Journal of the Korea Industrial Information Systems Research Vol.27 No.5, Oct. 2022 :1-11
http://dx.doi.org/10.9723/jksiis.2022.27.5.001 ISSN:1229-3741

크라우드 소싱 기반 딥러닝 선호 학습을 위한 쌍체 비교 셋 생성+

(Generating Pairwise Comparison Set for Crowed Sourcing based Deep Learning)

유 기 현[1], 이 동 기[1], 이 창 우[1], 남 광 우[2]*
(Kihyun Yoo, Donggi Lee, Chang Woo Lee, and Kwang Woo Nam)

요 약 딥러닝 기술의 발전에 따라 학습을 통해 선호도 랭킹 추정을 하기 위한 다양한 연구 개발이 진행되고 있으며, 웹 검색, 유전자 분류, 추천 시스템, 이미지 검색 등 여러 분야에 걸쳐 이용되고 있다. 딥러닝 기반의 선호도 랭킹을 추정하기 위해 근사(approximation) 알고리즘을 이용하는데, 이 근사 알고리즘에서 적정한 정도의 정확도를 보장할 수 있도록 모든 비교 대상에 k번 이상의 비교셋을 구축하게 되며, 어떻게 비교셋을 구축하느냐가 학습에 영향을 끼치게 된다. 이 논문에서는 크라우드 소싱 기반의 딥러닝 선호도 측정을 위한 쌍체 비교 셋을 생성하는 새로운 알고리즘인 k-disjoint 비교셋 생성 알고리즘과 k-체이닝 비교셋 생성 알고리즘을 제안한다. 특히 k-체이닝 알고리즘은 기존의 원형 생성 알고리즘과 같이 데이터 간의 연결성을 보장하면서도 안정적인 선호도 평가를 지원할 수 있는 랜덤적 성격도 함께 가지고 있음을 실험에서 확인하였다.

핵심주제어 : 선호도 예측, 쌍체 비교 알고리즘, k-정규 그래프

Abstract With the development of deep learning technology, various research and development are underway to estimate preference rankings through learning, and it is used in various fields such as web search, gene classification, recommendation system, and image search. Approximation algorithms are used to estimate deep learning-based preference ranking, which builds more than k comparison sets on all comparison targets to ensure proper accuracy, and how to build comparison sets affects learning. In this paper, we propose a k-disjoint comparison set generation algorithm and a k-chain comparison set generation algorithm, a novel algorithm for generating paired comparison sets for crowd-sourcing-based deep learning affinity measurements. In particular, the experiment confirmed that the k-chaining algorithm, like the conventional circular generation algorithm, also has a random nature that can support stable preference evaluation while ensuring connectivity between data.

Keywords : Preference prediction, pairwise comparision, k-regular graph

* Corresponding Author: kwnam@kunsan.ac.kr
+ 이 연구는 2020년도 정부(교육부)의 재원으로 한국연구재단의 기초연구사업(No.2020R1F1A1048432)과 2021년 한국국토정보공사 공간정보연구원의 산학협력 R&D 지원사업 자유과제 지원에 의하여 수행된 연구임.

Manuscript received August 23, 2022 / revised September 15, 2022 / accepted September 26, 2022
1) 군산대학교 컴퓨터정보통신공학부
2) 군산대학교 컴퓨터정보통신공학부, 교신저자

1. 서 론

최근 기술의 발전과 함께 특정 분야의 문서나 이미지들을 이용하여 딥러닝 기반의 자동적인 선호도(preference) 평가를 지원하기 위한 다양한 시도가 이루어지고 있다. 이러한 선호도 평가 시스템 개발의 가장 첫 번째 단계이면서 중요한 부분은 다양한 사람들로부터 선호도에 대한 양질의 평가 데이터베이스를 크라우드 소싱 형태로 구축하는 것이다. 좀 더 구체적으로 설명하면, 크라우드 소싱 구축 단계에서 인간 작업자(human worker)들은 웹이나 모바일 앱 사용자 인터페이스 등을 통해 화면으로 여러 개의 사진들과 같은 비교 대상들을 보게 되며 본인의 선호도를 점수나 순서, 클릭과 같은 형태로 입력하게 된다. 이때 어떤 형태로 작업자에게 비교 평가 셋(comparison set)을 제시하고 선택하게 할 것인지에 따라 구축되는 크라우드 소싱 데이터의 품질에 영향을 끼치게 된다(Saha et al., 2019, Nam et al., 2020, Yoo, 2019, Jeong et al., 2021).

작업자에게 평가를 위해 비교 평가 셋을 제시하는 방법에는 비교 대상의 갯수 측면에서 2개에 대한 선호도를 비교 평가하는 단순한 방법과 n개를 인간 작업자가 직접 정렬하게 하거나 점수를 입력하게 하는 방법 등이 있다. 위 방법들 가운데 작업자에게 두 개의 비교 대상을 쌍으로 제시하고 작업자가 순위를 매기거나 선택하는 방법을 쌍체 비교(pairwise comparison)라 하며, 비교적 높은 정확도와 작업자의 집중도를 높일 수 있어서 가장 많이 사용되고 있다(Chen et al., 2013, Koczkodaj et al., 2015, Lee et al., 2022). 쌍체 비교는 의사 결정 및 투표, 선호도 연구에 널리 사용된다.

