한국어 디비피디아의 자동 스키마 진화를 위한 방법

김선동*, 강민서**, 이재길**
*한국과학기술원 산업 및 시스템공학과
**한국과학기술원 지식서비스공학과
e-mail : {sundong.kim, minseo, jaegil}@kaist.ac.kr

A method of Automatic Schema Evolution on DBpedia Korea

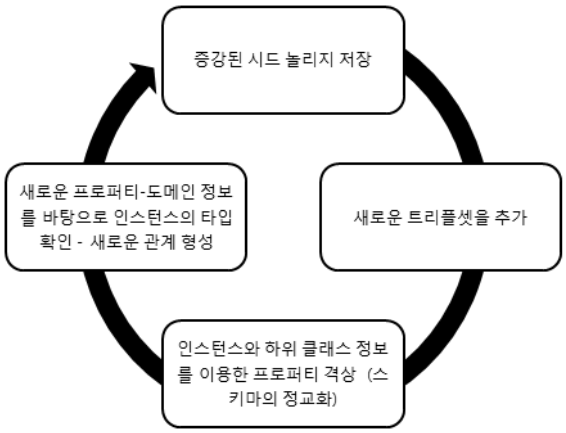
Sundong Kim*, Minseo Kang**, Jae-Gil Lee**
*Dept. of Industrial & Systems Engineering, KAIST
**Dept. of Knowledge Service Engineering, KAIST

요 약

디비피디아 온톨로지는 위키피디아에서 구조화된 데이터를 추출한 지식 베이스이다. 이러한 지식 베이스의 자동 증강은 웹을 구조화하는 속도를 증가시키는데 큰 기여를 할 수 있다. 본 연구에서는 한국어 디비피디아를 기반으로 새로운 트리플을 입력받아 기존의 지식 베이스를 자동 증강시키는 시스템을 소개한다. 스키마를 자동 증강하는 두 가지 알고리즘은 최하위 레벨인 인스턴스가 지닌 프로퍼티, 즉 rdf-triple 단위에서 진행되었다. 알고리즘을 사용한 결과 첫째, 확률적 격상 방법을 통해 단계별로 입력받는 인스턴스와 하위 클래스의 프로퍼티를 이용하여 상위 클래스의 스키마가 정교해졌다. 둘째, 이를 바탕으로 타입 분류가 되어 있지 않았던 인스턴스들이 가장 가까운 타입에 자동 분류되었다. 지식 베이스가 정교해지면서 재분류된 인스턴스와 새로운 트리플셋을 바탕으로 두 가지 알고리즘은 반복적으로 작동하며, 한국어 디비피디아 지식 베이스의 자동 증강을 이루었다.

1. 서 론

본 논문에서는 한국어 디비피디아[1, 6] 지식 베이스를 자동 증강하는 알고리즘을 제안한다. 제안한 알고리즘은 온톨로지 증강을 하는 방법론 중 데이터 중심 증강 방법론에 분류되며, 인스턴스가 지니고 있는 프로퍼티를 기반으로 기존의 스키마를 명확히 하고 전체 지식 베이스를 자동 증강시킨다. 소개하고자 하는 방법은 인스턴스의 타입을 찾아주는 알고리즘과 새로운 프로퍼티의 도메인을 찾아주는 알고리즘 두 가지로 구성되어 있다. 첫 번째 알고리즘은 이미 일정한 지식 베이스를 가지고 있지만, 그에 속한 인스턴스들의 분류가 정확히 되어 있지 않을 때 인스턴스들의 위치를 바로잡아 주는데 사용할 수 있으며, 또는 인스턴스가 서로 다른 여러 가지 타입의 성질을 가지고 있을 때, 기존에 분류된 타입 이외의 타입을 찾아내서 표기해주는 데에 이용할 수 있다. 두 번째 알고리즘은 인스턴스 및 하위 클래스가 가진 프로퍼티의 확률적 격상 및 삭제를 통해 스키마를 정교화하는 방법이다. 같은 방법으로 새로 입력받은 트리플 중에서 프로퍼티의 도메인이 정의되어 있지 않을 때, 해당 프로퍼티를 가진 인스턴스의 비율을 기반으로 프로퍼티의 주 도메인을 찾아주고, 지식 베이스에 새로 생성된 클래스의 정보를 채워주는데도 이용할 수 있다. 본 논문에서는 위 방법론을 한국어 디비피디아 데이터셋에 적용하여 지식 베이스의 자동 증강을 실현하는 기반을 마련하였다. (그림 1)은 소개한 알고리즘을 통해 지식 베이스가 자동으로 증강하는 원리를 보여준다.



