개선된 추천을 위해 클러스터링을 이용한 협동적 필터링 에이전트 시스템의 성능

황 병 연†

요 약

자동화된 협동적 필터링은 내용에 기반한 정보 필터링 시스템에서 처리할 수 없는 문제를 해결하면서 정보 과다 문제를 줄이기 위해 최근에 널리 사용되는 기술이다. 본 논문에서는 협동적 필터링을 수행하기 위해 기존의 대표적인 방법인 GroupLens와 GroupLens를 수정한 Best N 알고리즘, 그리고 추천의 정확도를 향상시키고자 클러스터링을 이용한 알고리즘을 기술한다. 클러스터링을 이용한 알고리즘은 실제 데이터를 이용한 실험을 통해서 GroupLens, Best N 알고리즘과 성능을 비교한다. 실험 결과 클러스터링을 이용한 알고리즘은 추천의 정확도 면에서 Best N 알고리즘과 비슷하고 GroupLens 알고리즘보다는 좋은 성능을 나타낸다. 에러율의 표준편차 관점에서 클러스터링을 이용한 알고리즘이 가장 작게 나타남으로써 가장 안정된 균일한 추천을 한다는 것을 알 수 있다. 또한 클러스터링을 이용한 알고리즘은 추천을 하는데 소요되는 시간을 단축시킨다

Performance of Collaborative Filtering Agent System using Clustering for Better Recommendations

Byung-Yeon Hwang†

ABSTRACT

Automated collaborative filtering is on the verge of becoming a popular technique to reduce overloaded information as well as to solve the problems that content-based information filtering systems cannot handle. In this paper, we describe three different algorithms that perform collaborative filtering: GroupLens that is the traditional technique; Best N, the modified one; and an algorithm that uses clustering. Based on the experimental results using real data, the algorithm using clustering is compared with the existing representative collaborative filtering agent algorithms such as GroupLens and Best N. The experimental results indicate that the algorithm using clustering is similar to Best N and better than GroupLens for prediction accuracy. The results also demonstrate that the algorithm using clustering produces the best performance according to the standard deviation of error rate. This means that the algorithm using clustering gives the most stable and the best uniform recommendation. In addition, the algorithm using clustering reduces the time for recommendation.

1. 서 론

인터넷 전자상거래 시장의 초기에는 전자상거래가 가능하도록 하는 기본적인 플랫폼을 담당한 커머스 서버나 새로운 분야의 진출에 따른 불안감을 해소할 수 있는 보안 솔루션 등이 주로 부각된 반면, 전자상거래 내의 경쟁력이 강조되는 현재에는 고객에게 양질의 서비스를 제공하고, 이를 이익 창출로 연결시키는데 도움이 되는 보다 특화된 기능의 다양한 솔루션들이 부각되고

※ 본 연구는 2000학년도 가톨릭대학교 교비연구비로 수행되었음.
† 종신회원 : 가톨릭대학교 컴퓨터전자공학부 교수
논문접수 : 2000년 3월 24일, 심사완료 : 2000년 5월 5일

있다[1-3]. 특히 1999년은 CRM(Customer Relationship Management)이란 용어가 자주 거론되었는데, 이는 신규고객의 유치보다도 기존 고객과의 관계를 증진시켜서 평생고객 가치를 지향하고자 하는 요구에 대한 하나의 대안으로 기대를 모았다. 또 인터넷 상용사이트를 개설하기만 하면 이익이 창출된다고 생각하기 보다 이제는 인터넷이란 새로운 채널에 맞는 새로운 마케팅 정책을 구사하고자 하는 노력들이 진행되고 있다. 즉, 판매자는 고객들이 자신의 홈페이지를 찾아오기만을 기다리는 것이 아니라, 방문객을 빠르게 구매자로 변화시키고, 각각의 고객에게 구매할 기회를 최대화하고, 한번 고객이 되면 그 관계를 오래도록 지속시킬 수 있는 정책적 활동에 대한 필요성을 점차 느끼고 있다[4, 5].

이에 따라 마케팅에 특화된 솔루션들에 대한 연구가 활발히 이루어지고 있고 관련된 제품들도 속속 시장에 등장하고 있다. 아마존(www.amazon.com)만 하더라도 'Who bought'[6] 서비스를 위해 NetPerception(www.netperceptions.com)의 솔루션을 이용하고 있다. 'Who bought' 서비스란 특정 책을 조회하면 그 책을 구매한 사람들이 많이 구매한 책 리스트를 제시하는 서비스를 말하는데, 여기서 협동적 필터링이 이용되었다.

협동적 필터링 에이전트는 자동화된 정보 필터링 에이전트로서 NetPerception의 GroupLens[7]와 MacroMedia의 LikeMinds(www.likeminds.com)와 같이 그 상용성을 이미 해외 시장에서 입증하고 있다. 본 연구에서는 협동적 필터링 에이전트의 의의를 설명하고, 관련연구로 Resnick의 GroupLens 오리지날 알고리즘과 그에 대한 이후의 수정 알고리즘을 기술한다. 한편 본 논문에서는 협동적 필터링에 의한 추천을 함에 있어 유사도가 높은 사용자들을 미리 클러스터링을 통해 분류하고 그 클러스터링에 기초한 추천을 하면 추천 결과의 정확도(prediction quality)를 향상시키고, 유의미하게 추천 가능한 사용자와 아이템의 포괄범위를 확장할 수 있을 것이라는 직관을 갖고 클러스터링을 이용한 협동적 필터링 알고리즘을 제시한다. 클러스터링을 이용한 알고리즘은 기존의 알고리즘들과 실험을 통해서 성능을 비교한다. 실험을 위해서 미국 Digital사로부터 EachMovie 데이터 셋 (http://www.research.digital.com/SRC/eachmovie/)을 얻어, 실제 데이터에 기초하여 실험하였고, 실험에 사용된 성능지수는 다음과 같다. 첫째, 에이전트의 추천 결과가 가지는 정확성, 둘째, 추천할 수 있는 사용자 및 아이템의 포괄범위, 셋째, 사용자 응답시간 등이다. 실험 결과 사용자의 유사도에 기초한 클러스터링을 이용한 알고리즘은 평가 이력(rating history)이 부족한 사용자들에 대해서도 보다 정확한 추천을 할 수 있었으며, 역시 평가 횟수가 부족한 아이템에 대한 추천에서도 향상된 추천 정확도를 보여주었다. 그리고 전체적으로 이러한 클러스터링은 어플리케이션이 한번의 추천을 하는데 소요되는 시간을 단축시켰다.

