# 4
# Identificação de tabelas

A exposição de grande volume de informação na forma de tabela é uma prática comum, principalmente pela facilidade de visualização e entendimento proporcionado por essa estrutura. Seja pela facilidade que proporciona ao leitor, ou mesmo pela facilidade de exposição de informações para a análise humana, essas estruturas estão presentes em vários tipos de documentos, e também em quase todo tipo de mídia.

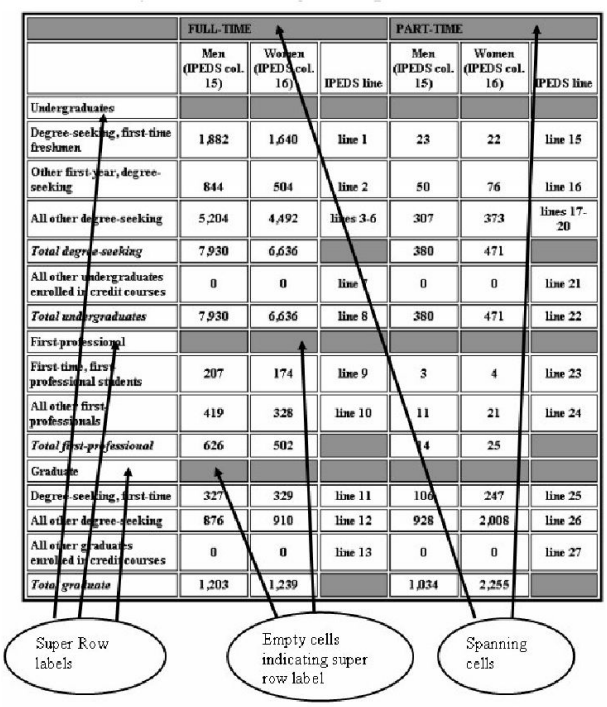


Figura 4.1: Tabela de difícil compreensão retirada de (Tengli et. al., 2004)

Com isso, é possível afirmar que um grande volume de informação na Web é exposto em forma de tabela, fato reforçado pela presença de diversos trabalhos que buscam identificar e extrair informação de tabelas de documentos HTML. Na referência (Pinto et. al., 2003), é destacada a importância do uso de tabelas, ressaltando a capacidade de exposição bidimensional de dados, e também é mencionada as dificuldades impostas pela própria estrutura para a extração de informação.

Diferente das tabelas criadas para o uso lógico, como as tabelas existentes em banco de dados, as tabelas em geral são criadas para a visualização e interpretação de dados por humanos, impondo diversas dificuldades já em sua concepção. Por exemplo, a existência de dados ambíguos é uma realidade e dificulta o problema de extração.

Tabelas que não apresentam ambiguidades também podem ser um desafio para o entendimento automático (realizado por uma máquina) de sua informação. No trabalho (Tengli et. al., 2004), é exposto um exemplo de como a estrutura tabular pode ser desafiadora. Esse exemplo consiste de uma tabela com diversos níveis de rótulos, ou seja, descrições de células, dificultando o cruzamento da célula com seu respectivo rótulo, como pode ser observado na Figura 4.1.

Para um maior entendimento da tarefa de extração de informação de tabelas, essa é dividida em etapas em (Pinto et. al., 2003), que são amplamente adotadas pelos estudos na área. As etapas são:

1 Identificação/localização da tabela;

2 Identificação de linhas e seus tipos;

3 Identificação de colunas e seus tipos;

4 Segmentação da tabela em células;

5 Identificação do tipo da célula (dado ou legenda);

6 Associação da legenda ao dado.

A identificação de uma tabela é uma etapa importante, pois torna possível a aplicação de uma estratégia específica de extração de informação. Outro ponto importante, confirmado pela referência (Pinto et. al., 2003), é que algoritmos de extração de informação de textos não apresentam bons resultados quando aplicados em tabelas.

As etapas seguintes, aplicadas após a identificação da tabela, podem ser interpretadas como um processamento específico ao segmento desse tipo. Assim, nesta dissertação focamos na tarefa de identificação de tabelas, ignorando qualquer tipo de processamento na tentativa de extrair informação dessa estrutura.

Essa abordagem é comum na literatura, existindo trabalhos que utilizam o identificador de tabelas proposto por outros, possibilitando que os trabalhos fiquem concentrados nas tarefas conseguintes à identificação. Um bom exemplo dessa abordagem é o trabalho (Tengli et. al., 2004), que apresenta resultados para a tarefa de extração de informação de tabelas, utilizando o algoritmo descrito em (Wang e Hu, 2002) para realizar a identificação.

