base de conhecimento utilizada pode ser uma base que, a partir de dois termos quaisquer, retorna um escore com o qual seria possível identificar se os termos são sinônimos ou não, ou uma base que retorne conceitos a partir de termos. Nesse segundo caso, seria necessário identificar conceitos em comum entre os termos para descobrir se são sinônimos ou não.

4.2.4 Saída

Após a comparação entre as tabelas, o método devolve como saída um escore de similaridade, conforme descrito na Seção 2.1 do Capítulo 2. Nesse momento aplica-se um *threshold* ao escore de similaridade, o qual é um valor de corte que determina se duas tabelas são similares ou não. Caso um escore de similaridade seja maior que o *threshold* considerado, entende-se que as tabelas são similares. Caso contrário, elas são distintas.

4.3 DETERMINAÇÃO DA SIMILARIDADE

O cálculo geral para determinação de similaridade entre tabelas na Web está descrito na Equação 4.1, onde a similaridade entre duas tabelas t_1 e t_2 é determinada a partir das similaridades entre seus cabeçalhos (*headers*) (H_1, H_2) e seus dados (D_1, D_2). Pesos são aplicados a cada componente da equação: o peso dos cabeçalhos w_H e o peso dos dados w_D. A soma desses pesos deve ser igual a 1 para manter a similaridade normalizada ($w_H + w_D = 1$).

$$Sim(t_1, t_2) = (Sim(H_1, H_2) * w_H) + (Sim(D_1, D_2) * w_D) \qquad (4.1)$$

Duas definições são necessárias para uma melhor compreensão desta equação geral 4.1 e são dadas a seguir.

Definição 4.3.1 *(Cabeçalho de uma tabela na Web) O cabeçalho de uma tabela t_i na Web é dado por $H_i = \{h_n\}$, onde cada $h_n \in H_i$ é o nome de uma propriedade de t_i, conforme a Definição 2.2.3.*

Definição 4.3.2 *(Dados de uma tabela na Web) O conjunto de dados de uma tabela t_i é dado por $D_i = \{d_n\}$, onde cada $d_n \in D_i$ é um dado de uma célula, conforme a Definição 2.2.4.*

Ainda, a base da equação geral é uma métrica denominada *SubSetSim*. Esta métrica é detalhada na sequência.

4.3.1 Métrica *SubSetSim*

Como visto na Equação 4.1 e nas Definições 4.3.1 e 4.3.2, a determinação similaridade entre tabelas na Web envolve o cálculo de similaridades entre conjuntos, sejam de dados ou cabeçalhos. Para realizar esta comparação de conjuntos de valores foi utilizada a métrica de similaridade *SubSetSim* (DORNELES et al., 2004). Ela é representada pela Equação 4.2, sendo ϵ_p e ϵ_d conjuntos de valores, e n e m tamanhos dos conjuntos ϵ_p e ϵ_d, respectivamente. A função calcula as similaridades de todos os valores com os demais e soma as máximas similaridades. A função *sim* pode ser alterada de acordo com a necessidade. Assim sendo, várias funções de similaridade de *string*, contidas na biblioteca *SimMetrics* (CHAPMAN, 2006), foram avaliadas para determinar as mais eficazes na comparação de tabelas na Web. Observando a forma de comparação de cada uma, bem como seus pontos fortes e fracos, foram selecionadas sete funções:

- **Cosine**: Verifica palavras iguais em uma frase;
- **Monge Elkan**: Executa distância de edição de string;
- **Chapman Length**: Compara o tamanho das Strings;
- **Euclidean Distance**: Verifica palavras iguais em uma frase;
- **Jaro Winkler**: Verifica caracteres iguais;
- **Jaccard**: Verifica palavras iguais em uma frase;
- **Soundex**: Compara fonética das palavras.

$$SubSetSim(\epsilon_p, \epsilon_d) = \frac{\sum \epsilon_p^i \cdot \eta = \epsilon_d^j \cdot \eta^{(max(sim(\epsilon_p^i, [\epsilon_d^1, \ldots, \epsilon_d^m])))}}{min(m, n)} \quad (4.2)$$

Alguns experimentos preliminares foram realizados para determinar quais funções melhor se adequariam à situação de comparação de tabelas na Web. Mais detalhes sobre esses experimentos são apresentados na Seção 5.

Conforme a Definição 4.1.1, a quantidade de dados similares entre as tabelas para considerá-las similares não é tão importante quanto o domínio. Em decorrência disso foi feita uma adaptação na métrica *SubSetSim* para considerar, como denominador, o menor conjunto (min (m, n)). Isso significa que, ao se comparar uma tabela com muitos dados com outra que possui poucos dados, a similaridade será normalizada pela menor quantidade de dados, possibilitando a detecção de similaridade entre duas tabelas mesmo que seus tamanhos variem bastante.

4.3.2 Casos de Comparação

Como visto na Seção 2.2, existem algumas variações nos cabeça lhos e nos dados a serem consideradas na comparação de tabelas na Web. Esses casos de comparação foram definidos baseando-se na taxonomia proposta por Lautert, Scheidt e Dorneles (2013) - Seção 2.2.1, de onde foram retiradas as principais características que diferem os tipos de tabelas. Com isso, é necessário que a comparação entre tabelas considere todas as combinações possíveis dessas variações.

Com base na taxonomia vista na Seção 2.2.1, percebeu-se que existem três tipos principais de cabeçalhos: horizontais (H_h), verticais (H_v) e matriciais (H_m). Assim sendo, existem seis casos de comparação: $H_h \times H_h$, $H_h \times H_v$, $H_h \times H_m$, $H_v \times H_v$, $H_v \times H_m$ e $H_m \times H_m$. Por serem muitas combinações, o ideal é definir uma única equação que abranja todos os casos de comparação.

Para se chegar a esta equação, buscou-se padronizar todos os tipos de cabeçalhos para que fiquem na mesma disposição. No caso, padronizou-se todos os cabeçalhos para a disposição horizontal (H_h). Uma vez padronizados os cabeçalhos, a comparação é feita célula por célula utilizando uma função de similaridade de *string* e o suporte opcional de uma base de conhecimento, caso se considere termos sinônimos na comparação.

Considerando inicialmente um cabeçalho vertical, ele pode ser visto como um cabeçalho horizontal transposto, conforme a Equação 4.3. Portanto, para fins de comparação, é necessário somente transpor esse cabeçalho para que fique na mesma disposição de um cabeçalho horizontal. A Equação 4.4 apresenta a comparação $H_h \times H_v$ em termos de H_h.

$$H_v \equiv H_h^T \tag{4.3}$$

$$H_{1h} \times H_{2v} \equiv H_{1h} \times H_{2h}^T \tag{4.4}$$

Com base ainda na Equação 4.3, pode-se modificar também a comparação do caso $H_v \times H_v$ para deixá-la em termos de H_h, conforme visto na Equação 4.5.

$$H_{1v} \times H_{2v} \equiv H_{1h}^T \times H_{2h}^T \tag{4.5}$$

Conforme visto na taxonomia descrita na Seção 2.2.1, cabeçalhos matriciais estão dispostos tanto na vertical quanto na horizontal. Portanto, é possível dizer que eles são uma união dos cabeçalhos verticais e horizontais (Equação 4.6). Com base nessa definição, e aplicando a propriedade distributiva da união de conjuntos, é possível desenvolver a comparação $H_h \times H_m$ e chegar à Equação 4.7 em termos de H_h.

$$H_m \equiv H_h \cup H_v \tag{4.6}$$

$$\begin{aligned} H_{1h} \times H_{2m} &\equiv H_{1h} \times (H_{2h} \cup H_{2v}) \\ &= (H_{1h} \times H_{2h}) \cup (H_{1h} \times H_{2v}) \\ &= (H_{1h} \times H_{2h}) \cup (H_{1h} \times H_{2h}^T) \end{aligned} \tag{4.7}$$

