ISSN 2383-630X(Print) / ISSN 2383-6296(Online)
Journal of KIISE, Vol. 47, No. 10, pp. 958-964, 2020. 10
https://doi.org/10.5626/JOK.2020.47.10.958

# 지식 그래프 임베딩 및 적응형 클러스터링을 활용한 오류 트리플 검출

## (Incorrect Triple Detection Using Knowledge Graph Embedding and Adaptive Clustering)

신 원 철 † 노 재 승 † 박 영 택 ††
(Won-Chul Shin) (Jea-Seung Roh) (Young-Tack Park)

**요 약** 최근 인터넷의 발전으로 정보의 양이 늘어나면서 대용량 지식 그래프를 이용한 연구가 활발히 이루어지고 있다. 또한 지식 그래프가 다양한 연구와 서비스에 활용됨에 따라 양질의 지식 그래프를 확보해야 하는 필요성이 대두되고 있다. 하지만 양질의 지식 그래프를 얻기 위해 지식 그래프 내 오류를 검출하는 연구가 부족하다. 오류 트리플 검출을 위해 임베딩과 클러스터링을 사용한 이전 연구가 좋은 성능을 나타냈다. 하지만 클러스터 최적화 과정에서 일괄적으로 동일한 임계값을 사용하여 각 클러스터의 특성을 고려하지 못하는 문제가 존재하였다. 본 논문에서는 이러한 문제를 해결하고자 지식 그래프 내 오류 트리플 검출을 위해 지식 그래프에 대한 임베딩과 함께 각 클러스터에 대한 최적의 Threshold를 찾아 적용함으로써 클러스터링을 진행하는 적응형 클러스터링 모델을 제안한다. 본 논문에서 제안하는 방법의 성능을 평가하기 위해 DBpeida, Freebase와 WiseKB 세 가지 데이터셋을 대상으로 기존 오류 트리플 검출 연구와 비교 실험을 진행하였으며 F1-Score를 기준으로 평균 5.3% 높은 성능을 확인하였다.

**키워드:** 지식 그래프, 임베딩, 클러스터링, 딥러닝

***Abstract*** Recently, with the increase in the amount of information from the development of the Internet, research using large-capacity knowledge graphs is being actively conducted. Additionally, as knowledge graphs are used for various research and services, there is a need to secure quality knowledge graphs. However, there is a lack of research to detect errors within the knowledge graphs to obtain quality knowledge graphs. Previous studies using the embedding and clustering for error triple detection showed good performance. However, in the process of the cluster optimization, there was a problem that the characteristics of each cluster could not be factored using the same threshold collectively. In this paper, to resolve these problems, we propose an adaptive clustering model in which clustering is conducted by finding and applying the optimum threshold for each cluster with the embedding for knowledge graph for error triple detection in the knowledge graph. To evaluate the performance of the method proposed in this paper, the existing error triple detection studies and comparative experiments were conducted on three datasets, DBpeida, Frebase and WiseKB, and the high performance was confirmed by an average of 5.3% based on the F1-Score.

**Keywords:** knowledge graphs, embedding, clustering, deep learning

· 이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No.2019000067,대용량 지식그래프 자동완성을 위한 시맨틱 분석 추론기술 개발)

† 학생회원 : 숭실대학교 컴퓨터학과 학생
ocshin1201@naver.com
rjs951001@gmail.com
†† 종신회원 : 숭실대학교 컴퓨터학과 교수(Soongsil Univ.)
park@ssu.ac.kr
(Corresponding author임)

논문접수 : 2020년 4월 29일
(Received 29 April 2020)
논문수정 : 2020년 8월 4일
(Revised 4 August 2020)
심사완료 : 2020년 8월 20일
(Accepted 20 August 2020)


정보과학회논문지 제47권 제10호(2020. 10)

