USO DE INFORMAÇÃO ESTRUTURAL PARA MELHORAR QUALIDADE DE BUSCA EM COLEÇÕES WEB

DAVID BRAGA FERNANDES DE OLIVEIRA

USO DE INFORMAÇÃO ESTRUTURAL PARA MELHORAR QUALIDADE DE BUSCA EM COLEÇÕES WEB

Tese apresentada ao Programa de Pós--Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação.

Orientador: Berthier Ribeiro-Neto

Belo Horizonte

Abril de 2010

DAVID BRAGA FERNANDES DE OLIVEIRA

USING STRUCTURAL INFORMATION TO IMPROVE SEARCH IN WEB COLLECTIONS

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Doctor in Computer Science.

ADVISOR: BERTHIER RIBEIRO-NETO

Belo Horizonte

April 2010

Oliveira, David Braga Fernandes de

O48u Using Structural Information to Improve Search in Web Collections / David Braga Fernandes de Oliveira. — Belo Horizonte, 2010

xxxviii, 104 f. : il. ; 29cm

Tese (doutorado) — Federal University of Minas Gerais

Orientador: Berthier Ribeiro-Neto

1. information retrieval. 2. block-weight functions. 3. block classes. 4. ranking. I. Título.

CDU 519.6*73

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Uso de informação estrutural para melhorar qualidade de busca em coleções web

DAVID BRAGA FERNANDES DE OLIVEIRA

Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:

[illegible]

Prof. Berthier Ribeiro de Araújo Neto - Orientador
Departamento de Ciência da Computação - UFMG

[illegible]

Prof. Edleno Silva de Moura - Co-orientador
Departamento de Ciência da Computação - UFAM

[illegible]

Prof. Carlos Alberto Heuser
Departamento de Informática - UFRGS

[illegible]

Prof. Marcos André Gonçalves
Departamento de Ciência da Computação - UFMG

[illegible]

Prof. Nivio Ziviani
Departamento de Ciência da Computação - UFMG

[illegible]

Prof. Ruy Luiz Milidiu
Departamento de Informática - PUC - RJ

Belo Horizonte, 14 de maio de 2010.

Aos meus pais.

Agradecimentos

Após uma longa jornada de muito trabalho e dedicação à esta obra, difícil não lembrar de todo o apoio recebido por inúmeras pessoas para que este sonho se tornasse uma realidade.

Primeiramente, gostaria de agradecer aos meus pais, Jorge Fernandes de Oliveira e Evenilda Braga Fernandes de Oliveira, que tornaram possíveis este e muitos outros sonhos. Exemplos de excelente conduta, excelente caráter, não se intimaram perante a luta de tornar este mundo um lugar melhor. Sua abnegação na ajuda ao próximo, sua coragem e seu exemplo de vida são como mil sois iluminando o meu caminho. Terei em vocês a minha eterna inspiração. Devo tudo a vocês, e por isso dedico toda a essa obra a vocês.

Outra fonte de inspiração é meu irmão, Horácio Antônio Braga Fernandes de Oliveira, atual professor doutor da Universidade Federal do Amazonas. Meu irmão caçula, quatro anos mais novo do que eu, e que terminou o doutorado dois anos antes de mim. Exemplo de excelente índole, excelente caráter e dedicação. Foi um grande companheiro e cúmplice desta longa jornada. Serei eternamente grato a você.

Continuando os agradecimentos aos membros da família, preciso lembrar com muito carinho da minha irmã, Ligia Fernandes Abdalla, e de seu esposo, João Hugo Abdalla. Que bom que estaremos mais próximos a partir de agora, e então poderei desfrutar muito mais do convívio de vocês.

Aproveitando que estou falando das pessoas mais queridas e mais importantes da minha vida, não posso esquecer de Joice Machado. Nossa amizade foi um marco na minha vida, e é uma das pessoas mais valiosas para mim. Poucos dias há que eu deixe de pensar em você. Você sempre foi o apoio na medida e na hora certa, e participou de vários dos momentos mais importantes para mim.

Como não poderia deixar de ser, gostaria de agradecer imensamente ao Prof. Berthier Ribeiro-Neto, meu orientador. Espero sinceramente que nossa jornada não tenha terminado por aqui, e que ainda poderemos contar um com o outro em trabalhos futuros. Com você aprendi muito mais do que apenas conhecimentos acadêmicos.

Você me mostrou muito sobre a vida, sobre como abraçar uma oportunidade e sobre como buscar aquilo que desejamos. Que você continue tendo muito sucesso em sua caminhada, e que continue sendo essa inspiração, tanto para mim como para tantos outros.

Um agradecimento muito especial devo ao Prof. Edleno Silva de Moura. Você será sempre uma inspiração e um exemplo de dedicação e competência. Certamente, nada disso teria acontecido sem você. Você não apenas me incentivou a iniciar o doutorado, como representou um apoio imprescindível ao longo de todos esses anos. Fico muito feliz em saber que nossa caminhada juntos não termina por aqui, e que ainda teremos muitos papers e muitos trabalhos feitos em parceria.

Não poderia deixar de lembrar também do pessoal do Latin, laboratório onde permaneci boa parte do doutorado. Meus sinceros agradecimentos à: Rickson Guidolini, Guilherme Menezes, Anisio Mendes Lacerda, Thierson Rosa, Wladmir Brandão e Fabiano Botelho.

Da mesma forma, meus muito sinceros agradecimentos ao pessoal do GTI/UFAM. Em especial, gostaria de lembrar de: Karane Mariano Vieira, Márcio Vidal, Roberto Oliveira, Klessius Berlt, André Carvalho, Márcia Sampaio, Eli Cortez, Raoni Simoes Ferreira, Mauro Rojas Herrera e Jucimar.

Também queria expressar minha sincera gratidão à todos os funcionários da secretaria de nossa pós-graduação. Em especial, gostaria de lembrar de Renata Viana Moraes Rocha. Acho que a Renata era uma das poucas pessoas além de mim que realmente desejavam que meu doutorado terminasse logo, dadas às minhas constantes idas à secretaria. Sou muito grato à você, Renata, por todo o apoio que você me prestou ao longo de todos esses anos.

Uma pessoa que foi muito importante durante todo esse tempo foi Moisés Carvalho, da minha turma de doutorado. Você foi um verdadeiro companheiro e cúmplice de todo o desenrolar desses últimos 5 anos. Há muito tempo já te disse e continuo a dizer: devo muito a você. Sua amizade, nossas conversas, tudo o que aprendi contigo, são coisas que jamais esquecerei.

Os 5 anos de doutorado foram marcados por uma reviravolta em minhas concepções filosóficas e religiosas. Durante minhas buscas conheci pessoas das mais diversas crenças, e muitas delas foram muito importantes para mim. Seria injusto citar apenas alguns poucos nomes neste espaço. Ainda assim, não poderia deixar de registrar a minha mais profunda gratidão e carinho à Pai Joaquim, preto velho, querido irmão. Você me mostrou um mundo novo, e mudou a minha vida para sempre. Muito obrigado por tudo.

Por último, e obviamente não menos importante, gostaria de expressar meus mais sinceros agradecimentos ao Augusto de Paula Pereira, à Josephine David Jorge, à Monique Olive Costa e Lara, e ao Firmino José Leite. O apoio e a amizade de vocês foram fundamentais nos últimos meses do trabalho na tese, coisa que jamais esquecerei. Durante muito tempo colherei o fruto da amizade de vocês.

"A vida é uma peça de teatro que não permite ensaios. Por isso, cante, chore, dance, ria e viva intensamente, antes que a cortina se feche e a peça termine sem aplausos."

(Charles Chaplin)

Resumo

Ao contrário dos documentos de textos planos, as páginas Web são comumente compostas de diferentes segmentos ou blocos tais como barras de navegação, formulários de interação, seções principais, anúncios de privacidade e *copyright*. Este é um fato de interesse, visto que trabalhos anteriores demonstraram que esses diferentes segmentos, que podem ser identificados automaticamente nas páginas Web, podem ser usados para melhorar atividades de recuperação de informação tais como busca, análise de Web links, e mineração de dados na Web. Por exemplo, informação sobre os blocos das páginas pode ser usada para estimar pesos de termos de acordo com a ocorrência desses termos dentro dos blocos (ao invés de dentro das páginas). Como consequência, a importância da ocorrência de um termo em uma página Web pode variar dependendo de sua localização (ou bloco) dentro das páginas Web. Por exemplo, a ocorrência de um termo no *conteúdo principal* de uma página pode ser mais importante para tarefas de ordenação de documentos que a ocorrência deste mesmo termo no *menu* desta página.

Nesta tese, são investigados diferentes meios de como melhorar processos de busca por informação em coleções de páginas Web através do uso da estrutura das páginas. Para tanto, nós propomos: (i) um novo modelo de representação do conteúdo de Web sites em sistema de recuperação de informação que leva em consideração a estrutura interna das páginas; (ii) um método de identificação automática da estrutura interna das páginas Web, de acordo com o modelo de representação do conteúdo de Web sites proposto neste trabalho; e (iii) um conjunto de nove funções capazes de distinguir o impacto de ocorrências de termos dentro dos blocos das páginas, on invés de dentro das páginas completas. Estas funções, que são usadas para compilar uma versão modificada do modelo BM25, possuem a vantagem de não requerer processos de aprendizagem nem qualquer outro tipo de intervenção manual para computar as ordenações de respostas para as consultas, tal como requerido por trabalhos anteriores.

Usando quatro coleções de páginas Web, foram executados experimentos para comparar nossos métodos baseados em blocos com (i) dois modelos de recuperação de informação baseados em blocos propostos na literatura, e com (ii) um método tradicional de *ranking* que não usa informação de blocos. Os resultados indicam que nossos métodos baseados em blocos são capazes de obter ganhos de qualidade de resposta em relação a todos os baselines, gerando ganhos médios de precisão de 5% a 20%.

Além de melhorar a efetividade da tarefa de busca, nossos métodos baseados em blocos reduzem o tamanho do índice usado nos processos de busca em até 27.9% quando comparado com os baselines, diminuindo os requisitos de armazenagem do sistema e o custo de processamento das consultas.

Abstract

Unlike plain text documents, Web pages are commonly composed of distinct segments or blocks such as service channels, decoration skins, navigation bars, main sections, copyright and privacy announcements. This is of interest because previous works have demonstrated that these different segments or blocks, which can be automatically identified in Web pages, can be used to improve information retrieval tasks such as searching, Web link analysis and Web mining. For instance, block information can be used to estimate term weights according to the occurrence of the terms inside blocks (instead of inside pages). As a consequence, the importance of each term occurrence may vary depending on its location (or block) within the Web page. The motivation is that, for instance, the occurrence of a term in the *main contents* section of a Web page is expected to be more important for ranking purposes than an occurrence of that same term in a *menu* of that page.

In this thesis, we investigate how to improve retrieval tasks by exploring the block structure of Web pages. For that, we propose: (i) a new model for representing the content of Web sites in information retrieval systems that takes into account the internal structure of their Web pages and the relationship of the structural components found on the pages; (ii) a method to automatically identify the internal structure of the Web pages, according to the model of representing the Web sites contents proposed in this work; and (iii) a set of 9 block-weight functions to distinguish the impact of term occurrences inside page blocks, instead of inside whole pages. These functions, that are used to compile a modified BM25 ranking function, have the advantage of not requiring a learning process nor any type of manual intervention to compute the ranking, as required by previous works.

Using 4 distinct Web collections, we ran extensive experiments to compare our block-weight ranking formulas with 3 other baselines: (i) a BM25 ranking applied to full pages, (ii) a BM25 ranking applied to pages after templates removal, and (iii) a BM25 ranking that takes into account best blocks. Our methods suggest that our block-weighting ranking method is superior to all baselines across all collections we used and that average gain in precision figures from 5% to 20% are generated.

Further, our methods decrease the cost of processing queries when compared to the systems using no structural information, decreasing indexing storage requirements and increasing the speed of query processing.

Resumo Estendido

Ao contrário dos documentos de textos planos, as páginas Web são comumente compostas de diferentes segmentos ou blocos tais como barras de navegação, formulários de interação, seções principais, anúncios de privacidade e *copyright*. Este fato é de interesse, visto que trabalhos anteriores demonstraram que esses diferentes segmentos, que podem ser identificados automaticamente nas páginas Web, podem ser usados para melhorar atividades de recuperação de informação tais como busca, análise de Web links, e mineração de dados na Web. Por exemplo, informação de bloco pode ser usada para estimar pesos de termos de acordo com a ocorrência desses termos dentro dos blocos (ao invés de dentro das páginas). Como consequência, a importância de cada ocorrência de termo pode variar dependendo de sua localização (ou bloco) dentro das páginas Web. A motivação é que, por exemplo, a ocorrência de um termo no *conteúdo principal* de uma página pode ser mais importante para propósitos de ordenação de documentos que a ocorrência deste mesmo termo no *menu* desta página.

Nesta tese, são investigados diferentes meios de como melhorar processos de busca por informação em coleções de páginas Web através do uso da estrutura das páginas. Para tanto, são propostos: (i) um novo modelo de representação do conteúdo de Web sites em sistema de recuperação de informação que leva em consideração a estrutura interna das páginas; (ii) um método de identificação automática da estrutura interna das páginas Web, de acordo com o modelo de representação do conteúdo de Web sites proposto neste trabalho; e (iii) um conjunto de 9 funções baseadas em blocos capazes de distinguir o impacto de ocorrências de termos dentro dos blocos das páginas, on invés de dentro das páginas completas. Estas funções, que são usadas para compilar uma versão modificada da função de ordenação BM25, possuem a vantagem de não requerer processos de aprendizagem nem qualquer outro tipo de intervenção manual para computar as ordenações de respostas para as consultas, tal como requerido por trabalhos anteriores.

Usando quatro coleções de páginas Web, foram rodados experimentos para comparar nossos métodos baseados em blocos com (i) dois modelos de recuperação de

informação baseados em blocos propostos na literatura, e com (ii) um método tradicional de *ranking* que não usa informação de blocos. Os resultados indicam que nossos métodos baseados em blocos são aptos a melhorar os resultados em relação a todos os baselines. Para o IG, uma de nossas coleções de Web sites adotada em nossos experimentos, foi obtida uma melhoria de 0.62 para 0.75 na métrica MAP, quando comparado nosso método com um modelo da literatura que não usa informação de blocos. Para a coleção CNN, outra coleção de Web sites, o sistema resultou uma melhoria de 0.69 to 0.80 na métrica MAP.

Além de melhorar a efetividade de tarefas de busca, nossos métodos baseados em blocos reduzem o tamanho da lista invertida em até 27.9% quando comparado com os baselines, diminuindo os requisitos de armazenagem do sistema e o custo de processamento das consultas.

Representado Web Sites como Blocos

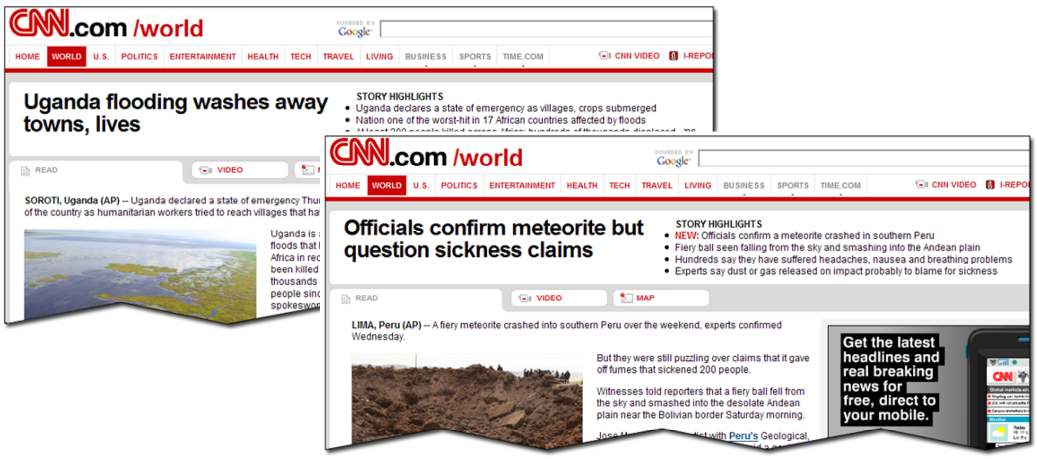
Quando os usuários observam uma página através de um Web browser, eles podem distinguir diferentes partes dela, tais como barras de navegação, formulários de interação, sessões principais, etc. Cada uma destas partes tem uma função particular dentro da página, e são genericamente referenciados como *blocos*. Nesta seção, apresentamos um conjunto de definições que são usadas para caracterizar a divisão das páginas em blocos.

Definição 1. *Um bloco é uma região lógica de uma página que (i) não é aninhada com nenhum outro bloco e (ii) é representado por uma tupla (l, c), onde l é o rótulo do bloco, representado como uma string, e c é uma porção de texto do bloco.*

A divisão de uma página em um conjunto de blocos não sobrepostos deve ser feita de acordo com a percepção dos usuários sobre a correta divisão lógica das páginas. Neste trabalho, o rótulo l é representado como o caminho entre a raiz da árvore DOM[1] da página e o bloco. A árvore DOM é uma representação hierárquica que provê meios de descrever a estrutura e o *layout* das páginas Web. Ela é usualmente adotada em trabalhos de pesquisa relacionados com a estrutura das páginas [Lin and Ho, 2002; ching Wong and Fu, 2000].

Definição 2. *Cada página Web ρ_n é representada como um conjunto finito de blocos lógicos não sobrepostos $\rho_n = \{b_1, ..., b_k\}$, com k variando de acordo com a estrutura da página.*

[1]http://www.w3.org/DOM/



Uganda flooding washes away towns, lives

STORY HIGHLIGHTS

- Uganda declares a state of emergency as villages, crops submerged
- Nation one of the worst-hit in 17 African countries affected by floods

SOROTI, Uganda (AP) -- Uganda declared a state of emergency Thu
of the country as humanitarian workers tried to reach villages that ha

Uganda is
floods that
Africa in rec
been killed
thousands
people sinc
spokeswo

Officials confirm meteorite but question sickness claims

STORY HIGHLIGHTS

- NEW: Officials confirm a meteorite crashed in southern Peru
- Fiery ball seen falling from the sky and smashing into the Andean plain
- Hundreds say they have suffered headaches, nausea and breathing problems
- Experts say dust or gas released on impact probably to blame for sickness

LIMA, Peru (AP) -- A fiery meteorite crashed into southern Peru over the weekend, experts confirmed Wednesday.

But they were still puzzling over claims that it gave off fumes that sickened 200 people.

Witnesses told reporters that a fiery ball fell from the sky and smashed into the desolate Andean plain near the Bolivian border Saturday morning.

Figura 1: Páginas de notícias ρ_1 and ρ_2, extraídas do site da CNN.

Definição 3. *Um Web site $\mathcal{S}$ é representado aqui como um conjunto de páginas Web $\mathcal{S} = \{\rho_1, ..., \rho_m\}$, cada um delas composta por um conjunto de blocos.*

Adicionalmente, os blocos são agrupados em classes, tal como segue:

Definição 4. *Uma* classe de blocos *$\mathcal{C}$ é um conjunto de blocos que pertencem a páginas distintas de um mesmo Web site e que compartilham o mesmo rótulo, isto é,*

$$
\begin{aligned}
\mathcal{C} = \{b_1, b_2, ..., b_{n_\mathcal{C}}\} \\
b_1 &= (l_c, c_1) \\
b_2 &= (l_c, c_2) \\
&\vdots \\
b_{n_\mathcal{C}} &= (l_c, c_{n_\mathcal{C}})
\end{aligned}
$$

onde $n_\mathcal{C}$ é o número total de blocos da classe $\mathcal{C}$, l_c é o rótulo dos blocos de $\mathcal{C}$, b_i é o i^o bloco da classe $\mathcal{C}$, e c_i é sua porção de texto.

Para ilustrar, considere as duas páginas apresentadas na Figura 1. Perceba que cada página contém um *menu* em sua parte superior, que é composto por opções tais como *Home*, *World*, *US*, *Politics*, e que é referenciado como bloco menu. Como estes dois blocos estão em uma mesma posição em ambas as páginas, eles possuem o mesmo caminho na árvore DOM, e por conseguinte o mesmo rótulo. Como consequência, eles são parte de uma mesma classe de blocos. É possível notar também que blocos com o mesmo rótulo tendem a possuir a mesma função dentro de suas respectivas páginas.

Desta forma, as classes de blocos são usualmente compostas de blocos que pertencem a páginas distintas de um Web site e que executam a mesma função dentro de suas páginas. Vários exemplos de blocos que pertencem a mesma classes são mostrados na Figura 1, onde cada página possui um bloco responsável por apresentar o título, um bloco para apresentar o corpo de sua notícia, e um bloco para apresentar o resumo da notícia.

Identificação dos Blocos e Classes de Blocos

Em nossa tese, apresentamos um método de *segmentação automática* que, baseado nas definições apresentadas na seção anterior, produz como saída todos os blocos e classes de blocos implicitamente presentes em um Web site. Nosso algoritmo procura segmentar as páginas Web de acordo com a percepção dos usuários sobre como cada página deveria ser segmentada, e sobre como os blocos deveriam ser agrupados em classes.

A idéia de que uma página Web pode ser dividida em blocos de acordo com as funções de cada segmento já foi apresentada em trabalhos anteriores. Por exemplo, esta idéia motivou a criação do conceito de *Pagelets*, apresentada em Bar-Yossef and Rajagopalan [2002], e o conceito de *regiões coesas* ou *blocos*, apresentado por Cai et al. [2003]; Chakrabarti et al. [2008]; Kohlschütter and Nejdl [2008]. Entretanto, identificar a função de diferentes regiões de uma página Web é uma tarefa difícil de ser realizada em uma abordagem inteiramente automática.

Até onde sabemos, todos os trabalhos anteriores sobre segmentação propuseram métodos capazes de identificar os blocos de uma página por vez, sem considerar o conteúdo de outras páginas de um mesmo Web site. Entretanto, considerando que muitas páginas de um mesmo Web site compartilham uma estrutura e aparência comum, trabalhamos nesta tese com a hipótese de que podemos obter uma segmentação mais acurada se considerarmos todas as páginas de um mesmo Web site durante o processo de segmentação destas páginas em blocos. A partir desta idéia, apresentamos um método automático que segmenta as páginas de acordo com propriedades de todo o Web site, ao invés de considerar apenas a informação de uma única página por vez. Além de segmentar as páginas Web, nossa abordagem de segmentação permite-nos agrupar os blocos em classes de blocos, que são conjuntos de blocos que possuem uma mesma função em um conjunto de páginas, e que possuem uma importante função nos métodos de ranking baseados em blocos, que serão introduzidos na próxima sessão.

Além do algoritmo de segmentação automática, apresentamos também uma abordagem *semi-automática* para identificação dos blocos e classes de blocos. Esta abordagem é uma alternativa viável para obter uma alta qualidade de segmentação e é usada como uma base de referência para avaliar a qualidade dos segmentos encontrados pelo método de segmentação automática.

Funções *Block-Weight*

Em sistemas de recuperação de informação para a Web, o conteúdo das páginas é usualmente representado como um conjunto de termos derivados ou inferidos do texto destas páginas. Cada termo é pesado para indicar sua importância dentro de cada página, e a efetividade de qualquer sistema de busca está diretamente relacionada com a acurácia destes pesos.

Neste contexto, a estrutura das páginas Web pode ser usada para variar os pesos dos termos, visto que a ocorrência de um termo t em um bloco b pode ser levada em consideração no momento do cálculo do peso de t na página de b. Por exemplo, para propósitos de ranking, a ocorrência de um termo em um bloco da seção principal de uma página pode ser mais importante que uma ocorrência do mesmo termo no menu desta página. Quantificamos esta importância através de métricas denominadas *block-weight*, que calcula o *peso* da ocorrência de um termo em um bloco, tal como segue:

Definição 5. *A função block-weight $bw(t,b)$ é uma métrica quantitativa associada com o par termo-bloco $[t,b]$ que é usada para computar o peso global do termo t em relação à página que contém o bloco b.*

Para compor as funções *block-weight $bw(t,b)$*, são introduzidas duas medidas estatísticas básicas, o *inverse class frequency* e o *spread*, que serão explicadas a seguir.

Definição 6. *Dado uma classe de blocos $\mathcal{C} = \{b_1, ..., b_{n(\mathcal{C})}\}$, contendo $n(\mathcal{C})$ elementos, e um termo t que ocorre em ao menos um bloco de $\mathcal{C}$, o* Inverse Class Frequency *de um termo t em $\mathcal{C}$ é definido como*

$$ICF(t,\mathcal{C}) = \log \frac{n(\mathcal{C})}{n(t,\mathcal{C})},$$

onde $n(t,\mathcal{C})$ é o número de blocos de $\mathcal{C}$ onde t ocorre.

Perceba que o ICF é similar ao conceito de IDF, mas considera cada classe como uma "coleção de documentos" separada. Como o IDF, o ICF é uma forma de se estimar a quantidade de informação que carrega a ocorrência de um dado termo em uma dada classe de blocos. Ele também pode ser visto como uma medida da capacidade do termo de discriminar um bloco de outros blocos da mesma classe.

Definição 7. *O* Spread *de um termo t em uma página Web ρ, $Spread(t, \rho)$, é o número de blocos em ρ que contém t, isto é:*

$$Spread(t, \rho) = \sum_{b \in \rho} i_b, \ \ onde\ i_b = \begin{cases} 1 & se\ t \in b \\ 0 & caso\ contrário \end{cases}$$

A intuição por trás desta medida é que, dado um termo t de uma página Web ρ, quanto maior o número de blocos de ρ que contém o termo t, melhor o termo t representa o conteúdo de sua página. Por exemplo, se o termo "Firefox" aparece na maioria dos blocos de uma dada página, haverá uma grande chance de que esta página esteja relacionada com o famoso browser. O Spread pode ser visto como uma frequência estrutural de um termo, agindo como um versão estrutural do fator tf adotado nos modelos BM25 [Robertson et al., 1995] e vetorial [Sparck Jones, 1972, 1973, 1979].

A seguir são apresentadas diferentes estratégias para se computar os fatores bw usando as definições de ICF e *Spread*. Estas estratégias são agrupadas em três categorias: *métodos focados em classes*, que atribuem um único valor de bw para todos os termos de uma classe de blocos; *métodos focados em blocos*, que atribuem um único valor de bw para todos os termos de um bloco; e *métodos focados em termos*, que permitem que os valores de bw variem para termos distintos de um mesmo bloco ou classe.

Métodos Focados em Termos

Os métodos focados em termos usam os fatores ICF e *Spread* para calcular os pesos bw. Adicionalmente, propomos um terceiro método focado em termos que combina ambas as funções *Spread* e ICF. As fórmulas desses métodos são listadas abaixo:

$$\begin{aligned} bw_1(t, b) &= ICF(t, \mathcal{C}_b) \\ bw_2(t, b) &= Spread(t, \rho_b) \\ bw_3(t, b) &= ICF(t, \mathcal{C}_b), \times Spread(t, \rho_b) \end{aligned}$$

onde ρ_b e $\mathcal{C}_b$ são a página e a classe do bloco b, respectivamente.

Métodos Focados em Blocos

A idéia dos métodos focados em blocos é computar um valor médio de bw para todos os termos de um bloco, de forma a amenizar o impacto de anomalias que podem ocorrer quando consideramos apenas um único termo durante o cálculo dos fatores bw. Por exemplo, um termo t que ocorre no título de uma notícia é importante mesmo que ele não seja mencionado no corpo da notícia. No entanto, a função *Spread* deste termo poderia ser baixa, visto que o mesmo só ocorre no bloco título. Este efeito pode ser evitado usando uma média da função *Spread* para todos os termos do bloco.

Os métodos focados em termos introduzidos na seção anterior são usados para criar métodos focados em blocos equivalentes, tal como mostrado na Tabela 2.

$$
\begin{aligned}
bw_4(t,b) &= \begin{cases} \frac{\sum_{t' \in b} ICF(t',\mathcal{C}_b)}{|b|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \\
bw_5(t,b) &= \begin{cases} \frac{\sum_{t' \in b} Spread(t',\rho_b)}{|b|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \\
bw_6(t,b) &= \begin{cases} \frac{\sum_{t' \in b} Spread(t',\rho_b) \times ICF(t',\mathcal{C}_b)}{|b|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

onde t' é um termo e $|b|$ é o número de termos distintos do bloco b.

Métodos Focados em Classes

Também foi experimentado o uso da média de valores de bw de todos os termos em uma classe de blocos, ao invés da média dos valores encontrados para blocos individuais. Desta forma, neste terceiro tipo de método para se computar os fatores bw, é atribuído um único peso para todos os termos de uma classe de blocos. Os métodos focados em classes propostos neste trabalho são bw_7, bw_8 e bw_9, tal como mostrado na Tabela 3.

$$
\begin{aligned}
bw_7(t,b) &= \begin{cases} \frac{\sum_{t' \in v(\mathcal{C}_b)} ICF(t',\mathcal{C}_b)}{|v(\mathcal{C}_b)|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \\
bw_8(t,b) &= \begin{cases} \frac{\sum_{b' \in \mathcal{C}_b} \frac{\sum_{t' \in b'} Spread(t',\rho_{b'})}{|b'|}}{|\mathcal{C}_b|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \\
bw_9(t,b) &= bw_7(t,b) \times bw_8(t,b)
\end{aligned}
$$

onde b' é um bloco, $|\mathcal{C}_b|$ é a quantidade de blocos da classe $\mathcal{C}_b$, $v(\mathcal{C}_b)$ é o conjunto de termos distintos que ocorrem em todos os blocos da classe $\mathcal{C}_b$ e $|v(\mathcal{C}_b)|$ é a sua norma.

Ordenação de Respostas Usando as Funções Block-Weight

Para incorporar a função $bw(t, b)$ em métodos de ordenação de documentos, usamos uma idéia previamente proposta por Robertson et al. [2004], que apresenta uma adaptação do modelo de recuperação de informação BM25 para lidar com documentos com múltiplos campos. O BM25 usa a função $tf(t, \rho)$ (frequência do termo) como um dos fatores para o cálculo do quão bem um dado termo t descreve ou representa o conteúdo do documento (ou página) ρ (caracterização intra-documento do termo) [Baeza-Yates and Ribeiro-Neto, 1999]. O tf de um termo t em um documento ρ é computado como uma função do número de ocorrências de t em ρ, usualmente atribuindo igual importância para cada ocorrência. Baseando-nos na proposta de Robertson et al. [2004], argumentamos que o cálculo do tf de um termo t em um documento ρ deveria levar em consideração as posições onde t ocorre em ρ, isto é, os blocos de ρ onde ele ocorre, pesando cada ocorrência de t em cada bloco $b \in \rho$ por $bw(t, b)$. Baseados nesta abordagem, propomos uma alternativa à forma tradicional de cálculo do fator tf, $tf'(t, \rho)$, que pode ser calculado por:

$$tf'(t, \rho) = \sum_{b \in \rho} tf(t, b) \times bw(t, b) \tag{1}$$

onde $tf(t, b)$ é a frequência do termo dentro do bloco b.

Adicionalmente, são propostas simples variantes de dois outros fatores da fórmula do BM25. Esses fatores são o parâmetro k_1, e o número de documentos onde um dado termo t ocorre, denominado n_t. Conforme mostraremos em detalhes em nossa tese, essas variantes podem ser acoplados diretamente à fórmula do modelo BM25 original.