Da mesma forma, é possível desenvolver a comparação $H_v \times H_m$ e chegar à Equação 4.8, que é a união das Equações 4.4 e 4.5.

$$\begin{aligned} H_{1v} \times H_{2m} &\equiv H_{1v} \times (H_{2h} \cup H_{2v}) \\ &= (H_{1v} \times H_{2h}) \cup (H_{1v} \times H_{2v}) \\ &= (H_{1h}^T \times H_{2h}) \cup (H_{1h}^T \times H_{2h}^T) \end{aligned} \tag{4.8}$$

Finalmente, com relação à comparação $H_m \times H_m$, é possível aplicar os mesmos princípios utilizados nas outras comparações. Com base

na Equação 4.6 para reduzir H_m a sua forma mais simples, aplicando a propriedade distributiva da união de conjuntos e utilizando H_h^T no lugar de H_v, chega-se à Equação 4.9.

$$\begin{aligned} & H_{1m} \times H_{2m} \\ \equiv\ & (H_{1h} \times H_{2h}) \cup (H_{1h} \times H_{2v}) \cup (H_{1v} \times H_{2h}) \cup (H_{1v} \times H_{2v}) \\ \equiv\ & (H_{1h} \times H_{2h}) \cup (H_{1h} \times H_{2h}^T) \cup (H_{1h}^T \times H_{2h}) \cup (H_{1h}^T \times H_{2h}^T) \end{aligned} \tag{4.9}$$

Como é possível observar, a Equação 4.9 abrange todos os casos de comparação já vistos. Desta forma, qualquer combinação de cabeçalho poderia ser comparada utilizando esta equação, o que leva à equação geral de comparação entre cabeçalhos de duas tabelas na Web. Nessa equação é calculada a similaridade entre cada combinação de tipo de cabeçalho. Para que o escore de similaridade final seja normalizado aplica-se um peso a cada componente da equação, sendo a soma desses pesos sempre igual a um.

Vale ressaltar que, nesta equação, não necessariamente todos os casos poderão estar presentes para duas tabelas sendo comparadas. Por exemplo, na comparação entre duas tabelas horizontais, somente a similaridade entre dois cabeçalhos horizontais ($Sim(H_{1_h}, H_{2_h})$) é aplicada. Os casos que não se aplicam não são considerados no cálculo e nesses casos os pesos são ajustados.

Portanto, dados dois cabeçalhos de tabelas H_1 e H_2, a equação geral de comparação entre eles, levando em conta o mapeamento entre seus diferentes tipos, é dada pela Equação 4.10, onde w_i é o peso aplicado, sendo $\left(\sum_{i=1}^{n} w_i\right) = 1$, onde n é a quantidade de componentes da equação.

$$\begin{aligned} Sim(H_1, H_2) \equiv\ & Sim(H_{1_h}, H_{2_h}) * w_1 + \\ & Sim(H_{1_h}, H_{2_h}^T) * w_2 + \\ & Sim(H_{1_h}^T, H_{2_h}) * w_3 + \\ & Sim(H_{1_h}^T, H_{2_h}^T) * w_4 \end{aligned} \tag{4.10}$$

Definida a equação geral de comparação entre cabeçalhos, resta definir a equação geral de comparação entre dados. Como foi visto na Definição 2.2.5, existem três tipos de dados (d): Monovalorados ($mono$), Multivalorados ($multi$) e Tabelas (t), ou seja, $d = mono|multi|t$. Para simplificar o processo de padronização, assume-se que dados Mo-

novalorados e dados Multivalorados são um mesmo tipo de dado d', o qual é um dado não-estruturado e que não inclui tabelas internas. Tabelas internas, nesse caso, são dados estruturados. Logo, $d = d'|t$. Seguindo a Definição 4.3.2, é possível expandir essa ideia para conjuntos de dados, resultando em $D = D'|T$. Com isso, define-se a equação de comparação de dados entre duas tabelas na Web t_1 e t_2, descrita pela Equação 4.11, onde D_1 e D_2 são os dados das tabelas t_1 e t_2 respectivamente, D'_1 e D'_2 são dados não estruturados das tabelas t_1 e t_2, e T'_1 e T'_2 são dados estruturados das tabelas t_1 e t_2, respectivamente. De forma análoga à equação geral de comparação de cabeçalhos, esta equação considera todos os casos possíveis de comparação de conteúdos de dados presentes em duas tabelas na Web.

$$\begin{aligned} Sim(D_1, D_2) = Sim(D'_1, D'_2) + \\ Sim(D'_1, T'_2) + \\ Sim(T'_1, D'_2) + \\ Sim(T'_1, T'_2) \end{aligned} \tag{4.11}$$

Na Equação 4.11, o cálculo de similaridade entre os dados não-estruturados $D'(Sim(D'_1, D'_2))$ pode ser feito utilizando a métrica *SubSetSim* para comparação de conjuntos, aplicando uma função de similaridade de *string*. O cálculo da similaridade entre dados estruturados $(Sim(T'_1, T'_2))$ remete ao cálculo geral de similaridade entre tabelas na Web, gerando uma recursividade no cálculo. Já o cálculo de similaridade entre dados não-estruturados e estruturados $(Sim(D'_1, T'_2)$ e $Sim(T'_1, D'_2))$, por sua vez, requer uma atenção especial.

De acordo com as Definições 2.2.1 e 2.2.2, uma tabela é um conjunto de células, as quais podem conter dados ou cabeçalhos. Logo, assume-se que uma tabela é um conjunto de dados (D) e cabeçalhos (H). Assim, a similaridade entre dados não-estruturados D' e estruturados (tabelas internas) pode ser calculada utilizando a Equação 4.12, onde $H_{2T'}$ e $D_{2T'}$ são os cabeçalhos e dados de T'_2, respectivamente.

$$sim(D'_1, T'_2) = sim(D'_1, H_{2T'}) + sim(D'_1, D_{2T'}) \tag{4.12}$$

Na Equação 4.12, a similaridade entre o cabeçalho da tabela interna e o dado não-estruturado $(sim(D', H_{T'}))$ pode novamente ser calculada utilizando a métrica *SubSetSim*. Porém, o cálculo de similaridade entre os dados D' e os dados da tabela interna $D_{T'}$ remete novamente à Definição 2.2.5 para tipos de dados de uma tabela na

Web, visto que os dados da tabela interna também podem, mais uma vez, conter dados estruturados e não-estruturados. Isso gera a recursividade ilustrada na Equação 4.13.

$$sim(D'_1, T'_2) = sim(D'_1, H_{2T'}) + sim(D'_1, D'_{2T'}) + sim(D'_1, T'_{2T'}) \quad (4.13)$$

Desta forma, calcula-se recursivamente a similaridade para tabelas internas mesmo que haja vários níveis de aninhamento entre elas. Na prática esse processamento tende a não ser custoso visto que não existem muitas tabelas aninhadas na Web e seus níveis de aninhamento são finitos. Logo, existe a garantia de parada.

As Equações 4.1, 4.10 e 4.11 são equações gerais da comparação entre tabelas na Web. Elas foram criadas para comparar diversos tipos de tabelas baseando-se nas diferentes estruturas de tabelas descritas pela taxonomia de Lautert, Scheidt e Dorneles (2013). A Equação 4.10, por exemplo, permite a comparação entre tabelas Horizontais, Verticais e Matriciais. A Equação 4.11, por sua vez, permite a comparação de tabelas aninhadas com as demais.

Considerando os demais tipos de tabelas, as tabelas do tipo concisa e dividida acabam sendo igualadas às demais tabelas durante o processo de identificação de estrutura, onde suas células mescladas e cabeçalhos divididos são agrupados com as células e cabeçalhos normais, o que não afeta a comparação, visto que o SubSetSim compara as células todas com todas.

