웹 이용 마이닝을 위한 데이터 전처리에서 사용자 구분에 관한 연구

최영환*, 이상용**
공주대학교 컴퓨터 공학과*, 공주대학교 정보통신공학부**
{cyhmad*, sylee**@kongju.ac.kr}

A Study of User Identification in Data Preprocessing for Web Usage Mining

Young-hwan Choi*, Sang-yong Lee**
Dept. of Computer Engineering*, Division of Information & Communication Engineering**, Kongju National University

요 약

웹 이용 마이닝은 거대한 웹 데이터 저장소의 로그들을 이용하여 웹 사용자의 사용 패턴을 분석하는 데이터 마이닝 기술이다. 마이닝 기술을 적용하기 위해서는 전처리 과정 중의 사용자와 세션을 정확하게 구분해야 하는데, 표준 웹 로그 형식의 웹 로그만으로는 사용자를 완전히 구분할 수 없다. 따라서 정확한 결과를 얻기 위해 사용자와 세션을 구분할 수 있는 모듈을 웹 서버에서 제공하거나, 각각의 페이지에 적당한 실행 코드를 삽입해야 한다. 사용자와 세션을 구분하는 데는 캐시 문제, 방화벽 문제, IP(ISP)문제, 프라이버시 문제, 쿠키 문제 등 많은 문제들이 있지만, 이 문제를 해결하기 위한 명확한 방법은 아직 없다.

이 논문은 참조 로그와 에이전트 로그, 그리고 액세스 로그 등 서버측 클릭스트림 데이터만을 이용하여 사용자와 세션을 구분하는 방법을 제안한다.

1. 서론

WWW은 트래픽의 양과 웹 사이트의 복잡도와 크기가 늘어나면서 눈부시게 발전하고 있다. 웹 사이트의 디자인, 웹 서버 디자인 등은 이러한 성장과 더불어 더욱 복잡해지고 있다. 데이터 마이닝은 많은 양의 데이터에 함축적으로 들어 있는 지식이나 패턴을 찾아내는 기술이라고 정의할 수 있다. 데이터 마이닝은 사용자가 방문한 웹 페이지와 사용자의 특징을 보관하고, 이 데이터를 분석하여 각각의 사용자에 맞는 웹 페이지를 동적으로 생성해줄 수 있다. 더욱이 웹 액세스의 다차원 웹 로그 분석을 이용한 트렌드 분석을 통해 웹에서 어떤 일이 일어나고 있는지에 대해서도 대략적인 정보를 제공해줄 수 있다.

웹 마이닝은 거대한 웹 데이터 저장소의 로그들을 이용하여 웹 사용자의 사용 패턴을 분석하는 데이터 마이닝 기술이다[1]. 웹 개인화를 위해서는 사용자와 세션을 정확하게 구분해야 한다. 표준 웹 로그 형식의 웹 로그만으로는 사용자를 완전히 구분할 수 없다. 따라서 정확한 결과를 얻기 위해 사용자와 세션을 구분할 수 있는 모듈을 웹 서버에서 제공하거나, 각각의 페이지에 적당한 실행 코드를 삽입해야 한다. 또 사용자 정보, 세션 정보 등 마이닝에 필요한 모든 정보를 웹 로그에 저장하고 있다고 해도 웹 로그 내의 필요 없는 정보를 제거하는 등의 이유로 전처리 과정은 필요하다. 전처리 과정 중에 중요한 단계가 사용자 구분과 세션 구분인데, 사실 캐시 문제, 방화벽 문제, IP(ISP)문제, 사용자 프라이버시 문제, 쿠키 문제 등 여러 가지 문제점들에 직면하게 된다.

이 논문은 이러한 문제점들을 해결하기 위해, 클라이언트측 데이터를 활용하지 않고, 서버측 클릭 스트림 데이터를 사용하여 고유의 사용자들과 세션을 구분하기 위한 방법을 제안한다.