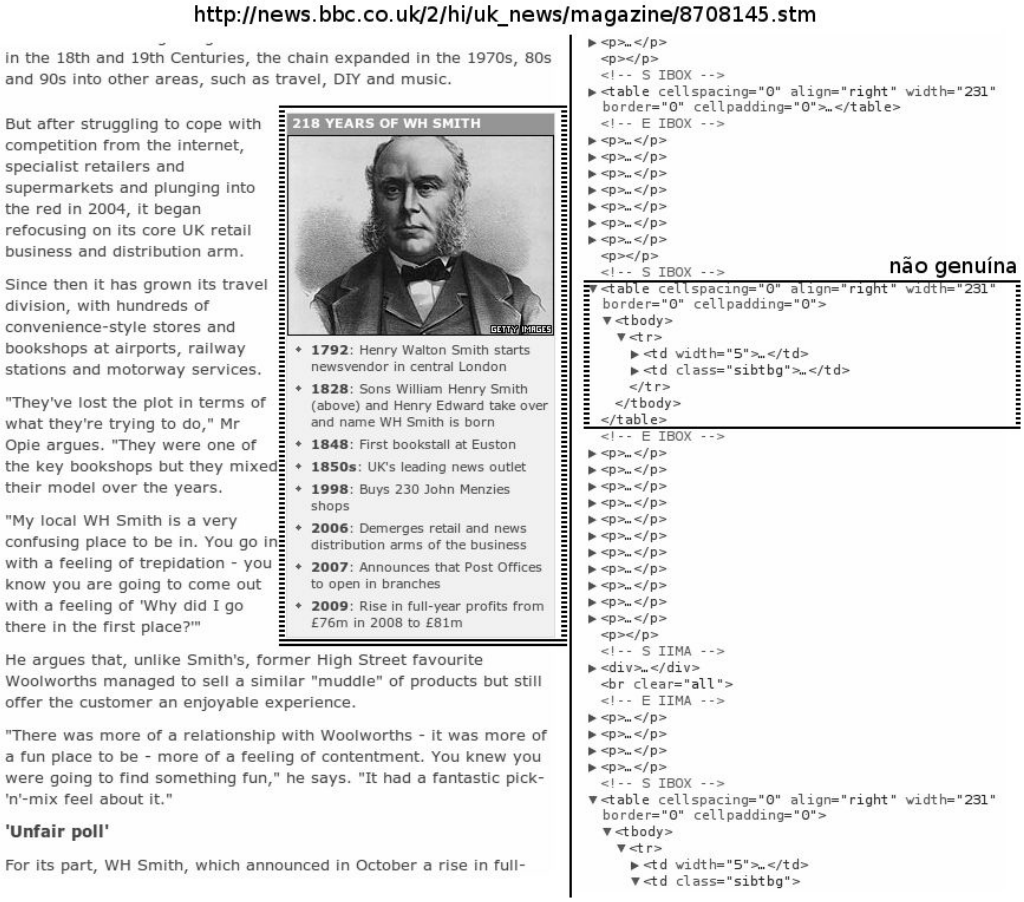


Figura 4.2: Tabela não genuína com a árvore HTML correspondente à direita

No contexto de identificação de tabelas em documentos HTML, os elementos marcados como *table* podem ser obtidos facilmente, o que solucionaria o problema de identificação das tabelas. No entanto, a marcação *table* não é utilizada estritamente para a representação de informação tabular, como visto na Seção 2.1. Com isso, é necessário separar as estruturas marcadas como *table* que apresentam informações tabulares das demais.

Com esse objetivo, a definição apresentada em (Wang e Hu, 2002) é utilizada para separar os elementos marcados com *table* em duas categorias. Wang define que os elementos *table* que apresentam informação de forma bidimensional, ou seja, tabular, são tabelas **genuínas**. Já os demais elementos são tabelas **não genuínas**. As Figuras 4.3 e 4.2 ilustram, respectivamente, um exemplo de tabela **genuína** e um de tabela **não genuína**

Outro ponto apresentado na definição de Wang, é que tabelas genuínas não podem conter tabela dentro dela (tabelas aninhadas). A existência do aninhamento de tabelas torna a estrutura da tabela confusa, podendo ocasionar no aninhamento dos itens da tabela de forma incorreta. Por essa razão, o aninhamento de tabelas não é uma prática comum quando se apresenta