## 1. 서 론

지식 그래프는 실세계에 존재하는 개체와 개체 사이의 의미적 관계를 효과적으로 표현할 수 있는 수단이며 최근 다양한 인공지능 연구에서 활용되고 있다. 대표적인 지식 그래프로 YAGO[1], Freebase[2], DBpedia[3]가 있다. 지식 그래프는 온톨로지 형태로 표현된다. 온톨로지는 일종의 지식표현으로 지식개념을 의미적으로 연결할 수 있는 도구로서 RDF, RDF-S, OWL 등의 언어를 이용해 표현된다. 가장 단순한 형태의 RDF는 <Subject, Predicate, Object>의 트리플 형태로 구성되어 개념을 표현한다. Subject와 Object는 표현하고자 하는 개념의 주어, 목적어에 해당하는 개체이고 Predicate는 지식개념의 주어와 목적어 개체사이의 관계를 나타내는데 사용한다. 이와 같이 온톨로지를 통한 데이터 표현 방식은 컴퓨터로 하여금 트리플로 표현된 개념을 이해하고 지식처리를 가능하게 한다. 하지만 지식 그래프는 대부분 웹으로부터 자동적으로 수집하여 생성된다. 이러한 방법으로 생성된 지식 그래프는 웹 데이터의 불완전성으로 인해 그래프의 트리플이 누락되거나, 존재하는 트리플에 오류가 생기는 문제가 있다. 이와 같은 지식 그래프의 불완전성 문제로 인해 잘못된 지식이 포함된 지식 그래프를 활용한 연구나 서비스는 치명적인 오류를 발생 시킬 수 있다. 하지만 이를 해결하기 위해 방대한 양의 지식 그래프를 정제하는 작업은 현실적으로 매우 어렵다. 따라서 지식 그래프 내 잘못된 정보를 검출해내는 기술이 요구 되고 있다. 지식 그래프 내 잘못된 정보를 검출하기 위해 이전 연구에서 지식베이스 임베딩 및 관계 모델을 활용한 오류 트리플 검출[4]을 제안한 바 있다. 이 방법에서는 지식 그래프 내 오류 검출을 위해 지식 그래프를 임베딩을 진행한 뒤 K-means[5] 클러스터링을 통해 특정 트리플에 대한 클러스터를 생성한다. 이후 클러스터 범위를 재조정하여 클러스터 외부에 위치한 것을 오류 트리플로 검출하였다. 하지만, 위의 방법에서는 모든 클러스터에 동일한 Threshold를 적용하여 클러스터의 범위를 일괄적으로 조정한다. 이는 클러스터의 범위를 기준으로 오류를 판별하는 모델임에도 불구하고 클러스터 각각의 특징을 고려하지 못하기 때문에 판별 정확도가 떨어지게 된다. 이러한 문제를 해결하기 위해 본 논문에서는 이전 연구와 달리 각 클러스터의 특성을 고려하여 클러스터마다 최적의 Threshold를 적용하는 적응형 클러스터링 방법을 제안한다. 개선된 클러스터링 방법의 성능을 확인하기 위해 이전 연구와 비교실험을 진행하였으며 보다 좋은 성능을 얻는 것을 확인하였다.

## 2. 관련 연구

### 2.1 Embedding

단어 임베딩은 텍스트로 존재하는 자연어의 단어를 특정 벡터를 사용하여 표현하기 위한 방법이다. 단어 임베딩의 목적은 유사한 단어들끼리 가까운 거리에 위치하도록 각 단어가 가지고 있는 의미를 학습하여 이를 벡터로 표현하는 것이다. 효과적인 단어 임베딩을 위해 기존 연구는 대표적으로 Word2Vec[6]과 GloVe[7] 등이 존재한다. 두 모델은 텍스트를 처리하는 신경망 모델로 문장을 입력으로 받아 벡터를 학습한다.

### 2.2 RDF2Sentence

지식 그래프는 대용량의 데이터를 RDF 형태의 트리플로 저장하고 있다. RDF2Sentence는 이러한 RDF 형태의 트리플을 연결하여 문장으로 생성하기위한 것으로 Deep Walk[8]와 연관성이 있다. 트리플로부터 문장을 생성하기위하여 우선 각 엔티티의 주변 트리플을 연결하고 엔티티 시퀀스를 추출하여 이를 트리 구조로 형성한다. 트리 구조로 형성된 그래프로부터 랜덤 워크를 통해 연결된 트리플들을 탐색하며 트리플들을 연결하여 문장으로 만들게 된다.

### 2.3 Knowledge Graph Embedding

프로그램을 통한 지식 그래프 처리를 위한 한 가지 방법으로 지식 그래프 임베딩 존재한다. 지식 그래프 임베딩을 위한 기존의 연구는 대표적으로 TransE[9], TransR[10], DistMult[11], ConvE[12] 등이 있다.

TransE 모델은 트리플의 head 엔티티 벡터와 릴레이션 벡터에 대하여 합연산을 수행하였을 때의 출력 벡터가 tail 엔티티 벡터를 가리킨다는 가정으로 임베딩을 수행하는 모델이다. 여기서 릴레이션은 head 엔티티와 tail 엔티티의 관계를 번역하는 연산자로써 사용되는데 이 때 엔티티와 릴레이션이 같은 시맨틱 공간을 사용하도록 구성하였다. 하지만 실제로 엔티티는 다양한 측면의 의미를 가질 수 있으며 각 릴레이션은 엔티티의 다른 의미와 관련될 수 있다. TransR에서는 엔티티와 릴레이션을 위한 임베딩 공간을 서로 독립된 공간에 구축하며 임베딩 벡터 학습을 위해 학습하고자 하는 릴레이션을 위한 임베딩 공간에 엔티티 공간의 벡터들을 투영하여 투영된 벡터들을 기반으로 head와 tail 엔티티를 연결하는 연산자로써 릴레이션 임베딩을 학습하게 된다. 임베딩 기반 연구 방법 중 최근 가장 좋은 결과를 보인 ConvE는 엔티티와 릴레이션 임베딩 벡터를 입력으로 받아 이를 2D Convolution을 수행하고 임베딩 벡터의 표현력을 높이기 위해 fully connected layer와 활성 함수를 통해 비선형성을 적용하여 벡터를 학습한다.