2. 관련연구

웹 서버 로그로부터의 데이터 마이닝 기술 적용 개념은 Chen[2], Mannila[3], Yan[4]에서 처음 제기되었다.

Chen 등은 사용자 세션을 트랜잭션으로 분류하기 위하여 최대 선점 참고를 이용하는 개념을 소개했다. 최대 선점 참고는 사용자가 특정 사용자 세션동안 이전에 보았던 페이지를 요청하는 곳에서 취소하기 이전에 사용자에 의한 마지막 요청 페이지이다. 예를 들어 사용자 세션이 A-B-A-C-D-C 페이지 순서로 요청이 구성되어 있다면 세션에 대한 최대 선점 참고는 B와 D일 것이다.

Pitkow[5]에 의하여 수행된 연구에서는 웹 서버 로그로부터 사용자와 사용자 세션을 구분하는 문제의 어려움을 보여준다. 확실한 사용 데이터 수집에 가장 큰 두 가지 장애는 로컬 캐싱과 프록시 서버이다. 성능향상과 네트워크 트래픽의 최소화를 위하여 요청된 페이지들을 대부분의 웹 브라우저가 캐시한다. 결과적으로, 사용자가 "뒤로" 버튼을 누를 때 캐시된 페이지는 보여지지만 웹 서버는 페이지 접근을 반복하지 않는다. 프록시 서버는 캐시의 중간 레벨을 지원하고 구분된 사이트 이용으로 더 많은 문제를 일으킨다. 웹 서버 로그에서 프록시 서버로부터의 모든 요청은 요청이 잠재적으로 더욱 많은 사용자를 제시한다 할지라도 같은 인식자를 갖는다. 프록시 서버 레벨 캐싱으로 인하여 서버로부터의 단일 요청은 확장된 시간의 영역을 통하여 다중 사용자에게 보여질 수 있다.

쿠키를 보내는 대신에, 자바 에이전트를 보내어 클라이언트측 브라우저에서 실행되어 웹 서버로 정확한 사용 정보를 보내주는 방법은 사용자 프라이버시 문제에 직면

하게 된다[6].

<표 1>에서는 사용자와 세션의 구분에 사용되어지는 방법들을 비교하였다[7].

표 1. 사용자 구분에 사용되는 방법들의 비교

Method	Description	Privacy Concern	Advantages	Disadvantage
IP Address & Agent	Assume each unique IP address/Agent pair is a unique user.	Low	Always available. No additional technology required.	Not guaranteed to be unique. Defeated by random or rotating IP.
Embedded Session ID	Use dynamically generated pages to insert ID into every link.	Low/ Medium	Always available. Independent of IP address.	No concept of a repeat visit. Requires fully dynamic site.
Registration	Users explicitly sign-in to site.	Medium	Can track single individuals, not just browsers.	Not all users may be willing to register.
Cookie	Save an identifier on the client machine.	Medium /High	Can track repeat visits.	Can be disabled. Negative public image.
Software Agent	Program loaded into browser that sends back usage data.	High	Accurate usage data across entire Web.	Likely to be refused. Negative public image.
Modified Browser	Browser records usage data.	Very High	Accurate usage data across entire Web.	Users must explicitly ask for software.

3. 사용자와 세션을 구분하기 위한 처리절차

본 논문에서 사용된 사용자 구분과 세션 구분을 위한 절차는 다음과 같다.

먼저 각 로그 데이터를 시간 순으로 정렬한 뒤 정렬된 로그 데이터를 IP와 에이전트, 그리고 시간 별로 구분한다. 그리고 구분된 로그 데이터를 타임아웃과 참조로그를 이용하여 세션을 구분한다.

