모바일 사전 응용의 반응속도 향상을 위한 선반입 기법

(A Prefetching Scheme for Improving the Response Time of Mobile Dictionary Applications)

옥 승 훈 † 김 기 성 † 김 형 주 ††
(Seunghun Ok) (Kisung Kim) (Hyoung-Joo Kim)

요 약 스마트 폰 및 태블릿 PC 등 다양한 모바일 기기들이 빠르게 보급되며 모바일 응용에 대한 관심이 증가하고 있다. 모바일 응용은 무선 인터넷 망을 통해 언제 어디서든 원하는 정보를 손쉽게 얻을 수 있는 장점을 제공한다. 그러나 모바일 응용은 모바일 기기의 제한된 자원과 상대적으로 느린 네트워크 속도로 인해 데스크탑 환경보다 느린 반응 속도를 보인다. 본 논문에서는 모바일 응용의 반응속도를 향상시키기 위해 선반입 기법을 적용하는 방안을 제시한다. 선반입이란 다음 질의를 미리 예측해 관련 있는 데이터를 미리 가져 오는 기법으로 기존의 웹 환경에서 많이 사용되고 있다. 본 논문에서는 모바일 사전 응용이라는 특정 도메인의 응용에 선반입을 적용하는 기법을 제안한다. 우선 예측 방법을 제안하기 위해 순서 정보를 고려하는 우선성 마이닝 기법과 협업 필터링을 비교, 분석한다. 그리고 사전 도메인에 맞는 예측 알고리즘으로 두 알고리즘을 혼합한 방법을 제안한다. 또한 네트워크 비용을 최소화하기 위한 2단계 선반입 기법을 제안한다. 2단계 선반입 기법이란 클라이언트 측 캐쉬와 프록시 서버 측 캐쉬를 활용하여 클라이언트 측에서 캐쉬 적중 실패가 발생하였을 때의 손실을 보완하는 기법이다. 마지막으로 시뮬레이션 데이터와 실제 데이터를 이용한 실험을 통해 제안한 기법이 모바일 사전 응용의 반응속도를 향상시킴을 보였다. 본 논문에서 제안한 기법들은 사전 도메인을 대상으로 했지만, 적절한 예측 방법을 선택하게 되면 다양한 모바일 응용에 적용할 수 있다.

키워드 : 모바일 응용, 선반입, 프록시 서버, 우선성 마이닝

Abstract As mobile devices such as smart phones and tablet PCs are getting popularity, mobile applications have attracted a lot of attentions. Mobile applications have advantages that they can provide information at any time and any place using mobile internet network. However mobile applications suffer from slower response time than desktop applications because of the limited resources of the mobile devices and relatively slow speed of the wireless network. In this paper, we propose a prefetching scheme to improve the response time of the mobile applications. The prefetching is a technique that transfers the next results in advance by predicting the next queries. The prefetching is widely used in web environments. In this paper, we propose to apply the prefetching techniques in a specific domain of the mobile applications; the mobile dictionary application. As prediction methods, we analyze and compare the precedence mining technique and the collaborative filtering and suggest the hybrid method of these two techniques. In addition, we propose a 2-level prefetching scheme in order to minimize network costs while improving the response time. The 2-level prefetching scheme uses a client-side cache and a proxy server-side cache to supplement the loss which are resulted from the client-side cache misses. Through experiments with the syntactic data

· 본 연구는 BK-21 정보기술 사업단의 연구결과로 수행되었음

† 비 회 원 : 서울대학교 컴퓨터공학부
shok@idb.snu.ac.kr
kskim@idb.snu.ac.kr
†† 종신회원 : 서울대학교 컴퓨터공학부 교수
hjk@snu.ac.kr

논문접수 : 2010년 12월 28일
심사완료 : 2011년 6월 29일

정보과학회논문지: 데이타베이스 제38권 제5호(2011.10)

and the real world data, we show that the proposed techniques can improve the response time of the mobile dictionary application. Although the proposed techniques are suggested in the dictionary domain, our techniques can be applied to other mobile applications with the adequate prediction methods.

Key words : Mobile applications, Prefetching, Proxy server, Precedence Mining

1. 서 론

최근 스마트 폰 및 태블릿 PC 등, 모바일 기기를 보유한 사용자의 수가 빠르게 증가하며 모바일 응용(application)의 개발이 활발해지고 있다. 이에 모바일 응용 시장을 새로운 소프트웨어 시장으로 주목 받으며 모바일 응용의 개발과 활용에 많은 관심을 가지게 되었다. 또한 스마트폰 개발 회사와 이동 통신사에서도 모바일 응용 시장에 관심을 두고 자신의 모바일 응용 마켓을 구축하여 활성화시키고자 하는 노력이 계속되고 있다. 대표적인 모바일 응용 마켓 중 하나인 애플의 앱스토어는 2010년 9월 1일을 기준으로 25만개 이상의 응용을 보유하고 있고 65억 건 이상의 다운로드 횟수를[1] 기록하고 있으며 현재에도 빠른 속도로 그 규모가 커지고 있는 상황이다.

이와 같이 빠르게 확산되고 있는 모바일 응용들은 기존의 데스크탑에서 수행하던 프로그램과 비슷한 기능을 모바일 환경에서도 사용할 수 있게 해준다. 현재 대부분의 모바일 응용들은 기존의 웹 사이트에서 정보를 가져와 모바일 환경에 적합하게 변경한 후 사용자에게 제공한다. 예를 들면, 날씨, 뉴스, 지도, 사전 등과 같은 모바일 응용들은 각 분야의 정보를 제공하는 웹 사이트에 접속해 정보를 가져오고 이를 가공하여 모바일 환경에 알맞게 보여주고 있다.

이와 같이 많은 수의 모바일 응용이 웹을 통해 정보를 제공하고 있지만 아직 검색 속도 면에서 많은 제약을 갖고 있다. 이는 모바일 기기가 상대적으로 느린 프로세서와 한정된 전원을 사용하고 있기 때문이기도 하지만 가장 큰 원인은 유선보다는 느린 무선 통신망을 이용해야 하는 제약이 있기 때문이다. 또한 정보를 모바일 기기 환경에 알맞게 가공하는 작업을 수행하는 경우에는 이로 이한 자원과 시간이 필요하게 된다. 이러한 이유들로 인하여 모바일 응용이 웹에 존재하는 정보를 이용할 때에는 기존의 데스크탑 환경에 비하여 느린 반응속도를 보이게 된다.

또한 웹 사이트에서 정보를 가져올 경우에는 네트워크 대역폭의 낭비도 발생한다. 웹 사이트에서 가져온 데이터는 대개 HTML이나 XML의 형식을 가지는데, 특히 HTML 형식의 경우 많은 부분은 모바일 응용에서 필요로 하지 않는 부분이 될 수 있다. 그러므로 모바일 응용은 가져온 데이터 중 필요한 부분을 추출하는 파싱 작업을 수행해야 한다. 이런 파싱 작업은 추가적인 비용과 시간을 소비하게 되고 이는 반응시간의 증가에 영향을 끼친다. 그리고 HTML 또는 XML 데이터를 그대로 웹사이트에서 가져오는 것은 불필요한 데이터를 모두 포함하여 가져오는 것이기 때문에 데이터 전송을 위한 네트워크 대역폭을 낭비하는 결과를 초래한다.

본 논문에서는 웹에 존재하는 데이터를 이용하는 모바일 응용에서 선반입(prefetching)을 적용하여 반응속도를 향상시키는 방안을 제시하고자 한다. 선반입은 앞으로 수행될 명령어나 사용될 데이터를 미리 가져와서 수행속도나 반응속도를 향상시키는 기법으로 마이크로프로세서의 명령어 선반입이나 웹 브라우저에서의 링크 선반입과 같이 널리 사용되는 기법이다. 하지만 이러한 시도들은 데스크탑 환경에서 풍부한 컴퓨터 자원과 빠른 네트워크를 기반으로 하고 있기 때문에, 상대적으로 열악한 환경인 모바일 환경에서 기존의 선반입을 그대로 적용하기에는 무리가 있다. 특히 모바일 환경에서는 선반입 된 정보가 사용되지 않았을 경우 선반입을 위해 소모된 자원이나 네트워크 대역폭이 낭비되는 결과를 가져온다. 모바일 환경에서는 사용한 네트워크 대역폭의 양은 사용자가 부담해야 할 요금과 직결되는 부분이므로 최대한 대역폭의 낭비를 줄이기 위한 노력이 필요하다.