### 2.4 지식 베이스 임베딩을 활용한 오류트리플 검출

기존에 제안된 지식베이스 임베딩 및 관계 모델을 활용한 오류 트리플 검출(Ji-Hun Hong et al. 2019)에서는 다차원 벡터공간에 표현된 엔티티 벡터를 클러스터링 하고 모든 클러스터의 범위에 동일한 Threshold δ를 일괄적으로 적용하여 조정함으로써 오류 트리플 검출 모델을 얻는다. 그림 1은 생성된 모델을 통해 오류 트리플을 검출하는 방법으로써 예를 들어 Person을 type으로 갖는 트리플 중 오류 트리플을 검출한다고 할 때 '*'는 실제 Person인 엔티티 벡터이고, '-'는 Person이 아닌 다른 type을 가지는 엔티티 벡터가 된다. 모델은 클러스터 범위를 기준으로 범위 외부에 위치한 엔티티 벡터를 오류 트리플의 엔티티 벡터라고 검출하게 된다. 이때 r은 클러스터범위 조정 전 클러스터 범위의 초기 값이고 오류판별을 가장 잘 수행할 수 있는 최적의 Threshold δ를 찾아 r에 곱함으로써 이를 모델 생성에 사용한다.

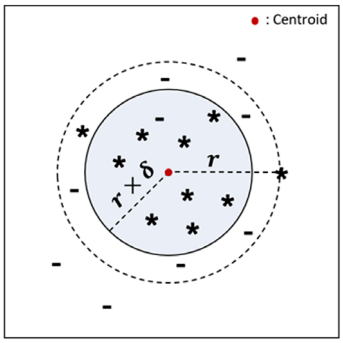


그림 1 임베딩을 통한 오류 트리플 검출의 예
Fig. 1 Example of error triple detection with the embedding

## 3. 연구 내용

### 3.1 Property Sentence 생성

Word2Vec은 문장 중간 단어로 주변 단어들을 예측하는 방법으로 말뭉치 데이터를 통해 모든 단어를 벡터공간에 임베딩하여 각 단어를 다차원 벡터공간에 표현한다. Glove는 말뭉치 내의 전체 단어의 통계 정보를 활용하지 못하는 Word2Vec의 단점을 보완한 알고리즘으로 두 단어 벡터의 내적 값이 말뭉치 전체에서 동시 등장 확률의 로그 값이 되도록 한다. 본 논문에서는 지식 그래프를 Word2Vec과 Glove를 통해 임베딩 하기 위해 트리플 형태의 데이터를 RDF-Sentence 형태로 확장하여 임베딩을 수행한다.

본 논문에서는 엔티티의 특징을 분류하여 오류 트리플을 검출해야 하므로, 기존의 RDF-Sentence방법을 사용하는 대신 Property-Sentence 방법론을 사용하였다. type정보가 Person인 트리플로부터 Sentence를 생성한다고 할 때 그림 2와 같이 <$e_0^1$ type Person>과 같은 트리플에서 주어인 $e_0^1$과 목적어인 Person으로부터 해당 엔티티를 주어나 목적어로 갖는 트리플을 연결하게 되면 $r_1^1, e_1^6, r_2^1, e_2^4$ 까지 확장될 수 있고, $e_0^1$으로부터 $r_1^0, e_1^3, r_2^0, e_2^3$ 까지 확장될 수 있다.

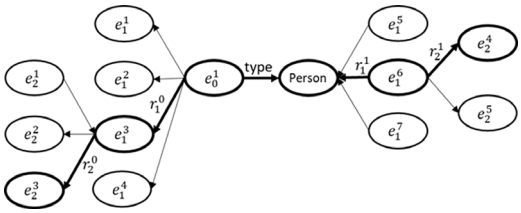


그림 2 RDF-Sentence 생성예시
Fig. 2 Example of the RDF-Sentence generation

RDF-sentence : "$e_2^3 r_2^0 e_1^3 r_1^0\ e_0^1$ type Person $r_1^1 e_1^6 r_2^1 e_2^4$"
Property-sentence : "$r_2^0 r_1^0\ e_0^1$ type Person $r_1^1 r_2^1$"

이와 같이 확장된 그래프로부터 RDF-sentence를 생성할 수 있고 property-sentence는 생성된 RDF-sentence에서 엔티티는 제외하고 릴레이션만 포함하게 된다. 이와 같은 방법은 엔티티의 특징을 부각시키게 되어 엔티티 클러스터링에 도움을 주게 된다.

