http://www.jsebs.org
pISSN 2288-3908
eISSN 2765-3846

The Journal of Society for e-Business Studies
Vol.29, No.3, August 2024, pp.177-198
https://doi.org/10.7838/jsebs.2024.29.3.177

뉴럴 임베딩 기술을 활용한 정형 데이터의 자동 조인 융합 기법

A Join-based Data Integration Technique for Structured Data Using Neural Embedding

최윤석(Yoonseok Choi)*, 김한준(Han-joon Kim)**

초 록

우리는 둘 이상의 테이블을 하나의 테이블로 조인(Join) 융합하여 새로운 유의미한 정보를 생성할 수 있다. 하지만 데이터 레이크(Data Lake)와 같은 대규모 데이터셋 공간 내에서 조인 가능한 테이블을 찾는 것은 매우 노동 집약적이고 시간이 많이 걸리는 작업이다. 본 논문은 컬럼명 임베딩과 개체명 인식 기법을 통합하여 자동으로 조인 융합 가능한 테이블을 찾아내는 새로운 기법인 CNE-Join을 제안한다. 또한 CNE-Join 기법의 결과를 활용해 조인 융합 가능한 3개 이상의 테이블 데이터 조합들을 찾아내고, 이들을 조인 테이블과 조인에 사용된 소스 테이블간 코사인 유사도 값을 기준으로 랭킹하는 새로운 기법인 N-Join-Pair를 제안한다. 우리는 Kaggle에서 수집한 다수의 테이블 데이터셋을 사용한 실험을 통해 제안 기법의 우수성을 보인다.

ABSTRACT

We can create new meaningful information by joining two or more tables into one table. However, finding joinable tables within a large dataset space such as a data lake is a very labor-intensive and time-consuming task. This paper proposes CNE-Join, a new technique that integrates column name embedding and named entity recognition techniques to automatically find tables that can be joined. Furthermore, we propose a new technique called N-Join-Pair that leverages the results of the CNE-Join method to discover combinations of three or more joinable table data, and then ranks these combinations based on the cosine similarity values between the join table and the source tables used in the join. We demonstrate the superiority of the proposed technique through experiments using multiple table datasets collected from Kaggle.

키워드 : 딥러닝, 데이터 융합, 조인, 임베딩, 테이블 데이터
Deep Learning, Data Integration, Join, Embedding, Table Data

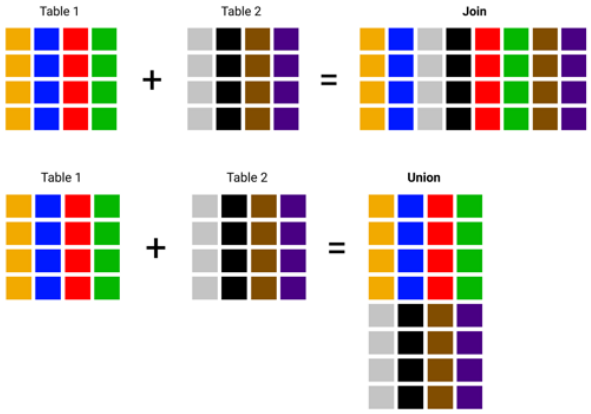
* First Author, Master's Degree, Department of Electrical and Computer Engineering, University of Seoul (choiys8819@naver.com)
** Corresponding Author, Professor, Department of Electrical and Computer Engineering, University of Seoul (khj@uos.ac.kr)
Received: 2024-07-08, Review completed: 2024-08-20, Accepted: 2024-08-28

1. 서 론

1.1 연구의 배경

최근 인공지능(AI) 시대가 도래하면서 AI의 성능을 높이기 위해 AI의 학습데이터 품질을 높이는 방법에 대한 관심이 커지고 있다[1, 2, 5, 10]. 그리하여 학습데이터의 품질을 높일 수 있는 방법중 하나인 데이터 융합에 대한 관심 또한 커지고 있다[11, 13, 19, 20]. 데이터 융합이란 서로 다른 데이터간 적절한 연계 및 가공 연산을 통해 새로운 정보를 생성하는 과정이며, 데이터 융합을 통해 유의미한 새로운 정보를 생성함으로써 데이터 품질을 높이는 것이 가능하다. 데이터 융합은 융합에 사용하는 데이터의 종류에 따라 많은 종류가 있지만, 본 연구에서는 테이블 데이터를 사용하는 테이블 데이터 융합에 초점을 두고 연구를 진행한다. 테이블 데이터 융합 방법은 크게 2가지가 있다. 하나는 2개 이상의 테이블을 수평으로 결합하여 하나의 테이블을 만드는 조인(Join) 융합이며, 다른 하나는 2개 이상의 테이블을 수직으로 결합하여 하나의 테이블을 만드는 유니온(Union) 융합이다. 이에 대한 예시가 <Figure 1>에 제시되어 있다. 우리는 융합하려는 테이블이 대부분 비슷한 컬럼을 가져야 융합하여 유의미한 정보를 얻을 수 있는 유니온 융합보다, 융합하려는 테이블이 하나의 공통된 조인 컬럼만 있어도 융합하여 유의미한 정보를 얻을 수 있는 조인 융합이 더 유용하다고 판단하였다. 이러한 이유로 본 연구에서는 테이블 데이터 융합 방법 중 조인 융합에만 초점을 두고 연구를 진행한다.

앞서 설명하였듯 우리는 2개 이상의 테이블을 특정 컬럼을 기준으로 조인 융합하여 조인하기 이전에는 알 수 없었던 유의미한 정보를 얻을 수 있다. <Table 1>은 그 예시이다. <Table 1>에 보이는 3개의 테이블 중에서 상단 좌측 테이블은 '대학명'과 '위치' 컬럼, 상단 우측 테이블은 '음식점'과 '위치' 컬럼으로 구성되어 있다. 우리는 이 2개의 테이블을 '위치' 컬럼을 기준으로 조인 융합하여 하단의 조인 융합된 테이블을 얻을 수 있다. 하단의 테이블을 통해 우리는 조인 융합하기 이전에는 알 수 없었던 각 대학교 주변에 있는 음식점 정보를 알아낼 수



〈Figure 1〉 Table Data Fusion Method

〈Table 1〉 Example of joinable tables

University Name	Location
UOS	Dongdaemun-gu
SNU	Gwanak-gu
KU	Seongbuk-gu

Restaurant	Location
Steak&Cheese	Dongdaemun-gu
Bibimbap House	Dongdaemun-gu
Pasta King	Gwanak-gu
Pizza Mall	Seongbuk-gu

University Name	Restaurant	Location
UOS	Steak&Cheese	Dongdaemun-gu
UOS	Bibimbap House	Dongdaemun-gu
SNU	Pasta King	Gwanak-gu
KU	Pizza Mall	Seongbuk-gu

있다. <Table 1>의 예시에서 우리는 주어진 2개의 테이블에 대해 조인 가능한지를 쉽게 판단할 수 있었다. 하지만, 테이블의 개수가 수백 또는 수천 개 이상으로 많아진다면, 사람이 수작업으로 조인 가능한 테이블을 찾아내는 것은 매우 어려운 작업이 될 것이다. 따라서 방대한 규모의 테이블 데이터가 주어질 때, 조인 융합이 가능한 테이블을 자동으로 찾아내는 기법이 필요하다. 본 논문은 다수의 테이블 데이터가 주어질 때, 조인 가능한 2개 이상의 테이블을 자동으로 찾아내는 기법을 제안한다.

1.2 논문의 구성

본 논문은 다음과 같은 구성으로 진행된다. 2장에서 자동 조인 융합과 관련된 선행 연구들을 소개하고, 3장에서는 2개의 테이블에 대한 자동 조인 융합 기법인 CNE-Join 기법과 이를 확장하여 3개 이상의 테이블에 대한 자동 조인 융합 및 랭킹을 지원하는 N-Join-Pair 기법에 대해 서술한다. 4장에서는 실험에 사용한 데이터셋, 실험 방법 및 실험에 사용한 평가 지표에 관해 설명하고, 이를 통해 얻은 실험 결과로 제안 기법의 우수성을 평가하고 제안 기법의 가치와 고려 사항을 고찰한다. 마지막으로 5장에서 본 논문의 전체적인 연구 내용을 정리하고 향후 연구 내용에 관해 다루며 마무리한다.

2. 관련 연구

2.1 컬럼명을 통해 조인 가능성을 판단하는 기법

컬럼명을 통해 조인 가능성을 판단하는 기법에는 한 컬럼명과 다른 컬럼명이 부분적으로 일치하는지 N-gram 알고리즘을 사용하여 확인하는 기법[3]이 있다. N-gram 알고리즘은 문자열에서 N개의 연속된 요소를 추출하는 방법이다. 예를 들어, 한 컬럼명이 '대학명'이고 다른 컬럼명이 '대학명칭'일때 N의 값이 2이면 각각의 N-gram은 {대학, 학명}, {대학, 학명, 명칭}이다. 기법은 두 N-gram에 대해 Jaccard 유사도 계산을 진행하여 그 값이 임계값 이상이면, 두 컬럼을 조인 컬럼으로 하여 두 컬럼이 속해있는 테이블이 조인 가능하다고 판단한다.