(그림 1) 한국어 디비피디아 지식 베이스가 자동 증강하는 원리

2. 관련 연구

온톨로지 증강 방법론은 크게 데이터 중심 증강(data-driven evolution), 구조 중심 증강(structure-driven evolution), 사용 중심 증강(usage-driven evolution)으로 분류된다[2]. 구조 중심 증강론은 온톨로지의 구조를 분석함으로써 더 나은 온톨로지를 만드는 방법이다. 특정 클래스의 하위 클래스가 많다면 하나의 계층을

더 만들어 주어서 하위 클래스의 개수를 조정해주는 방법 역시 구조 중심 증강론에 속한다. 데이터 중심 증강론은 온톨로지의 인스턴스를 분석함으로써 온톨로지를 증강하는 방법론이다. 그 예로는 클래스 A 의 모든 인스턴스가 클래스 B 의 인스턴스에 포함이 된다면 클래스 A 를 클래스 B 의 하위 클래스로 만들어 주는 방법이 있다. 그리고 사용 중심 증강론은 지식 베이스에 접근하는 사용자의 로그를 분석해서 온톨로지를 증강하는 방법이다. 그 예로는 특정 클래스의 하위클래스들이 사용자의 쿼리에 의해 호출되는 빈도수를 근거로 호출 빈도가 낮은 클래스들과 호출 빈도가 높은 클래스들을 따로 그룹을 지어주는 방법이 있다. 이렇게 함으로써 호출 빈도가 높은 하위 클래스들의 그룹을 먼저 검색하고 그 이후에 호출 빈도가 낮은 하위 클래스들이 모인 곳을 검색하는 식으로 필요한 자료에 접근하는 시간을 줄일 수 있다.

현재 구조 중심 증강 방법으로는 휴리스틱 방법과 프로그램 코드 리팩토링(refactoring) 방법[3]이 사용되고 있다. 그리고 데이터 중심 증강 방법으로는 데이터마이닝 방법을 비롯한 휴리스틱 방법이 많이 사용되고 있다[4]. 마지막으로 usage-driven evolution 의 방법으로는 web usage mining 기법이 많이 사용되고 있다[5].

3. 인스턴스들의 타입을 찾아주는 알고리즘

일반적으로 디비피디아의 인스턴스 각각은 일정한 타입에 속해 있고, 타입은 타입의 성질을 정의하는 프로퍼티를 포함한다. 따라서 인스턴스는 해당하는 타입의 프로퍼티 값들을 지니게 되는데, 모든 인스턴스가 하나의 타입에 정확히 속해 있지는 않다. 가령 한국어 디비피디아의 인스턴스 '박정희'[7]의 경우는 대통령이면서 군인의 프로퍼티를 모두 가지고 있다. 혹은 '노홍철'[8]과 같이 '방송인'에 관한 프로퍼티가 존재하지만 타입 정보가 'Person'으로 디비피디아 내에서 정확히 타입이 분류되지 않은 인스턴스도 존재한다. 인스턴스의 타입이 상세히 분류되어 있지 않으면 해당 인스턴스는 온톨로지 증강에 쓰일 수가 없고 따라서 정보의 손실을 가져오게 된다. 본 알고리즘은 타입이 분류가 명확히 되어 있지 않은 인스턴스의 타입 정보를 프로퍼티 분석을 통해서 찾아 주는 역할을 한다.