본 논문의 구성은 다음과 같다. 2장에서는 정보과다와 그 문제를 해결하기 위한 방법들을 기술하고 기존의 대표적인 협동적 필터링 에이전트 알고리즘들을 소개한다. 3장에서는 클러스터링을 이용한 협동적 필터링 에이전트 알고리즘을 제시한다. 4장에서는 GroupLens, Best N, 그리고 클러스터링을 이용한 알고리즘을 추천의 정확도 관점에서 성능비교를 위한 실험을 한다. 끝으로 5장에서 결론을 맺는다.

2. 협동적 필터링 에이전트

2.1 정보과다와 그 해법

인터넷에서 정보과다를 해결하는 방법으로는 정보 검색(information retrieval), 정보 필터링(information filtering), 협동적 필터링(collaborative filtering) 등이 있다[8]. 정보 검색 시스템은 사용자가 자신이 원하는 정보를 찾기 위해 특정한 방식의 질의(query)를 하도록 한다. 대체로 문서 검색의 경우에는 문서와 그 초록에 대한 인덱싱에 기반한다. 문서 형태가 아닌 영화, 음악 CD 등에 대해서는 장르, 제작자 등의 속성들로 검색할 수 있도록 해준다. 그렇지만 정보 검색 시스템은 사용자의 순간적 정보 필요성에만 의존할 뿐이다.

한편 정보 필터링은 사용자의 요구나 프로필에 기반한다. 단순한 시스템의 경우에는 사용자가 수동으로 프로필을 작성하게 하고, 부분적인 지원만을 할 수 있을 뿐이다. 예를 들면 사용자는 자신의 취미를 입력할 수 있고, 그러면 시스템은 사용자가 체크한 취미에 관한 내용을 보다 많이 보여주고, 그렇지 않은 경우는 적게 보여준다. 이렇게 대상 분류법에 의하기 보다는 사용자의 취향을 미리 입력받는 경우가 프로필에 기반하는 경우다. 사용자의 요구에 기반한다는 것은 사용자들이 자신이 필요로 하는 정보를 정보 분류에 따라 표시하도록 하는 것이다. 대표적인 경우가 PointCast와 같은 온라인 뉴스 푸쉬 서비스다. PointCast 프로그램

을 설치한 사용자는 자신이 보고 싶은 뉴스 종류에만 체크할 수 있도록 하여, 해당 종류에 속하는 뉴스들만 보내주도록 한다. 정보 필터링 시스템은 대상 정보를 미리 분류해 놓거나, 그 속성들을 알고 있어야 하는 제약을 가지고 있고, 사용자가 찾고 싶은 내용을 분명히 알지 못할 경우에는 대응하지 못하는 문제가 있다.

이에 반해서 협동적 필터링 시스템은 활용 가능한 아이템에 대한 사용자들의 의견을 데이터베이스로 구축한다. 시스템은 의견이 유사한 사용자(유사 그룹)들을 발견하고, 특정한 사용자의 특정한 아이템에 대한 의견을 유사 그룹의 의견을 참조하여 예측해 낸다. 영화라는 아이템 도메인을 예로 들면, 사용자는 각각의 영화들에 대해서 1에서 5 사이의 정수로 자신의 취향을 평가(rating)할 수 있다고 하자. 평가는 특정 사용자가 특정 아이템에 대해서 의견을 표시하는 가장 단순한 방법이다. <표 1>에서 순이는 영희보다는 철수에 더 가까운 취향을 가지고 있다는 것을 추측할 수 있다. 따라서 순이가 아직 평가하지 않은 러브레터에 대한 예측은 영희보다는 철수를 참조하는 것이 더 낫다고 볼 수 있는데, 이와 같은 가정과 방법에 기초한 것이 바로 협동적 필터링이다.

〈표 1〉 영화에 대한 평가 예

	타이타닉	쉬 리	매트릭스	러브레터
철 수	5	5	3	2
영 희	3	3	5	5
순 이	4	4	3	?

협동적 필터링이 정보 검색이나 정보 필터링과 비교해서 가지는 장점은 아이템에 대한 사전 분류나 파악에 의존하기 보다는 사용자들의 의견에 참조한다는 점과 사용자가 예상하지 않았지만 자신의 취향에 맞는 새로운 아이템을 발견할 가능성이 있다는 점을 들 수 있다. 유명한 아마존의 경우를 예로 들면, 사용자가 '죄와 벌'이라는 책을 조회하면, 그 책을 구매한 사람들이 가장 많이 구매한 책들의 top-N 리스트를 볼 수 있고, 또 '죄와 벌'의 저자인 도스토예프스키의 책을 구매한 사람들이 가장 많이 구매한 다른 저자들도 참조할 수 있다. 이 기능은 단순한 듯 하지만, 만약 도스토예프스키를 좋아하던 사람이 자신에게 맞는 다른 저자를 찾고 싶지만, 아직 어떤 저자가 좋은지 모른다고 할 때, 이와 같은 협동적 필터링은 많은 도움이 된다고 할 수 있다. 무엇보다 협동적 필터링은 시스템에서 자동화될 가능성이 매우 높다는 점에 주목할 필요가 있으며, 이는 이미 몇몇 상용화 제품인 NetPerception과 LikeMind를 통해서 알 수 있다.