〈Table 2〉 Differences between CNE-Join and comparision methods

Method	Compare column names	Compare column values	Embedding	Named Entity Recognition
CNE-Join	O	O	O	O
Equi-Join	X	O	X	X
Jaccard-Join	X	O	X	X
CE-Join	O	X	O	X

컬럼명을 통해 조인 가능성을 판단하는 다른 기법으로는, 두 컬럼명에 대한 임베딩 벡터를 생성하고 생성된 벡터에 대해 코사인(cosine) 유사도 계산을 통해 두 컬럼명이 의미적으로 유사한지 판단하는 기법[8, 9, 16]이 있다. 기법은 두 컬럼명 임베딩 벡터에 대해 계산한 코사인 유사도 값이 임계값 이상이면 두 컬럼명이 의미적으로 유사하다고 판단하고 두 컬럼을 조인 컬럼으로 하여 두 컬럼이 속해 있는 두 테이블이 조인 가능하다고 판단한다. 해당 기법에서 컬럼명 임베딩 벡터를 생성할 때, Word2Vec[14]나 FastText[4]과 같은 단어 임베딩 모델을 사용한다.

2.2 컬럼값을 통해 조인 가능성을 판단하는 기법

컬럼값을 통해 조인 가능성을 판단하는 기법에는 우선 특정 컬럼의 값들이 다른 컬럼의 값들과 일치하는지 확인하는 기법[21]이 있다. 해당 기법은 두 컬럼값이 일치하면 두 컬럼을 조인 컬럼으로 두 테이블이 조인 가능하다고 판단한다. 다른 기법으로는 한 컬럼의 컬럼값 집합과 다른 컬럼의 컬럼값 집합간의 유사도를 계산하는 기법[7]이 있다. 집합간의 유사도를 계산할 때에는 자카드 유사도 식을 사용하며, 집합간 자카드 유사도를 계산해 그 값이 임계값 이상이면 기법은 두 컬럼을 조인 컬럼으로 두 테이블이 조인 가능하다고 판단한다.

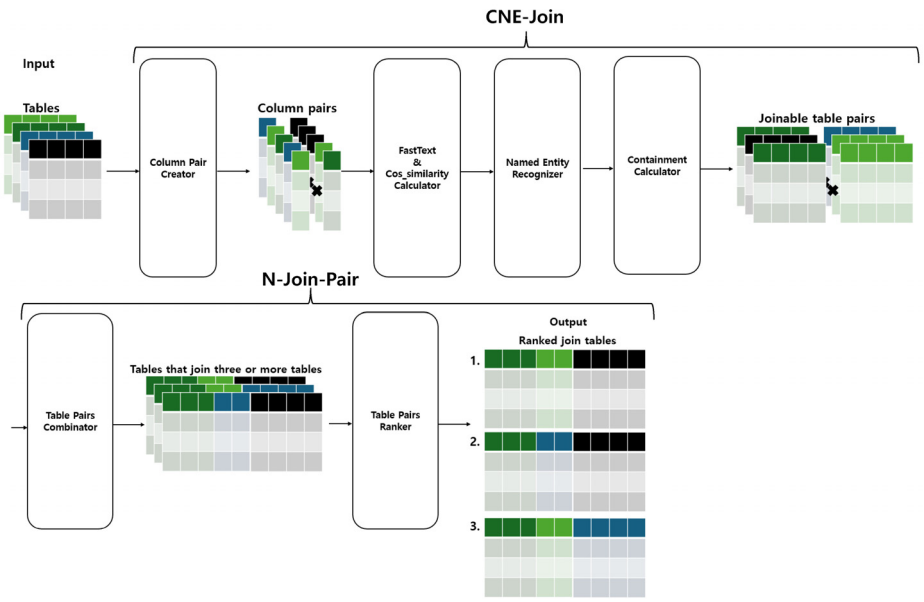
2.3 CNE-Join과 관련 기법의 비교

CNE-Join 기법과 상대적으로 비교할 기법은 Equi-Join[21], Jaccard-Join[7], CE-Join[17]이다. <Table 2>는 CNE-Join과 비교 기법의 차이점을 보여준다. Equi-Join은 한 컬럼에 포함된 값이 다른 컬럼의 포함된 값과 완전히 겹치면, 두 컬럼을 기준으로 테이블이 조인 가능하다고 판단하는 기법이다. Jaccard-Join은 두 컬럼의 Jaccard 유사도 값이 임계값 이상이면, 두 컬럼을 기준으로 테이블이 조인 가능하다고 판단하는 기법이다. CE-Join은 두 컬럼명에 대한 임베딩 벡터를 생성하여 코사인 유사도 계산을 하여 그 값이 임계값 이상이면 두 컬럼을 기준으로 테이블이 조인 가능하다고 판단하는 기법이다.

3. 제안 기법

3.1 제안 기법의 아키텍처

<Figure 2>는 제안 기법의 아키텍처이다. 우선 사용자가 입력한 다수의 테이블 데이터가 CNE-Join의 입력으로 들어간다. CNE-Join은 Column Pair Creator를 통해 입력받은 테이블에



〈Figure 2〉 Architecture of the Proposed Method

서 컬럼을 추출하고, 추출한 컬럼을 조합해 컬럼쌍을 만든다. 만들어진 컬럼쌍은 CNE-Join의 FastText, Named Entity Recognizer, Containment Calculator를 거치게 되며, 이 과정에서 조인 컬럼으로 사용하기에 부적합한 컬럼쌍은 걸러진다. CNE-Join은 걸러지고 남은 컬럼쌍을 조인 컬럼으로 하여 조인 가능한 테이블쌍(Table Pairs)을 만들고 이를 출력한다. CNE-Join의 출력인 조인 가능한 테이블쌍은 N-Join-Pair의 입력으로 들어간다. N-Join-Pair는 Table Pairs Combinator를 통해 입력받은 조인 가능한 테이블쌍을 조합해 3개 이상의 조인 가능한 테이블 조합을 새로 만든다. 이렇게 만들어진 테이블 조합은 N-Join-Pair의 Table Pairs Ranker를 거쳐 랭킹이 매겨지고, N-Join-Pair는 출력으로 랭킹이 매겨진 테이블 조합을 사용자에게 제공한다. CNE-Join과 N-Join-Pair에 대한 자세한 설명은 2절과 3절에서 진행한다.

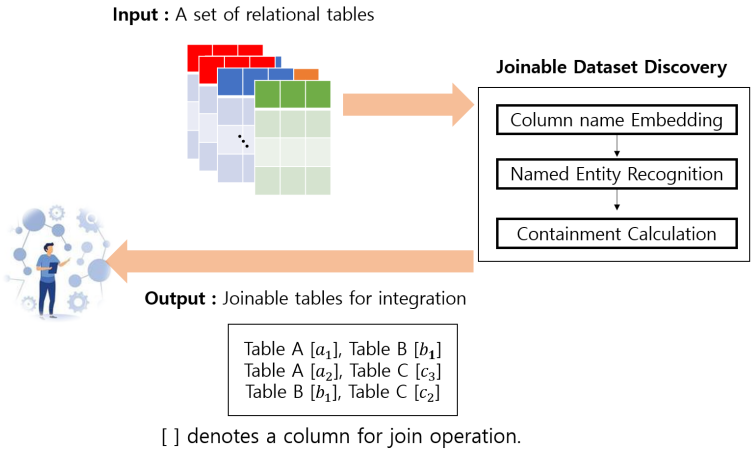
3.2 CNE-Join

<Figure 3>은 CNE-Join 기법의 수행과정을 보여준다. CNE-Join은 다수의 테이블을 입력받아 각 테이블에 대해 컬럼명 임베딩(Column Name Embedding), 개체명 인식(Named Entity Recognition) 및 포괄성 계산(Containment Calculation)을 수행하고, 출력으로 사용자에게 조인 가능한 테이블쌍과 조인의 기준이 되는 컬럼쌍(Column Pairs)을 제공한다.

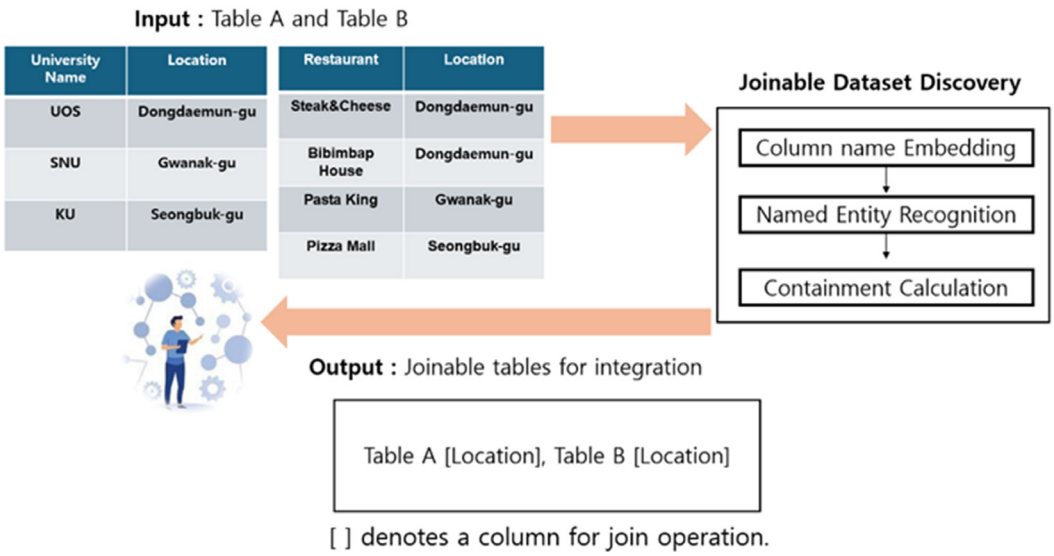
<Figure 4>는 CNE-Join의 수행과정 예시이다. CNE-Join에 입력으로 Table A와 Table B가 들어오면, CNE-Join은 출력으로 Table A와 Table B가 각각의 위치 컬럼을 조인의 기준이 되는 컬럼으로 하여 조인이 가능하다는 정보를 제공한다. <Figure 5>는 CNE-Join의 수행과정을 의사코드로 나타낸 것이다.