< 세션 구분 절차 >

1. $S_i = \{f_1,\cdots,f_n\}$(시간순서에 따른 session History)
2. l_j, f_j, r_j, t_j(log entry, request, referrer, time)
3. 데이터를 IP Address, agent, time에 따라 정렬
4. T = 30minute(session time out)
5. 같은 IP Address와 Agent를 갖는 l_j에 대해

```
for each l_j do
  if ((t_j - t_j-1) > T or r_j ∋ {S_0,···,S_m} then
      i = i + 1
      Add l_j to S_i
  else
      for i = 0 to n
          if r_j ∈ S_i then
              Add l_j to S_i
          else
              i = i + 1
      end
end
```

4. 사용자 구분

유일한(unique) 사용자들을 구분하는 작업은 로컬 캐시, 방화벽, 프록시 서버의 존재로 매우 복잡해진다. 웹 마이닝은 이 문제에 대해서 다루는 가장 쉬운 길인 사용자 협력에 크게 의존한다. 로그/사이트 의존적 방법이라 할지라도, 휴리스틱 방법은 유일한 사용자를 구분하는데 유용한 방법이다. IP 주소가 같다 할지라도 에이전트 로그가 브라우저나 OS에서 바뀐다하더라도, 다른 사용자를 표시하는 IP 주소에 대해서 다른 에이전트 형식을 만들어주는 것이다.

본 논문에서 제안한 방법은, 같은 IP 주소에서 접근한 사용자라 할지라도, 페이지 참조들의 연관성을 처리하여 다른 사용자로 구분하게 된다.

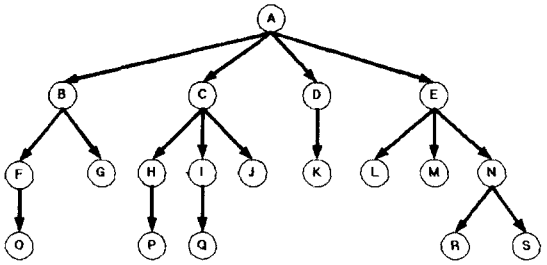

그림 1. 웹 사이트의 예 - Hyper Links로 표현된 페이지 사이의 접근경로

<그림 1>의 웹 사이트를 고려해 보자. 샘플 정보는 <그림 2>에서 보여진 접속, 참조, 에이전트 로그들로부터 수집된다. 모든 로그 엔트리들은 같은 IP 주소와 저장되지 않은 사용자 ID를 갖는다. 그러나 다섯 번째, 여섯 번째, 여덟 번째, 그리고 열 번째 엔트리는 다른 에이전트를 이용하여 접근되고 있기 때문에 적어도 두 사용자 세션을 나타내는 로그로 가정한다.

No	IP Address	Userid	Time	Method/URL/Protocol	Status	Size	Referred	Agent
1	210.106.81.115	-	[25/Jul/2001:09:05:45+0900]	"Get A.html HTTP/1.0"	200	3290	-	Mozilla/4.2(Win98,U)
2	210.106.81.115	-	[25/Jul/2001:09:06:37+0900]	"Get E.html HTTP/1.0"	200	2050	A.html	Mozilla/4.2(Win98,U)
3	210.106.81.115	-	[25/Jul/2001:09:06:43+0900]	"Get H.html HTTP/1.0"	200	4130	-	Mozilla/4.2(Win98,U)
4	210.106.81.115	-	[25/Jul/2001:09:07:12+0900]	"Get N.html HTTP/1.0"	200	5096	E.html	Mozilla/4.2(Win98,U)
5	210.106.81.115	-	[25/Jul/2001:09:07:59+0900]	"Get A.html HTTP/1.0"	200	3290	-	Mozilla/(IE5.0,WinNT)
6	210.106.81.115	-	[25/Jul/2001:09:08:35+0900]	"Get E.html HTTP/1.0"	200	2050	A.html	Mozilla/(IE5.0,WinNT)
7	210.106.81.115	-	[25/Jul/2001:09:08:47+0900]	"Get P.html HTTP/1.0"	200	8140	H.html	Mozilla/4.2(Win98,U)
8	210.106.81.115	-	[25/Jul/2001:09:12:21+0900]	"Get B.html HTTP/1.0"	200	1820	A.html	Mozilla/(IE5.0,WinNT)
9	210.106.81.115	-	[25/Jul/2001:09:12:57+0900]	"Get R.html HTTP/1.0"	200	2270	N.html	Mozilla/4.2(Win98,U)
10	210.106.81.115	-	[25/Jul/2001:09:13:16+0900]	"Get G.html HTTP/1.0"	200	9430	B.html	Mozilla/(IE5.0,WinNT)
11	210.106.81.115	-	[25/Jul/2001:09:14:27+0900]	"Get M.html HTTP/1.0"	200	7220	E.html	Mozilla/4.2(Win98,U)
12	210.106.81.115	-	[25/Jul/2001:11:25:22+0900]	"Get A.html HTTP/1.0"	200	3290	-	Mozilla/4.2(Win98,U)
13	210.106.81.115	-	[25/Jul/2001:11:28:23+0900]	"Get D.html HTTP/1.0"	200	1680	A.html	Mozilla/4.2(Win98,U)

