KIPS Trans. Softw. and Data Eng.
Vol.8, No.9 pp.363~372 pISSN: 2287-5905

https://doi.org/10.3745/KTSDE.2019.8.9.363

Purchase Transaction Similarity Measure Considering Product Taxonomy

Yu-Jeong Yang† · Ki Yong Lee††

ABSTRACT

A sequence refers to data in which the order exists on the two items, and purchase transaction data in which the products purchased by one customer are listed is one of the representative sequence data. In general, all goods have a product taxonomy, such as category/ sub-category/ sub-sub category, and if they are similar to each other, they are classified into the same category according to their characteristics. Therefore, in this paper, we not only consider the purchase order of products to compare two purchase transaction sequences, but also calculate their similarity by giving a higher score if they are in the same category in spite of their difference. Especially, in order to choose the best similarity measure that directly affects the calculation performance of the purchase transaction sequences, we have compared the performance of three representative similarity measures, the Levenshtein distance, dynamic time warping distance, and the Needleman-Wunsch similarity. We have extended the existing methods to take into account the product taxonomy. For conventional similarity measures, the comparison of goods in two sequences is calculated by simply assigning a value of 0 or 1 according to whether or not the product is matched. However, the proposed method is subdivided to have a value between 0 and 1 using the product taxonomy tree to give a different degree of relevance between the two products, even if they are different products. Through experiments, we have confirmed that the proposed method was measured the similarity more accurately than the previous method. Furthermore, we have confirmed that dynamic time warping distance was the most suitable measure because it considered the degree of association of the product in the sequence and showed good performance for two sequences with different lengths.

Keywords : Sequence Similarity Measure, Transaction Data Analysis, Product Taxonomy, Levenshtein Distance, Dynamic Time Warping

상품 분류 체계를 고려한 구매이력 유사도 측정 기법

양 유 정† · 이 기 용††

요 약

시퀀스란 두 항목 간의 순서가 존재하는 데이터를 말하며, 고객 한 명이 구매한 상품들이 나열된 구매이력 데이터는 대표적인 시퀀스 데이터 중 하나이다. 일반적으로 모든 상품은 대분류/ 중분류/ 소분류와 같은 상품 분류 체계를 가지며, 서로 다른 상품이더라도 비슷하다면 그 특성에 따라 동일한 범주로 분류된다. 따라서 본 논문에서는 두 구매이력 시퀀스 비교 시 상품의 구매 순서를 고려할 뿐만 아니라, 비교하고자 하는 두 상품이 다르더라도 서로 동일한 상품 군에 속한다면 더 높은 유사도를 부여하여 계산한다. 특히 구매이력 시퀀스 유사도 계산 성능에 직접적인 영향을 미치는 시퀀스 유사도 측정 방법을 선택하기 위해 본 연구에서는 대표적인 시퀀스 간 유사도 측정 방법인 레벤슈타인 거리, 동적 타임 워핑 거리, 니들만-브니쉬 유사도의 성능을 비교하였으며, 항목간의 계층구조도 반영하여 계산하도록 확장하였다. 기존의 유사도 측정 방법의 경우 시퀀스 내 상품 비교 시 상품의 일치 유무에 따라 단순히 0 또는 1의 값을 부여하여 계산한다. 하지만 제안 방법의 경우 서로 다른 상품이더라도 두 상품 간의 연관정도를 다르게 부여하기 위하여 상품 분류 트리를 사용하여 0에서 1 사이의 값을 가지도록 세분화하였다. 실험을 통해 세 알고리즘에 제안 방법을 적용한 경우 기존 방법에 비하여 구매이력 시퀀스 간의 유사도를 더 정확히 측정함을 확인하였다. 또한 정확성 측정 비교 실험을 통해 동적 타임 워핑 유사도가 다른 두 유사도 측정 방법에 비하여 시퀀스 내 상품의 연관 정도를 고려할 뿐만 아니라 두 시퀀스의 길이가 다른 경우에도 좋은 성능을 보였기 때문에 구매이력 데이터에서 시퀀스 간의 유사도 비교 시 가장 적합한 측정 방법임을 확인하였다.

키워드 : 시퀀스 유사도 측정, 구매이력 데이터 분석, 상품 분류 체계, 레벤슈타인 거리, 동적 타임 워핑

※ 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No. 2018-0-00269, A research on safe and convenient big data processing methods).
※ 이 논문은 2019년도 한국정보처리학회 춘계학술발표대회에서 '구매이력 데이터에서 상품 분류 체계를 고려한 시퀀스 유사도 측정 기법'의 제목으로 발표된 논문을 확장한 것임.

† 준 회 원 : 숙명여자대학교 컴퓨터과학과 석사과정
†† 종신회원 : 숙명여자대학교 소프트웨어학부 교수
Manuscript Received : July 5, 2019
Accepted : July 25, 2019
* Corresponding Author : Ki Yong Lee(kiyonglee@sookmyung.ac.kr)

1. 서 론

빅데이터 시대에 도래함에 따라 고객들은 자체적으로 엄청난 양의 새로운 디지털 데이터를 생성한다. 이렇게 생성된 데이터는 고객의 특성 자체가 되며 더 나아가 고객을 데이터로 표현할 수 있다. 구매 이력 데이터는 고객이 생성하는 대표적인 데이터 중 하나이다. 구매 이력 데이터에는 물건을 구매하는 사람의 특성과 이에 따라 소비되는 제품, 소비 패턴 등이 담겨있다. 따라서 기업은 주어진 고객의 데이터를 분석하는 능력이 필요하며, 분석된 결과를 바탕으로 의사 결정시 반영할 수 있다[1-3].

본 논문에서는 시간에 따른 고객들의 구매 항목들로 구성된 구매이력 데이터에서 임의의 두 고객의 구매이력이 얼마나 유사한지 계산하는 새로운 유사도 방법을 제안한다. 본 논문에서 제안한 유사도 측정방법은 구매이력 데이터와 같이 항목간의 순서가 존재하는 데이터 간의 유사도를 측정하는 기존의 유사도 방식을 항목 간의 계층관계까지 반영하도록 확장하였다. 구매이력 데이터는 구매 항목들 간의 순서가 담긴 데이터로 대표적인 시퀀스(sequence) 데이터 중 하나이다. 본 논문에서 제안하는 유사도 측정 방법은 기존의 구매이력 데이터 분석 방법들과 달리 시퀀스 내 항목간의 순서를 고려하여 유사도를 계산할 뿐만 아니라, 상품 분류 체계를 활용하여 구매한 상품 간의 서로 다른 유사도를 고려한다.

현재 기존 방법들만을 활용하여 구매이력 시퀀스 간의 유사도 측정하는 경우 시퀀스를 구성하는 상품의 일치 유무만이 유사도 판단 기준이 된다.