<Figure 6>은 CNE-Join에서 사용하는 2개

의 임베딩 모델의 입출력을 그림으로 정리한 것이다. CNE-Join 기법은 컬럼명 임베딩 단계에서 사전 학습된 FastText 임베딩 모델을 사용한다. 본 연구에서 사용한 FastText는 특정 단어를 입력받으면 그 단어에 해당하는 300차원의 벡터를 출력하는 모델로, Wikipedia 문서를 학습 데이터로 사전 학습하여 만들어진 것이다. 컬럼명 임베딩 단계에서 FastText는 컬럼명을 입력으로 받으면 그에 해당하는 300차원의 컬럼명 임베딩 벡터를 출력한다. 개체명 인식 단계에서는 BERT[6] 임베딩 모델을 Kaggle의 NER 데이터셋[16]으로 학습하여 구



〈Figure 3〉 CNE-Join Execution Process



〈Figure 4〉 Example of CNE-Join Execution Process

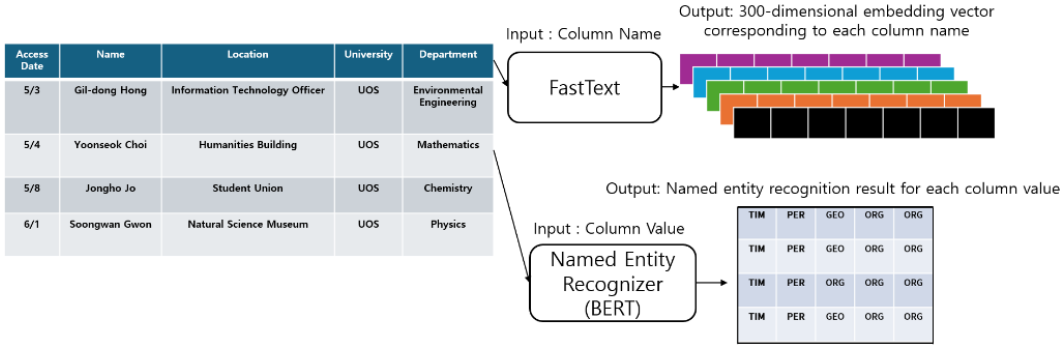
축한 개체명 인식기[15]를 사용한다. 개체명 인식기는 컬럼값을 입력으로 받으면, 각 컬럼값의 개체명 인식 결과를 출력한다. 개체명 인식기는 문장 수준의 데이터를 입력받아야 하므로, 대부분의 값이 단어 수준인 컬럼값에 대해 우리는 '[컬럼명] + [is] + [컬럼값]' 형식의 문장을 인위적으로 구성하여 개체명 인식기에 입력한다. <Figure 7>은 단어 수준의 컬럼값에 대해 문장을 인위적으로 구성하는 예시이다. 예를 들어, 컬럼명이 "Genre"이고 컬럼값이 "Action"인 경우에는 "Genre is Action"과 같이 문장을 인위적으로 구성하여 개체명 인식기에 입력한다. 이렇듯 문장을 구성하여 개체명 인식기에 입력하므로, 우리는 문맥을 고려하는 임베딩이 가능한 BERT 임베딩 모델을 사용하였다.

Algorithm 1: CNE-Join

Data: Table $\{T_1, T_2, \ldots, T_n\}$
Result: Pairs of joinable tables and their respective join columns

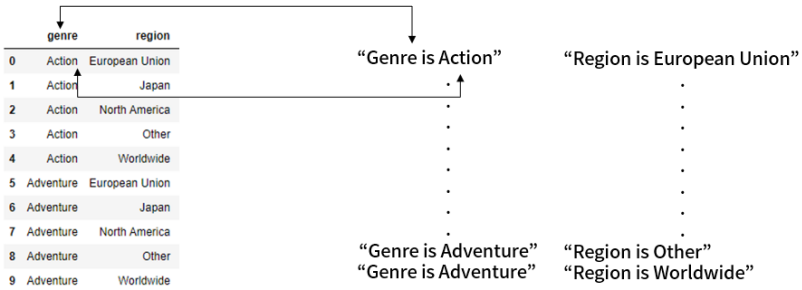
```
columns ← ExtractColumns(Data);
columnPairs ← CreateColumnPairs(columns);
foreach columnPair ∈ columnPairs do
    vectorPair ← ColumnNameEmbedding(columnPair);
    cosineSim ← CosineSimilarity(vectorPair);
    if cosineSim > cosineThreshold then
        NER1 ← NamedEntityRecognition(columnPair.column1);
        NER2 ← NamedEntityRecognition(columnPair.column2);
        repCategory1 ← MostFrequent(NER1)
        repCategory2 ← MostFrequent(NER2)
        if repCategory1 = repCategory2 then
            containment ←
             ContainmentCalculate(columnPair.column1, columnPair.column2);
             if containment > containmentThreshold then
                joinablePairs ← (columnPair.column1 and original
                 table to which columnPair.column1 belonged,
                 columnPair.column2 and original table to which
                 columnPair.column2 belonged);
return joinablePairs;
```

〈Figure 5〉 Pseudocode of CNE-Join Execution Process

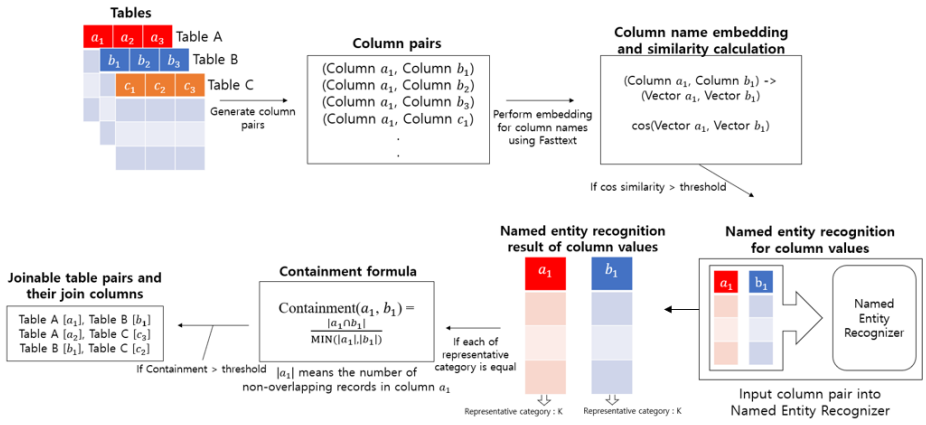


〈Figure 6〉 Input and Output of the Embedding Model Used in CNE-Join

<Figure 8>은 CNE-Join의 구체적인 수행 과정이다. CNE-Join은 입력으로 주어진 테이블에 대해, 우선 서로 다른 테이블에 포함된 컬럼 2개를 묶어 컬럼쌍으로 만드는 작업을 진행한다. 예를 들어, <Figure 8>과 같이 테이블 A,B,C 총 3개의 테이블이 주어질 때, 각 테이블이 3개의 컬럼을 가지고 있으므로, 총 27개의 컬럼쌍이 생성된다. CNE-Join은 만들어진 모든 컬럼쌍에 대해 컬럼명 임베딩을 진행하여, 하나의 컬럼쌍마다 하나의 벡터쌍을 생성하고, 생성한 벡터쌍에 대해 코사인 유사도 계산을 진행한다. CNE-Join은 벡터쌍의 코사인 유사도 계산 결과가 임계값 이상이면, 그 벡터쌍에 해당하는 컬럼쌍에 대해 개체명 인식을 진행한다. 코사인 유사도 임계값은 0.3정도의 낮은 값으로 정한다. 개체명 인식 단계에서는 컬럼쌍의 컬럼값들에 대해 개체명 인식기를 사용해 고유명사 수준의 개체명을 추출한다. 본 연구에서 사용하는 개체명은 기관(ORG), 인물(PER), 지리(GEO)등을 포함한 8개의 카테고리로 분류된다. CNE-Join은 각 컬럼에서 가장 많이 출현한 개체명 카테고리를 그 컬럼의 대표 카테고리(Representative Category)로 지정한다. 예를 들어, <Figure 8>에서 a_1컬럼의 3개



〈Figure 7〉 An Example of Artificially Creating Sentences based on Word-level Column Values