2.2 기존의 협동적 필터링 알고리즘

2.2.1 Tapestry 알고리즘

Tapestry[9]는 Xerox Palo Alto Research Center에서 개발한 문서 필터링 시스템이다. Tapestry로부터 협동적 필터링의 개념이 유래되었다. Tapestry에서의 협동은 사용자가 그들이 읽은 문서에 대한 주석(annotation)을 다는 것을 허락한다. 이렇게 문서에 대한 주석이 달려 있으므로 다른 Tapestry 사용자는 문서를 검색할 때 키워드 매칭을 통한 검색뿐만 아니라 다른 사용자의 문서에 대한 주석을 통해서도 검색할 수 있다. 이러한 주석은 형식이 없는 형태(free text)일 수도 있고 "likeit"과 "hateit"처럼 Tapestry에서 미리 정해진 형태로 문서에 주석을 부여할 수도 있다. Tapestry는 클라이언트/서버 구조를 가지고 있다.

2.2.2 GroupLens 알고리즘

GroupLens는 Tapestry의 문제점을 해결하면서 Netnews의 개별화된 추천을 위해 협동적 필터링을 적용한 시스템이다[10]. GroupLens는 문서에 대한 선호도를 숫자로 나타내며 사용자 프로파일에 이 정보를 포함시켜 서버에 저장한다. GroupLens는 두 가지 평가방법을 가진다. 첫 번째는 사용자의 직접적인 평가에 의한 방법, 두 번째는 사용자의 직접적인 평가가 없는 경우 다른 사용자의 프로파일을 기반으로 한 상호관계에 의해 문서에 대한 평가를 예측하는 방법이다. GroupLens에서는 Tapestry에서 문제가 되었던 사용자가 읽은 문서에 대해 평가를 하지 않은 경우 다른 유저와의 유사성을 기반으로 해결하려는 방법을 시도하였다. 예를 들어 다음과 같이 문서 6개에 대한 4명의 사용자의 평가가 <표 2>와 같다고 가정하자.

〈표 2〉 각 유저의 아이템에 대한 평가

It \ Us	Ken	Lee	Meg	Nan
1	1	4	2	2
2	5	2	4	4
3			3	
4	2	5		5
5	4	1		1
6	?	2	5	?

<표 2>에서 빈칸으로 되어 있는 것은 아직 사용자가 그 문서를 읽지 않은 것을 나타낸다. 그리고 물음표가 되어 있는 것은 문서를 읽었지만 문서에 대한 평가를 하지 않은 것을 의미한다. GroupLens는 Ken의 6번째 문서에 대한 평가를 예측하기 위해서 상관계수를 사용한다. 이 값은 -1과 1사이의 값을 가지게 된다. Ken와 Lee의 상관계수는 다음과 같이 구해진다.

$$r_{KL} = \frac{Cov(K, L)}{\delta_K \delta_L} \quad (2.1)$$

$$= \frac{\sum_i (K_i - \overline{K})(L_i - \overline{L})}{\sqrt{\sum_i (K_i - \overline{K})^2} \sqrt{\sum_i (L_i - \overline{L})^2}}$$

($\overline{K}$: Ken의 문서에 대한 평가 평균
$\overline{L}$: Lee의 문서에 대한 평가 평균)

실제로 계산을 하면 -0.8이 나온다. 상관계수 값이 1이면 완전한 양의 관계라고 하며 값이 -1이면 완전한 음의 관계라고 한다. 만약 값이 0이면 관계성이 존재할 수도 있고 존재하지 않을 수도 있는 경우가 발생하게 된다.

위와 같은 방법으로 다른 사람들과의 상관관계를 구해보면 Meg과의 관계는 1이며 Nan과의 관계는 0이 된다. 다른 사람과의 상관관계가 모두 구해졌으면 A의 6번째 평가를 예측하게 된다. 평가의 예측에는 상관관계를 포함한 평균값을 이용한다.

$$K_{6Pr} = \overline{K} + \frac{\sum_{J \in raters} (J_6 - \overline{J}) r_{KJ}}{\sum_J |r_{KJ}|} \quad (2.2)$$

$$= 3 + \frac{2r_{KM} - r_{KL}}{|r_{KM}| + |r_{KL}|} = 3 + \frac{2 - (-.8)}{|1| + |-.8|}$$

$$= 4.56$$

Nan의 상관관계가 0이므로 관계가 있을 수도 있고 없을 수도 있으므로 Nan은 수식에 의해서 제외되게 된다. 따라서 Ken의 6번째 문서에 대한 평가는 4.56으로 예측되었다. 이와 같은 방법으로 평가를 하지 않은 문서에 대한 예측을 행한다.

본 연구에서는 유저간의 유사성에 가중치를 부여하기 위해 피어슨 상관계수를 사용하였다. 그리고 다른 유저들의 가중치가 부여된 분산 평균을 계산하여 최종적으로 prediction을 하게 된다. 위의 식 (2.1)을 피어슨 상관계수를 적용하여 다시 쓰면 다음과 같다.

$$p_{a,i} = \overline{r}_a + \frac{\sum_{u=1}^{n} (r_{u,i} - \overline{r}_u) * w_{a,u}}{\sum_{u=1}^{n} w_{a,u}} \quad (2.3)$$

$p_{a,i}$ 은 유저 a의 아이템 i에 대한 prediction score, n은 나머지 다른 유저, 그리고 $w_{a,u}$는 유저 a와 다른 유저 u 사이의 similarity weight를 나타내며, 다음과 같이 피어슨 상관계수로 정의 된다.

$$w_{a,u} = \frac{\sum_{i=1}^{m} (r_{a,i} - \overline{r}_a) * (r_{u,i} - \overline{r}_u)}{\delta^*_a \delta^*_u} \quad (2.4)$$

GroupLens는 최초로 제안된 자동화된 협동적 필터링 알고리즘이다. GroupLens 알고리즘이 가지는 문제들은 초기 데이터베이스가 많이 구축되어 있지 않으면 정확한 추천이 불가능하다는 점, 아이템의 전체 수에 비해서 사용자들이 평가한 아이템 수가 매우 부족한 점, 비동질적 아이템들을 묶어서 추천하기 힘들다는 점 등을 들 수 있다. 본 연구는 이와 같은 GroupLens 알고리즘의 문제들을 해결하기 위해 GroupLens의 기본 알고리즘을 바탕으로 하면서 사용자 응답시간을 단축하고 추천의 정확성을 높이며, 의미있게 추천할 수 있는 영역(사용자와 아이템)을 확장하는 것을 통해서 시스템의 성능을 향상시키고자 한다

3. 클러스터링을 이용한 협동적 필터링

클러스터링이란 실세계 현상에 대해 수집된 데이터의 분류 구조를 알아내기 위해 설계된 수학적 기법[11]이며, 클러스터란 데이터의 입장에서 보았을 때 다른 데이터들과는 달리 유사성을 지니고 있는 개체들로 구성된 일련의 데이터들이다. 데이터를 클러스터링을 하면 다음과 같은 효과가 기대된다.