As tabelas do tipo multivaloradas, tanto simples como compostas, também são agrupadas juntamente com as demais durante a recuperação de estrutura. Nos métodos propostos não se utilizou nenhum tratamento especial para esse tipo de tabela, visto que se optou por dar maior foco aos cabeçalhos das tabelas. Entretanto, tabelas multivaloradas poderiam receber um tratamento especial durante a comparação de dados. Uma forma de fazer isso seria utilizando algum tipo de processamento de linguagem natural, como a remoção de *stop words* (preposições, artigos e etc.), detecção de sujeitos, verbos e até mesmo indexação de termos mais importantes utilizando, por exemplo, o TF-IDF (RAMOS, 2003).

Alguns métodos propostos possibilitam ainda a consideração de termos sinônimos na comparação de tabelas na Web, conforme detalhado a seguir.

4.3.3 Tratamento de Sinônimos

As Equações vistas na seção anterior, ainda não levam em conta o tratamento de termos sinônimos presentes nas tabelas. Esse tratamento pode ser amparado por uma base de conhecimento. Nesta primeira versão da abordagem proposta foi utilizado o Wordnet (MILLER, 1995), uma base de dados léxica que contém palavras da língua inglesa, seus sinônimos e outros relacionamentos semânticos. Em particular, foi utilizada a API Java para o Wordnet criada por Shima (2014) e um método desta API que calcula um valor de similaridade semântica entre termos. Assim sendo, este método da API para o Wordnet foi empregado como uma função de similaridade utilizada em conjunto com o *SubsetSim*. A Equação 4.14 ilustra, de forma simplificada, o uso desta função, denominada *wordnet*, com o *SubSetSim*, definindo, assim, o cálculo de similaridade entre tabelas na Web que considera termos sinônimos.

$$SubSetSim_{wordnet}(\epsilon_p, \epsilon_d) = \frac{\sum \epsilon_p^i \cdot \eta = \epsilon_d^j \cdot \eta^{(max(wordnet(\epsilon_p^i, [\epsilon_d^1, \ldots, \epsilon_d^m])))}}{min(m, n)} \tag{4.14}$$

Vale observar que o foco aqui foi no tratamento de sinônimos dos cabeçalhos das tabelas, visto que, na identificação de tabelas de mesmo domínio, considera-se mais importante que os cabeçalhos sejam similares por caracterizarem melhor entidades de um domínio. Vale observar também que a função *wordnet* pode ser substituída por um método de comparação de sinônimos de outra base de conhecimento que suporte relacionamentos de sinonímia ou que recupere conceitos a partir de termos possibilitando a comparação desses conceitos.

Com base nestas equações para cálculo de similaridade, métodos específicos para comparação de tabelas na Web foram propostos e avaliados, conforme segue.

4.4 MÉTODOS

Os métodos de comparação de tabelas na Web são apresentados nesta seção. Alguns deles foram criados desconsiderando a padronização de cabeçalhos e dados de tabelas para fins de comparação, ou seja, desconsideram o tratamento dos diversos tipos de tabelas na

Web. Esses métodos são chamados neste trabalho de *Métodos Básicos*. Por outro lado, também foram definidos métodos que consideram a padronização de tabelas, sendo estes denominados *Métodos com Padronização*. Ambos os tipos de métodos são detalhados a seguir.

4.4.1 Métodos Básicos

4.4.1.1 Basic Matcher

O primeiro método básico proposto chama-se *Basic Matcher* (BM). Ele segue exatamente a equação geral de comparação de tabelas (Equação 4.1), onde existe a comparação de conjuntos de cabeçalhos e de dados sem nenhum tratamento especial. Essa comparação de conjuntos foi feita utilizando a métrica *SubSetSim* tanto para cabeçalhos quanto para dados. Este método também não possui tratamento de sinônimos.

Dessa forma, o BM compara os diferentes tipos de cabeçalhos e dados somente como dois grandes conjuntos, não importando se são cabeçalhos verticais, horizontais, se os dados estão em tabelas aninhadas e etc. O Algoritmo 1 descreve o funcionamento do BM. A recuperação de cabeçalhos e dados se dá pela recuperação das tags `<th>` e `<td>` respectivamente.

Algoritmo 1: Basic Matcher

Entrada: t1 e t2
Saída: Escore de similaridade
início
1 H_1 = BuscaElementosPelaTagTH(t1);
2 H_2 = BuscaElementosPelaTagTH(t2);
3 D_1 = BuscaElementosPelaTagTD(t1);
4 D_2 = BuscaElementosPelaTagTD(t2);
5 Sim_H = SubSetSim(H_1, H_2) $*w_H$;
6 Sim_D = SubSetSim(D_1, D_2) $*w_D$;
retorna $Sim_H + Sim_D$
fim

4.4.1.2 Basic Matcher Plus

Além dos cabeçalhos e dados, verificou-se que documentos HTML possuem algumas *tags* que dão destaque ao texto, como por exemplo, as *tags* de título `<h1>`, `<h2>`, `<h3>` e a *tag* de negrito `<b>`. Ainda, é possível acrescentar às *tags* HTML atributos como *id*, *classe* e *descrição* que especificam o que aquela *tag* representa, como por exemplo:

```
<table id='tabela-pessoas' class='tabela-full-width'>
```

Considerando essas características do HTML, foi desenvolvido um segundo método básico de comparação de tabelas chamado *Basic Matcher Plus* (BM+).

Primeiramente, acrescentou-se as *tags* de destaque `<h1>`, `<h2>`, `<h3>` e `<b>` no conjunto de valores de cabeçalho. Assim sendo, essas *tags* são consideradas na comparação $Sim(H_1, H_2)$ da Equação 4.1. A seguir, acrescentou-se um novo tipo de conjunto de valores a ser comparado: os atributos existentes nas tags `<table>`. O BM+ é representando pela Equação 4.15, onde $Sim(A_1, A_2)$ é a comparação dos atributos das tabelas t_1 e t_2 utilizando a métrica *SubSetSim*, e w_H, w_D e w_A são os pesos dos cabeçalhos, dados e atributos, respectivamente, sendo $w_H + w_D + w_A = 1$.

$$\begin{aligned} Sim(t_1, t_2) = {} & Sim(H_1, H_2) * w_H + \\ & Sim(D_1, D_2) * w_D + \\ & Sim(A_1, A_2) * w_A \end{aligned} \tag{4.15}$$

O Algoritmo 2 descreve o BM+, sendo a recuperação de cabeçalhos e dados novamente através das tags, porém para cabeçalhos acrescentam-se as tags de destaque além da `<th>`.

Algoritmo 2: Basic Matcher Plus

Entrada: t1 e t2
Saída: Escore de similaridade
início
1 $Tags = th$, $h1$, $h2$, $h3$, b;
2 H_1 = BuscaElementosPelasTags($Tags$,t1);
3 H_2 = BuscaElementosPelasTags($Tags$,t2);
4 D_1 = BuscaElementosPelaTagTD(t1);
5 D_2 = BuscaElementosPelaTagTD(t2);
6 A_1 = BuscaAtributosHTML(t1);
7 A_2 = BuscaAtributosHTML(t2);
8 Sim_H = SubSetSim(H_1, H_2) $*w_H$;
9 Sim_D = SubSetSim(D_1, D_2) $*w_D$;
10 Sim_A = SubSetSim(A_1, A_2) $*w_A$;
retorna $Sim_H + Sim_D + Sim_A$
fim

4.4.1.3 Header Matcher Wordnet

Além desses dois métodos, definiu-se um método básico que considera o tratamento de sinônimos. Ele é denominado *Header Matcher Wordnet* (HMW). Como o nome já sugere, ele considera somente os cabeçalhos na comparação e utiliza o Wordnet para a verificação de sinônimos. O método determina a similaridade de acordo com a Equação 4.16.

$$Sim(t_1, t_2) = SubSetSim_{Wordnet}(H_1, H_2) \tag{4.16}$$

O Algoritmo 3 ilustra o funcionamento do HMW, o qual é muito semelhante ao BM, porém, como já foi dito, utiliza o Wordnet na comparação de cabeçalhos.