Resultados Experimentais

Para confirmar a eficácia de nossos métodos, realizamos experimentos de busca em 4 coleções de páginas Web denominadas IG, CNN, CNET, e BLOGs, e avaliamos o impacto de nossos métodos na qualidade dos resultados.

Nossos métodos baseados em blocos foram comparados com 3 diferentes métodos de recuperação de informação da literatura. O primeiro método usa o BM25 com nenhuma informação de bloco, isto é, considera que as coleções de Web sites são formadas por documentos planos tradicionais. Para o segundo método, removemos manualmente todos os templates das páginas das coleções de Web sites, e então aplicamos a função de ordenação BM25 no conteúdo restante. Finalmente, o terceiro método foi proposto por Cai et al. [2004b], e é baseado na junção de dois tipos de ordenação de respostas: 1) uma *ordenação baseada em documentos*, que usa o BM25 na coleção de documentos; e 2) uma *ordenação baseada em blocos*, que é a ordenação dada pelo BM25 para o melhor bloco de cada documento (ou página).

No entanto, concluímos através de experimentos que os três métodos citados acima apresentam resultados similares e que, no contexto de nossas quatro coleções de documentos, são basicamente equivalentes. Desta forma, neste resumo nos limitaremos a mostrar os resultados de comparação entre nossos métodos baseados em blocos, e o método BM25 tradicional.

Para comparar os resultados, adotamos duas métricas de avaliação. A primeira métrica é chamada *mean average precision* (MAP), e a segunda é chamada *precision at 10* (P@10), que mede a quantidade de documentos relevantes nos 10 primeiros documentos retornados pelo método a ser avaliado [Baeza-Yates and Ribeiro-Neto, 1999].

A Tabela 4 apresenta os resultados obtidos quando modificamos o modelo BM25 para o uso das funções bw_3, bw_6 e bw_9, e aplicamos esse modelo modificado para nossas 4 coleções de documentos segmentadas pelos métodos de segmentação automática e semi-automática. A primeira observação importante é que os três fatores bw considerados obtiveram resultados superiores à abordagem tradicional, em ambos os métodos de segmentação. Nossos métodos de ranking baseados em blocos obtiveram melhorias na maioria dos casos, o que sugere que os fatores bw introduziram informação complementar sobre a importância dos termos.

Foi observado que a performance do método bw_3 é em geral ligeiramente pior quando comparado com os fatores baseados em blocos e em classes, para ambas as coleções. Quando comparamos os fatores baseados em blocos com os fatores baseados em classes, observamos que eles obtiveram resultados similares para ambas abordagens de segmentação.

Esses resultados sugerem que o modelo BM25 adaptado ao uso de dois de nossos bw fatores, bw_6 and bw_9, foram consistentemente superior que o BM25 tradicional, enquanto os demais métodos de ranking não proporcionaram melhorias evidentes. Enquanto nossos experimentos não permitiram concluir qual método é melhor (bw_6 ou bw_9), eles mostraram que ambas as estratégias de cálculo dos fatores bw introduzem informação complementar sobre a ocorrência dos termos, e que melhorias significantes podem ser obtidas quando consideramos a estrutura dos documentos em modelos de recuperação de informação.

		IG		CNN	
	método	P@10	MAP	P@10	MAP
	BM25	0.594	0.621	0.612	0.691
semi-	bw_9	**0.667**(12.3%)	**0.749**(20.6%)	**0.642**(04.9%)	**0.786**(13.7%)
automático	bw_6	**0.653**(09.9%)	**0.730**(17.5%)	**0.648**(05.9%)	**0.800**(15.7%)
	bw_3	**0.655**(10.3%)	**0.698**(12.4%)	**0.638**(04.2%)	**0.769**(10.0%)
automático	bw_9	**0.659**(10.9%)	**0.733**(18.0%)	**0.630**(02.9%)	**0.779**(12.7%)
	bw_6	**0.655**(10.2%)	**0.715**(15.1%)	**0.636**(03.9%)	**0.790**(14.3%)
	bw_3	0.638 (07.4%)	**0.680**(09.5%)	**0.628**(02.6%)	**0.759**(09.8%)
		BLOGs		**CNET**	
	método	P@10	MAP	P@10	MAP
	BM25	0.584	0.644	0.476	0.458
semi-	bw_9	0.604 (03.4%)	**0.678**(05.3%)	**0.552**(16.0%)	**0.498**(08.7%)
automático	bw_6	0.610 (04.4%)	**0.675**(04.8%)	**0.558**(17.2%)	**0.513**(12.0%)
	bw_3	0.588 (00.7%)	0.628 (-2.5%)	**0.545**(14.5%)	0.482 (05.2%)
automático	bw_9	0.602 (03.4%)	**0.677**(05.1%)	**0.512**(07.5%)	0.470 (02.6%)
	bw_6	0.604 (04.4%)	**0.671**(04.2%)	**0.527**(10.8%)	0.445 (-2.8%)
	bw_3	0.592 (00.7%)	0.632 (-1.8%)	**0.514**(08.0%)	0.459 (00.2%)

Tabela 4: Resultados obtidos quando comparamos três alternativas para se computar os fatores bw (bw_9, bw_6, e bw_3) com o modelo BM25 tradicional. As porcentagens indicam o ganho médio em precisão e os resultados em negrito são estatisticamente significantes.

List of Figures

1.1 A news page and its blocks 3

2.1 A DOM tree example (lower level tags are omitted) 12
2.2 The label of a block in a Web page is represented as the root-to-block path in the page DOM tree. 13
2.3 The DOM tree of a very simple Web page, and a set of possible blocks found in the page. 13
2.4 News pages ρ_1 and ρ_2, extracted from CNN Web site. 14

3.1 Traditional segmentation purpose: given a single Web page as input, to divide the page in a set of logical blocks. 19
3.2 An example of HTML code and the pre-processed version of its DOM tree. 21
3.3 Example of tag with nested content. 23
3.4 Recurrent structures: the regions pointed by the arrows contain sets of items (be links or books) disposed in a regular or recurrent way. 24
3.5 Recurrent substructures of the DOM trees are not represented in our DOM tree representation. 24
3.6 An example of a SOM_{tree} construction. The *pageList* values are depicted below each node of this structure. 26
3.7 (a) A SOM_{tree} segment example and (b) its pruned version, considering a α threshold equal to 6. Notice that, instead of presenting the list of documents in each leaf element node, this figure presents only the number of documents in each list. 29
3.8 (a) A SOM_{tree} segment example and (b) its pruned version, considering a β threshold equal to 8. 31
3.9 A SOM_{tree} segment example, where the node `p` remains with the counter value smaller than the threshold β after the refining process. 31

3.10 After the refining process each leaf node of the SOM_{tree} refers to a block class of the Web Site. 32
3.11 Clustering the pages of a Web site according to their internal structure. . . 33
3.12 Number of blocks in each block classes found in four document collection. . 41
3.13 Number of blocks in each block classes found on in four document collection. 43
3.14 A typical page present on the Engadget blog, from *BLOG* collection. . . . 44
3.15 A typical page present on the reviews Web site, from *CNET* collection. . . 46
3.16 Adjusted RAND Index graphs found for IG, CNN, BLOGs and CNET collections. 47
3.17 Average Mutual Information graphs found for IG, CNN, BLOGs and CNET collections. 48

4.1 Document that would have high similarity with queries "Weather forecast" and "Olympic torch" when applying the method presented by Cai et al. [2004b] 55
4.2 Spatial and content features can be used to differentiate the importance of blocks 58
4.3 RIPB's informative block cluster recognition process 60
4.4 The process of block clustering of RIPB 61

5.1 A news page and its blocks. 66
5.2 A page where the term Firefox is spread in its blocks. 68

6.1 Kendall tau distance between ranking results obtained when computing *bw* factors based on the whole collection and ranking of results obtained when computing *bw* factors based on the blocks available on collections $\mathcal{B}_2, \mathcal{B}_4, \mathcal{B}_6, ..., \mathcal{B}_{200}$. (a) The graph obtained from IG collection; and (b) the graph obtained from CNN collection. 87

List of Tables

3.1 Notation for the contingency table for comparing partitions U and V. . . . 35
3.2 The set of Web sites crawled for each document collection. 38
3.3 The number of blocks and block classes found by the semi-automatic approach in all Web site collections. Table also shows the number of blocks found in small classes, i.e., block classes with less than β elements (Blocks SC). 40
3.4 The number of blocks and block classes found by our automatic segmentation approach in all Web sites of the four collections. Table also shows the number of blocks found in small classes, i.e., block classes with less than β elements (Blocks SC). 42

5.1 Examples of bw factors assigned to blocks found in the Web page depicted in Figure 3.14, extracted from the BLOGs collection. 72
5.2 Examples of bw factors assigned to blocks found in the Web page depicted in Figure 3.15, extracted from the CNET collection. 73

6.1 Results obtained by BM25 and BM25BB baselines. 81
6.2 Results obtained by BM25 and BM25NT (BM25 removing templates). . . 82
6.3 Results obtained when using the three alternative bw factors and using no bw factor (BM25). The percentages indicates the average gains in precision figures and results in bold are statically significant. 84
6.4 Comparison between results obtained when using the automatic and the semi-automatic segmentation methods. 85
6.5 Percentual of deleted index entries performed by bw_3, bw_6 and bw_9 methods. 86

Contents

Agradecimentos **xi**

Resumo **xvii**

Abstract **xix**

Resumo Estendido **xxi**

Representado Web Sites como Blocos xxii

Identificação dos Blocos e Classes de Blocos xxiv

Funções *Block-Weight* xxv

Métodos Focados em Termos xxvi

Métodos Focados em Blocos xxvii

Métodos Focados em Classes xxvii

Ordenação de Respostas Usando as Funções Block-Weight xxviii

Resultados Experimentais xxviii

List of Figures **xxxi**

List of Tables **xxxiii**

1 Introduction **1**

1.1 Information Retrieval 1

1.2 Using Structural Information to Improve Search in Web Collections . . 2

1.3 Related Work 4

1.4 Thesis Contributions 7

1.5 Publications 8

1.6 Thesis Outline 9

2 Representing Web Sites as Block Components **11**

2.1 The DOM Tree Representation 11

2.2 Representing Web Sites Contents 12

3 Identifying Blocks and Block Classes 15

3.1 Traditional Web Page Segmentation 16

3.1.1 VIPS - Vision-based Page Segmentation Algorithm 16

3.1.2 A Densitometric Approach to Web Page Segmentation 18

3.1.3 Discussion 19

3.2 A New Approach for Web Page Segmentation 20

3.2.1 Phase 1: Pre-processing Web Pages 20

3.2.2 Phase 2: Constructing the SOM_{tree} 25

3.2.3 Phase 3: Refining the SOM_{tree} 27

3.3 A Semi-automatic Approach for Identifying Blocks and Block Classes . 32

3.3.1 Evaluating Segmentation Algorithms 34

3.4 Experimental Results 37

3.4.1 Segmenting the Collections 38

4 Information Retrieval and Web page Segmentation 49

4.0.2 Term Weighting 49

4.0.3 The Vector Space Model 50

4.0.4 The BM25 Model 51

4.0.5 Discussion 52

4.1 Using Block Information to Improve Ranking Results 53

4.1.1 Block-based Web Search 53

4.1.2 Block-level Link Analysis 54

4.1.3 Improving Pseudo-Relevance Feedback Using Block Information 55

4.1.4 Learning Block Importance Models for Web Pages 57

4.1.5 Template Removal 59

4.1.6 Information Extraction 60

5 Block-Weight Functions 65

5.1 Inverse Class Frequency and Spread 66

5.2 Term-level Strategies 68

5.3 Block-Level Strategies 69

5.4 Class-Level Strategies 69

5.5 Dealing with Block Classes with Few Elements 70

5.6 Block-Weight Values and Their Meaning 71

6 Ranking Using Block-Weight Functions 75

6.1 BM25 Extension for Using *bw* Information 76
6.2 Experimental Results 77
6.2.1 Experimental Setup 77
6.2.2 Determining the Threshold β 78
6.2.3 Baselines 79
6.2.4 Ranking Results 81
6.2.5 Index Pruning Using *bw* Factors 85

7 Conclusions and Further Work 89

7.1 Thesis Summary 89
7.2 Further Work 91

Bibliography 93

Chapter 1

Introduction

Most information retrieval techniques for Web considers Web pages as indivisible and monolithic units, i.e, they treat different portions in a Web page equally. However, recent works have demonstrated that Web pages can be sub-divided into different segments (or blocks), usually with different types of contents and purposes. This work concerns the use of such block structure to improve the retrieval effectiveness of information retrieval systems for Web collections. In this chapter, we develop and discuss the goals and contributions of our thesis.

1.1 Information Retrieval

Information Retrieval (IR) focuses on finding information from some source of material (usually documents) stored digitally. An information retrieval process begins when a user enters a query into an IR system. Queries are formal statements of information needs addressed to an IR system by the user. Given a user's query, the goal of an IR system is to retrieve the documents that are relevant to the user, while retrieving as few non-relevant document as possible. Usually, this task consists of retrieving a set of documents and ranking them according to the likeliness that they will satisfy the user's query.

The most popular models for ranking documents of a collection (including Web document collections) are the vector space model [Salton and Lesk, 1968; Salton, 1971], the probabilistic relevance models [Maron and Kuhns, 1960; Robertson and Jones, 1988; Robertson and Walker, 1994], and the statistical language models [Ponte and Croft, 1998; Berger and Lafferty, 1999; Lafferty and Zhai, 2001]. Such models use different methods of representing queries and documents, different schemes of term weighting, and different formulas for computing the ranking.

Designing effective schemes for term weighting is a critical step in an information retrieval system. However, finding good term weighting schemes continues to pose a significant challenge. In this work, we propose new term weighting schemas for Web document collections that leads to improve ranking by considering the structural content of Web pages.

The best known term weighting schemes use weights that are a function of the number of times the index term occurs in a document and the number of documents in which the index term occurs. Such term weighting strategies are called $tf \times idf$ (term frequency times inverse document frequency) schemes [Salton and McGill, 1986; Baeza-Yates and Ribeiro-Neto, 1999]. A modern variation of these strategies is the BM25 weighting scheme used by the Okapi system [Robertson and Walker, 1994].

Most of existing term weighting schemes (including $tf \times idf$ and BM25 weighting schemes), to this date, consider documents as indivisible units and treat their different portions equally. Although this approach of term weighting is appropriate for retrieval on plain document collections, it might be too simplistic on Web document collections. The Web pages are commonly composed by different segments (or blocks), usually with different types of contents and purposes (see Figure 1.1). Thus, in this work we argue that weighting schemes for information retrieval models for Web document collections should take into account the block structure of Web pages, in order to improve their retrieval effectiveness.

1.2 Using Structural Information to Improve Search in Web Collections

Unlike plain text documents, Web pages are commonly composed of distinct segments such as service channels, decoration skins, navigation bars, copyright and privacy announcements, all of which might have contents quite distinct from that in the *main* segment of the Web pages. These different segments, commonly referred to in the literature as *blocks*, can be automatically identified in Web pages [Bar-Yossef and Rajagopalan, 2002; Cai et al., 2003; Hattori et al., 2007; Lin and Ho, 2002] and can be used to improve information retrieval tasks such as ranking [Ahnizeret et al., 2004; Cai et al., 2004b; Fernandes et al., 2007; Song et al., 2004a], Web link analysis [Cai et al., 2004a], and Web mining [Cao et al., 2008; Kang and Choi, 2007; Li et al., 2007]. To improve ranking, for instance, block information can be used to vary term weights according to their occurrences inside blocks (instead of inside pages), which allows quantifying differently the importance of the distinct occurrences of a term within a same Web

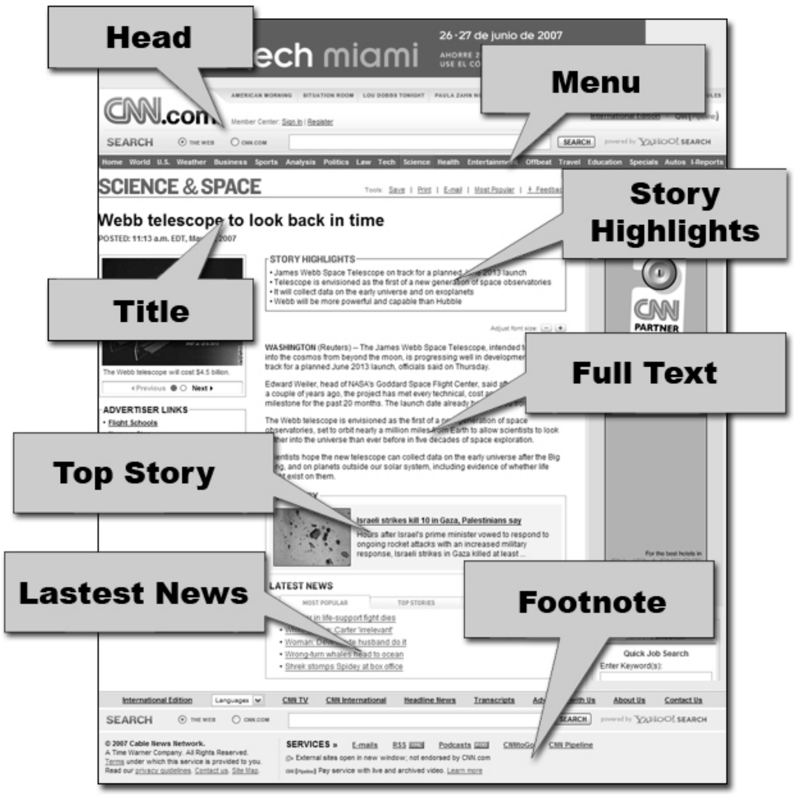


Figure 1.1. A news page and its blocks

page. For instance, the occurrence of a term in the *title* of the Web page depicted on Figure 1.1 is expected to be more important for ranking purposes than an occurrence of that same term in a *menu* of that page.

In this thesis, we investigate how to improve retrieval results by exploring the block structure of Web pages. For that, we propose (i) a new approach for representing the content of Web sites in information retrieval systems that takes into account the internal structure of their Web pages and the relationship of the structural components found on the pages; (ii) a method to automatically identify the internal structure of the Web pages, according to the approach of representing the Web sites contents proposed in this thesis; and (iii) a set of distinct block-based term weighting schemes divided into three types: *term-level weighting*, *block-level weighting*, and *class-level weighting* methods. These schemes have the advantage of not requiring a learning process nor any type of manual intervention to compute the ranking, as required by previous works [Ahnizeret et al., 2004; Song et al., 2004a].

Using four Web site collections, we have run experiments to compare our block-based term weighting schemes with (i) two other block-based retrieval models proposed in the literature, one of them based on templates removal and the other one based on a block ranking function and (ii) with a traditional ranking function that does not use block information. The results indicate that our block-based weighting schemes lead to improved results with regard to all baselines. For IG, one of the Web test collections we used, an improvement from 0.62 to 0.75 in Mean Average Precision (MAP) scores was achieved relatively to a ranking that does not take blocks into account. For CNN, another Web site collection we used, the improvements in MAP figures were from 0.69 to 0.80.

Further, since our block-based weighting schemes are able to quantify how important is a given term occurrence for ranking purposes, they are also able to detect the presence of terms in the pages that has no effect on retrieval results. By identifying and removing these entries in advance, our block-based methods allow reducing the size of the inverted index used for retrieval by over 27%, decreasing indexing storage requirements and the cost of processing queries.

1.3 Related Work

Before computing block-based term weight values, it is necessary to identify blocks occurring in a Web page collection. Identifying the logical blocks that compose the Web pages of a site is by itself a research problem known as *Web page segmentation*, which counts with some solutions proposed in the literature [Kohlschütter and Nejdl, 2008; Hattori et al., 2007; Cai et al., 2003; Bar-Yossef and Rajagopalan, 2002; Chakrabarti et al., 2001] . One of most popular solutions in the literature, proposed by Cai et al. [2003], aims at segmenting a page by simulating visual perceptions of the users about the page, and is referred to as the Vision-based Page Segmentation algorithm (VIPS). The VIPS algorithm works by identifying visual cues in Web pages that can be used to detect blocks such as lines, blank areas, colors, pictures, fonts types, etc. The VIPS algorithm requires an extra human effort, since it is necessary to manually set parameters to each page to be segmented.

In Hattori et al. [2007] the segmentation method proposed is based on calculating the distance between content elements within the HTML tag hierarchy, i.e., the number and the depth of HTML tags in Web. Chakrabarti et al. [2008] also consider the problem of segmenting Web pages into visually and semantically cohesive pieces. This work is based on Kleinberg and Tardos [2002], which propose approximation algorithms for clustering problems based on weighted graphs. For the segmentation problem proposed by Chakrabarti et al. [2008], the weights of edges of the graph capture the likelihoods of two regions of the Web page be placed together or apart in the segmentation, and the clusters are the blocks occurring in the pages.

Kohlschütter and Nejdl [2008] examined the problem of Web page segmentation from a textual perspective. The key assumption in their work is that the number of tokens in a text fragment (or more precisely, its token density) is a valuable feature for segmentation decisions. Based on this assumption, the authors presented a Block Fusion algorithm for Web page segmentation using text density metrics. The Section 3.1.2 presents the Block Fusion algorithm in details.

As far as we know, all previous segmentation methods of the literature segment each Web page based on its content only, not considering the content of other pages present in the same Web site. However, since many pages belonging to a same Web site share a common structure and look and feel, we hypothesize that we can achieve a more accurate segmentation by taking all pages of a same Web site into account when dividing them into blocks. Based on this idea, in this thesis we present a segmentation approach which segment pages according to properties of the whole Web site, instead of just information from a single page. Besides segmenting the Web pages, our segmentation approach enables us to cluster blocks into block classes, that are set of blocks that have a common role on a group of pages, and that play an important function in the structure-based term weighting schemes proposed in this thesis.

In Cai et al. [2004b], the authors study the problem of improving search results in the presence of segmented pages. They discuss a ranking strategy that considers blocks as passages and uses previously proposed passage level ranking strategies [Callan, 1994] to generate the results. Their work suggests two important conclusions: (i) block segmentation produces results that are superior to those produced by standard passage segmentation algorithm based on paragraphs and fixed size windows, and (ii) the best ranking was obtained by combining a similarity score based on whole page with a similarity score based on blocks. Given this compelling result, we adopted this ranking strategy (which is better detailed on Section 4.1.1) as one of the baselines in our experiments.

In Song et al. [2004a] block division is used to compute an importance rank for the blocks found in a page, using a learning approach. Spatial features such as the position and size of blocks and content features such as the number of images and links on each block are extracted from each block. Then, two learning algorithms, neural network and SVM, are used to rank the blocks according to their importance. A very similar learning approach is presented in Liu et al. [2006], with the difference that it considers the influence of the variation of features across different pages. The methods we here propose assign an importance weight to each block, instead of just ranking them, and have the advantage of not requiring a training phase.

Trotman [2005] presented a learning approach to estimate block importance values in collections of documents with the same structure. In this work, each block of this shared structure is represented as a chromosome in a GA learning simulation. Selective pressure is then applied to maximize mean average precision. The authors conducted experiments with the TREC [Harman, 1993] Wall Street Journal (WSJ) collection, and the results demonstrate an about 5% improvement in mean average precision. The methods presented here can be applied in any Web site collection, and not only on collections of pages with similar structure.

Experiments presented by us in Ahnizeret et al. [2004] indicate that manual assignment of block weights may result in an improvement on the quality of search results, since these weights can be used to derive effective block-based term weighting methods. However, the task of manually assigning such weights requires specialized knowledge about the ranking function adopted in the search system, as well as a complex and expensive human effort to evaluate the relative importance of all regions found in the target Web site. For Web collections containing a large number of Web pages spread across multiple Web sites, which is a common scenario, this manual assignment of block weights may be unfeasible in practice.

To avoid some of these practical problems, we proposed in Fernandes et al. [2007] a method for automatically computing block weight factors, as well as a block-based ranking method based on these weights. These methods use statistical information available in the Web page collection to compute block weights and do no require a learning process nor intensive manual intervention, as in Ahnizeret et al. [2004]; Song et al. [2004a]. However, our method in Fernandes et al. [2007] requires the Web pages to be clustered into *page classes*, i.e., sets of pages that have a common block structure, which does require a limited amount of human effort. Even if limited, the need of human intervention might preclude the application of our technique to very large Web page collections.

In this thesis, we present an improvement of the method presented by Fernandes et al. [2007] that removes the need to cluster the Web pages. Further, we introduce eight variants of the block-based term weighting scheme proposed in Fernandes et al. [2007] and run a more extensive set of experiments. Our results indicate that our enhanced block-weighting methods lead to superior results when compared to (i) a method that uses no block information, (ii) a method based on templates removal, and (iii) another block-based ranking method in the literature. Further, we present a new completely automatic Web page segmentation algorithm that was designed to fulfill the requirements of the block-weighting methods presented in this work.

Several authors have proposed methods for using the structure of Web pages for identifying noisy content. For instance, Bar-Yossef and Rajagopalan [2002] proposed a mechanism for segmenting Web pages into blocks, termed as Pagelets, and classifying them into useful or noisy content. In their work, the authors considered that any information present on a page template is noisy and should be disregarded. Template removal has also proposed in other works as Vieira et al. [2006]; Yi et al. [2003].

The experimental results in these works [Bar-Yossef and Rajagopalan, 2002; Vieira et al., 2006; Yi et al., 2003] suggest that template removal methods can be used to improve results, an observation that is not confirmed by our experimental results in this work. Since the removal of templates can be considered as an structural information, we included a search system that does not index template contents as one of the baselines in our experiments.

1.4 Thesis Contributions

This thesis focuses in the use of structural organization of Web pages to improve the retrieval effectiveness of information retrieval systems for Web sites. Thus, this work aims to provide answers to the following research questions:

- How to identify the structural organization of pages in a Web site?
- How to use such a structural organization to improve ranking results when searching for information on Web collections?

To answer the first question, we propose a new approach for representing the contents of the pages in a Web site, and a new method for detecting internal structure of Web pages. To answer the second question, we propose new block-based schemes for computing the weight of the index terms in each Web page of a collection. These weights are computed by analyzing the influence that each occurrence of a term in a page must have about the overall weight of the term in that page.

At last, the major contributions of this thesis are, therefore:

- A new approach for representing Web sites in information retrieval systems based on the internal structure of their Web pages. This new model of representing the pages of a Web site captures important informations about how the content is organized within the site, and is a key piece of our work (Chapter 2).

- A new automatic segmentation algorithm which, based on our approach of representing Web sites contents, produces as output all the blocks and block classes implicitly present in a Web site (Chapter 3).

- New block-based weighting schemes designed to search on Web collections, that aim at to improve retrieval tasks by estimating terms weights according to the location of their occurrences within the pages (Chapter 5).

- An adaptation of the BM25 ranking formula based on the work of Robertson et al. [2004], that adjusts the BM25 to the use of our block-based weighting schemes (Chapter 6).

1.5 Publications

The following is a list of publications for which the author has been either a primary author or a co-author, and which are related to, or have influenced, the work in this thesis.

1. K. Ahnizeret, D. Fernandes, J.M.B. Cavalcanti, E.S. de Moura, and A. Silva "*Information retrieval aware web site modelling and generation*", Proceedings of the 23th Internacional Conference on Conceptual Modeling, Shangai, China, 2004.

   This paper proposes an approach to Web site modelling and generation of Enterprise search engines, combining application modelling and information retrieval techniques. Our assumption is that giving search engines access to the information provided by conceptual representations (or block segmentation) of the Web site improves their performance and accuracy. We demonstrate this proposal by describing a Web site modelling language that represent both traditional modelling features and information retrieval aspects (such as block importance), as well as presenting experiments to evaluate the resulting intrasite search engine generated by our method.

2. D. Fernandes, E.S. de Moura, B. Ribeiro-Neto, A.S. da Silva, Marcos André Gonçalves. "*Computing Block Importance for Searching on Web Sites*", Proceedings of the 16th International Conference on Information and Knowledge Management, Lisbon, Portugal, 2007. ACM Press.

   In this paper we consider the problem of using the *block structure* of a Web page to improve ranking results when searching for information on Web sites.

Given the block structure of the Web pages as input, we propose a method for computing the importance of each block (in the form of block weights) in a Web collection. The experiments presented in this paper show that our method may allow a significant improvement in the quality of search results.

3. Figueiredo Flavio , Edleno Moura, Jussara Almeida, Marcos André Gonçalves, Belém Fabiano, Henrique Pinto, David Fernandes, Marco Cristo. "*Evidence of Quality of Textual Features on the Web 2.0*", Proceedings of the 18th International ACM Conference on Information and Knowledge Management, Hong Kong, 2009. ACM Press.

   This work aims at assessing the relative usefulness for IR tasks of different textual features available on the Web 2.0. Four features (title, tags, description and comments) were analyzed in four different popular applications (CiteULike, Last.FM, Yahoo! Video, and Youtube). Each feature was evaluated according its usage, amount of information, content diversity, descriptive and discriminative power.

4. Flavio Figueiredo, Fabiano Belém, Henrique Pinto, Jussara Almeida, Marcos André Gonçalves, David Fernandes, Virgilio Almeida. "*Evidências de Qualidade de Atributos Textuais na Web 2.0*" Proceedings of the Simpósio Brasileiro de Sistemas Multimédia e Web (Webmedia), Fortaleza, Brazil, 2009.

   Another investigation on the quality of the different textual features available on the Web 2.0. The three aspects evaluated are: usage, descriptive and discriminative power of each feature.

1.6 Thesis Outline

The text of this thesis is organized as follows. Chapter 2 introduces a new approach for representing Web Sites contents, based on the internal structure of their pages. Chapter 3 presents the automatic segmentation algorithm which, based on the representing definitions proposed on Chapter 2, produce as output all the blocks and classes of blocks implicitly present in a Web site. Chapter 4 gives some background on information retrieval models, and describe some techniques of the literature of using block structure in retrieval tasks. Chapter 5 presents our block-based weighting functions. Chapter 6 discusses how we use these block-weight functions to compute a raking, the evaluation metrics we used during experimentation, and discusses the performance evaluation we conducted. Finally, Chapter 7 presents some conclusions and directions for future work.

Chapter 2

Representing Web Sites as Block Components

When users observe a page through a Web browser, they can distinguish different parts in the page such as navigation bars, decoration skins, interaction forms, copyrights, main section, each one with a particular role within the pages. This suggests that Web pages can be divided into portions, that are frequently referred to as *blocks*. Identifying this logical elements of Web pages is a research problem known as *Web page segmentation*.

This Chapter introduces a new approach for representing Web Sites contents based on the block structure of their pages. This approach is able to capture important information about how the content is organized within the site, and is a key piece of our work. Further, this Chapter presents some basic definitions which are useful for better characterizing the problem addressed here and for supporting the explanation of our work.