S_i : A Transaction Sequence

$S_1 = < Coke\ Sneakers >$
$S_2 = < Sprite\ Sneakers >$
$S_3 = < Jeans\ Sneakers >$

Fig. 1. Example of Transaction Sequences

이는 가령 구매이력 시퀀스가 Fig. 1과 같은 경우, 기존의 유사도 측정 방법은 시퀀스 S_1, S_2의 유사도와 S_1, S_3의 유사도를 동일하게 계산한다. Fig. 1의 시퀀스들은 모두 두 번째 구매 상품이 동일하기 때문에 각 시퀀스의 첫 번째 구매 상품만이 시퀀스 간 유사도에 영향을 미친다. 기존 방법들은 오직 상품의 일치 유무만을 판단하기 때문에 S_1의 'Coke'와 S_2의 'Sprite'를 비교하는 경우와 S_1의 'Coke'와 S_3의 'Jeans'를 비교하는 경우 같은 유사도를 부여한다. 하지만 실제로 'Sprite'는 'Jeans' 보다 상대적으로 'Coke'와 음료라는 동일 범주에 속하며, 이를 고려하여 'Coke'와 'Sprite' 비교 시 'Jeans' 보다 더 높은 유사도를 부여할 필요가 있다. 따라서 본 논문에서는 구매이력 시퀀스 간의 유사도 계산 시 상품 분류 체계를 고려하도록 기존의 시퀀스 유사도 측정 방법을 확장하여 더 정확히 두 구매이력 간의 유사도를 계산한다.

또한 본 논문에서는 항목들의 순서를 고려하여 유사도를 계산하기 때문에 유사도 계산 성능에 영향을 미치는 시퀀스 간 유사도 측정 방법을 선택하기 위하여 대표적인 시퀀스 유사도 측정 방법인 레벤슈타인(Levenshtein) 거리, 동적 타임 워핑(DTW: Dynamic Time Warping) 거리, 니들만-브니쉬(Needleman-Wunsch) 유사도의 성능을 비교하였다. 가상의 시퀀스 데이터를 사용한 실험을 통해, 본 논문에서는 상품 분류 체계를 고려할 뿐만 아니라 서로 다른 길이를 가지는 두 시퀀스에 대해서도 정확히 유사도를 측정하는 동적 타임 워핑 거리가 가장 적합한 유사도 측정 방법임을 확인하였다.

본 논문의 구성은 다음과 같다. 제2장에서는 본 논문에서 사용한 시퀀스 유사도 측정 방법들에 대하여 살펴본다. 제3장에서는 본 논문에서 제안한 유사도 측정 방법에 대하여 자세히 설명한다. 제4장에서는 제안 방법과 기존 방법과의 성능 평가 결과를 보이며, 제5장에서는 결론을 맺는다.

2. 관련 연구

2.1 구매이력 데이터 분석

고객의 소비활동이 담긴 대규모의 구매이력 데이터에는 고객의 소비패턴이 담겨 있다. 이렇게 고객의 구매내역에서 빈번하게 발생한 구매 패턴을 찾아 나가는 과정을 연관성 분석이라고 한다. 연관성 분석은 각 상품을 독립적인 개체로 판단하여 규칙을 생성하며, 여기서 장바구니 분석은 항목간의 계층관계까지 고려하는 분석이다. 이와 다르게 순차 패턴 분석은 구매 내역의 선후관계까지 고려하여 규칙을 생성한다. 세 가지 분석 모두 지지도(support), 신뢰도(confidence), 향상도(lift)라는 평가 기준을 활용하여 항목 간의 규칙을 생성한다[4].

고객들의 구매이력 데이터를 활용한 대표적인 사례로는 미국의 월마트(Wal-Mart)가 있다. 월마트는 매출에 직접적인 영향을 미치는 고객의 장바구니에 관심을 가진 최초의 업체로 방대한 양의 데이터를 분석하여 맥주와 기저귀의 관계를 밝혀냈다. 매출을 늘리기 위해 다양한 마케팅 활동을 전개하는 기업의 입장에서는 이러한 데이터를 활용하여 실질적인 제품 간의 관계를 파악하여 마케팅 전략으로 활용할 수 있다[5].

2.2 시퀀스 유사도 측정 방법

시퀀스(sequence)란 두 개 이상의 항목들로 구성된 데이터로 이 항목들 간에 순서가 담긴 데이터를 말한다. 대표적인 시퀀스 데이터로는 웹 로그 데이터, 단백질 시퀀스 데이터가 있다. 시퀀스 데이터를 분석하여 웹 로그 파일에서 비슷한 사용자들을 그룹화[6]하거나 비슷한 구조를 가지는 단백질 시퀀스들을 그룹화 하여 비슷한 기능을 갖는 단백질 시퀀스를 발견할 수 있다[7]. 항목간의 선후 관계가 존재하는 시퀀스 데이터에서 순서를 고려하여 유사도를 정의하는 것이 중요하며, 유사도

를 계산하는 방법에 따라 다음과 같이 나눌 수 있다.

1) 편집 기반(edit-based) 유사도 측정 방법

두 개의 문자열이 같아지기 위한 최소 수정 연산 횟수를 구하는 알고리즘이다. 수정 연산은 추가(add), 대체(substitute), 삭제(delete) 연산을 말하며 가장 대표적인 알고리즘은 레벤슈타인(Levenshtein) 거리 알고리즘이 있다. 수정 연산 횟수가 유사도 판단 척도로 사용되며 그 값이 작을수록 두 문자열이 유사하다고 판단한다.

2) 정렬(alignment) 유사도 측정 방법

주로 단백질 서열이나 핵산 서열 사이의 상관관계 분석 시 두 서열 간의 유사한 구역을 찾아낼 때 사용한다. 정렬 범위에 따라 국소(local) 정렬 방법과 전역(global) 정렬 방법이 있으며, 대표적인 국소 정렬 알고리즘으로는 스미스-워터맨(Smith-Waterman) 알고리즘[8]이 있고 전역 정렬 알고리즘으로 니들만-브니쉬(Needleman-Wunsch) 알고리즘이 있다. 두 알고리즘은 정렬 범위에 따라 두 시퀀스가 가장 유사하도록 공백(gap)을 사용하여 정렬한다.

3) 집합 기반(set-based) 유사도 측정 방법

문자열을 문자의 집합 혹은 토큰(token)의 집합 형태로 바꾸어 계산한다. 집합 관계를 이용하여 연산하며 문자열을 토큰으로 나누는 경우에는 N-gram 개념을 사용하여 문자열을 길이가 N개의 기준 단위로 절단하여 사용한다. 대표적인 알고리즘으로는 자카드(Jaccard) 유사도가 있다. 자카드 유사도는 집합을 구성하는 원소들 간의 합집합과 교집합 간의 비율을 나타내며 0에서 1 사이의 값을 가진다[9].