〈Figure 8〉 Specific Execution Process of CNE-Join

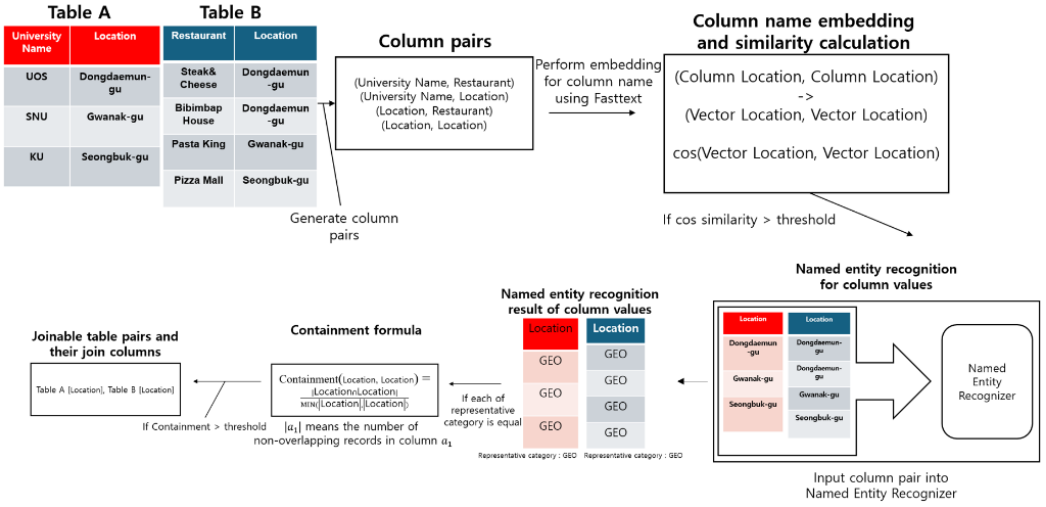
컬럼값에 대한 개체명 카테고리가 모두 개체명 K였다면, a_1컬럼의 대표 개체명 카테고리는 K이다. 유사하게 b_1컬럼의 3개 컬럼값에 대한 개체명 카테고리가 모두 개체명 K였다면 b_1컬럼의 대표 개체명 카테고리는 K가 된다. 하지만 실제로 사용하는 테이블 데이터의 경우 컬럼값의 개수가 수천 개 또는 수십만 개까지 달하는 경우가 많다. 따라서 개체명 인식에 걸리는 시간을 단축하기 위해 각 컬럼값의 20%만 샘플링하고 샘플링된 컬럼값에 대해 개체명 인식을 수행하여 각 컬럼의 대표 개체명을 얻는 샘플링 방법도 사용한다. CNE-Join은 컬럼쌍에 속한 두 컬럼의 대표 개체명 카테고리의 일치 여부를 확인하고, 일치하면 다음 단계로 넘어간다. 다음 단계에서 CNE-Join은 컬럼쌍에 대해 포괄성 계산을 진행한다. 포괄성 계산 단계에서는 컬럼쌍에 대해 포괄성 계산을 진행하여 그 값이 임계값 이상인 컬럼쌍을 가려낸다. 포괄성 계산에서 임계값은 0.6이상의 높은 값으로 정한다. 포괄성(containment) 계산 수식은 (1)과 같다. (1)에서 $|a_1|$은 컬럼 $|a_1|$의 유니크한 컬럼값의 개수를 의미한다.

$$containment(a_1, b_1) = \frac{|a_1 \cap b_1|}{MIN(|a_1|, |b_1|)} \quad (1)$$

<Figure 9>는 CNE-Join의 구체적 수행과정 예시이다. 입력으로 두 테이블 A와 B가 들어오면, CNE-Join은 앞서 <Figure 8>에서 설명한 수행과정을 거쳐 최종적으로 Table A와 Table B가 각각의 '위치' 컬럼을 통해 조인 가능하다고 출력한다.

3.3 N-Join-Pair

<Figure 10>은 N-Join-Pair의 수행과정을 나타낸 의사코드이다. N-Join-Pair는 사용자로부터 다수의 테이블을 입력받으면, 우선 CNE-Join을 활용해 조인 가능한 테이블쌍을 찾아내고, 찾아낸 조인 가능한 테이블쌍을 조



〈Figure 9〉 Example of a Specific Execution Process of CNE-Join

합해 3개 이상의 조인 가능한 테이블 조합을 만들어낸다. 조인 가능한 테이블 조합을 만드는 작업은 <Figure 11>과 같이 진행된다. <Figure 11>은 <Figure 2>의 Table Pairs Combinator 모듈에서 진행된다. <Figure 11>을 보면 첫 번째 줄의 'Table A [], Table B []'과 세 번째 줄에 'Table B [], Table C []'가 Table B []을 공통적으로 가지는 것을 확인할 수 있다. Table Pairs Combinator는 공통된 테이블인 Table B []을 매개로 'Table A [], Table B [], Table C []' 테이블 조합을 만들어낸다. Table Pairs Combinator는 만들어진 테이블 조합에 대해 조인을 진행하여 조인 융합 테이블을 만든다. 이후 N-Join-Pair는 조인 융합 테이블의 랭킹을 매기기 위해, 조인 융합 테이블과 조인에 사용된 소스 테이블 간에 코사인 유사도를 계산하여 평균을 낸다. <Figure 12>는 조인 융합 테이블과 소스 테이블 간 코사인 유사도 계산 방법을 나타낸 것이다. <Figure 12>의 코사인 유사도 계산은 <Figure 2>의 Table Pairs

Algorithm 1: N-Join-Pair

Data: Input table data $\{T_1, T_2, \ldots, T_n\}$
Result: Ranked list of joined tables

Step 1: Find joinable table pairs using CNE-Join
joinablePairs ← `CNEJoin`($\{T_1, T_2, \ldots, T_n\}$);

Step 2: Explore 3 or more join tables
joinedTables ← `ExploreJoinTables`(*joinablePairs*);

# Step 3: Compute average cosine similarity
for *each joinedTable in joinedTables* **do**
  avgCosineSimilarity ← `CalculateCosineSimilarity`(*joinedTable*);
  joinedTable.similarity ← avgCosineSimilarity;
end

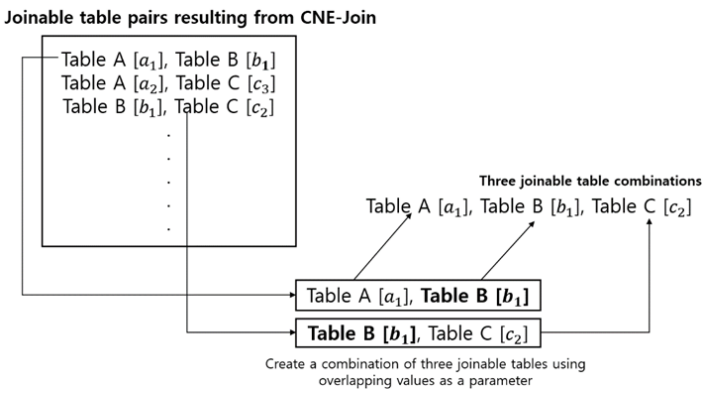
Step 4: Rank tables based on average cosine similarity
rankedTables ← `RankTables`(*joinedTables, joinedTable.similarity*);

Output the results to the user
return rankedTables;

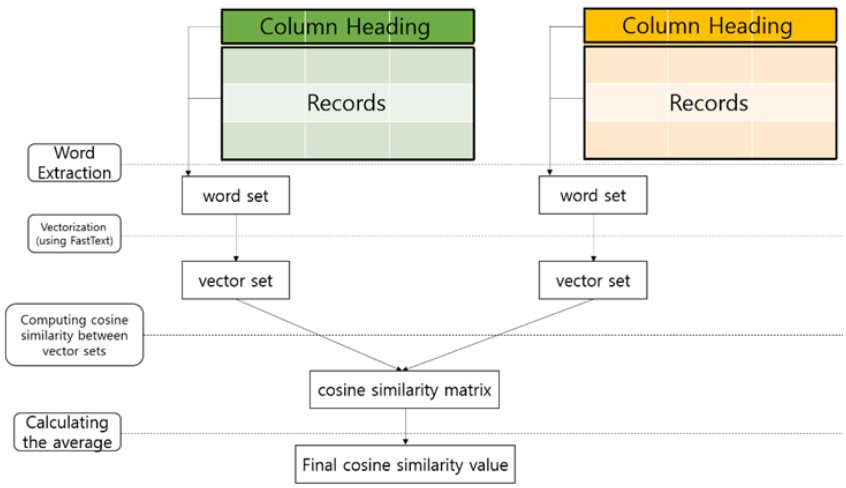
Function `CNEJoin`(*tables*)
  # Find joinable table pairs using CNE-Join
  return joinablePairs;
Function `ExploreJoinTables`(*joinablePairs*)
  # Explore joins of 3 or more tables
  joinedTables ← Joined tables created using overlapping values among JoinablePairs values;
  return joinedTables;
Function `CalculateCosineSimilarity`(*joinedTable*)
  # Compute average cosine similarity between the joined table and the source tables used for joining
  sourceTable$_1$, $sourceTable_2$, ..., $sourceTable_n$← Retrieve the source tables used to create the joinedTable from the DB ;
  avgCosineSimilarity ← average(cos(joinedTable, sourceTable$_1$), $cos(joinedTable, sourceTable_2)$, ..., $cos(joinedTable, sourceTable_n)$)
  return avgCosineSimilarity;
Function `RankTables`(*joinedTables, joinedTable.similarity*)
  # Rank the joined tables based on the average cosine similarity values
  rankedTables ← Sort joinedTables in descending order by highest joinedTable.similarity value
  return rankedTables;

〈Figure 10〉 Pseudocode for N-Join-Pair Execution Process

Ranker 모듈에서 진행된다. Table Pairs Ranker는 우선 <Figure 12>와 같이 조인 융합 테이블의 모든 컬럼 헤딩과 레코드에 속한 단어들을 추출한다. 추출한 모든 단어들은 FastText 임베딩 모델을 통해 벡터로 변환되어 벡터 집합을 구성한다. 본 실험에서 FastText 임베딩 모델을 사용한 이유는 기존 Word2Vec와 같은 단어 임베딩 모델이 해결하지 못한 Out of Vocabulary 문제를 서브 워드를 고려하는 학습을 통해 해결한 단어 임베딩 모델이기 때문이다. Table Pairs Ranker는 소스 테이블에서도 마찬가지 방법으로 벡터 집합을 얻어낸다. 이후 조인 테이블의 벡터 집합에 포함된 모든 벡터들과 각 소스 테이블의 벡터 집합에 포함된 모든 벡터들간 cosine 유사도 계산을 진행해 코사인 유사도 행렬을 얻는다. 이후 코사인 유사도 행렬의 평균을 계산해 최종 코사인 유사도 값을 구한다. Table Pairs Ranker는 최종 코