디비피디아에서 각각의 프로퍼티는 여러 개의 도메인을 삼고 있다. 예를 들면 '이름'은 '국가원수_정보', '인물_정보' 등 다수의 타입에 포함되어 있고, '부통령명칭' 프로퍼티는 '국가원수_정보' 단 하나의 도메인 타입을 가지고 있다. 이와 같이 프로퍼티가 가진 도메인은 각기 다르고, 인스턴스의 프로퍼티는 타입 정보의 영향을 받을 것이라는 가정에서 출발한다. 알고리즘은 한 인스턴스에 포함된 프로퍼티 전체를 대상으로 프로퍼티가 속한 도메인(타입) 빈도수를 카운트하여 가장 많이 등장한 도메인을 인스턴스의 타입으로 정한다. <표 1>에서는 디비피디아 인스턴스인 '김대중'의 프로퍼티 분석을 통해 인스턴스의 타입을 정하는 예를 보여준다. '김대중'은 '이름', '그림', '국가', '출생지', '취임일', '부통령명칭' 등의 프로퍼티를 가지고 있고, 각각의 프로퍼티는 서로 다른 도메인 정보를 가지고 있다. '김대중'의 프로퍼티들이 속한 도메인의 빈도수를 모두 카운트했을 때 '국가원수_정보'라는 도메인이 25 번으로 가장 많이 등장하였고, 그 뒤로 '대통령_정보'가 16 번, '공직자_정보'가 15 번이 등장하였다. 따라서 인스턴스 '김대중'은 '국가원수_정보' 타입에 가장 잘 속한다. 또한 필요에 따라 인스턴스 '김대중'의 메인 타입은 '국가원수_정보' 이고 '대통령_정보', '작가_정보', '공직자_정보' 등을 관련된 타입으로 분류할 수 있다.

<표 1> 디비피디아 인스턴스 ' 김대중' 의 프로퍼티 분석

Property Name	# of Domains	Domains
prop-ko:이름	101	'국가원수_정보', '인물_정보'
prop-ko:그림	61	'예술가_정보', '인물_정보'
prop-ko:국가	34	'대통령_정보', '공직자_정보'
prop-ko:설명	33	'국가원수_정보', '모델_정보'
prop-ko:출생지	31	'국가원수_정보', '군주_정보'
prop-ko:사망일	29	'왕_정보', '국가원수_정보'
prop-ko:출생일	28	'대통령_정보', '군주_정보'
prop-ko:사망지	28	'군주_정보', '인물_정보'
…	…	…
prop-ko:취임일	2	'국가원수_정보', '대통령_정보'
prop-ko:부통령명칭	1	국가원수_정보

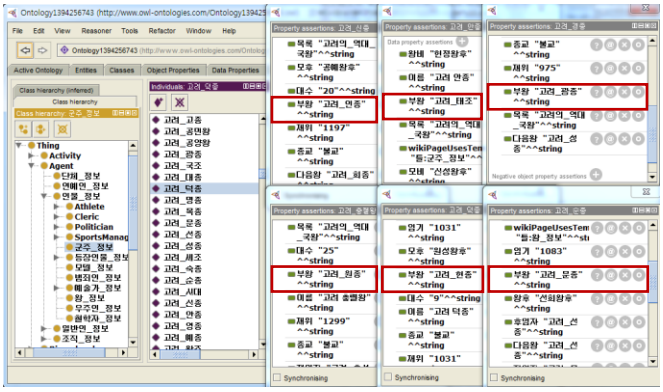
Domain Name	Frequency
'국가원수_정보'	25
'대통령_정보'	16
'작가_정보'	16
'공직자_정보'	15
'정치인_정보'	14
'군주_정보'	14
'왕_정보'	11
'공무원_정보'	9

위 알고리즘은 트리플셋이 들어오는 상황에서 주기적으로 실행하여 아직 분류되지 않은 인스턴스들이 분류될 수 있게 하고, 이미 분류가 완료된 인스턴스들의 관련 타입을 찾아 rdf:type 정보를 풍부하게 해줄 수 있다. 또한 저장된 인스턴스의 프로퍼티를 기반한 bottom-up 방법으로 인스턴스의 타입을 결정하기 때문에, 프로퍼티의 도메인만 미리 정의가 되어 있다면, 디비피디아 온톨로지가 아닌 새로운 온톨로지의 증강에도 적용할 수 있다.

4. 새로운 프로퍼티의 도메인을 찾아주는 알고리즘

앞서서 말한 인스턴스의 타입을 찾아주는 알고리즘이 원활하게 작동하기 위해서는 프로퍼티들의 도메인이 이미 정의되어 있다는 가정이 있어야 한다. 기초 지식 베이스가 존재한다고 하더라도 새로 입력받은 트리플셋에 들어있는 프로퍼티 정보의 경우에는 인스턴스에 특화된 프로퍼티일 수 있고, 따라서 아직 도메인이 없는 상태일 경우가 많다. 이와 같이 아직 도메인이 존재하지 않는 프로퍼티를 인스턴스 각자의 프로퍼티로 설정을 한 후, 해당 클래스에서 인스턴스들이 가지고 있는 프로퍼티를 분석하여 일정 비율 이