### 3.2 적응형 클러스터링 적용 방법

K-means알고리즘을 통해 생성된 클러스터들은 지식 그래프 임베딩 결과에 따라 각각 서로 다른 데이터 분포를 갖는다. 하지만 기존의 클러스터링 방식은 생성된 여러 클러스터의 범위를 동일 한 Threshold를 가지고 일괄적으로 조정하기 때문에 지식 그래프에서 특정 트리플의 각 엔티티에 대한 포괄적인 특징에 대해서만 모델링 하는데 그친다. 본 논문에서는 클러스터의 범위를 조정할 때, 생성된 클러스터 각각의 특징을 고려하도록 클러스터에 따라 서로 다른 Threshold를 갖는 적응형 클러스터링 방법을 제안하여 기존 방법의 문제를 해결하고자 한다. 본 논문에서 제안하는 클러스터링 방법은 먼저 지식 그래프의 트리플 데이터를 문장구조의 RDF-Sentence형태로 변환하고, Word2Vec과 GloVe를 사용하여 벡터 공간에 임베딩하는 방법과, 트리플 형태의 데이터를 TransR과 ConvE와 같은 임베딩 알고리즘을 사용하여 벡터공간에 임베딩하여 각 엔티티 벡터를 생성하였다. 본 논문에서 제안하는 방법은 엔티티의 특징을 군집화하여 오류 트리플을 검출하는 것이므로 먼저 생성된 엔티티 벡터 중 오류가 아닌 올바른 트리플의 엔티티

벡터를 K-means 클러스터링을 통해 클러스터링 한다.

K-means 클러스터링은 주어진 데이터를 k개의 클러스터로 묶는 알고리즘으로, 각 클러스터는 하나의 중심을 가진다. 이 때 클러스터의 중심과 클러스터 내의 오브젝트와의 거리의 제곱 합을 비용 함수로 정한다. 다시 말해, $\mu_i$가 집합 $S_i$의 중심점이라 할 때 각 집합별 중심점으로부터 집합 내 오브젝트간 거리의 제곱합을 최소로 하는 집합 $S = \{S_1, S_2, ..., S_k\}$를 찾는 것이 K-means 알고리즘의 목표이다.

$$\arg\min_{S} \sum_{i=1}^{k} \sum_{x \in S_i} \| x - \mu_i \|^2$$

위 수식은 K-means 의 목적함수로, 함수를 만족하는 집합을 찾아줌으로써 클러스터링을 수행하게 된다. 클러스터 수에 해당하는 변수 k는 Gap 알고리즘[13]을 사용하여 최적의 k 값을 찾는다. 올바른 트리플의 엔티티 벡터의 클러스터가 생성 되면 해당 트리플의 엔티티 벡터의 특징을 클러스터링을 통해 모델링 했다고 볼 수 있다. centroid는 클러스터의 중심점으로 해당 클러스터의 centroid로부터 가장 먼 거리에 위치한 클러스터 내 엔티티 벡터를 연결하여 그림 3과 같이 클러스터의 범위를 설정하게 된다. 이와 같이 초기 클러스터는 centroid로부터 가장 먼 거리에 위치한 클러스터 내 엔티티 벡터와의 거리를 초기 범위로 설정하였기 때문에 올바른 트리플의 학습 임베딩 벡터('*')가 전부 포함되는 것을 확인할 수 있다. 즉 올바른 트리플은 전부 올바르다 판별할 수 있지만 오류 트리플 판별에 대한 최적화 작업이 수행되지 않은 상태이다. 본 논문에서 제안하는 방법은 비지도 학습인 K-means가 선행되어야 하고, 이후 클러스터 최적화를 위해 클러스터에 포함된 요소들의 레이블을 필요로 하므로 지도 학습 방법이라고 볼 수 있다.

앞서 올바른 트리플의 엔티티 벡터를 클러스터링 과정을 통해 임베딩 모델과 수치화된 초기 클러스터의 범위 r을 얻게 되면 클러스터 범위 조절이 가능한 상태가 된다. 예를 들어 생성한 클러스터가 지식 그래프에서 type이 Person이라는 정보를 나타내는 트리플의 엔티티 벡터로 구성되어있는 클러스터라고 할 때, 지식 그래프에서 type이 Person이라는 정보를 나타내는 트리플 중 엔티티가 실세계에선 사람이 아닌 엔티티의 임베딩 벡터가 생성된 클러스터 내부에 포함되어 있다면 클러스터의 범위를 축소함으로써 해당 엔티티 벡터가 클러스터 범위에 포함되지 않도록 조정한다.