Algoritmo 3: Header Matcher Wordnet

Entrada: t1 e t2
Saída: Escore de similaridade
início

```
1  H1 = BuscaElementosPelaTagTH(t1);
2  H2 = BuscaElementosPelaTagTH(t2);
3  D1 = BuscaElementosPelaTagTD(t1);
4  D2 = BuscaElementosPelaTagTD(t2);
5  Sim_H = SubSetSim_Wordnet(H1, H2) *w_H;
6  Sim_D = SubSetSim(D1, D2) *w_D;
   retorna Sim_H + Sim_D
```

fim

O Algoritmo 4 mostra de forma simplificada o *SubbetSim* com o uso do Wordnet. Como é possível observar, entre as linhas 7 e 9 é calculada a similaridade entre dois cabeçalhos utilizando a métrica de similaridade de string Cosine (a qual poderia ser trocada por outra métrica). A seguir, é calculada a similaridade utilizando o Wordnet, o qual retorna um valor de similaridade dependendo do quão sinônimas são as palavras, e por fim é selecionado o maior valor entre os dois cálculos de similaridade (linha 9).

Algoritmo 4: $SubSetSim_{Wordnet}$

Entrada: H_1, H_2
Saída: Escore de similaridade
início
1 $MenorConjunto$ = BuscaMenorConjunto(H_1, H_2);
2 $MaiorConjunto$ = BuscaMaiorConjunto(H_1, H_2);
3 MaximasSimilaridades = 0;
4 **para** *cada* h_{menor} *em* $MenorConjunto$ **faça**
5 MaximaSimilaridade = 0;
6 **para** *cada* h_{maior} *em* $MaiorConjunto$ **faça**
7 Sim = Cosine(h_{menor},h_{maior}) ;
8 SimWordNet = WordNet(h_{menor},h_{maior});
9 MaiorValor = Maior($Sim, SimWordNet$) ;
10 MaximaSimilaridade = Maior($MaiorValor, MaximaSimilaridade$);
fim
11 MaximasSimilaridades += MaximaSimilaridade;
fim
retorna
$MaximasSimilaridades/MenorConjunto.Tamanho$
fim

4.4.2 Métodos com Padronização

Além dos métodos básicos, métodos que consideram a padronização de tabelas com estruturas heterogêneas são propostos nesta dissertação.

4.4.2.1 Standardized Matcher

O primeiro dos métodos com padronização denomina-se *Standardized Matcher* (SM). Ele considera a padronização de cabeçalhos e de dados conforme as Equações 4.10 e 4.11, porém não trata sinônimos.

O Algoritmo 5 descreve o funcionamento do SM. Sendo que nas funções Busca Cabeçalho Horizontal e Busca Cabeçalho Vertical os cabeçalhos são recuperados por linha e/ou por coluna. Isso é feito percorrendo as tags HTML e recuperando as tags `<th>` de cada linha ou coluna. Nesse caso, existem alguns tratamentos como veri-

ficar se existe apenas uma tag `<th>` na linha ou coluna e verificar se os cabeçalhos não são de tabelas aninhadas. Também com relação às funções Busca Dados Nao Estruturados e Busca Dados Estruturados o HTML é novamente percorrido e são recuperados dados utilizando a tag `<td>` e tabelas aninhadas com a tag `<table>` respectivamente. Nesses métodos de recuperação de dados novamente é tomada a precaução de não misturar os tipos de dados, verificando se os dados não estruturados não pertencem a uma tabela aninhada. Todos esses tratamentos são feitos utilizando a biblioteca de tratamento de HTML em java Jsoup (hedley, 2009), a qual permite a manipulação do HTML utilizando seletores CSS.

Com relação aos pesos aplicados em todos os métodos, com exceção dos pesos w_H e w_D que, conforme descrito no Capítulo 5, recebem uma maior diferenciação, todos são ajustados para serem iguais e sua soma igual a 1. Também esses pesos são reajustados conforme a necessidade. Por exemplo, na inexistência de tabelas aninhadas ou cabeçalhos horizontais, seus pesos são ajustados para zero e os demais são rebalanceados.

Algoritmo 5: Standardized Matcher

Entrada: t1 e t2
Saída: Escore de similaridade
início
1 Sim_H = ComparaCabeçalhos(t_1, t_2) $*w_H$;
2 Sim_D = ComparaDados(t_1, t_2) $*w_D$;
retorna $Sim_H + Sim_D$
fim
Função ComparaCabeçalhos(t_1, t_2)
início
1 H_{1H} = BuscaCabeçalhoHorizontal(t_1);
2 H_{1V} = BuscaCabeçalhoVertical(t_1);
3 H_{2H} = BuscaCabeçalhoHorizontal(t_2);
4 H_{2V} = BuscaCabeçalhoVertical(t_2);
5 Sim_{HH} = SubSetSim(H_{1H}, H_{2H}) $*\mathrm{W}_{HH}$;
6 Sim_{HV} = SubSetSim(H_{1H}, H_{2V}) $*\mathrm{W}_{HV}$;
7 Sim_{VH} = SubSetSim(H_{1V}, H_{2H}) $*\mathrm{W}_{VH}$;
8 Sim_{VV} = SubSetSim(H_{1V}, H_{2V}) $*\mathrm{W}_{VV}$;
retorna $Sim_{HH} + Sim_{HV} + Sim_{VH} + Sim_{VV}$
fim
Função ComparaDados(t_1, t_2)
início
1 D_1 = BuscaDadosNaoEstruturados(t_1);
2 D_2 = BuscaDadosNaoEstruturados(t_2);
3 T'_1 = BuscaDadosEstruturados(t_1);
4 T'_2 = BuscaDadosEstruturados(t_2);
5 Sim_{DD} = SubSetSim(D_1, D_2) * w_{DD};
6 Sim_{TT} = StandardizedMatcher(T'_1, T'_2) * w_{TT};
7 Sim_{DT} = ComparaDadosComTabela(D_1, T'_2) * w_{DT};
8 Sim_{TD} = ComparaDadosComTabela(D_2, T'_1) * w_{TD};
retorna $Sim_{DD} + Sim_{TT} + Sim_{DT} + Sim_{TD}$
fim
Função ComparaDadosComTabela(D, T)
início
1 D_T = BuscaDadosNaoEstruturados(T);
2 H_T = BuscaElementosPelaTagTH(T);
3 T'_T = BuscaDadosEstruturados(T);
4 Sim_{DD} = SubSetSim(D, D_T) * w_{DD};
5 Sim_{DH} = SubSetSim(D, H_T) * w_{DH};
6 Sim_{DT} = ComparaDadosComTabela(D, T'_T) * w_{DT};
retorna $Sim_{DD} + Sim_{DH} + Sim_{DT}$
fim

4.4.2.2 Header Standardized Matcher

Outro método definido, semelhante ao SM, é o *Header Standardized Matcher* (HSM). O Algoritmo 6 detalha o funcionamento do HSM que, como o nome já diz, compara somente os cabeçalhos das tabelas utilizando a padronização descrita pela Equação 4.10. Este método também não considera a análise de sinônimos e, da mesma forma que o SM, ele faz a recuperação dos tipos de cabeçalhos utilizando as funções Busca Cabeçalho Horizontal e Busca Cabeçalho Vertical.