쌍체 비교 방법을 통해 배우들간의 선호도 평가 데이터베이스를 구축한다고 가정하자. 이때 선호도 평가의 가장 이상적인 결과는 가장 선호도가 높은 배우부터 낮은 배우들의 순으로 정렬 알고리즘을 이용하여 랭킹(ranking)을 부여하는 것이다(Furnkranz, 2003). 이때 정렬 알고리즘을 기반으로 인간 작업자들의 쌍체 비교 데이터를 구축해야 한다면 일반적으로 $n\log_2 n$ 정도의 쌍체 비교 셋이 필요하므로 너무 큰 비용을 초래하게 된다. 그러므로 딥러닝 등을 통해 선호도 랭킹 유추를 위한 근사(approximation) 알고리즘을 적용하게 된다. 이 근사 알고리즘에서 적정한 정도의 정확도를 보장할 수 있도록 모든 비교 대상에 대해 k번 이상의 비교셋을 구축하게 된다(Sunahase, et. al, 2017). 이때 모든 비교 대상에 대해 k번의 비교를 보장하는 데이터 셋을 생성한다는 것은 컴퓨터 알고리즘의 측면에서 Wormald(1999)에서 제안한 랜덤 k-정규 그래프(random k-regular graph)와 비슷하다. 그러나 랜덤 k-정규 그래프는 k개의 연결만이 보장될 뿐 랜덤 생성이라는 본질로 인해 생성된 그래프가 전역적으로 연결되어 있음을 보장하지 않는다. 즉, 생성된 쌍체 비교 셋이 전역적으로 골고루 연결되지 않는다면, 학습된 결과 또한 전역적 랭킹을 부여할 수 없으며 랜덤 k-정규 그래프 생성 알고리즘을 그대로 적용할 수 없다. 이 문제를 해결하기 위해 Burton(2003)은 원형 생성 알고리즘을 제안하였고, 쌍체 비교 데이터 셋 생성을 위해 지금까지 많이 사용되고 있다(Burton, 2003, Miranda et al., 2009). 그러나 이 알고리즘은 한 번의 원형 램덤 데이터를 생성해서 연결하여 생성하거나 스킵(skip)하며 생성하는 방법만을 지원하며, 짝수번의 비교생성에서만 동작하고, 한정된 배열에서 스킵을 통해 비교셋을 재생성하기 때문에 낮은 비교 횟수에서 생성된 비교 그래프가 클러스터 단위로 뭉쳐져 있는 단점이 있다.

이 논문에서는 k-disjoint 비교셋 생성 알고리즘과 k-체이닝 비교셋 생성 알고리즘을 제안한다. 특히 k-체이닝 비교셋 생성 알고리즘은 원형 랜덤 생성 알고리즘에 비해 다음과 같은 차별성을 갖는다.

- 비교 대상당 짝수만 생성가능한 제한이 없음.
- 매 반복당 랜덤 데이터를 재생성함으로써, 스킵으로 인한 클러스터 부분이 발생하지 않음
- 대상당 1회 비교를 지원하는 새로운 disjoint 랜덤 생성 알고리즘

이 논문의 2장에서는 관련 연구 및 생성된 데이터 셋에 대해 정량적으로 측정하기 위해 사용하는 방법들에 대해서 서술하며, 3장에서는 제안하는 새로운 알고리즘에 대하여 기술한다. 4장에

서는 제안하는 새로운 알고리즘들의 성능을 비교하고 분석한다.

2. 관련 연구

이 장에서는 작업자를 위한 쌍체 비교 데이터 셋을 생성하는 방법에 대하여 기술한다.

2.1 크라우드 소싱을 위한 선호 비교 방법

작업자의 평가를 위해 비교 평가 셋을 제시하는 방법에는 비교 대상의 개수 측면에서 2개에 대한 선호도를 비교 평가하는 단순한 방법과 n개를 인간 작업자가 직접 정렬하게 하는 방법 등이 있다. 또한, 방법의 측면에서는 작업자가 단순하게 선호를 둘 중에서 이진 선택(binary)하게 하는 방법과 선호나 소속의 정도에 대한 점수를 기재하도록 하는 방법, 스크롤바를 이용해 둘 간의 상대적 선호 강도를 표현하게 하는 방법, 이진 선택 방법에서 동등 점수를 추가해서 3개 중에 선택하게 하는 방법이 있다(Chen et al., 2013). 위 방법들 가운데 작업자에게 두 개의 비교 대상을 쌍으로 제시하고 작업자가 순위를 매기거나 선택하는 방법을 쌍체 비교(pairwise comparison) 기법이라 하며, 이 방법이 비교적 높은 정확도와 작업자의 집중도를 높일 수 있어서 많이 사용되고 있다(Koczkodaj et al., 2015). 이 논문에서는 대량의 데이터에서 크라우드 소싱 작업자에게 제시되는 쌍체 비교를 효과적으로 만드는 방법을 제시하는 데 목적을 두고 있다.

2.2 정렬형 비교 셋과 랜덤 k-정규 그래프

특정 데이터 셋에 대하여 사용자의 선호를 기반으로 순서를 정하거나 높은 점수부터 낮은 점수까지 부여함으로써 점수(score)화하는 것은 컴퓨터 정렬(sort) 알고리즘과 비슷하다(Kou et al., 2017). 퀵 정렬(quick sort)이나 병합 정렬(merge sort)에서 데이터 구성 요소들 간의 크기 비교 연산 부분이 크라우드 소싱 평가의 경우 사람(human) 기반의 평가로 대체되는 것이다. 컴퓨터 정렬 알고리즘의 평균 비교 횟수는 $O(nlogn)$이다(Saha et al., 2019). 30,000개의 데이터에 대하여 병합 정렬 알고리즘 기반의 비교 평가 셋을 생성한다면 446,180번의 비교를 필요로 한다. 퀵 정렬의 경우 최악의 비교 횟수는 $O(n^2)$이며, 이 때 필요한 비교 횟수는 9억 번이 된다.

컴퓨터 정렬 알고리즘 기반의 비교 데이터 셋은 정확도를 보장할 수 있지만 크게 세 가지 문제점을 갖는다. 첫째, 사람 기반의 비교 평가로 사용하기에 너무 많은 비교들을 필요로 한다. 둘째, 비교들의 집합을 미리 알 수 없으며, 앞의 비교가 끝난 다음에야 다음의 비교 쌍을 알 수 있다. 셋째, 동일 비교 쌍이라 할지라도 사람의 선호도로 인해 서로 다른 선택을 할 수 있으며 이에 대한 보완책이 필요하다.

