Understanding the Function of Web Elements Using Random Walk Models

Xinyi Yin
Department of Computer Science,
National University of Singapore,
Singapore 117543.
yinxinyi@comp.nus.edu.sg

Wee Sun Lee
Department of Computer Science and
Singapore –MIT Alliance
National University of Singapore,
Singapore 117543.
leews@comp.nus.edu.sg

ABSTRACT
In this paper, we describe a method for understanding the function of web elements. It classifies web elements into five functional categories: Content (C), Related Links (R), Navigation and Support (N), Advertisement (A) and Form (F). We construct five graphs for a web page, and each graph is designed such that most of the probability mass of the stationary distribution is concentrated in nodes belong to its corresponding category. We perform random walks on these graphs until convergence and classify based on its probability values in the different graphs. Our experiment shows that the new method performed very well comparing to basic machine learning methods.

Categories & Subject Descriptors
H.4.3 [**Communications Applications**]: Communications Applications - *Information browsers*; H.5.2 **[User Interfaces]:** User Interfaces - *Graphical user interfaces (GUI).*

General Terms
Documentation, Design.

Keywords
HTML, WWW (World Wide Web), Classification.

1. INTRODUCTION
A web page is made up of hundreds of basic elements. The functional role of each element is different. For example, an image can be the banner of a website, an advertisement, or a picture for a news article. This paper aims to solve the problem of accurately understanding the functional role of each element. We will classify the content into six categories:

- Content (C): The main content of a web page, including main article, title, author, date, supporting picture etc
- Related Link (R): links to web pages that tell the related story and help a reader to further explore a topic.
- Form (F): Forms can be used to enable the reader to interact with the webpage.
- Advertisement (A): Web page is filled with internal or external advertisements.
- Navigation and Support (N): The links that helps a reader to navigate or content like the banner, and the logos.

We propose a method for automatically categorizing the web elements. It is based on random walks on specially designed graphs. For each web page we build five graphs, one for each functional category, with the basic elements in the web page as vertices. Each graph is specifically designed such that most of the probability mass of the stationary distribution of a random walk on it is concentrated in nodes that belong to its corresponding category. In analogy to Google's PageRank, the score of each element in the graph after random walk is called CategoryRank. We compare an element's CategoryRank in the five graphs and classify the element into the category where it has maximum value.

2. BUILDING CATEGORY GRAPHS
Many different methods have been proposed to partition an HTML page into blocks. For example [3] [6] [7]. We use the algorithm similar to [6] that uses the DOM interface provided by the web browser to divide the web page into non-overlapping visible elements. Figure 1 shows how a CNN web page will be divided.

Figure 1. Homepage of www.cnn.com.

We construct directed graphs for each functional category based on the elements such that the sum of the weights coming out of each node to 1. The weight of edge (i, j) is the probability of a random walker at node i moving to node j at the next time instance.

Based on the features of two basic elements, we will connect them with a weight that increases with the likelihood that the two nodes belong to the same object. We take the following features into consideration:

1. Match(*Pm*): The cosine similarity between elements.
2. Distance(Pd): The pixel distance between elements
3. Neighborhood(Nb): if positioned next to each other
4. Same Edge(Eg): with same left, right top or bottom edge.
5. Same Tag(Tg): two elements have same tag

6. Same Parent(Pr): children of the same parent node in the HTML hierarchy tree.

The weight from the element i to element j is calculated by the function *W(i,j)* where

$$W(i,j) = I(i,j) / \sum_{j=1}^{N} I(S,j)$$

$$\mathrm{I(i,j)} = \mathrm{Pm(i,j)} + \mathrm{Pd(i,j)} + \mathrm{Nb(i,j)} + \mathrm{Pr(i,j)} + \mathrm{Eg(i,j)} + \mathrm{Tg(i,j)}$$