그림 2. Access, Referrer, Agent Logs로부터 얻어진 정보의 예

사용자 구분에 대한 다른 휴리스틱 방법은 참조 로그와 각 사용자 브라우징 경로를 구성하기 위한 사이트 토폴로지를 결합한 액세스 로그를 이용한다. 만약 또다시 사용자가 페이지를 방문한다면 휴리스틱 방법은 같은 IP 주소로 다른 사용자가 접근한 것으로 인식한다.

<그림 2>를 보면, 세 번째 엔트리-페이지 H는 A와 E 페이지에 직접적으로 접근하지 못할 뿐만 아니라, 다른 이전 로그 엔트리의 어느 곳에서도 온 것이 아니다. 이것은 세 번째 사용자가 같은 IP 주소로 들어왔다는 것을 보여준다. 그러므로, 샘플 로그의 사용자 구분 후에, 세 개의 유일한 사용자가 A-E-N-R-M-A-D, A-E-B-G, H-P로 각각 구분된다. 사용자 구분에 대해서는 휴리스

틱 방법만이 존재한다는 것은 매우 중요하다. 같은 타입의 기계에서 같은 브라우저를 이용하는 같은 IP 주소의 두 사용자는 만약 그들이 페이지들의 같은 집합을 보고 있다면 사용자 구분을 쉽게 할 수 없다. 반대로, 다른 브라우저에서 실행하는 단일 사용자나, 사이트 링크 구조를 이용하지 않고 URL을 직접 입력하는 것은 다중 사용자 상황에서 놓칠 수 있는 부분이다.

5. 세션 구분

세션은 한 사용자가 주어진 사이트에 요청을 처음 시도해서 사이트를 떠날 때까지로 구분한다. 따라서 실제 사용자의 웹 사용 패턴을 발견하기 위한 세션 구분은 데이터 마이닝 입력 데이터로서 매우 중요한 의미를 가진다.

세션 구분의 목표는 개인 세션으로 각각의 사용자 페이지 접근을 구분하는 것이다. 그러나 실제 웹 로그만으로 세션을 구별하기에는 어려움이 있다. 사용자들은 로그아웃 버튼을 누르지 않고, 웹 브라우저를 닫는 경우가 많다. 역시 표준 웹 로그 형식을 따르는 로그만이 있는 경우, 사용자 구별이 어렵다면, 세션 구분도 어려워진다. 어쩔 수 없이 IP에 따라 사용자를 구별한다.

세션의 정의에 따라 같은 IP에서 다른 시간에 접속하는 경우, 이 세션은 다른 세션으로 보아야 한다. 이 경우 역시 사용자가 사이트를 떠나는 시점을 파악해야 한다. 이를 위한 가장 간단한 방법은 타임아웃을 사용하는 것이다. 본 논문에서는 30분을 디폴트 타임아웃으로 이용한다. 샘플 로그로부터 사용자에 대한 경로는 마지막 두 참조가 첫 번째 다섯보다 타임아웃 이후에 일어나기 때문에 두 개의 세션으로 나뉘어진다. 세션 구분의 결과는 A-E-N-R-M, A-D, A-E-B-G, H-P로 구성된 네 개의 사용자 세션이다. <그림 3>에서는 샘플 로그의 세션 구분 결과를 보여주고 있다.