클러스터 내부의 데이터 분포와 밀집정도와 같은 클러스터 각각의 특징을 모델링하기 위해선 범위 조정 시 클러스터 각각이 독립적으로 수행되도록 해야 한다. 클러스터 범위 조정은 앞서 수치화된 클러스터 범위 r에 1보다 작은 값을 가지는 임계값 δ를 곱함으로써 수행된다.

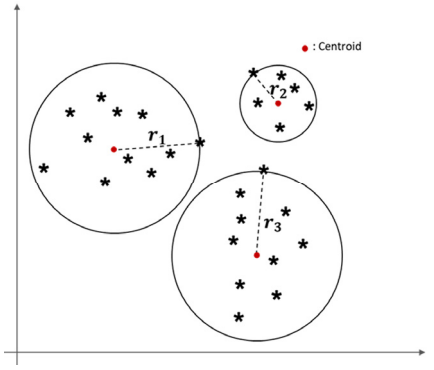


그림 3 K-means를 통해 생성 된 클러스터 모델
Fig. 3 Cluster models created with the K-Means

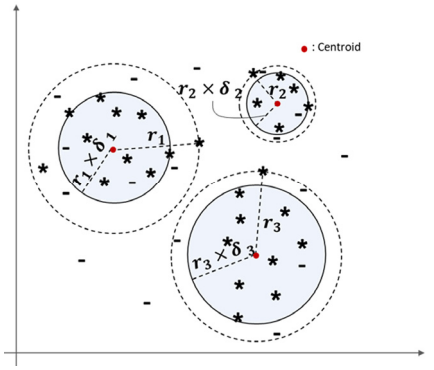


그림 4 최적의 δ를 적용한 적응형 클러스터 모델
Fig. 4 Adaptive cluster model with the optimal δ

오류 트리플을 가장 잘 검출하는 Threshold δ를 찾아 클러스터에 적용하였을 때 그림 4와 같은 클러스터 모델을 얻을 수 있다. 본 논문에서는 클러스터 각각을 독립적으로 최적화하는 Threshold δ를 찾는 방법을 제안한다. 클러스터 각각의 Threshold를 찾기 위해선 먼저 최적의 Threshold를 결정하기 위한 지표가 필요하다. 올바른 트리플의 엔티티 벡터('*')가 클러스터 범위 내부에 속해 있다면 True Positive이고, 외부에 위치한다면 False Negative 오류 트리플의 엔티티 벡터('-')가 클러스터 범위 내부에 속해있다면 False Positive 외부에 위치한다면 True Negative로 클러스터 각각에 대한 컨퓨전 행렬을 얻을 수 있다. 이후에 F1-Score가 최대가 되도록 하는 Threshold를 찾아 적용하게 되면 해당 클러스터의 범위 내에 오류 트리플의 엔티티는 최대한 적게 포함 하면서 클러스터 범위 내에 올바른 트리플의 엔티티는 최대한 많이 포함할 수 있는 클러스터를 얻을 수 있다. 위와 같은 과정을 생성된 모든 클러스터 각각에 대해 독립적으로 적용하여 모델을 생성한다.

## 4. 실 험

본 연구에서 제안하는 모델의 성능을 검증하기 위해 기존의 지식베이스 임베딩 및 관계 모델을 활용한 오류 트리플 검출(Ji-Hun Hong et al. 2019)모델과 비교 실험을 수행하였다. 기존 오류 트리플 검출 방법도 K-means가 선행 되어야 하고 클러스터 최적화 과정에서 데이터의 레이블을 필요로 하므로 지도 학습이라고 볼 수 있다. 비교 실험은 DBpedia, FreeBase, WiseKB 데이터를 통해 지식 그래프 내 오류 트리플 검출 실험을 진행한다.

### 4.1 실험 데이터

실험을 위해 세 가지 지식 그래프로부터 오류 트리플 검출을 위한 부분 그래프를 추출하여 사용하며 세 가지 지식 그래프의 통계를 표 1에서 나타낸다. 본 논문에서는 위 3가지 지식 그래프로부터 오류 트리플 검출을 위한 실험을 진행하고자 전체 지식 그래프로부터 정제 작업을 수행하여 구축한다. 정제 작업에서는 타겟이 되는 클래스와 유사 타입을 선정하여 지식 그래프로부터 각 35.000개의 type 정보에 대한 트리플을 추출하였으며 이와 관련된 통계는 표 2에서 나타낸다. 각 지식 그래프에서는 학습을 위해 타겟 클래스로 20,000개와 유사 클래스 5,000개를 사용하고 나머지 타겟 5,000개와 유사 클래스 5,000개를 테스트셋으로 사용한다. 임베딩 벡터 학습은 유사클래스의 type정보를 Person으로 하여 오류 트리플을 생성 하고, 각 트리플마다 500개의 Sentence를 생성하여 Skip-Gram, Glove를 사용하여 엔티티 임베딩을 진행하였고, TransE, TransR, ConvE는 추출된 type 정보에 대한 트리플 외에 해당 엔티티를 주어나 목적어로 갖는 다른 트리플들을 포함하여 임베딩을 수행하였다.