2.1 The DOM Tree Representation

Each Web page contains a hierarchical representation called DOM (an acronym to Document Object Model) tree, that defines the logical structure and the layout of the page. Figure 2.1 depicts a fragment of HTML code and its corresponding DOM tree. The DOM tree of an HTML document contains objects called *element nodes* or *tags* (such as `<body>` and `<p>`). Each tag has a *name property* that specifies the function of the object within the page, and a *list of attributes* that defines the behavior of the function performed by the tag. To exemplify, the name and the list of attributes of the most-left object of the DOM tree depicted on Figure 2.1 are `div` and `{align=right,` `class=c1}`, respectively. As depicted in Figure 2.1, each tag can have a textual content assigned to it. Textual content do not constitute new nodes of DOM structure.

```
<html>
  <body bgcolor=white>
    <div align=right class=c1>
      Some text 1
    </div>
    <img src=image.gif>
    <div align=left class=c2>
      <h1>Some text 2</h1>
      <p>Some text 3</p>
    </div>
  </body>
</html>
```

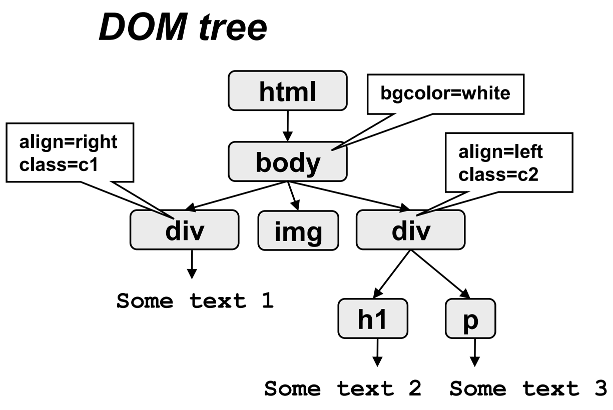


Figure 2.1. A DOM tree example (lower level tags are omitted)

2.2 Representing Web Sites Contents

Unlike plain text documents, Web pages usually encompass multiple regions such as navigation bars, decoration stuffs, copyrights, advertisements and contact information. Each of these produce a visually cohesive and continuous region on the browser window, play a particular role within the page, and are generically referred to as *blocks*. To characterize the division of Web pages into blocks in formal terms, we introduce the following definitions.

Definition 1. *A* block *b is a self-contained logical region within a Web page that (i) is not nested within any other block and (ii) is represented by a pair (l, c), where l is the label of the block, represented as a string, and c is the portion of text of the block. That is,*

$$b = (l, c)$$

It follows that a Web page might be divided into a set of non-overlapping blocks, subject to the interpretation of a human analyst or to the behavior of the segmentation method adopted. To represent the label l we use the root-to-block path in the Web page DOM tree (see Figure 2.2).

Definition 2. *We represent each Web page ρ_n as a finite set of (non-overlapping) logical blocks $\rho_n = \{b_1, ..., b_k\}$, with k varying according to the page structure.*

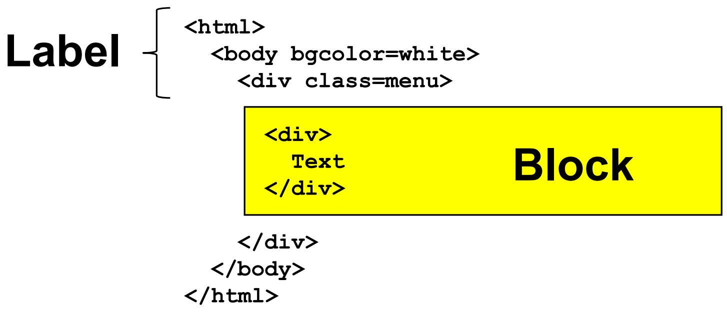


Figure 2.2. The label of a block in a Web page is represented as the root-to-block path in the page DOM tree.

The definition of blocks we adopt here allows several types of partitions for a given Web page, and even the DOM tree of the page may be considered as a partition itself, although with a very high level of granularity. Since our block definition admit that any arbitrary subtree of a DOM structure be assigned as a block, the decision of how to divide a page into blocks should be done by the segmentation algorithm. In this work, we adopt the criterium that the division of a Web page into a set of non-overlapping blocks must be performed according to the perception of users about the best logical division of the page.

Notice that, according our block definition, if a node of a DOM tree belongs to a particular block, then all descendants of that node also belong to that particular block. Entire or partial overlap between blocks in the DOM tree is not allowed. Figure 2.3 depicts the DOM tree of a very simple Web page, and a set of possible blocks of the page.

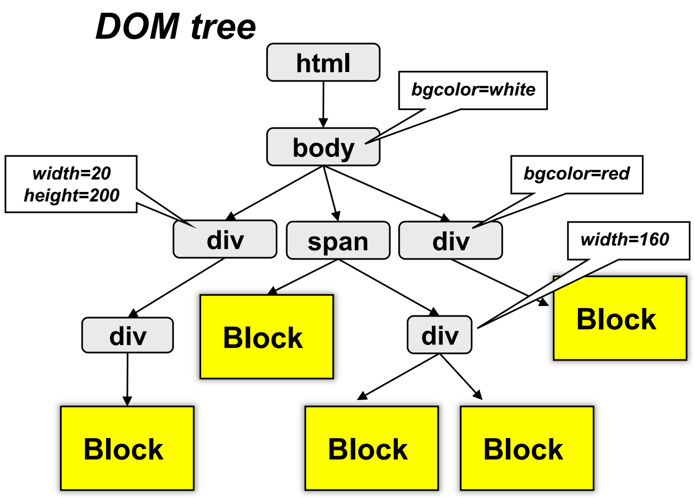


Figure 2.3. The DOM tree of a very simple Web page, and a set of possible blocks found in the page.

Definition 3. *A Web site $\mathcal{S}$ is represented here as a set of Web pages $\mathcal{S} = \{\rho_1, ..., \rho_m\}$, each of them composed of a set of blocks.*

Further, blocks of distinct pages can be clustered into classes as follows:

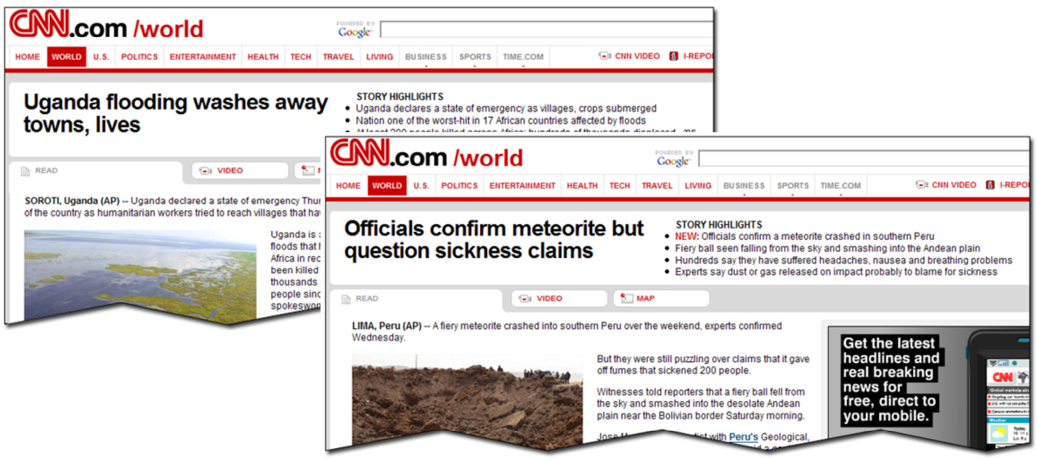
Definition 4. *A* block class *$\mathcal{C}$ is a set of blocks that belong to distinct pages of a given Web site and that share a same label, i.e.,*

$$\mathcal{C} = \{b_1, b_2, ..., b_{n_\mathcal{C}}\}$$

$$\begin{aligned} b_1 &= (l_c, c_1) \\ b_2 &= (l_c, c_2) \\ &\vdots \\ b_{n_\mathcal{C}} &= (l_c, c_{n_\mathcal{C}}) \end{aligned}$$

where $n_\mathcal{C}$ is the total number of blocks in class $\mathcal{C}$, l_c is the common label, b_i is the i_{th} block in class $\mathcal{C}$, and c_i is its corresponding portion of text.

To illustrate, consider the two Web pages depicted on Figure 2.4. Notice that each page contains a *menu* at the top composed of options such as Home, World, US, Politics, which we refer to as *menu blocks*. Since these menu blocks are in the same position in both pages, they have a common root-to-block path and thus, a same label. As a consequence, they are part of a same block class. We also notice that blocks with the same label tend to have the same role within their respective pages. That is, a *block class* is usually composed of blocks that belong to distinct pages of a Web site and that play the same role within the site. Various examples of blocks that belong to the same class occur in the news pages depicted in Figure 2.4, such as blocks that represent the title, the news body, the summary of the news in the page.



Uganda flooding washes away towns, lives

STORY HIGHLIGHTS
- Uganda declares a state of emergency as villages, crops submerged
- Nation one of the worst-hit in 17 African countries affected by floods

SOROTI, Uganda (AP) -- Uganda declared a state of emergency Thur
of the country as humanitarian workers tried to reach villages that ha

Uganda is
floods that
Africa in rec
been killed
thousands
people sinc
spokeswo

Officials confirm meteorite but question sickness claims

STORY HIGHLIGHTS
- NEW: Officials confirm a meteorite crashed in southern Peru
- Fiery ball seen falling from the sky and smashing into the Andean plain
- Hundreds say they have suffered headaches, nausea and breathing problems
- Experts say dust or gas released on impact probably to blame for sickness

LIMA, Peru (AP) -- A fiery meteorite crashed into southern Peru over the weekend, experts confirmed Wednesday.

But they were still puzzling over claims that it gave off fumes that sickened 200 people.

Witnesses told reporters that a fiery ball fell from the sky and smashed into the desolate Andean plain near the Bolivian border Saturday morning.

Get the latest headlines and real breaking news for free, direct to your mobile.

Figure 2.4. News pages ρ_1 and ρ_2, extracted from CNN Web site.

Chapter 3

Identifying Blocks and Block Classes

This Chapter presents the automatic segmentation algorithm which, based on the definitions presented on Chapter 2, produces as output all the blocks and block classes implicitly present in a Web site. Our algorithm tries to segment Web pages close to the perception of users about how each page should be divided into blocks and how these blocks should be clustered into block classes.

The idea that a Web page can be divided into blocks according to their role was presented in previous work. For instance, this idea motivates the concept of Pagelets presented in Bar-Yossef and Rajagopalan [2002], and the concept of cohesive regions or blocks in Cai et al. [2003]; Chakrabarti et al. [2008]; Kohlschütter and Nejdl [2008]. However, identifying the role of the different regions of a Web page is a hard task to accomplish in a completely automatic way.

Besides considering the concept of blocks, the approach for representing Web Sites contents we have presented in Chapter 2 also considers the concept of block classes, that are set of blocks that have a common role on a group of pages. In order to identify the block classes, our segmentation algorithm takes the whole set of pages into account when dividing each Web page into blocks. For instance, let us consider a menu that occurs in multiple pages of a given Web site. Since the menu has the same function within each page it occurs, we can say that these occurrences form a block class. Thus, the segmentation algorithm should assign equal labels for all menus of all pages where it appears as a block.

Besides the automatic segmentation algorithm, we also present in this Chapter a semi-automatic approach to identify blocks and block classes. This approach is a viable alternative to obtain a high quality segmentation and is used in this thesis as a reference to evaluate the quality of the segments found by the automatic segmentation method.

3.1 Traditional Web Page Segmentation

Approaches for automatic Web page segmentation consider a variety of methods and exploit different characteristics of the pages [Chen et al., 2001; Kovacevic et al., 2002; Cai et al., 2003]. In this Section, we describe two of the most prominent segmentation methods of the literature. We start by presenting the VIPS method, that is one of most popular solutions to this problem.

3.1.1 VIPS - Vision-based Page Segmentation Algorithm

In Cai et al. [2003], the authors propose the Vision-based Page Segmentation algorithm (VIPS), in which various visual cues are taken into account to achieve an accurate segmentation. When the users view a Web page through a Web browser, they get a 2D presentation which provides many visual elements to help distinguish different parts of the page, such as titles, menus, images, tables, etc [Yu et al., 2003]. VIPS tries to segment a Web page by simulating these visual perceptions of the users.

For VIPS, a Web page Ω is defined by a triple $\Omega = (O, \Phi, \delta)$. $O = \{\Omega_1, \Omega_2, ..., \Omega_N\}$ is a finite set of (non-overlapping) blocks. Each block can be recursively viewed as a sub-page associated with sub-structure induced from the whole page structure. $\Phi = \{\varphi_1, \varphi_2, ..., \varphi_N\}$ is a finite set of visual separators, including horizontal separators and vertical separators. δ is a function that indicates if two blocks in O share a separator and therefore are adjacent ($\delta = O \times O \rightarrow \Phi \cup NULL$).

Since each Ω_i is a sub-page of the original page, its content structure is similar to Ω. Recursively, VIPS defines $\Omega_s^t = (O_s^t, \Phi_s^t, \delta_s^t)$, $O_s^t = \Omega_{st}^1, \Omega_{st}^2, ..., \Omega_{st}^N$, $\Phi_s^t = \varphi_{st}^1, \varphi_{st}^2, ..., \varphi_{st}^N$ and $\delta_s^t = O_s^t \times O_s^t \rightarrow \Phi_s^t \cup NULL$, where Ω_s^t is the t_{th} object in the block level s.

VIPS associates a degree of Coherence (DoC) for each block Ω_i, to indicate the degree of granularity of the block. We can control the granularity obtained in a segmentation process by pre-defining the Permitted Degree of Coherence (PDoC), which works as a threshold of the DoC value.

From the root element in the DOM tree structure of a Web page (as discussed in Section 2.1, the DOM tree is a hierarchical representation that provides a means to describe the structure and layout of Web pages), VIPS performs the following iterative process to identify the blocks of the page:

Phase 1: Visual Block Extraction

VIPS starts by obtaining information about each tag in the DOM structure (such as type of tag, position, color, font size, font weight, etc). Based in such an information, VIPS starts extracting the most evident blocks contained in the current subtree. Examples of visual cues that are used by VIPS to identify the blocks in this phase (for a more complete list, refer to Cai et al. [2003]):

- Tag cue: Tags such as `<HR>` are often used to separate different topics from visual perspective.
- Color cue: VIPS divides a DOM node if its background color is different from one of its children's.
- Text cue: If most of the children of a DOM node are Text nodes, VIPS does not divide it.

At the end of this phase, for each node that represents a visual block, its DoC value is set according to its intra visual difference.

Phase 2: Visual Separator Detection

In this phase, VIPS finds the separators between the nodes found in phase 1. A separator is represented by a pair (P_s, P_e), where P_s is the start pixel and P_e is the end pixel. Each separator has a weight indicating its visibility. To calculate P_s, P_e, and separator weight values, VIPS uses the renderer embedded in the Internet Explorer Web browser[1].

Phase 3: Content Structure Construction

After the actual separators are detected, the blocks on the same sides of all the separators are merged and represented as a node in the content structure. This process starts from the separators with the highest weight, and it iterates until all separators with minimum weights are met. The DoC of each node is defined. After that, each node is checked whether it meets the granularity requirement. For every node that fails, VIPS returns to phase 1 to further construct the sub content structure within the node. If the DoC of all the nodes is smaller than PDoC, the iterative process is then stopped and the vision-based content structure for the whole page is obtained.

[1]Internet Explorer is a trademark of Microsoft Corporation

As VIPS uses the renderer embedded in the Internet Explorer Web browser to segment pages, it is difficult to analyze the complexity of this algorithm. However, since each page must be rendered (like in a browser) prior to analysis, the VIPS algorithm is naturally too slow to be incorporated into ranking methods for big Web collections.

3.1.2 A Densitometric Approach to Web Page Segmentation

Kohlschütter and Nejdl [2008] examined the problem of identifying the block structure of Web pages from a textual perspective. Numerous observations confirm that the creation process of language (spoken or written) follows some probabilistic regularities and statistical laws, in particular Zipf's law (the frequency of a term is inversely proportional to its rank), Frumkina's law (when dividing text into segments of almost the same size, the frequency of a particular word follows a negative hypergeometric distribution) and the Altmann's law (the longer a linguistic construct, the smaller its constituents, and vice versa). The authors argue that such low-level properties of text constitute valuable features for segmentation decisions.

To illustrate, the authors define a Web page segmentation algorithm based on Altmann's findings [Antic et al., 2005] about the length dependence of neighbored sentences within a text flow, and their corresponding findings on the text density. The method proposed by the authors uses the number of tokens in a text fragment (or more precisely, its token density) instead of DOM-structural information. For that, they model a Web page as a series of text portions interleaved by a sequence of one or more tags, regardless of their meaning. They call such a sequence a *gap*. This simplifies the segmentation problem in distinguishing the gaps which separate two segments from gaps which do not.

For conducting the segmentation, a text-based density measure $\rho(s)$ was introduced, derived from the pixel-based density of Computer Vision-based approaches [Stockman and Shapiro, 2001]. Basically, it counts the number of tokens $|T_s|$ in a particular text segment s divided by the number of lines $|L_s|$ covered after word-wrapping the text at a fixed column width w_{max} (the empirically estimated optimal value for English text is between 80 and 90 characters). Due to the side effect of having an incompletely filled last line after wrapping, the latter is not taken into consideration unless it is the only line in the segment:

$$T'_s = \{t|t \in T_l,\ l_{first}(s) \leq l < l_{last}(s)\} \tag{3.1}$$

$$\rho(s) = \begin{cases} \frac{|T'_s|}{|L_s|-1}, & \text{if } |L_s| > 1 \\ |T_s|, & \text{otherwise} \end{cases} \tag{3.2}$$

where t is a term, l is a line identification, T_l is the set of terms in line l, $l_{first}(s)$ and $l_{last}(s)$ are the identifications of the first and the last lines of the text segment s, respectively.

The segmentation algorithm presented in Kohlschütter and Nejdl [2008] is based on a merge strategy called *Block-Fusion*. Adjacent text fragments of similar text density (interpreted as "similar class") are iteratively fused until the blocks' densities (and therefore the text classes) are distinctive enough. Using various settings, including a rule-based approach, the authors show that the resulting block structure closely resembles a manual segmentation, achieving a segmentation performance better than those achieved by Chakrabarti et al. [2008], which is another recente work that approached the Web page segmentation problem from a graph-theoretic perspective.

Unlike for the VIPS algorithm, computing Block-Fusion complexity is a trivial task. Assuming there are N atomic blocks on a page, the cost per iteration is $N-1$ comparisons and a maximum of $N-1$ fusions per iteration occur. The total number of operations for a maximum of k iterations until convergence therefore is:

$$(N-1) + (N-1-1) + \cdots + (N-k-1) = O(N) \tag{3.3}$$

3.1.3 Discussion

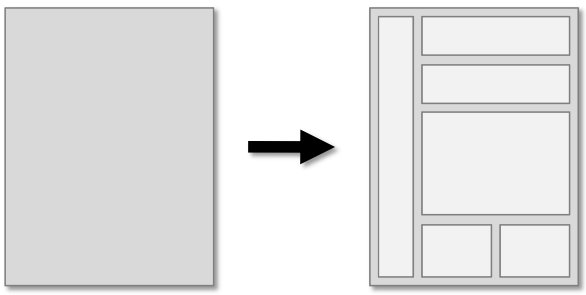


Figure 3.1. Traditional segmentation purpose: given a single Web page as input, to divide the page in a set of logical blocks.

As far as we know, all previous segmentation methods of the literature (including the methods proposed by Kohlschütter and Nejdl [2008] and Cai et al. [2003]) segment each Web page based on its content only, not considering the content of other pages present in the same Web site (see Figure 3.1). However, since many pages belonging to a same Web site share a common structure and look and feel, we hypothesize that we can achieve a more accurate segmentation by taking all pages of a same Web site into account when dividing them into blocks. Based on this idea, we here present a segmentation approach which segment pages according to properties of the whole Web site, instead of just information from a single page.

Besides segmenting the Web pages, our segmentation approach enables us to cluster blocks into block classes, that are set of blocks that have a common role on a group of pages, and that play an important function in the structure-based term weighting schemes proposed in this thesis.

3.2 A New Approach for Web Page Segmentation

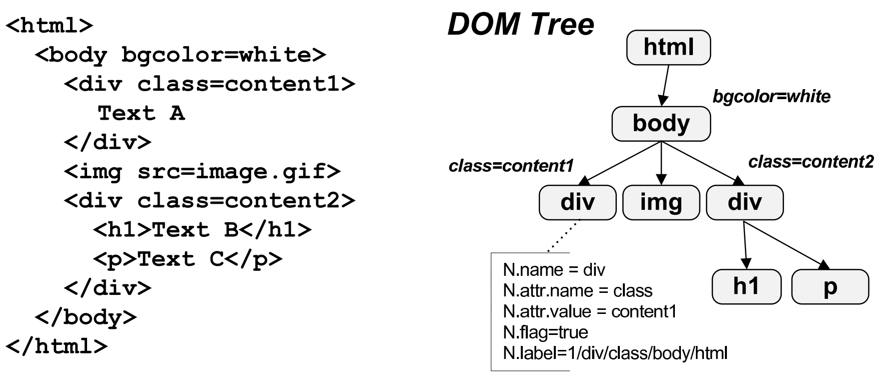
Algorithm 1 describes our automatic segmentation process, which tries to approximate the perception of users about how each page should be divided into segments and how these segments should be clustered into segment classes. It takes the set of pages $\mathcal{S}$ found in the Web site as input and produces as output a set of segment classes. Our segmentation algorithm is divided into three phases, that will be discussed in the following sections.

3.2.1 Phase 1: Pre-processing Web Pages

The first phase of our algorithm takes as input the pages of the site. The DOM trees of the pages are pre-processed to allow a representation that is closer to the final goal of automatically dividing the pages into segments. Such representation aggregates further information about each node and adjusts the DOM trees to be used in the second phase of the algorithm. Figure 3.2 depicts a segment of HTML code and the pre-processed version of its DOM tree. To simplify, only the leftmost node of the Figure depicts the additional information introduced by the pre-processing phase. This example is used in the following sections to show how each individual page is pre-processed (see Lines 1 to 4). The function *PreProcess* of our algorithm consists in four main operations which are described on the following.

Algorithm 1 *Segment*($\mathcal{S}$)

```
 1: { Phase 1: Pre-processing Web Pages}
 2: for each $p_i \in \mathcal{S}$ {$i$ is the number of a page} do
 3:     $\rho_i \leftarrow$ preProcess($p_i$);
 4: end for
 5: { Phase 2: Constructing the $SOM_{tree}$}
 6: $SOM_{tree} \leftarrow \emptyset$;
 7: for each $\rho_i \in \mathcal{S}$ {$i$ is the number of a page} do
 8:     for each node $N_\rho \in \rho_i$ traversed in pre-order do
 9:         if $\exists N_s \in SOM_{tree} \mid N_s.label = N_\rho.label$ then
10:             $N_s.counter \leftarrow N_s.counter + 1$;
11:             if $N_\rho.flag$ then
12:                 Append $i$ in $N_s.pageList$;
13:             end if
14:         else
15:             insertNewNode($SOM_{tree}$,$N_\rho$,$i$);
16:         end if
17:     end for
18: end for
19: {Phase 3: Refining the $SOM_{tree}$ }
20: for each node $N_s \in SOM_{tree}$ traversed in pre-order do
21:     if $N_s$ is a internal node AND $N_s.pageList \neq \emptyset$ then
22:         for each node $n_s \in$ descendants($N_s$) traversed in pos-order do
23:             if distance($N_s$,$n_s$)$< \alpha$ AND $n_s$ is a leaf node then
24:                 Append $n_s.pageList$ to $N_s.pageList$;
25:                 Remove $n_s$ from $SOM_{tree}$;
26:             end if
27:         end for
28:     end if
29: end for
30: for each node $N_s \in SOM_{tree}$ traversed in pos-order do
31:     if $n_s.counter < \beta$, $\forall n_s \in children(N_s)$ then
32:         for each $n_s \in children(N_s)$ do
33:             Remove $n_s$ from $SOM_{tree}$;
34:             Add $n_s.pageList$ to $N_s.pageList$;
35:             Add $n_s.counter$ value to $N_s.counter$;
36:         end for
37:     end if
38: end for
39: {Each node $N_s$ of $SOM_{tree}$ represents now a block class, where $N_s.pageList$ contains the list of blocks of
    $N_s$, and $N_s.label$ contains the label of the blocks}
40: Return the $SOM_{tree}$
```



```
<html>
  <body bgcolor=white>
    <div class=content1>
      Text A
    </div>
    <img src=image.gif>
    <div class=content2>
      <h1>Text B</h1>
      <p>Text C</p>
    </div>
  </body>
</html>
```

Figure 3.2. An example of HTML code and the pre-processed version of its DOM tree.

Creating the node labels

Given a page ρ, its DOM tree is a structure that represents the hierarchical relationship between the tags found in the page. Each tag is represented by a node N that contains information about the *name* of the tag ($N.name$) and a list of its *attributes* ($N.attr$), where each attribute is composed by a pair of *name* ($N.attr.name$) and *value* ($N.attr.value$). We use this information to recursively define the *label* of each node N ($N.label$) as the concatenation of its $N.name$, $N.attr.name$ field values and the *label* value of its parent in the DOM tree (considering it as empty when the node is the root of the tree), separating each token in the concatenation by a slash. Whenever two sibling nodes get equal values assigned by this strategy, we distinguish them by adding a sequential number to their *label* values, so that each node has a unique *label*.

For instance, in the leftmost node of Figure 3.2, the value of *name* is "div" and the only value of attribute name is "class", thus attr.name={"class"}. Its *label* is "1/div/class/body/html". The value "1" is added because there is a second node that would get the same *label* (see the rightmost div tag). This node is the identified by "2/div/class/body/hmtl" in our representation. For the node labelled as "img" in the Figure, the *label* is "img/body/html".

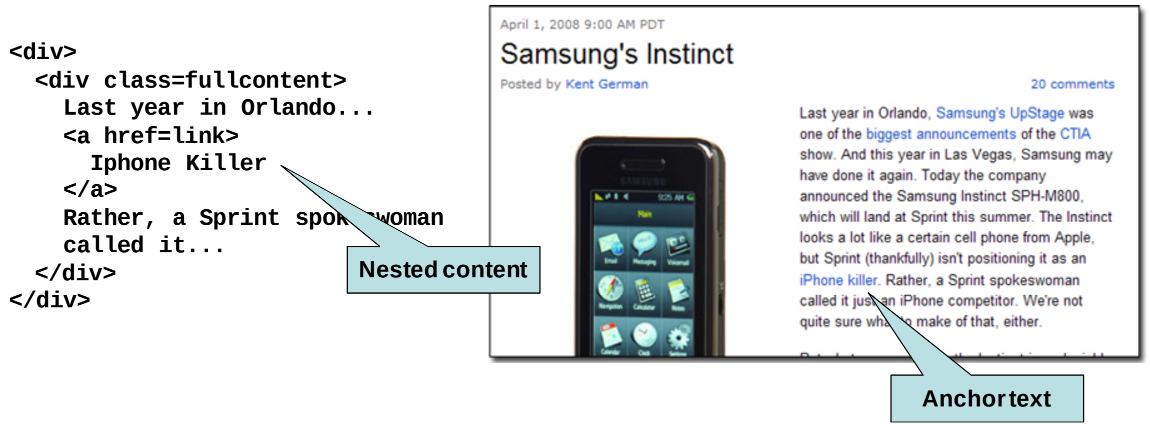
Labelling tags associated to textual content

Although information about textual content is not included in the DOM tree, it is useful for our segmentation algorithm. Thus, we add to each node N a boolean value ($N.flag$), which indicates whether it has textual content assigned to it ($N.flag = true$) or not($N.flag = false$).

Dealing with nested content

Our segmentation algorithm assumes that only the leaf nodes of a DOM tree may have textual content assigned to them. Thus, whenever we find internal nodes with textual content, we do not represent their children in our DOM tree representation, and thus it becomes a leaf node. The textual content associated to the removed tags is then associated to this new leaf node.

For instance, let us consider the second tag `<div>` in DOM tree of Figure 3.3. Notice that such a tag contains a text directly attached to it (Last year in Orlando...). Besides, inside this text there is a tag `<a>`, that also has a textual content (anchor text). In this case, we say that the content of `<a>` is nested to the content of `<div>`. Therefore, the tag `<a>` is removed from our DOM tree representation.



```
<div>
  <div class=fullcontent>
    Last year in Orlando...
    <a href=link>
      Iphone Killer
    </a>
    Rather, a Sprint spokeswoman
    called it...
  </div>
</div>
```

April 1, 2008 9:00 AM PDT

Samsung's Instinct

Posted by Kent German 20 comments

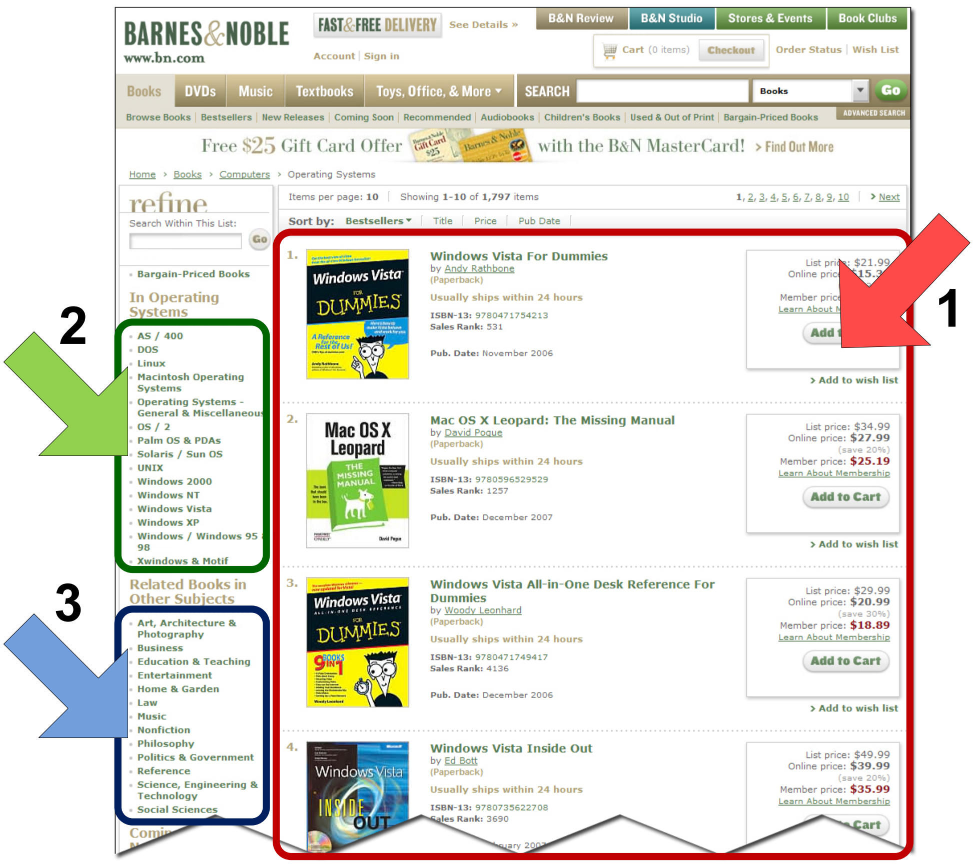
Last year in Orlando, Samsung's UpStage was one of the biggest announcements of the CTIA show. And this year in Las Vegas, Samsung may have done it again. Today the company announced the Samsung Instinct SPH-M800, which will land at Sprint this summer. The Instinct looks a lot like a certain cell phone from Apple, but Sprint (thankfully) isn't positioning it as an iPhone killer. Rather, a Sprint spokeswoman called it just an iPhone competitor. We're not quite sure what to make of that, either.