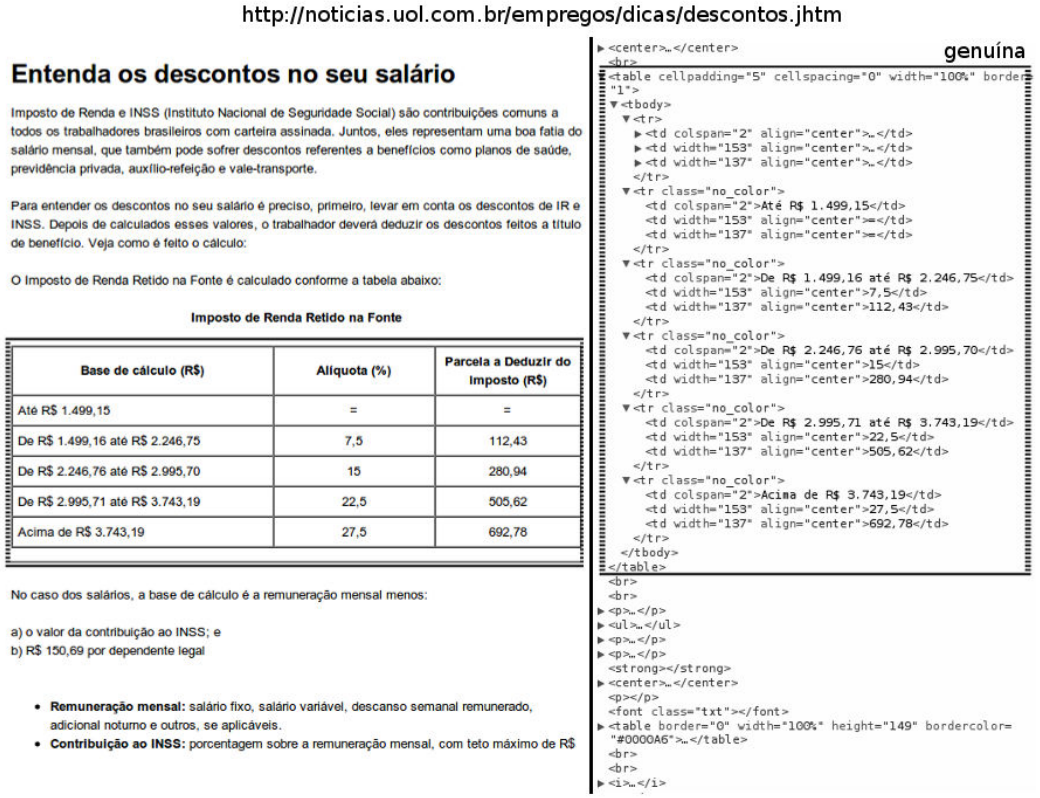


Figura 4.3: Tabela genuína com a árvore HTML correspondente a direita

informação de forma tabular. Então, as tabelas **genuínas** só podem ser as tabelas mais internas na árvore DOM, também chamadas de tabelas folhas.

Finalmente, a tarefa de identificar tabelas em documentos HTML pode ser descrita como a capacidade de separar os elementos *table* em **genuínos** e **não genuínos**. O objetivo deste capítulo é descrever uma técnica capaz de classificar um elemento *table* como genuíno utilizando a semelhança entre subestruturas da árvore DOM, e comparar os resultados obtidos com essa técnica a resultados já conhecidos.

## 4.1 Trabalhos existentes

O trabalho (Wang e Hu, 2002) apresenta resultados importantes para a área de identificação de tabelas utilizando técnicas de aprendizado de máquina. Mais especificamente, são utilizadas árvores de decisão e SVM, para criação de modelos de classificação de tabelas genuínas. Wang e Hu apresentam três grupos de atributos, utilizados para a criação de seus modelos de classificação:

– Atributos de exibição: onde são calculados médias e desvios da utilização de marcações como tr, td, th, br;

– Atributos de tipo de dados: onde são calculados médias e desvios dos tipos de dados encontrados dentro da estrutura como string, inteiros, reais e datas, dentre outros;

– Atributos de grupos de palavras: onde são analisadas as palavras utilizadas em tabelas genuínas, sendo criado um índice de palavras-chave.

Quando são utilizados os três grupos de atributos, o classificador de Wang e Hu apresenta 95,88% de F1 para a tarefa. Porém, a utilização de palavras-chave e tipos de dados restringe a aplicação desses modelos a um mesmo idioma.