표 1 지식 그래프 정보 요약
Table 1 Summary of the knowledge graphs

| Dataset | # Triples | # Entities | # Class | # Relations |
|---|---|---|---|---|
| DBpedia | 17,887K | 6,600K | 6,699 | 1,159 |
| Freebase | 22,955K | 5,919K | 418 | 663 |
| WiseKB | 31,068K | 9,422K | 1,673 | 555 |

### 4.2 실험 방법 및 결과

실험 방법으로는 본 논문에서 제안하는 모델의 성능을 검증하기 위하여 기존의 오류 트리플 검출(Ji-Hun Hong et al. 2019)모델과 비교 실험을 수행하였다. 비교 실험은 DBpedia, Freebase, WiseKB 세 가지 지식 그래프에 대하여 Skip-gram, GloVe, TransE, TransR과 ConvE 다섯 가지 임베딩 알고리즘을 통해 임베딩 벡터를 추출하여 오류 트리플 검출성능을 몇 가지 측정 방법을 통해 비교하였다. 모델의 정량적 평가를 위해 결과를 Precision, Recall과 F1-Score를 통해 비교하였다. Precision 값이 클수록 클러스터 내의 오류 트리플보다 올바른 트리플의 비중이 높다는 것을 확인할 수 있으며 Recall 값이 클수록 올바른 트리플을 많이 찾아냈다는 것을 확인할 수 있다. 하지만 위 두 값만으로는 오류 트리플을 많이 검출하였지만 올바른 트리플을 또한 오류 트리플로 잘못 예측하는 경우가 존재하거나 올바른 트리플을 찾는데 집중되어 오류 트리플을 제대로 분류하지 못하는 경우가 존재할 수 있다. 이를 위해 Precision과값이 클수록 오류 트리플 검출이 잘 이루어졌다고 판단할 수 있다.

표 3은 각 임베딩 알고리즘마다 DBpedia, Freebase, WiseKB 3개의 테스트 데이터 셋에 대한 오류 트리플 검출 실험을 진행하여 F1-Score의 평균을 구한 결과로 기존의 지식베이스 임베딩 및 관계 모델을 활용한 오류 Recall을 모두 고려하기 위해 F1-Score를 사용하여 이

표 3 기존 모델과 제안한 모델의 성능 비교
Table 3 Performance comparison of the existing and proposed models

| | Avg. F1-Score | |
|---|---|---|
| | fixed (Ji-Hun Hong et al. 2019) | adaptive (Our Approach) |
| Skip-gram | 0.7238 | 0.7707 |
| GloVe | 0.6840 | 0.7039 |
| TransE | 0.7551 | 0.8202 |
| TransR | 0.8211 | 0.9089 |
| ConvE | 0.8815 | 0.9284 |
| Avg. | 0.7731 | 0.8264 |

표 2 실험 데이터 요약
Table 2 Summary of the experiment data

| Dataset | Target Class (# of entities) | Similar Class (# of entities) | | | | |
|---|---|---|---|---|---|---|
| DBpedia | Person 25.000 | City 2,000 | Country 2,000 | School 2,000 | Sports Team 2,000 | Film 2,000 |
| Freebase | Person 25.000 | City 2,000 | Company 2,000 | School 2,000 | Album 2,000 | Television Show 2,000 |
| WiseKB | Person 25.000 | City 2,000 | Organization 2,000 | Song 2,000 | Show 2,000 | Education Institute 2,000 |

트리플 검출(Ji-Hun Hong et al. 2019)방법과 제안한 방법의 성능을 비교한 것이다. 표 3에서 알 수 있듯이, 제안한 방법의 전체F1-Score 평균이 기존 모델에 비해 5.3% 성능 향상을 보였으며, TransR의 경우 제안한 방법이 기존 방법보다 평균 8.7%더 좋은 성능을 보이는 것을 확인하였다.