Figure 3.3. Example of tag with nested content.

Dealing with recurrent regions

Another change we perform in the DOM trees is to remove sequences of tags disposed in a recurrent (or regular) way. In the DOM tree of a typical Web page, it is very common to find regions with a quite regular structure. Let us consider the Web page depicted in Figure 3.4. Notice that the *menu* signalized by the arrow 2 is formed by a set of links disposed in a vertical bar. As the set of tags that separate two adjacent links is always the same, we can say these tags are disposed in a regular or recurrent way.

Regions with recurrent structure commonly contain a list of items highly related to each other such as list of links, products, or paragraphs (notice that the regions



Items per page: 10 | Showing 1-10 of 1,797 items 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 | > Next

Sort by: Bestsellers | Title | Price | Pub Date

- Bargain-Priced Books

In Operating Systems

- AS / 400
- DOS
- Linux
- Macintosh Operating Systems
- Operating Systems - General & Miscellaneous
- OS / 2
- Palm OS & PDAs
- Solaris / Sun OS
- UNIX
- Windows 2000
- Windows NT
- Windows Vista
- Windows XP
- Windows / Windows 95 & 98
- Xwindows & Motif

Related Books in Other Subjects

- Art, Architecture & Photography
- Business
- Education & Teaching
- Entertainment
- Home & Garden
- Law
- Music
- Nonfiction
- Philosophy
- Politics & Government
- Reference
- Science, Engineering & Technology
- Social Sciences

Comin

1. Windows Vista For Dummies
by Andy Rathbone
(Paperback)
Usually ships within 24 hours
ISBN-13: 9780471754213
Sales Rank: 531
Pub. Date: November 2006
List price: $21.99
Online price: $15.3
Member price
Learn About M
Add t
> Add to wish list

2. Mac OS X Leopard: The Missing Manual
by David Pogue
(Paperback)
Usually ships within 24 hours
ISBN-13: 9780596529529
Sales Rank: 1257
Pub. Date: December 2007
List price: $34.99
Online price: $27.99
(save 20%)
Member price: $25.19
Learn About Membership
Add to Cart
> Add to wish list

3. Windows Vista All-in-One Desk Reference For Dummies
by Woody Leonhard
(Paperback)
Usually ships within 24 hours
ISBN-13: 9780471749417
Sales Rank: 4136
Pub. Date: December 2006
List price: $29.99
Online price: $20.99
(save 30%)
Member price: $18.89
Learn About Membership
Add to Cart
> Add to wish list

4. Windows Vista Inside Out
by Ed Bott
(Paperback)
Usually ships within 24 hours
ISBN-13: 9780735622708
Sales Rank: 3690
List price: $49.99
Online price: $39.99
(save 20%)
Member price: $35.99
Learn About Membership

Figure 3.4. Recurrent structures: the regions pointed by the arrows contain sets of items (be links or books) disposed in a regular or recurrent way.

signalized by the arrows 1 and 3 are also examples of regions with recurrent structures). Thus, each list has a single purpose within their respective pages and then would probably be seen by an user as a single segment. In this way, the DOM subtree of these regions is represented by a single node in our DOM tree representation, in order that all textual content of the subtree with recurrent structure become directly linked to this node. This process is depicted in Figure 3.5.

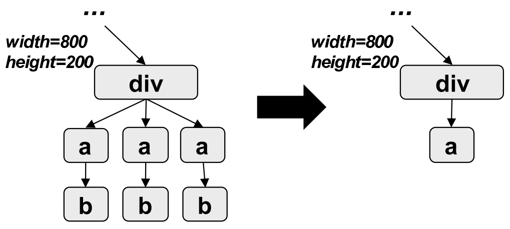


Figure 3.5. Recurrent substructures of the DOM trees are not represented in our DOM tree representation.

Identifying recurrent structures in the DOM tree of Web pages is a procedure commonly adopted in information extraction research area [Liu et al., 2003; Mehta et al., 2005; Álvarez et al., 2008], that regards the generation of wrappers able to extract particular information from Web documents. The motivation is that the information to be extracted is often placed in a particular order, and repetitive patterns can be found in these Web pages when multiple records aligned together. For instance, Chang and Lui [2001] propose a linear algorithm based on Patricia trees [Frakes and Baeza-Yates, 1992] to identify repeated patterns in a DOM tree given as input. Algur and Hiremath [2006] also deals with the problem of identification and extraction, and propose a algorithm based on the visual clue information to identify recurrent regions.

3.2.2 Phase 2: Constructing the SOM_{tree}

The second phase of the Algorithm 1 creates an auxiliary hierarchical structure named SOM_{tree} (SOM is an acronym to *Site Object Model*) that summarizes the DOM trees of all pages found in a Web site (see lines 5-18).

Each node N_s of the SOM_{tree} contains the same information found in a node N_ρ of our DOM tree representation, but has two extra fields: $N_s.counter$, with the number of pages where it occurs in the site (i.e., the number of pages that contain a node N_ρ such that $N_\rho.label = N_s.label$); and $N_s.pageList$, with the list of pages where it occurs associated to textual content in the site. Notice that the size of $N_s.pageList$ may be smaller than the value of $N_s.counter$, since only tags occurrences associated with textual content are inserted in $N_s.pageList$. A single tag may occur in a page associated to content and appear in a second page without any content.

As an example of a SOM_{tree} construction, consider a very simple Web site $\mathcal{S}$ formed by only two pages, ρ_1 and ρ_2, depicted in Figure 3.6. A SOM_{tree} for this site can be defined by merging ρ_1 and ρ_2 in a single structure. Phase 2 of our algorithm starts by reading the page ρ_1, through a pre-order traversal (lines 8-17). For each node N_ρ of ρ_1, the algorithm checks if the SOM_{tree} already have a node N_s such that $N_s.label = N_\rho.label$ (see line 9). However, since the SOM_{tree} is created empty (see line 6), all nodes N_ρ found in this first page are merged into SOM_{tree} through the function *insertNewNode*, that is called on line 15, and detailed in Algorithm 2. The insertion point is guided by the value of $N_\rho.label$. Function *insertNewNode* assigns the value of $N_\rho.label$ to $N_s.label$, and assigns 1 to $N_s.counter$. If N_ρ has textual content ($N_\rho.flag = true$), then function *insertNewNode* also adds the id of page ρ_1 (value 1) to the pageList of the node N_s.

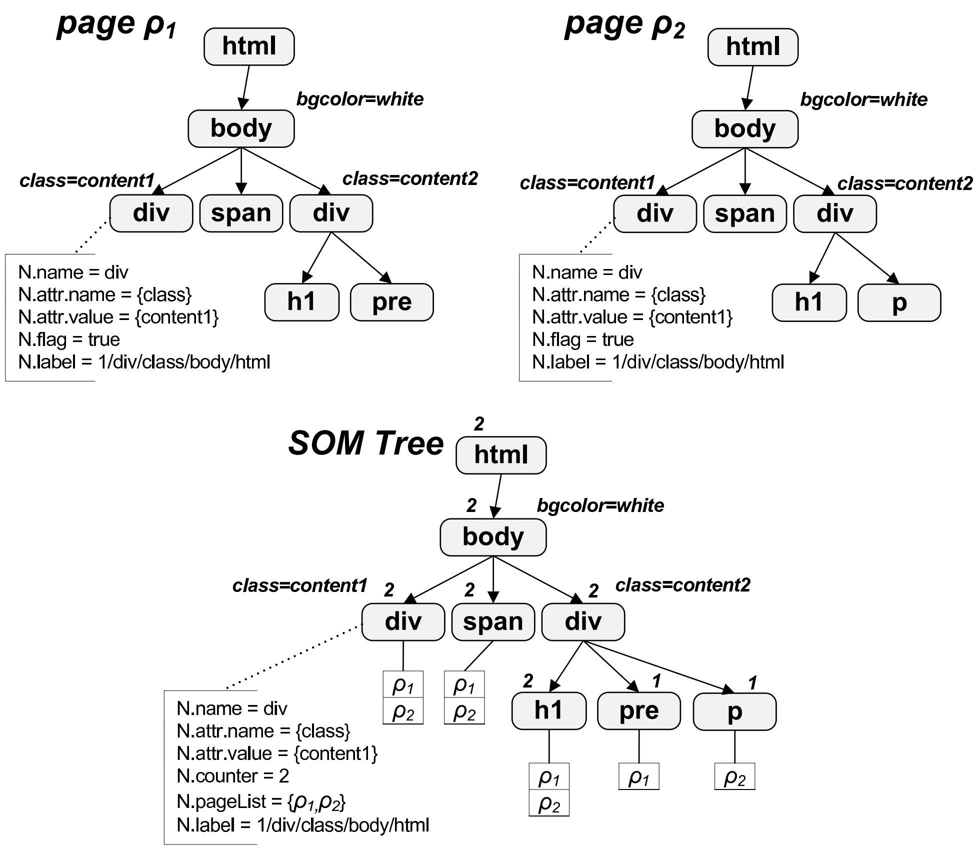


Figure 3.6. An example of a SOM_{tree} construction. The *pageList* values are depicted below each node of this structure.

After that, the algorithm reads the second page of $\mathcal{S}$, ρ_2, to merge it into SOM_{tree}. In Figure 3.6, we can see that this page starts with the tag `<html>`. In the line 9, it is verified that the SOM_{tree} has already a node N_s with the label `html`, and thus the algorithm only updates the information of N_s (lines 10-12): its counter value is incremented.

In Figure 3.6, we see that our page representation of ρ_1 and ρ_2 diverge below the nodes `div` on right. Below this node, the page ρ_1 has the nodes `h1` and `pre`, and the page ρ_2 has the nodes `h1` and `p`. As the nodes `h1` have the same path in the DOM tree of both pages, then they result in a single node in the SOM_{tree}. On the other hand, the nodes `pre` (in ρ_1) and `p` (in ρ_2) occur only once in their respective pages. Thus, each of them results in a different node in the SOM_{tree}. The counter of these two nodes is 1 in both cases.

Algorithm 2 insertNewNode(SOM_{tree}, N_ρ, i)

```
1: Create a empty node $N_s$;
2: $N_s.label = N_\rho.label$;
3: $N_s.counter = 1$;
4: if $N_\rho.flag$ then
5:     $N_s.pageList = \{i\}$;
6: end if
7: Insert $N_s$ in $SOM_{tree}$ according to $N_s.label$
```

The complexity of the second phase of our segmentation algorithm is $O(|\rho| \times P)$, where P is the number of pages of the Web collection, and $|\rho|$ is the number of nodes (or tags) of the pages. In other words, the insertion of a page into SOM_{tree} has a linear complexity in the number of nodes of that page.

3.2.3 Phase 3: Refining the SOM_{tree}

The third phase of our algorithm has the goal of removing nodes of the SOM_{tree} that would be considered by a human as internal parts of one or more segments of the site. By removing these nodes, that we refer to as noisy nodes, the SOM_{tree} become a structure formed only by nodes that belong to the label (root-to-segment path) of one or more segments, and each each leaf node of this new structure will refer to a distinct segment class of the site.

To better illustrate what is a noisy node, lets consider that there is a table inside the body of a news. This table would probably be considered by a human as part of a single block containing the body of the news. However, when including this page in the SOM_{tree}, the content of the table would create new nodes in the structure, and thus would let the segmentation system to consider the table as a separate block.

When analyzing the results produced by Phase 2 of our segmentation algorithm for several samples, we realized that in general, these noisy segments occur when a given set of pages has a region with pure textual content, while another set of pages has textual content mixed with other type of information in the same region, such as a table containing text, or a different textual style. When humans see such variations, they can easily realize that the region has the same function in both sets of pages. However, the small structural differences introduced affect the representation of the site in the SOM_{tree}. The main goal of the third phase of our algorithm is to remove the noisy nodes, what is done by applying two different heuristics.

The first heuristic performs a join of nested nodes that have textual content and that occur close in the SOM_{tree}. The distance adopted is the difference in the depth between the two nested nodes analyzed. We noticed that as this distance increases, also increases the chance that nodes represent different block classes of the site. On the other hand, when this distance is small, it is mostly due to small variations in pages, and thus the nodes probably represent the same block class.

The ideal value for this distance, represented by the threshold α in Algorithm 1, can be empirically set through experiments. An alternative to avoid the necessity of such threshold is to use a clustering algorithm to group the pages of the site according to their templates [Vieira et al., 2009; Crescenzi et al., 2005, 2003]. After clustering the pages, we can run the algorithm 1 for each cluster. This procedure would avoid the presence of pages with large differences in their structure as input to the segmentation algorithm, and then would allow all nested content to be joined, without a risk of joining blocks of pages with distinct structure. On the other hand, the use of clustering would make the segmentation process more expensive and the results we obtained by using our heuristic were good enough for avoiding such extra computational cost.

As an example, consider the SOM_{tree} segment depicted in Figure 3.7(a). Notice that this segment contains two nodes, `div` and `p`, and that the `p` node is descendant of the `div` node. As one can see, there is a small variation in the structure in this region of the pages, since a set of pages contain textual content connected to the first node (`div`), and another set of pages contain textual content associated to the second node `p`.

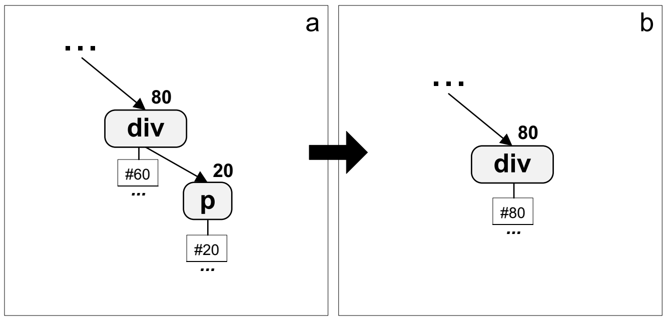


Figure 3.7. (a) A SOM_{tree} segment example and (b) its pruned version, considering a α threshold equal to 6. Notice that, instead of presenting the list of documents in each leaf element node, this figure presents only the number of documents in each list.

Line 23 of the algorithm 1 verifies whether the distance between these two nodes is smaller than the threshold α or not. If it is smaller, then the pageList of the descendant node is appended to the ancestor, and the descendant node is deleted from the SOM_{tree}(lines 24-25). On the other side, if this distance is greater than the threshold α, then we assume that these nodes refer to distinct segment classes. In our experiments, we empirically set the value 6 for this threshold based on training experiments. Considering this value for the threshold α, the node `p` depicted in Figure 3.7(a) must be pruned during the refining process, and its pageList merged to the pageList of the node `div`. The result of this process is depicted in Figure 3.7(b).

Notice that, although the join can fail in some cases, it does not cause a significant change in the final result of the method, since if we set threshold α too high, the only change is that a few blocks with different classes may be joined in a single class. On the other hand, if we set it too low, some small differences in the structure of pages may cause the split of blocks considered by humans as single blocks. While these changes may contribute to result in a set of block classes different from the ones produced by humans, the impact of these changes for ranking purposes is expected to be small for Web sites with regular structure, since only a few pages would have a few changes in our proposed term weighting scheme.

To obtain the complexity of this heuristic, consider that the pos-order transversal performed on line 22 is limited by the descendentes of the node N_s whose distance to N_s is at most α (see line 23). Considering that each node of a SOM_{tree} can have at most k children, the maximum number of steps of each transversal performed on line 22 is:

$$1 + k + k^2 + k^3 + ... + k^\alpha = \sum_{i=0}^{\alpha} k^i = \frac{1 - k^{\alpha+1}}{1 - k}$$

That is a geometric serie. However, since k tend to be very small when compared to the number of nodes of the SOM_{tree}, we can consider it as a constant value. In this case, the number of steps performed by each transversal of the line 22 is actually $O(1)$. Thus, the complexity of the first heuristic is defined by the complexity of the transversal performed on line 20, that is linear on the number of nodes of the SOM_{tree}.

The second heuristic joins all the children of a node N to it whenever all of them have counter value smaller than a threshold β (lines 31-37). In our experiments, the value of β was empirically set to eight, as described in Section 6.2.2. This parameter is related to the minimum quantity of elements in a block class in order to allow safe use of our block-based term weighting methods. To illustrate, consider the node `div` and `p` of the SOM_{tree} segment depicted in Figure 3.8(a). Assuming that the counter value of its children (nodes `pre` and `p`) are lower than the value of the threshold β, then such nodes would be pruned during the refining process, and their pageList merged to the pageList of the node `div`. The result of this process is depicted in Figure3.8(b).

Notice that the children of a node N can be joined to their parent only when all

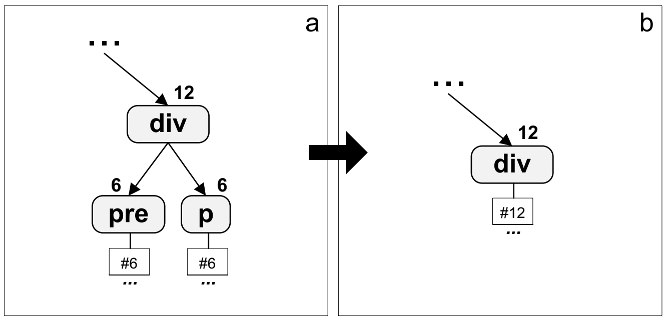


Figure 3.8. (a) A SOM_{tree} segment example and (b) its pruned version, considering a β threshold equal to 8.

of them have counter value smaller than the threshold β. The purpose of this rule is to avoid the generation, during this stage of the refining process, of new nested nodes with textual content. Thus, even after this second phase of the refining process, some nodes might remain with counter value smaller than the threshold β. To illustrate, consider the SOM_{tree} segment depicted on Figure 3.9). Considering a β threshold equal to 8, the nodes `pre` and `p` can not be pruned, since the count value of the node `pre` is bigger than the threshold.

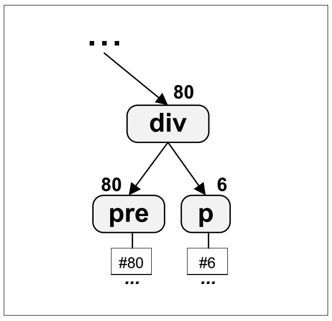


Figure 3.9. A SOM_{tree} segment example, where the node `p` remains with the counter value smaller than the threshold β after the refining process.

Since the transversal of the line 30 is performed on all nodes of the SOM_{tree}, then this second heuristic has also a linear complexity - $O(N_{SOM})$, where N_{SOM} is, as before, the total number of nodes of the SOM_{tree}.

The result of the algorithm after the refining is a set of block classes that are described by the SOM_{tree} (see Figure 3.10). Each leaf node N of the final SOM_{tree} represents a block class, with label $N.label$ and occurring in pages $N.pageList$. From

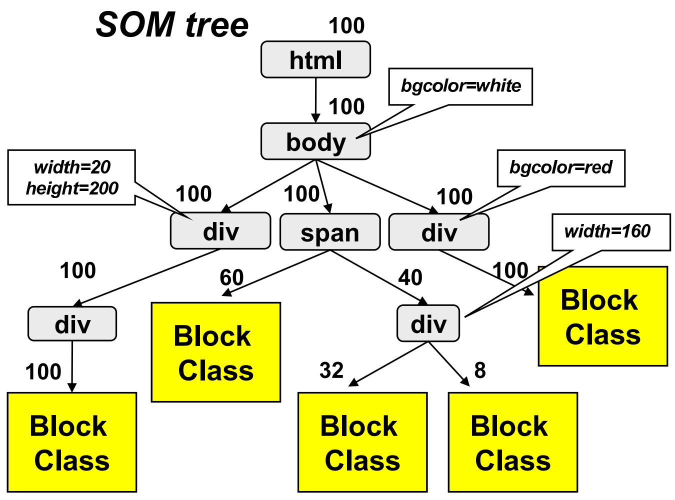


Figure 3.10. After the refining process each leaf node of the SOM_{tree} refers to a block class of the Web Site.

the structure of the SOM_{tree} it is possible to define how each page is partitioned into blocks by performing a matching between the SOM_{tree} and the DOM tree of the page. In the next chapter, we show ranking methods that take this structural information into account.

3.3 A Semi-automatic Approach for Identifying Blocks and Block Classes

In this Section we present a semi-automatic technique for identifying blocks and block classes occurring in a Web site. This approach is a viable alternative to obtain a high quality segmentation and its results can be used as a reference to check the quality of results provided by our automatic segmentation algorithm. This segmentation approach is based on the Vision-based page segmentation algorithm (VIPS), proposed by Cai et al. [2003] and discussed on Section 3.1.1.

The first step of our semi-automatic technique is to cluster the pages of the Web site according to their internal structure. This procedure, that is illustrated on Figure 3.11, can be made either manually or automatically, through some techniques available on literature [Vidal et al., 2006; Chehreghani et al., 2008; Crescenzi et al., 2005]. Notice that this clustering procedure places pages with the same set of blocks in a same cluster. In other words, after the pages are clustered, for every block found in given page ρ of a cluster $\mathcal{P}$ there is a block with the same label (root-to-block path) in all other pages of $\mathcal{P}$.

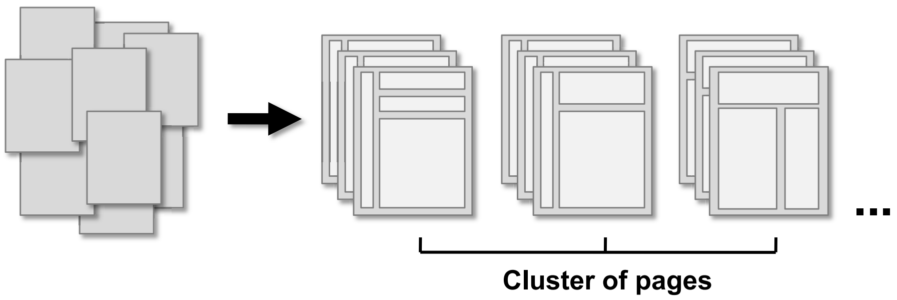


Figure 3.11. Clustering the pages of a Web site according to their internal structure.

After the pages are clustered, the VIPS algorithm is used to segment the pages of each cluster through a manual selection of the parameter PDoC (described in Section 3.1.1). Since the pages of a same cluster have a similar block structure, all of them can be segmented through the same PDoC value. Further, to assert the quality of the segmentation, whenever users do not agree with the segmentation performed by the VIPS algorithm, they can adapt the division of the blocks to their perception about how it should be performed. Hence, this semi-automatic process requires an extra human effort, since it is necessary to identify the clusters (manually or not), to manually set the parameter PDoC for each cluster found in the collection, and adjust the segmentation performed by VIPS (if needed).

Notice that, in this segmentation approach, each page cluster has a particular set of block classes. For instance, consider a *menu* that occurs in all pages of a given Web site. Since the menu has the same label in all pages, the automatic segmentation approach identifies these occurrences as a single block class. However, in the semi-automatic approach, each page cluster has a particular block class containing the menu of the pages. As one can see, this difference between the segmentation approaches affect only the block classes containing blocks of the templates of the pages

3.3.1 Evaluating Segmentation Algorithms

In this section we discuss methods of evaluating the quality of the segments found by segmentation algorithms. The evaluation metrics presented in this section are used in Section 3.4 to evaluate the quality of the blocks found by the fully automatic segmentation algorithm proposed in this thesis. According to Kohlschütter and Nejdl [2008], a segmentation algorithm must be evaluated according to its ability of segmenting Web pages close to the perception of users about how these pages should be divided into blocks. Thus, to compare different segmentation algorithms, we use statistic measures to evaluate the agreement between a collection of blocks manually identified by specialists, and the collection of blocks obtained by each segmentation technique we want to compare.

Since each token in a document is assigned to only one block, measures of agreement between two partitions, like the *Adjusted Rand index (AdjRAND)* [Hubert and Arabie, 1985; Vinh et al., 2009], can be used [Chakrabarti et al., 2008]. The Adjusted RAND index between two partitions measures the fraction of pairs of terms that are either grouped together (in a same block) or placed in separate blocks in both partitions. Hence, the higher the Adjusted RAND index between the segmentation output (obtained by the algorithm we want evaluate) and a manually labeled segmentation, the better the segmentation quality of the algorithm.

Given a set of n terms $T = \{t_1, t_2, \ldots, t_n\}$, suppose $U = \{u_1, u_2, \ldots, u_R\}$ and $V = \{v_1, v_2, \ldots, v_C\}$ represent two different partitions of the terms in T such that $\bigcup_{i=1}^{R} u_i = \bigcup_{j=1}^{C} v_j = T$ and $u_i \cap u'_i = v_j \cap v'_j = \emptyset$ for $1 \leq i \neq i' \leq R$ and $1 \leq j \neq j' \leq C$. Suppose that U is our manually labeled segmentation and V is the set of segments obtained by the segmentation technique we want evaluate. The contingency Table 3.1

U/V	v_1	v_2	$\dots$	v_C	$Sums$
u_1	n_{11}	n_{12}	$\dots$	n_{1C}	a_1
u_2	n_{21}	n_{22}	$\dots$	n_{2C}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
u_R	n_{R1}	n_{R2}	$\dots$	n_{RC}	a_R
$Sums$	b_1	b_2	$\dots$	b_C	

(3.4)

Table 3.1. Notation for the contingency table for comparing partitions U and V.

can be formed to indicate group overlap between U and V (n_{ij} denotes the number of common terms in blocks u_i and v_j: $n_{ij} = |u_i \bigcap v_j|$).

Let a be the number of pairs of terms that are placed in the same block in U and in the same block in V, b be the number of pairs of terms in the same block in U but not in the same block in V , c be the number of pairs of terms in the same block in V but not in the same block in U , and d be the number of pairs of terms in different blocks in both partitions. The quantities a and d can be interpreted as agreements, and b and c as disagreements. The Rand index [Rand, 1971] between two partitions is given by:

$$RI = \frac{a+d}{a+b+c+d} \tag{3.5}$$

The Rand index lies between 0 and 1 (when the two partitions agree perfectly, the Rand index is 1). A shortcoming with the Rand index is that the expected value of the Rand index of two random partitions does not take a constant value (say zero). The adjusted Rand index proposed by Hubert and Arabie [1985] is the corrected-for-chance version of the Rand index, and its formula is presented below:

$$ARI = \frac{\binom{n}{2}(a+d) - [(a+b)(a+c)+(c+d)(b+d)]}{\binom{n}{2}^2 - [(a+b)(a+c)+(c+d)(b+d)]} \tag{3.6}$$

or, considering the notation of the contingency table 3.1:

$$ARI = \sum_{i,j} \frac{\binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}} \tag{3.7}$$

AdjRAND adjusts the values of RAND index so that it is upper bounded by 1 and, and scores a value close to 0 for a random segmentation.

The second metric we use to evaluate segmentation algorithms is the *Average Mutual Information* [Strehl and Ghosh, 2003; Strehl et al., 2000]. Mutual information is a measure of the quantity of information shared between two partitions, and provides an indication of the shared information between two different segmentations of a same page. Let T the set of n terms occurring in a Web page, and suppose φ_1 and φ_2 represent two different partitions of T, with κ_1 and κ_2 blocks respectively. Let X and Y be the random variables described by φ_1 and φ_2. If X and Y are independent random variables, then X have no information about Y, and vice versa. In this case, we say the mutual information between the variables is zero. On the other hand, if X and Y are identical, then all information of X is contained in Y, i.e., the knowledge of X add nothing to the knowledge of Y, and vice versa. In this case, we say that the mutual information if given by the amount of information, or entropy, of X.

Let $I(X;Y)$ denote the mutual information between X and Y, and $H(X)$ denote the entropy of X. Based on the previous idea, the mutual information measure can be expressed by: I(X,Y) = H(X) - H(X|Y).

There is no upper bound for $I(X;Y)$, so for easier interpretation and comparisons, a normalized version of $I(X;Y)$ that ranges from 0 to 1 is desirable. Hence, Strehl and Ghosh [2003] proposed the definition of normalized mutual information (NMI) using geometrical mean:

$$NMI(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}} \tag{3.8}$$

Equation above needs to be estimated by the sampled quantities provided by the partitions. Thus, in practice, the normalized mutual information between two clusterings is given by:

$$NMI(\varphi_1, \varphi_2) = \frac{\sum_{h=1}^{\kappa_1} \sum_{l=1}^{\kappa_2} n_{h,l} \log(\frac{n \cdot n_{h,l}}{n_h n_l})}{\sqrt{(\sum_{h=1}^{\kappa_1} n_h \log \frac{n_h}{n})(\sum_{l=1}^{\kappa_2} n_l \log \frac{n_l}{n})}} \tag{3.9}$$

where n is the total number of terms, n_h is the number of terms in block h, n_l the number of terms in block l and $n_{h,l}$ the number of terms in both blocks h and class l. The NMI value is 1 when partitions agree perfectly, and close to 0 for a random partitioning.

3.4 Experimental Results

To validate our automatic segmentation algorithm we performed experiments on 4 real Web collections namely IG, CNN, CNET, and BLOGs:

- IG Collection: The first collection contains 34,460 pages crawled from IG (see www.ig.com.br), that is one of the largest Brazilian Web portals. This collection is composed of a recipe site, a forum site, and a news Web site, as detailed in Table 3.2.

- CNN Collection: The second collection is a crawling of the international edition of CNN.com (see www.cnn.com), and is composed of 16,257 Web pages.

- CNET Collection: This collection was obtained by crawling four Web sites affiliated to the CNET Web portal (see www.cnet.com): CNET News, that provides news, blogs, and special reports about technology; CNET Download, composed by a large set of pages containing free downloads; CNET Shopper, which is a virtual shop of tech products; and CNET Reviews, composed by a set of pages containing reviews of products.

Collection	**Site**	**Number of Pages**	**Domain**
IG	News	26,466	www.ultimosegundo.com.br
	Forum	6,389	www.jornaldedebates.com.br
	Recipe	1,605	www.panelinha.com.br
	Total	**34,460**	
CNN	News site	16,257	www.cnn.com
	Total	**16,257**	
CNET	News	131,474	www.news.com
	Downloads	99,186	www.downloads.com
	Reviews	64,142	reviews.cnet.com
	Shopper	57,968	www.shopper.com
	Total	**352,770**	
BLOGs	Boing Boing	14,173	www.boingboing.net
	CNET	8,054	news.cnet.comtech-blogs
	Engadget	6,343	www.engadget.com
	Gizmodo	4,454	us.gizmodo.com
	Google	1,050	googleblog.blogspot.com
	Life Hacker	3,997	www.lifehacker.com
	Mashable	7,410	www.mashable.com
	Slash Film	5,376	www.slashfilm.com
	Tech Crunch	3,198	www.techcrunch.com
	Total	**54,055**	

Table 3.2. The set of Web sites crawled for each document collection.

- BLOGs Collection: The fourth collection was obtained by crawling the posts of 9 popular blogs from the top popular list presented in Technorati Blog [2]. Table 3.2 presents the list of the crawled blogs.

3.4.1 Segmenting the Collections

In this Section we present statistics about the segmentation process of the pages of each collection adopted in the experiments. The pages of each collection was segmented by using the semi-automatic approach (discussed in section 3.3), the SOM_{tree} based approach (discussed in section 3.2), and the Block-fusion method (proposed by Kohlschütter and Nejdl [2008] and described in section 3.1.2). We use the results obtained by the semi-automatic approach as a reference to evaluate the quality of

[2]http://technorati.com/blogs/top100

results provided by the automatic segmentation algorithms, through the metrics we introduced in Section 3.3.1.