Repare que os atributos de exibição não utilizam informações de posicionamento da tela, ou seja, são informações de *tag* que podem ser obtidos da árvore DOM, não sendo necessária a geração da representação visual do documento HTML em um navegador. Esse ponto é importante, pois torna a obtenção desses atributos mais rápida, diferente de outras técnicas que utilizam o posicionamento $(x, y)$ de elementos na tela.

Os atributos do conjunto de *exibição* podem ser ressaltados como os mais interessantes, porque podem ser replicados para diversos idiomas, são simples de reproduzir e, quando utilizados sem os demais atributos, atingem 87,70% de F1. Por esse motivo, esse resultado é utilizado para a comparação com as técnicas discutidas neste capítulo.

Gatterbauer e Bohunsky (Gatterbauer e Bohunsky, 2006) modelam a tarefa de extração de informação de forma diferente da exposta anteriormente. Ao invés de separar a tarefa de extração de informação em seis sub-tarefas, os autores listam apenas três sub-tarefas que podem ser interpretadas como: identificação da tabela *(table location)*, que localiza uma tabela dentro de um documento HTML; identificação das células *(table recognition)*, que localiza as células de uma tabela; interpretação das células *(table interpratation)*, que identifica a informação contida em uma célula e a sua relação com um rótulo. Porém, pode ser observado que a identificação de tabela corresponde a uma das seis sub-tarefas utilizadas na modelagem de extração de informação apresentada em (Wang e Hu, 2002).

No trabalho de Gatterbauer e Bohunsky, é utilizada a informação visual do documento, para apresentar um algoritmo *bottom-up* que identifica os segmentos do tipo tabela. Tal abordagem é interessante e apresenta resultados perto de 90% de F1. Porém, a necessidade de calcular a posição relativa de cada bloco é um ponto negativo do algoritmo proposto. O processamento para o cálculo das informações visuais eleva significantemente o tempo de processamento de um documento, como mencionado no Capítulo 2.

Gatterbauer e Bhunsky apresentam resultados da aplicação de seu algoritmo no corpus fornecido por Wang e Hu. O experimento apresentado utiliza uma amostragem de cinquenta documentos, dentre os 1392, e obtém 89,2% de F1.

## 4.2 Métricas de avaliação

Por ser uma métrica clara e amplamente adotada em outros estudos da área, a métrica proposta em (Wang e Hu, 2002) foi adotada para a avaliação da qualidade dos algoritmos de identificação de tabelas genuínas. Essa métrica se resume à medida de *recall*, que busca avaliar a capacidade de não perder tabelas genuínas classificando-as como não genuínas; *precision*, que busca identificar a capacidade de não classificar tabelas que não são genuínas como genuínas; e F-measure, que é a média harmônica entre o *recall* e a *precision*.

Na Tabela 4.2, são exemplificados os casos que podem ocorrer durante a classificação de tabelas genuínas, sendo **VP** (verdadeiro positivo) o número de tabelas genuínas que são classificadas como genuínas, **FN** (falso negativo) o número de tabelas genuínas que são classificadas como não genuínas, **FP** (falso positivo) o número de tabelas não genuínas que são classificadas como genuínas e **VN** (verdadeiro negativo) o número de tabelas não genuínas que são classificadas como não genuínas. Em seguida, são apresentadas as formas de cálculo das métricas citadas.

| Classe correta | Classificado como genuína | Classificado como não genuína |
|---|---|---|
| Genuína | VP | FN |
| Não genuína | FP | VN |

Tabela 4.1: Classificações possíveis de uma tabela para o cálculo das métricas

A métrica *recall* representa a quantidade de objetos positivos que foram identificados corretamente.

$$\textbf{recall}, R = \frac{VP}{VP+FN}$$

A métrica *precision* representa a fração de objetos positivos que foram identificados como tal.

$$\textbf{precison}, P = \frac{VP}{VP+FP}$$

As métricas *recall* e *precision* podem ser resumidas pela medida $F$, que é calculada da seguinte forma:

$$F1 = \frac{2RP}{R+P} = \frac{2VP}{2VP+FP+FN}$$

A medida F1 é a média harmônica entre *precision* e *recall*. Existem variações da medida $F$ onde uma equação $F_b$ é apresentada, porém essa não será abordada neste trabalho, por não ser utilizada.