표 4에서는 기존 방법(Ji-Hun Hong et al. 2019)과 제안된 방법 간의 비교 실험에 사용한 3개의 데이터 셋과 각 임베딩 알고리즘에 대한 테스트 결과를 Precision, Recall과 F1-Score로 측정하여 기술했으며, 데이터 별로 가장 높은 성능을 나타내는 F1-Score 결과의 경우 볼드체로 나타내었다. 본 논문에서 제안하는 방식은 TransE 임베딩 알고리즘을 사용한 경우 WiseKB 데이터에서 기존 방법을 적용하였을 때 보다 제안된 방법을 사용하였을 경우 F1-Score가 14.2% 향상됨으로써 가장 큰 성능 개선 폭을 보였다. 또한 ConvE를 통해 학습한 임베딩 벡터를 사용한 적응형 클러스터링 모델이 세 개의 데이터 셋에 대한 평균 F1-Score가 92.8%로 가장 좋은 성능을 보였다.

그림 5, 6은 기존의 지식베이스 임베딩 및 관계 모델을 활용한 오류 트리플 검출(Ji-Hun Hong et al. 2019) 방법을 사용한 경우 클러스터의 Threshold δ에 따른 모델의 F1-Score 변화에 대한 실험 결과를 보여준다. 그림 5와 6은 각각 Skip-gram, ConvE 임베딩 알고리즘을 사용한 결과이다. δ를 최적화 하기 전엔 δ가 1의 값을 갖게 된다. 이는 K-means만 사용했을 때의 결과라고 볼 수 있고 δ를 최적화 하기 전보다 F1-Score가 낮다. 그림 5에서 기존의 방식은 FreeBase 데이터 셋에 대해서 최대 83%의 F1-Scroe를 갖을 수 있고 기존 방법으로 트레이닝 데이터에 대해 δ를 최적화 했을 때 표 4에서 알 수 있듯이 F1-Score가 79%임을 보인다. 기존 방법은 생성된 모든 클러스터에 동일한 δ값을 갖지만

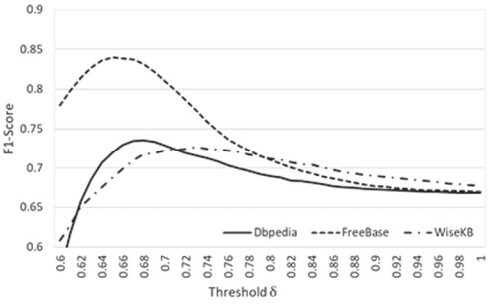


그림 5 Skip-gram 임베딩을 사용한 테스트 셋의 Threshold에 따른 성능 변화

Fig. 5 Performance variation of the test set according to the threshold using the skip-gram embedding

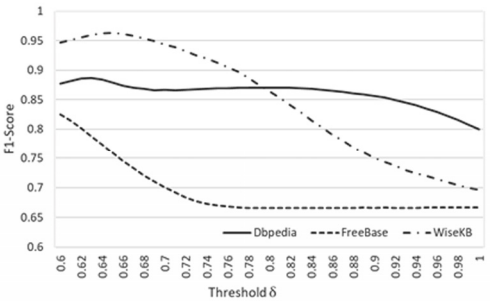


그림 6 ConvE 임베딩을 사용한 테스트 셋의 Threshold에 따른 성능 변화

Fig. 6 Performance variation of the test set according to the threshold using the convE Embedding

동일한 데이터에 대해서 제안한 방법을 사용하면 생성된 클러스터 각각에 대해 독립적으로 δ를 최적화함으로써 기존 방법보다 좋은 84%의 F1- Score를 보였다. 그림 6에서 기존의 방법은 WiseKB 데이터 셋에 대해서 최대

표 4 DBpedia, Freebase, WiseKB에서의 오류 트리플 검출 결과

Table 4 Incorrect triple detection results on the DBpedia, Freebase, and the WiseKB

| | | Skip-gram | | GloVe | | TransE | | TransR | | ConvE | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Dataset | Metrics | fixed | adaptive | fixed | adaptive | fixed | adaptive | fixed | adaptive | fixed | adaptive |
| DBpedia | Precision | 0.5031 | 0.5827 | 0.5003 | 0.5117 | 0.6647 | 0.6962 | 0.6632 | 0.8169 | 0.7898 | 0.8595 |
| | Recall | 0.9978 | 0.9594 | 0.9984 | 0.9818 | 0.9940 | 0.9862 | 0.9964 | 0.9714 | 0.9692 | 0.9996 |
| | F1-Score | 0.6689 | 0.7250 | 0.6669 | 0.6727 | 0.7966 | 0.8163 | 0.7964 | 0.8875 | 0.8704 | **0.9242** |
| Freebase | Precision | 0.7125 | 0.7728 | 0.5762 | 0.5576 | 0.6538 | 0.7055 | 0.6589 | 0.7710 | 0.7560 | 0.8042 |
| | Recall | 0.9916 | 0.9208 | 0.9488 | 0.9742 | 0.9944 | 0.9838 | 0.9932 | 0.9728 | 0.9076 | 0.9866 |
| | F1-Score | 0.7998 | 0.8403 | 0.7197 | 0.7305 | 0.7888 | 0.8217 | 0.7922 | 0.8603 | 0.8249 | **0.8861** |
| WiseKB | Precision | 0.5510 | 0.6074 | 0.4999 | 0.5584 | 0.5128 | 0.7040 | 0.7781 | 0.9650 | 0.9236 | 0.9675 |
| | Recall | 0.9700 | 0.9692 | 0.9956 | 0.9696 | 0.9886 | 0.9900 | 0.9988 | 0.9934 | 0.9768 | 0.9828 |
| | F1-Score | 0.7027 | 0.7468 | 0.6656 | 0.7087 | 0.6801 | 0.8228 | 0.8747 | **0.9790** | 0.9494 | 0.9750 |