〈Figure 11〉 A Method to Create Combinations of Three Joinable Tables

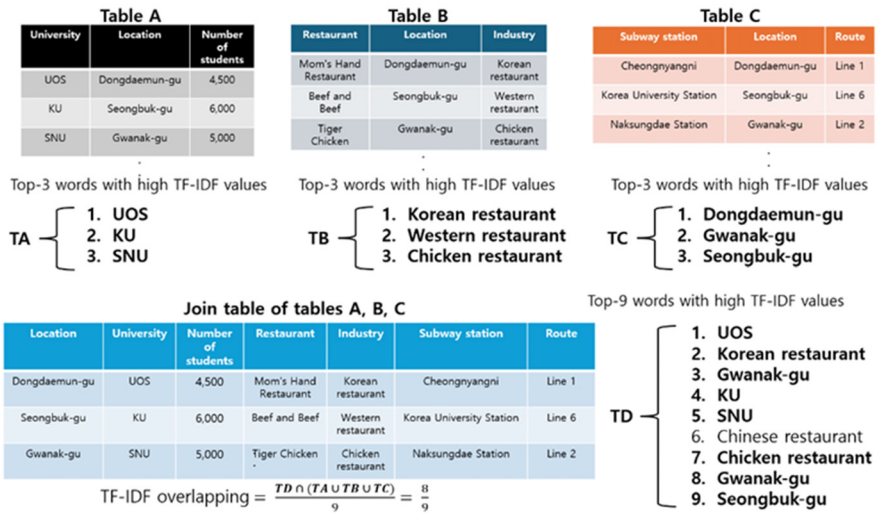


〈Figure 12〉 A method to Calculate Cosine Similarity between Join Table and Source Table

사인 유사도 값을 기준으로 조인 테이블들을 랭킹하고, 이를 N-Join-Pair의 최종 출력으로 제공한다.

<Figure 13>은 N-Join-Pair의 평가 기법인 TF-IDF 중첩(overlapping)의 예시이다. <Figure 13>과 같이 테이블 A, B, C와 테이블 A, B, C를 조인한 조인 융합 테이블이 있다면, 우선 Table A에서 TF-IDF[18] 값이 큰 N개의 단어를 추출한다. <Figure 13>의 예시에서는 N의 값을 3으로 지정하였고, 각 테이블에 등장하는 각 단어의 TF 값은 각 테이블에 등장한 단어의 빈도를 해당 문서의 총 단어 수로 나누어 계산하고, 각 테이블의 등장하는 각 단어의 IDF 값은 총 테이블의 개수를 단어가 등장한 테이블의 개수로 나누고 전체에 로그를 씌워 계산한다. 최종적으로 TF 값과 IDF 값을 곱해 각 단어의 TF-IDF 값을 구한다. 테이블 B와 C에서도 마찬가지로 TF-IDF 값이 큰 N개의 단어를 추출한다. 이후 테이블 A, B, C를 조인 융합한 테이블에서 TF-IDF 값이 큰 3N개의 단어를 추출한다. 이 3N개의 단어와 테이블 A, B, C 각각의 TF-IDF 값이 큰 N 개의 단어 중 겹치는 단어의 개수를 3N으로 나눈 것이 테이블 A, B, C를 조인한 조인 테이블의 조인 품질 척도인 TF-IDF 중첩이다. <Figure 13>의 경우 테이블 A, B, C에서 TF-IDF 값이 큰 3개의 단어 중, 테이블 A, B, C의 조인 테이블의 TF-IDF 값이 큰 9개의 단어와 겹치는 것의 개수는 8개로, TF-IDF 중첩 값은 $\frac{8}{9}$이 된다.

N-Join-Pair 기법은 사용자에게 출력으로 랭킹이 매겨진 조인 융합 테이블을 제공하고, 랭킹을 매기는 기준은 조인이 잘 된 정도이다. 우리는 조인 융합 테이블이 소스 테이블의 정보를 잘 반영하면 조인이 잘 되었다고 판단하기로 하였다. 따라서 TF-IDF 중첩을 활용해 각 소스 테이블의 중요한 정보인 TF-IDF 값이 높은 단어들을 추출하고 이를 조인 융합 테이블의 TF-IDF 값이 높은 단어들과 겹치는지 비교하는 방식으로 N-Join-Pair의 성능평가를 진행한다.



〈Figure 13〉 Example of TF-IDF Overlapping

4. 실험 및 결과

4.1 CNE-Join 실험 및 결과

제안 기법의 우수성을 보이기 위해, 우리는 Kaggle에서 1,000개의 테이블 데이터셋을 수집하였다. 수집한 테이블 데이터셋은 영어로 구성되어 있으며, 300MB 미만의 크기를 가지는 테이블 데이터셋으로 수집하였다. 벤치마크 데이터셋을 사용하여 실험을 진행하는 것도 가능하지만, 실제 사용자들이 많이 사용하는 Kaggle의 인기 데이터셋을 사용하는 것이 의미가 있을 것으로 판단되어 Kaggle의 테이블 데이터셋을 사용하였다. 성능 평가 지표는 Dong et al.[7]의 논문을 참고하여 Precision(2)과 Recall(3)의 조화평균인 F1 score를 사용하였다.

Precision의 분모는 각 기법이 조인 가능하다고 판단한 컬럼쌍의 수이며, 이 중 실제로 조인 가능한 컬럼쌍의 수가 Precision의 분자에 들어간다. 데이터셋의 모든 테이블을 보고 조인 가능한지를 라벨링하는 것은 노동집약적이고 시간 소모가 큰 작업이기에, 우리는 CNE-Join과 비교기법들에서 찾은 실제로 조인 가능한 컬럼쌍 집합에 대해 합집합 연산을 수행하고 이 집합의 개수를 Recall의 분모로 사용한다. Precision과 Recall 계산에 대한 예시는 <Figure 14>에 나와 있다. <Figure 14>는 100개의 테이블 데이터가 주어진 상황에서 각 기법이 찾아낸 조인 가능한 컬럼쌍의 수와 실제로 조인 가능한 컬럼쌍의 수, 그리고 이를 바탕으로 계산한 각 기법의 Precision, Recall, F1 score가 나와있는 그림이다. 이 경우에 CNE-Join의 Precision을 계산해보면 $\frac{40}{50}=0.800$이며, Recall은 $\frac{40}{60}=0.667$임을 알 수 있다. 같은 방법으로 다른 비교 기법에 대해서도 Precision과 Recall 값을 계산하여 각

$$Precision = \frac{\text{실제로 조인 가능한 컬럼쌍의 수}}{\text{조인 가능하다고 판단한 컬럼쌍의 수}} \tag{2}$$

$$Recall = \frac{\text{실제로 조인 가능한 컬럼쌍의 수}}{\text{실제로 조인 가능한 컬럼쌍의 합집합의 개수}} \tag{3}$$

• **Given 100 table data**

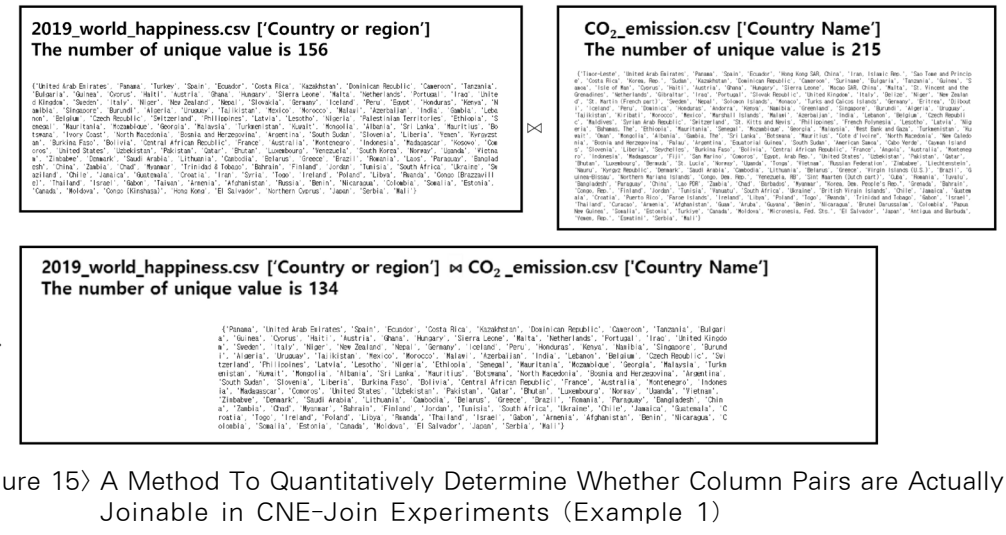
Method	The number of column pairs judged to be joinable	The number of column pairs that can actually be joined	Precision	Recall	F1-score
A	50	40	0.800	0.667	0.727
B	30	20	0.667	0.333	0.444
C	40	20	0.500	0.333	0.400
D	40	20	0.500	0.333	0.400