## 4.3 Corpus de exploração

O corpus disponibilizado por Wang et al (Wang e Hu, 2002) foi utilizado na tarefa de identificação de tabelas. Esse corpus é constituído de 1392 documentos HTML, obtidos em 200 web sites distintos. Nesses documentos existem 14.609 marcações de tabela (tag table), sendo 11.472 tabelas internas (tabelas folhas). Das tabelas folhas, 1.755 são tabelas genuínas e 9.717 não genuínas. A anotação do corpus é realizada no próprio HTML com o atributo adicional **genuinetable="yes"**, o que torna possível avaliar o resultado com base na própria árvore.

Para a experimentação foram separados aproximadamente 20% do corpus (280 documentos) com 2.442 tabelas internas, das quais 393 são genuínas. Essa fração do corpus é chamada conjunto de treino e é utilizada para a verificação da corretude dos algoritmos apresentados, assim como para o aprendizado de alguns dos parâmetros necessários para a execução dos algoritmos. É importante destacar que foi escolhido trabalhar apenas com 20% dos documentos os algoritmos apresentados necessitam apenas de poucos parâmetros, sendo possível ajustá-los com poucos exemplos. Outro ponto positivo para utilizarmos poucos documentos no treino é que sobraram mais documentos para testar a capacidade de generalização dos algoritmos. O restante (1.112 documentos), chamado conjunto de teste, foi utilizado para verificar os resultados das técnicas implementadas nesta dissertação e também para a comparação com os demais trabalhos relacionados.

Alguns trabalhos na literatura também utilizam esse corpus, adotando técnicas diferentes de arranjo e utilização. Com isso, algumas organizações especiais foram realizadas, para possibilitar a comparação desses trabalhos com esta dissertação. Essas organizações serão explicadas quando utilizadas.

## 4.4 Abordagem proposta

A utilização da estrutura da árvore DOM pode ser vista como um processo natural para a identificação de tabelas genuínas. Analisando a criação de um documento HTML, é possível notar o sequenciamento de elementos *tr* , que apresentam uma linha, com diversos *td* , que apresentam as células. Para

exemplificar, a Figura 4.3 apresenta a árvore gerada pelo documento HTML e a visualização desse documento em um navegador.

Na tentativa de identificar as estruturas repetidas que indicam a existência de uma tabela genuína, foi aplicado, inicialmente, o algoritmo de **Casamento Simples** (CS), descrito na Seção 3.1.2. A motivação principal é eliminar as estruturas enraizadas por um elemento *table* que não apresentam pelo menos um casamento em sua estrutura. Tal abordagem visa diminuir o tamanho do espaço de busca, já que elimina, do conjunto de subárvores enraizadas pelo elemento *table*, aquelas estruturas que aparentemente não são tabelas genuínas.

Na Tabela 4.2, são expostos os resultados dos algoritmos propostos sobre o conjunto de treino. Nessa tabela, é possível observar a diminuição esperada de elementos, quando aplicado o algoritmo CS. É interessante destacar que, mesmo utilizando a função **Distância em Caracteres** (DC), os resultados foram surpreendentes. A diminuição de aproximadamente 50% do conjunto de tabelas é interessante, pois elimina poucas tabelas genuínas, apresentando recall de 97.20%. Essa diminuição também significa que as tabelas genuínas têm sub-estruturas semelhantes exatamente como foi assumido por hipótese. Infelizmente, como também pode ser observado na Tabela 4.2, existem outros tipos de estruturas que, por definição, não são tabelas genuínas, mas também apresentam subestruturas semelhantes. Com isso, apenas a informação da existência de uma repetição de subestruturas não resolve o problema de classificação de tabelas genuínas. No entanto, a técnica de detecção de subestruturas semelhantes serve ao propósito esperado: diminui o conjunto de busca sem eliminar as tabelas genuínas o que, novamente, reforça a hipótese que as tabelas genuínas apresentam uma subestrutura similar.

| função utilizada | Recall | Precision | F1 | # retornadas |
|---|---|---|---|---|
| todas tags table | 100 | 00.16 | 00.27 | 2442 |
| CS + DC | 97.20 | 29.56 | 45.34 | 1292 |
| CS + DG | 96.69 | 30.30 | 46.14 | 1254 |
| CS + DT | 96.18 | 30.07 | 45.81 | 1257 |
| CA | 81.67 | 29.15 | 42.97 | 1101 |

Tabela 4.2: Resultados sobre o conjunto de treino sem utilizar técnicas específicas para tabelas

Como analisado na Seção 3.1.1, quando utilizamos a função de Levenshtein como função de distância, o problema da atribuição do maior peso a *tags* com nomes grandes pode aparecer. Para verificar se esse comportamento desfavorece a classificação de tabelas, foi testada a variação em que a

unidade de medida é a *tag*, função de **Distância em Tags** (DG). A mudança na unidade de medida proporciona uma pequena melhora no *recall*, como pode ser observado na segunda linha da Tabela 4.2.