3.4.1.1 Semi-automatic segmentation

We start by presenting statistics about the segmentation process performed by the semi-automatic approach (presented in Section 3.3). To assert the quality of the segmentation, the clustering of the Web pages according to their block structure was performed manually. Table 3.3 presents the total number of blocks and block classes found by this method. This table also shows the number of blocks found in small classes, that are block classes containing less than β (8) elements. Notice that the small classes found by this segmentation approach are derived from page clusters containing less than β pages. For instance, the structure of the *main page* of a Web site tend to differ from all other pages of the same Web site. In this case, each block found in the *main page* will generate a distinct block class, each one with just one block.

As one can see in Table 3.3, the semi-automatic method has found no small classes in BLOGs collection, since it is formed only by posts of their respective blogs, and all posts from each crawled blog tend to have the same set of blocks. In this way, for each block found in a given page of the BLOGs collection, there are many other blocks with the same label occurring in other pages of this collection.

Figure 3.12 shows the distribution of the number of blocks on the block classes found in four collections. Notice that some block classes have the same number of blocks. This happens when the block classes belong to the same cluster of pages.

3.4.1.2 Automatic Segmentation

Table 3.4 presents the total number of blocks, block classes, and the number of blocks present in small block classes found by the automatic segmentation approach. We can see that, despite the number of pages in CNN collection be smaller than the other collections, the number of block classes found in this collection is relatively high when

Semi-automatic segmentation approach				
Collection	**Site**	**B Classes**	**Blocks**	**Blocks SC**
IG	News	104	406,965	23
	Forum	101	201,325	15
	Recipe	41	25,443	17
	Total	**246**	**633,733**	**55**
CNN	News site	158	257,139	17
	Total	**158**	**257,139**	**17**
CNET	News	129	1,834,661	18
	Downloads	115	1,494,135	15
	Reviews	50	800,860	17
	Shopper	79	682,473	19
	Total	**373**	**4,812,129**	**69**
BLOGs	Boing Boing	22	311,806	0
	CNET	16	128,864	0
	Engadget	33	209,319	0
	Gizmodo	12	53,448	0
	Google	19	19,950	0
	Hacker	12	47,953	0
	Mashable	22	163,020	0
	Slash Film	13	69,888	0
	Tech Crunch	12	38,376	0
	Total	**161**	**1,042,624**	**0**

Table 3.3. The number of blocks and block classes found by the semi-automatic approach in all Web site collections. Table also shows the number of blocks found in small classes, i.e., block classes with less than β elements (Blocks SC).

compared with those found in other collections. This happens because blocks with the same role in the CNN collection may have two or more versions of labels (root-to-block path), i.e., it is possible to exist two or more block classes formed by blocks with the same role in such collection. For instance, news pages found in the CNN collection usually contain tables, graphics, photos and other illustrative components in different parts of their text. However, such characteristic of the CNN collection does not significantly interfere in the quality of the weights obtained by block-based term weighting methods, since each block class continues to be formed by blocks with the same role.

On the other hand, notice that the number of block classes found in BLOGS collection is quite small, if compared with those found in other collections. Again, this

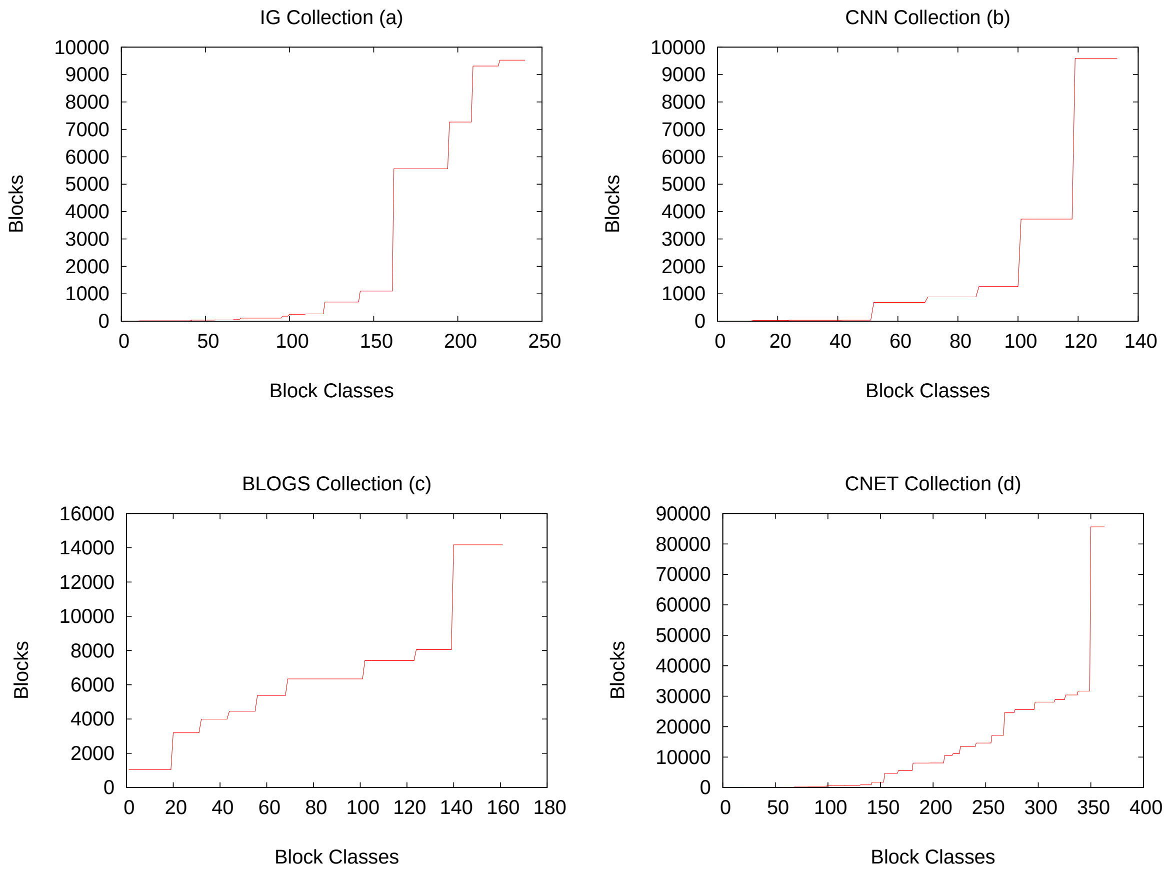


Figure 3.12. Number of blocks in each block classes found in four document collection.

happens because such collection is formed only by posts of their respective blogs, and the posts of a same blog tend to have the same block structure. Thus, since there is not a big variety of blocks from post to post, the number of block classes found in each blog of this collection tend to be quite small.

We can also see that the results of the automatic segmentation were far different from the semi-automatic method for CNET collection. When checking the results, we realized that this is due to irregularities in the page structures found in CNET which affects the automatic identification of recurrent structures (see Section 3.2.1). Such irregularities do not affect the semi-automatic method since a human can easily deal with them. For instance, each product page of the CNET Shopper has a list of virtual markets that sell that product. The semi-automatic method identified the list of each

Automatic segmentation approach				
Collection	**Site**	**B Classes**	**Blocks**	**Blocks SC**
IG	News	797	1,017,960	887
	Forum	256	366,202	212
	Recipe	42	20,350	10
	Total	**1,095**	**1,404,512**	**1,109**
CNN	News site	1,945	597,301	1,911
	Total	**1,945**	**597,301**	**1,911**
CNET	News	2,155	6,287,577	5,568
	Downloads	916	3,182,811	800
	Reviews	1,563	6,320,501	620
	Shopper	2,953	9,667,676	690
	Total	**7,587**	**25,458,565**	**7,678**
BLOGs	Boing Boing	59	421,074	9
	CNET	194	415,108	85
	Engadget	69	282,681	16
	Gizmodo	77	119,728	37
	Google	14	14,503	0
	Hacker	54	93,046	56
	Mashable	500	560,079	367
	Slash Film	213	145,151	283
	Tech Crunch	203	97,669	176
	Total	**1,383**	**2,149,039**	**1,029**

Table 3.4. The number of blocks and block classes found by our automatic segmentation approach in all Web sites of the four collections. Table also shows the number of blocks found in small classes, i.e., block classes with less than β elements (Blocks SC).

product page as a single block, and identified the set of lists of all product pages as a single block class. However, the set of tags used to display the virtual markets of a same list can be different to each other (for instance, if a virtual market was evaluated for one or more users, then it requires additional tags to be displayed). Since the tags of the lists of virtual markets are not disposed in a regular or recurrent way, each virtual market becomes a single block. Since this type of irregularity is very common in pages of the CNET collection, the number of blocks found by the automatic method is bigger then that found by the semi-automatic method.

An important point to observe in Table 3.4 is that in all collections just a few blocks where grouped into classes with less than β elements. For instance, from the total of 1, 404, 512 blocks found in IG, only 1, 109 were grouped into classes with less than β elements, which is less than 0.08% of the total number of blocks found. In CNN the number is less than 0.03%, in CNET less than 0.3%, and in BLOGS it is less than 0.05% of the blocks found.

Figure 3.13 shows the distribution of the number of elements on the block classes found in the four collections. Notice that, unlike the graphs presented in Figure 3.12, these distributions do not present big set of block classes with the same number of blocks, since the automatic segmentation approach does not work with the page cluster concept.

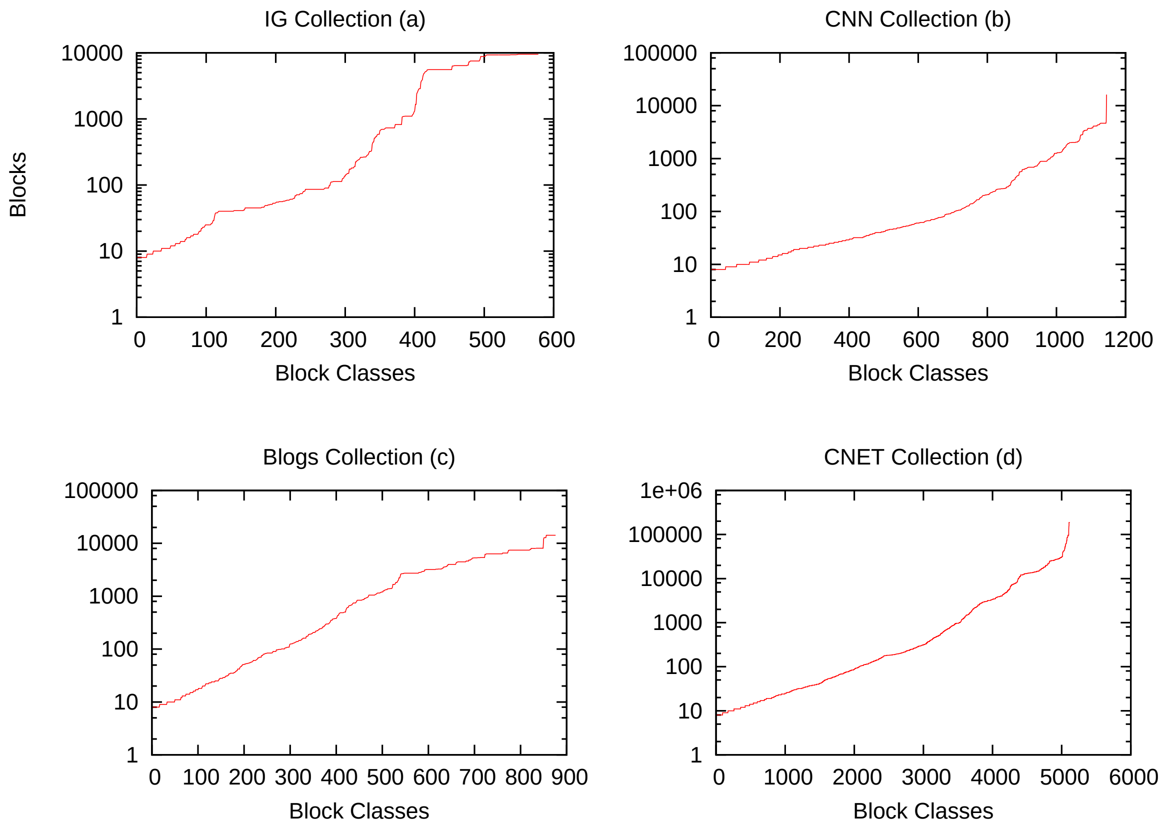
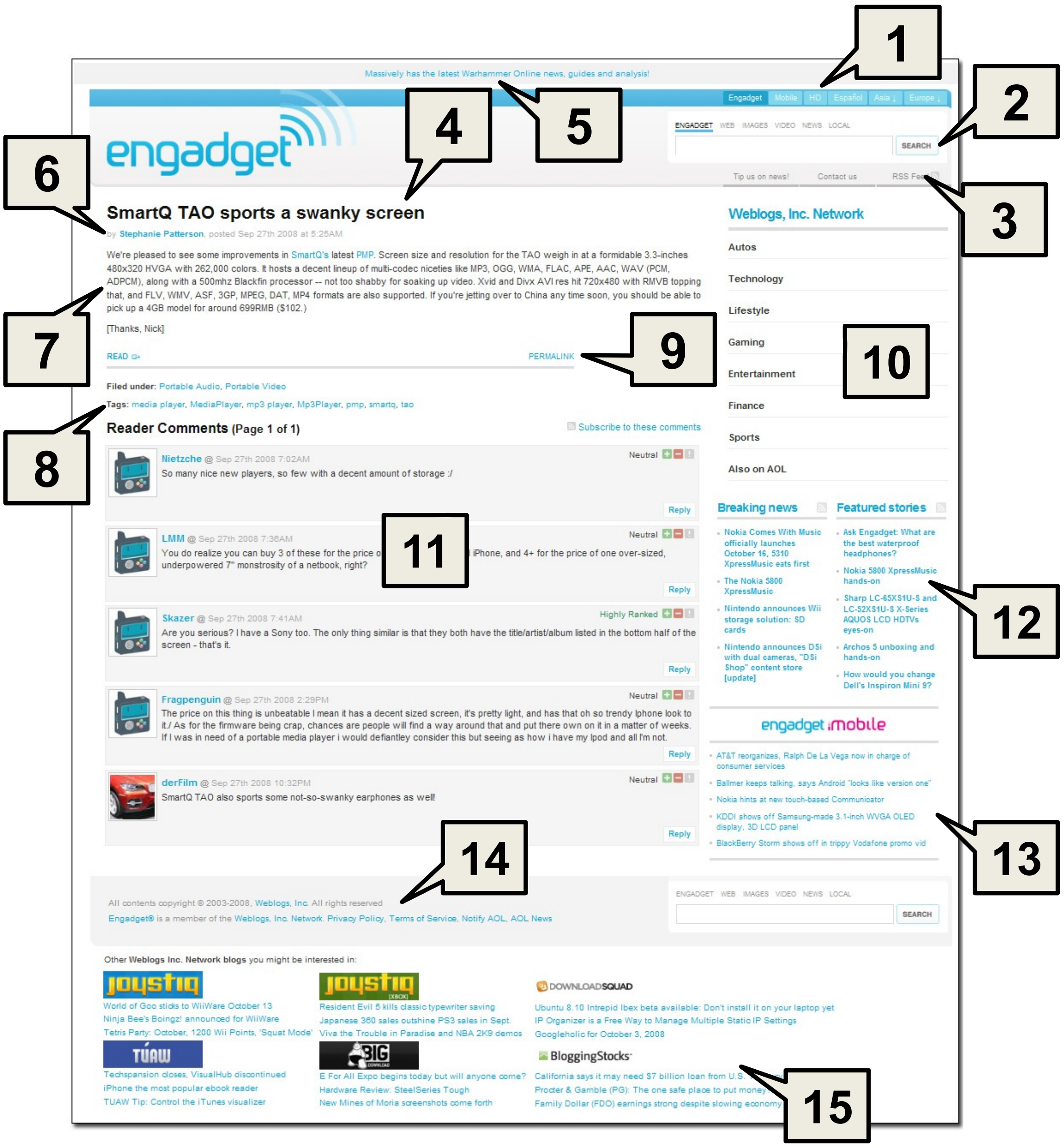


Figure 3.13. Number of blocks in each block classes found on in four document collection.

To provide examples of how the Web sites were segmented by the automatic approach, we present examples of blocks found in typical Web pages of the BLOG and the CNET collection. The screenshots in Figures 3.14 and 3.15 show a page extracted



Massively has the latest Warhammer Online news, guides and analysis!

SmartQ TAO sports a swanky screen

by Stephanie Patterson, posted Sep 27th 2008 at 5:25AM

We're pleased to see some improvements in SmartQ's latest PMP. Screen size and resolution for the TAO weigh in at a formidable 3.3-inches 480x320 HVGA with 262,000 colors. It hosts a decent lineup of multi-codec niceties like MP3, OGG, WMA, FLAC, APE, AAC, WAV (PCM, ADPCM), along with a 500mhz Blackfin processor -- not too shabby for soaking up video. Xvid and Divx AVI res hit 720x480 with RMVB topping that, and FLV, WMV, ASF, 3GP, MPEG, DAT, MP4 formats are also supported. If you're jetting over to China any time soon, you should be able to pick up a 4GB model for around 699RMB ($102.)

[Thanks, Nick]

Filed under: Portable Audio, Portable Video

Tags: media player, MediaPlayer, mp3 player, Mp3Player, pmp, smartq, tao

Reader Comments (Page 1 of 1) Subscribe to these comments

Nietzche @ Sep 27th 2008 7:02AM
So many nice new players, so few with a decent amount of storage :/

LMM @ Sep 27th 2008 7:36AM
You do realize you can buy 3 of these for the price o iPhone, and 4+ for the price of one over-sized, underpowered 7" monstrosity of a netbook, right?

Skazer @ Sep 27th 2008 7:41AM
Are you serious? I have a Sony too. The only thing similar is that they both have the title/artist/album listed in the bottom half of the screen - that's it.

Fragpenguin @ Sep 27th 2008 2:29PM
The price on this thing is unbeatable I mean it has a decent sized screen, it's pretty light, and has that oh so trendy Iphone look to it./ As for the firmware being crap, chances are people will find a way around that and put there own on it in a matter of weeks. If I was in need of a portable media player i would defiantley consider this but seeing as how i have my Ipod and all I'm not.

derFilm @ Sep 27th 2008 10:32PM
SmartQ TAO also sports some not-so-swanky earphones as well!

Weblogs, Inc. Network
Autos
Technology
Lifestyle
Gaming
Entertainment
Finance
Sports
Also on AOL

Breaking news
- Nokia Comes With Music officially launches October 16, 5310 XpressMusic eats first
- The Nokia 5800 XpressMusic
- Nintendo announces Wii storage solution: SD cards
- Nintendo announces DSi with dual cameras, "DSi Shop" content store [update]

Featured stories
- Ask Engadget: What are the best waterproof headphones?
- Nokia 5800 XpressMusic hands-on
- Sharp LC-65XS1U-S and LC-52XS1U-S X-Series AQUOS LCD HDTVs eyes-on
- Archos 5 unboxing and hands-on
- How would you change Dell's Inspiron Mini 9?

engadget mobile
- AT&T reorganizes, Ralph De La Vega now in charge of consumer services
- Ballmer keeps talking, says Android "looks like version one"
- Nokia hints at new touch-based Communicator
- KDDI shows off Samsung-made 3.1-inch WVGA OLED display, 3D LCD panel
- BlackBerry Storm shows off in trippy Vodafone promo vid

All contents copyright © 2003-2008, Weblogs, Inc. All rights reserved
Engadget® is a member of the Weblogs, Inc. Network. Privacy Policy, Terms of Service, Notify AOL, AOL News

Other Weblogs Inc. Network blogs you might be interested in:

joystiq
World of Goo sticks to WiiWare October 13
Ninja Bee's Boingz! announced for WiiWare
Tetris Party: October, 1200 Wii Points, 'Squat Mode'

joystiq [XBOX]
Resident Evil 5 kills classic typewriter saving
Japanese 360 sales outshine PS3 sales in Sept.
Viva the Trouble in Paradise and NBA 2K9 demos

DOWNLOADSQUAD
Ubuntu 8.10 Intrepid Ibex beta available: Don't install it on your laptop yet
IP Organizer is a Free Way to Manage Multiple Static IP Settings
Googleholic for October 3, 2008

TUAW
Techspansion closes, VisualHub discontinued
iPhone the most popular ebook reader
TUAW Tip: Control the iTunes visualizer

BIG DOWNLOAD
E For All Expo begins today but will anyone come?
Hardware Review: SteelSeries Tough
New Mines of Moria screenshots come forth

BloggingStocks
California says it may need $7 billion loan from U.S.
Procter & Gamble (PG): The one safe place to put money
Family Dollar (FDO) earnings strong despite slowing economy

Figure 3.14. A typical page present on the Engadget blog, from *BLOG* collection.

from the BLOG and the CNET collection, respectively. We numerated the blocks to facilitate their references through the text of this thesis.

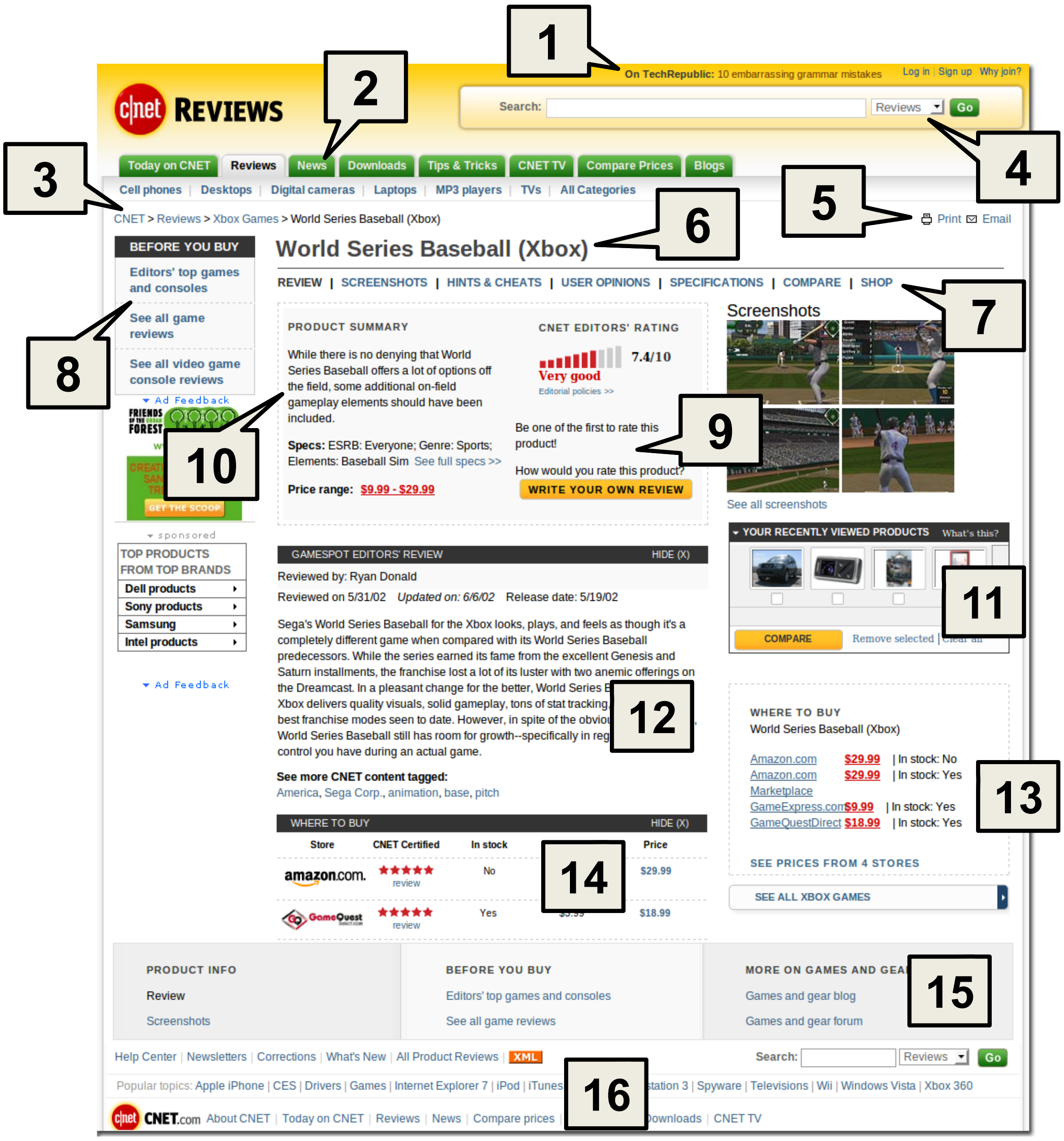
We performed some experiments to compare the quality of the segments found by our automatic segmentation algorithm with those found by the algorithm proposed by Kohlschütter and Nejdl [2008]. As described in Section 3.1.2, Kohlschütter and Nejdl [2008] approached the Web page segmentation problem from a textual perspective, and achieved a segmentation performance better than those achieved by Chakrabarti et al. [2008], which is another recent work of the literature.

To compare the quality of the segments, we used the statistic measures *Adjusted RAND Index* and *Average Mutual Information*, described in Section 3.3.1. By using these statistics, we evaluated the agreement between the collection of blocks identified by the semi-automatic approach, and the collection of blocks obtained by each automatic method we want to compare.

The results for Adjusted RAND Index is depicted on Figure 3.16. To compose each curve of the Figure (for instance, the curve SOM Tree in IG collection graph), we computed the Adjusted RAND index achieved by the method of the curve for each page of the collection, and then plotted these values in increasing order. For instance, considering the graphs for IG collection, the k_{th} page of the curve SOM Tree refers to a page segmented by our automatic segmentation method that achieved the k_{th} smaller Adjusted RAND index.

As one can see, these graphs show that our method achieved a segmentation performance better than those achieved by Kohlschütter and Nejdl [2008], in all collections. As expected, the CNET collection presented the smaller agreement between the semi-automatic and automatic approaches. However, even for this collection, our automatic approach presented a segmentation performance better than Kohlschütter and Nejdl [2008].

The results for Average Mutual Information is depicted on Figure 3.17. The conclusions are similar to those found by Adjusted RAND Index measure.



On TechRepublic: 10 embarrassing grammar mistakes | Log in | Sign up | Why join?

cnet REVIEWS

Search: Reviews Go

Today on CNET | Reviews | News | Downloads | Tips & Tricks | CNET TV | Compare Prices | Blogs

Cell phones | Desktops | Digital cameras | Laptops | MP3 players | TVs | All Categories

CNET > Reviews > Xbox Games > World Series Baseball (Xbox)

Print Email

BEFORE YOU BUY

Editors' top games and consoles

See all game reviews

See all video game console reviews

Ad Feedback

World Series Baseball (Xbox)

REVIEW | SCREENSHOTS | HINTS & CHEATS | USER OPINIONS | SPECIFICATIONS | COMPARE | SHOP

PRODUCT SUMMARY

While there is no denying that World Series Baseball offers a lot of options off the field, some additional on-field gameplay elements should have been included.

Specs: ESRB: Everyone; Genre: Sports; Elements: Baseball Sim See full specs >>

Price range: $9.99 - $29.99

CNET EDITORS' RATING

7.4/10

Very good

Editorial policies >>

Be one of the first to rate this product!

How would you rate this product?

WRITE YOUR OWN REVIEW

Screenshots

See all screenshots

sponsored

TOP PRODUCTS FROM TOP BRANDS

Dell products

Sony products

Samsung

Intel products

Ad Feedback

GAMESPOT EDITORS' REVIEW HIDE (X)

Reviewed by: Ryan Donald

Reviewed on 5/31/02 *Updated on: 6/6/02* Release date: 5/19/02

Sega's World Series Baseball for the Xbox looks, plays, and feels as though it's a completely different game when compared with its World Series Baseball predecessors. While the series earned its fame from the excellent Genesis and Saturn installments, the franchise lost a lot of its luster with two anemic offerings on the Dreamcast. In a pleasant change for the better, World Series B Xbox delivers quality visuals, solid gameplay, tons of stat tracking best franchise modes seen to date. However, in spite of the obviou World Series Baseball still has room for growth--specifically in reg control you have during an actual game.

See more CNET content tagged:

America, Sega Corp., animation, base, pitch

YOUR RECENTLY VIEWED PRODUCTS What's this?

COMPARE Remove selected | Clear all

WHERE TO BUY

World Series Baseball (Xbox)

Amazon.com	$29.99	In stock: No
Amazon.com	$29.99	In stock: Yes
Marketplace		
GameExpress.com	$9.99	In stock: Yes
GameQuestDirect	$18.99	In stock: Yes

SEE PRICES FROM 4 STORES

SEE ALL XBOX GAMES

WHERE TO BUY HIDE (X)

Store	CNET Certified	In stock		Price
amazon.com	★★★★★ review	No		$29.99
GameQuest	★★★★★ review	Yes	$9.99	$18.99

PRODUCT INFO

Review

Screenshots

BEFORE YOU BUY

Editors' top games and consoles

See all game reviews

MORE ON GAMES AND GEA

Games and gear blog

Games and gear forum

Help Center | Newsletters | Corrections | What's New | All Product Reviews | XML

Search: Reviews Go

Popular topics: Apple iPhone | CES | Drivers | Games | Internet Explorer 7 | iPod | iTunes tation 3 | Spyware | Televisions | Wii | Windows Vista | Xbox 360

cnet CNET.com About CNET | Today on CNET | Reviews | News | Compare prices ownloads | CNET TV

Figure 3.15. A typical page present on the reviews Web site, from *CNET* collection.

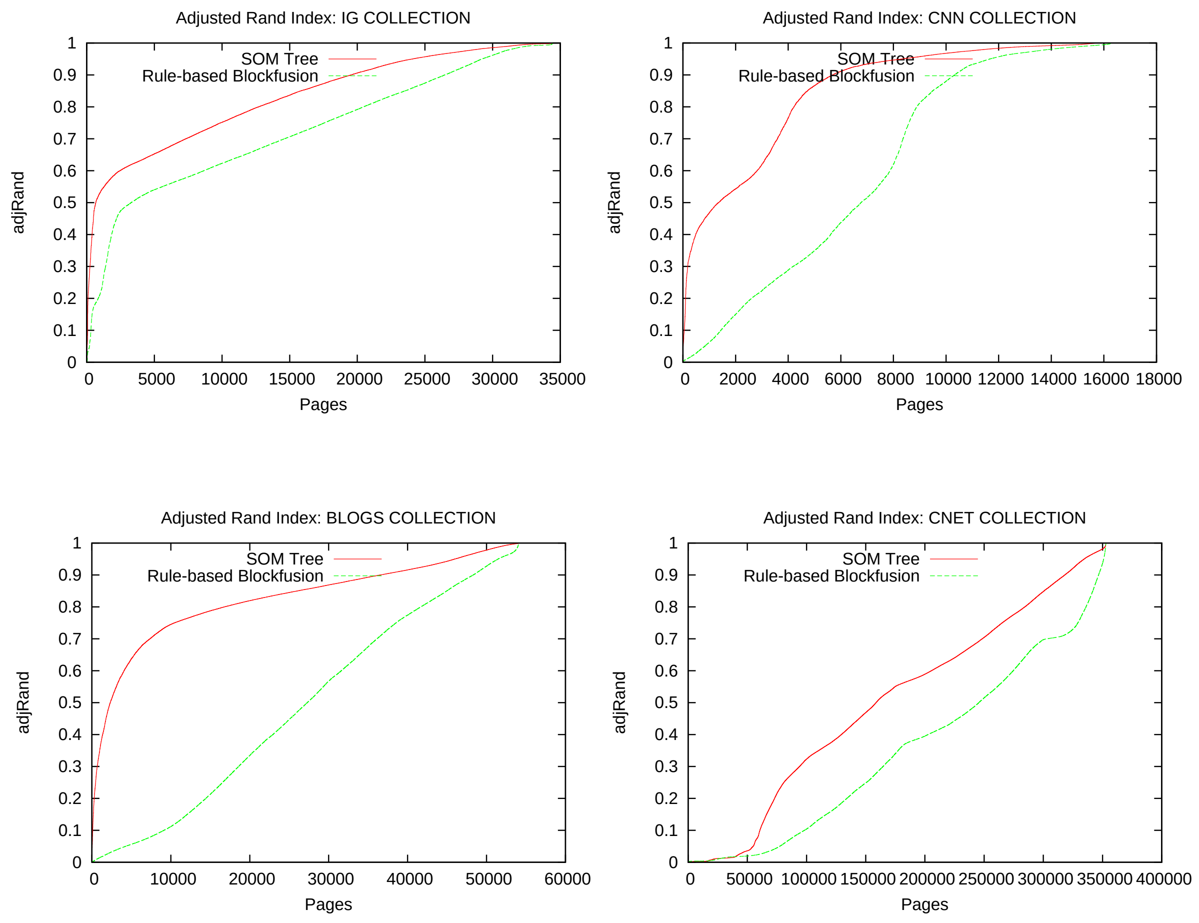


Figure 3.16. Adjusted RAND Index graphs found for IG, CNN, BLOGs and CNET collections.