상의 인스턴스가 해당 정보를 가지고 있을 경우 클래스 레벨로 격상시켜 그 클래스를 프로퍼티의 도메인으로 삼아준다. (그림 2)을 보면 고려 왕에 속한 인스턴스들이 지식 베이스에 입력이 추가가 되었고, 인스턴스의 'rdf:type'인 '군주_정보' 클래스가 지식 베이스에 추가되었다. '군주_정보'에 속하는 인스턴스의 프로퍼티를 보면 모두 '부왕'이라는 프로퍼티를 공유하고 있는데, 이 때 '부왕'이란 프로퍼티의 도메인을 '군주_정보'로 지정해 주어서 '군주_정보' 스키마의 정교화를 노릴 수 있다. 프로퍼티 격상을 하는 척도로는 1/(1+log n) 와 같은 일정 비율을 설정하여 10 개의 인스턴스 중 5 개, 1000 개의 인스턴스 중 250 개 이상의 인스턴스가 특정 프로퍼티를 가지고 있을 때, 프로퍼티 격상을 진행하도록 하였다. 같은 방법으로 '군주_정보'가 도메인인 프로퍼티를 일정 비율 이하의 인스턴스만 보유하고 있다면 프로퍼티의 도메인을 삭제하는 방법을 사용해 초기 지식 베이스와 인스턴스가 가진 프로퍼티 정보 사이의 불일치를 제거해 주었다.



(그림 2) 프로퍼티 격상이 필요한 예

(그림 3)은 지식 베이스가 자동 증강하는 전체 시스템을 설명한다. 기초 지식 베이스와 트리플을 입력받아 두 가지 알고리즘은 작동하며, 프로퍼티 격상과 삭제, 그리고 인스턴스의 타입 변경은 스키마의 가장 하위 클래스부터 순차적으로 작동한다. 두 가지 알고리즘은 상호 작용하며 증강을 완료한 지식 베이스는 다음 단계에서 새롭게 입력받은 트리플과 함께 재입력된다. 이와 같은 반복 과정을 거쳐 지식 베이스는 자동 증강을 이룬다. (그림 4)와 (그림 5)은 두 가지 알고리즘의 pseudo-code 이다.

```
Overall Procedure
    Create Knowledge base
    While there exists Instance triples to add
        Load Knowledge Base
        Add Instance triple sets
        for all classes from the leaf do
            propertyGeneralization()
            propertyDeletion()
        end for
        for all classes from the leaf do
            findRelatedClass()
        end for
        Save evolved Knowledge base
    end while
```

(그림 3) 한국어 디비피디아 증강 시스템 구성도

```
function findRelatedClass(class)
    for all direct instances of class
        Instantiate Map<Property, Frequency> m
        for all rdf-Properties of instance
            for all domain of property
                Put property-domain pair frequency into m
            end for
        end for

        for all property-domain pair in m
            Aggregate frequency for each domain
            if frequency is the highest
                Change instance domain into class
        end if
    end for
end function
```

(그림 4) 인스턴스의 타입을 찾아주는 알고리즘

```
function propertyGeneralization(class)
    Instantiate Map<Property, Frequency> m
    for all direct instances of class
        for all rdf-Properties of instance
            Put property and its frequency into m
        end for
    end for

    for all Property in m
        if metThreshold() is true
            Add domain class to property
        end if
    end for
end function
// Repeat this procedure for direct subclasses of class
```