Session No	IP Address/Agent	Time	Request URL	Referring URL
1	210.106.81.115/ Mozilla/4.2(Win98.I)	[25/Jul/2001:09:05:45+0900]	A.html	-
		[25/Jul/2001:09:06:37+0900]	E.html	A.html
		[25/Jul/2001:09:07:12+0900]	N.html	E.html
		[25/Jul/2001:09:12:57+0900]	R.html	N.html
		[25/Jul/2001:09:14:27+0900]	M.html	E.html
2	210.106.81.115/ Mozilla/4.2(Win98.I)	[25/Jul/2001:09:06:43+0900]	H.html	-
		[25/Jul/2001:09:08:47+0900]	P.html	H.html
3	210.106.81.115/ Mozilla/(IE5.0,WinNT)	[25/Jul/2001:09:07:59+0900]	A.html	-
		[25/Jul/2001:09:08:35+0900]	E.html	A.html
		[25/Jul/2001:09:12:21+0900]	B.html	A.html
		[25/Jul/2001:09:13:16+0900]	G.html	B.html
4	210.106.81.115/ Mozilla/4.2(Win98.I)	[25/Jul/2001:11:26:22+0900]	A.html	-
		[25/Jul/2001:11:28:23+0900]	D.html	A.html

그림 3. 샘플로그의 세션 구분 결과

6. 결론 및 향후 연구과제

본 논문에서는 데이터 전처리 과정에서 서버측 클릭스트림 데이터를 사용하여 사용자와 세션을 구분하는 방법을 다루었다. 앞으로의 연구에서는 실제 시스템에 적용하여 실세계의 데이터를 가지고, 세션 보정과 트랜잭션 보정에 관한 연구가 필요하다.

또 웹 로그는 웹 서버에 사용자가 접근한 기록만을 저장하고 있기 때문에, 웹 로그를 통해 세션을 구한 경우 클라이언트 캐시 때문에 실제 웹 사이트의 구조에서 생성될 수 없는 세션이 발생할 수 있다. 이런 경우 세션을 보정할 수 있는 연구도 필요하다.

참고문헌

[1] R. Cooley, B. Mobasher, J. Srivastava. Web mining : Information and pattern discovery on the World Wide Web. In: *International Conference on Tools with Artificial Intelligence*, Newport Beach, CA, pp.558-567, 1997.

[2] M. S. Chen, J. S. Park, P. S. Yu. Data Mining for path traversal patterns in a Web environment. In: *Proc. 16 th International Conference on Distributed Computing Systems*, pp.385-392, 1996.

[3] H. Mannila, H. Toivonen, Discovering Generalized episodes using minimal ocurrences. In: *Proc. Second International Conference on Knowledge Discovery and Data Mining*, Portland, Oregon, pp.146-151, 1996.

[4] T. Yan, M. Jacobson, H. Gracia-Molina, U. Dayal. From user access patterns to dynamic hypertext linking. In : *Fifth International World Wide Web Conference*, Paris, France, 1996.

[5] J. Pitkow. In search of reliable usage data on the WWW. In : *Sixth International World Wide Web Conference*, Santa Clara, CA, pp.451-463, 1997.

[6] P. P. Bonissone and K. S. Decker. Selecting uncertainty calculi and granularity: An experiment in trading-off precision and complexity. *Uncertainty in Artificial Intelligence*, pp. 2217-2247, 1986.

[7] R. Cooley, B. Mobasher, J. Srivastava. Data Preparation for mining World Wide Web Browsing Patterns, In: *Knowledge and Information Systems 1*, Springer-Verlag, 00-00, pp. 1-26. 1999.