랜덤 k-정규 그래프(random k-regular graph)는 각 꼭짓점에 k개의 모서리가 입사하는 랜덤하게 연결된 n개의 꼭짓점의 집합이다(Wormald, 1999, Kim and Vu, 2003). 이 정의는 그래프를 생성하기 위한 단순한 접근법과 마찬가지로 다소 간단하다. 이론적으로, 우리는 n개의 꼭짓점과 k개의 꼭짓점을 가진 모든 가능한 그래프를 고려해야 한다. 그리고 그중 하나를 무작위로 선택해야 한다. 불행히도, 가능한 k-정규 그래프의 공간은 너무 커서 이 접근법은 실제로 작동하지 않는다. 수년간의 연구에서 나타났듯이, 다른 방법으로 무작위 k-정규 그래프를 생성하는 것도 쉬운 일이 아니다. 그뿐만 아니라 Wormald(1999)와 Kim and Vu(2003)에서 제시된 알고리즘들은 랜덤 생성이라는 본질로 인해 생성된 그래프가 전역적으로 연결되어 있음을 보장하지 않는 문제가 있다. 생성된 그래프가 끊어져 있을 가능성이 조금이라도 있다면 전역적 랭킹을 부여할 수 없게 되므로 쌍체 비교 셋 생성을 위해 사용할 수 없게 된다. 그러므로 연결성이 보장되는 그래프 생성 알고리즘이 필수적이다.

2.3 원형 쌍체 비교 셋 생성 알고리즘

쌍체 비교 데이터 셋을 생성하는 방법들로 가장 많이 사용되는 방법들은 다음과 같이 크게 두 가지로 구분할 수 있다.

- 원형 랜덤 기법(cyclical generation) : 데이터 셋을 랜덤한 순서로 정렬한 후, 데이터의 처음 i번째 데이터와 i+1번째 또는 매 s번째 데이터를 비교 데이터 쌍으로 선정하는 방법이다. 제일 마지막 데이터에서 원형으로 연결된다(Burton, 2003). Fig. 1은 원형 랜덤 기법의 비교 데이터 쌍의 생성 예시를 보이고 있다. 모든 데이터에 대해 비슷한 수의 비교 쌍을 생성할 수 있지만, 하나의 데이터에 대한 비교 쌍의 갯수(r)가 짝수여야 하며 s가 1이상 일 때 그래프가 연결되지 않는 문제가 있다(Miranda, 2009). 또 하나의 단점은 r이 홀수일 때의 해결 방안이 없으며, 랜덤 배열을 한번만 생성하므로 r이 큰 수일 때는 같은 s를 단 한번만 사용할 수 있는 단점이 있다.
- 그룹 기법(group generation) : 유사성 등을 이용하여 데이터 셋을 m 크기의 그룹들로 분할하고, 그룹내의 비교셋과 그룹과 그룹 간 비교셋을 생성하는 방법이다. 그룹 간의 비교에서 원형 랜덤 기법을 이용하여 다시 생성할 수 있다. 데이터 간 비교 횟수가 일정하지 않은 단점이 있다(Whang, et. al, 2013).

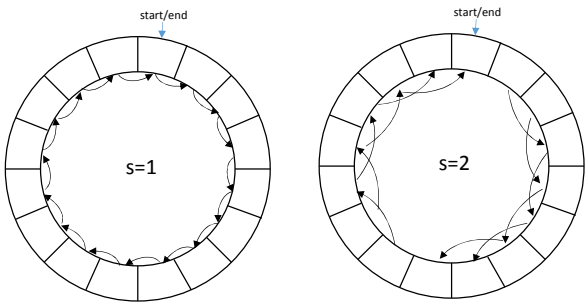


Fig. 1 Examples of cyclic random pairwise comparisions generation

이 연구에서는 데이터 크기에 따른 비용의 문제와 비교 횟수의 제한사항 등을 고려하여 원형 랜덤 기법을 기본으로 확장한 기법을 제안한다.

2.4 생성된 데이터 셋의 정량적 측정

사람 기반의 평가는 큰 비용을 필요로 한다. 그러므로 사람과 비용의 한계 내에서 비교 데이터 셋을 생성하기 위해서는 샘플링 기법이 사용된다. n개의 데이터 셋에 대하여 두 개의 쌍을 비교하기 위한 평가 데이터 셋(random dataset)의 생성 가능한 총 개수는 수식 1과 같다.

$$C(n,2) = \binom{n}{2} = \frac{n!}{2!(n-2)!} = \frac{n^2-n}{2} \quad \text{Eq. (1)}$$

위와 같이 비교 데이터 셋의 크기가 상당히 크기 때문에, 사람과 비용의 한계 내에서 좋은 샘플링 전략을 선택하는 것은 높은 품질의 크라우드 소싱 평가를 보장하게 된다.

비교 데이터 셋을 생성하는 아이디어들은 랜덤 그래프(random graph)에 기반하고 있다. 즉, n개의 데이터에 대해 최소 k개의 작업자 비교를 보장하는 비교 데이터 셋을 생성하는 것은 n개의 정점(vertex)과 각 정점이 최소 k개의 간선(edge)를 갖는 랜덤 그래프 $G_{n,k}$를 생성하는 것으로 볼수 있다.