The number of column pairs resulting from the union operation is 60

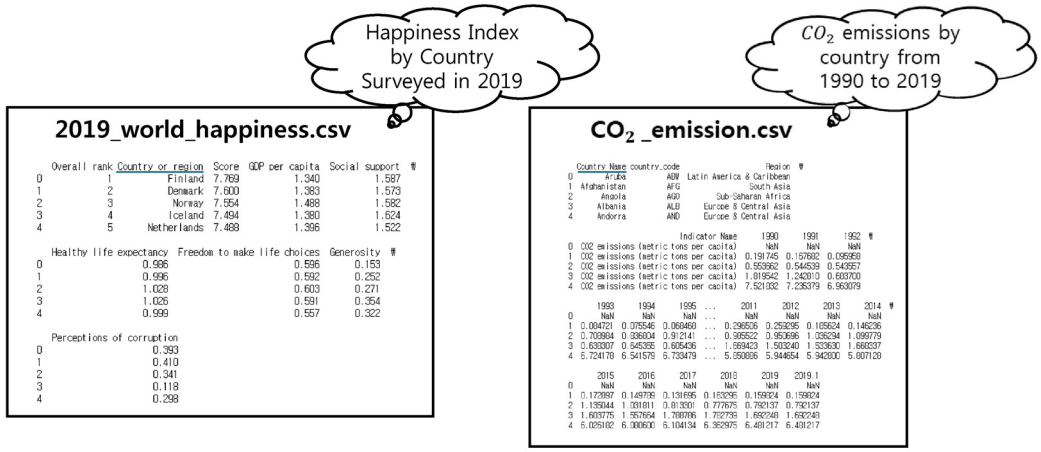
〈Figure 14〉 Example of Calculating the Performance Evaluation Index of CNE-Join

기법의 F1-score 값을 알아낼 수 있다.

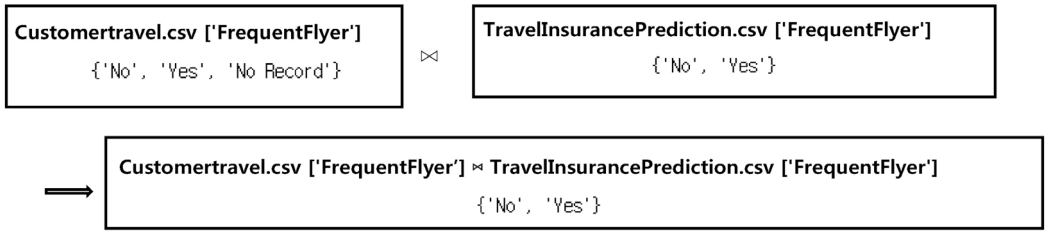
CNE-Join 실험 평가를 위해서는 각 기법이 찾아낸 조인 가능한 컬럼쌍에 대해 실제로 조인이 가능한지 판단해야 한다. 컬럼쌍이 실제로 조인 가능한지 아닌지는 사람이 직접 판단해야 하는 문제이므로, 우리는 정량적, 정성적 근거를 들어 컬럼쌍이 실제로 조인 가능한지 판단한다. <Figure 15>는 컬럼쌍이 실제로 조인 가능한지 정량적으로 판단하는 방법이다. <Figure 15>의 컬럼쌍의 첫 번째 컬럼은 '2019_world_happiness' 테이블의 'Country or region' 컬럼이고, 두 번째 컬럼은 'CO_2_emission' 테이블의 'Country Name' 컬럼이다. 우리는 두 컬럼에 대해 포괄성 계산을 진행한다. 포괄성 계산을 하면 $\frac{134}{156} = 0.859$로 포괄성 계산값이 0.6 이상임을 확인할 수 있다. 포괄성 계산값이 0.6 이상이므로 우리는 정량적으로 'Country or region' 컬럼과 'Country Name' 컬럼을 조인 컬럼으로 하여 '2019_world_happiness' 테이블과 'CO_2_emission' 테이블을 조인할 수 있다고 판단한다. 이후에 우리는 두 컬럼이 속한 각 테이블의 정보를 정성적으로 확인한다. <Figure 16>은 두 컬럼이 속한 각 테이블에 대한 정보이다. '2019_world_happiness' 테이블은 2019년에 조사한 국가별 행복지수에 관한 내용을 담고 있고, 'CO_2_emission' 테이블은 1990년부터 2019년까지의 국가별 배출량에 관한 내용을 담고 있다. 두 테이블을 국가명 정보를 가지는 'Country or region' 컬럼과 'Country Name' 컬럼으로 조인한다면 배출량과 국가별 행복지수간의 관계라는 유의미한 정보를 얻는 것이 가능할 것으로 보인다. 우리는 두 테이블의 정보를 정성적으로 확인하여 'Country or region' 컬럼과 'Country Name' 컬럼을 조인 컬럼으로 하여 '2019_world_happiness' 테이블과 'CO_2_emission' 테이블을 조인할 수 있다고 판단한다. 예시 1의 경우 정량적, 정성적으로 조인 가능하므로, 우리는 예시 1의 컬럼쌍에 대해 실제로 조인 가능하다고 판단한다.



〈Figure 15〉 A Method To Quantitatively Determine Whether Column Pairs are Actually Joinable in CNE-Join Experiments (Example 1)



〈Figure 16〉 Information about the '2019_world_happiness' and 'CO_2_emission' Tables (Example 1)

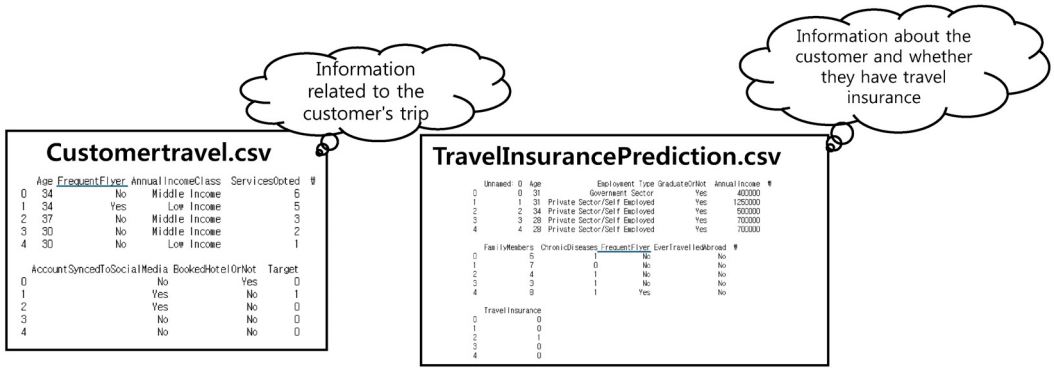


〈Figure 17〉 A Method to Quantitatively Determine Whether Column Pairs are Actually Joinable in CNE-Join Experiments (Example 2)

<Figure 17> 또한 컬럼쌍이 실제로 조인 가능한지 정량적으로 판단하는 방법이다. 컬럼쌍 중 첫 번째 컬럼은 'Customertravel' 테이블의 'FrequentFlyer' 컬럼이고, 두 번째 컬럼은 'TravelInsuarancePrediction' 테이블의 'FrequentFlyer' 컬럼이다. 두 컬럼에 대해 포괄성 계산을 하면 $\frac{2}{2}=1.000$으로 포괄성 계산값이 0.6 이상임을 확인할 수 있다. 포괄성 계산값이 0.6 이상이므로 우리는 정량적으로 'FrequentFlyer' 컬럼과 'FrequentFlyer' 컬럼을 조인 컬럼으로 하여 'Customertravel' 테이블과 'TravelInsuarancePrediction' 테이블을 조인할 수 있다고 판단한다. 이후에 우리는 두 컬럼이 속한 각 테이블의 정보를 정성적으로 확인한다. <Figure 18>은 두 컬럼이 속한 각 테이블에 대한 정보이다. 'Customertravel' 테이블은 여행을 하는 각 고객에 대한 정보를 담고 있고, 'TravelInsuarancePrediction' 테이블은 여행을 하는 각 고객에 대한 정보와 여행보험 가입 여부를 담고 있다. 두 테이블을 'Customertravel' 테이블의 'FrequentFlyer' 컬럼과 'TravelInsuarancePrediction' 테이블의 'FrequentFlyer' 컬럼으로 조인하면 서로 다른 고객의 정보가 조인되어 의미없는 조인 융합

테이블이 나오게 된다. 우리는 정성적으로 두 테이블을 확인하여 'FrequentFlyer' 컬럼과 'FrequentFlyer' 컬럼을 조인 컬럼으로 하여 'Customertravel' 테이블과 'TravelInsuarancePrediction' 테이블을 조인할 수 없다고 판단한다. 예시 2의 경우 정량적으로 조인이 가능하지만 정성적으로 조인 불가능하므로, 우리는 예시 2의 컬럼쌍에 대해 실제로 조인 가능하지 않다고 판단한다. 하지만 예시 2의 경우 CNE-Join은 해당 예시를 조인 가능하다고 잘못 판단한다. 이 부분은 향후 연구를 통해 개선할 예정이다.

<Table 3>은 CNE-Join과 비교기법의 F1 score 값을 비교한 표이다. 우리는 CNE-Join의 하이퍼 파라미터인 컬럼명 임베딩 임계값과 포괄성의 임계값을 조정하며 F1 score를 측정하고, 이를 비교기법의 F1 score와 비교하였다. 우리는 컬럼명 임베딩의 임계값을 낮추면 Recall 값이 오를 것이라 예상하였고 포괄성의 임계값을 높이면 Precision 값이 오를 것이라 예상하여, 낮은 컬럼명 임베딩 임계값과 높은 포괄성 임계값으로 실험을 진행하였다. <Table 3>을 살펴보면, 컬럼명 임베딩 임계값이 0.3이고 포괄성 임계값이 0.8일 때 CNE-Join의 F1 score가 가장 높은 것을 확인할 수 있다. 또한, 표 4.1.2를 보면 CNE-Join이 비교기법보다 높은 F1 score를 가지는 것을 확인할 수 있다.