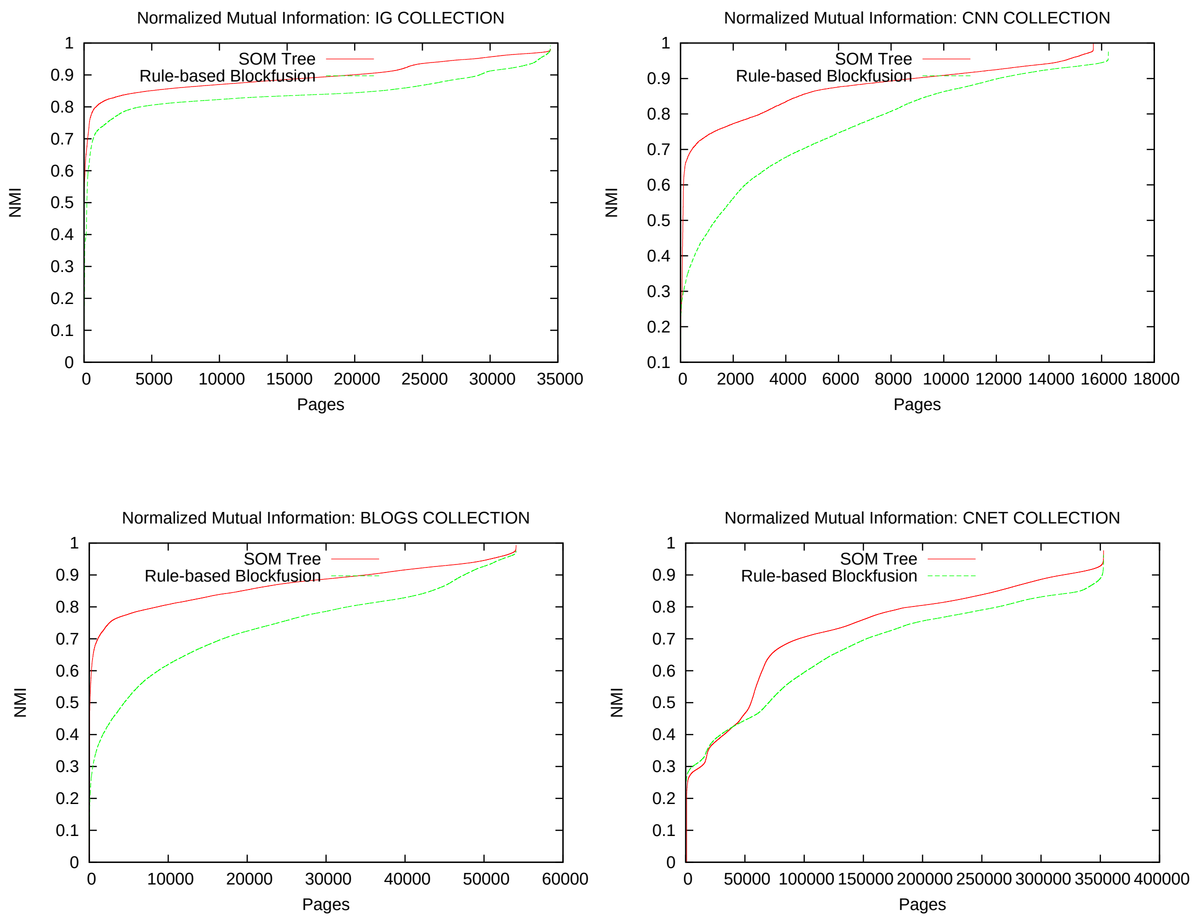


Figure 3.17. Average Mutual Information graphs found for IG, CNN, BLOGs and CNET collections.

Chapter 4

Information Retrieval and Web page Segmentation

Information retrieval models establish different paradigms to provide users with those documents that satisfy their information needs. It is important to understand the foundations of the principal models in order to develop a more accurate appreciation of their relative strengths and shortcomings. In this Chapter, we briefly discuss IR models and introduce the basis of the information retrieval problem we try to address in this thesis. For that, we quickly describe the concept of term weighting, provides an overview of two of the major textual retrieval models in the literature (Vector Space and BM25 models), and introduce the problem of using the *block structure* of Web pages to improve ranking results when searching for information on Web collections.

4.0.2 Term Weighting

Classical information retrieval models consider that each Web page or document is represented as a set of keywords called *index terms*. An index term is simply a pre-selected word which can be used to refer and describe the content of a document. Usually, index terms are nouns or noun groups. In the Web, however, some search engines use all the words in a document as index terms. Given a set of index terms, we notice that not all terms are equally useful for describing the document contents [Baeza-Yates and Ribeiro-Neto, 1999]. The importance of a term in a document is captured through the assignment of numerical weights.

Definition 5. *The weight of a term t in a document d_j is captured through the assignment of a numerical value $w_{t,j} \geq 0$ associated to the pair (t, d_j). If an index term t does not occurs in the document d_j, then $w_{t,j} = 0$.*

The weight (or importance) $w_{t,j}$ measures the ability of the index term t for describing the content of d_j document. The $w_{t,j}$ factors create a term-document matrix, that is one of the key pieces of evidence used for computing similarities between queries and documents. In the case of a collection of Web pages, other key pieces of evidence come from information provided by link analysis, such as the page rank [Brin and Page, 1998; Page et al., 1998] or authority of a Web page [Kleinberg, 1999]. However, when the collection is composed of Web pages of a pre-selected set of sites or domains, link-analysis is not of great help and the term weights become the central foundation for the ranking system [Fernandes et al., 2007].

4.0.3 The Vector Space Model

The vector space model [Sparck Jones, 1972, 1973, 1979] represents documents and queries as vectors in a T-dimensional Euclidean space, where T is the number of distinct index terms in the document collection.

$$\begin{aligned} \vec{d_j} &= (w_{1,j}, w_{2,j}, w_{3,j}, \ldots, w_{T,j}) \\ \vec{q} &= (w_{1,q}, w_{2,q}, w_{3,q}, \ldots, w_{T,q}) \end{aligned} \quad (4.1)$$

where $w_{t,j}$ is the weight of the term t in the document d_j, and $w_{t,q}$ the weight of t in the query q.

The term weights in the document vectors are given by two parameters: (i) $tf(t,j)$, computed as the number of times that the term t occurs in a document d_j, and (ii) $idf(t)$, that is a function of the number of documents where the term t occurs. Thus, the weight of an index term t in a document d_j is given by:

$$w_{t,j} = tf(t,j) \times idf(t) = tf(t,j) \times \log \frac{N}{n_t} \tag{4.2}$$

where N is the total number of documents in the collection, and n_t is the number of documents that contains the term t. In spite of the success of this term weighting formulation, it has a disadvantage in Web scenario, since it does not considers the locations of the term t within the page d_j to compute the $w_{t,j}$ factor [Fernandes et al., 2007].

The ranking of a document with regard to query is defined as the vector distance measure between their respective vector representations. This ranking is assumed to be correlated with the probability of relevance of the document. In practice, the distance measure is defined as the cosine of the angle between the vectors:

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{t=1}^{t} w_{t,j} \times w_{t,q}}{\sqrt{\sum_{t=1}^{t} w_{t,j}^2} \times \sqrt{\sum_{t=1}^{t} w_{t,q}^2}} \tag{4.3}$$

where $w_{t,q}$ corresponds to the weight of term t in query q, whose definition is equivalent to the weight of a term in a document. The factors $|\vec{d_j}|$ and $|\vec{q}|$ correspond to the norm of document and query vectors, respectively. The ranking calculation is not affected by $\vec{q}$ because its value is the same for all documents.

4.0.4 The BM25 Model

The BM25 (Best Match 25) is a model based on the probabilistic retrieval framework developed by Robertson et al. [1995], and is one of the most successful information retrieval models in literature. The BM25 weighting scheme is a function of the number of occurrences of the term in a document, in the whole collection, and a function of the document length. In the original BM25 method, the similarity ranking $sim(d_j, q)$ of a document d_j with regard to a user query q is given by

$$sim(d_j, q) = \sum_{t \in d_j \wedge t \in q} \frac{(k_1 + 1) \times tf(t, d_j)}{k_1 \left((1 - b) + b\frac{|d_j|}{\overline{d_j}}\right) + tf(t, d_j)} \times \log \frac{N - n_t + 0.5}{n_t + 0.5} \tag{4.4}$$

where $tf(t, d_j)$ is a function based on the occurrences of the term t in the document d_j, $|d_j|$ corresponds to a document length function, $\overline{d_j}$ is the average document length, N is the number of documents in the collection, and n_t is the number of documents containing the term t. Further, k_1 and b are free parameters, whose values can be fine-tuned experimentally to a particular collection.

4.0.5 Discussion

The tf (term frequency) factor used by the Vector Space and BM25 model consider that all occurrences of a term in a page are equally useful for estimating the importance of the term in the page. On the other side, there are *types of blocks*, such as menus, advertisements, and blocks containing information not associated with the main *topics* or *themes* of a Web page, whose contents should be considered of less importance in the ranking formula. This is one piece of information that we can factor into the ranking by applying the approach we propose in this thesis.

4.1 Using Block Information to Improve Ranking Results

Several authors have presented methods that rely on using block information to improve results effectiveness of information retrieval systems. We select some of these works to present in this section.

4.1.1 Block-based Web Search

In Cai et al. [2004b], the authors use block information to investigate how to take advantage of block-level evidence to improve retrieval performance in the Web context. According to the authors, the content of a Web page is usually much more diverse compared with traditional plain text document and encompasses multiple regions with unrelated topics. Moreover, for the purpose of browsing and publication, non-content materials, such as navigation bars, decoration stuffs, interaction forms, copyrights, and contact information, are usually embedded in Web pages. Instead of treating a whole Web page as an unit of retrieval, the paper argues that such characteristics of Web pages make the block a more effective mechanism for information retrieval.

The major shortcoming of treating a Web page as a single semantic unit is that it does not consider multiple topics in a page [Cai et al., 2004b]. For example, if the query terms occurs at regions with different topics, it could cause low retrieval precision. So, the authors argue that a Web page with a region of high density of matched terms is likely to be more relevant than a Web page with matched terms distributed across the entire page even if it has higher overall similarity. On the other hand, a highly relevant region in a Web page may be obscured because of low overall relevance of that page.

The authors present a ranking strategy where the blocks are processed as semantic passages and then apply previously proposed passage level ranking strategies [Callan, 1994] to evaluate the possible improvements obtained in ranking quality with their method. Thus, the authors propose to compute two scores for each document of a collection: one for the whole document, and another considering the blocks as units. The two scores are then combined by using the function:

$$score(d_j, q) = \alpha \; rank_{doc}(d_j, q) + (1 - \alpha) \; rank_{bestblock}(d_j, q) \qquad (4.5)$$

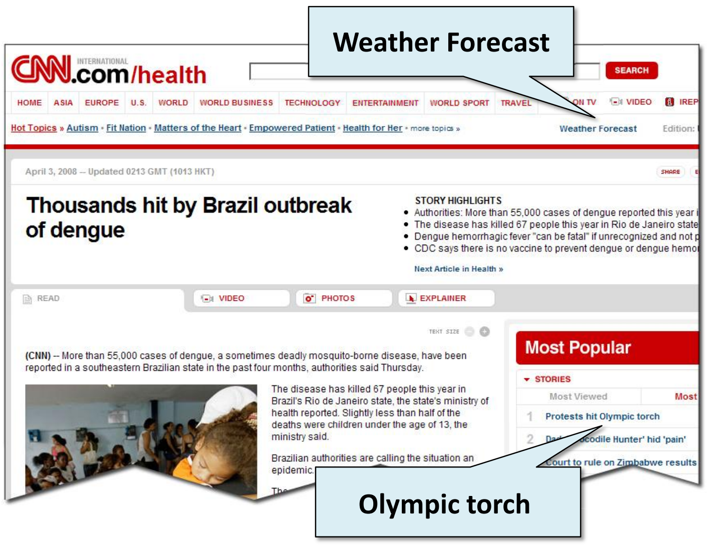
where $rank_{doc}$ is the rank given by BM25 to the document d_j for query q and $rank_{bestblock}(d_j, q)$ is the rank given by BM25 to the best ranked block of d_j when considering blocks as retrieval units. The parameter α should be adjusted through a training process, which is a disadvantage of this approach when compared to our method.

Cai et al. [2004b] presents a interesting method of using block information in the ranking strategy, and we adopted it as one of the baselines in our experiments. However, a shortcoming regarding this ranking strategy is that it do not distinguish noise from informative blocks. Thus, if the query terms occurs at the templates of the pages, it could cause low retrieval precision. This shortcoming is illustred in Figure (see figure 4.1).

4.1.2 Block-level Link Analysis

Another interesting work based on the block concept is Cai et al. [2004a]. The authors of this paper proposed two novel link analysis algorithms called *Block Level PageRank* (BLPR) and *Block Level HITS* (BLHITS) which treat blocks as information units. These novel algorithms of link analysis are also based in the assumption that a single Web page often contains multiple semantics and the different parts of the Web page have different importance in that page. Therefore, links in high important blocks should be more important than those in low important blocks. In other words, a user might prefer to follow links in important blocks.

To compute the block importance values, Cai et al. [2004a] uses the assumption


Weather Forecast
INTERNATIONAL
CNN.com/health
SEARCH
HOME ASIA EUROPE U.S. WORLD WORLD BUSINESS TECHNOLOGY ENTERTAINMENT WORLD SPORT TRAVEL ON TV VIDEO IREP
Hot Topics » Autism • Fit Nation • Matters of the Heart • Empowered Patient • Health for Her • more topics »
Weather Forecast
Edition:
SHARE
READ
VIDEO
PHOTOS
EXPLAINER
TEXT SIZE
Most Popular
STORIES
Most Viewed
Most
Olympic torch

April 3, 2008 -- Updated 0213 GMT (1013 HKT)

Thousands hit by Brazil outbreak of dengue

STORY HIGHLIGHTS
- Authorities: More than 55,000 cases of dengue reported this year i
- The disease has killed 67 people this year in Rio de Janeiro state
- Dengue hemorrhagic fever "can be fatal" if unrecognized and not p
- CDC says there is no vaccine to prevent dengue or dengue hemo

Next Article in Health »

(CNN) -- More than 55,000 cases of dengue, a sometimes deadly mosquito-borne disease, have been reported in a southeastern Brazilian state in the past four months, authorities said Thursday.

The disease has killed 67 people this year in Brazil's Rio de Janeiro state, the state's ministry of health reported. Slightly less than half of the deaths were children under the age of 13, the ministry said.

Brazilian authorities are calling the situation an epidemic.

The

1 Protests hit Olympic torch
2 Da ocodile Hunter' hid 'pain'
Court to rule on Zimbabwe results

Figure 4.1. Document that would have high similarity with queries "Weather forecast" and "Olympic torch" when applying the method presented by Cai et al. [2004b]

that some blocks with big size and centered position are probably more important than those blocks with small size and margin position. However, the authors admit that incorporating more advanced block importance models, such as the schemes described in Song et al. [2004a,b], they expect that a better result might be achieved.

4.1.3 Improving Pseudo-Relevance Feedback Using Block Information

Pseudo-relevance feedback, also known as local feedback or blind feedback, is a technique commonly used to improve retrieval effectiveness [Efthimiadis, 1996; Yu et al., 2003]. Its basic idea is to extract expansion terms from the top-ranked documents to formulate a new query for a second retrieval of the documents. Through a query expansion, some relevant documents missed in the initial round can then be retrieved to improve the overall performance.

The effect of pseudo-relevance feedback strongly depends on the quality of selected expansion terms. If the words added to the original query are unrelated to the query, the quality of the retrieval is likely to be degraded. However, a typical Web page contains various types of materials that are not related to the topic of the Web-page, such as navigation blocks, decoration blocks and interaction blocks [Yu et al., 2003].

Therefore, Yu et al. [2003] suggests combine page segmentation methods with pseudo-relevance feedback algorithm according to the following steps:

Phase 1: Initial Retrieval

An initial list of ranked Web pages is obtained by using any traditional information retrieval methods.

Phase 2: Page Segmentation

In this step, a page segmentation method (Yu et al. [2003] used VIPS algorithm) is applied to divide retrieved Web pages into segments. Since it is very expensive to process all retrieved Web pages, the authors suggest select a few (e.g. 80) top pages for segmentation. The candidate segment set is made up of these resulting segments.

Phase 3: Segment Selection

This step aims to choose most relevant segments from the candidate segment set. Some ranking methods are used to sort the candidate segments and the top (e.g. 20) segments are selected for expansion term selection in the next step.

Phase 4: Expansion Term Selection

Using an approach similar to the traditional pseudo-relevance feedback algorithm to select expansion terms. The difference is that expansion terms are selected from the selected segments instead of from the whole Web pages.

The experimental results demonstrated that by partitioning a Web page into segments, better query expansion terms can be selected to improve the overall retrieval performance.

4.1.4 Learning Block Importance Models for Web Pages

In Song et al. [2004a,b] the authors use VIPS algorithm to identify the blocks of a Web page, and then compute an importance rank of the blocks through a learning approach. Spatial features, such as position and size of blocks, and content features, such as the number of images and links on each block, are extracted from each block. Then, two learning algorithms, neural network and SVM, are used to learn about what features can be used to differentiate the important parts from unimportant parts of a Web page.

According to the authors, Web designers would like to put the most important information in the center, put the navigation bar on the header or the left side and the copyright on the footer. Thus, the importance of a block would be a reflection of the spatial features like position, size, etc. The authors used 4 spatial features to judge the importance of the blocks:

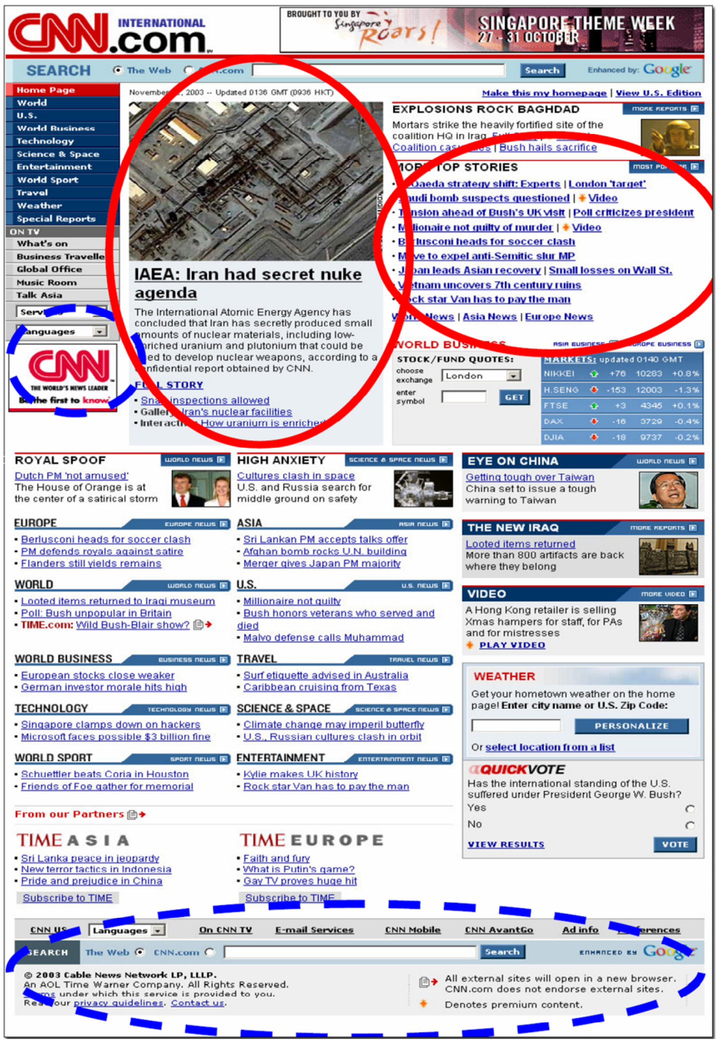
{BlockCenterX, BlockCenterY, BlockRectWidth, BlockRectHeight}

where *BlockCenterX* and *BlockCenterY* are the coordinates of the center point of the block and *BlockRectWidth*, *BlockRectHeight* are the width and height of the block.

The authors also argue that the contents of the blocks would be also useful to judge their importance. For example, the spatial features of both of the two solid circles in Figure 4.2 are similar. But one contains a picture, a highlighted title and some words to describe a news headline and another contains pure hyperlinks pointing to other top stories. Based on the contents of the blocks, would be possible to differentiate their importance. The following 9 features are used by Song et al. [2004a] to represent the content of a block:

{ImgNum, ImgSize, LinkNum, LinkTextLength, InnerTextLength,
InteractionNum, InteractionSize, FormNum, FormSize}

where *ImgNum* and *ImgSize* are the number and size of images contained in the block. *LinkNu* and *LinkTextLength* are the number of hyperlinks and anchor text length of the


CNN.com INTERNATIONAL

BROUGHT TO YOU BY Singapore Roars! SINGAPORE THEME WEEK 27 - 31 OCTOBER

SEARCH The Web .com Search Enhanced by: Google

Home Page
World
U.S.
World Business
Technology
Science & Space
Entertainment
World Sport
Travel
Weather
Special Reports
ON TV
What's on
Business Travelle
Global Office
Music Room
Talk Asia
Serv
anguages

CNN THE WORLD'S NEWS LEADER the first to know

November , 2003 -- Updated 0136 GMT (0936 HKT)

Make this my homepage | View U.S. Edition

IAEA: Iran had secret nuke agenda

The International Atomic Energy Agency has concluded that Iran has secretly produced small mounts of nuclear materials, including low-riched uranium and plutonium that could be ed to develop nuclear weapons, according to a nfidential report obtained by CNN.

L STORY
- Sna inspections allowed
- Galler Iran's nuclear facilities
- Interact How uranium is enriche

EXPLOSIONS ROCK BAGHDAD — MORE REPORTS

Mortars strike the heavily fortified site of the coalition HQ in Iraq
Coalition cas es | Bush hails sacrifice

MOR TOP STORIES — MOST PO
- Qaeda strategy shift: Experts | London 'target'
- udi bomb suspects questioned | Video
- T nsion ahead of Bush's UK visit | Poll criticizes president
- M lionaire not guilty of murder | Video
- B lusconi heads for soccer clash
- M ve to expel anti-Semitic slur MP
- J pan leads Asian recovery | Small losses on Wall St.
- V etnam uncovers 7th century ruins
- ck star Van has to pay the man

W News | Asia News | Europe News

WORLD BUS — ASIA BUSINESS EUROPE BUSINESS

STOCK/FUND QUOTES:
choose exchange London
enter symbol GET

MARKETS: updated 0140 GMT				
NIKKEI	⬆	+76	10283	+0.8%
H.SENG	⬇	-153	12003	-1.3%
FTSE	⬆	+3	4345	+0.1%
DAX	⬇	-16	3729	-0.4%
DJIA	⬇	-18	9737	-0.2%

ROYAL SPOOF — WORLD NEWS
Dutch PM 'not amused'
The House of Orange is at the center of a satirical storm

HIGH ANXIETY — SCIENCE & SPACE NEWS
Cultures clash in space
U.S. and Russia search for middle ground on safety

EYE ON CHINA — WORLD NEWS
Getting tough over Taiwan
China set to issue a tough warning to Taiwan

EUROPE — EUROPE NEWS
- Berlusconi heads for soccer clash
- PM defends royals against satire
- Flanders still yields remains

ASIA — ASIA NEWS
- Sri Lankan PM accepts talks offer
- Afghan bomb rocks U.N. building
- Merger gives Japan PM majority

THE NEW IRAQ — MORE REPORTS
Looted items returned
More than 800 artifacts are back where they belong

WORLD — WORLD NEWS
- Looted items returned to Iraqi museum
- Poll: Bush unpopular in Britain
- TIME.com: Wild Bush-Blair show?

U.S. — U.S. NEWS
- Millionaire not guilty
- Bush honors veterans who served and died
- Malvo defense calls Muhammad

VIDEO — MORE VIDEO
A Hong Kong retailer is selling Xmas hampers for staff, for PAs and for mistresses
PLAY VIDEO

WORLD BUSINESS — BUSINESS NEWS
- European stocks close weaker
- German investor morale hits high

TRAVEL — TRAVEL NEWS
- Surf etiquette advised in Australia
- Caribbean cruising from Texas

TECHNOLOGY — TECHNOLOGY NEWS
- Singapore clamps down on hackers
- Microsoft faces possible $3 billion fine

SCIENCE & SPACE — SCIENCE & SPACE NEWS
- Climate change may imperil butterfly
- U.S., Russian cultures clash in orbit

WEATHER
Get your hometown weather on the home page! **Enter city name or U.S. Zip Code:**
PERSONALIZE
Or select location from a list

WORLD SPORT — SPORT NEWS
- Schuettler beats Coria in Houston
- Friends of Foe gather for memorial

ENTERTAINMENT — ENTERTAINMENT NEWS
- Kylie makes UK history
- Rock star Van has to pay the man

QUICKVOTE
Has the international standing of the U.S. suffered under President George W. Bush?
Yes
No
VIEW RESULTS VOTE

From our Partners

TIME ASIA
- Sri Lanka peace in jeopardy
- New terror tactics in Indonesia
- Pride and prejudice in China

Subscribe to TIME

TIME EUROPE
- Faith and fury
- What is Putin's game?
- Gay TV proves huge hit

Subscribe to TIME

CNN US Languages On CNN TV E-mail Services CNN Mobile CNN AvantGo Ad info erences

EARCH The Web CNN.com Search ENHANCED BY Goog

© 2003 Cable News Network LP, LLLP.
An AOL Time Warner Company. All Rights Reserved.
ms under which this service is provided to you.
Rea our privacy guidelines. Contact us.

All external sites will open in a new browser. CNN.com does not endorse external sites.
Denotes premium content.

Figure 4.2. Spatial and content features can be used to differentiate the importance of blocks

block. *InnerTextLength* is the number of words. *InteractionNum*, and *InteractionSize* are the number and size of elements with the tags of <INPUT> and <SELECT>. *FormNum* and *FormSize* are the number and size of element with the tag <FORM>. According to the authors, all of these content features can be related to the importance of the blocks.

These content features were also normalized by the feature values of the whole page. For example, the *LinkNum* of a block is normalized by the link number of the whole page.

Therefore, Song et al. [2004a,b] suggests that the importance of a block is a function of its features, and can be formalized as:

$$< \text{block features} > \rightarrow \text{block importance} \tag{4.6}$$

The learning method was employed for labelling the blocks with the following 4-level importance values:

- Level 1: noisy information such as advertisement, copyright, decoration, etc.
- Level 2: useful information, but not very relevant to the topic of the page, such as navigation, directory, etc.
- Level 3: relevant information to the theme of the page, but not with prominent importance, such as related topics, topic index, etc.
- Level 4: the most prominent part of the page, such as headlines, main content, etc.

In Chapter 5 we propose some methods of assigning an importance degree to each block, instead of just ranking the blocks according with fixed importance labels. Further, our proposed methods can also be used to also rank blocks if necessary, with the advantage of not requiring a training phase.

4.1.5 Template Removal

Several authors have proposed methods for using the Web pages structure for identifying blocks that contain little or no information to offer readers, and that are then considered as noise or irrelevant content [Yi et al., 2003; Vieira et al., 2006; Yin and Lee, 2004; Bar-Yossef and Rajagopalan, 2002; Fu et al., 2007]. To illustrate, Bar-Yossef and Rajagopalan [2002] proposed a mechanism for segmenting Web pages into blocks, represented there as Pagelets, and then classifying them into useful or noise content. In their work, authors considered any information present on templates

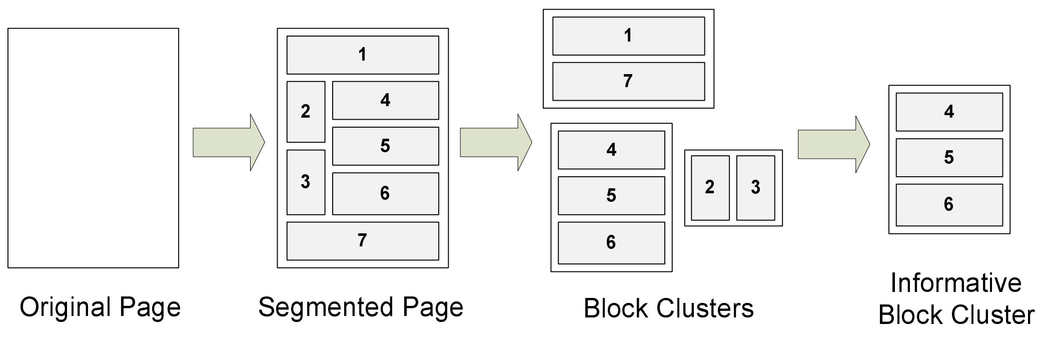


Figure 4.3. RIPB's informative block cluster recognition process

of the pages as noise and proposed to disregard such an information when processing the content of Web pages. Fu et al. [2007] also proposed a method to discover template content without supervision. The basic assumption is that Web pages in one site are always generated by a common template or layout. Once the templates occurring in a set of sample pages are summarized, it is possible discover the template occurring in the other pages of the site.

Since the removal of templates can be considered as an structural information, we included a search system that does not index template contents as one of the baselines in our ranking experiments.

4.1.6 Information Extraction

One of the main problems in most automatized information extraction methods is the difficulty in discriminating the blocks that contain the most important information on a page (informative blocks), from the noise blocks, that contains irrelevant information such as advertisements, menus, or copyright statements [Debnath et al., 2005; Kang and Choi, 2007].