Algoritmo 6: Header Standardized Matcher

Entrada: t1 e t2
Saída: Escore de similaridade
início
1 Sim_H = ComparaCabeçalhos(t_1, t_2;
retorna Sim_H
fim
Função ComparaCabeçalhos(t_1, t_2)
início
1 H_{1H} = BuscaCabeçalhoHorizontal(t_1);
2 H_{1V} = BuscaCabeçalhoVertical(t_1);
3 H_{2H} = BuscaCabeçalhoHorizontal(t_2);
4 H_{2V} = BuscaCabeçalhoVertical(t_2);
5 Sim_{HH} = SubSetSim(H_{1H}, H_{2H}) $*\mathrm{W}_{HH}$;
6 Sim_{HV} = SubSetSim(H_{1H}, H_{2V}) $*\mathrm{W}_{HV}$;
7 Sim_{VH} = SubSetSim(H_{1V}, H_{2H}) $*\mathrm{W}_{VH}$;
8 Sim_{VV} = SubSetSim(H_{1V}, H_{2V}) $*\mathrm{W}_{VV}$;
retorna $Sim_{HH} + Sim_{HV} + Sim_{VH} + Sim_{VV}$
fim

4.4.2.3 Métodos com Wordnet

Além dos métodos que não consideram o tratamento de sinônimos, outros dois métodos que utilizam o Wordnet para tal tarefa são propostos: o *Standardized Wordnet Matcher* (SWM) e o *Header Wordnet Standardized Matcher* (HWSM). Ambos são métodos que estendem o SM e o HSM respectivamente. Portanto, seus algoritmos são basicamente os mesmos, com exceção da função Compara Cabeçalhos, que em ambos utiliza o SubSetSim complementado com o Wordnet, o qual já foi apresentado no Algoritmo 4.

4.4.2.4 Método Híbrido

Um método denominado *Hibrid* foi igualmente proposto e avaliado experimentalmente para averiguar a hipótese de combinar contribuições desta dissertação com um bom *baseline*. Com base em experimentos prévios, decidiu-se utilizar o baseline *HeaderSim*, que é o mesmo utilizado por Fan et al. (2014). Com isso, foi construído o método *Hibrid* que combina a comparação de cabeçalhos do *HeaderSim* e a comparação de dados com padronização proposta nessa dissertação.

O Algoritmo 7 demonstra de forma simplificada o funcionamento do Hibrid. Ele novamente é muito semelhante ao SM, porém, como é possível observar na linha 1 da função principal, ele utiliza o *HeaderSim* na comparação de cabeçalhos.

Algoritmo 7: Hibrid

Entrada: t1 e t2
Saída: Escore de similaridade
início
1 Sim_H = HeaderSim(t_1, t_2) $*w_H$;
2 Sim_D = ComparaDados(t_1, t_2) $*w_D$;
retorna $Sim_H + Sim_D$
fim
Função ComparaDados(t_1, t_2)
início
1 D_1 = BuscaDadosNaoEstruturados(t_1);
2 D_2 = BuscaDadosNaoEstruturados(t_2);
3 T'_1 = BuscaDadosEstruturados(t_1);
4 T'_2 = BuscaDadosEstruturados(t_2);
5 Sim_{DD} = SubSetSim(D_1, D_2) * w_{DD};
6 Sim_{TT} = Hibrid(T'_1, T'_2) * w_{TT};
7 Sim_{DT} = ComparaDadosComTabela(D_1, T'_2) * w_{DT};
8 Sim_{TD} = ComparaDadosComTabela(D_2, T'_1) * w_{TD};
retorna $Sim_{DD} + Sim_{TT} + Sim_{DT} + Sim_{TD}$
fim
Função ComparaDadosComTabela(D, T)
início
1 D_T = BuscaDadosNaoEstruturados(T);
2 H_T = BuscaElementosPelaTagTH(T);
3 T'_T = BuscaDadosEstruturados(T);
4 Sim_{DD} = SubSetSim(D, D_T) * w_{DD};
5 Sim_{DH} = SubSetSim(D, H_T) * w_{DH};
6 Sim_{DT} = ComparaDadosComTabela(D, T'_T) * w_{DT};
retorna $Sim_{DD} + Sim_{DH} + Sim_{DT}$
fim

Todos esses métodos propostos foram avaliados através de experimentos preliminares, que são descritos no próximo capítulo.

5 EXPERIMENTOS

Este capítulo apresenta avaliações experimentais realizadas com os métodos propostos nesta dissertação para a comparação de tabelas na Web. Inicialmente, justifica-se a função de similaridade de string utilizada nos métodos e, na sequência, apresenta-se os resultados obtidos com a avaliação dos métodos e algumas variações dos mesmos.

5.1 FUNÇÕES DE SIMILARIDADE

Conforme comentado no capítulo anterior, foram selecionadas sete funções de similaridade de string para serem utilizadas com a métrica *SubSetSim* na comparação:

- Cosine
- Monge Elkan
- Chapman Length
- Euclidean Distance
- Jaro Winkler
- Jaccard
- Soundex

Para determinar quais métricas seriam mais adequadas na comparação de tabelas HTML, alguns experimentos de comparação de tabelas foram realizados utilizando o método BM visto na Seção 4.4.1.

5.1.1 Massa de Dados

Uma massa de dados com 450 tabelas HTML contendo informações sobre o laboratório LISA (UFSC) e informações sobre projetos Wikimedia foi utilizada na realização desses experimentos. Essas tabelas foram extraídas a partir dos seguintes endereços da internet:

- http://lisa.inf.ufsc.br/wiki/
- https://meta.wikimedia.org/wiki/Main_Page

5.1.2 Metodologia dos Experimentos

Para a realização dos experimentos, primeiramente foi necessário analisar manualmente a massa de dados e anotar quais tabelas eram realmente semelhantes entre si. Esse processo foi realizado comparando cada par de tabelas para determinar se elas eram similares ou não. Isso gerou um arquivo no formato JSON contendo o identificador de cada tabela relacionado com suas tabelas similares. Esse arquivo é exemplificado pela Figura 18, onde 1 seria o identificador de uma tabela e 2, 13, 44 e 52 são as tabelas similares a 1.

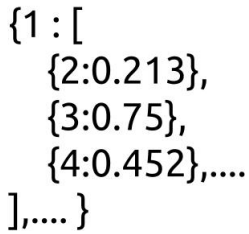
Figura 18: Arquivo JSON relacionando uma tabela com as similares

A seguir, foi executado o método BM comparando todas as tabelas entre si e gerados escores de similaridade para cada comparação. Com isso foi gerado um outro arquivo no formato JSON, relacionando cada tabela com as demais, porém dessa vez incluindo os escores de similaridade, como é demonstrado pela Figura 19.

Os dois arquivos JSON foram comparados, e foi aplicado um *threshold* com valor 0,7 sobre os escores de similaridade. Esse threshold indica que, para escores acima desse valor, as tabelas seriam consideradas similares. O valor 0,7 em específico foi aplicado por ter sido o que gerou melhor desempenho em termos de similaridades corretas em experimentos preliminares.

Uma vez realizada essa comparação dos arquivos JSON, foi possível

Figura 19: Arquivo JSON contendo os escores de similaridade entre as tabelas



calcular as medidas clássicas de avaliação *Precision*, *Recall* e *F-Measure* representadas pelas equações 5.1, 5.2 e 5.3, respectivamente. A medida *Precision*, indica o quão corretos estão os resultados ditos similares, a medida *Recall* indica a relação do que foi dito similar com o que realmente é similar e a medida *F-Measure* é a média harmônica das duas primeiras medidas.

$$Precision = \frac{VerdadeiroPositivos}{VerdadeiroPositivos + FalsoPositivos} \tag{5.1}$$

$$Recall = \frac{VerdadeiroPositivos}{VerdadeiroPositivos + FalsoNegativos} \tag{5.2}$$

$$F - Measure = 2 \times \frac{precision \times recall}{precision + recall} \tag{5.3}$$

5.1.3 Resultados

A Figura 20 apresenta os resultados dos experimentos com as métricas de similaridade de String em termos de *F-Measure*.

Como é possível observar nesta Figura, as funções que obtiveram melhores resultados foram a Cosine e a Jaccard. Com base nesses resultados, optou-se por utilizar somente a função Cosine nos métodos de comparação propostos. Mesmo assim, vale lembrar que a escolha da função depende muito do domínio dos dados comparados.