본 논문에서는 사전 응용이라는 특정 도메인의 응용을 선택해 모바일 환경에서의 선반입 기법을 제안하고자 한다. 이는 사전 응용이 웹사이트를 통해 정보를 제공 받으며, 여러 단어를 연속으로 검색하는 특성을 갖기 때문에 선반입의 효과를 잘 보여줄 수 있기 때문이다. 그러나 본 논문에서 제안하는 방법은 사전 응용에만 적용 가능한 것이 아니라 다른 도메인의 응용에도 적용가능하다.

우리는 사전 응용에 적합한 예측 방법으로 협업 필터링 기법과 우선성 마이닝 기법을 적용했다. 협업 필터링은 기존의 추천 시스템에서 자주 사용된 기법으로 유사한 사용자의 활동을 통해 예측을 하는 기법이며 우선성 마이닝은 시간적 접근 패턴, 즉 우선성 정보에 기반하여 예측을 하는 기법이다. 본 논문에서는 2단계 선반입 시스템을 통해 이 두 예측 기법을 단계적으로 적용하여 예측의 효율성을 높이고자 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 웹 환경에서의 선반입 기법들에 대한 관련 연구와 본 논문에서

1) http://en.wikipedia.org/wiki/App_store

적용할 우선성 마이닝(precedence mining) 기법에 대한 관련 연구를 살펴본다. 3장에서는 본 논문에서 제시하는 선반입 기법에서 필수적인 요소인 단어 예측 기법에 대해서 논하고 4장에서는 이를 활용하는 2단계 선반입 시스템이 어떤 특징을 가지고 있는가에 대해 설명한다. 5장에서는 실험을 통해 제안한 시스템을 평가하며 마지막으로 6장에서는 전체적으로 결론을 맺으며 향후 발전 방향에 대해서 논한다.

2. 관련연구

[1]은 처음으로 월드 와이드 웹(WWW)에서 지연시간을 향상시키기 위해 예측에 기반한 선반입 기법을 적용한 논문이다. 이 연구에서는 웹 브라우저를 사용할 때 사용자가 웹 페이지를 탐색할 때 페이지를 이동할 때마다 발생하는 지연시간을 줄이고자 하였다. 선반입 할 페이지를 예측하는 방법으로는 기존의 페이지 링크의 클릭 히스토리를 조사하여 해당 페이지에서 어떤 웹 페이지로의 이동이 가장 많은지를 확률적으로 계산하고 높은 확률의 페이지를 미리 다운로드 함으로써 지연시간을 감소시킨다. 이후로도 선반입에 대한 연구는 지속되어 왔다[2,3].

[4,5]에서는 모바일 환경에서 웹을 이용할 때 선반입을 적용하는 방안에 대하여 연구하였다. 특히 [4]에서는 모바일 환경에서 네트워크를 이용하는 데에 추가적인 비용이 필요함을 인지하고 비용에 대해 선반입의 임계값을 정하는 모듈을 제안하였다.

선반입을 적용함에 있어서 항상 중요시되는 것은 예측의 정확성에 관한 부분이다. 웹의 발달로 많은 사용자들이 정보를 생산하게 되었고 이를 통해서 다양한 예측이나 추천 기법들에 대한 연구가 이루어졌다. 특히 데이터마이닝 분야에서 연관성 규칙 마이닝이나 협업 필터링과 같은 기법들이 온라인 전자상거래와 같은 사이트에서 사용자의 구매 활동을 돕는 추천 기법으로 많이 활용되었다.

협업 필터링에 대한 연구는 [6]에서 처음 제안된 이후 많은 연구가 이루어져 왔다. 이 연구에서는 사용자의 과거의 행동에 기반하여 유사한 아이템을 추천하는 내용기반 추천 기법(content-based recommendation)과 사용자와 유사한 경향을 지니는 다른 사용자들이 선택한 아이템들을 추천하는 협업 추천 기법(collaborative recommendation)을 소개하고 있다. 이후 협업 필터링은 많은 추천 분야에서 활용되어 왔으며 추천 알고리즘 또한 지속적으로 발전되어 왔다[7]. 특히 [8]에는 협업 필터링이 아마존과 같은 온라인 전자상거래 웹 사이트에서 사용자가 상품을 구매할 때 유용한 정보를 줄 수 있음을 보여주었다.

반면 협업 필터링에서 더 발전된 개념의 추천 기법에 대하여 많은 연구가 이루어져 왔는데, [9]에서는 사용자의 선호도를 확장하는 방안으로 특정 아이템에 대한 시간적 접근 정보를 활용하였다. 여기서의 시간적 접근 정보는 히스토리에 기록된 데이터들의 기록 순서를 고려하는 것이다. 어떠한 아이템이 특정 아이템에 비해 먼저 나타나거나 혹은 나중에 나타날 확률을 구하는 우선성 마이닝을 통해 시간적 순서가 중요한 분야에서 추천의 질을 높이고자 하였다. 기존의 [10]과 같은 연구들이 시간적 정보를 고려하던 것에 비해 우선성 마이닝은 간단한 우선성 정보만 고려함으로써 더 적은 비용으로 추천 기법을 적용할 수 있는 장점이 있다.

본 논문에서는 위의 두 가지 기법을 이용해 사용자가 사전에서 검색할 단어를 예측한다. 본 논문의 핵심인 선반입 기법에서 가장 중요한 요소인 단어 예측의 정확도를 높이기 위해 두 추천 기법을 비교하고 분석하여 모바일 응용의 반응속도를 향상시키는 데에 기여함을 보인다. 또한 본 연구에서는 기존 선반입 기법과는 달리 이 두 가지 예측 기법을 혼합한 2단계 선반입 구조를 제안한다. 기존의 선반입에서는 고려 하지 않은 프록시 서버를 사용한 2단계 선반입 기법을 제안함으로써 모바일 환경에서 보다 적합한 선반입 기법을 제안한다.

3. 단어 예측 기법

이 장에서는 모바일 응용, 특히 모바일 사전 응용에서의 반응속도를 개선하기 위해 선반입을 적용함에 있어서 가장 중요시되는 단어 예측 기법에 관하여 살펴본다. 선반입 기법에서 가장 핵심이 되는 부분은 예측 알고리즘이다. 선반입의 목적은 다음에 사용할 데이터를 미리 가져와서 미래의 검색 속도를 향상시키는 것이며 이 목적을 달성하기 위해서는 미래에 사용할 데이터를 예측하여야 한다. 이를 위해 우선 모바일 사전 응용에서의 사용자 검색 패턴의 특징에 대해 소개하고 이런 특징들을 활용한 단어 예측 기법에 대해서 알아본다.

3.1 사용자 검색 패턴의 특징

본 논문에서는 사전 응용을 사용하는 사용자의 검색 패턴은 다음과 같은 특징을 갖고 있음을 가정한다.

- 사전 검색은 특정 문서나 책을 읽을 때 발생한다. 따라서 문서나 책에서 나열되어 있는 단어의 순서와 사전에서 검색한 단어의 순서가 같거나 비슷해지는 경향이 있다. 따라서 문서에 발생한 단어의 순서 정보를 이용해 단어 예측을 할 수 있다.
- 한 사용자가 검색한 단어는 일정한 주제를 가진다. 온라인 쇼핑몰에서 사용자에게 상품을 추천할 때 사용자의 프로필을 활용하듯이 사전에서는 한 사용자가 이전에 검색했던 히스토리가 그 사용자의 현재 관심

분야를 나타낼 수 있다. 예를 들어 토익 공부를 하는 사용자와 해리 포터 소설을 읽으며 사전 검색을 하는 사용자는 서로 다른 패턴의 검색 히스토리를 갖게 된다. 따라서 사용자의 검색 패턴과 유사한 히스토리를 가진 다른 사용자의 히스토리를 이용하면 단어를 추천하는 것이 가능하게 된다.

3.2 예측 알고리즘

선반입 기법은 대부분 사용자의 히스토리에 기반해 예측을 수행한다. 예를 들어, 기존의 웹 페이지 선반입과 같은 경우에는 웹 페이지 방문 기록을 분석하여 페이지 간의 이동 가능성을 확률적으로 계산하였다. 본 논문에서도 이런 히스토리에 기반한 예측 기법을 사용한다.

사전 응용의 히스토리는 다음과 같이 구성한다.

- 단어 검색이 이루어진 세션
- 단어 검색 시간
- 검색 단어

세션은 클라이언트와 서버가 연결되었을 때, 그 연결을 구분 짓는 식별자 역할을 한다. 그러므로 세션이 동일한 일련의 단어 목록들은 사용자가 일련의 단어 검색을 수행한 활동을 나타낸다. 시간 정보는 같은 세션 내에서 단어가 검색된 순서를 파악하기 위해 사용한다. 이는 앞으로 적용하게 될 우선성 마이닝에서 중요한 정보가 된다.