To solve this problem, Kang and Choi [2007] proposes the RIPB (*Recognizing Informative Page Blocks*) algorithm that detects the informative blocks in a Web page by exploiting the visual block segmentation scheme. The informative block recognition process of the RIPB method is shown in the figure 4.3. Such a method uses VIPS

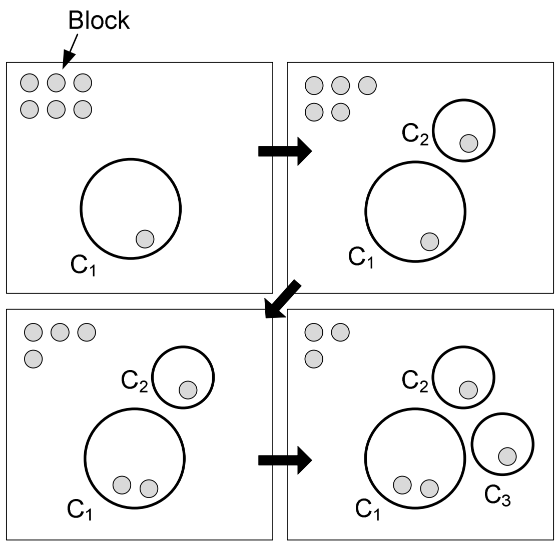


Figure 4.4. The process of block clustering of RIPB

[Cai et al., 2003] to segment the Web pages, and then groups related blocks with similar structures into a block cluster by using an edit distance method. The clustering process is shown in Figure 4.4.

After creating the clusters, RIPB recognizes the clusters with informative blocks. Such clusters contain the blocks with important information that would be the target of information extraction. The main idea of RIPB is to recognize the informative block clusters by measuring the number of tokens and the cluster area size. This measure is made by the equation below:

$$Score(C) = (1 - \beta) \cdot |C_i| + \beta \cdot img_i \tag{4.7}$$

where C_i is a cluster of blocks, $|C_i|$ is the number of tokens on cluster C_i, img_i is the total area of the cluster C_i, and β is the weight value for balancing the importance between the number of tokens and the cluster area.

Gibson et al. [2007] address the problem of identifying the portions of news-source Web-pages (blocks) that contain relevant content i.e. the news article itself. The approach of this work is to divide the original Web document into a sequence blocks (the authors developed their own method of segmentation) and then categorize each block as **Content** or **NotContent**. Given this formulation as a sequence labelling problem, three different statistical machine learning methods was employed for labelling the blocks: Conditional Random Fields (CRF), Maximum Entropy Classifiers (MaxEnt), and Maximum Entropy Markov Models (MEMM).

Gibson et al. [2007] used 11 different feature types to make the classification:

1. Words. This is a set of features, one for each token that appears in the block. The value for each feature for a given block instance is the number of times that token occurs in the block.

2. Inverse Stop-Wording. Number of tokens of a given block that occur in a list of English stop words. This feature can help to identify that English prose was being used in a block without becoming too dependent upon the particular vocabulary of an article.

3. Named Entities. A count of the named entities in a block, grouped by the following types: person, organization, location, date, time, monies, and percentages.

4. Title Casing. Features are generated when every token in a block begins with an upper case letter or when any token begins with a lower case letter.

5. Anchor Percentage. The percentage of tokens in a block contained within an anchor tag.

6. Title Matching. If some tokens in a block match with the tokens contained in the `<title>` of the document then the percentage of the title that matched is recorded.

7. Ancestor Tags. The names of the parent and grandparent (if present) tags of a block.

8. Descendant Tags. The names any descendant tags found within a block.

9. Sibling Tags. The names of the previous and next sibling tags of the current block (if present).

10. Word Count. A single feature capturing the count of the tokens found in a block.

11. After Image Tag. A feature that indicates if an <img> appears before the current block and after the previous one. This feature is intended to exclude photo captions from inclusion in the content.

In Li et al. [2007] is presented a visual segmentation-based data record extraction (VSDR) method to extract data records from Web pages. Given a Web page, the VSDR method consists of following steps: (i) segment the page into blocks (using VIPS); (ii) remove the noisy blocks such as navigational bar blocks, dropdown menu blocks, etc.; and (iii) in the remaining blocks, identify the data records.

The VSDR applies some heuristics to identify noisy blocks. For instance, a block is considered as a noisy content if the number of contiguous link blocks is at least 5 and the ratio of the number of link blocks (including both text links and image links) to the number of all blocks in the same level in DOM tree is greater than a specific threshold. These noisy tree are removed before extracting data records.

Similarly, the VSDR method considers that blocks containing text boxes, dropdown menus, and/or action buttons, and occupying a relatively small area of the whole page, are noise blocks.

Chapter 5

Block-Weight Functions

In Web IR systems, the content of a Web page is usually represented as a collection of terms derived or inferred from the text of the page. For ranking purposes, these terms are usually weighted to indicate their importance within each page, such that the effectiveness of any IR system is directly related to the accuracy of these term weights.

In this context, the block structure of Web pages may be used to alter the overall weight of the term in a page. To illustrate, the occurrence of a term in the block *title* of a Web page, as the one depicted in Figure 5.1, can be made more important than an occurrence of the same term in the *menu* of that page. To quantify the importance of a term in a block we define a *block-weight function*, which computes the *weight* of the term occurrence in the block, as follows:

Definition 6. *The block-weight function $bw(t, b)$ is a quantitative metric associated with the term-block pair $[t, b]$ which we use to compute a weight for term t relative to the page that contains block b.*

To compute a block-weight function $bw(t, b)$, we introduce two basic statistical measures, the *inverse class frequency* and the *spread*, as follows.

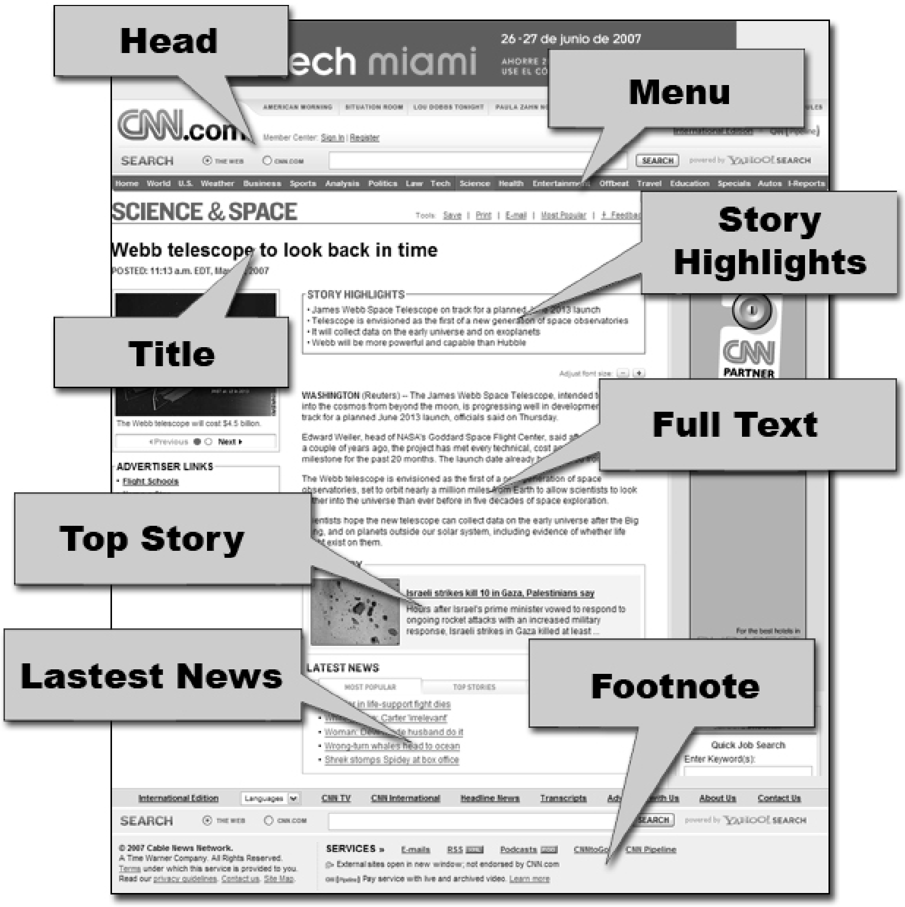


Figure 5.1. A news page and its blocks.

5.1 Inverse Class Frequency and Spread

These two measures are variants of the *inverse document frequency* (IDF) and *term frequency* (TF) factors used in the Vector Space Model [Salton et al., 1975], as follows.

Definition 7. *Given a block class* $\mathcal{C} = \{b_1, ..., b_{n_\mathcal{C}}\}$*, containing* $n_\mathcal{C}$ *elements, and a term* t *that occurs in at least one block of* $\mathcal{C}$*, the* Inverse Class Frequency *(ICF) of a term* t *in* $\mathcal{C}$ *is defined as*

$$ICF(t, \mathcal{C}) = \log \frac{n_\mathcal{C}}{n_{t,\mathcal{C}}} \tag{5.1}$$

where $n_{t,\mathcal{C}}$ is the number of blocks of $\mathcal{C}$ in which t occurs.

Notice that ICF is similar to IDF, but considers each block class as a separate "collection of documents". Using Shannon's entropy [Shannon, 1948], it quantifies how much information is associated with the occurrence of a given term in a given block class. It can also be interpreted as a measure of how well the term discriminates a block from other blocks. To illustrate, let us consider the block *menu* of the Web page depicted in Figure 5.1. Since this block is derived from the template of the CNN Web site, it includes the term "politics", which also occurs in many other blocks of the block class *Menu*. In this case, the term "politics" will have low ICF value in the block class *Menu*, which indicates that the this occurrence of the term is not very discriminative.

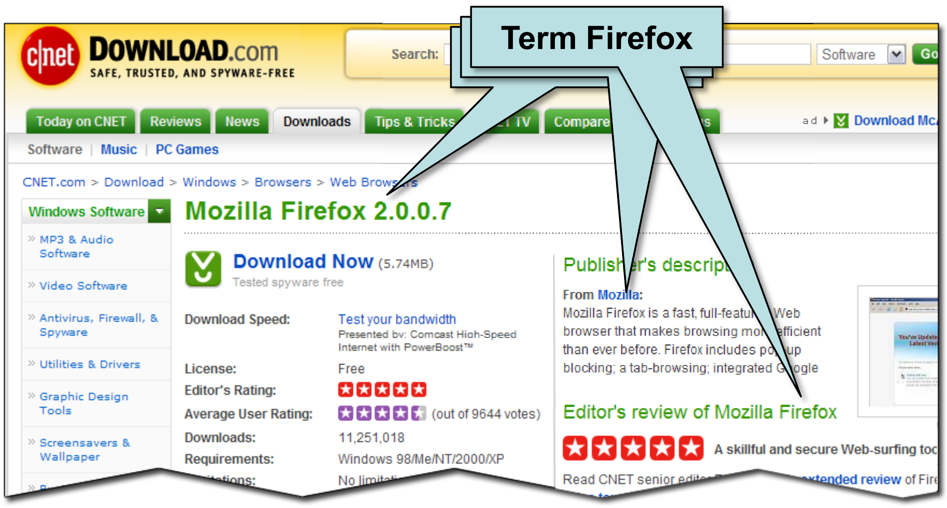
As a counterpoint, let us consider the term "telescope", which occurs in the *title* of the Web page depicted in Figure 5.1. Since this term occurs infrequently in page titles, it will have a high ICF value. We say that the occurrence of the term "telescope" in a page title is a very discriminative event.

Definition 8. *The* Spread *of a term t in a Web page ρ_n, $Spread(t, \rho_n)$, is the number of blocks in ρ_n that contain t, that is:*

$$Spread(t, \rho_n) = \sum_{b \in \rho_n} i, \ \ where\ i = \begin{cases} 1 & if\ t \in b \\ 0 & otherwise \end{cases} \tag{5.2}$$

The intuition behind this measure is that, given a term t and a Web page ρ_n, the greater the number of blocks of ρ_n that contain the term t, the better the term t represents the contents of the Web page. For instance, if the term "Firefox" appears in most of the blocks of a given page, there is a high chance that this page is related to the famous browser (this effect can be seen in Figure 5.2). The Spread can be seen as the structural frequency of the term, acting as an structural version of the TF factor adopted in the Vector Space Model.

In the immediate following, we propose different strategies to compute block-weight factors *bw* using the definitions of ICF and *Spread*. These strategies are grouped into three categories: *class-level strategies*, which assign a unique *bw* value to all terms



DOWNLOAD.com SAFE, TRUSTED, AND SPYWARE-FREE

Search: Software Go

Today on CNET | Reviews | News | Downloads | Tips & Tricks | Compare

Software | Music | PC Games

CNET.com > Download > Windows > Browsers > Web Browsers

Windows Software

» MP3 & Audio Software
» Video Software
» Antivirus, Firewall, & Spyware
» Utilities & Drivers
» Graphic Design Tools
» Screensavers & Wallpaper

Mozilla Firefox 2.0.0.7

Download Now (5.74MB)
Tested spyware free

Download Speed: Test your bandwidth
Presented by: Comcast High-Speed Internet with PowerBoost™
License: Free
Editor's Rating: ★★★★★
Average User Rating: ★★★★☆ (out of 9644 votes)
Downloads: 11,251,018
Requirements: Windows 98/Me/NT/2000/XP

Publisher's descrip

From Mozilla:
Mozilla Firefox is a fast, full-featu Web browser that makes browsing mo efficient than ever before. Firefox includes po up blocking; a tab-browsing; integrated G ogle

Editor's review of Mozilla Firefox
★★★★★ A skillful and secure Web-surfing too
Read CNET senior edito xtended review of Fire

Figure 5.2. A page where the term Firefox is spread in its blocks.

in a block class; *block-level strategies*, which assign a unique *bw* value to all terms in a block; and *term-level strategies*, which allow the *bw* values to vary for distinct terms inside a block.

5.2 Term-level Strategies

The term-level strategies use the *ICF* and *Spread* factors to compute *bw* weights as follows. Given a term t and a block b, a first form to compute the *bw* factor is

$$bw_1(t,b) = ICF(t,\mathcal{C}_b) \tag{5.3}$$

where $\mathcal{C}_b$ is the block class of block b. A second form to compute the *bw* factor is

$$bw_2(t,b) = Spread(t,\rho_b) \tag{5.4}$$

where ρ_b is the Web page that b belongs to. Further, we also explore a third form of the *bw* factor that combines both *spread* and *ICF* as follows:

$$bw_3(t,b) = ICF(t,\mathcal{C}_b), \times Spread(t,\rho_b) \tag{5.5}$$

5.3 Block-Level Strategies

The idea here is to compute an average *bw* value for all terms in a block, one that allows smoothing the impact of anomalies that may arise when looking just for a single term. For instance, a term t that appears in a news title, as in Figure 5.1, is important even if it is not mentioned in the body of the news. However, the spread of such term across all blocks in the page would be low. This effect can be avoided by using an average spread value for all terms in the block.

The term-level strategies we introduced previously can now be used to create equivalent block-level methods, as follows.

$$bw_4(t, b) = \begin{cases} \frac{\sum_{t' \in b} ICF(t', \mathcal{C}_b)}{|b|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

where $|b|$ is the size of block b. Also,

$$bw_5(t, b) = \begin{cases} \frac{\sum_{t' \in b} Spread(t', \rho_b)}{|b|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

We further combine these two block-level strategies to obtain

$$bw_6(t, b) = \begin{cases} \frac{\sum_{t' \in b} Spread(t', \rho_b) \times ICF(t', \mathcal{C}_b)}{|b|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

5.4 Class-Level Strategies

We also experiment the use of average values in a whole class of blocks, instead of looking at individual blocks. Thus, in this third strategy for computing *bw* factors, terms in a block class are assigned an unique weight. For instance, all page title terms in a Web site are assigned a unique *bw* weight.

We first define the function bw_7 of a class $\mathcal{C}$ as the average value of ICF values for all terms that occur in block class $\mathcal{C}$. This gives an estimate of how discriminative is the occurrence of each term in a class on average, and is computed as:

$$bw_7(t,b) = \begin{cases} \frac{\sum_{t' \in v(\mathcal{C}_b)} ICF(t',\mathcal{C}_b)}{|v(\mathcal{C}_b)|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

where $v(\mathcal{C}_b)$ is the vocabulary composed of the distinct terms that appear in all blocks of class $\mathcal{C}_b$ and $|v(\mathcal{C}_b)|$ is its size.

We also experimented with a strategy based on the average value of the block-level metric bw_5 of all blocks in a class. This new class-level metric, referred to as bw_8, is computed as follows:

$$bw_8(t,b) = \begin{cases} \frac{\sum_{b' \in \mathcal{C}_b} \frac{\sum_{t' \in b'} Spread(t',\rho_{b'})}{|b'|}}{|\mathcal{C}_b|} & \text{if } t \in b \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

where b' is a block variable whose block size is given by $|b'|$, $\mathcal{C}_b$ is the block b class, and $|\mathcal{C}_b|$ is the number of blocks in it.

Finally, we also introduce a class-level metric based on the combination of bw_7 and bw_8, as follows:

$$bw_9(t,b) = bw_7(t,b) \times bw_8(t,b) \tag{5.11}$$

5.5 Dealing with Block Classes with Few Elements

A practical problem that arises when applying term weighting schemes based on the structure of pages is the occurrence of block classes that contain just a few elements. This is a problem specially for computing ICF values. For instance, it is not possible to determine whether any given term is common or not in a block class with only 1 or

2 blocks.

The threshold β, also adopted in the automatic segmentation algorithm (Section 3.2) is adopted in our experiments to indicate the minimum quantity of blocks that a class must have in order to allow a properly computation of ICF values. The value of β threshold was set through experiments described in Section 6.2.2 and was set to 8. For term occurrences in block classes containing less than 8 blocks, we assign an intermediate ICF value obtained by computing the average of the ICF values found on the document collection. Notice that this heuristic changes just a few pages in the collections, since it is applied only to block classes with small number of elements. In our experiments, less than 0.05% of the blocks were in classes with less than 8 blocks.

5.6 Block-Weight Values and Their Meaning

Table 5.1 illustrates values of block-based and class-based bw factors computed for the page of the BLOGs collection depicted in Figure 3.14. We observe distinct properties of our block-weighs functions, as follows:

1. Functions bw_4 and bw_7 (ICF based strategies) yield low values to term occurrence in blocks that are part of the template of the site, such as blocks 1, 2, 3, 9, 10, and 14. These blocks are not useful for distinguishing a page from other pages in the site.

2. Blocks that contain information that humans would usually consider important are assigned higher values by all block-weight functions. This is the case of block 4, for instance, which acts as a page title.

3. Functions bw_6 and bw_9 stress the impact of specific block types by assigning very high values to the page title (block 4), high values to the main contents section (block 5) and tags section (block 8), intermediary values to the user comments section (block 11), and low values to the remaining blocks.

	bw Factors					
Blocks	bw_4	bw_5	bw_6	bw_7	bw_8	bw_9
1	0.0001	2.4285	0.0003	0.0000	2.1902	0.0000
2	0.0000	5.3333	0.0000	0.0000	5.2350	0.0000
3	0.0000	4.4000	0.0000	0.0000	4.5529	0.0000
4	6.5988	4.8333	31.8945	8.0726	5.2013	41.9884
5	1.5246	3.6000	5.4886	1.9064	3.9491	7.5287
6	2.2942	2.4000	5.5062	4.4214	2.7695	12.2455
7	3.9600	2.5816	10.2234	7.6453	2.7937	21.3591
8	3.8632	2.9285	11.3136	8.2209	2.9587	24.3241
9	0.0000	1.0000	0.0000	0.0000	1.0004	0.0000
10	0.0244	1.9186	0.0469	0.0268	1.8621	0.0500
11	2.8076	1.6213	4.5520	7.7667	1.5067	11.7023
12	0.9553	2.4931	2.3818	0.9993	2.6658	2.6641
13	2.2035	2.0260	4.4646	2.7203	2.1750	5.9169
14	0.0000	3.5217	0.0000	0.0000	3.5638	0.0000
15	2.9731	1.9277	5.7314	3.4839	1.9844	6.9137

Table 5.1. Examples of *bw* factors assigned to blocks found in the Web page depicted in Figure 3.14, extracted from the BLOGs collection.

Table 5.2 illustrates values of block-based and class-based *bw* factors computed for the page of the CNET collection depicted in Figure 3.15. As one can see, the properties of these *bw* factors are similar to those enumerated for the Table 5.1.

These examples illustrate that the weight assignments generated by our methods are usually meaningful to humans. Notice that the number of classes to analyze in the experiments is too high to report, since we have several hundred of block classes. Further, it is also difficult to show examples of term-level weights, since they assign different weights for each term occurrence in the page.

Since the bw_9 and bw_6 methods yield the more intuitive *bw* values, and present a bigger capability to discriminate noise from useful information, we only consider the combined methods (including the term-level bw_3 strategy) in our ranking experiments, which will be discussed in the following chapter.

	bw Factors					
Blocks	bw_4	bw_5	bw_6	bw_7	bw_8	bw_9
1	1.2010	1.4000	1.6814	7.3415	1.3883	10.1924
2	0.3126	3.3333	1.0419	0.4129	3.5989	1.4860
3	2.7125	9.0000	24.4125	9.4814	8.5198	80.7800
4	0.0000	2.0000	0.0000	0.0000	2.0435	0.0000
5	0.0000	1.0000	0.0000	0.0000	1.2855	0.0000
6	4.4852	10.000	44.8520	9.4952	9.4654	89.8762
7	0.4765	2.7500	1.3103	0.8201	2.9844	2.4476
8	0.3900	4.7500	1.8525	3.5666	5.0028	17.8432
9	0.5217	3.2857	1.7141	4.5606	3.3337	15.2034
10	3.3834	3.1176	10.5480	8.0418	2.5623	20.6057
11	0.0000	1.0000	0.0000	0.0000	0.5356	0.0000
12	2.7724	1.1856	3.2869	7.3504	1.2706	9.3398
13	2.0839	4.0000	8.3356	9.3764	4.1492	38.9041
14	1.4606	4.0344	5.8926	8.1832	3.8400	31.4237
15	0.2942	3.1428	0.9246	3.0774	3.6761	11.3127
16	0.9018	2.1935	1.9780	9.2998	1.7369	16.1530

Table 5.2. Examples of *bw* factors assigned to blocks found in the Web page depicted in Figure 3.15, extracted from the CNET collection.

Chapter 6

Ranking Using Block-Weight Functions

In this chapter, we present a simple approach of integrating block-weight functions into BM25 ranking formula. As we show through extensive experiments we ran on the 4 Web collections described in Chapter 3, the deployment of our block-weight functions may allow a significant improvement in the quality of search results. We ran experiments to compare the quality of search results when using our methods to the quality obtained when using (i) a BM25 ranking applied to full pages, (ii) a BM25 ranking applied to pages after templates removal, and (iii) a BM25 ranking that takes into account best blocks. These experiments suggest that our block-weighting ranking method is superior to all baselines across all collections we used and that average gain in precision figures from 5% to 20% are generated. Further, our methods decrease the cost of processing queries when compared to the systems using no structural information, decreasing indexing storage requirements and increasing the speed of query processing.

6.1 BM25 Extension for Using bw Information

To incorporate the $bw(t, b)$ information in a ranking function, we use an idea previously described by Robertson et al. [2004], which suggests an adaptation of the BM25 ranking formula for dealing with documents that have multiple fields.

In the original BM25 model, the similarity ranking $sim(\rho, q)$ of a document (page) ρ with regard to a user query q is given by the Formula 4.4. As described in Section 4.0.5, the BM25 ranking model uses term-frequency factors tf computed relatively to the whole document (or page) ρ. Based on the proposal by Robertson et al. [2004] we argue that the computation of tf factors could be done relatively to the blocks that compose a document (page) ρ. This leads to an alternative term-frequency formula computed as

$$tf'(t, \rho) = \sum_{b \in \rho} tf(t, b) \times bw(t, b) \tag{6.1}$$

where $tf(t, b)$ is the term frequency of t in block b of document ρ, and $bw(t, b)$ is one of our block-weight function, as defined in Chapter 5.

As discussed in Robertson et al. [2004], the k_1 parameter in Equation 4.4 controls the impact of the tf factor in BM25 ranking formula. The adoption of term-frequency factors tf', in place of tf factors, might require a distinct fine-tuning of the parameter k_1. For this, we optimize k_1 considering the composite factor

$$k_1 \times \frac{\overline{tf'}}{\overline{tf}} \tag{6.2}$$

where $\overline{tf'}$ and $\overline{tf}$ are average values computed over all term occurrences in the document collection.

Additionally, we also adjust the value of n_t (see Equation 4.4) to count all pages ρ for which $tf'(t, \rho) > 0$, instead of counting all pages ρ for which $tf(t, \rho) > 0$. This adjusted page count is referred to as n'_t. Notice that, in this scheme, only pages that have a block b such that $bw(t, b) > 0$ will be considered in the computation of n'_t.

Given all the considerations above, the similarity ranking $sim(\rho, q)$ of a document or page ρ with regard to a user query q is modified as follows:

$$sim(\rho, q) = \sum_{t \in \rho \wedge t \in q} \frac{(k_1 \times \frac{\overline{tf'}}{\overline{tf}} + 1) \times tf'(t, \rho)}{k_1 \times \frac{\overline{tf'}}{\overline{tf}} \left((1-b) + b\frac{|\rho|}{\overline{\rho}}\right) + tf'(t, \rho)} \times \log \frac{N - n'_t + 0.5}{n'_t + 0.5} \tag{6.3}$$

where the term-frequency $tf'(t, \rho)$ is given by Equation 6.1, with bw factors computed using Equations (5.3) to (5.11).

6.2 Experimental Results

To assert the usefulness of our methods, we performed search experiments on the Web collections described in Section 3.4 (IG, CNN, CNET, and BLOGs), and evaluated the impact of our methods on the quality of results.

6.2.1 Experimental Setup

To perform the ranking experiments, we compose 50 test queries for each Web collection described in Section 3.4. The set of test queries for the IG collection is composed of 50 popular queries extracted from a log of queries submitted to the IG Web search service. Relevance assessments for these queries were made by 85 volunteers from 5 different Brazilian universities. A pooling method [Hawking and Craswell, 1998; Hawking et al., 1999] was used to collect relevance judgments. For each of the 50 queries, we composed a query pool formed by the union of the top 20 documents retrieved by each ranking method considered in our experiments, and then presented those documents in random order to the users. To avoid noise and erroneous judgements, each document retrieved by a given query was evaluated by three distinct volunteers. We considered a document as relevant to a query when at least two of the volunteers considered it as relevant to that query. The relevance evaluations were gathered by presenting the following question to the volunteers: "If you wish to make a query (in a search engine) about the proposed theme (a query extracted from the log), would this page fit the profile of your research?". In case of a positive answer, the user classified the document as relevant to the query.

The queries used for experimentation with CNN, BLOGs and CNET collections were proposed by 50 volunteers who were able to read and understand English texts, such that each volunteer wrote a single query for each collection. We then inspect the proposed queries to ensure that all of them are representative. As in the IG collection, for each query submitted we computed a pool formed by the union of the top 20 results of all the methods we considered, and then presented those documents in random order to the users. The relevance evaluations were gathered by presenting the following question to the volunteers: "The content of the document below attend the type of content you wanted when you formulated your query?"

6.2.2 Determining the Threshold β

A first question that should be addressed when using our block-based weighting schemes is to determine the threshold β, which is adopted both by the SOM tree based segmentation and by our ranking methods. We used the semi-automatic segmentation algorithm to study the impact of choosing different values of β.

The number of elements in a block class affect most the methods based on ICF and could also affect the bw_8, since it is an average of values found in a block class. Further, it is difficult to study the impact the number of elements in a class in term-level or block-level bw factors. Thus, we decided to study such impact only in the class-level bw factors proposed by us, bw_7, bw_8 and bw_9.

To analyze the impact of the number of elements of a block class in the computation of such bw factors, we randomly selected blocks from block classes found by the semi-automatic method in IG and CNN, varying the number of samples taken from each class in the experiment. Instead of computing the *bw* factors using the whole block classes found in these collections, we computed the weight using only the samples.

Thus, let $\mathcal{B}_n$ be the set of blocks obtained when we randomly select n blocks from each block class existing in a Web site collection. For instance, the set $\mathcal{B}_2$ has 2 blocks of each block class of the original collection (with exception of the blocks classes that contains only one block, that will derive a single block in set $\mathcal{B}_2$). Following the same idea, we create sets of blocks $\mathcal{B}_4$, $\mathcal{B}_6$, $\mathcal{B}_8$... and so on, until $n = 50$.

We then studied the similarity between the ranking obtained when computing the weight based on each sample sets $(\mathcal{B}_2, \mathcal{B}_4, \mathcal{B}_6, ..., \mathcal{B}_{50})$ and the ranking obtained when computing the weight based on the whole collection. We use the *Kendall tau* distance [Fagin et al., 2003] to compute the similarity between rankings. The Kendall tau distance between two ranked lists is a metric that counts the number of pairwise agreements between the lists. Closer rankings have Kendall tau similarity close to 1, and value 1 means completely equal ranking.

Figure 6.1 shows the results of Kendall tau obtained when varying the samples used to compute the weights in CNN (a) and IG: (b) when using bw_7, bw_8 and bw_9 as *bw* factors. Notice that for the three *bw* factors studied, in both collections, the ranking obtained quickly converges to the one obtained when computing the *bw* factors taking the whole collections as input. Particularly, we observed that the rankings are almost similar when taking samples with more than eight elements. Further, the differences in the ranking grow fast as the as we reduce the number of elements. Thus, we assigned the value 8 to the threshold β in our experiments.

6.2.3 Baselines

We adopted 3 different baselines to evaluate the impact of our methods on the search results. The first baseline uses BM25 [Robertson and Walker, 1994] with no block information, which is similar to considering the collection as a plain text collection. This baseline, whose ranking is based on Equation 4.4, is referred to as **BM25** in our experiments. The BM25 parameters were set to $k_1 = 1.2$ and $b = 0.75$, which are reasonable values that work well for most collections [Robertson et al., 1996; Robertson and Walker, 2000].

In the second baseline method, we manually removed all the templates for the Web pages, and then applied the BM25 ranking formula in Equation 4.4 to the remaining contents. This second baseline method will be referred to as **BM25NT**, which means BM25 without templates. Notice that the first two baselines do not use block information.

The third baseline is the method proposed in Cai et al. [2004b] that computes two rankings to each user query: 1) a *document-based ranking* (DR), given by BM25 (Equation 4.4), in which whole documents are considered; and 2) *block-based ranking* (BR), that is the ranking given by BM25 to the best-ranked block of each document (or page). The two rankings are then combined as follows:

$$sim(\rho, q) = \alpha\ rank_{DR}(\rho, q) + (1 - \alpha)\ rank_{BR}(\rho, q) \tag{6.4}$$

where $rank_{DR}(\rho, q)$ is the position of the document ρ in the BM25 document-based ranking for query q, and $rank_{BR}(\rho, q)$ is the position of the document ρ in the BM25 block-based ranking for query q. In our experiments the best values obtained for α were 0.4 for IG and 1 for the other collections. This method will be referred in the experiments as **BM25BB** (BM25 with best blocks). This approach was chosen because it is an alternative ranking strategy proposed in the literature [Cai et al., 2004b] which also attempts to take advantage of structural information to improve results.