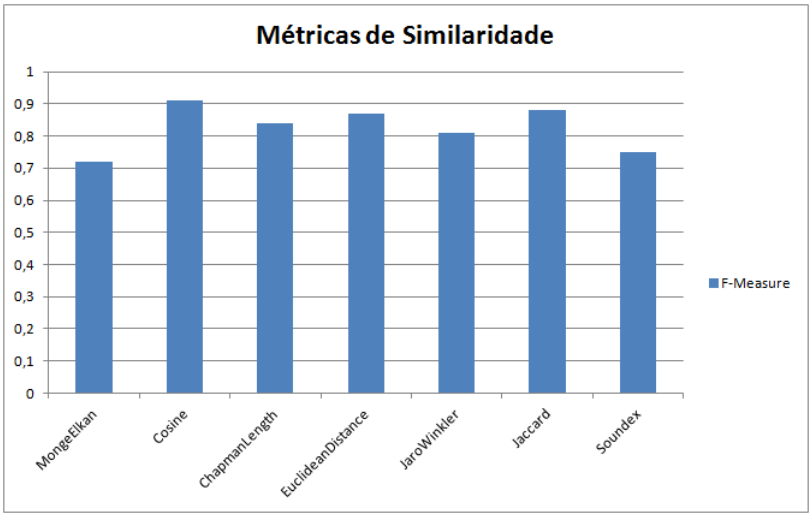
5.2 AVALIAÇÃO DOS MÉTODOS PROPOSTOS

Esta seção detalha os experimentos realizados com os métodos de comparação de tabelas na Web vistos no Capítulo 4.

5.2.1 Massa de Dados

Os experimentos com os métodos de similaridade propostos consideraram como massa de dados tabelas HTML previamente extraídas da *Wikipedia*. Essa massa de dados foi novamente analisada manualmente e foram anotadas quais tabelas eram realmente similares entre

Figura 20: Métricas de similaridade



si, gerando um arquivo JSON exemplificado pela Figura 18.

Essas tabelas pertencem a um grande número de domínios do conhecimento incluindo: pessoas, filmes, dados demográficos e esportes. Elas também contêm diferentes estruturas: 72% delas são horizontais, 10% são matriciais, 1,6% são verticais e 0.65% possuem tabelas aninhadas. Cabe salientar que, para 16% das tabelas coletadas, não foi possível classificar o seu tipo devido a características incomuns em sua estrutura HTML, como tabelas sem cabeçalhos, tabelas que são somente listas de valores, e tabelas com cabeçalhos aleatoriamente posicionados.

5.2.2 Metodologia dos Experimentos

A metodologia utilizada nesses experimentos foi a mesma utilizada para verificar a melhor métrica de similaridade. Para cada método de comparação visto no Capítulo 4, foi novamente gerado um arquivo JSON contendo os escores de similaridade, foram comparados com os dados obtidos manualmente e gerados valores de *Precision*, *Recall* e *F-Measure*.

Como dito anteriormente, dado que os cabeçalhos possuem grande influência na identificação de similaridade entre as tabelas na Web, os

valores dos pesos nos métodos foram definidos, em todos os testes, para 0,8 para a similaridade de cabeçalhos e 0,2 para a similaridade de dados. Esses pesos foram assim definidos após a execução de experimentos prévios que os indicaram como sendo os melhores.

Diferente dos experimentos com as métricas de similaridade de string, na comparação dos métodos variou-se ainda o *threshold* final para avaliar como ele afeta os resultados. Os valores de *threshold* testados foram 0.6, 0.7 e 0.8. *Thresholds* acima e abaixo desses valores se mostraram ineficazes por prejudicarem muito os valores de *Recall* e *Precision*, respectivamente. Portanto, tais thresholds foram desconsiderados.

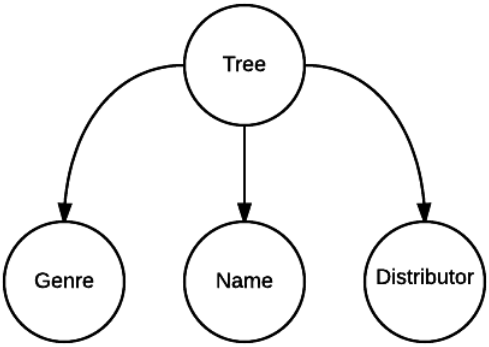
5.2.3 Baselines

Alguns baselines foram também considerados na realização desses experimentos. Um dos baselines utilizados foi o RTED (PAWLIK; AUGSTEN, 2011), devido ao fácil acesso ao seu código fonte. Entretanto, conforme descrito na Seção 3.3, o RTED, em sua essência, compara somente estruturas de árvores. Por isso, foi necessário adaptar as tabelas HTML para serem comparadas pelo RTED de forma mais justa. Essa adaptação foi realizada definindo estruturas em árvore somente com os valores dos cabeçalhos das tabelas. A Figura 21 apresenta a adaptação da Tabela 7 para o RTED.

A adaptação do RTED recupera os cabeçalhos das tabelas da mesma forma que os métodos propostos anteriormente, ou seja, percorrendo a árvore HTML e recuperando as devidas tags. Dessa forma, o RTED não compara a estrutura das tabelas, mas o seu conteúdo de forma semelhante ao método HSM proposto, que compara somente cabeçalhos. Cabe salientar que alguns experimentos iniciais foram realizados considerando também os dados das tabelas nessa adaptação do RTED. Porém, os resultados não foram satisfatórios pois os dados variam bastante. Devido a isso, utilizou-se somente os cabeçalhos nessa adaptação da abordagem.

Outros *baselines* considerados nos experimentos foram os mesmos utilizados no trabalho de Fan et al. (2014): *HeaderSim* e *InstanceSim*. Estes métodos calculam a similaridade de tabelas na Web levando em conta os cabeçalhos e os dados, respectivamente.

Figura 21: Tabela adaptada para o RTED



5.2.4 Análise dos Resultados

Os métodos descritos no capítulo anterior, assim como os *baselines* apresentados, foram executados sobre um conjunto de mil tabelas HTML. Variou-se, ainda, o *threshold* final para avaliar como ele afeta os resultados. Os valores de *threshold* testados foram 0.6, 0.7 e 0.8. As Figuras 22, 23 e 24 apresentam os resultados em termos de *Precision*, *Recall* e *F-Measure*, respectivamente.

Figura 22: Resultados em termos de Precision

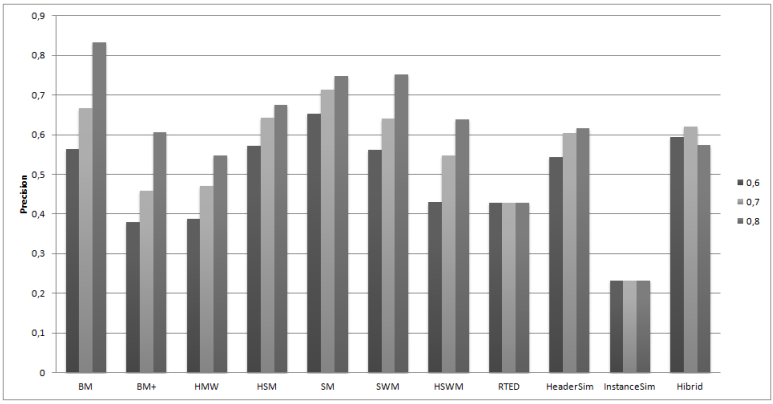
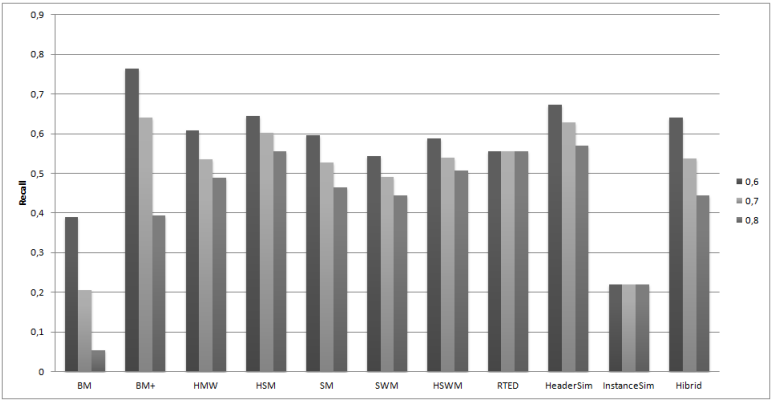


Figura 23: Resultados em termos de Recall



Como é possível observar nos resultados, os métodos com padronização, ou seja, os métodos que lidam com representações heterogêneas de tabelas na Web durante a comparação, em particular HSM, SM e SWM, obtiveram os melhores resultados em termos de *Precision*. Isto significa que a maioria das tabelas indicadas como similares por esses métodos são realmente similares (verdadeiros positivos). O método BM também obteve bons resultados em termos de *Precision*, porém seu *Recall* foi bastante baixo, indicando que ele não identificou como similares diversas tabelas que na realidade são similares (falsos negativos).