이와 같은 히스토리 정보를 이용한 예측 기법으로 협업 필터링 방법과 우선성 마이닝 기법을 사용한다. 두 기법 모두 기존에 추천 시스템에서 많이 사용되는 기법으로 히스토리 정보를 사용하고 있다. 그러나 두 기법은 서로 다른 특성을 갖고 있으며 본 논문에서는 예측의 정확성을 높이기 위해서 두 기법을 조합하는 방식을 택하였다. 3.2.1절과 3.2.2절에서 두 기법에 대한 설명을 제시하고 3.2.3절에서는 구체적인 예를 통해 어떻게 사전 검색에 적용하였는지 설명한다. 그리고 3.2.4절에서는 두 기법이 단어 예측에서 갖는 의미를 비교 설명하도록 한다.

3.2.1 협업 필터링 개괄

사전 응용에서의 협업 필터링을 이용한 추천의 경우 해당 단어를 검색한 다른 사용자들의 히스토리들로부터 공통적으로 자주 나타나는 단어를 추천한다. 본 논문에서 사용하는 히스토리 데이터에는 사용자를 구별할 수 있는 기준이 존재하지 않는다. 단지 세션으로만 일련의 단어 검색 목록을 구분하여 사용하고 있는데 같은 사용자라도 다른 시간, 다른 장소에서 단어 검색을 수행했다면 다른 세션으로 나타나게 되므로 사용자의 히스토리를 누적하여 그 사용자의 검색 경향을 파악하는 것과 같은 작업을 수행하지 못한다. 하지만 굳이 특정 사용자별 검색 히스토리를 알지 못하더라도 협업 필터링을 활용한 추천에는 크게 지장을 주지 않는다. 각각의 세션을 모두 다른 사용자로 간주하더라도 협업 필터링을 적용하면 주어진 단어에 대해 추천을 하고자 할 때 주어진 단어를 포함하는 세션을 모두 검색하고 해당 세션들에서 공통적으로 자주 나타나는 단어를 추천함으로써 협업 필터링을 활용한 추천을 수행할 수 있다.

협업 필터링의 추천 결과는 지지도(support)와 신뢰도(confidence)의 값으로 접근 확률을 결정한다. 지지도와 신뢰도의 개념은 연관성 규칙 분석 기법(association rule mining)에서 사용된 것이지만 본 논문에서 협업 필터링을 통한 추천에도 적용될 수 있다. 사용자가 A라는 단어를 검색하였을 때 협업 필터링을 통해 B라는 단어를 추천하기 위해서는 Support(A➔B) 그리고 Confidence(A➔B) 값을 구한다.

$$\text{Support}(A \rightarrow B) = \text{Support}(AB) = \text{Support}(A \text{ union } B)$$

$$\text{Confidence}(A \rightarrow B) = \frac{\text{Support}(AB)}{\text{Support}(A)}$$

$$\text{Support}(X) = \frac{\text{Count}(X)}{|D|}$$

여기서 D는 전체 아이템 집합을 말한다. Support(X) 값은 전체 집합 중에서 X를 포함하는 집합의 비율이다. Support 값과 Confidence 값의 가장 큰 차이점은 단지 두 아이템이 함께 나타날 확률과 한 아이템이 나타났을 때 나머지 아이템이 나타날 확률이라는 차이이다. 본 논문에서 다양한 검색 히스토리에 대하여 지지도 값과 신뢰도 값을 구해보았는데 두 값에 의한 랭킹에는 큰 차이가 없었다. 구별되는 차이점은 전체 히스토리에서 나타나는 빈도가 극히 적은 아이템에서 나타난다. 예를 들어 {google, android} 라는 검색 단어 집합이 전체 1000개의 히스토리 중에서 단 한번 나타났고 “google”, “android”라는 단어도 단 한번씩만 나타났다고 가정한다. 이 경우 Support(google➔android) = 0.1% 이고 Confidence(google➔android) = 100% 이다. 이러한 결과를 볼 때 지지도는 해당 아이템들의 출현 빈도를 우선시하고 신뢰도는 해당 아이템의 동시 출현 여부를 우선시함을 알 수 있다. 기존의 온라인 전자상거래 사이트에서와 같은 경우에는 비교적 많이 판매된 물품을 추천하는 것이 일반적이므로 지지도 값이 아주 중요한 요소로 작용한다. 하지만 사전 응용에서는 지지도 값이 작다고 해서 추천을 하지 않는 것은 바람직하지 못하다. 왜냐하면 사전이라는 분야의 특성상 의미가 어렵고 평소에 자주 사용되지 않는 단어의 경우에는 전체 히스토리 상에서 나타나는 빈도가 매우 적을 것이다. 평소에 자주 사용되지 않는 단어에 대해서도 추천은 이루어져야 하므로 신뢰도 값을 이용하면 지지도 값을 이용한 추천의 경우보다 더 만족스러운 결과를 얻을 수 있다. 따라서 사전 응용

에서는 신뢰도 값을 이용하여 추천을 하는 것이 올바른 선택이다.

3.2.2 우선성 마이닝 개괄

본 논문에서는 단어의 순서 정보를 활용하여 추천의 정확도를 높이고자 우선성 마이닝 기법[9]을 활용한다. 우선성 마이닝기법은 순서 정보가 있는 상황에서 추천을 통해 사용자의 의사 결정을 돕기 위한 기법이다. [9]에서는 우선성 마이닝을 수행하는 알고리즘으로 여러 가지를 소개하였는데 우리는 그 중에서 Single Item Max-Confidence(SingleMC)라는 알고리즘을 사용한다. SingleMC 알고리즘은 결과로 나타난 아이템과 그 아이템이 나타나도록 한 원인이 되는 아이템 사이에는 필요 관계가 존재한다는 원칙을 활용한다.

SingleMC 알고리즘을 통해 아이템을 추천하는 방법은 다음과 같다. 우선 일련의 검색된 단어 집합을 I라고 정의한다. I = {a, b, …, z} 라고 할 때, I에 속한 모든 단어들은 같은 세션에 속한다. 즉 I는 특정 시간에 특정 주제에 대하여 사용자가 검색한 단어들의 집합이다. 그리고 a는 해당 집합 내에서 가장 먼저 검색되었던 단어이며 z는 가장 나중에 검색되었던 단어이다. 단어를 추천하기 위해 추천될 단어의 점수가 필요한데 이를 Single Item Max-Confidence Score라고 지칭하고 있다. Single Item Max-Confidence Score는 다음과 같이 정의된다.

$$score(x) = \max(confidence(y, x) | y \in set\ of\ I)$$

여기서 x는 현재 추천하고자 하는 단어이다. score(x) 함수가 적용될 수 있는 모든 단어에 대해서 score(x)의 값을 구하고 그 중 가장 높은 값을 가지는 단어가 가장 추천될 가능성이 높은 단어이다.

$$confidence(y, x) = \frac{g(y, x)}{f(y) - g(x, y)}$$

confidence(y,x)는 우선성에 기반한 신뢰도를 나타내는 값이다. 위의 score(x) 식에서 신뢰도가 가장 높게 나타나는 값을 취한다. confidence(y,x) 식에서 표현된 함수들의 의미는 다음과 같다.

- f(x): x가 포함되어 있는 단어 집합의 개수, 즉 x가 검색된 세션의 개수
- g(x,y): x가 y보다 먼저 검색된 단어 집합의 개수

confidence(y,x) 함수의 의미는 y가 나타나는 단어 집합들 중에서 x가 먼저 나타난 경우를 제외하고 x가 나중에 나타나는 경우의 비율을 나타낸다.

위에서 소개한 SingleMC 알고리즘을 사용하여 단어의 순서를 고려한 추천이 가능하다. 사전에서와 같이 단어 검색의 순서가 의미를 지니는 상황일 때, 우선성 마이닝을 통한 추천은 협업 필터링을 통한 추천보다 정확도가 높다.

3.2.3 단어 예측의 예시

우선성 마이닝과 협업 필터링은 그 특성의 차이로 인해 추천된 단어가 다를 뿐 아니라 추천된 단어의 개수에도 차이를 보이게 된다. 아래와 같이 검색된 히스토리를 가정해 보자.

{smartphone, apple, iphone, ipad}

위의 히스토리만 존재한다고 가정하였을 때, 우선성 마이닝을 통해서 추천할 수 있는 단어는 다음과 같다.

- smartphone → {apple, iphone, ipad}
- apple → {iphone, ipad}
- iphone → {ipad}
- ipad → { }

위의 경우와 같이 히스토리 상에서 이후에 검색된 단어가 있는 경우 추천을 할 수가 있지만 해당 단어 이후로 검색된 단어가 없다면 우선성 마이닝으로는 추천해 줄 수 있는 단어가 존재하지 않는다. 위의 예에서 보듯이 "ipad", "iphone"와 같이 항상 모든 히스토리 상에는 마지막 단어가 존재하며 또한 우선성 마이닝으로 추천해줄 수 있는 단어가 적은 경우도 다수 존재한다.