2.3 구매이력 시퀀스 유사도 측정 방법

본 절에서는 본 논문에서 제안하는 시퀀스 유사도 측정 시 성능 비교를 위해 사용하는 레벤슈타인 거리, 동적 타임 워핑 유사도, 니들만-브니쉬 유사도에 대하여 상세히 설명한다.

1) 레벤슈타인(Levenshtein) 거리

편집 거리 알고리즘으로도 알려져 있는 레벤슈타인 거리는 하나의 문자열을 다른 문자열로 변환하기 위해 필요한 연산의 최소 횟수를 의미한다[10]. 두 문자열 간의 수정 연산은 추가(insert), 대체(substitute), 삭제(delete) 연산을 말한다. 비교하고자 하는 두 문자열의 문자를 한자씩 비교하며 추가와 삭제 시 연산 비용은 1을 부여하고 대체 연산은 문자의 일치 유무에 따라 0 또는 1을 부여하여 유사도를 계산한다.

2) 동적 타임 워핑(DTW) 거리

속도가 다른 두 개의 시계열 패턴의 유사성을 측정하는 알고리즘으로 음성인식, 필기체 문자인식 등에 사용된다. 두 시계열 간의 거리를 최소화하는 방향으로 움직이면서 거리를 계산하기 때문에 유클리디안(Euclidean) 거리로 계산할 때와 달리 부분적으로 왜곡되거나 변형된 파형에 대해서도 계산할 수 있다. 예를 들어, 두 개의 시계열 그래프 A와 B가 각각 $A = a_1, a_2, a_3, ..., a_i$, $B = b_1, b_2, b_3, ..., b_j$ 벡터로 표현된다고 하자. Fig. 2는 두 시계열 그래프 A, B에 대하여 유클리디안(Euclidean) 유사도로 계산한 경우와 동적 타임 워핑 유사도로 계산한 경우이다. 동적 타임 워핑 유사도로 계산한 경우 유클리디안 유사도와 다르게 시계열 그래프의 한 점에서 다른 시계열 그래프의 하나 혹은 그 이상의 점에 대응하여 계산 할 수 있다[11]. 이러한 특성으로 인해 서로 다른 길이의 시퀀스에 대해서도 효과적으로 유사도를 계산할 수 있다.

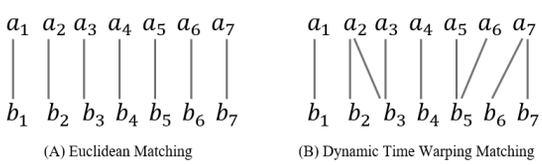

Fig. 2. Comparison of the Euclidean Distance and DTW Distance

3) 니들만-브니쉬(Needleman-Wunsch) 유사도

생물정보학(bioinformatics) 분야에서 단백질이나 뉴클레오타이드(nucleotide)의 시퀀스 간의 서열 비교를 위한 알고리즘으로, 두 시퀀스가 가장 유사도가 높도록 공백(gap)을 사용하여 두 시퀀스를 정렬한다. 공백 패널티(gap penalty), 일치(match), 불일치(mismatch) 값은 사용자 지정 값으로 값에 따라 시퀀스 간의 정렬 결과가 달라진다. 레벤슈타인 거리와 같이 두 시퀀스에 대한 삽입, 삭제, 일치 연산 비용을 활용하여 두 시퀀스 간의 유사도를 계산한다[12].

본 논문에서는 상기 3개의 유사도 측정 방법을 사용하여 유사도를 측정하였다. 이들은 모두 두 시퀀스 간의 항목을 하나씩 비교 가능하며 시퀀스를 집합 원소로 나누는 등의 별도 연산은 필요하지 않다. 하지만 기존 방법만을 사용하여 구매이력 시퀀스를 비교하는 경우, 상품 일치 유무만을 판단하여 0 또는 1의 값만을 부여하여 유사도를 계산하기 때문에 상품 간의 계층관계를 고려하도록 이를 확장하였다. 본 논문의 제안 방법은 상품 분류 트리를 활용하여 상품 간의 유사도를 0에서 1사이의 값으로 부여하여 유사도 계산 시 각 상품 간의 연관 정도를 보다 세분화한다. 따라서 본 논문에서는 구매이력 시퀀스 간의 유사도 계산 시 상품 분류 체계를 반영하기 위하여 이들의 수행 과정 중 일부분을 변형하여 사용하였으며, 제4.2절에서는 이들에 대한 성능 평가 결과를 보인다.

3. 상품 분류 체계를 고려한 시퀀스 유사도 측정

본 장에서는 본 논문에서 제안하는 상품 분류 체계를 활용하여 구매이력 데이터에서 두 시퀀스 간의 유사도를 계산하

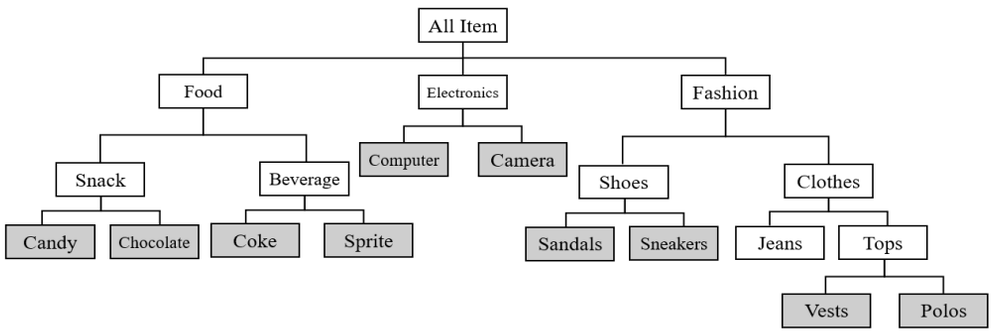

Fig. 3. Product Taxonomy Tree

는 제안 방법에 대해 자세히 설명한다.

3.1 개요

본 논문에서 제안하는 시퀀스 간의 유사도 측정 방법은 구매이력 데이터에서 주어진 상품 분류 체계를 활용하여 두 시퀀스 간의 유사도를 측정한다. 구매이력 데이터에서 하나의 구매이력은 고객 한 명이 순차적으로 구매한 항목들이 나열된 시퀀스 데이터이다. 이러한 임의의 두 시퀀스 데이터가 주어졌을 때, 본 논문에서 제안한 유사도 측정방법은 항목간의 순서를 고려하여 유사도를 측정한다. 또한 주어진 상품 분류 체계를 활용하여 상품 간의 분류 체계도 고려하여 유사도를 계산한다는 점에서 기존의 시퀀스 유사도 측정 방법과는 차이가 있다.

본 논문에서 제안하는 구매이력 시퀀스 간의 유사도 측정 방법은 대표적인 시퀀스 유사도 측정 방법인 레벤슈타인 거리, 동적 타임 워핑 거리, 니들만-브니쉬 유사도를 사용하여 시퀀스 내 항목 간의 유사도를 계산한다. 유사도 계산 시 상품 분류 체계를 반영하기 위해 비교하고자 하는 두 항목의 간의 연관성을 세분화하도록 수행 과정 중 일부분을 변용하였다.