For each of all the edges in the basic graph we will add category dependent weight to generate category graphs. For Content (C) graph, we will consider features including the size of the element, the location of the element, the element tag type (tag like "<P>" or "<B>" is normally used for main content), and the text length of the two. For Advertisement (A) graph, we use the tag features. (Animated Flash is mainly used for advertisements). We also collect a collection of words that are often be used in advertisement, for example "advertisement", "sponsor", "sale", "classified", further more, if an element contains links to an advertisement website like "doubleclick.com" or the URL structure is under a folder "/ads/" or "ad=" etc, we will increase its weight. For Relate (R) Graph, we use features including Tag information ("<Li>", "<a>") the length of anchor text and link category (internal or external). For Navigation and Support Graph (N), we will use the tag category, location, anchor text length. For Form Graph (F) the tag category information is a very important indicator, together with the appearance of words like "search", "submit", "form", "login" etc. In this manner each category graph is constructed

We perform random walk on the graph and use the stationary distribution as a measure of the likelihood that an element in a web page belongs to certain category. Inspired by "PageRank" proposed by [4], we call the score obtained by an element in each graph its "CategoryRank" (CR). Each element will get five "CategoryRank" from the five graphs. We then compare the element's "CategoryRank" in five graphs and classify an element to a category with the maximum "CategoryRank". The formula for updating the rank in each round is:

$$CR^{t}(i) = (1-d)CR^{t-1}(i) + d \sum_{(j,i)\in E} W(j,i) * CR^{t-1}(j)$$

Each round the CategoryRank of node i will be updated with the contribution from itself and all linking vertices in the graph. We set d as 0.85, We set a threshold to check whether the probability distribution has converged in each round of iteration. This is done by summing up all the changes in the CategoryRank of every element and stopping the iterations if the sum is smaller than 0.0001. Normally it takes about 10 rounds before the scores stabilize.

3. EXPERIMENT RESULT

We implemented the system to validate our ideas and evaluate the performance of this approach. To compare with simple machine learning approaches, we used the off-the-shelf system WEKA [6] for this experiment. Precision (P), recall (R) and F-measure as the evaluation

We manually labeled 12,134 elements from 150 websites into one of the five categories. We set aside the first 10,009 as the training set and the rest 2,125 as the test set. The corresponding results on the test set are showed in Table 1.

Table 1, Experiment result with test set

Class	Precision	Recall	F-Measure
Content	0.92	0.93	0.93
Advertisement	0.75	0.90	0.82
Relate	0.58	0.66	0.62
Navigation	0.76	0.56	0.64
Form	0.93	0.74	0.82

To compare the experiment result, we use the WEKA package J48 classifier on the same feature set. The results are listed in table 2:

Table 2, Experiment result on test sets

Class	Precision	Recall	F-Measure
Content	0.85	0.77	0.80
Advertisement	0.40	0.60	0.48
Related Link	0.12	0.58	0.20
Navigation	0.85	0.68	0.76
Form	0.47	0.75	0.58

The result on the training and test set shows that the multi-graph random walk method is an effective and practical for method for classifying web elements. As we can see from the Table 1 and Table 2, the performance of our approach is better than the machine learning approach for all categories except for the navigation.

4. REFERENCE

[1] Ian H. Witten and Eibe Frank, Data Mining: Practical machine learning tools with Java implementations," Morgan Kaufmann, San Francisco, 2000.

[2] Lawrence Kai Shih and David R. Karger. Using URLs and Table Layout for Web Classification Tasks In Proceedings of the 13th International World Wide Web Conference, 2004

[3] Ruihua Song, Haifeng Liu, Jirong Wen, Wei-Ying Ma. Learning Block Importance Models for Web Pages. In *Proceedings of 13th International World Wide Web Conference*, 2004.

[4] Sergey Brin, Lawrence Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. . In *Proceedings of the 7th International World Wide Web Conference*, 1998.

[5] Xiao-Dong Gu, Jinlin Chen, Wei Ying Ma, Guo-Liang Chen. Visual Based Content Understanding towards Web Adaptation. In *Second International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, 2002.

[6] Xinyi Yin, Wee Sun Lee. Using Link Analysis to Improve Layout on Mobile Devices. In *Proceedings of 13th International World Wide Web Conference*, 2004.

[7] Yudong Yang, HongJiang Zhang. HTML Page Analysis Based on Visual Cues. In *7th International Conference on Document Analysis and Recognition*, 2001.