반면에 같은 히스토리에 대해서 협업 필터링으로 추천할 수 있는 단어는 다음과 같다.

- smartphone → {apple, iphone, ipad}
- apple → {smartphone, iphone, ipad}
- iphone → {smartphone, apple, ipad}
- ipad → {smartphone, apple, iphone}

이와 같이 협업 필터링에서는 히스토리 상에서 함께 출현했던 단어를 모두 추천하므로 추천된 단어의 개수가 우선성 마이닝의 경우보다 평균적으로 더 많다. 추천된 단어의 개수가 많을 경우 실제 사용자가 검색할 단어가 존재할 확률이 높아지므로 추천의 재현율(recall)을 높이는데 도움이 된다. 반면 추천 단어의 개수에 비하여 실제 사용자가 검색한 단어의 비율은 낮게 되므로 추천의 정확도(precision)는 우선성 마이닝의 경우보다 낮게 나타나는 경향이 있다.

3.2.4 예측 기법의 비교

협업 필터링은 최근 다양한 방면에서 활용되고 있는 추천 기법으로 사용자간의 유사성을 기반으로 하여 추천을 하고 있다. 하지만 협업 필터링은 추천을 결정하는 요소로 추천할 아이템의 출현 빈도만을 고려하고 있다. 추천의 정확도 향상을 위해서는 출현 빈도 이외의 요소를 고려하는 것이 필요하다.

우선성 마이닝은 사용자들의 아이템에 대한 시간적 접근 패턴을 활용하는 추천 기법이다. 특히 사전 응용에서는 단어 검색을 수행할 때마다 시간적 정보가 생성되게 된다. 이러한 시간적 정보를 활용하여 사전 응용에서 더욱 정확한 단어 예측이 가능하게 된다. 하지만 우선성

마이닝은 시간적 정보에서도 특히 우선성 정보, 즉 순서 정보에 기반을 두고 있다. 그러므로 3.2.3절에서 살펴본 바와 같이 기존에 히스토리에서 나타났던 순서와 다른 단어는 추천에서 제외시키는 경향이 있다.

아래의 표 1은 앞에서 언급한 두 가지 예측 기법의 특징을 비교한다.

이와 같이 협업 필터링과 우선성 마이닝은 예측의 기반이 되는 정보뿐만 아니라 정확도와 재현율에서도 차이를 보이고 있다. 본 논문에서는 위 두 가지 예측 기법을 상호보완적으로 적용하여 전체적인 예측의 효율을 높이고자 한다. 이 두 기법을 조합하는 방법은 다음 절에서 설명하도록 한다.

4. 2단계 선반입 기법

3절에서 본 바와 같이 사전 응용에 적합한 예측 기법인 협업 필터링과 우선성 마이닝은 서로 장단점을 지니고 있다. 우리는 각 예측 기법의 장단점을 보완하기 위해 2단계 선반입 기법을 제안한다.

4.1 시스템 구조

2단계 선반입 기법이란 모바일 사전 응용을 서버-클라이언트 구조에서의 클라이언트로 간주하였을 때, 프록시 서버와 클라이언트, 두 위치에서 선반입을 수행하는 기법이다. 프록시 서버는 클라이언트가 자신을 통해서 다른 네트워크 서비스에 간접적으로 접속할 수 있게 해주는 컴퓨터나 응용 프로그램을 가리킨다. 서버와 클라이언트 사이에서 중계기로서 대리로 통신을 수행하는 기능을 가리켜 '프록시', 그 중계 기능을 하는 것을 프록시 서버라고 부른다.

2단계 선반입 기법에서 프록시 서버는 크게 세 가지 역할을 한다. 첫 번째는 웹 프록시에서와 같이 사용자가 웹 사이트로부터 가져올 정보를 캐쉬에 저장하여 정보에 대한 접근을 빠르게 한다. 두 번째는 반응속도 향상을 위한 선반입을 수행한다. 특히 선반입에서 가장 중요한 것은 정확한 예측인데 본 논문에서는 사전 도메인에서의 정확한 예측을 위해 협업 필터링과 우선성 마이닝을 상호보완적으로 적용하여 예측을 수행한다. 세 번째는 웹 사이트에서 획득한 데이터의 가공이다. 데스크탑 환경에 알맞게 만들어진 HTML 문서와 같은 경우 모바일 응용에게는 불필요한 정보를 많이 포함하고 있다. 프록시 서버에서 필요한 정보만을 파싱하는 작업을 수행하여 모바일 응용의 작업 부하를 줄여 준다.

본 논문에서 제안하는 2단계 선반입 기법은 그림 1과 같다.

2단계 선반입 기법이란 선반입한 아이템들을 2단계에 걸쳐 캐쉬에 저장하는 기법이다. 1차적으로 클라이언트 측 캐쉬에 주된 아이템들을 저장하고 나머지 아이템들을 프록시 서버 측 캐쉬에 저장하여 2차적으로 접근 가능하도록 설계되었다. 클라이언트 측 캐쉬는 상대적으로

표 1 협업 필터링과 우선성 마이닝의 비교

	협업 필터링	우선성 마이닝
예측의 기반 정보	사용자의 유사성	사용자 활동의 시간적 접근 패턴
예측의 재현율(recall)	높음	낮음
예측의 정확도(precision)	낮음	높음

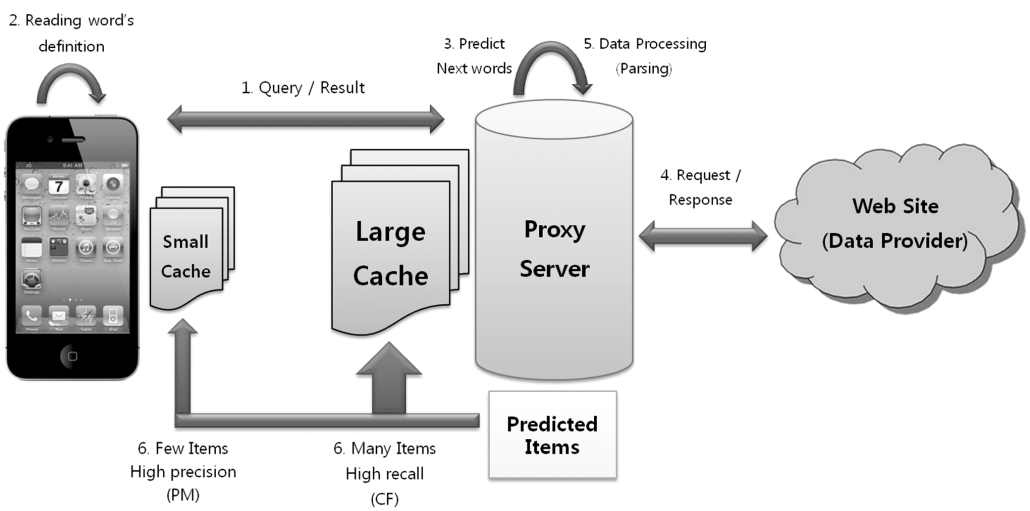


그림 1 2단계 선반입 기법의 구조

작은 크기를 가지고 있기 때문에 접근 확률이 높은 아이템들을 저장하도록 한다. 반면에 프록시 서버 측 캐쉬는 보다 큰 크기를 저장할 수 있기 때문에 많은 개수의 아이템들을 저장하도록 한다.

위에서 언급한 바와 같이 2가지 캐쉬의 특징이 구별되므로 각각의 특징에 맞게 선반입을 적용하는 것이 중요하다. 3장에서 언급했던 예측 기법의 특징을 고려하여 각 캐쉬의 특징에 맞는 예측 기법을 선택할 수 있다. 먼저 크기가 작고 접근성이 높은 클라이언트 측 캐쉬에는 우선성 마이닝을 통해 얻어진 정확도가 높고 개수가 적은 예측 단어들을 저장한다. 반면 크기가 크고 상대적으로 접근성이 낮은 프록시 서버 측 캐쉬에는 협업 필터링을 통해 얻어진 재현율이 높고 개수가 많은 예측 단어들을 저장한다.