3.2 문제 정의

본 절에서는 본 논문에서 시퀀스 유사도 계산 시 사용하는 상품 분류 트리의 개념과 본 논문에서 정의한 유사한 구매이력에 대하여 설명한다.

1) 상품 분류 트리

일반적으로 백화점, 마트와 같은 유통 업체는 모든 상품에 대하여 대분류/ 중분류/ 소분류와 같이 계층이 나누어진 상품 분류 체계를 가진다. 예를 들어 '바지'라는 상품은 먼저 소분류인 '하의'에 속하며, 여기서 하의는 다시 중분류인 '의류'에 속하게 된다. 이렇게 모든 상품은 상품 분류 체계의 가장 하위 부분을 차지한다. 본 논문에서는 이러한 상품 분류 체계를 트리(tree) 자료구조로 표현하였으며, Fig. 3은 본 논문에서 사용한 상품 분류 트리 중 일부이다.

본 논문에서 사용한 상품 분류 트리의 체계는 미국 전자상업 회사인 아마존(Amazon)의 분류 체계를 참고하여 구성하였다. 트리의 각 노드(node)에는 실제 상품이나 상품의 상위 범주명이 저장된다. Fig. 3의 상품 분류 트리에서 트리의 단말 노드(leaf node)는 상품에 해당하며, 내부 노드(internal node)는 각 상품에 대한 상위 범주이다. 이 상품 분류 트리는 구매이력 시퀀스 간의 유사도 비교 시 사용된다. 상품 분류 트리를 사용하여 시퀀스 내 항목들은 알고리즘 수행 과정 중 상품 분류 트리 내에서 서로 다른 연관정도에 따라 수치화되어 계산 시 반영된다.

본 논문에서 제안한 구매이력 시퀀스 간의 유사도 측정 방법은 Fig. 3과 같은 상품 분류 체계를 사용하여 항목 간의 유사한 정도를 계산한다. 따라서 두 상품이 일치하지 않더라도 상품 분류 트리 내에서 두 상품 간의 가까운 정도를 반영하여 계산하기 때문에 상품간의 유사한 정도를 세분화 하여 계산 할 수 있다.

2) 유사 구매이력 정의

본 논문에서 계산하는 구매이력 데이터는 각 고객들이 구매한 항목들이 순서대로 나열된 시퀀스 데이터이다. 구매이력 데이터에서 각 시퀀스는 고객 한명에 대한 구매이력을 나타내며, n개의 구매 항목으로 이루어진 시퀀스 S를 $S = < x_1 x_2 \dots x_{n-1} x_n >$라 하자. x_i는 시퀀스 S의 i번째 구매 항목을 나타내며, 이 시퀀스는 상품 x_1 부터 x_n 까지 순서대로 구매한 이력을 나타낸다.

본 논문에서 제안하는 시퀀스 간의 유사도 측정방법은 두 시퀀스의 세부 항목들이 다르더라도 항목들의 상위 범주가 같은 경우 유사하다고 판단한다. Fig. 4는 본 논문에서 정의한 유사한 구매이력 대한 예이다.

Fig. 4에서 두 시퀀스 S_1, S_2는 두 고객에 대한 구매이력 데이터로 S_1과 S_2는 모두 서로 다른 구매항목으로 구성되어 있다. 하지만 두 시퀀스의 세부 항목들은 매우 연관성이 높으며, 유사한 구매 순서를 보인다. 시퀀스 내 각 항목을 상위 범주로 바꿀 경우, 두 시퀀스 모두 음료, 간식, 의류 범주에 속

S_i : A Transaction Sequence

$S_1 = < Sprite\ Candy\ Jeans >$
$S_2 = < Coke\ Chocolate\ Skirt >$

⬇

$S_1 = < Beverage\ Snack\ Clothes >$
$S_2 = < Beverage\ Snack\ Clothes >$

Fig. 4. Example of Similar Sequences

한 상품 순으로 구성된 구매 이력인 것을 볼 수 있다. 이처럼 두 상품이 완전히 동일하지 않더라도 동일한 상위 범주에 속한다면 그렇지 않은 경우보다 더 높은 유사도를 부여할 필요가 있다. 따라서 본 논문에서는 상품 분류 트리를 활용하여 두 항목들이 다르더라도 서로 동일한 상위 범주를 가진다면 더 높은 유사도를 부여하여 계산한다.

3.3 시퀀스 유사도 측정

본 절에서는 제2.3절에서 설명한 대표적인 시퀀스 간의 유사도 측정 방법인 레벤슈타인 거리, 동적 타임 워핑 거리, 니들만-브니쉬 유사도를 사용하여 상품 분류 체계를 활용한 구매이력 시퀀스 유사도 측정 방법에 대해 상세히 설명한다. 상품 분류 트리를 적용하기 위하여 알고리즘의 수행 과정 중 일부분을 변용하였으며, Fig. 5, 7, 8의 의사코드 중 네모 칸 부분은 제안 방법을 적용하기 위하여 새롭게 변형한 부분이다.

세 알고리즘 모두 유사도 계산을 위해 구매이력 시퀀스 S_1과 S_2에 대한 2차원 배열 M을 생성한다. 각 알고리즘의 배열 $M[i][j]$의 값은 이전 원소($M[i-1][j]$, $M[i-1][j-1]$, $M[i-1][j]$)를 활용하여 구한다($0 \le i \le |S_1|,\ 0 \le j \le |S_2|$). 배열 M의 마지막 원소가 각 알고리즘 별 유사도 값이 되며, 알고리즘 별로 다른 유사도 값의 범위를 가진다. 다음은 각 알고리즘 별 유사도 측정 방법에 대하여 상세히 설명한다.

1) 레벤슈타인(Levenshtein) 거리

레벤슈타인 거리는 두 문자열을 한자 씩 비교해 나가며, 계산 후 배열의 가장 마지막 원소의 값이 두 문자열의 최소 편집 거리가 된다. 레벤슈타인 거리 Levenshtein Dist(S_1, S_2)는 0에서 $\max(|S_1|, |S_2|)$사이 값을 가진다.

두 시퀀스 S_1, S_2에 대하여 문자열이 없는 경우도 포함하여 2차원 배열 M을 행의 개수가 $|S_1|+1$이고 열의 개수가 $|S_2|+1$인 $(|S_1|+1)\times(|S_2|+1)$ 크기로 생성한다. 배열의 첫 번째 행과 열은 0에서부터 두 문자열의 길이만큼 증가시켜가며 초기화한다. 배열의 두 번째 행과 열부터는 이전까지 계산된 배열 값을 사용하여 채워나간다.