O método HSM obteve, em média, os melhores resultados em termos de *F-measure* dentre os métodos avaliados. Isto indica que a comparação de cabeçalhos é mais relevante para determinar a similaridade entre tabelas na Web de modo geral. Apesar disso, a comparação de dados não pode ser desconsiderada, visto que o método SM obteve bons resultados para alguns valores de *threshold*.

Com relação aos *thresholds*, é possível observar que os valores 0,6 e 0,7 geraram, em média, melhores resultados que 0,8. Acredita-se que este fato se deve à grande heterogeneidade das tabelas na Web, o que faz com que *thresholds* mais altos não sejam capazes de recuperar um grande número de tabelas similares.

Outro ponto a salientar é que os métodos HeaderSim e HSM obtiveram resultados muito próximos em termos de *F-measure*. A Figura 25 apresenta um comparativo mais detalhado entre os dois métodos. Como é possível observar, o HSM obteve *F-Measure* e *Precision* superiores ao HeaderSim. Entretanto, HeaderSim obteve melhor *Recall*.

Figura 24: Resultados em termos de F-Measure

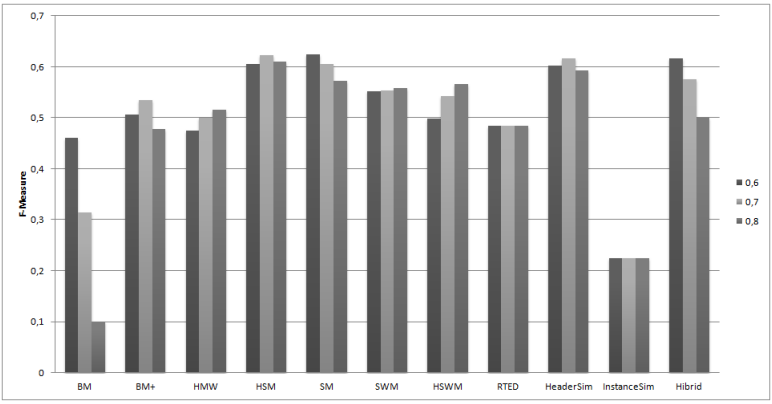


Figura 25: Comparativo entre HSM e HeaderSim

F-Measure			
	0,6	0,7	0,8
HSM	0,606	0,622	0,610
HeaderSim	0,602	0,616	0,592

Precision			
	0,6	0,7	0,8
HSM	0,571	0,643	0,676
HeaderSim	0,543	0,605	0,617

Recall			
	0,6	0,7	0,8
HSM	0,645	0,602	0,555
HeaderSim	0,673	0,629	0,569

Isto indica que o HSM obteve menos falsos positivos comparado com o *baseline*, porém seria interessante tentar melhorar a sensibilidade do HSM para recuperar as tabelas similares que não foram retornadas.

O método *Hibrid* obteve um resultado levemente melhor em termos de *Precision*, principalmente para os *thresholds* menores, se comparado com o HeaderSim. Entretanto, o *Recall* foi inferior a este *baseline* e o seu *F-Measure* foi inferior, na média, ao *F-Measure* do baseline e ao *F-Measure* dos métodos com padronização que obtiveram melhor desempenho. Isto demonstra que os métodos com padronização originais se mostraram melhores, em média, que esta tentativa de hibridização.

Os resultados do método InstanceSim já eram esperados visto que os dados das tabelas na Web variam bastante. Neste trabalho, a definição de similaridade não requer que estes valores sejam exatamente

iguais. Assim sendo, um método que considera somente os dados das tabelas não é a melhor alternativa.

O *baseline* RTED não obteve os melhores resultados, porém é uma alternativa bastante estável visto que seus resultados não variam com o *threshold*. Uma vez que o seu foco não é a comparação de strings, seus escores de similaridade variaram de muito altos a muito baixos. Por isso, para os *thresholds* utilizados nestes experimentos, os resultados não variaram. Para *thresholds* superiores a 0,9 ou inferiores a 0,1 deve haver alguma variação.

Observa-se ainda que o método BM+ obteve os melhores resultados em termos de *Recall* para os *thresholds* 0,6 e 0,7. Acredita-se que isso se deve à comparação de atributos HTML presentes nas *tags table* das tabelas na Web e consideradas pelo método. Aqui vale ressatar que, na construção de páginas Web dinâmicas, utilizam-se modelos HTML estáticos que são preenchidos com dados provenientes de um banco de dados. Desta forma, são construídas várias páginas diferentes utilizando um mesmo modelo. Em geral, quando as tabelas provêm de um mesmo modelo, elas possuem os mesmos atributos HTML, o que produz mais correspondências. Porém, existe o risco de se considerar atributos muito genéricos, o que diminui a precisão do método.

Um outro ponto interessante a destacar é por quê o suporte de uma base de conhecimento, no caso o *Wordnet* nesses experimentos, não aumentou o *F-Measure*. Acredita-se que a principal razão disso seja a grande quantidade de termos considerados sinônimos que não necessariamente pertencem ao mesmo domínio do conhecimento, como por exemplo, *nome* e *ano*, que podem ser propriedades tanto de carros quanto de livros ou mesmo de filmes.

6 CONCLUSÃO

A Web é cada vez mais uma vasta fonte de informação. Entretanto, a comunidade de Banco de Dados ainda carece de abordagens efetivas para a extração, detecção de similaridade e integração de dados úteis para consumo humano em diferentes domínios de aplicação.

Com o intuito de contribuir com essa problemática, essa dissertação propõe métodos de comparação de tabelas na Web que consideram, ao mesmo tempo, as diferentes estruturas das tabelas e o tratamento de sinônimos. Comparando com trabalhos relacionados, não foram encontradas abordagens que lidam com esses dois problemas ao mesmo tempo na comparação de tabelas na Web. Na criação desses métodos foi utilizada a taxonomia proposta por Lautert, Scheidt e Dorneles (2013) para identificar as principais estruturas a serem comparadas nas tabelas. Também esses métodos foram criados a partir de uma arquitetura em comum capaz de ser configurada para comparar um conjunto específico de tabelas na Web e até mesmo para a criação de novos métodos. Os métodos propostos foram avaliados através de alguns experimentos que mostraram alguns resultados promissores para os métodos com padronização definidos nesta dissertação, em particular, o método HSM, o qual obteve melhores resultados gerais e pode apontar uma direção a ser estudada para se obter resultados ainda melhores.

Uma questão relevante a observar é o motivo pelo qual os métodos não obtiveram valores de *F-measure* superiores a 0,6 na maioria dos experimentos realizados com métodos com padronização, devido principalmente aos baixos valores de *Recall* obtidos em alguns testes. Isso se deve à grande heterogeneidade da formatação do HTML das tabelas existentes. Conforme comentado anteriormente, em alguns casos nem mesmo foi possível classificar as tabelas devido a suas características incomuns, como tabelas sem cabeçalhos, tabelas cujas células possuem longas listas de valores ou textos e tabelas com múltiplos atributos no mesmo cabeçalho. Apesar disso, notou-se que o baseline *HeaderSim* também não obteve um valor de *F-measure* muito superior a 0.6, o que demonstra que houve contribuição por parte desta dissertação em termos de determinação de similaridade de tabelas na Web, considerando o melhor desempenho obtido no geral.