일반적으로 그래프를 정량적으로 측정하기 위해 사용되는 방법들은 크게 차수 분산, 직경, 클러스터링 계수, 연결성과 같이 네 가지로 구분될 수 있다. 이 중에서 차수 분산은 k개 제한이 있으므로 배제될 수 있고 생성된 랜덤 비교 쌍 데이터 셋 그래프의 정량적 평가에서 사용될 때 의미가 있는 척도는 그래프의 직경과 평균 클러스터링 계수 등이다. 이를 기반으로 랜덤 그래프 $G_{n,k}$의 정량적 평가를 위한 척도와 영향에 대해 구체적으로 기술하면 다음과 같다.

- 연결성(connectivity): $G_{n,k}$의 모든 정점들이 단일 그래프에 연결되어 있어야 한다. $G_{n,k}$를 구성하는 서브 그래프간에 단절이 있다면 비교 대상이 없으므로 단일한 랭킹 결과를 생성할 수 없다.
- 직경(diameter): $G_{n,k}$를 구성하는 특정 노드에서 다른 노드들간을 연결하는 최대 거리가 균일하며 작아야 한다. 그래프의 직경이 크면 쌍체 비교의 효과가 전달되는 거리가 길어지게 되며 정확도가 떨어지게 된다.
- 클러스터링 계수 (clustering coefficient): $G_{n,k}$를 구성하는 서브 그래프들간에 강하고 균형있

게 연결되어 있어야 한다. 서브 그래프들간의 균형에 쏠림이 있다면, 두 서브 그래프를 구성하는 데이터들의 랭킹 결과에 왜곡이 발생할 수 있다.

이 논문은 연결성을 반드시 갖도록 보장하면서 낮은 직경계수와 낮은 클러스터링 계수를 갖는 알고리즘을 제시하는 것을 목적으로 한다.

3. 쌍체 비교 셋 생성 알고리즘

이 장에서는 이 논문에서 제안하고 있는 k-disjoint 및 k-체이닝 쌍체 비교셋 생성 알고리즘에 대해 기술한다.

3.1 k-disjoint 쌍체 비교셋 생성 알고리즘

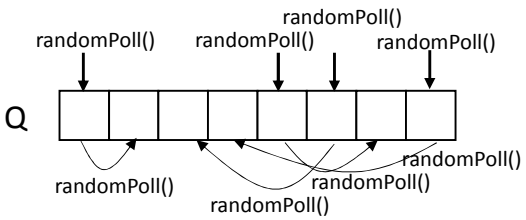


Fig. 2 An example of k-disjoint pair comparisons generation

이 절에서는 기본 알고리즘으로서 랜덤 기법에 기반하여 비교셋을 생성하는 새로운 알고리즘인 k-disjoint 비교셋 생성 알고리즘에 대해 기술한다. Fig. 2는 제안된 k-disjoint 비교셋 생성 알고리즘의 동작 예시를 보이고 있다. 그림에서 보이는 것과 같이 쌍체 비교 셋 생성을 위한 기본 데이터 D는 램덤 셔플링을 통해 섞여진 후 하나씩 반환하는 랜덤 셔플 큐인 Q에 저장되며, randomPoll() 함수를 이용하여 값을 하나씩 반환하게 된다. 이 알고리즘에서 주목할만한 부분은 데이터 D를 랜덤 셔플링을 통해 섞는 것이다. 랜덤 셔플링을 위해 가장 많이 사용되는 알고리즘은 Knuth(1969) 알고리즘으로 알려진 Fisher & Yates(1953) 셔플링 알고리즘이며, RandomShuffleQueue()는 이 알고리즘을 구현하고 있다.

Fisher & Yates(1953) 셔플링 알고리즘은 기본 데이터에서 추출할 위치를 랜덤하게 생성된 값에 의해 선택하게 함으로서 편향을 줄이는 방법이다. Q.randomPoll() 함수는 랜덤값에 의해 큐에 있는 값들 중에서 임의의 값을 선택하여 반환하고 Q에서 삭제하게 된다. Fig. 2는 randomPoll()들이 기본 데이터의 임의의 위치에서 값을 추출하는 예를 보이고 있다. k-disjoint 비교셋 생성 알고리즘은 랜덤 셔플링 알고리즘을 기반으로 서로 연결되지 않으면서(disjoint), 겹치지 않는 쌍으로 비교 데이터 쌍을 생성하는 알고리즘이다.

Algorithm : kDisjointPairComparisions

Input: D: list of ids in target data ;
R: set of pairwise comparisons ;
Output: **R** set of pairwise comparison

```
1  Q ← RandomShuffleQueue(D)
2  c1 ← Q.randomPoll()
3  while c1! = NULL∧!Q.empty() do
4      c2 ← Q.randomPoll()
5      while R.exists(sort(c1, c2)) do
6          Q.add(c2)
7          c2 ← Q.randomPoll()
8      end
9      R.add(sort(c1, c2))
10     c1 ← Q.randomPoll()
11     if c1! = NULL ∧ Q.isEmpty() then
12         t ← RandomChoose(D)
13         while R.exists(sort(c1, t) do
14             t ← RandomChoose(D)
15         end
16         R.add((c1, t))
17     end
18 end
```

Fig. 3 Algorithm for generation of k-disjoint pair comparision set

Fig. 3은 k-disjoint 비교쌍 생성 알고리즘을 보이고 있다. Fig. 3의 3번줄부터 9번줄까지에서 보이는 것과 같이 randomPoll()에 의해 i 번째 값을 추출하고, 다음의 i+1 번째 랜덤 추출된 데이터를 이용하여 하나의 비교 쌍 (c_i, c_{i+1})을 생

성한다. 그 다음의 비교쌍은 앞의 비교쌍과 연결없이 새로운 c_{i+2}와 c_{i+3}를 추출하여 비교 쌍 (c_{i+2}, c_{i+3})으로 생성하는 알고리즘이다. 끝으로 알고리즘의 10번줄부터 16번줄에서 보이는 것과 같이 데이터가 홀수여서 가장 마지막 데이터 하나가 남을 경우에는 기본 데이터에서 임의로 하나를 선택하고 쌍으로 설정한 후 비교 셋으로 추가된다. 이렇게 기본 데이터가 홀수일 때 마지막 1개 세트 중 D에서 추출된 데이터만 제시된 r보다 1이 큰 비교 횟수를 갖는다.