(1) 예측된 rating의 평균 절대 에러(mean absolute error)가 최소화된다. 만약{r_1 ..... r_n}이 target set의 실제 값이고, {p_1 ..... p_n}이 예측된 값이라면, 평균 절대 에러는 식 (3.1)과 같다. 평균 절대 에러가 낮을수록 알고리즘의 정확도는 높아진다.

$$|\overline{E}| = \frac{\sum_{i=1}^{N} |\varepsilon_i|}{N} \quad (3.1)$$

(2) 에러의 표준편차(식 3.2)가 최소화된다. 편차를

줄이면 다양한 환경에서 알고리즘의 정확도는 높아지고 균일한 성능을 나타낸다.

$$\sigma = \sqrt{\frac{(\sum(E - \overline{E})^2)}{N}} \quad (3.2)$$

(3) 각 예측된 값에 대한 신뢰도를 계산할 수 있다. 신뢰도의 수준은 예측된 값이 얼마나 신뢰할 수 있는지를 나타낸다.

K-means 알고리즘[12]은 많은 어플리케이션에서 좋은 클러스터링 결과를 나타내고 있다. 그러나 k-means 알고리즘은 클러스터의 수와 패턴의 수에 비례하여 시간이 걸린다. 이것은 거대한 데이터 셋에서 계산시 막대한 비용이 소요됨을 의미한다. 또한 k-means 알고리즘은 유클리디안 거리(Euclidean distance)에서만 적용 가능하다는 제약이 있다. 즉, 어떤 차원에 대해 절대 좌표(공간)에서 표시 할 수 있어야 한다. 따라서 유저의 유사성을 이용하여 거리를 나타내고자 하는 본 연구의 실험에서 이 알고리즘은 적용될 수 없다. 이것은 유저 사이의 유사성은 절대 좌표로 표시할 수 없는 상대적인 값이기 때문이다

K-means 알고리즘이 비공간적 데이터에서 적용할 수 없는 제약이 있기 때문에 비공간적 데이터에서 사용할 수 있는 알고리즘이 필요하다. 이러한 알고리즘으로 PAM[13], CLARA[14], CLARANS[14] 등의 K-medoid 알고리즘이 있다. 우리는 실험을 통해서 CLARANS가 가장 좋은 성능을 보임을 알았다. 따라서 본 논문에서는 CALRANS 클러스터링을 이용한 알고리즘을 기술한다. 다음 알고리즘 Cluster_Recommend는 사용자들의 유사도를 측정하여 k 개의 클러스터 그룹으로 사용자들을 분류한 후, 주어진 어떤 사용자의 대상 항목의 예상되는 평가 값을 계산하여 추천을 하는 과정을 보여준다.

Algorithm Cluster_Recommend
(ACTIVEUSER, ACTIVEITEM, NUMLOCAL, MAXNEIGHBOR)

Input : *ACTIVEUSER*는 추천을 하고자 하는 사용자를 의미하고, *ACTIVEITEM*은 추천을 하고자 하는 항목을 의미한다. *NUMLOCAL*은 k 개의 임의의 대표 사용자를 선택하는 횟수를 의미하고, *MAXNEIGHBOR*는 임의의 선택된 k개의 대표 사용자에 대해 가장 낮은 유사도 비용을 갖는 neighbor 노드를 찾기 위해 최대로 비교되는 neighbor 노드의 수를 의미한다. neighbor 노드는 k개의 임의의 대표 사용자로 구성된 어떤 노드에서 한 사용자만 다른 노드를 나타낸다.

Output : 주어진 어떤 사용자의 대상 항목의 예상되는 평가 값을 계산하여 추천한다.

Method :

R1. I를 1로, *MINCOST*를 무한대의 큰 값으로 초기화한다.
R2. 전체 사용자 중에서 k개의 임의의 대표 사용자를 선택해서 current 노드로 설정한다.
R3. J를 1로 설정한다.
R4. Current 노드의 임의의 neighbor 노드 S를 선택하고 두 노드의 비용차를 계산하기 위해 함수 **Similar_Cost**(O_i, O_h)를 call한다. 여기서 O_i는 current 노드를 구성하는 k개의 선택된 대표 사용자들 중에서 교체되어질 한 사용자이고, O_h는 선택되지 않은 나머지 사용자들 중에서 neighbor 노드에 포함되는 한 사용자이다. 예를 들어 Current 노드가 (A, B, C)이고 neighbor 노드가 (A, B, D)이면 O_i는 C이고, O_h는 D이다.
R5. 만일 S의 비용이 적다면 즉, **Similar_Cost**(O_i, O_h) 값이 음수이면 Current 노드를 S로 설정하고 단계 R3으로 간다.
R6. S의 비용이 적지 않다면 즉, **Similar_Cost**(O_i, O_h) 값이 양수이면 J를 1만큼 증가 시킨다. 만일 J <= *MAXNEIGHBOR*이면 단계 R4로 간다.
R7. J > *MAXNEIGHBOR*이면, current 노드의 총 비용과 *MINCOST*를 비교하여 current 노드의 총 비용이 작다면 *MINCOST*를 current 노드의 총 비용으로 설정한다. 여기서 current 노드의 총 비용이란 current 노드에 속하지않는 사용자들에 대해 current 노드를 구성하는 대표 사용자들 중에서 가장 가까운 대표 사용자를 찾아서 클러스터를 구성한 후, 각 클러스터에 대해 대표 사용자와 나머지 사용자들 사이의 비용을 구하여 모든 클러스터의 비용을 합한 것이다.
R8. I를 1만큼 증가시킨다. 만일 I <= *NUMLOCAL* 이면 단계 R2로 간다.
R9. *ACTIVEUSER*가 속한 클러스터 그룹 내에서 유사도가 가장 높은 N 개의 사용자를 구한다.
R10. N 명의 사용자에 대해서 상관계수를 이용한 식 (2.1)과 식 (2.2)를 통해서 *ACTIVEUSER*의 *ACTIVEITEM*에 대한 예측되는 평가 값을 구하고 이 값에 의해 추천을 한다.