6.2.4 Ranking Results

To compare the various ranking formulas, we adopted two evaluation metrics. The first metric is the traditional mean average precision (MAP) and the second if precision at 10 (P@10), which measures the amount of relevant documents in the top 10 answers provided on each system [Baeza-Yates and Ribeiro-Neto, 1999].

6.2.4.1 Comparing the Baseline Ranking Formulas

Our first set of experiments aim at comparing ranking results generated by the BM25 and BM25BB baselines. Table 6.1 depicts the results baselines to the four collections adopted. As one can see, the performance of BM25 and BM25BB baselines are the same for collections CNN, BLOGs and CNET and very similar for the IG collection. This is because the training phase yielded $\alpha = 1$ for the first three collection and $\alpha = 0.4$ for the IG collection (see Equation 6.4). These results suggest that, in many cases, the block retrieval ranking by BM25BB is not useful to improve retrieval, and that the use of block information is not always a guarantee of better results.

	Methods	**P@10**	**MAP**
IG	BM25	0.593	0.621
	BM25BB	0.581	0.625
CNN	BM25	0.612	0.691
	BM25BB	0.612	0.691
BLOGs	BM25	0.584	0.644
	BM25BB	0.584	0.644
CNET	BM25	0.476	0.458
	BM25BB	0.476	0.458

Table 6.1. Results obtained by BM25 and BM25BB baselines.

Next, we compare results obtained by the BM25 and BM25NT baselines. Table 6.2 depicts the results achieved by the two baselines for the four collections adopted. As one can see, it is not clear which ranking method is better for the IG, CNN and BLOGs collections. For CNET collection, the Wilcoxon test revealed that the improvements obtained by BM25NT were statistically significant at $p < 0.05$, when using P@10.

However, the improvements obtained in the CNET collection are consequence of a peculiarity of its query set, which contains many terms that appear in the templates of pages. When checking the results, we realized that BM25NT is superior to the BM25 baseline only for these queries. That is, despite the general belief that template removal may improve the quality of search results [Vieira et al., 2006; Yin and Lee, 2004], our results suggest that, in view of the queries we used for experimentation, the improvements are not consistent across various sites, and that template removal is able to achieve superior results only in special cases.

	Methods	**P@10**	**MAP**
IG	BM25	0.593	0.621
	BM25NT	0.571	0.649
CNN	BM25	0.612	0.691
	BM25NT	0.602	0.704
BLOGs	BM25	0.584	0.644
	BM25NT	0.594	0.632
CNET	BM25	0.476	0.458
	BM25NT	0.534	0.481

Table 6.2. Results obtained by BM25 and BM25NT (BM25 removing templates).

We conclude that our three baselines, BM25, BM25NT and BM25BB, yield similar results and that, in the context of our 4 test collections, are basically equivalent. Thus, in the remaining of our experiments, we considered just the BM25 baseline.

6.2.4.2 Results for the Block-Weight Ranking Formulas

Table 6.3 presents the results obtained when we modify the BM25 ranking formula to use the bw_3, bw_6 and bw_9 weighting functions, according to Equation 6.3, and apply them to our 4 test collections segmented by both semi-automatic and automatic algorithms. The first important observation is that the three bw factors yield improved results relative to the alternative of using no bw factor, for both segmentation methods. Our block-based ranking methods achieve improvements in most of the cases, which suggests that the bw factors introduce complementary information about the importance of specific term occurrences inside blocks of the page.

We applied the Wilcoxon statistical test with a 95% confidence level to analyze the results. The statistically significant results are represented in bold in Table 6.3. We first consider the values obtained with the collections segmented by the semi-automatic approach, which is expected to provide the best segmentation results and thus is useful to compare the methods in a scenario which minimizes noisy provided by possible segmentation errors. For this segmentation approach, the bw_6 and bw_9 achieved significant improvements for all metrics when compared to BM25 without bw factors for the IG, CNN and CNET collections. For BLOGs, the improvement of these two methods was significant only for the MAP metric. Considering the term-based bw_3 method, the results are statistically significant for all metrics with the IG and CNN collections. For BLOGs, the improvements obtained by bw_3 are not significant in both P@10 and MAP metrics; and for CNET, only for P@10 the results are significant.

For the automatic segmentation approach, the bw_6 and bw_9 achieved significant improvements for all metrics when compared to BM25 without bw factors only for the IG and CNN collections. As for semi-automatic segmentation approach, the improvement of these two methods in BLOGs collection was significant only for the MAP metric. On the other hand, the results of bw_6 and bw_9 methods in CNET collection were significant only for the P@10 metric, which is a result quite different from those found in semi-automatic approach. We also observe that the performance of bw_3 is significant for all metrics only for CNN collections. For IG, the improvements obtained by the term-based method are significant only for MAP metric. For BLOGs, the improvements of these method are not significant in both P@10 and MAP metrics; and for CNET, only for P@10.

Table 6.4 presents a comparison between the results of the automatic and semi-automatic segmentation for our method using bw_6 and the bw_9 factors. The values obtained for most cases are higher when using the semi-automatic method. However,

		IG		**CNN**	
	bw factor	P@10	MAP	P@10	MAP
	BM25	0.594	0.621	0.612	0.691
semi-automatic	bw_9	**0.667**(12.3%)	**0.749**(20.6%)	**0.642**(04.9%)	**0.786**(13.7%)
	bw_6	**0.653**(09.9%)	**0.730**(17.5%)	**0.648**(05.9%)	**0.800**(15.7%)
	bw_3	**0.655**(10.3%)	**0.698**(12.4%)	**0.638**(04.2%)	**0.769**(10.0%)
automatic	bw_9	**0.659**(10.9%)	**0.733**(18.0%)	**0.630**(02.9%)	**0.779**(12.7%)
	bw_6	**0.655**(10.2%)	**0.715**(15.1%)	**0.636**(03.9%)	**0.790**(14.3%)
	bw_3	0.634 (06.7%)	**0.680**(09.5%)	**0.628**(02.6%)	**0.759**(09.8%)

		BLOGs		**CNET**	
	bw factor	P@10	MAP	P@10	MAP
	BM25	0.584	0.644	0.476	0.458
semi-automatic	bw_9	0.604 (03.4%)	**0.678**(05.3%)	**0.552**(16.0%)	**0.498**(08.7%)
	bw_6	0.610 (04.4%)	**0.675**(04.8%)	**0.558**(17.2%)	**0.513**(12.0%)
	bw_3	0.588 (00.7%)	0.628 (-2.5%)	**0.545**(14.5%)	0.482 (05.2%)
automatic	bw_9	0.602 (03.4%)	**0.677**(05.1%)	**0.512**(07.5%)	0.470 (02.6%)
	bw_6	0.604 (04.4%)	**0.671**(04.2%)	**0.527**(10.8%)	0.445 (-2.8%)
	bw_3	0.592 (00.7%)	0.632 (-1.8%)	**0.514**(08.0%)	0.459 (00.2%)

Table 6.3. Results obtained when using the three alternative *bw* factors and using no *bw* factor (BM25). The percentages indicates the average gains in precision figures and results in bold are statically significant.

the fully automatic method has the advantage of not requiring manual intervention at any point of the segmentation process. Further, we applied Wilcoxon statistical test to compare the results and concluded that the difference is not significant in most of the cases (values where the difference between the automatic and the semi-automatic methods is significant are represented with bold in the winner option). From the results, we can say that the automatic method performed worse in $CNET$.

When investigating the cause for the worse performance of the automatic method in CNET, we realized that it is due a single problem. As discussed in Section 3.4.1.2, the segmentation process of the CNET collection has fail due to small differences in the structure of pages, fact that created a larger number of blocks in that collection. This fact occurred due the many number of lists (such as a list of virtual markets that sell the product of a page in the CNET Shopper) that the semi-automatic segmentation approach identified as a single block, while the automatic approach identified as a set of blocks (one block to each element of the list). We found that such irregularities do not affect the ICF factor, but affect significantly the Spread based measures, since these lists do not belongs to the main sections of the pages, and then many terms of them received a spread degree bigger than the terms of the main regions.

These results suggest that the BM25 ranking formula modified using two of our bw factors, bw_6 and bw_9, were consistently superior to the BM25 baseline, while the BM25NT and BM25BB baselines provided no clear improvements. While our experiments do not allow concluding which method is better (bw_6 or bw_9), they show that both strategies of computing bw factors introduce complementary information about the term occurrences, and that significant improvements can be obtained when considering the structure of documents in information retrieval models.

		semi-automatic		**automatic**	
Collection	bw Factor	P@10	MAP	P@10	MAP
IG	bw_9	0.667	**0.749**	0.659	0.733
	bw_6	0.653	0.730	0.655	0.715
CNN	bw_9	0.642	0.786	0.630	0.779
	bw_6	0.648	0.800	0.636	0.790
BLOGS	bw_9	0.604	0.678	0.602	0.677
	bw_6	0.610	0.675	0.604	0.671
CNET	bw_9	**0.552**	**0.498**	0.512	0.470
	bw_6	**0.558**	**0.513**	0.527	0.445

Table 6.4. Comparison between results obtained when using the automatic and the semi-automatic segmentation methods.

6.2.5 Index Pruning Using bw Factors

The bw factors quantify how important is a given term occurrence for ranking purposes. In this context, when $bw = 0$ for all occurrences of a term t in a given page ρ, i.e., $tf'(t, \rho) = 0$, the presence of the term in the page has no effect on retrieval results. By identifying and removing these entries from the inverted file index we can prune the index and reduce its size, which also allows speeding up query processing [Moura et al., 2008].

In this section, we study the use of bw_3, bw_6 and bw_9 weights to prune index entries. Notice that the amount of pruning performed by each one of these strategies is determined by the ICF factor associated to them. For instance, for the term-level strategy, whenever a given term t occurs in all blocks of a class C (i.e., $ICF(t,C) = 0$), then $bw_3(t,b) = bw_1(t,b) = 0 \; \forall b \in C$. For the block-level strategy, $bw_6(t,b) = bw_4(t,b) = 0$ only when all terms of b also occurs in all other blocks of the class of b. At last, for the class-level strategy, $bw_9(t,b) = bw_7(t,b) = 0$ only when the content of all blocks of C is equal.

Table 6.5 shows the percentage of deleted index entries obtained with the three bw schemes. As one can see, the percentage of pruning obtained with bw_6 and bw_9 weights is similar. With bw_3 weights, the percentage of pruning is higher, particularly in the case of the CNET collection. This happens because, for bw_6 and bw_9 methods, the terms of a block or class are pruned only when all of then have ICF value equal to zero. For instance, for bw_6, if a single term of a block has ICF bigger than zero, then the content of the block cannot be pruned. On the other side, bw_3 is a measure based on ICF value of only one term, so as this method prune all terms of the index with ICF value equal to zero.

	bw_3	bw_6	bw_9
IG	27.9%	20.5%	19.7%
CNN	12.1%	08.3%	07.2%
BLOGs	22.6%	11.1%	12.0%
CNET	34.5%	10.4%	09.3%

Table 6.5. Percentual of deleted index entries performed by bw_3, bw_6 and bw_9 methods.

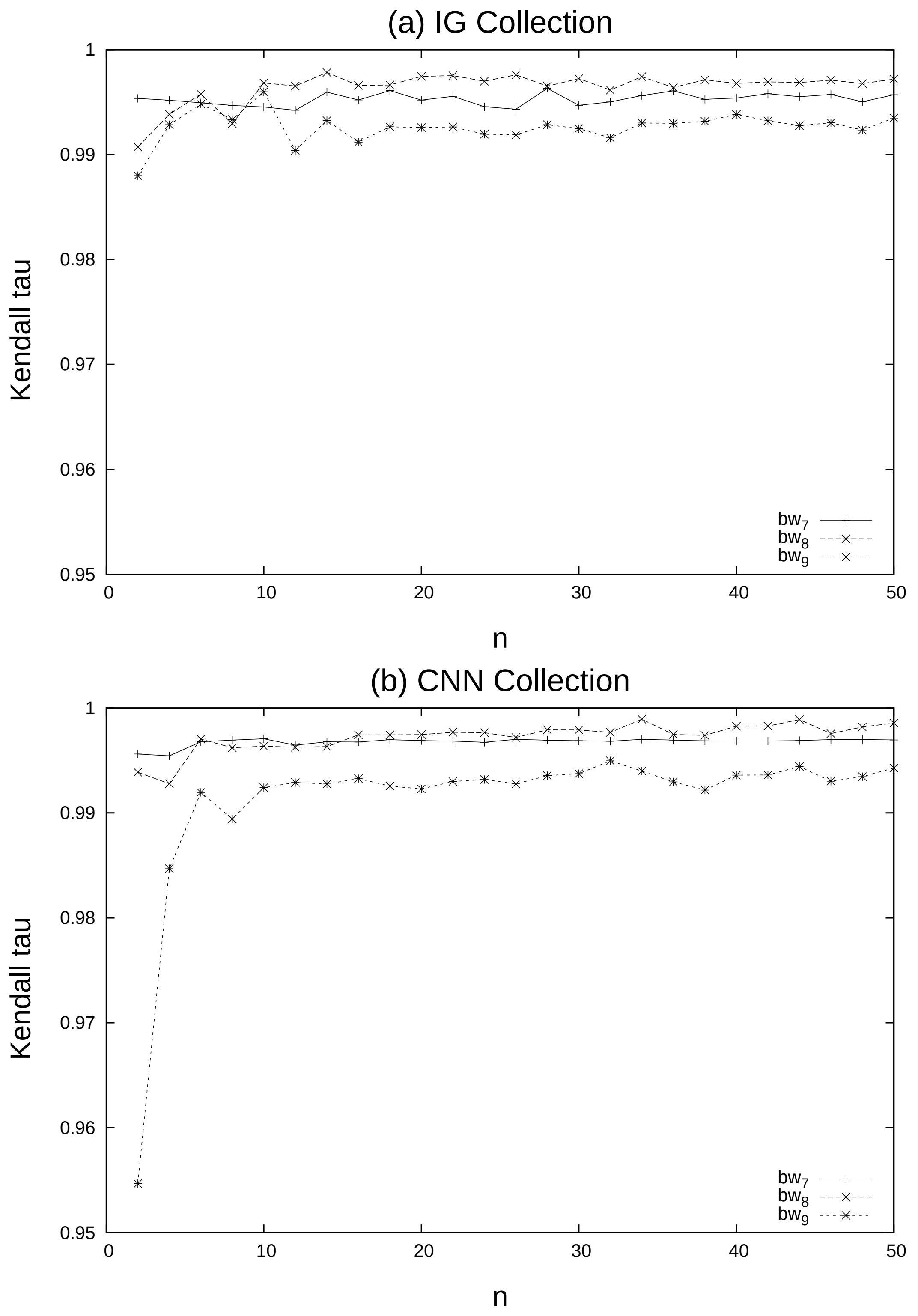


Figure 6.1. Kendall tau distance between ranking results obtained when computing *bw* factors based on the whole collection and ranking of results obtained when computing *bw* factors based on the blocks available on collections $\mathcal{B}_2, \mathcal{B}_4, \mathcal{B}_6, ..., \mathcal{B}_{200}$. (a) The graph obtained from IG collection; and (b) the graph obtained from CNN collection.

Chapter 7

Conclusions and Further Work

In this chapter, we present a brief summary of the achievements of this work. In Section 7.1, is presented a the summary of this work and a final analysis of the results obtained here. Following, in Section 7.2, we present directions for further research.

7.1 Thesis Summary

In this work, we proposed methods that improve the quality of search results by taking advantage of structural information available on Web site collections. For that, we proposed (i) a new approach for representing Web sites in information retrieval systems based on the internal structure of their Web pages; (ii) a method of analyzing Web pages to automatically detect their internal structure; and (iii) new term weighting schemas for Web document collections that leads to improve ranking by considering the location (or block) of the terms occurrences within the Web pages.

Previous approaches of using page structure information to improve the quality of results in Web search have shown the potential of the technique. However, if uncovering page structure requires a costly manual segmentation, the technique becomes not practical. In this thesis, we proposed a fully automatic method of page segmentation to identify the major constituent blocks of a Web page and also to cluster blocks found in a Web site into block classes. As we show through experiments, our segmentation method yields results that are close to the perception of users about how each page should be divided into blocks, and achieved a segmentation performance better than those achieved by Chakrabarti et al. [2008], which is an recent work of the literature that approached the Web page segmentation problem from a textual perspective.

Starting with basic intuitions provided by the concepts of *term frequency* and *inverse document frequency*, we proposed 9 block-weight functions to distinguish the impact of term occurrences inside page blocks, instead of inside whole pages. We used two types of measures to compose the block-weight functions. One of them measure the amount of information that carries an occurrence of a term in the a block class, adopting a metric we call *Inverse Class Frequency*, or ICF. The second type of measure captures how spread is a term in the blocks of a Web page, adopting a metric we call *Spread*. This measure is based on the assumption that blocks with terms spread also in other blocks have a higher chance of being more related to the main subject of their respective pages, implying also in higher weights.

The block-weight functions are then used to compute a modified BM25 ranking function. We compare the effectiveness of our functions against three baselines, by running extensive experiments on 4 Web collections. The first did not consider any structural (block) information. The second one applied a template removal pre-processing. And the third one used a training process to optimize the combination of ranks when considering the whole documents and the blocks as retrieval units. Our results suggest that our block-weighting functions incorporate complementary information on the occurrence of terms inside blocks and, because of that, yield superior results relatively to all three baselines across all 4 Web collections we used, and the average gain in precision figures from 5% to 20% are generated.

Also of interest, our experiments suggest that the simple removal of templates from Web pages does not necessarily lead to improved results. Indeed, for 3 of our test collections, results obtained with template removal were equivalent to results without template removal, for the queries we experimented with.

7.2 Further Work

Block information can be applied to other interesting applications that were not considered in this work. For instance, it can be used to indicate blocks with useless content, which can negatively impact the performance of searching tasks. The detection and removal of noisy content will be considered in future works. Further, we are also interested in studying the impact of block importance information on other information retrieval tasks, such as clustering, filtering tasks and classification of documents.

As discussed in Chapter 3, the results of our automatic segmentation algorithm were far different from the semi-automatic method for CNET collection. When checking the results, we realized that this is due to irregularities in the page structures found in CNET, which affect the automatic identification of recurrent structures. To resolve this problem, we pretend to study new methods for identifying these recurrent regions in the DOM trees of Web pages. An idea is to use information extraction techniques that aim to identifying recurrent regions to extract particular information from Web documents containing multiples records aligned in a regular way [Liu et al., 2003; Mehta et al., 2005; Álvarez et al., 2008].

We are also interested in experimenting with some variations of the ideas proposed in this thesis. For instance, our methods work only with flat block divisions in the Web page collection, which suggests that they can be expanded to process Web page collections in which the page structure is modelled as a hierarchy, including the possibility of nested blocks.

Finally, we are interested in studying the impact of using block information when searching on the Web. To do this, we are working on new block-weight functions based on link information and anchor-text. Preliminary results suggest that anchor text can also be useful to identify the importance of the occurrence of a given term in a block. Eiron and McCurley [2003] found that anchor texts is typically less ambiguous than other types of texts; thus they could can be used to produce a good representation of Web pages. Glover et al. [2002] reached the same conclusion. Our idea is to use anchor texts to identify how well the terms of a block or block class represent the content of Web pages. The first results obtained with this approach are promising.

Bibliography

Ahnizeret, K., Fernandes, D., Cavalcanti, J. M., de Moura, E. S., and da Silva, A. S. (2004). Information retrieval aware web site modelling and generation. In *ER '04: Proceedings of the twentieth-third International Conference on Conceptual Modeling*, pages 402--419, Berlin, Heidelberg. Springer.

Algur, S. P. and Hiremath, P. S. (2006). Extraction of flat and nested data records from web pages. In *AusDM '06: Proceedings of the fifth Australasian conference on Data mining and analystics*, pages 163--168, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Álvarez, M., Pan, A., Raposo, J., Bellas, F., and Cacheda, F. (2008). Extracting lists of data records from semi-structured web pages. *Data Knowl. Eng.*, 64(2):491--509.

Antic, G., Grzybek, P., and Stadlober, E. (2005). Mathematical aspects and modifications of fucks' generalized poisson distribution. In Köhler, R., Altmann, G., and Piotrovski, G., editors, *Quantitative Linguistics. An International Handbook*, volume 27 of *Handbücher zur Sprach- und Kommunikationswissenschaft*, pages 158–180. Walter de Gruyter, Berlin.

Baeza-Yates, R. A. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Bar-Yossef, Z. and Rajagopalan, S. (2002). Template detection via data mining and its applications. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 580--591, New York, NY, USA. ACM.

Berger, A. and Lafferty, J. (1999). Information retrieval as statistical translation. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 222--229, New York, NY, USA. ACM.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107--117.

Cai, D., He, X., Wen, J.-R., and Ma, W.-Y. (2004a). Block-level link analysis. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 440--447, New York, NY, USA. ACM.

Cai, D., Yu, S., Wen, J.-R., and Ma, W.-Y. (2003). Vips: a vision-based page segmentation algorithm. Microsoft Technical Report.

Cai, D., Yu, S., Wen, J.-R., and Ma, W.-Y. (2004b). Block-based web search. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 456--463, New York, NY, USA. ACM.

Callan, J. P. (1994). Passage-level evidence in document retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 302--310, New York, NY, USA. Springer-Verlag New York, Inc.

Cao, Y., Niu, Z., Dai, L., and Zhao, Y. (2008). Extraction of informative blocks from web pages. In *ALPIT '08: Proceedings of the 2008 International Conference* *on Advanced Language Processing and Web Information Technology*, pages 544--549, Washington, DC, USA. IEEE Computer Society.

Chakrabarti, D., Kumar, R., and Punera, K. (2008). A graph-theoretic approach to webpage segmentation. In *Proceeding of the 17th international conference on World Wide Web*, pages 377--386.

Chakrabarti, S., Joshi, M., and Tawde, V. (2001). Enhanced topic distillation using text, markup tags, and hyperlinks. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 208--216, New York, NY, USA. ACM.

Chang, C.-H. and Lui, S.-C. (2001). Iepad: information extraction based on pattern discovery. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 681--688, New York, NY, USA. ACM.

Chehreghani, M. H., Abolhassani, H., and Chehreghani, M. H. (2008). Improving density-based methods for hierarchical clustering of web pages. *Data Knowl. Eng.*, 67(1):30--50.

Chen, J., Zhou, B., Shi, J., Zhang, H., and Fengwu, Q. (2001). Function-based object model towards website adaptation. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 587--596, New York, NY, USA. ACM.

ching Wong, W. and Fu, A. W.-C. (2000). Finding structure and characteristics of web documents for classification. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 96–105.

Crescenzi, V., Merialdo, P., and Missier, P. (2003). Fine-grain web site structure discovery. In *WIDM '03: Proceedings of the 5th ACM international workshop on Web information and data management*, pages 15--22, New York, NY, USA. ACM.

Crescenzi, V., Merialdo, P., and Missier, P. (2005). Clustering web pages based on their structure. *Data Knowl. Eng.*, 54(3):279--299.

Debnath, S., Mitra, P., and Giles, C. L. (2005). Automatic extraction of informative blocks from webpages. In *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, pages 1722--1726, New York, NY, USA. ACM.

Efthimiadis, N. E. (1996). Query expansion. In *Annual Review of Information Systems and Technology*, pages 121–187.

Eiron, N. and McCurley, K. S. (2003). Analysis of anchor text for web search. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 459--460, New York, NY, USA. ACM.

Fagin, R., Kumar, R., and Sivakumar, D. (2003). Comparing top k lists. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 28--36, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

Fernandes, D., de Moura, E. S., Ribeiro-Neto, B., da Silva, A. S., and Gonçalves, M. A. (2007). Computing block importance for searching on web sites. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 165--174, New York, NY, USA. ACM.

Frakes, W. and Baeza-Yates, R., editors (1992). *Information retrieval, data structures and algorithms.* Prentice Hall, New York, Englewood Cliffs, N.J.

Fu, Y., Yang, D., Tang, S., Wang, T., and Gao, J. (2007). Using xpath to discover informative content blocks of web pages. In *SKG '07: Proceedings of the Third International Conference on Semantics, Knowledge and Grid*, pages 450--453, Washington, DC, USA. IEEE Computer Society.

Gibson, J., Wellner, B., and Lubar, S. (2007). Adaptive web-page content identification. In *WIDM '07: Proceedings of the 9th annual ACM international workshop on Web information and data management*, pages 105--112, New York, NY, USA. ACM.

Glover, E. J., Tsioutsiouliklis, K., Lawrence, S., Pennock, D. M., and Flake, G. W. (2002). Using web structure for classifying and describing web pages. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 562--569, New York, NY, USA. ACM.

Harman, D. (1993). Overview of the first trec conference. In *SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 36--47, New York, NY, USA. ACM.

Hattori, G., Hoashi, K., Matsumoto, K., and Sugaya, F. (2007). Robust web page segmentation for mobile terminal using content-distances and page layout information. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 361--370, New York, NY, USA. ACM.

Hawking, D. and Craswell, N. (1998). Overview of TREC-7 very large collection track. In *Proc. of the Seventh Text Retrieval Conf.*, pages 91--104.

Hawking, D., Craswell, N., Thistlewaite, P., and Harman, D. (1999). Results and challenges in web search evaluation. *Comput. Netw.*, 31(11-16):1321--1330.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2(1):193–218.

Kang, J. and Choi, J. (2007). Detecting informative web page blocks for efficient information extraction using visual block segmentation. In *ISITC '07: Proceedings of the 2007 International Symposium on Information Technology Convergence*, pages 306--310, Washington, DC, USA. IEEE Computer Society.

Kleinberg, J. and Tardos, E. (2002). Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields. *J. ACM*, 49(5):616--639.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604--632.

Kohlschütter, C. and Nejdl, W. (2008). A densitometric approach to web page segmentation. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 1173--1182, New York, NY, USA. ACM.

Kovacevic, M., Diligenti, M., Gori, M., and Milutinovic, V. (2002). Recognition of common areas in a web page using visual information: a possible application in a page classification. In *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)*, page 250, Washington, DC, USA. IEEE Computer Society.

Lafferty, J. and Zhai, C. (2001). Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111--119, New York, NY, USA. ACM.

Li, L., Liu, Y., Obregon, A., and Weatherston, M. (2007). Visual segmentation-based data record extraction from web documents. In *Information Reuse and Integration, 2007. IRI 2007. IEEE International Conference on*, pages 502--507.

Lin, S.-H. and Ho, J.-M. (2002). Discovering informative content blocks from web documents. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 588--593, New York, NY, USA. ACM.

Liu, B., Grossman, R., and Zhai, Y. (2003). Mining data records in web pages. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601--606, New York, NY, USA. ACM.

Liu, Y., Wang, Q., Wang, Q., Liu, Y., and Wei, L. (2006). An adaptive scoring method for block importance learning. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 761--764, Washington, DC, USA. IEEE Computer Society.

Maron, M. E. and Kuhns, J. L. (1960). On relevance, probabilistic indexing and information retrieval. *J. ACM*, 7(3):216--244.

Mehta, R. R., Mitra, P., and Karnick, H. (2005). Extracting semantic structure of web documents using content and visual information. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 928--929, New York, NY, USA. ACM.

Moura, E. S. d., Santos, C. F. d., Araujo, B. D. s. d., Silva, A. S. d., Calado, P., and Nascimento, M. A. (2008). Locality-based pruning methods for web search. *ACM Trans. Inf. Syst.*, 26(2):1--28.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web.

Ponte, J. M. and Croft, W. B. (1998). A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275--281, New York, NY, USA. ACM.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846--850.

Robertson, S., Walker, S., Jones, S., Hancock-Beaulieu, M., and Gatford, M. (1995). Okapi at TREC-3. In Harman, D. K., editor, *The Third Text REtrieval Conference (TREC-3)*, pages 109–128, Gaithersburg, MD, USA. Dept. of Commerce, National Institute of Standards and Technology.

Robertson, S., Walker, S., Jones, S., Hancock-Beaulieu, M., and Gatford, M. (1996). Okapi at trec-3. In *The Third Text REtrieval Conference (TREC-3)*, pages 109--128, Gaithersburg, MD, USA. Dept. of Commerce, National Institute of Standards and Technology.

Robertson, S., Zaragoza, H., and Taylor, M. (2004). Simple bm25 extension to multiple weighted fields. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 42--49, New York, NY, USA. ACM.

Robertson, S. E. and Jones, K. S. (1988). Relevance weighting of search terms. pages 143--160.

Robertson, S. E. and Walker, S. (1994). Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 232--241, New York, NY, USA. Springer-Verlag New York, Inc.

Robertson, S. E. and Walker, S. (2000). Okapi/keenbow at trec-8. In *The Eighth Text REtrieval Conference (TREC 8)*, pages 151+. NIST.

Salton, G. (1971). *The SMART Retrieval System—Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Salton, G. and Lesk, M. E. (1968). Computer evaluation of indexing and text processing. *J. ACM*, 15(1):8--36.

Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc., New York, NY, USA.

Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, 18(11):613--620.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell system technical journal*, 27.

Song, R., Liu, H., Wen, J.-R., and Ma, W.-Y. (2004a). Learning block importance models for web pages. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 203--211, New York, NY, USA. ACM.

Song, R., Liu, H., Wen, J.-R., and Ma, W.-Y. (2004b). Learning important models for web page blocks based on layout and content analysis. *SIGKDD Explor. Newsl.*, 6(2):14--23.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application to retrieval. *Journal of Documentation*, 28(1):11–20.

Sparck Jones, K. (1973). Index term weighting. *Information Storage and Retrieval*, 9(11):619–633.

Sparck Jones, K. (1979). Experiments in relevance weighting of search terms. *Information Processing & Management*, 15(13):133–144.

Stockman, G. and Shapiro, L. G. (2001). *Computer Vision.* Prentice Hall PTR, Upper Saddle River, NJ, USA.

Strehl, A. and Ghosh, J. (2003). Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583--617.

Strehl, A., Strehl, E., Ghosh, J., and Mooney, R. (2000). Impact of similarity measures on web-page clustering. In *In Workshop on Artificial Intelligence for Web Search (AAAI 2000*, pages 58--64. AAAI.

Trotman, A. (2005). Choosing document structure weights. *Inf. Process. Manage.*, 41(2):243--264.

Vidal, M. L. A., da Silva, A. S., de Moura, E. S., and Cavalcanti, Jo a. M. B. (2006). Structure-driven crawler generation by example. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 292--299, New York, NY, USA. ACM.

Vieira, K., da Costa Carvalho, A. L., Berlt, K., de Moura, E. S., da Silva, A. S., and Freir, J. (2009). On finding templates on web collections. *World Wide Web*, 12(2):171--211.

Vieira, K., da Silva, A. S., Pinto, N., de Moura, E. S., Jo a. M. B. C., and Freire, J. (2006). A fast and robust method for web page template detection and removal. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 258--267, New York, NY, USA. ACM.

Vinh, N. X., Epps, J., and Bailey, J. (2009). Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1073--1080, New York, NY, USA. ACM.

Yi, L., Liu, B., and Li, X. (2003). Eliminating noisy information in web pages for data mining. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 296--305, New York, NY, USA. ACM.

Yin, X. and Lee, W. S. (2004). Using link analysis to improve layout on mobile devices. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 338--344, New York, NY, USA. ACM.

Yu, S., Cai, D., Wen, J.-R., and Ma, W.-Y. (2003). Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 11--18, New York, NY, USA. ACM.