결과적으로 2단계 선반입 기법을 적용하면 2번의 캐쉬 적중 가능성으로 인해 반응속도가 향상된다. 그림 2에서 볼 수 있듯이 화살표 1과 같이 클라이언트 측 캐쉬에서 캐쉬 적중이 발생하였을 때 가장 빠른 반응속도를 보이고 그 다음 화살표 2와 같이 프록시 서버 측 캐쉬에서 캐쉬 적중이 되었을 때는 클라이언트 측 캐쉬 적중의 경우보다는 느리지만 실제 웹 사이트에서 데이터를 가져오는 경우보다 빠른 반응속도를 보인다. 화살표 3은 2단계 선반입 기법을 통해서 캐쉬 적중이 발생하지 않았을 경우 실제 데이터를 웹 사이트에서 가져오는 경우이다. 이는 선반입 기법을 사용하지 않는 경우와 일치하며 모바일 응용은 느린 반응속도를 보이게 된다.

4.2 2단계 선반입 기법의 수행

3절에서 사용자가 검색한 단어에 대해서 미래에 검색할 것으로 예상되는 단어를 추천하는 방법에 대하여 설명하였다. 이번 절에서는 이 기법들을 어떻게 2단계 선반입에서 사용하는지 전체적인 선반입 과정을 설명한다.

2단계 선반입 기법에서 첫 번째로 클라이언트 측 캐쉬에 저장할 단어들을 선출한다. 클라이언트 측 캐쉬는 크기가 작고 클라이언트 측으로 데이터를 전송하는 비용이 필요하게 되므로 가장 접근 확률이 높은 소수의 단어들만을 저장하도록 한다. 프록시 서버에서 우선성 마이닝을 통해 SingleMC Score 를 측정한 다수의 단어들 중 가장 점수가 높은 상위 k개의 단어를 선택한다 [11]. 우선성 마이닝은 실시간으로 수행되지 않고 프록시 서버에서 일정한 주기를 가지고 미리 수행하여 결과 점수를 데이터베이스에 저장한다. 데이터베이스 상에 존재하는 많은 데이터들로부터 우선성 마이닝을 수행하는 것은 상당히 부하가 큰 작업이다. [12]에서는 관계형 데이터베이스에서 많은 연관성 규칙 마이닝을 수행하는 방법에 대해서 소개하였다. 이 연구에서와 같이 우선성 마이닝을 수행할 때에도 테이블간의 많은 조인 연산이 수행되기 때문에 실시간으로 우선성 마이닝을 수행하는 것은 힘든 작업으로 보여진다. 그러므로 프록시 서버는 일정한 주기마다 우선성 마이닝을 수행하고 단어 검색이 요청되었을 때에는 마이닝을 통해 구해진 결과를 데이터베이스에서 빠르게 불러와서 사용자에게 제공할 수 있도록 한다. 실제 클라이언트로부터 단어 입력이 이루어졌을 경우 해당 단어에 대한 추천 단어들의 점수들을 데이터베이스로부터 불러온다. 이 때 데이터베이스에 접근하는 시간은 오래 걸리지만 이는 사용자가 단어의 정보를 확인하는 동안 프록시 서버에서 이루어지는 작업이므로 실제 사용자측의 반응속도에는 영향을 끼치지 않는다. 프록시 서버에서 상위 k개의 추천 단어들을 클

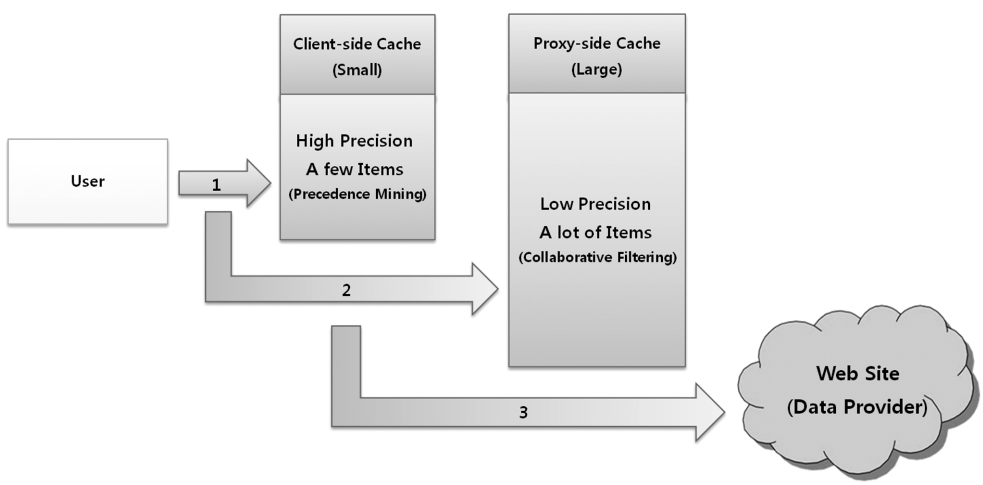


그림 2 2단계 선반입 기법을 통한 반응속도 향상

라이언트로 전송하고 전송된 단어들은 클라이언트 측 캐쉬에 저장된다.

두 번째로는 프록시 서버 측 캐쉬에 저장할 단어들을 선출한다. 앞에서 언급한 바와 같이 프록시 서버 측 캐쉬는 크기가 크고 클라이언트 측 캐쉬에서의 실패를 보완하는 역할을 한다. 그러므로 최대한 많은 단어들을 저장하는 것이 중요하다. 추천 개수를 고려하였을 때 협업 필터링을 통한 추천이 우선성 마이닝을 통한 추천보다 프록시 서버 측 캐쉬에 더욱 적절하다. 그리고 추천의 정확도를 고려하였을 때 협업 필터링의 결과 중에서 지지도 값보다 신뢰도 값을 선택하는 것이 극히 빈도가 적은 단어의 경우에도 질 좋은 추천을 할 수 있다. 그러므로 프록시 서버 측 캐쉬에 저장한 단어를 추천하기 위해서는 협업 필터링을 통해 단어들의 신뢰도 값을 측정하고 상위 k개의 단어를 선출하여 프록시 서버 측 캐쉬에 저장한다. 프록시 서버 측 캐쉬에는 많은 단어 정보를 저장해야 하는데 클라이언트 측 캐쉬처럼 데이터를 전송해야 할 부담이 없으므로 많은 정보 저장에도 부담이 없다.

각각의 캐쉬에 저장된 정보들은 실제 사용자의 단어 검색이 이루어졌을 때 우선적으로 탐색된다. 먼저 클라이언트 측 캐쉬에서 사용자가 검색을 요청한 단어가 있는지 검사하고 캐쉬 적중이 이루어졌을 경우 빠른 반응속도로 사용자에게 단어 정보를 제공한다. 클라이언트 측 캐쉬가 적중되지 않았을 경우 2차적으로 프록시 서버 측 캐쉬에서 요청 단어의 존재 여부를 검사하고 캐쉬 적중이 이루어졌을 경우 프록시 서버 측 캐쉬에 저장되어 있는 정보를 전송하여 사용자에게 제공한다. 이 경우 프록시 서버와의 통신은 필요하지만 프록시 서버 측의 캐쉬에 저장된 정보를 접근하므로 많은 시간이 걸리지 않고 저장된 정보는 이미 프록시 서버가 파싱과 같은 전처리를 완료한 상태이기 때문에 실제 HTML이나 XML 문서에 비해 용량이 작아 전송 시간도 짧으며 모바일 응용에서 다시 데이터 처리를 수행할 필요가 없다. 이러한 장점들로 인해 클라이언트 측 캐쉬 적중이 실패하더라도 프록시 서버 측 캐쉬를 통해 이를 보완함으로써 전체적인 모바일 응용의 반응속도를 향상시킨다.

캐쉬를 사용할 때 고려해야 할 사항 중 하나는 캐쉬 교환 정책이다. 캐쉬가 가득 차있는 상태에서 새로운 데이터를 캐쉬에 저장하고자 할 때 기존 존재하는 데이터 중 어떤 것과 교체를 해야 하는 지를 결정해야 한다. 사전 응용에서는 최소 최근 사용(LRU, least recently used) 알고리즘과 최다 최근 사용(MRU, most recently used) 알고리즘을 결합한 방식으로 캐쉬를 교환하는 것이 가장 효율적이다. 왜냐하면 사전 검색의 특성상 사용자가 이미 검색한 단어를 다시 검색하는 확률은 극히 적다. 그리고 캐쉬에 저장된 데이터의 대부분은 추천 단어인데 과거에 추천된 단어보다 현재 추천된 단어가 이후에 사용될 확률이 훨씬 높다. 그러므로 2단계 선반입 기법에서 사용되는 캐쉬에서는 캐쉬에 가장 오랫동안 유지된 단어들과 사용자가 이미 검색한 단어를 캐쉬에서 제거하는 방식으로 캐쉬를 교환하여 캐쉬의 높은 적중률을 유지한다.