96%의 F1-Score를 갖을 수 있고 기존방식으로 트레이닝 데이터에 대해 δ를 최적화했을 때 표 4에서 알 수 있듯이 F1-Score가 94%임을 보인다. 동일한 데이터에 대해서 제안한 방법을 사용하면 기존 방법보다 더 좋은 97%의 F1-Score를 보였다.

## 5. 결 론

본 논문에서는 임베딩 기반 오류 트리플 검출 모델에서 새로운 클러스터 모델 생성 방법인 적응형 클러스터링 방법을 제안하였다. 제안한 방법은 클러스터범위 조정 Threshold인 δ를 찾기 위한 지표인 F1-score를 임베딩된 전체 벡터공간에서 산출하는 대신 클러스터 각각의 local F1-score를 구하여 모든 클러스터를 독립적으로 최적화함으로써 클러스터 간의 독립성을 보장하였으며, 이를 통해 각 클러스터가 갖는 특징을 고려한 오류 트리플 검출 모델을 생성하여 오류 트리플 검출 효과를 향상시켰다. 실험에서는 3가지 데이터 셋에 대해 각각 5가지 임베딩 알고리즘을 사용하여 제안한 방법의 성능을 검증하였으며, 유의미한 오류검출 성능 향상을 보인 것을 확인하였다.

## References


[1] Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum, "Yago: a core of semantic knowledge," *Proc. of the 16th International Conference on World Wide Web*, ACM, 2007.

[2] Bollacker. K, Evans. C, Paritosh. P, Sturge. T, and Taylor, J., "Freebase: A collaboratively created graph database for structuring human knowledge," *Proc. of the 2008 ACM SIGMOD International Conference on Management of data*, pp. 1247-1250, 2008.

[3] Auer, S ren, et al., "Dbpedia: A nucleus for a web of open data," Springer Berlin Heidelberg, 2007.

[4] Ji-Hun Hong, Hyun-Young Choi, Wan-Gon Lee, Young-Tack Park, "Incorrect Triple Detection Using Knowledge Base Embedding and Relation Model," *Journal of KIISE*, Vol. 44, No. 2, pp. 131-140, 2019.

[5] MacQueen, James, "Some methods for classification and analysis of multivariate observations," *Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1, No. 14, 1967.

[6] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Advances in Neural Information Processing Systems*, pp. 3111-3119, 2013.

[7] Jeffrey. Pennington, Richard. Socher, and Christopher, D. Manning, "GloVe: Global Vectors for Word Representation," *Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[8] Perozzi. B, Al-Rfou, R. Skiena, S., "Deepwalk: On line learning of social representations," *Proc. of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 701-710, 2014.

[9] Antoine. Bordes, Nicolas. Usunier, Alberto. Garcia-Duran, Jason. Weston, Oksana. Yakhnenko, "Translating Embeddings for Modeling Multi-relational Data," *Advances in Neural Information Processing Systems*, pp. 2787-2795, 2013.

[10] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, Learning entity and relation embeddings for knowledge graph completion, *Proc. of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2181-2187, Austin, TX, 2015.

[11] B. Yang, W. Yih, X. He, J. Gao, and L. Deng, Embedding Entities and Relations for Learning and Inference in Knowledge Bases, *Proc. of ICLR 2015*.

[12] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, Sebastian Riedel, Convolutional 2D Knowledge Graph Embeddings, *Proc. of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 1811-1818, 2018.

[13] Tibshirani, Robert, Guenther Walther, and Trevor Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* Vol. 63, No. 2, pp. 411-423, 2001.



신 원 철

2018년 건양대학교 의료IT공학과(학사)
2019년~현재 숭실대학교 컴퓨터학과(석사과정). 관심분야는 인공지능, 지식 표현 및 추론, 딥러닝

노 재 승

정보과학회논문지
제 47 권 제 6 호 참조

박 영 택

정보과학회논문지
제 47 권 제 4 호 참조