M에서 행에 해당하는 S_1은 원본 문자열을 의미하며 열에 해당하는 S_2는 바꾸고자 하는 목적 문자열을 의미한다. $M[i][j]$는 추가($M[i][j-1]+1$), 대체($M[i-1][j-1]+\text{cost}$), 삭제($M[i-1][j]+1$) 비용 중 가장 작은 값으로 채워진다. 삭제와 추가 연산은 이전 배열 값에 삭제, 추가 비용 1을 더한다. 대체 연산은 만일 현재 계산하는 $S_1[i-1]$번째 문자와 $S_2[j-1]$번째 문자가 일치한다면 대체 비용(cost)은 0을 더해주고, 다르다면 1을 더해준다. Fig. 5는 레벤슈타인 거리 알고리즘을 활용하여 본 논문에서 제안하는 구매이력 시퀀스 유사도 계산 방법을 나타내는 의사코드이다. Fig. 5의 의사코드 중 네모 칸은 새롭게 제안한 대체 연산 비용을 계산하는 함수로 Fig. 6의 의사코드가 수행된다. 만일 해당 부분이 문자 일치 유무에 따라 0 또는 1의 값을 가진다면 기존 레벤슈타인 거리 알고리즘이 수행된다.

```
function computeLevenshteinDistance(s1, s2)
  int M[|s1|+1][|s2|+1] = 0
  for i in (|s1|+1) do
    M[i][0] = i
  for j in (|s2|+1) do
    M[0][j] = j
  for i in (|s1| + 1) do
    for j in (|s2| + 1) do
      cost = computeItemRelevanceCost(i-1, j-1)
      M[i][j] = min(M[i][j-1] + 1,    //Insert
                    M[i-1][j] + 1,    //Delete
                    M[i-1][j-1]+cost))//Substitute
    endFor
  endFor
  return M[|s1|+1][|s2|+1]
endFunction
```

Fig. 5. Pseudo-code of Levenshtein Distance Algorithm

본 논문에서 제안하는 알고리즘은 기존의 레벤슈타인 거리 알고리즘의 대체 연산 중 더해지는 비용의 값을 상품 분류 트리를 이용하여 0에서 1사이로 세분화하여 보다 정확히 상품 간의 유사도를 계산한다. 따라서 본 논문에서 제안한 대체 연산 비용은 상품 분류 트리 내에서 가장 먼 두 항목인 경우에만 전혀 관련 없는 두 항목이라고 판단하여 최댓값 1을 부여하고, 그렇지 않은 경우에는 상품 분류 트리 내에서 서로 다른 두 상품 간의 최단 경로의 길이를 가지도록 대체 연산 비용의 값을 세분화 한다. Fig. 6은 기존의 대체 연산 비용 계산 방법과 달리 상품 분류 트리를 활용하여 두 상품간의 연관정도를 세분화하여 계산하는 제안 방법에 대한 의사코드이다.

```
function computeItemRelevanceCost(i, j)
  if (s1[i] != s2[j])
    longestPath = searchLongestPath()
    itemPath = searchPath(s1[i], s2[j])
    cost = itemPath/longestPath          /*식(1)*/
  else
    cost = 0
  return cost
endFunction
```

Fig. 6. Pseudo-code of the Proposed Method

비교하고자 하는 시퀀스 내 두 상품이 다른 경우 상품 분류 트리를 탐색하여 두 상품간의 연관정도를 계산한다. Equation (1)은 본 논문에서 제안하는 상품 분류 트리를 활용하여 두 상품관의 연관성을 계산하는 식이다.

$$cost = \frac{itemPath}{longestPath} \ (0 \le cost \le 1) \quad (1)$$

cost는 상품 분류 트리에서 $S_1[i]$번째 항목과 $S_2[j]$번째 항목과의 가까운 정도이다. cost 값은 상품 분류 트리 내에서 찾고자 하는 두 항목의 최단 경로의 길이가 가장 먼 두 항목의 길이에서 차지하는 비율을 나타낸다. longestPath는 상품 분류 트리 내에서 가장 먼 두 노드의 경로의 길이이며, itemPath는 상품 분류 트리 내에서 $S_1[i]$번째 항목과 $S_2[j]$번째 항목과의 최단 경로의 길이이다. 두 항목 사이의 최단 경로는 두 항목에서 가장 가까운 범주까지의 엣지(edge)의 수이다. 따라서 itemPath가 작을수록 두 항목은 높은 연관성을 가지며 가까운 범주에 분류되어 있는 상품임을 의미한다. itemPath가 0인 경우 두 상품은 같은 노드이며 cost는 0이 되고, itemPath가 1인 경우 longestPath와 같은 값을 가지게 되며 cost는 1이 된다. 레벤슈타인 거리 알고리즘의 경우 새롭게 계산된 비용은 대체 연산 시 더해지는 비용으로 추가, 대체, 삭제 비용 중 최솟값 선택 시 사용된다.

2) 동적 타임 워핑(DTW) 거리

동적 타임 워핑 알고리즘은 두 시계열 간의 거리를 최소화하는 방향으로 움직이면서 매칭시켜 누적 거리를 활용하여 거리를 계산한다. 시퀀스 S_1, S_2에 대하여 $(|S_1|)\times(|S_2|)$ 크기의 2차원 무한대 값을 가지는 배열 M을 생성한다. $M[i][j]$는 레벤슈타인 거리 알고리즘과 달리 비교하고자 하는 두 항목에 대한 비용(cost)을 먼저 계산한 후, 삭제($M[i][j-1]$), 일치($M[i-1][j-1]$), 삽입($M[i-1][j]$) 비용 중 최솟값에 더한다.

Fig. 7의 의사코드 중 네모 칸은 Fig. 6의 제안 방법을 적용한 동적 타임 워핑 유사도에 대한 의사 코드이며, 해당 부분이 두 값의 차이($|i-j|$)인 경우 기존의 동적 타임 워핑 알고리즘이 수행된다. 본 논문에서는 S_1의 한 지점에서 S_2의 다른 지점으로 할당되는 매핑(mapping) 비용을 두 값의 차이가 아닌 제안 방법을 통하여 0에서 1 사이의 값을 가지도록 하였다.