Algorithm : ExtendedkDisjointPairComparisions

Input: D: list of ids in target data ;
k: minimum comparisons per each data ;
Output: R: set of pairwise comparison

```
1  R ← ∅
2  while k ≥ 1 ∧ |D| ≥ 2 do
3  |  DisjointPairComparisions(D, R)
4  |  k ← k − 1
5  end
6  return R
```

Fig. 4 Algorithm for generation of extended k-disjoint pair comparision set

Fig. 3의 k-disjoint 비교셋 생성 알고리즘이 보편적으로 사용되기 위해서는 대상체당 최소 k번의 비교셋이 랜덤하게 생성되도록 알고리즘이 확장될 필요가 있다. Fig. 4는 k-disjoint 비교셋 생성 알고리즘을 확장한 k-disjoint 쌍체 비교셋 생성 알고리즘을 보이고 있다. 이 알고리즘은 k-disjoint 비교셋 생성 알고리즘을 k번 반복하는 것으로 간단하게 구현될 수 있다. 이 알고리즘은 k번 비교셋 생성 시의 완전한 랜덤성을 보장하기 위해 제안되었으나, 기존의 원형 랜덤 알고리즘의 결과셋이 고정되어 나타나는 클러스터링 단점을 보완하고 있다. 그러나 랜덤 생성을 기본으로 수행하기 때문에 원형 랜덤 알고리즘의 가장 큰 장점인 대상 데이터 간의 연결성에 대해 완전한 보장이 이루어지지 않는 단점이 있다. 이 논문에서는 이것을 보완하기 위해 다음 절의 k-체이닝 쌍체 비교셋 생성 알고리즘을 함께 제안한다.

3.2 k-체이닝 쌍체 비교셋 생성 알고리즘

이 절에서는 두번째 새로운 알고리즘인 k-체이닝 쌍체 비교셋(k-chaining pair comparions) 생성 알고리즘에 대해서 설명한다. 원형 랜덤 생성 기법은 대상 데이터간의 완전한 연결성을 보장하지만 데이터에 대한 비교 쌍의 갯수가 짝수개 일 때만 응용 가능하며, s가 작을 때는 그래프의 랜덤성이 약해지는 단점이 있다. 또한, 앞의 절에서 제안한 k-disjoint 비교셋 생성 알고리즘은 랜덤성은 확보되지만 연결성은 보장하지 않는 단점이 있다. 이 연구에서는 원형 랜덤 기법을 활용하면서 k-disjoint 알고리즘을 함께 사용할 k-체이닝 비교셋 생성 알고리즘을 제안한다.

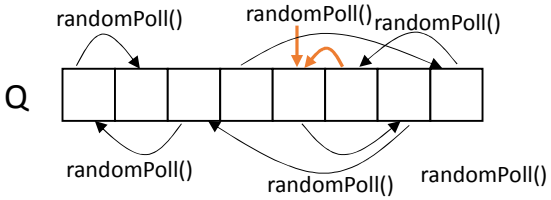


Fig. 5 An example of k-chaining pair comparions generation

Fig. 5는 제안된 체이닝 랜덤 생성 알고리즘의 동작의 예를 보이고 있다. 쌍체 비교 셋의 기본 데이터는 데이터를 램덤 셔플링을 통해 하나씩 반환하는 랜덤 우선순위큐 Q에 저장된다. 이 Q는 randomPoll() 함수를 이용하여 값을 하나씩 반환하게 되는데, i 번째 값을 추출하고, 다음의 i+1 번째 랜덤 추출된 데이터를 이용하여 하나의 비교 쌍 (c_i, c_{i+1})을 생성한다. 그 다음 비교쌍은 c_{i+2}을 추출하여 비교 쌍 (c_{i+1}, c_{i+2})을 연결하여 추출하게 된다. 끝으로 가장 마지막에 생성된 데이터는 제일 첫번째 추출된 head 값과 쌍으로 비교셋이 생성된다. k-disjoint 비교셋 생성 알고리즘이 각 쌍을 연결되지 않는 개별적인 쌍으로 생성되며 연결되지 않는 것에서 k-체이닝 비교셋 생성 알고리즘과 비교될 수 있다.

제안된 k-체이닝 비교쌍 생성 알고리즘의 기

본 아이디어는 원형 생성 알고리즘과 유사하지만, 반복 단계의 스킵을 통해 추가적인 비교셋을 생성하지 않고, 체이닝 비교 셋 생성 단계를 반복한다는 점에서 차별성을 갖는다. 또한, 생성단계에서 연결성을 갖도록 비교셋을 생성하므로 하나의 데이터에 대해 항상 2개 씩의 비교 셋이 생성되어야 하고 k가 짝수 일 때 만 사용할 수 있다. 그러므로 이 연구에서는 r이 2보다 큰 홀수일때는 체이닝 비교쌍 생성 방법을 이용해 쌍 비교 데이터 세트를 생성하다가 마지막 남은 1개의 비교 데이터 세트는 disjoint 쌍 생성 알고리즘을 이용하여 1:1의 짝을 생성하여 단 1개씩의 비교 셋만 생성되게 한다.