Algorithm Similar_Cost(O_i, O_h)

Input: O_i는 k 개의 선택된 대표 객체들 중의 한 객체이고, O_h는 선택되지 않은 나머지 객체들 중의 한 객체이다.

Output : 두 사용자인 O_i와 O_h를 교환할 때 소요되는 유사도에 의한 비용을 계산한다.

Method :

C1. 객체 O_i와 객체 O_h를 교환한다.
C2. O_i와 O_h 사이의 교환에 의한 영향을 계산하기 위해 모든 선택되지 않은 객체 O_j에 대한 비용 C_{ijh}을 계산한다.
C3. O_j가 다음의 어떤 경우에 해당되느냐에 따라 C_{ijh}는 다음과 같이 4가지 경우로 정의된다.

첫번째 경우 : O_j가 k개의 대표 객체들 중에서 O_i로 대

표되는 클러스터에 가장 유사하고, O_j가 O_h 보다 $O_{j,2}$와 더 유사하다고($d(O_j, O_h) \geq d(O_j, O_{j,2})$) 가정한다. 여기서 $O_{j,2}$는 k개의 대표 객체들 중에서 O_j와 두 번째로 가까운 대표 사용자이고, $d(O_j, O_h)$에서 d는 dissimilarity를 나타내며 유사도를 측정하는 식 (2.1)에 반대되는 의미로 사용된다. 만일 대표 사용자 O_i가 O_h로 대체되면, O_j는 $O_{j,2}$가 대표하는 클러스터에 포함될 것이다. 그러므로 교환에 따른 비용은 다음과 같이 된다.

$$C_{ijh} = d(O_j, O_{j,2}) - d(O_j, O_i) \quad (3.3)$$

이 방정식은 항상 0 이상의 양의 값을 갖는다. 이것은 O_i를 O_h로 대체하면 양의 비용이 든다는 의미이다.

두 번째 경우 : O_j가 k개의 대표 객체들 중에서 O_i로 대표되는 클러스터에 가장 유사하고, O_j가 $O_{j,2}$보다 O_h와 유사하다고($d(O_j, O_h) < d(O_j, O_{j,2})$) 가정한다. 만일 대표 사용자 O_i가 O_h로 대체되면, O_j는 O_h가 대표하는 클러스터에 포함될 것이다. 그러므로 교환에 따른 비용은 다음과 같이 된다.

$$C_{ijh} = d(O_j, O_h) - d(O_j, O_i) \quad (3.4)$$

식 (3.3)과는 달리, C_{ijh}은 음의 값 혹은 양의 값을 가질 수 있다. 이것은 O_j가 O_i와 O_h중 어느것과 더 유사한지에 의존한다.

세번째 경우 : O_j가 현재 O_i로 대표되는 클러스터보다 다른 클러스터에 더 유사하고, 이 클러스터의 대표 사용자가 $O_{j,2}$이며, O_j가 O_h 보다 $O_{j,2}$와 유사하다고($d(O_j, O_h) \geq d(O_j, O_{j,2})$) 가정한다. 만일 대표 사용자 O_i가 O_h로 대체되면, O_j는 $O_{j,2}$가 대표하는 클러스터에 머물러 있을 것이다. 그러므로 교환에 따른 비용은 다음과 같이 된다.

$$C_{ijh} = 0 \quad (3.5)$$

네번째 경우 : O_j가 현재 $O_{j,2}$로 대표되는 클러스터보다 다른 클러스터에 더 유사하고, O_j가 $O_{j,2}$보다 O_h와 유사하다고($d(O_j, O_h) < d(O_j, O_{j,2})$) 가정한다. 만일 대표 사용자 O_i가 O_h로 대체되면, O_j는 O_h가 대표하는 클러스터로 옮겨 갈 것이다. 그러므로 교환에 따른 비용은 다음과 같이 된다. 그리고 이 값은 항상 음의 값이 된다.

$$C_{ijh} = d(O_j, O_h) - d(O_j, O_{j,2}) \quad (3.6)$$

C4. 이 네 가지의 경우에 의해 O_i와 O_h를 교환할 때의 총 비용 TC_{ih}는 다음 식 (3.7)과 같다.

$$TC_{ih} = \sum_j C_{ijh} \quad (3.7)$$

C5. 총 비용 TC_{ih}를 return한다.

4. 성능 평가

본 장에서는 클러스터링을 이용한 협동적 필터링 알고리즘과 기존 알고리즘과의 성능평가를 실험을 통해 알아본다.

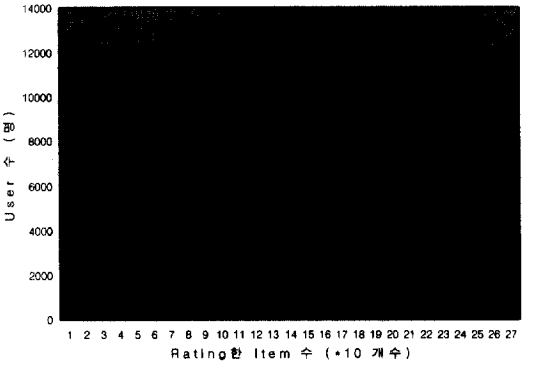
4.1 데이터 셋에 대한 설명과 실험의 목적

미국 Digital사로부터 EachMovie의 recommendation service를 통해 수집된 데이터 셋을 제공 받아 실험하였다. 이 데이터는 72916명의 유저가 18개월에 걸쳐 1628개의 영화에 평가한 것으로, 총 2811983건에 달한다. 각 영화에 대해 0-1까지 0.2 간격으로 총 6단계로 평가되어 있다. <표 3>은 평가 점수에 따른 유저 수를 나타낸 것이다.