니들만-브니쉬 알고리즘은 두 시퀀스 간의 변화(mutation)를 최소화 하면서 공백을 사용하여 두 시퀀스의 유사도가 가장 높도록 정렬한다. 시퀀스 S_1, S_2에 대하여 $(|S_1|+1)\times(|S_2|+1)$ 크기의 2차원 배열 M을 생성하여 0으로 초기화한다. 알고리즘 수행을 위해 공백 페널티(gap penalty), 일치 보상(match award), 불일치 페널티(mismatch penalty) 세 변수에 대한 사용자 정의 값을 지정한다. 사용자 정의 값에 따라서 계산된 유사도

```
function computeDTWdistance(s1, s2)
  int M[|s1|][|s2|] = infinity
  M[0][0] = computeItemRelevanceCost(0, 0)
  for i in (1 to |s1|) do
    M[i][0] = computeItemRelevanceCost(i, 0)
  for j in (1 to |s2|) do
    M[0][j] = computeItemRelevanceCost(0, j)
  for i in (|s1|) do
    for j in (|s2|) do
      cost = computeItemRelevanceCost(i, j)
      M[i][j] = cost + min(M[i-1][j],    //Insert
                           M[i][j-1],    //Delete
                           M[i-1][j-1])  //Match
    endFor
  endFor
  return M[|s1|][|s2|]
endFunction
```

Fig. 7. Pseudo-code of DTW Algorithm

값의 범위가 달라지며, 본 논문에서는 상품 분류 트리를 활용한 연산 비용은 0에서 1 사이의 값을 가지므로 일치 보상은 1, 불일치 페널티는 0, 공백 페널티는 -1로 주었다. 사용자 정의 값이 다음과 같은 경우 니들만-브니쉬 유사도 NW Sim(S_1, S_2)는 $\pm \max(|S_1|, |S_2|)$ 사이 값을 가진다.

3) 니들만-브니쉬(NW: Needleman-Wunsch) 유사도

Fig. 8은 제안 방법을 적용한 니들만-브니쉬 알고리즘에 대한 의사 코드이며, 네모 칸은 상품 분류 트리 내에서 두 상품 간의 연관성을 계산하는 제안 방법에 대한 Fig. 6의 의사 코드이다. 만일 해당 부분이 비교 하고자 하는 두 항목($S_1[i-1]$, $S_2[j-1]$)의 일치 유무에 따라서 일치 보상(match award) 혹은 불일치 페널티(mismatch penalty)가 부여된다면 기존의 니들만-브니쉬 거리가 계산된다.

```
function computeNWsimilarity(s1, s2)
  int gap_penalty = -1
  int match_award = 1
  int mismatch_penalty = 0
  int M[|s1|+1][|s2|+1] = 0
  for i in (|s1|+1) do
    M[i][0] = gap_penalty * i
  for j in (|s2|+1) do
    M[0][j] = gap_penalty * j
  for i in (|s1| + 1) do
    for j in (|s2| + 1) do
      insert = M[i][j-1] + gap_penalty
      delete = M[i-1][j] + gap_penalty
      match = M[i-1][j-1] +
              1 - computeItemRelevanceCost(i-1, j-1)
      M[i][j] = max(insert, delete, match)
    endFor
  endFor
  return M[|s1|+1][|s2|+1]
endFunction
```

Fig. 8. Pseudo-code of Needleman-Wunsch Algorithm

4. 실험 결과

본 장에서는 본 논문에서 제안한 상품 분류 체계를 활용한 구매이력 시퀀스 간 유사도 측정 방법에 대한 성능 측정 결과를 보인다. 제2.3절에서 설명한 대표적인 유사도 측정 방법인 레벤슈타인 거리, 동적 타임 워핑 거리, 니들만-브니쉬 유사도에 대하여 기존 방법과 제안 방법과의 유사도 측정 비교 결과와 세 알고리즘에 대한 수행 시간 비교 결과를 보인다.

4.1 실험 환경 및 방법

본 논문에서 제안한 구매이력 데이터 시퀀스간의 유사도 측정 방법은 Python 3.7을 사용하여 동적 프로그래밍으로 구현하였으며, 상품 분류 트리는 anytree 라이브러리를 사용하여 구현하였다. 실험은 Intel i7-5820 3.3 GHz CPU, 8GB 메모리가 장착된 Windows 10 운영체제 환경의 PC에서 수행하였다.

실험에서 사용한 상품 분류 트리의 높이는 5이며, 총 37개의 노드를 가진다. 트리는 상품을 의미하는 26개의 단말 노드(leaf node)와 상품의 범주에 해당하는 11개의 노드로 구성된다. 실험에서는 시퀀스 내 항목의 개수가 3에서 10 사이를 가지는 가상의 시퀀스 데이터를 생성하여 사용하였다.

4.2 실험 결과

본 절에서는 다양한 시퀀스 간의 유사도 측정 방법에 따른 성능 측정 결과를 보인다.

1) 정확성 비교 실험

먼저 제안 방법이 기존의 시퀀스 유사도 측정 방법에 비하여 구매이력 데이터에서 시퀀스 간의 유사도를 더 정확히 계산할 수 있는지 평가하였다.

Fig. 9는 정확성 측정 실험에 사용한 가상의 시퀀스 데이터이다. Fig. 9에서 예제 1과 예제 2는 제안 방법이 기존의 유사도 방법에 비하여 구매 이력 시퀀스 간의 유사도를 더 정확히 계산하는지 측정하기 위한 실험 데이터로 예제 1은 가상의 시퀀스 데이터이며, 예제 2는 실제 소비자의 아마존 구매이력에 대한 실험 데이터이다. 마지막으로 예제 3은 서로 다른 크기를 가지는 시퀀스에 대한 유사도 측정 가상의 시퀀스 데이터이다.

Example 1) $S_1 = < Chocolate\ Camera\ Coke >$
$S_2 = < Jeans\ Beef\ Sneakers >$
$S_3 = < Candy\ Computer\ Sprite >$

Example 2)

	S_4	S_5	S_6
$item_1$	SNICKERS Ice Cream, Pint (4 Count)	Cutter & Buck Men's Cb Weathertec Summit Half-Zip Vest	David's Cookies Coconut Cloud - 10" Layer Cake
$item_2$	Levi's Men's 501 Original Jeans Customized by Sterling Shepard	Nikon AF-S FX NIKKOR 50mm f/1.8G Lens	Sprite Mini Can - 150ml (5.07fl oz)
$item_3$	Reef Comfortable Mens Slides Sandals	Laura's Lean 96% Lean Ground Beef - 1lb bricks - 8 per case	FRYE Men's Ludlow Chukka Sneaker

Example 3) $S_7 = < Chocolate\ Camera\ Coke >$
$S_8 = < Candy\ Chocolate\ Computer\ Camera\ Sprite\ Coke >$

Fig. 9. Examples of Sequence Data

예제 1의 S_1, S_2과 예제 2의 S_4, S_5는 서로 유사하지 않은 구매이력 시퀀스이며 예제 1의 S_1, S_3과 예제 2의 S_4, S_6는 서로 매우 유사도가 높은 구매 상품으로 구성된 시퀀스이다. 이를 통하여 제안 방법과 기존 방법과의 정확도를 비교하였다. 예제 3의 경우 S_8는 S_7과 시퀀스의 크기가 다르지만 크게 보면 동일한 범주의 구매순서(간식, 전자 제품, 음료 순)를 가진다. 이를 통해 서로 다른 길이를 가지는 시퀀스에 대해서도 정확히 유사도를 계산하는지 측정하였다.