Os resultados da aplicação direta dos algoritmos propostos, sem especialização, se mostraram satisfatórios, dado que o objetivo é diminuir o conjunto de busca sem eliminar as tabelas genuínas. No entanto, o algoritmo CS (descrito na Seção 3.1.2) limita-se a realizar uma busca par a par, ou seja, em conjuntos generalizadores de tamanho 1. Essa abordagem pode ser considerada muito rígida para a identificação de alguns tipos de tabelas. Por esse motivo, foi testado o algoritmo de busca **Casamento de Árvores** (CA) (descrito na Seção 3.1.2), que faz busca de conjuntos generalizadores de tamanho maior que 1. Como pode ser observado na última linha da Tabela 4.2, a função CA apresentou resultados próximos aos resultados obtidos com a função CS, o que consolida os resultados obtidos.

Observando as medidas de *recall* apresentadas na Tabela 4.2, é possível verificar que poucas tabelas genuínas são perdidas com as diversas técnicas: foram perdidas no máximo 18.33% e na média 7.06%. Com isso, é possível afirmar que as tabelas genuínas apresentam uma estrutura como esperada. Mais de 80% das tabelas genuínas de nosso conjunto de treino apresentam uma estrutura com sequenciamento de subárvores semelhantes. Vale lembrar que o algoritmo CA obriga que o sequenciamento tenha, necessariamente, um conjunto generalizador de tamanho 2. Por esse motivo, tabelas genuínas que utilizam elementos de separação, como uma coluna uma linha vazia ou elementos invisíveis como spam ou comentários são perdidas quando esse algoritmo é utilizado.



### Técnicas especializadas em tabelas

Conforme pode ser observado na coluna *precision* da Tabela 4.2, um grande número de estruturas é classificado como tabelas genuínas incorretamente. Porém, esse comportamento é inicialmente esperado. Para contornar essa classificação incorreta, foi avaliado manualmente dentre alguns documentos do conjunto de treino, os tipos de estruturas que apresentavam subestrutura semelhante e não deveriam ser classificadas como tabelas genuínas. Nessa direção, foram identificadas diversas estruturas de listas que não deveriam ser classificadas como tabela genuína, segundo a definição de tabela adotada por Wang e Hu (Wang e Hu, 2002). Esse comportamento ocorre, pois as estruturas de lista utilizam a *tag* table para o sequenciamento, porém não contêm informação em mais de uma coluna. Finalmente, para eliminar essas estruturas, foi desenvolvida uma especialização que pode ser aplicada sobre

o conjunto de estruturas que apresentam subestruturas similares. Essa especialização parte da hipótese de que as tabelas genuínas podem ser separadas das listas, observando a existência de informação em mais de uma coluna de uma tabela, ou seja, contando o número de linhas e colunas dessa tabela que contém informação.

| função utilizada | % Recall | % Precision | % F1 | # retornadas |
|---|---|---|---|---|
| RL | 95.67 | 63.40 | 76.26 | 593 |
| RL + CS + DC | 92.87 | 85.28 | 88.91 | 428 |
| RL + CS + DG | 92.36 | 85.81 | 88.97 | 423 |
| RL + CS + DT | 91.85 | 85.74 | 88.69 | 421 |
| RL + CA | 78.88 | 88.06 | 83.22 | 352 |

Tabela 4.3: Resultados no conjunto de treino utilizando a função razão de linhas (RL) com as técnicas de semelhança de estrutura

A regra de especialização criada foi a **Razão de Linhas** (RL). Essa regra tenta separar as tabelas de outras estruturas que utilizam a *tag* table, contando as células (*tags td*) e as linhas (*tags tr*), para calcular a razão das células por linhas. Se a razão for maior so que dois, a tabela é classificada como genuína. Ou seja, se uma tabela apresentar em média pelo menos duas células em cada linha, ela é classificada como genuína.

A regra RL foi aplicada separadamente sobre o conjunto de treino, para avaliar a qualidade obtida somente com essa regra. Na Tabela 4.3 é apresentado o resultado desse teste, tornando possível avaliar seu impacto sobre os resultados gerais, quando utilizamos a razão de linhas junto a outras técnicas.