Algorithm : kChainingPairComparisions

Input: D: list of ids in target data ;
k: minimum comparisons per each data ;
Output: R set of pairwise comparison

$R \leftarrow \emptyset$
$Q \leftarrow RandomShuffleQueue(D)$
while $k \geq 2 \wedge |Q| \geq 2$ **do**
  $head, \leftarrow c_1 \leftarrow Q.randomPoll()$
  while $!Q.empty()$ **do**
    $c_2 \leftarrow Q.randomPoll()$
    while $R.exists(sort(c_1, c_2)$ **do**
      $Q.add(c_2)$
      $c_2 \leftarrow Q.randomPoll()$
    end
    $R.add(sort(c_1, c_2))$
    $c_1 \leftarrow c_2$
  end
  $R.add((head, c_1))$
  $Q \leftarrow RandomShuffleQueue(D)$
  $k \leftarrow k - 2$
end
if $k = 1 \wedge |D| \geq 2$ **then**
  $DisjointPairComparisions(D, R)$
end
return R

Fig. 6 Algorithm for generation of k-chaining pair comparison set

Fig. 6은 k-체이닝 랜덤 비교쌍 기법의 알고리즘을 보이고 있다. 이 알고리즘에서 특이한 점은 랜덤 우선순위 큐의 사용과 생성된 비교 쌍에 대하여 이전에 생성된 적이 있는지를 검사하는 부분이다. 랜덤 우선순위 큐는 랜덤하게 데이터를 선택하는 우선순위 큐이며, 체이닝 형태로 비교쌍이 생성되는 알고리즘을 단순화하기 위해 사용되었다. 예를 들면, 비교 쌍 존재 여부를 검사하는 과정에서 k가 4보다 큰 경우 체이닝 랜덤 비교 쌍 생성이 두 번 이상 반복되는데, 이때 동일한 비교 쌍 생성을 방지하기 위해 사용되었다. 또한, 알고리즘에서 비교 쌍을 추가하거나 존재 여부를 확인하기 위해서 항상 데이터의 식별자를 정렬한다.

4. 실험 및 평가

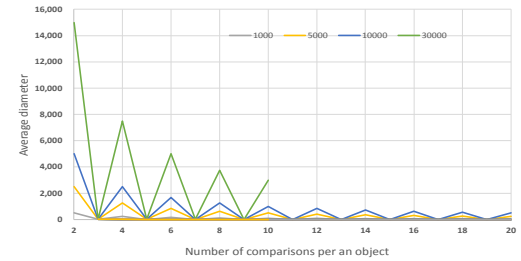
이 장에서는 기존의 원형 랜덤 비교셋 생성 알고리즘과 이 논문에서 제안하고 있는 k-disjoint 및 k-체이닝 비교셋 생성 알고리즘의 성능을 비교하고 평가한다.

4.1 실험 설정

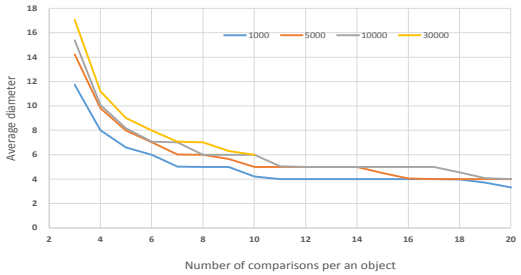
제안된 k-disjoint 및 k-체이닝 비교셋 생성 알고리즘의 성능에 대한 실험은 크라우드 소싱 기반의 딥러닝 보행성 평가용 데이터를 구축하기 위한 쌍체 비교셋을 대상으로 하였다. 데이터 크기에 대한 안정성을 평가하기 위해 데이터를 1000개로 시작하여 5000개, 10000개, 30000개로 크기를 증가시켜가며 실험을 하였다. 또한, 하나의 대상체 당 최소 비교 횟수를 2회에서 20회까지 증가시키면서 그에 따른 성능 변화에 대한 실험을 수행하였다. 생성된 그래프의 성능을 평가하기 위하여 2.4절에서 서술한 것과 같이 생성된 그래프의 직경과 클러스터링 계수를 이용한다.

4.2 실험 및 평가

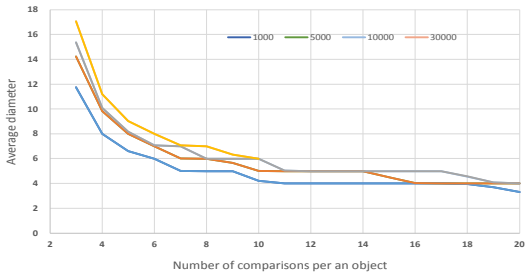
첫 번째 실험은 기존의 원형 랜덤 비교셋 생성 알고리즘 및 k-disjoint와 k-체이닝 비교셋 생성 알고리즘에 의해 생성된 그래프들의 직경을 비교하는 것이다. 생성된 그래프의 직경이 크다는 것은 데이터 간의 거리가 멀다는 것을 의미하며, 이것은 더 많은 비교를 해야 정해진 적정한 수준의 선호도 정확도에 도달할 것임을 의미한다.