Table 1은 Fig. 9의 가상의 시퀀스 데이터를 사용하여 기존의 세 알고리즘에서의 유사도 측정 결과와 제안 방법을 적용한 유사도 측정 결과에 대한 결과표이다. 예제 1에 대한 실험 결과, 기존의 레벤슈타인 거리와 니들만-브니쉬 유사도의 경우 세 시퀀스를 구성한 구매 상품이 모두 다르기 때문에

Table 1. Evaluation Results for Different Similarity Measures

	Example 1		Example 2		Example 3
Similarity Measure	$S_1 - S_2$	$S_1 - S_3$	$S_4 - S_5$	$S_4 - S_6$	$S_7 - S_8$
Levenshtein	3	3	3	3	3
New Levenshtein	2.5	0.75	2.625	0.65	3
DTW	9	3	9	3	3
New DTW	2.5	0.75	2.625	0.65	0.75
NW	0	0	0	0	0
New NW	0.5	2.25	0.55	2.75	0

동일한 수치를 계산한다. 하지만 동적 타임 워핑 거리의 경우 2차원 배열을 다른 두 유사도 측정 방법과 달리 두 값의 차이($|i-j|$)를 사용하여 초기화하기 때문에 S_1, S_2의 거리와 S_1, S_3의 거리가 다르게 계산되지만 상품 간의 연관 정도는 고려하지 않고 계산한다. 반면 제안 방법을 적용한 세 알고리즘의 경우 모두 상품 간의 유사도가 낮은 S_1, S_2에 비하여 상대적으로 시퀀스 내 상품 간의 연관성이 높은 S_1, S_3에 대하여 더 정확히 계산하는 것을 볼 수 있다. 또한 실제 데이터를 사용한 예제 2의 경우에도 예제 1과 같이 상품간의 연관성이 더 높은 두 시퀀스가 그렇지 않은 경우보다 더 높은 유사도를 보임을 실험을 통해 확인하였다. 이를 통해 기존의 방법들은 서로 다른 상품에 대해서 단순히 1을 부여 하여 계산하지만 제안 방법을 적용한 경우 상품 분류 트리를 고려하여 계산하기 때문에 서로 다른 상품으로 구성된 시퀀스라도 더 정확하게 유사도를 계산하게 된다.

예제 3의 실험 결과, 레벤슈타인 거리와 니들만-브니쉬 유사도의 경우 두 시퀀스의 상품 하나하나에 대해서만 비교하기 때문에 비슷한 구매 순서를 보이더라도 제안 방법과 기존 방법 모두 동일한 유사도 값을 보인다. 하지만 제안 방법을 적용한 동적 타임 워핑 거리의 경우 길이가 다르더라도 기존 방법으로 측정한 거리(3)와 달리 더 낮은 수치(0.75)를 계산한다. 이는 동적 타임 워핑 거리의 경우 누적 거리를 활용하여 시퀀스 내 하나의 항목이 다른 시퀀스의 여러 항목과 대응되어 계산하기 때문에 있기 때문에 S_7과 S_8의 길이가 다르더라도 더 정확하게 유사도가 계산된 것이라고 판단된다.

구매이력 시퀀스에 대한 실험 결과 제안 방법을 적용한 세 알고리즘 모두 기존의 알고리즘에 비해 상품간의 관련성을 고려하여 계산하기 때문에 더 정확하게 유사도를 측정함을 확인하였다. 또한 동적 타임 워핑 알고리즘의 경우 서로 다른 길이의 시퀀스에 대한 유사도 측정에 대해서도 다른 두 유사도 측정 방법과 달리 정확히 측정함을 실험을 통해 확인하였다.

2) 수행 속도 비교 실험

다음으로는 제안하는 시퀀스 유사도 측정 기법의 수행 시간을 측정하여 수용가능한 수준인지 평가하였다. 실험은 기존 방법과 제안 방법을 적용한 레벤슈타인 거리, 동적 타임 워핑 거리, 니들만-브니쉬 유사도에 대해서 측정하였다. 두 시퀀스가 주어졌을 때, 시퀀스 간의 유사도 측정 횟수를 2000번에서 10000번까지 증가시켜가며 수행 시간을 측정하였다. 시퀀스 유사도 측정 시 매번 다른 임의의 상품 구성과 길이를 가지도록 가상의 시퀀스 데이터를 생성하였다. Table 2는 시퀀스 유사도 측정횟수가 10000번인 경우에 대한 기존 방법의 수행 시간을 나타낸 표이며, Fig. 10은 제안 방법을 적용한 수행 시간 측정 결과이다. 실험 결과 연산 속도는 세 유사도 비교 방법 모두 시퀀스 내 상품의 개수와 시퀀스 내 상품 구성에 따라 의존하는 경향을 보였으며, 기존 방법과 제안 방법의 수행 시간은 10번 측정 후 중앙값을 취하였다.

Table 2. Evaluation Results for Conventional Methods

Similarity Measure	Levenshtein	DTW	NW
Processing Time (sec)	0.4	0.66	0.52

number of similarity measures = 1000

측정 결과 비교 횟수가 증가할수록 수행 시간이 선형적으로 증가하는 모습을 보이며, 세 알고리즘은 모두 비슷한 수행 시간을 보인다. 이는 세 알고리즘 모두 알고리즘의 수행 과정

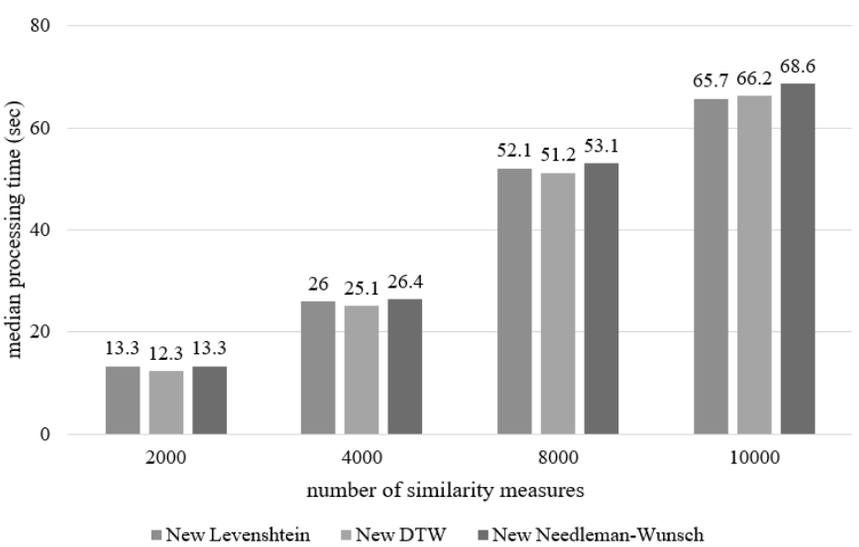

Fig. 10. Experimental Results for Proposed Methods

이 두 시퀀스에 대한 2차원 배열을 활용하여 계산하기 때문에 비슷한 수행 시간을 보인 것으로 판단된다. 또한 Table 2의 기존 방법의 수행 시간에 비하여 제안 방법의 수행 시간은 상품 분류 트리를 탐색 과정으로 인해 더 긴 수행시간을 보인다. 하지만 제안 방법을 적용한 세 가지 알고리즘 모두 수행시간은 최대 2분을 넘지 않으며, 이는 제안 방법의 수행 시간이 실제 사용 가능한 수준임을 나타낸다.