〈Figure 18〉 Information about the 'Customertravel' and 'TravelInsuarancePrediction' Tables (Example 2)

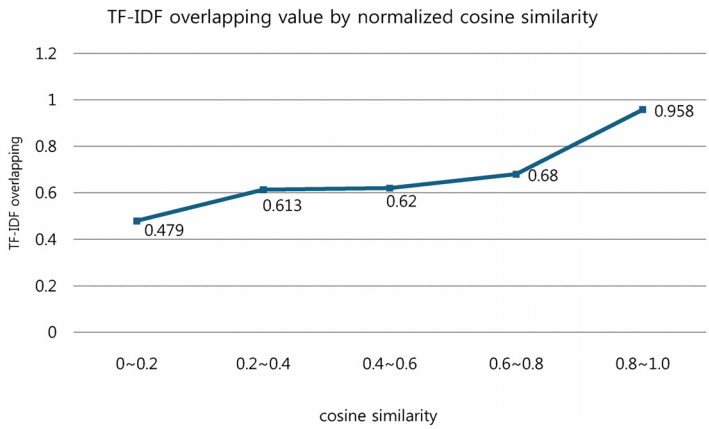
〈Table 3〉 Comparison of F1 Scores of CNE-Join and Comparison Methods

CNE-Join Hyperparameters		CNE -Join	Equi -Join	Jaccard-Join	CE -Join
Column name Embedding Threshold	Containment Threshold				
0.3	0.6	0.656	0.322	0.271	0.288
	0.7	0.729	0.322	0.271	0.288
	0.8	**0.767**	0.322	0.271	0.288
	0.9	0.751	0.326	0.276	0.291

4.2 N-Join-Pair 실험 및 결과

제안 기법의 우수성을 보이기 위해, 우리는 앞선 CNE-Join 실험에서 얻은 조인 가능한 테이블쌍을 조합해 3개 테이블에 대해 조인 가능한 테이블 조합을 만든다. 이때 만들어진 테이블 조합의 개수는 약 40,000개로 상당히 많은 수가 나왔다. 우리는 효율적인 N-Join-Pair 실험 평가를 위해 20개의 질의 테이블을 선정하여 40,000개의 테이블 조합 중 해당 질의 테이블을 소스 테이블로 가지는 테이블 조합에 대해서만 N-Join-Pair 실험을 진행하였다. 실제로도 사용자는 40,000개 테이블 조합 전체의 랭킹보다 특정 테이블과 조인 융합된 테이블의 랭킹을 알고 싶어하기에, 우리의 질의 테이블을 선정하는 실험 방식이 유의미할 것으로 보인다. 선정한 20개의 질의 테이블은 전부 다른 조인 컬럼을 가지며 조인 컬럼의 종류는 '국가명', '도시명', '질병명' 등으로 다양하다.

N-Join-Pair는 조인 융합 테이블과 소스 테이블간 코사인 유사도 값을 기준으로 조인 융합 테이블에 대해 랭킹을 매긴다. 이것이 유의미한지 보이기 위해 우리는 다음과 같은 실험을 진행하였다. 우선 0에서 1 사이를 0.2 간격으로 5개의 구간으로 나누고, 이후 조인 융합 테이블과 소스 테이블간의 코사인 유사도 값을 구하여 그 값에 해당하는 구간에 조인 융합 테이블을 매핑한다. 이후 각각에 구간에 매핑된 조인 융합 테이블에 대해 TF-IDF 중첩을 계산해 제안 기법의 랭킹이 유의미한지 평가한다. 실험 결과는 <Figure 19>와 같다. 그림의 그래프는 각 코사인 유사도 값 구간에 매핑된 조인 융합 테이블들에 대해 TF-IDF 중첩을 계산하고 구간마다 평균을 낸 것이다. 그래프의 x축은 코사인 유사도 값의 구간으로 5개가 존재하고, 그래프의 y축은 x축의 구간에 매핑된 조인 융합 테이블과 소스 테이블간 TF-IDF 중첩을 계산하고 그 평균을 낸 값이다. 그래프를 보면 코사인 유사도 값이 높아질수록 TF-IDF 중첩의 값이 커지는 것을 확인할 수 있다. 또한 우리는 각 조인 융합 테이블과 소스 테이블간 코사인 유사도 값을 변수 X로 하고, 조인 융합 테이블과 소스 테이블간 TF-IDF 중첩을 변수 Y로 하여, 두 변수 X, Y의 피어슨 상관계수를 계산

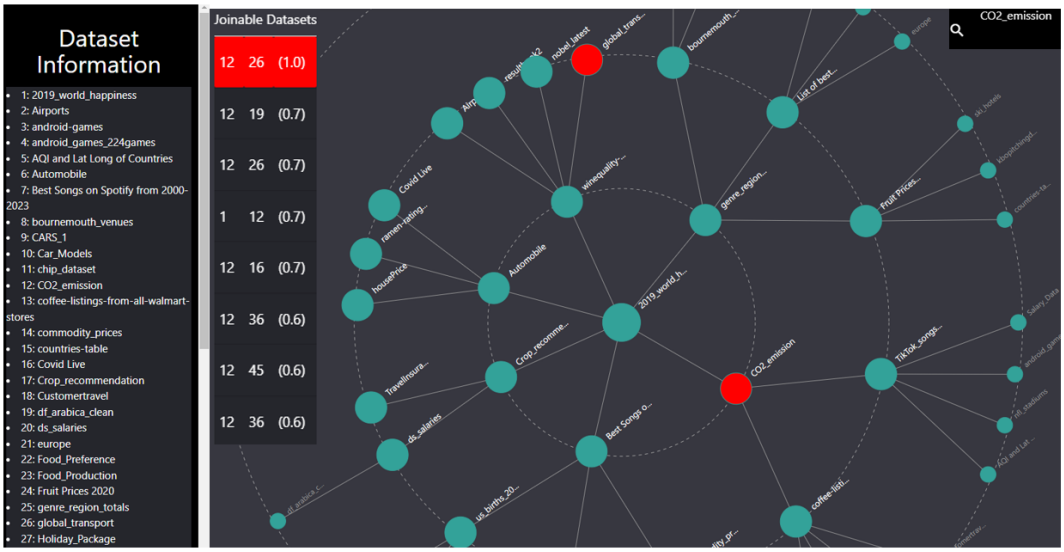


〈Figure 19〉 N-Join-Pair Performance Evaluation Results

해 보았다. 그 결과 상관계수가 0.624로 상관관계가 있음을 확인할 수 있었다. 이를 통해 N-Join-Pair의 조인 테이블을 랭킹하는 기능이 유의미하다고 판단할 수 있다.

4.3 Case Study

본 절에서는 CNE-Join을 활용하여 사용자가 한 눈에 테이블간의 관계와 조인 가능성을 파악할 수 있는 UI인 데이터맵에 대해 설명한다. 데이터맵은 사용자가 CNE-Join을 활용하여 조인 가능한 테이블쌍을 쉽게 찾을 수 있게 도와주는 UI로, <Figure 20>과 같은 구조를 가진다. 각각의 노드는 하나의 테이블에 대응되며, <Figure 20>은 총 48개의 테이블을 가지고 데이터맵을 실행한 화면이다. 데이터맵은 중앙 노드를 중심으로, 중앙 노드에 매핑된 테이블과 유사한 테이블들을 중앙과 가까운 노드에 매핑한다. 테이블간 유사도 계산은 다음과 같은 방식으로 진행된다. 우선 각 테이블의 컬럼명들을 전부 FastText를 활용해 벡터로 임베딩하고, 임베딩된 모든 벡터들의 평균을 내 테이블을 대표하는 하나의 벡터를 생성한다. 두 테이블간 유사도는 앞서 구한 테이블을 대표하는 벡터간의 코사인 유사도 계산을 통해 구한다. 중앙 노드는 처음 실행하면 임의의 테이블로 매핑되는데, 중앙 노드에 매핑된 테이블을 변경하고 싶으면, 변경하고 싶은 테이블이 있는 노드를 더블클릭하면 해당 노드가 중앙으로 배치되며, 해당 노드를 중심으로 노드들이 화면에 재구성된다. 데이터맵의 좌측에 'Joinable Datasets'에는 데이터맵에 존재하는 테이블에 대해 조인 가능한 테이블쌍이 나와 있다. 데이터맵은 각 테이블쌍에 대해 조인 가능한 정도를 0에서 1 사이의 값으로 정량화하고, 이 값을 기준으로 내림차순 정렬하여 사용자에게 보여준다. 'Joinable Datasets'에 있는 조인 가능한 테이블쌍을 클릭하면 <Figure 20>과 같이 해당 테이블쌍이 매핑된 노드에



〈Figure 20〉 Searching for Tables In Datamap

CNE-Join Time Complexity : $O(T^2 * c_1 * c_2 * (r_1 + r_2))$

- **T : Number of tables**
- c_1 **: Number of columns in the first table of the table pair**
- c_2 **: Number of columns in the second table of the table pair**
- r_1 **: Number of records in the first table of the table pair**
- r_2 **: Number of records in the second table of the table pair**

〈Figure 21〉 Time Complexity of CNE-Join

색이 칠해진다. 데이터맵은 검색 기능도 구현되어 있어, 검색창에 특정 테이블 명을 검색하면 해당 테이블과 조인 가능한 테이블쌍 정보가 'Joinable Datasets'에 제공된다. <Figure 20>을 보면 우측 상단 검색창에 'CO_2_emission'을 검색하여 'Joinable Datasets'에 'CO_2_emission' 테이블과 조인 가능한 테이블이 제공되는 것을 확인할 수 있다. 마지막으로 데이터맵은 자동 조인 기능을 갖추고 있다. <Figure 20>의 'Joinable Datasets'의 조인 가능한 테이블쌍을 3번 연속으로 클릭하면 해당 테이블쌍을 조인한 파일이 자동으로 생성된다. 이처럼 사용자는 데이터맵을 활용해 CNE-Join을 손쉽게 활용할 수 있다.