Ainda na Tabela 4.3, pode ser observado o resultado da união dos algoritmos de casamento de subárvores com a regra razão de linhas. Os resultados apresentados exemplificam o ganho de informação proporcionado pela topologia da estrutura do documento, tornando um pouco mais claro os benefícios da utilização da informação estrutural.

Finalmente, realizamos uma validação sobre o conjunto denominado teste, onde a qualidade da abordagem apresentada pode ser avaliada. Para esse experimento utilizamos somente o algoritmo de busca de subestruturas semelhantes que atingiu o melhor resultado. Esse algoritmo foi o **Casamento Simples** com a função de distância de Levenshtein modificada para que a unidade de comparação seja a tag. Como pode ser observado na Tabela 4.4, os resultados se mantiveram próximos aos obtidos no conjunto de treino com uma diferença de 3.16% ( de 88.97% para 85.81%) o que sugere que os algoritmos propostos são robustos e confiáveis.

A capacidade de generalização dos algoritmos é um aspecto importante a ser citado. Nenhuma etapa de treinamento é necessária para a execução do algoritmo, já que o conjunto de treino foi utilizado apenas para o ajuste de alguns parâmetros necessários para a execução. Outro aspecto interessante a ser ressaltado é o tamanho dos conjuntos (treino e teste), já que o conjunto de treino é menor que o conjunto de teste, o que vai em direção contrária ao costume para a aplicação de técnicas de aprendizado de máquina. Todos esses argumentos podem ser utilizados para afirmar que a abordagem proposta pode minimizar o *overfitting*, isto é, diminuir o conhecimento a priori sobre o conjunto estudado.

| função utilizada | % Recall | % Precision | % F1 | # retornadas |
|---|---|---|---|---|
| todas as tags table | 100 | 15.11 | 26.25 | 8721 |
| RL | 95.97 | 58.21 | 72.47 | 2173 |
| CS + DG | 96.50 | 28.24 | 43.70 | 4503 |
| RL + CS + DG | 92.48 | 80.03 | 85.81 | 1523 |

Tabela 4.4: Resultado das técnicas sobre o conjunto de teste



### Utilizando aprendizado de máquina

Para verificar com maior rigor a contribuição do uso da informação topológica, implementamos a técnica apresentada por Wang e Hu (Wang e Hu, 2002) utilizando a árvore de decisão C4.5 (Quinlan, 1993). Avaliamos os resultados com a mesma técnica de validação cruzada com 10 partições utilizada pelos autores. Foram utilizados somente os atributos de *layout* descritos por Wang e Hu, pois consideramos esses atributos os mais relacionados ao tipo de informação obtida a partir da estrutura, tornando possível avaliar a qualidade de nosso trabalho. Os resultados, apresentados pelo classificador utilizando tais atributos, foram 85,60% de F1, como pode ser observado na última linha da Tabela 4.5, ficando a 2.1% dos resultado apresentado pelos autores de 87,70%. Acreditamos que essa diferença seja devido à maneira de contar as *tags*, e também pela variedade de modelos e parâmetros existentes dentre as técnicas de aprendizado de máquina. No entanto, achamos razoável afirmar que a reprodução dos atributos descritos por Wang e Hu foi bem sucedida.

Dando continuidade à avaliação da nossa abordagem, foi gerado um atributo que assume valor 1, quando a nossa abordagem classifica a tabela como genuína e 0 quando contrário. Avaliando nosso método com uma validação cruzada, utilizando o atributo descrito, e foi obtido 86,70% de F1. Como pode ser observado na Tabela 4.5, utilizar esse atributo de classificação, apresenta

resultados melhores do que a reprodução do trabalho (Wang e Hu, 2002). Finalmente, realizamos um último experimento: juntamos os atributos de *layout* propostos por Wang e Hu ao nosso atributo de classificação. Como pode ser observado na Tabela 4.5 o resultado é 90,40% de F1, demonstrando que a técnica apresentada contribui para a classificação das tabelas genuínas.