5. 성능평가

이 장에서는 제안한 2단계 선반입 기법을 사용하였을 때 모바일 응용의 반응속도 향상이 이루어짐을 실험을 통해 보이고자 한다. 실험을 위한 환경 설정과 실험을 위해 사용된 데이터들에 대하여 소개하며 본 논문에서 적용된 우선성 마이닝과 협업 필터링을 통한 추천 기법의 차이점에 대해서 다룬다.

5.1 실험 환경 및 실험 데이터

2단계 선반입 시스템에서 사용되는 프록시 서버는 JAVA 버전의 Google App Engine[2] 1.3.8 이용하여 구현되었다. Google App Engine은 구글의 클라우드 컴퓨팅 환경을 활용하여 웹 서버, 데이터베이스, 분산 처리 등을 사용할 수 있는 PaaS(Platform as a Service) 서비스이다. 사전 단어 검색 히스토리를 저장하는 데이터베이스 시스템은 Google App Engine에서 제공하는 Google Datastore를 사용하였다. Google Datastore는 구글 파일시스템 상에서 Bigtable을 기반으로 한다. 개발 환경으로는 JAVA 1.6을 기반으로 하고 Eclipse Helios 버전의 IDE를 사용하였다. 그리고 모바일 응용에서의 실험을 위한 환경은 ARMv7-720MHz의 CPU와 315MB RAM, 운영체제로 Android 2.2 Froyo 버전을 사용하는 삼성전자의 Galaxy A 제품을 사용하였다. 모바일 응용 개발을 위해서는 JAVA와 Eclipse 환경 위에서 Android 2.2 SDK API 8 버전을 사용하였다. 마지막으로 사전 응용에서 사용되는 사전 데이터는 http://www.wordnik.com에서 제공되는 데이터를 사용하였다.

본 논문에서는 2단계 선반입 기법에서 단어 예측을 위해 사용자들이 검색한 단어의 히스토리를 이용한다. 하지만 개인 정보 보호의 정책에 의해서 모든 온라인 사전 사이트에서 히스토리 정보를 제공하지 않는 실정이다. 그렇기 때문에 현실에 존재하는 단어 검색 히스토리가 아닌 직접 가상적으로 생성한 히스토리를 사용하였다. 우리는 사전 검색이 주로 책이나 문서를 읽을 때 이루어진다고 가정하였으므로 실제 웹 문서에 대해서 단어를 검색하고 그 히스토리를 생성하기로 하였다. 모든 웹 문서를 대상으로 하게 되면 대상 주제가 너무 광범위해

2) http://code.google.com/intl/ko-KR/appengine/

서 우리는 http://www.itnews.com를 대상으로 히스토리를 생성하였다. 해당 사이트는 IT 관련 뉴스들을 제공하는 웹 사이트이다. 히스토리를 생성하기 위해 우선적으로 해당 사이트에서 뉴스 문서들을 수집하였다. 뉴스 문서를 수집하기 위해 뉴스 문서가 포함된 총 5226개의 HTML 문서들을 크롤링하였다. 크롤링된 문서들에서 실제 뉴스 기사부분을 추출하기 위해 HTML 태그로 구분된 부분들을 파싱하였다. 추출된 5226개의 뉴스 기사에서 단어들의 출현 빈도를 측정하기 위해 모든 단어를 추출하고 해당 단어의 출현 횟수를 측정하였다. 사용자가 실제 사전에서 검색하는 단어의 경향을 고려하면 “a”, “an”, “the” 와 같은 관사나 “one”, “man”, “hello” 와 같이 거의 모든 사람이 의미를 알고 있는 단어는 사전을 통해 검색하지 않을 것으로 보여진다. 따라서 이러한 단어들을 제외한 나머지 단어들로부터 IT 분야와 관계된 것으로 보이는 531개의 단어를 추출하여 해당 단어들을 실제 사용자가 사전에서 검색한 단어들로 간주하였다. 각 뉴스 기사에서 해당 단어들이 출현하면 검색 리스트에 저장하는 방식으로 5226개의 검색 단어 집합을 생성하였다. 그리고 뉴스 기사에서의 단어 출현 순서에 따라 시간 차이를 두어서 실제 히스토리에 저장될 때에도 이러한 출현 순서가 반영되도록 하였다. 생성된 히스토리는 데이터베이스에 저장되었고 데이터베이스 상에서 선반입을 위한 데이터마이닝이 수행되었다.

5.2 실험 결과

제안한 방법의 성능을 평가하기 위해 2가지 측면에서 실험을 수행하였다. 첫 번째는 실제 모바일 사전 응용에서의 반응속도 측정이고 두 번째는 2단계 선반입 시스템에서의 캐쉬 적중 확률의 측정이다.

모바일 사전 응용에서 반응속도를 측정할 때에는 2단계 선반입 기법을 사용하지 않았을 때와 2단계 선반입 기법을 사용하였을 때의 두 가지로 구분하여 반응속도를 측정한다. 그리고 2단계 선반입 기법을 사용하지 않았을 경우에도 웹 사이트로부터 가져오는 데이터가 HTML 문서일 때, API를 사용하여 XML 문서를 가져올 때를 구분하여 반응속도를 측정한다. 또한 2단계 선반입 기법을 사용하였을 경우에는 클라이언트 측 캐쉬에서 캐쉬 적중이 발생하였을 때, 프록시 서버 측 캐쉬에서 캐쉬 적중이 발생하였을 때를 구분하여 반응속도를 측정한다.

캐쉬 적중 확률의 측정에서는 예측을 위해 우선성 마이닝과 협업 필터링을 사용하였을 때의 캐쉬 적중률을 비교한다. 다양한 상황에서의 캐쉬 적중률을 구하기 위해 선반입 크기를 달리 하였을 때의 결과를 비교하며 또한 캐쉬 크기를 달리 하였을 때의 결과도 비교한다.

5.2.1 반응속도

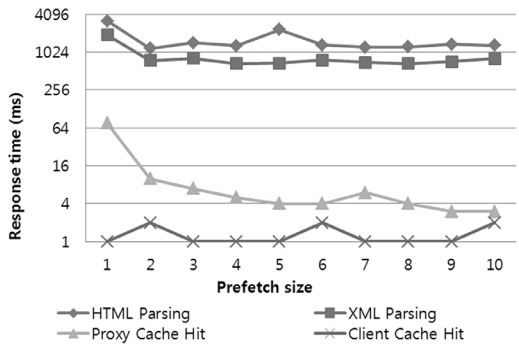


그림 3 선반입 유무에 따른 반응속도 (3G)

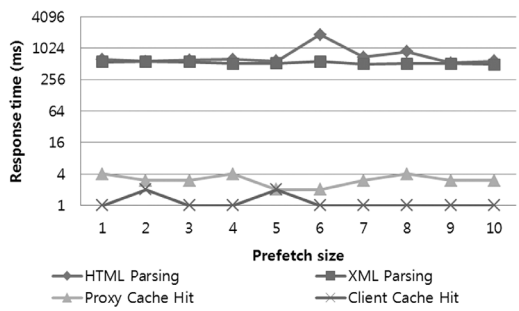


그림 4 선반입 유무에 따른 반응속도 (Wi-Fi)

먼저 반응속도를 측정하는 실험은 실제 스마트폰 (Samsung Galaxy A) 상에서 이루어졌다. 모바일 네트워크의 속도에 의한 반응속도의 차이가 있을 수 있으므로 스마트폰이 모바일 네트워크로 3G 네트워크를 사용할 경우와 Wi-Fi 네트워크를 사용할 경우를 구분하여 실험하였다.

그림 3과 그림 4를 비교해보았을 때 전체적인 반응속도는 Wi-Fi 네트워크 상에서 더 빠르다는 것을 확인할 수 있다. 이것은 두 네트워크의 속도 차이에 기인하는 것으로 3G 네트워크는 HSDPA 기술을 이용하여 이론적으로 최대 7Mbps의 전송 속도를 낼 수 있으며 Wi-Fi (802.11g)는 이론적으로 최대 54Mbps의 전송 속도를 낼 수 있다. 하지만 실제적으로는 최대의 속도는 보여주지 못하며 평균적으로 3G 네트워크는 1Mbps, Wi-Fi (802.11g)는 18Mbps의 전송 속도를 낸다. 이러한 전송 속도의 차이로 인해 실제 데이터를 전송하는 경우에 반응속도의 차이가 발생한다. 특히 HTML 문서의 경우 XML 문서에 비해 단어 정보 이외에도 불필요한 정보를 많이 포함하고 있기 때문에 데이터의 크기가 상대적으로 크다. 그렇기 때문에 HTML 문서 처리의 경우가 평균적으로 가장 느린 반응속도를 보인다. 특히 3G 네트워크에서의 그래프를 보면 Wi-Fi 네트워크 그래프에 비해서 HTML과 XML의 반응시간 차이가 뚜렷한데 이

는 전송해야 할 데이터의 크기가 HTML이 더 크기 때문에 발생하는 것이라고 볼 수 있다.