〈표 3〉 아이템(영화)에 대한 평가 점수별 유저 수

점 수	유 저 수
0	347191
0.2	150495
0.4	339718
0.6	701236
0.8	761676
1	511667

실제 실험에서는 계산상의 편의와 메모리 절약을 위해 0-1까지 0.2 간격으로 표현된 데이터에 5를 곱하여 0-5까지 1간격으로 변환하여 사용하였다. 이 데이터 셋은 총 3개의 텍스트 파일로 구성되어 있는데, 각 유저의 프로파일 정보가 기록된 person.txt와 각 영화에 대한 정보를 담고 있는 movie.txt, 그리고 실제 평가 데이터인 vote.txt로 구성된다.



(그림 1) 평가한 아이템(영화) 수에 따른 유저 수

(그림 1)은 평가한 아이템에 따른 유저 수를 보여주고 있다. 그림에서 알 수 있듯이 10개 미만의 아이템에 평가한 유저 수가 약 13000명으로 가장 많고 평가한 아이템 수에 비례하여 유저 수가 줄어 든다. 한편 한 아이템에 대해 평가한 유저 수와 아이템 개수 사이

의 관계를 나타내보면 100명 미만의 유저가 평가한 아이템 수가 약 190개 정도로 가장 많고 평가한 유저 수가 증가할수록 그 아이템 개수는 줄어든다. 이와 같이 데이터 셋이 고르게 분포하지 않기 때문에 데이터의 희소성(sparsity)에 대한 대책이 요구된다. 이러한 이유로 본 논문에서는 실험 과정에서 유저별, 아이템별 그룹으로 나누어 희소성에 의한 prediction 테스트의 정확도 저하에 대처하였다.

본 실험의 목적은 데이터를 여러 그룹으로 나누어 각 그룹별로, GroupLens 알고리즘과 GroupLens를 수정한 Best N 알고리즘, 그리고 본 연구에서 제안된 클러스터링을 이용한 알고리즘에 대해 prediction의 편차 평균(에러율)을 비교하여 GroupLens의 prediction의 정확도를 향상 시키며, 클러스터링의 기여도를 살피고자 한다.

4.2 Prediction의 정확도 향상에 대한 실험

본 실험에 사용된 컴퓨터 사양은 Intel 펜티엄 MMX 200, 메모리 80MB이며, NT 운영체제에서 C^{++} 프로그래밍 언어를 사용하여 각 알고리즘을 구현하고 시뮬레이션 하였다.

(1) GroupLens 알고리즘에 의한 실험

실험 과정

① 유저가 평가한 영화 개수나 영화에 평가한 유저 수에 따라 prediction score가 영향을 받기 때문에 데이터의 희소성에 의한 대처 방안으로 <표 4>와 같이 총 25개의 그룹으로 나눈다. 먼저 유저가 평가한 영화의 개수로 5개의 유저 그룹으로 나누는데, 각 5개의 그룹이 대략 1/5씩을 차지하도록 임의의 유저가 평가한 아이템 수를 정한다. 유저 0그룹은 10개 미만(12958명), 1그룹은 20개 미만(11649명), 2그룹은 40개 미만(14664명), 3그룹은 80개 미만(11974명), 4그룹은 80개 이상(10020명)을 나타낸다. 같은 방법으로 영화 0그룹은 50명 미만(322개), 1그룹은 200명 미만(360개), 2그룹은 650명 미만(324개), 3그룹은 1900명 미만(313개), 4그룹은 1900명 이상(304개)을 나타낸다.

② GroupLens 알고리즘에 의해 각 그룹별로 임의로 선택된 100개(s=100)의 유저/아이템 쌍에 대해 모든 사람의 유사도를 고려하여 에러율(편차 평균)을 구한다.

〈표 4〉 유저가 평가한 영화 개수와 영화에 평가한 유저 수에 의한 그룹

Us \ It	< 50	< 200	< 650	< 1900	≥ 1900
< 10	(0,0)	(0,1)	(0,2)	(0,3)	(0,4)
< 20	(1,0)	(1,1)	(1,2)	(1,3)	(1,4)
< 40	(2,0)	(2,1)	(2,2)	(2,3)	(2,4)
< 80	(3,0)	(3,1)	(3,2)	(3,3)	(3,4)
≥ 80	(4,0)	(4,1)	(4,2)	(4,3)	(4,4)

이때 유저/아이템 쌍에서 임의의 유저에 대한 아이템은 prediction score와의 비교를 위해 그 유저가 이미 평가한 것이다. 이 아이템을 제외한 다른 아이템들에 대하여 prediction score를 구하고, 이 prediction score와 실제로 유저가 그 아이템에 평가한 score와의 에러율을 비교한다.

실험 결과

<표 5>는 GroupLens 알고리즘에 의한 각 그룹별 에러율을 나타낸다. 각 그룹별로 100개씩 총 2500 유저/아이템 쌍의 추천을 하는데 걸린 시간은 7343초 였다. 전체적인 경향이 유저 그룹과 아이템 그룹 번호가 증가하는 방향(오른쪽 아래방향)으로 갈수록 에러율이 감소하고 있다. 이것은 유사성을 계산할 때 많은 아이템을 평가한 유저일수록, 그리고 많은 유저가 평가한 아이템일수록 정확도가 높아지기 때문이다. 에러율의 전체 평균은 1.16174로 나타났다. 이것은 평균적으로 실제로 평가한 점수 보다 1.16174정도 많게 혹은 적게 예측된다는 의미이다. 또한 모든 그룹의 에러율의 표준편차는 0.29440으로 나타났다.

〈표 5〉 GroupLens 알고리즘에 의한 각 그룹별 에러율

Us \ It	0	1	2	3	4
0	1.49963	2.60179	1.36738	1.50784	1.34679
1	1.24892	1.40099	1.64214	1.45328	1.41212
2	1.26142	1.70256	1.59696	1.44539	1.49726
3	1.20245	1.70180	1.47608	1.32226	1.31662
4	1.21653	1.40372	1.23557	1.37011	1.11370

(2) GroupLens를 수정한 Best N에 의한 실험

실험 과정

① 실험 (1)과 같이, 유저가 평가한 영화 개수, 그리고 영화에 평가한 유저 수로 그룹을 나눈다. 그리고 모든 유저에 대해서 Best N을 구한다. 이 실험에서는

유저별로 Best 100 즉, N=100으로 하였다.