(a) Cyclical pair comparison



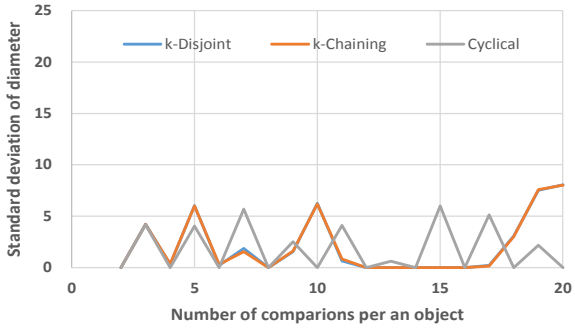
(b) k-Disjoint pair algorithm



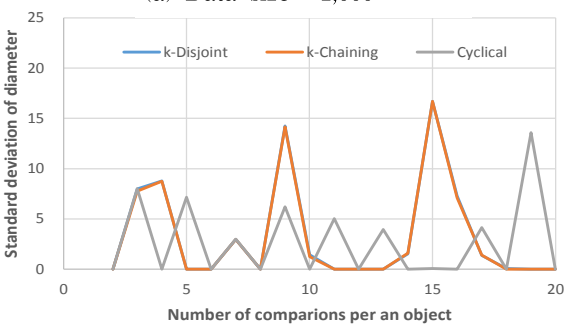
(c) k-Chaining pair algorithm

Fig. 7 Average of diameters against number of comparisons per an object

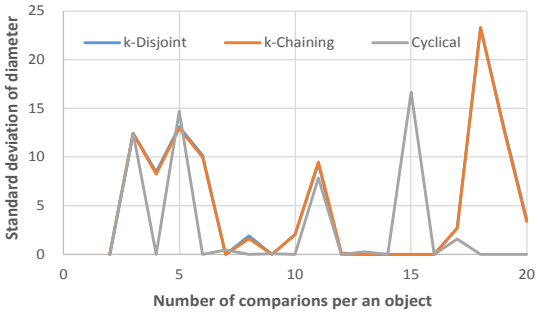
Fig. 7은 3개의 알고리즘들에 대해 데이터 개수 1000개로 시작하여 5000개, 10000개, 30000개일 때에 대하여 대상체 당 비교 횟수를 2회에서 20회까지 증가시켜가면서 측정한 평균 직경을 보이고 있다. 그림에서 보이는 것과 같이 원형 랜덤 생성 알고리즘은 가장 작은 5000개 데이터에 대해 비교 횟수가 2회일 때에도 2000이 넘는 직경을 가지며, 점점 감소하는 특성을 갖는다. 제안하고 있는 k-disjoint와 k-체이닝 알고리즘은 랜덤성의 특성을 잘 반영하고 있으며, 데이터의 크기와 관계없이 18 이하의 직경을 가지며, 데이터의 크기에 따른 영향도가 거의 없는 것을 확인할 수 있다. 특히, k-체이닝 알고리즘은 랜덤 알고리즘인 k-disjoint 알고리즘과 매우 유사한 특성을 가지면서도 원형 랜덤 생성 알고리즘과 동일하게 연결성을 보장하고 있다는 점에서 제안된 알고리즘의 우수성을 확인할 수 있다. 또한, 소숫점 두자리까지 확인했을때 k-체이닝 알고리즘이 k-disjoint 알고리즘에 비해 약간씩 우수한 점도 발견되었다.



(a) Data size : 1,000



(b) Data size : 5,000

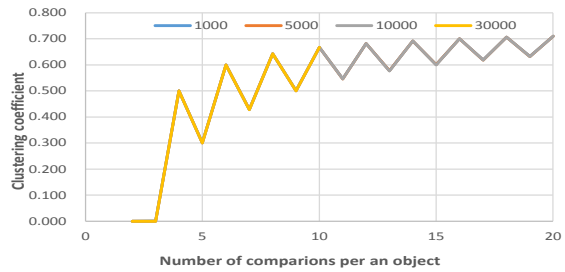


(c) Data size : 10,000

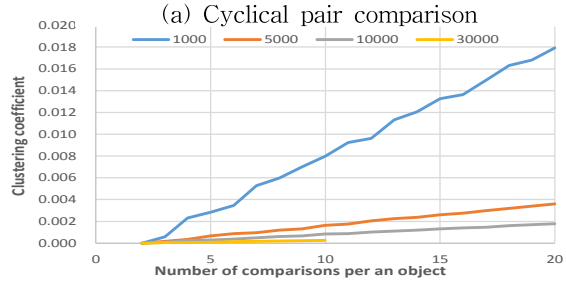
Fig. 8 Standard deviation of diameters against number of comparisons per an object

Fig. 8은 3개의 알고리즘들에 대해 데이터 개수 1000개, 5,000개, 10,000개일 때의 직경의 표준편차를 보이고 있다. 표준 편차는 데이터의 개수

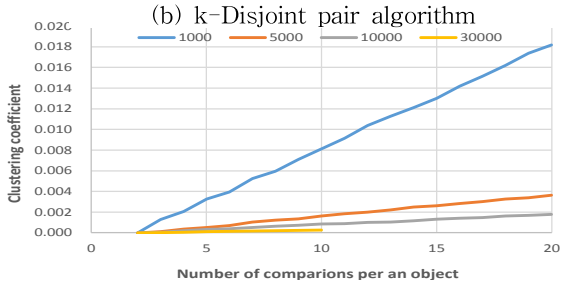
가 많아질수록 최대 8정도에서 23정도까지 증가하고 있는 것을 확인할 수 있다. 이 그림에서 k-disjoint 알고리즘과 k-체이닝 알고리즘이 매우 유사한 특성을 보여 마치 하나의 선인 것처럼 보이는 것도 확인할 수 있다.



(a) Cyclical pair comparison



(b) k-Disjoint pair algorithm



(c) k-Chaining pair algorithm

Fig. 9 Clustering coefficient against number of comparions per an object

두 번째 실험은 기존의 원형 랜덤 비교셋 생성 알고리즘 및 k-disjoint와 k-체이닝 비교셋 생성 알고리즘에 의해 생성된 그래프들의 클러스터링 계수를 비교하는 것이다. 생성된 그래프의 클러스터링 계수가 크다는 것은 데이터가 특정 부분에서 응집되어 있을 확률이 높다는 것을 의미한다. 이것은 클러스터로 응집되어 있는 부분에서는 비교적 높은 정확도로 선호도 비교 결과를 얻을 수 있으나 다른 부분에서는 낮은 결과를 보이는 것을 의미한다. 그러므로 알고리즘들은 낮은 클러스터링 계수를 갖는 것이 좋다.