HTML과 XML의 경우가 느린 반응속도를 보이는 것은 비단 데이터의 크기 때문만은 아니다. 프록시 서버 측 캐쉬에서 데이터를 가져올 때도 네트워크의 전송속도의 영향을 받지만 HTML이나 XML의 경우보다 빠른 반응속도를 보이고 있다. 프록시 서버 측 캐쉬에서 가져온 데이터는 이미 프록시 서버에서 전처리가 끝난 상황이므로 추가적인 처리 시간이 필요하지 않는 것에 비해 HTML과 XML은 원하는 정보를 파싱하는 작업에 추가적인 시간이 필요하다. 또한 HTML과 같은 경우에는 직접 문자열 처리를 통해 특정 HTML 태그를 찾아서 원하는 정보를 추출해야 한다. 그리고 XML과 같은 경우에는 비교적 데이터가 잘 구조화되어 있지만 XML 문서를 탐색하기 위한 XML 파서 클래스의 생성과 같은 작업에 추가적인 시간이 소모된다. 이와 같이 HTML과 XML의 경우에는 데이터 처리를 위한 추가 작업으로 인해 더 느린 반응속도를 보인다.

프록시 서버 측 캐쉬에서 데이터를 가져오는 경우를 살펴보면 HTML, XML에 비해서 평균 1초 정도 빠른 반응속도를 보인다. 이는 프록시 서버에서 이미 전처리를 수행했기 때문에 데이터 처리에 대한 추가 시간이 필요하지 않고 그로 인해 상대적으로 작은 크기의 데이터를 전송하기 때문에 훨씬 개선된 반응속도를 보이는 것이다.

클라이언트 측 캐쉬에서 데이터를 가져올 때는 가장 빠른 반응속도를 보인다. 이는 네트워크를 통해 데이터를 전송할 필요가 없으며 데이터의 파싱과 같은 처리가 필요하지도 않으므로 0~1 ms의 시간 내에 정보를 사용자에게 제공할 수 있다.

여기서 한가지 상기해야 하는 사실은 이 반응 속도는 모두 무선 네트워크 상에서의 연결이 양호하고 트래픽이 몰리지 않은 상황에서 측정한 데이터라는 것이다. 특히 Wi-Fi의 경우 반응 속도가 1초이기 때문에 실제 사용자에게 있어 그다지 긴 지연 시간이 아닐 수 있다. 그러나 실제로는 신호가 약한 곳에서는 네트워크 연결이 중간에 중단되어 다시 재연결을 맺게 되는 경우도 빈번하며 트래픽이 많이 몰리는 곳에서는 상당히 네트워크 속도가 느려지게 된다. 따라서 실제 환경에서는 여기서 제공한 반응 속도보다는 느려 질 수 있음을 유의해야 한다.

마지막으로 측정 결과에서 볼 수 있듯이 3G 환경에서의 반응 속도가 2초~4초 정도로 사용자가 지연을 느낄 수 있을 정도임을 볼 수 있다. 반면 WiFi 환경에서는 상대적으로 지연이 길지 않은데, 실제 모바일 환경에서는 3G 네트워크를 사용하는 경우가 많다는 점을 생각하면 이런 지연 시간을 줄이기 위한 노력이 필요함을 알 수 있다.

5.2.2 예측 정확도

다음은 선반입을 수행할 때 우선성 마이닝과 협업 필터링을 통해 예측된 단어들이 얼마나 정확한지에 관한 실험이다. 우선성 마이닝은 SingleMC 알고리즘을 사용하며 협업 필터링에서는 Support 값과 Confidence 값을 통해 추천된 단어들의 정확도를 측정한다. 또한 선반입을 통해 미리 가져온 데이터들을 저장하는 캐쉬의 크기를 달리 하였을 때 캐쉬 적중률의 변화를 살펴본다.

실험을 수행하기 위해 히스토리를 생성하기 위해 수집했던 뉴스 기사 중 임의로 10개를 추출하였고 해당 뉴스에서 히스토리에 포함되는 단어들이 출현할 때마다 사전 응용에서 검색을 수행하였다.

그림 5는 3가지 추천 기법에 대해 선반입 크기를 달리하였을 때의 캐쉬 적중률에 대해서 나타내고 있다. 그래프에서PM은 precedence mining, 즉 우선성 마이닝을 뜻하며 CF는 collaborative filtering, 즉 협업 필터링을 뜻한다. 그래프를 통해 확인할 수 있듯이 20개 이하의 단어를 선반입 하였을 때에는 우선성 마이닝을 통한 추천이 가장 높은 캐쉬 적중률을 보이고 있다. 반면에 50개 이상의 많은 수의 단어를 선반입 하는 경우에는 협업 필터링을 통한 추천이 더 높은 캐쉬 적중률을 보인다.

이는 5.2절에서 설명한 바와 같이 우선성 마이닝이 순서 정보가 존재하지 않는 단어에 대해서는 추천을 하지 않는 점에서 기인한다. 따라서 많은 수의 단어를 저장할 수 있는 프록시 서버 측 캐쉬에서는 협업 필터링을 이용하여 선반입을 하는 것이 전체 평균 캐쉬 적중률을 높일 수 있다. 반면에 비용의 문제로 많은 수의 단어를 선반입 할 수 없는 클라이언트 측의 경우에는 적은 수의 선반입에서 캐쉬 적중률이 높은 우선성 마이닝을 적용하는 것이 좋다.

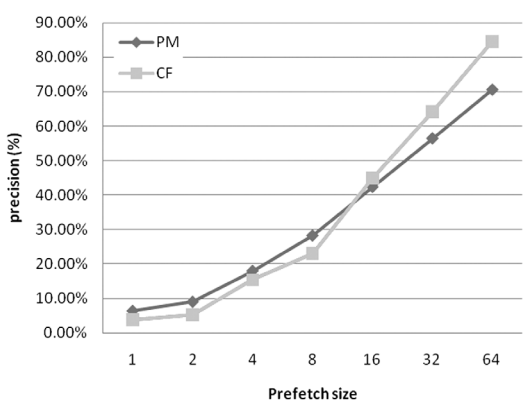


그림 5 예측 기법에 따른 선반입 정확도

그래프에서 50개 이상의 선반입을 할 때 협업 필터링을 통한 선반입이 90%에 달하는 캐쉬 적중률을 보이는데 이것은 실험 데이터의 제한으로 인한 결과로 보여진다. 본 실험에서는 531개의 단어를 추출하여 검색 히스토리를 생성하였다. 그 중 50개 이상의 단어를 선반입한다면 자연스럽게 캐쉬 적중률이 상승할 것으로 보인다. 따라서 위 그래프에서는 선반입 크기가 커질수록 협업 필터링을 사용한 선반입이 더 적합함을 보여주는 것일 뿐이고 실제 상황에서는 선반입 크기가 커질수록 캐쉬 적중률이 높아지는 것은 자명하지만 지나치게 높은 적중률을 나타내지는 않을 것으로 보인다.

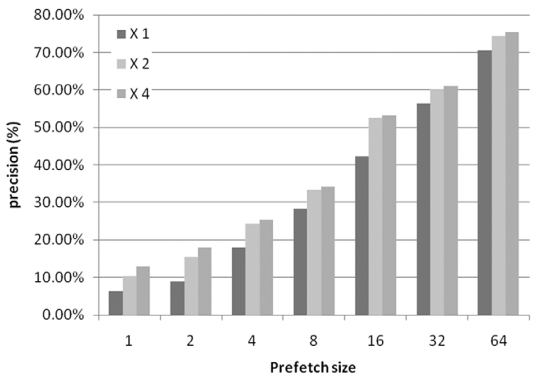


그림 6 캐쉬 크기에 따른 선반입 정확도

위의 그래프에서는 캐쉬의 크기를 변화시켰을 때 캐쉬 적중률을 측정한 결과이다. 캐쉬의 크기가 선반입 크기의 2배와 4배일 때의 캐쉬 적중률을 비교하였다. 일반적인 상황에서 캐쉬 크기를 확장하게 되면 캐쉬 적중률이 높아지는 것이 자명하다. 하지만 본 실험에서는 캐쉬의 크기를 더 증가시켰음에도 불구하고 캐쉬 적중률의 상승폭은 낮다. 이러한 결과는 사전 검색에서의 시간적 지역성이 낮음을 보여준다. 4.3절에서 설명한 바와 같이 사전에서는 이전에 검색했던 단어를 다시 검색할 확률이 낮다. 이에 따라 이전에 추천되었던 단어는 최근에 추천되었던 단어보다 검색될 확률이 낮음이 분명하다. 캐쉬 크기를 증가시키는 것은 이전의 검색 단어들과 이전의 추천 단어들을 더 많이 저장할 수 있는 것일 뿐 최근의 단어와 추천 단어를 저장하는 것과는 관련이 없다. 그러므로 캐쉬의 크기가 증가되었다 하더라도 추가적으로 더 저장된 단어들은 접근 확률이 낮은 단어들이므로 사전 응용에서 캐쉬의 크기 증가가 캐쉬 적중률에 끼치는 영향은 작다.