② 실험 (1)과 같이, 그룹별로 100개씩 임의로 선택된 유저/아이템 쌍에 대해 유사도가 높은 N명에 대해서만 영향력을 평가하여 각 그룹별 에러율을 구한다

실험 결과

<표 6>은 Best N 알고리즘에 의한 각 그룹별 에러율을 나타낸다. 각 그룹별로 100개씩 총 2500 유저/아이템 쌍의 추천을 하는데 걸린 시간은 1001초 였다. 하지만 실험 과정에서 각 유저별 Best N을 저장한 Best N matrix를 구하기까지 108000여 초가 걸렸다. Best N 알고리즘에 의한 에러율 분포는 오리지날 그룹렌즈 알고리즘과 반대의 경향을 보이고 있다. 즉, 전체적인 경향이 유저 그룹과 아이템 그룹 번호가 증가하는 방향(오른쪽 아래방향)으로 갈수록 에러율이 증가하고 있다. 그리고 오리지날 그룹렌즈 알고리즘에서 가장 낮은 에러율을 보이는 (4,4) 그룹이 Best N 알고리즘에서는 가장 높은 에러율을 보이고 있다. 이 그룹은 가장 많은 유저가 평가한 아이템에 가장 많은 아이템에 평가한 유저들의 그룹이다.

〈표 6〉 Best N 알고리즘에 의한 각 그룹별 에러율

Us \ It	0	1	2	3	4
0	0.30877	0.36111	0.39318	0.47448	0.33901
1	0.35588	0.37281	0.38691	0.48365	0.37179
2	0.40168	0.39981	0.43799	0.60806	0.40697
3	0.50330	0.44498	0.51257	0.55225	0.57276
4	0.68371	0.67801	0.70041	0.70387	1.16073

Best N 알고리즘의 전체 에러율 평균은 0.41336으로 나타났고, 이는 오리지날 그룹렌즈 알고리즘의 평균 1.16175보다 0.74839정도 낮다. 이것은 유사도가 낮은 사람의 영향력을 배제하는 것이 추천의 정확도를 향상시킨다는 것을 보여준다. 또한 모든 그룹의 에러율의 표준편차가 0.18241으로 표준편차가 0.29440인 오리지날 GroupLens 알고리즘보다 낮고 이것은 전체적으로 에러율 분포가 균일한 안정된 예측을 하고 있음을 나타낸다.

③ CLARANS 알고리즘을 적용한 Best N에 의한 실험

실험 과정

① 실험 (1)과 같이, 유저가 평가한 영화 개수, 그리고 영화에 평가한 유저 수로 그룹을 나눈다.

② CLARANS 알고리즘에 의해 구한 k개의 medoid를 기준으로 클러스터링 한다. 이 실험에서는 CLARANS에 의해 k = 5 즉 5개의 medoid를 구하였다. 그런 다음 각 medoid와의 유사성 비교를 통해 가장 유사성이 높은 medoid가 대표하는 클러스터에 나머지 유저들을 할당한다.

③ 실험 (1)과 같이 각 그룹별로 에러율(편차 평균)을 구한다. 임의로 선택된 유저와 각 클러스터의 medoid와의 유사성을 비교하여 그 유저가 속한 클러스터내의 유저 중에서 유사도가 높은 Best N명의 유저와의 관계로 prediction score를 구한다.

실험 결과

<표 7>은 Clarans(K)&Best N에 의한 각 그룹별 에러율을 나타낸다. 각 그룹별로 100개씩 총 2500 유저/아이템 쌍의 추천을 하는데 걸린 시간은 989초 였다. 하지만 실험 과정에서 5개의 medoid를 구하는데 4044초가 걸렸고 모든 유저의 Best N을 구하는데 24530초가 걸렸다. 각 그룹별로 임의로 선택된 100개의 샘플에 대한 전체 에러율 평균은 0.52570으로, GroupLens 알고리즘의 평균 1.16174보다는 낮게, Best N 알고리즘의 평균 0.41336보다는 높게 나왔다. 그리고 특별히 높거나 낮은 그룹이 없이 전체적으로 매우 균일한 에러율 분포를 보이고 있다. 실제로 모든 그룹의 에러율의 표준편차가 0.12899로 GroupLens 알고리즘의 0.29440이나, Best N 알고리즘의 0.18241보다도 작게 나타나, 가장 안정적인 예측을 하고 있음을 알 수 있다. 또한 추천에 걸리는 시간도 989초로 GroupLens 알고리즘의 7343초나 Best N 알고리즘의 1001초보다 빠르다는 것을 알 수 있다.

〈표 7〉 Clarans(K)&Best N에 의한 각 그룹별 에러율

Us \ It	0	1	2	3	4
0	0.34452	0.32554	0.43844	0.64729	0.38017
1	0.38604	0.39924	0.41649	0.66310	0.39753
2	0.45550	0.48821	0.45467	0.70613	0.44177
3	0.56294	0.49441	0.54830	0.58015	0.61505
4	0.66350	0.74811	0.67659	0.72855	0.58031

한편 우리는 클러스터링의 개수 K를 5라고 가정하고 실험을 하였다. <표 8>은 각 알고리즘의 성능을 비교한 표이다. 수행시간은 s=100일 때 25개의 그룹 즉, 임의의 2500유저에 대해 prediction을 하는데 걸린 시간을 의미한다. Best N에서 괄호로 표시된 값은 pre-

diction을 수행함에 앞서 각 유저에 대해 유사도가 높은 Best N을 구하는데 걸리는 시간이고 Clarans(K) &Best N에서 괄호로 표시된 값은 K개의 medoid를 찾는데 걸리는 시간과 Best N을 구하는데 걸리는 시간이다. 실험을 통해서 클러스터링을 이용한 협동적 필터링 알고리즘은 전체적으로 에러율이나 에러율의 표준편차, 추천시간 등에서 가장 안정된 성능을 보여준다는 것을 알 수 있다. <표 8>은 3가지 알고리즘의 성능 비교를 나타낸다.