5. 결 론

본 논문에서는 기존의 시퀀스 유사도 측정방법을 확장하여 구매이력 데이터에서 상품 분류 체계를 고려하여 두 시퀀스 간의 유사도를 계산하는 새로운 방법을 제안하였다. 본 논문에서 계산하고자 하는 구매이력 시퀀스란 고객 한명이 구매한 항목들이 순서대로 나열된 데이터이다. 이러한 두 구매이력 시퀀스와 상품 분류 체계가 주어졌을 때, 기존의 유사도 측정 방법은 시퀀스를 구성한 구매 상품의 순서만을 고려하여 유사도를 계산한다. 이는 두 시퀀스가 구매 이력인 경우 비교하고자 하는 두 상품간의 연관성은 무시되게 된다. 따라서 본 논문에서는 시퀀스 내 구성 상품의 순서를 고려할 뿐만 상품 분류 체계를 활용하여 서로 다른 상품으로 구성된 시퀀스에 대해서도 보다 정확히 유사도를 계산한다.

본 논문에서는 구매이력 시퀀스에 적합한 유사도 측정 방법을 찾기 위해 다양한 시퀀스 유사도 측정방법들을 고려하였다. 이를 위해 현재 대표적인 시퀀스 유사도 측정 방법인 레벤슈타인 거리, 동적 타임 워핑 거리, 니들만-브니쉬 유사도를 사용했으며, 제안 방법을 적용하기 위해 이들의 수행 과정 중 일부분을 변형하였다. 기존의 세 알고리즘은 시퀀스 내 항목 간의 연산 비용 계산 시 상품 일치 유무에 따라 단순히 0 또는 1을 부여하였다면, 본 논문에서 제안한 방법은 상품 분류 트리를 사용하여 연산 비용을 0에서 1 사이의 값을 가지도록 세분화하였다. 새롭게 계산한 연산 비용은 비교하고자 하는 두 상품이 상품 분류 트리에서 가장 가까운 공통 범주까지의 경로를 의미하며, 가장 가까운 공통 범주가 루트 노드(root node)인 경우에만 1을 부여한다. 이를 통해 상품간의 서로 다른 중요도를 고려할 수 있으며 독립적인 개체로 계산할 때 보다 의미 있는 결과가 도출됨을 실험을 통해 확인하였다. 세 알고리즘에 대한 정확성 측정 비교 실험 결과 동적 타임 워핑 유사도가 다른 두 유사도 측정 방법에 비하여 시퀀스 내 상품의 연관 정도를 고려할 뿐만 아니라 두 시퀀스의 길이가 다른 경우에도 좋은 성능을 보였기 때문에 구매이력 데이터에서 시퀀스 간의 유사도 비교 시 가장 적합한 측정 방법임을 확인하였다.

본 논문에서는 시퀀스 유사도 측정 과정 중 서로 다른 상품이더라도 두 상품 간의 연관 정도를 계산하기 위하여 주어진 상품 분류 트리를 탐색하는 과정을 거친다. 추후 연구에는 시퀀스 내 상품 구성 복잡도를 높여가며 다양한 시퀀스 길이를 가지는 두 시퀀스에 대한 상품 분류 트리 고속화 탐색 방법과 유사도 측정 성능 향상 방법에 대하여 연구할 계획이다.

References

[1] M. Sforna, “Data mining in a power company customer database,” Electric Power Systems Research, 2000.

[2] C. Rygielski, J. Wang, and D. C. Yen, “Data mining techniques for customerrelationship management,” *Technology in Society*, Vol.24, No.4, pp.483-502, 2002.

[3] M. Kaur and S. Kang, “Market Basket Analysis: Identify the Changing Trends of Market Data Using Association Rule Mining,” *Procedia Computer Science*, Vol.85, pp.78-85, 2016.

[4] E.W.T Ngai, L. Xiu, and D.C.K Chau, “Application of data mining techniques in customer relationship management: A literature review and classification,” *Expert Systems with Applications*, Vol.36, No.2, pp.2592-2602, 2009.

[5] T. Brijs, G. Swinnen, K. Vanhoof, and G. Wets, “Using association rules for product assortment decisions: a casestudy,” in *Proc. of the Fifth International Conference on Knowledge Discovery and Data Mining*, pp.254-260, 1999.

[6] S. Park, N. C. Suresh, and B. K. Jeong, “Sequence-based clustering for Web usage mining: A new experimental framework and ANN-enhanced K-means algorithm,” *Data & Knowledge Engineering*, Vol.65, No.3, pp.512-543, 2008.

[7] E. Zorita, P. Cusco, and G. J. Filion, “Starcode: sequence clustering based on all-pairs search,” *Bioinformatics*, Vol.31, No.12, pp.1913-1919, 2015.

[8] T. F. Smith and M. S. Waterman, “Identification of Common Molecular Subsequences,” *Journal of Molecular Biology*, Vol.147, pp.195-197, 1981.

[9] P. Jaccard, “Étude comparative de la distribution florale dans une portion des Alpes et des Jura,” *Bulletin de la Société Vaudoise des Sciences Naturelles*, Vol.37, pp.547-579, 1901.

[10] Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” *Soviet Physics Doklady*, Vol.10, pp.707-710, 1966.

[11] D. Berndt and J. Clifford, “Using Dynamic Time Warping to Find Patterns in Time Series,” In *Proc. of KDD Workshop*, 1994.

[12] S. B. Needleman and C. D. Wunsch, “A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins,” *Journal of Molecular Biology*, Vol.48, pp.443-453, 1970.

양 유 정

https://orcid.org/0000-0002-7155-5367
e-mail : diddbwjd96@sookmyung.ac.kr
2019년 숙명여자대학교 소프트웨어학부 (학사)
2019년~현 재 숙명여자대학교 컴퓨터과학과 석사과정
관심분야 : 빅데이터, 데이터마이닝

이 기 용

https://orcid.org/0000-0003-2318-671X
e-mail : kiyonglee@sookmyung.ac.kr
1998년 KAIST 전산학과(학사)
2000년 KAIST 전산학과(석사)
2006년 KAIST 전산학과(박사)
2006년~2008년 삼성전자 소프트웨어연구소 책임연구원
2008년~2010년 KAIST 전산학과 연구조교수
2010년~현 재 숙명여자대학교 소프트웨어학부 교수
관심분야 : 데이터베이스, 데이터마이닝, 빅데이터, 데이터스트림