4.4 실험에 대한 고찰

본 절에서는 CNE-Join과 N-Join-Pair의 가치와 고려 사항에 대해 소개한다. 우선 CNE-Join은 기존 조인 융합 기법보다 좋은 성능의 기법이며, 데이터맵을 통해 사용자가 손쉽게 CNE-Join의 결과를 활용하여 방대한 규모의 테이블 데이터에 대해 조인 가능한 테이블쌍의 정보를 얻을 수 있다는 점에서 가치가 있다. CNE-Join이 고려할 사항은 기법의 실행시간이 길다는 점이다. <Figure 21>은 CNE-Join의 시간복잡도를 나타낸 그림으로, 높은 시간복잡도를 가지는 것을 확인할 수 있다. 또한 실험 성능 평가에 연구자의 주관이 들어갈 수 있다는 점도 고려할 사항이다. 다만 CNE-Join의 실행시간복잡도가 높다는 단점은 샘플링 기법을 활용해 일부 컬럼값만 샘플링함으로써 실제 실행시간을 크게 줄여 극복할 수 있다. 또한 CNE-Join에서 조인 융합 가능한 테이블을 찾을 때 LLM을 통한 임베딩 방안도 성능 향상을 위해 고려해볼 만하다. N-Join-Pair는 테이블 2개를 조인 융합하는 것에서 확장하여 3개 이상의 테이블을 조인 융합하는 기법을 개발하고, 조인 융합한 테이블들을 랭킹하는 것이 의미가 있음을 증명하였다. 사용자는 N-Join-Pair 기법을 활용해 양질의 조인 융합된 테이블들을 랭킹순으로 제공받을 수 있다는 점에서 가치가 있다. 다만 N-Join-Pair가 고려해야 할 점으로는 조인 가능한 N개의 테이블을 자동으로 찾아내고 융합하는 것이 목적이었지만, 아직 실험적으로 4개 이상의 테이블 융합은 다루지 못했다는 점이다. 다만 이번에 실험하여 얻은 3개 테이블의 융합 실험 평가 결과가 좋았던 것을 통해 4개 이상의 테이블 융합도 충분히 좋은 결과가 나올 것으로 예상해 볼 수 있다.

5. 결 론

본 논문은 입력으로 다수의 테이블이 주어졌

을 때, 컬럼명 임베딩, 개체명 인식, 포괄성 계산을 통해 조인 융합 가능한 테이블을 자동 탐색하는 기법인 CNE-Join을 제안한다. 또한 CNE-Join의 결과인 조인 가능한 테이블 쌍을 활용해 조인 가능한 3개 이상의 테이블을 탐색하고, 이를 랭킹하는 기법인 N-Join-Pair를 제안한다. 우리는 제안 기법의 우수성을 보이기 위해 CNE-Join과 기존의 자동 조인 융합 기법들의 조인 가능한 테이블들을 찾아내는 정확도와 양을 비교하는 실험을 진행하였고, 그 결과 CNE-Join 기법의 우수성을 증명하였다. N-Join-Pair는 기존에 존재하지 않는 3개 이상의 테이블을 조인하고 이를 랭킹하는 기법으로, 이를 활용하여 사용자는 다수의 테이블 데이터 내에서 양질의 조인 테이블들을 얻어내는 것이 가능하다. 향후 연구에서는 자동 조인 융합뿐만 아니라 자동 유니온 융합도 진행할 수 있는 기법을 구상할 예정이다. 이를 통해, 다수의 테이블 데이터가 주어졌을 때, 이를 조인 및 유니온 융합이 가능한지 판단하고 자동 융합을 진행해 새로운 정보를 창출해내는 기법을 개발할 예정이다.

References

[1] Batini, C., Cappiello, C., Francalanci, C., and Maurino, A., “Methodologies for data quality assessment and improvement,” ACM Computing Surveys (CSUR), Vol. 41, No. 3, pp. 1-52, 2009.

[2] Bharadiya, J., Thomas, R. K., and Ahmed, F., “Rise of artificial intelligence in business and industry,” Journal of Engineering Research and Reports, Vol. 25, No. 3, pp. 85-103, 2023.

[3] Bogatu, A., Fernandes, A. A. A., Paton, N. W., and Konstantinou, N., “Dataset discovery in data lakes,” In 2020 IEEE 36th International Conference on Data Engineering (ICDE), pp. 709-720, 2020.

[4] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T., “Enriching word vectors with subword information,” Transactions of the association for computational linguistics, Vol. 5, pp. 135-146, 2017.

[5] Budach, L., Feuerpfeil, M., Ihde, N., Nathansen, A., Noack, N., Patzlaff, H., Naumann, F., and Harmouch, H., “The effects of data quality on machine learning performance,” 2022.

[6] Devlin, J., Chang, M., Lee, K., and Toutanova, K., “Bert: Pre-training of deep bidirectional transformers for language understanding,” 2018.

[7] Dong, Y., Takeoka, K., Xiao, C., and Oyamada, M., “Efficient joinable table discovery in data lakes: A high-dimensional similarity-based approach,” In 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 456-467, 2021.

[8] Dong, Y., Xiao, C., Nozawa, T., Enomoto, M., and Oyamada, M., “DeepJoin: Joinable Table Discovery with Pre-trained Language Models,” 2022.

[9] Fernanadez, R.C., Mansour, E., A. Qahtan,

A., Elmagarmid, A., Ilyas, I., Madden, S., Ouzzani, M., Stonebraker, M., and Tang, N., “Seeping semantics: Linking datasets using word embeddings for data discovery,” 2018 IEEE 34th International Conference on Data Engineering (ICDE), pp. 989-1000, 2018.

[10] Gupta, N., Mujumdar, S., Patel, H., Masuda, S., Panwar, N., Bandyopadhyay, S., Mehta, S., Guttula, S., Afzal, S., Mittal, R. S., and Munigala, V., “Data quality for machine learning tasks,” Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021.

[11] Himeur, Y., Rimal, B., Tiwary, A., and Amira, A., “Using artificial intelligence and data fusion for environmental monitoring: A review and future perspectives,” Information Fusion, Vol. 86, pp. 44-75, 2022.

[12] “Inshorts Dataset - English News,” https://www.kaggle.com/datasets/shivamtaneja2304/inshorts-dataset-english.

[13] Meng, T., Jing, X., Yan, Z., and Pedrycz, W., “A survey on machine learning for data fusion,” Information Fusion, Vol. 57, pp. 115-129, 2020.

[14] Mikolov, T., Chen, K., Corrado, G., and Dean, J., “Efficient estimation of word representations in vector space,” 2013.

[15] “Named Entity Recognizer,” https://towardsdatascience.com/named-entity-recognition-with-bert-in-pytorch-a454405e0b6a.30, 2017.

[16] “NER Data,” https://www.kaggle.com/datasets/rajnathpatel/ner-data.

[17] Pilaluisa, J., Tomas, D., Navarro-Colorado, B., and Mazon, J.-N., “Contextual word embeddings for tabular data search and integration,” Neural Computing and Applications, Vol. 35, No. 13, pp. 9319-9333, 2023.

[18] Qaiser, S. and Ali, R., “Text mining: Use of TF-IDF to examine the relevance of words to documents,” International Journal of Computer Applications, Vol.181, No. 1, pp. 25-29, 2018.

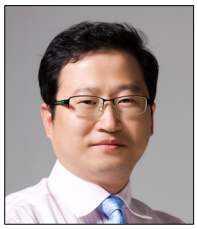
[19] Whang, S. E., Roh, Y., Song, H., and Lee, J.-G., “Data collection and quality challenges in deep learning: A data-centric AI perspective,” The VLDB Journal, Vol. 32, pp. 791-813, 2023.

[20] Yoon, J. C. and Kim, H. J., “A relational table embedding techniqure for data fusion,” The Journal of Society for e-Business Studies, Vol. 27, No. 3, pp. 1-19, 2022.

[21] Zhu, E., Deng, D., Nargesian, F., and Miller, R. J., “Josie: Overlap set similarity search for finding joinable tables in data lakes,” In 2019 International Conference on Management of Data, pp. 847-864, 2019.

저 자 소 개

최윤석 (E-mail : choiys8819@naver.com)
2023년~현재 서울시립대학교 전자전기컴퓨터공학과 석사과정
관심분야 딥러닝, 빅데이터 분석, 테이블 데이터 융합, 임베딩



김한준 (E-mail : khj@uos.ac.kr)
1994년 서울대학교 계산통계학과 (이학사)
1996년 서울대학교 전산과학과 (이학석사)
2002년 서울대학교 컴퓨터공학부 (공학박사)
2002년~현재 서울시립대학교 전자전기컴퓨터공학부 정교수
관심분야 빅데이터 분석, 머신러닝, 텍스트마이닝, 데이터베이스, 정보검색