〈표 8〉 3가지 알고리즘 성능 비교

	평균 에러율	에러율 표준편차	수행 시간(초)
GroupLens	1.16174	0.29440	7343
Best N	0.41336	0.18241	1001 (108500)
Clarans(K)& Best N	0.52570	0.12899	989 (4044+24540)

5. 결 론

본 연구에서는 협동적 필터링 에이전트의 의의를 설명하고, 관련연구로 Resnick의 GroupLens 알고리즘과 그에 대한 이후의 수정된 Best N 알고리즘을 기술하였다. 또한 본 논문에서는 협동적 필터링에 의한 추천을 함에 있어 유사도가 높은 사용자들을 미리 클러스터링을 통해 분류하고 그 클러스터링에 기초한 추천을 하면 추천 결과의 정확도(prediction quality)를 향상시키고, 유의미하게 추천 가능한 사용자와 아이템의 포괄범위를 확장할 수 있을 것이라는 직관을 갖고 클러스터링을 이용한 협동적 필터링 알고리즘을 제시하였다. 클러스터링을 이용한 알고리즘은 약 28만 건의 실제 영화에 대한 평가 데이터를 이용하여 기존의 알고리즘들과 실험을 통해서 성능을 비교하였다. 실험 결과 사용자의 유사도에 기초한 클러스터링을 이용한 알고리즘은 평가 이력(rating history)이 부족한 사용자들에 대해서 보다 정확한 추천을 할 수 있었으며, 역시 평가 횟수가 부족한 아이템에 대한 추천에서도 향상된 추천 정확도를 보여주었다. 또한 클러스터링을 이용한 알고리즘은 에러율의 표준편차가 가장 작게 나타남으로써 가장 안정된 균일한 추천을 한다는 것을 알 수 있었고, 전체적으로 이러한 클러스터링은 어플리케이션이 한번의 추천을 하는데 소요되는 시간을 단축시켰다.

앞으로 더 연구되어야 할 것은 클러스터링의 결과로 잘 정제된 가상 사용자 조합을 적절히 형성하고, 이 가상 유저들에만 기초한 추천 에이전트를 개발하는 것이다. 한편 본 연구에서는 오프라인으로 클러스터링을 진행하여 그 결과를 적당한 포맷으로 미리 저장해 놓고, 추천에 이용했다. 이는 클러스터링에 소요되는 시간이 아직도 상당히 길고, 시스템 자원의 소비가 많기 때문이었다. 따라서 앞으로 더 빠른 시간에 시스템 자원을 덜 소비하면서도 실시간으로 동일한 클러스터링을 해 낼 수 있는 실시간의 점증적인 클러스터링 알고리즘에 대한 연구가 필요하다고 본다.

참 고 문 헌

[1] P. Maes, "Agent that Reduce Work and Information Overload," Communications of the ACM, Vol.37, No.7, pp.30-40, 1994.

[2] Oracle, Oracle iMarketing Release 3i, http://www.oracle.com

[3] A. Moukas, R. Guttman, and P. Maes, "Agent-Mediated Electronic Commerce : An MIT Media Laboratory Perspective," Proceedings of Int. Conf. on Electronic Commerce, Seoul Korea, pp.9-15, 1998.

[4] M. Ma, "Agents in E-Commerce," Communications of the ACM, vol.42, No.3, pp.79-80, 1999.

[5] P. Maes, R. Guttman, and A. G. Moukas, "Agents That Buy and Sell," Communications of the ACM, Vol.42, No.3, pp.81-91, 1999.

[6] J. B. Schafer, J. Konstan, and J. Riedl, "Recommender Systems in E-Commerce," Proceedings of the ACM Conference on Electronic Commerce, November 3-5, 1999.

[7] P. Resnic, N. Iacocou, M. Sushak, P. Bergstrom, and J. Riedl, "GroupLens : An Open Architecture for Collaborative Filtering of Netnews," Proceedings of the Computer Supported Collaborative Work Conference, 1994.

[8] N. Good, J. B. Schafer,, J. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl, "Combining Collaborative Filtering with Personal Agents for Better Recommendations," Proceedings of the American Association of Artificial Intelli-

gence, pp.439-446, 1999.

[9] D. Golfberg, D. Nichols, B. M. Oki, and D. Terry, "Using Collaborative Filtering to Weaves an Information TAPESTRY," Communications of the ACM, Vol.35, No.12, pp.61-70, 1992.

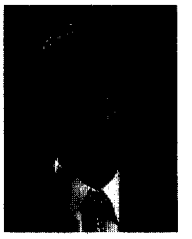
[10] J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl, "GroupLens : Applying Collaborative Filtering to Usenet News," Communications of the ACM Vol.40, No.3, pp.77-87, 1997.

[11] B. Mirkin, 'Mathematical Classification and Clustering', Kluwer Academic Publisher, pp.428, 1996.

[12] K. Alsabi, S. Ranka, and V. Singh, "An Efficient K-Means Clustering Algorithm," IPPS/SPDP Workshop on High Performance Data Mining, Orlando, 1998.

[13] R. T. Ng and J. Han, "Efficient and Effective Clustering Methods for Spatial Data Mining," Proceedings of the Int. Conf. on Very Large Database, Santiago, Chile, pp.144-155, 1994.

[14] L. Kaufman and P. J. Rousseeuw, 'Finding Groups in Data : an Introduction to Cluster Analysis', Wiley Series in Probability and Mathematical Statistics, pp.342, 1990.



황 병 연

e-mail : byhwang@dbs.cuk.ac.kr

1986년 서울대학교 전자계산기 공학과 졸업(공학사)

1989년 한국과학기술원 전산학과 졸업(공학석사)

1994년 한국과학기술원 전산학과 졸업(공학박사)

1999년~2000년 Univ. of Minnesota Visiting Scholar

1994년~현재 가톨릭대학교 컴퓨터전자공학부 부교수

관심분야 : 공간 데이터베이스(GIS), WWW 데이터베이스, 전자상거래