Esta primeira versão dos métodos apresenta, como limitação, a consideração apenas dos cabeçalhos e dos dados de tabelas na Web, que são os seus componentes básicos e por isso foram priorizados nesta

dissertação. Para aprimorar futuras versões desses métodos sugere-se, como trabalhos futuros, uma melhor análise dos cabeçalhos assim como comparações mais contextualizadas, como por exemplo, considerar outros dados presentes na página HTML na qual as tabelas estão inseridas, análise da *URL*, além do uso de outras bases de conhecimento.

Também como trabalho futuro, propõe-se um método para detecção de tabelas na Web onde não existam as *tags* `<table>` explicitamente. Atualmente, devido ao surgimento do HTML 5 e CSS 3, tem-se buscado evitar o uso dessas *tags* e tem-se utilizado outras formas de organizar dados na Web. Por isso, uma contribuição interessante para a recuperação de informação na Web seria a detecção desses dados, sejam eles listas ou dados tabulares descritos em outros formatos.

A pesquisa realizada com esta dissertação gerou, até o momento, duas publicações: um artigo completo na X Escola Regional de Banco de Dados (ERBD 2014), que descreve o estado da arte sobre gerenciamento de tabelas na Web, e um artigo completo na 13th International Conference on WWW/Internet (ICWI 2014), que apresenta os métodos propostos e alguns dos experimentos relatados nesta dissertação. Este último evento é classificado como B2 no Qualis da CAPES em Ciência da Computação. Pretende-se produzir e submeter um outro artigo para um evento ou periódico qualificado com uma maior discussão sobre os experimentos realizados, bem como a apresentação de novos experimentos com configurações diferentes e alterações nos métodos, conforme sugerido como trabalhos futuros.

REFERÊNCIAS

BALAKRISHNAN, S. et al. Applying webtables in practice. In: **CIDR**. [S.l.]: www.cidrdb.org, 2015.

BANKO, M.; ETZIONI, O. The tradeoffs between open and traditional relation extraction. In: **ACL**. [S.l.: s.n.], 2008. p. 28–36.

BOLLACKER, K. et al. Freebase: A collaboratively created graph database for structuring human knowledge. In: **Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 2008. (SIGMOD '08), p. 1247–1250.

BRAUNSCHWEIG, K. et al. Column-specific context extraction for web tables. In: **Proceedings of the 30th Annual ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2015. (SAC '15), p. 1072–1077.

CAFARELLA, M. J. et al. Webtables: Exploring the power of tables on the web. **Proc. VLDB Endow.**, VLDB Endowment, v. 1, n. 1, p. 538–549, ago. 2008.

CHAPMAN, S. Simmetrics, a similarity metric library. 2006.

CHEN, S.; MA, B.; ZHANG, K. The normalized similarity metric and its applications. **2013 IEEE International Conference on Bioinformatics and Biomedicine**, IEEE Computer Society, Los Alamitos, CA, USA, v. 0, p. 172–180, 2007.

CHEN, S.; MA, B.; ZHANG, K. On the similarity metric and the distance metric. **Theoretical Computer Science**, v. 410, n. 24–25, p. 2365 – 2376, 2009. Formal Languages and Applications: A Collection of Papers in Honor of Sheng Yu.

COHEN, W. W.; RAVIKUMAR, P. D.; FIENBERG, S. E. A comparison of string distance metrics for name-matching tasks. In: **Proceedings of IJCAI-03 Workshop on Information Integration on the Web (IIWeb-03), August 9-10, 2003, Acapulco, Mexico**. [S.l.: s.n.], 2003. p. 73–78.

CRESTAN, E.; PANTEL, P. Web-scale table census and classification. In: **Proceedings of the Fourth ACM**

International Conference on Web Search and Data Mining. New York, NY, USA: ACM, 2011. (WSDM '11), p. 545–554.

DORNELES, C. F. et al. Measuring similarity between collection of values. In: **Proceedings of the 6th Annual ACM International Workshop on Web Information and Data Management**. New York, NY, USA: ACM, 2004. (WIDM '04), p. 56–63.

EMBLEY, D. W. et al. Factoring web tables. In: **Proceedings of the 24th international conference on Industrial engineering and other applications of applied intelligent systems**. Berlin, Heidelberg: Springer-Verlag, 2011. (IEA/AIE'11), p. 253–263.

FAN, J. et al. A hybrid machine-crowdsourcing system for matching web tables. In: **IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014**. [S.l.: s.n.], 2014. p. 976–987.

HAVELIWALA, T. H. Topic-sensitive pagerank. In: **Proceedings of the 11th International Conference on World Wide Web**. New York, NY, USA: ACM, 2002. (WWW '02), p. 517–526.

HEDLEY, J. **jsoup: Java HTML Parser**. 2009. Disponível em: <http://jsoup.org/>.

JACCARD, P. **Nouvelles recherches sur la distribution florale**. [S.l.: s.n.], 1908.

LAI, P. P. Y. Adapting data table to improve web accessibility. In: **Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility**. New York, NY, USA: ACM, 2013. (W4A '13), p. 33:1–33:4.

LAUTERT, L. R.; SCHEIDT, M. M.; DORNELES, C. F. Web table taxonomy and formalization. **SIGMOD Rec.**, ACM, New York, NY, USA, v. 42, n. 3, p. 28–33, out. 2013.

MERGEN, S. L. S.; FREIRE, J.; HEUSER, C. A. Indexing relations on the web. In: **Proceedings of the 13th International Conference on Extending Database Technology**. New York, NY, USA: ACM, 2010. (EDBT '10), p. 430–440.

MILLER, G. A. Wordnet: A lexical database for english. **Commun. ACM**, ACM, New York, NY, USA, v. 38, n. 11, p. 39–41, nov. 1995.

PAWLIK, M.; AUGSTEN, N. Rted: A robust algorithm for the tree edit distance. **Proc. VLDB Endow.**, VLDB Endowment, v. 5, n. 4, p. 334–345, dez. 2011.

RAMOS, J. Using tf-idf to determine word relevance in document queries. In: **Proceedings of the first instructional conference on machine learning**. [S.l.: s.n.], 2003.

SARMA, A. D. et al. Finding related tables. In: **Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 2012. (SIGMOD '12), p. 817–828.

SEVARAC, Z. et al. **Java Neural Network Framework**. 2008.

SHIMA, H. **WS4J (WordNet Similarity for Java)**. nov. 2014. [Online; accessed 04-November-2014]. Disponível em: <https://code.google.com/p/ws4j/>.

SILVA, F.; MELLO, R. Análise de abordagens para recuperação de informação em tabelas na web. In: **ERBD 2014**. [S.l.: s.n.], 2014.

SON, J.-W.; PARK, S.-B. Web table discrimination with composition of rich structural and content information. **Appl. Soft Comput.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 13, n. 1, p. 47–57, jan. 2013.

SREEJA, S. R.; CHAUDHARI, S. Review of web crawlers. **Int. J. Knowl. Web Intell.**, Inderscience Publishers, Inderscience Publishers, Geneva, SWITZERLAND, v. 5, n. 1, p. 49–61, out. 2014.

VENETIS, P. et al. Recovering semantics of tables on the web. **Proc. VLDB Endow.**, VLDB Endowment, v. 4, n. 9, p. 528–538, jun. 2011.

WANG, J. et al. Understanding tables on the web. In: **Proceedings of the 31st international conference on Conceptual Modeling**. Berlin, Heidelberg: Springer-Verlag, 2012. (ER'12), p. 141–155.

WU, W. et al. Probase: A probabilistic taxonomy for text understanding. In: **Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 2012. (SIGMOD '12), p. 481–492.

YAKOUT, M. et al. Infogather: Entity augmentation and attribute discovery by holistic matching with web tables. In: **Proceedings of**

the 2012 ACM SIGMOD International Conference on Management of Data. [S.l.]: ACM, 2012. (SIGMOD '12), p. 97–108.