Fig. 9는 데이터의 개수를 1,000개부터 30,000까지 증가시키고, 대상체 당 비교 횟수를 2회에서 20회까지 증가시켰을 때의 클러스터링 계수 측정 실험을 수행한 결과이다. 이 그림에서 보이는 것과 같이 기존의 원형 랜덤 생성 알고리즘은 0.7 정도 수준의 높은 클러스터링 계수를 나타내고 있지만, 제안된 k-disjoint 알고리즘과 k-체이닝 알고리즘은 둘다 최대 0.018 정도의 매우 낮은 수준의 클러스터링 계수를 보여 매우 우수한 점을 확인할 수 있다. 또한, k-disjoint 알고리즘은 1,000개 데이터에서 확연히 보이는 것과 같이 랜덤성을 반영하여 k-체이닝 알고리즘보다 불규칙적으로 흔들리는 점도 그림에서 확인된다.

5. 결 론

이 논문에서는 크라우드 소싱 기반의 딥러닝 선호도 측정을 위한 쌍체 비교 셋을 생성하는 새로운 알고리즘인 k-disjoint 비교셋 생성 알고리즘과 k-체이닝 비교셋 생성 알고리즘을 제안하였다. 특히, k-체이닝 알고리즘은 기존의 원형 랜덤 생성 알고리즘과 같이 데이터 간의 연결성을 보장하면서도 안정적인 선호도 평가를 지원할 수 있는 랜덤적 성격도 함께 가지고 있음을 실험에서 확인하였다. 향후 연구에서는 비교 데이터셋 생성과정에서 잦은 스킵으로 인한 성능 평가와 제안된 알고리즘을 기반으로 실제 딥러닝 선호도 판단을 수행했을 때 더 높은 정확도를 보장할 수 있는지 확인하는 연구가 필요하다.

References

Burton, M. L. (2003). *Too Many Questions? The Uses of Incomplete Cyclic Designs for Paired Comparisons*. Field Methods. 15(2), 115 - 130.

Chen, X., Bennett, P.N., Collins-Thompson, K. and Horvitz, E. (2013). *Pairwise ranking*

aggregation in a crowdsourced setting. In proceedings of the 6th ACM International Conference on Web Search and Data Mining(WSDM), 193 - 202.

Fisher, R. A., & Yates, F. (1953). *Statistical tables for biological*. Agricultural and Medical Research. Hafner Publishing Company, 26-27.

Furnkranz, J., & Hullermeier, E. (2003). *Pairwise preference learning and ranking*. Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), 2837, 145-156.

Jeong, Y. & Lee., C. (2021). *Indoor autonomous driving through parallel reinforcement learning of virtual and real environments.* Journal of the Korea Industrial Information Systems Research, 26(4), 11-18.

Kim, J. H., & Vu, V. H. (2003). *Generating random regular graphs*. Proceedings of the thirty-fifth annual ACM symposium on Theory of computing, STOC'03, 213-222.

Koczkodaj, W. W., & Szybowski, J. (2015). *Pairwise comparisons simplified*. Applied Mathematics and Computation, 253, 387 - 394.

Knuth, D. E. (1969). *Seminumerical algorithms*. The Art of Computer Programming. Vol. 2. Reading, MA: Addison Wesley. 139-140.

Lee, K., Nam, K., & Lee, C. (2022). *A study on the walkability scores in Jeonju city using multiple regression model*. Journal of the Korea Industrial Information Systems Research, 27(4), 1-10.

Nam, S., Lee, M., Heo, C., & Choi, K. (2020). *Cost-effective multi-task crowdsourcing method for knowledge extraction*. KIISE Transactions on Computing Practices, 26(11), 507-512.

Miranda, E., Bourque, P., & Abran, A. (2009). *Sizing user stories using paired comparisons*. Information and Software Technology, 51(9), 1327-1337.

Saha, A., Shivanna, R. & Bhattacharyya, C. (2019). *How many pairwise preferences do we need to rank a graph consistently?*. 33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 4830 - 4837.

Sunahase, T., Baba, Y., & Kashima, H. (2017). *Pairwise HITS: Quality estimation from pairwise comparisons in creator-evaluator crowdsourcing process*. 31st AAAI Conference on Artificial Intelligence, AAAI 2017, Kleinberg, 977-983.

Whang, S. E., Lofgren, P., & Garcia-Molina, H. (2013). *Question selection for crowd entity resolution*. Proceedings of the VLDB Endowment, 6(6), 349-360.

Wormald, N. C. (1999). *Models of random regular graphs*. London Mathematical Society Lecture Note Series, 239-298.

Yoo, S. (2019). *SPGS: Smart parking space guidance system based on user preferences in a parking lot*. Journal of the Korea Industrial Information Systems Research, 24(4), 29-36.

유 기 현 (Kihyun Yoo)

- 군산대학교 컴퓨터정보공학과 학사
- 군산대학교 컴퓨터정보공학과 석사
- (현재)군산대학교 컴퓨터정보공학과 박사과정
- 관심분야 : 데이터베이스, 공간정보시스템, 웹 서비스 컴퓨팅,

이 동 기 (Donggi Lee)

- (현재)군산대학교 컴퓨터정보통신공학부 학사과정
- 관심분야 : 데이터베이스, 웹 서비스 컴퓨팅

이 창 우 (Chang Woo Lee)

- 경일대학교 컴퓨터공학과 학사
- 경북대학교 컴퓨터공학과 석사
- 경북대학교 컴퓨터공학과 박사
- (현재)군산대학교 컴퓨터정보 통신공학부 교수
- 관심분야 : 인공지능, 딥러닝, 컴퓨터비젼

남 광 우 (Kwang Woo Nam)

- 충북대학교 컴퓨터과학과 학사
- 충북대학교 전자계산학과 석사
- 충북대학교 전자계산학과 박사
- 한국전자통신연구원 선임연구원
- (현재)군산대학교 컴퓨터정보 통신공학부 교수
- 관심분야 : 데이터베이스, 인공지능, 공간정보 시스템