É importante ressaltar que para a realização desses últimos experimentos foi utilizado o corpus completo, não sendo respeitada a divisão em conjunto de treino e teste. Por esse motivo, lembramos que o ajuste de alguns parâmetros de nosso algoritmo foram utilizados 20% do corpus. Infelizmente essa foi a melhor maneira encontrada para que fosse possível a comparação dos resultados obtidos nesta dissertação com os resultados reportados por Wang e Hu e com os demais trabalhos da literatura.

| função utilizada | % Recall | % Precision | % F1 | # retornadas |
|---|---|---|---|---|
| (Wang e Hu, 2002) layout | 87.24 | 88.15 | 87.70 | - |
| RL + CS + DG + reprodução layout Wang e Hu | 93.20 | 87.80 | 90.40 | 1635 |
| RL + CS + DG | 92.80 | 81.40 | 86.70 | 1628 |
| reprodução layout Wang e Hu | 88.70 | 82.60 | 85.60 | 1557 |

Tabela 4.5: Resultados de aprendizado de máquina com validação cruzada sobre o corpus completo



### Comparando os resultados

Para concluir a análise da abordagem proposta é apresentada a Tabela 4.6, onde é comparado o melhor resultado obtido por nossa abordagem com os demais resultados conhecidos na área de identificação de tabela genuínas. Os resultados foram separados em dois grupos. No grupo superior, são apresentados os resultados que é possível realizar uma comparação direta, pois esses trabalhos utilizam o mesmo corpus que foi utilizado durante esta dissertação ou utilizam técnicas que podem ser consideradas similares às apresentadas nesta dissertação. No grupo inferior, são listados os resultados que se destacam dentre os trabalhos da área e foram apresentados na Seção Trabalhos existentes (4.1).

Como pode ser observado, dentre os trabalhos onde é possível fazer uma comparação direta dos resultados, nossos resultados são bastante atraentes, já que superam a qualidade dos demais. Outro fato importante é o tempo de processamento, como já discutido na seção dos trabalhos relacionados, e também a ausência de treinamento para que nossa heurística funcione. Quanto

[1] 20% do corpora de Wang e Hu

| função utilizada | % Recall | % Precision | % F1 |
|---|---|---|---|
| RL + CS + DG + reprodução layout Wang e Hu | 93.20 | 87.80 | 90.40 |
| (Gatterbauer e Bohunsky, 2006)[1] | 84.20 | 94.10 | 89.20 |
| (Wang e Hu, 2002) layout only | 87.24 | 88.15 | 87.70 |
| (Wang e Hu, 2002) | 95.98 | 95.81 | 95.89 |
| (Gatterbauer e Bohunsky, 2006) | 89.00 | 96.70 | 92.80 |
| (Pinto et. al., 2003) | - | - | 91.80 |

Tabela 4.6: Comparação dos resultados de identificação de tabelas genuínas com os trabalhos relacionados

aos demais resultados, é difícil fazer qualquer tipo de afirmação quanto à qualidade, porém nossos resultados aparentemente estão próximos dos demais, ou seja, têm qualidade acima de 90% de F1. Acreditamos que a adição de nossa técnica como um pré-processamento pode melhorar a qualidade de qualquer técnica já apresentada na literatura. Isso pode ser sugerido, já que em nossos experimentos é possível observar uma melhora na qualidade da classificação, quando utilizados os atributos de *layout* descritos por Wang e Hu.

| função utilizada | tempo total | média por documento |
|---|---|---|
| RL | 340.36 | 0.244 |
| todas as tabelas | 348.41 | 0.250 |
| CS + DG | 363.87 | 0.261 |
| RL + CS + DG | 370.04 | 0.265 |
| CS + DT | 379.67 | 0.272 |
| CA + DG | 425.71 | 0.305 |
| CS + DC | 748.57 | 0.537 |

Tabela 4.7: Tempo de processamento em segundos dos 1393 documentos com os algoritmos propostos

A Tabela 4.7 apresenta os tempos de processamento dos 1393 documentos, utilizando as abordagens discutidas neste capítulo. Repare, na linha 2 da Tabela 4.7 que para a obtenção de todas as tabelas de um documento HTML é necessário mais tempo do que quando adicionamos o processamento RL. Isso acontece pois o número de tabelas processadas pelo avaliador é muito superior ao da linha 1, ocasionando um maior tempo total de processamento. Esse ponto é interessante, pois demonstra como a diminuição no número de tabelas processadas interfere diretamente no tempo de processamento.

Outro ponto interessante, que pode ser observado quando comparamos a última linha com as demais, é a redução do tempo de processamento quando utilizamos a técnica DG em vez de DC. A diminuição de tempo observada é esperada, pois quando a unidade de comparação é a *tag* (processamento

DG), a quantidade de operações passa a ser em função do número de nós e não do tamanho dos nomes das tags (número de nós vezes o nome da *tag*). Podemos observar que os tempos de processamentos são próximos, sendo o melhor algoritmo em relação à qualidade da identificação (RL+CS+DG) também muito bom em tempo de processamento.