5.2.3 실제 사용자 실험

다음은 실제 상황에서 2단계 선반입 기법을 통해 얼마나 모바일 응용의 반응속도가 개선되는지에 대한 실험이다. 앞에서 수행했던 실험들은 이미 데이터마이닝을 통해 학습했던 검색 패턴들에 대한 실험으로 그 성능이 이상적으로 나타난 것이라고 볼 수 있다.

실험을 수행하기 위해 11명의 실험 참가자에게 55개의 IT 관련 뉴스 기사를 읽고 원하는 단어를 검색하도록 하였다. 11명의 참가자는 컴퓨터공학과의 대학원생들로 IT 관련 분야의 기본 지식을 충분히 지니고 있는 상황이다. 실험은 모바일 네트워크를 3G 네트워크를 사용하며 클라이언트 측 선반입 크기는 3, 프록시 서버 측 선반입 크기는 50으로 정하고 이루어졌다.

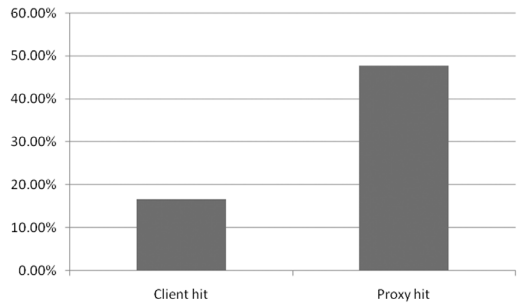


그림 7 실제 사용자의 검색에 대한 캐쉬 적중률

실험 참가자들이 검색한 총 단어의 수는 508개 이며 이 중에서 캐쉬가 적중된 단어의 개수를 측정하여 나타난 클라이언트 측과 프록시 서버 측의 캐쉬 적중률은 위의 그래프와 같다. 클라이언트 측 캐쉬에서 16.7%의 적중률이 나타났고 프록시 서버 측 캐쉬에서 47.71%가 적중되어 선반입을 하지 않았을 때보다 64%의 반응속도의 향상이 있음을 알 수 있다.

새로 입력된 데이터에 대해서 학습을 하고 다시 실험을 수행하였을 때 실험 결과가 좋아지는 것은 당연한 결과이지만 위 실험은 앞으로 지속적인 사전 검색이 이루어질 경우 반응속도의 개선 효과가 어떻게 나타날 것인가에 대한 연구로 볼 수 있다. 데이터마이닝을 통한 미래 예측 기법들은 항상 풍부한 데이터를 가지고 있을 때 효율적으로 작동할 수 있다. 많은 사용자들이 다양한 검색을 하여 히스토리가 충분히 확보된다면 본 논문에서 제안한 기법은 효과적으로 반응속도를 향상시킬 수 있다.

6. 결 론

스마트폰과 같은 모바일 기기가 발달되고 확산되면서 모바일 응용에 대한 수요와 공급이 급격하게 증가하게 되었다. 많은 모바일 응용들은 기존의 웹에서 활용하던 정보들을 모바일 응용에서도 활용하고자 하였다. 기존의 웹 환경에서는 웹 페이지를 빠르게 탐색하기 위한 선반

입 기법들이 많이 연구되어 왔지만 모바일 응용에서 선반입을 적용하는 사례는 부족한 실정이다.

본 논문에서는 모바일 사전 응용에서 반응속도를 향상시키기 위해 사용자가 검색할 것으로 예상되는 단어를 미리 가져오는 선반입 기법을 적용하였다. 특히 모바일 기기의 제한된 자원과 네트워크 비용의 문제를 고려하여 모바일 기기의 부담을 최소화하면서 반응속도를 향상시킬 수 있는 2단계 선반입 기법을 제안하였다. 그리고 선반입에서 중요한 요소로 간주되는 예측의 정확성을 위해 사전 도메인에 적합한 우선성 마이닝을 적용하여 선반입의 효율을 높일 수 있음을 보였다.

5장의 실험 결과에서 확인할 수 있듯이 본 논문에서 제안한 기법은 모바일 응용에서 추가 비용을 최소화하면서 반응속도 향상을 달성할 수 있었다. 그리고 사전과 같은 특정 도메인에서 우선성 마이닝이 기존의 협업 필터링을 통한 예측에 비해 효율적일 수 있음을 보였다. 특히 모바일 응용과 같이 제한된 상황에서 예측의 정확성을 높이는 것은 충분히 의미를 지닌다.

결론적으로 모바일 응용에서 반응속도를 향상시키기 위해 선반입 기법을 적용하였고 여기서 발생할 수 있는 문제점을 2단계 선반입 기법을 적용함으로써 해결하였다. 제안된 2단계 선반입 기법은 모바일 응용 전반에 걸쳐 적용될 수 있는 범용성을 가지고 있으며 특히 반응속도가 중요시되는 모바일 응용을 개발할 때 큰 도움이 될 것이다.

참 고 문 헌

[1] V. Padmanabhan and J. Mogul, "Using predictive prefetching to improve World Wide Web latency," *ACM SIGCOMM Computer Communication Review*, vol.26, no.3, p.36, 1996.

[2] G. Cho, "Using predictive prefetching to improve location awareness of mobile information service," *Computational Science--ICCS 2002*, pp.1128-1136, 2009.

[3] T. M. Kroeger, et al., "Exploring the bounds of web latency reduction from caching and prefetching," *presented at the Proceedings of the USENIX Symposium on Internet Technologies and Systems on USENIX Symposium on Internet Technologies and Systems*, Monterey, California, 1997.

[4] Z. Jiang and L. Kleinrock, "Web prefetching in a mobile environment," IEEE Personal Communications, vol.5, no.5, pp.25-34, 1998.

[5] H. Song and G. Cao, "Cache-miss-initiated prefetch in mobile environments," *Computer Communications*, vol.28, no.7, pp.741-753, 2005.

[6] M. Balabanovi and Y. Shoham, "Fab: content-based, collaborative recommendation," Communications of the ACM, vol.40, no.3, p.72, 1997.

[7] B. Sarwar, et al., "Item-based collaborative filtering recommendation algorithms," *presented at the Proceedings of the 10th international conference on World Wide Web*, Hong Kong, Hong Kong, 2001.

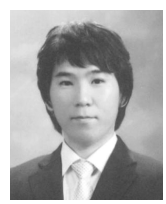
[8] G. Linden, et al., "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol.7, no.1, pp.76-80, 2003.

[9] A. G. Parameswaran, et al., "Recsplorer: recommendation algorithms based on precedence mining," *presented at the Proceedings of the 2010 international conference on Management of data*, Indianapolis, Indiana, USA, 2010.

[10] M. J. Zaki, "Sequence mining in categorical domains: incorporating constraints," *presented at the Proceedings of the ninth international conference on Information and knowledge management*, McLean, Virginia, United States, 2000.

[11] I. F. Ilyas, et al., "Supporting top-k join queries in relational databases," *The VLDB Journal*, vol.13, no.3, pp.207-221, 2004.

[12] R. Srikant and R. Agrawal, "Mining quantitative association rules in large relational tables," *SIGMOD Rec.*, vol.25, no.2, pp.1-12, 1996.



옥 승 훈

2009년 부산대학교 컴퓨터공학부 학사 2011년 서울대학교 컴퓨터공학부 석사 2011년~현재 LIG 넥스원 재직 중. 관심분야는 데이터베이스, 시맨틱 웹, 데이터 마이닝

김 기 성

2003년 서울대학교 응용화학부 학사. 2006년~2009년 티맥스소프트. 2003년~현재 서울대학교 컴퓨터공학부 박사과정 재학 중. 관심분야는 데이터베이스, 시맨틱 웹, 분산 병렬 데이터 처리

김 형 주

1982년 서울대학교 전산학과(학사). 1985년 Univ. of Texas at Austin(석사). 1988년 Univ. of Texas at Austin(박사) 1988년~1990년 Georgia Institute of Technology(부교수). 1991년~현재 서울대학교 컴퓨터공학부 교수. 관심분야는 데이타베이스, XML, 시맨틱 웹, 온톨로지