(그림 5) 프로퍼티의 도메인을 찾아주는 알고리즘

5. 시스템 구현

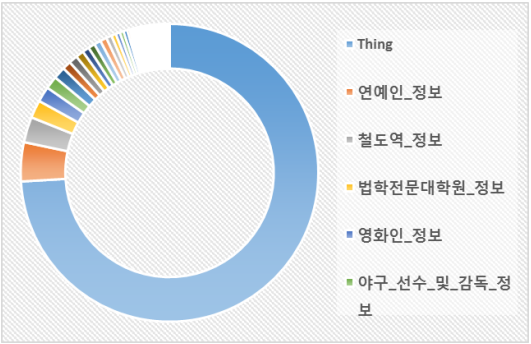
지식 베이스의 자동 증강 알고리즘을 실행하기 위해 Protégé-OWL API 를 이용하여 한국어 디비피디아 지식 베이스를 구현하였다. 지식 베이스의 틀은 영문 디비피디아 온톨로지[9]와 한국어 맵핑 정보[10]에 의거하였다. 인스턴스의 경우 http://ko.dbpedia.org/직업별_조선_사람으로부터 크롤링을 시작하여 30,000 개를 대상으로 하였다. 인스턴스를 지식 베이스에 입력할 때에는 인스턴스 csv 파일(rdf:triple)의 predicate 중에서 인스턴스의 프로퍼티와 타입 정보를 추출하였고, 인스턴스의 초기 타입은 rdf:type 의 정보 중 가장 하위 클래스로 배정해 주었다. 크롤링한 인스턴스 파일을 한국어 디비피디아 스키마에 단계별로 3,000 개씩 추가하는 동시에 가장 하위 클래스부터 순차적으로 프로퍼티 격상 및 삭제, 인스턴스 타입의 재배치를 수행하였다. 새롭게 증강한 지식 베이스는 다음 단계에서 트리플과 함께 다시 사용되었다. 실험은 타입별 프로퍼티 학습을 위한 초기 답안 인스턴스가 없는 상태에서 시작하였으며, 두 가지 증강 알고리즘만을 통해 만들어진 지식 베이스와 원본 디비피디아 지식 베

이스의 인스턴스 - 클래스 구별 정도를 비교하였다.

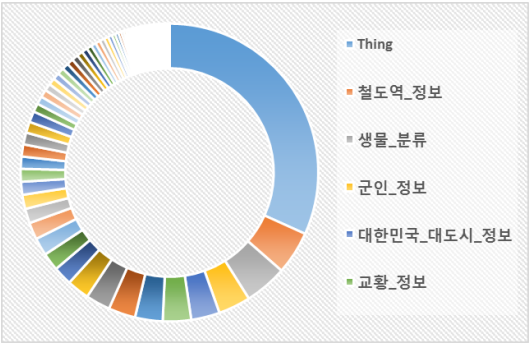
6. 실험 결과

크롤링한 30,000 개의 인스턴스 중 최종적으로 18.305 개의 인스턴스가 지식 베이스에 추가되었고, 미분류된 인스턴스(“Thing”)의 비율이 13,567 개(74%)에서 5,812 개(32%) 로 감소하였다. 프로퍼티 분석을 통해 rdf:type 정보가 입력되지 않은 인스턴스의 타입 분류가 이루어졌음을 알 수 있다. 또한 인스턴스 수가 100 개 이상인 클래스의 비율은 184 개의 클래스 중 14 개에서 35 개로 증가하였다.

(그림 6)과 (그림 7)을 통해 자동 증강 알고리즘과 rdf:type 정보만을 이용한 한국어 디비피디아 인스턴스의 분류 결과를 비교할 수 있다.



(그림 6) rdf:type 정보만을 이용한 한국어 디비피디아 인스턴스의 분류 결과



(그림 7) 자동 증강 알고리즘을 통한 한국어 디비피디아 인스턴스의 분류 결과

7. 결 론

한국어 디비피디아 지식 베이스를 증강하기 위해 인스턴스 단위에서 작용하는 두 가지 알고리즘을 개발하였고, 그에 따라 한국어 디비피디아 지식 베이스를 자동으로 정교화할 수 있는 기반을 마련하였다. 이어서 알고리즘을 이용하여 스키마를 자동 증강하는 알고리즘은 최하위 레벨인 인스턴스가 지닌 프로퍼티, 즉 rdf-triple 단위에서 진행되었고, 이를 통해 지식 베이스에 존재하지 않았던 클래스를 새로 생성하고 해당 클래스의 프로퍼티를 확률적 격상 방법을 통해 추가 혹은 삭제하여 스키마의 정교화를 자동화할 수 있게 되었다. 또한 타입 분류가 되어 있지 않거나 타입이 잘못 분류된 인스턴스는 인스턴스의 프로퍼티를 바탕으로 가장 적합한 타입에 자동 분류가 되었고, 정확한 타입 정보를 가지고 있는 인스턴스의 경우, 관련성이 높은 다른 타입으로의 링크가 생성되었다. 두 가지 알고리즘은 인스턴스 정보에 기반하여 스키마를 정교화하고, 정교화된 스키마를 통해 인스턴스의 타입이 정확해지는 선순환을 가져와 지식 베이스의 자동 증강을 실현할 수 있었다.

8. 사 사

본 연구는 미래창조과학부 및 한국산업기술평가관리원의 SW 컴퓨팅산업원천기술개발사업(SW)의 일환으로 수행하였음. [10044494, WiseKB:빅데이터 이해 기반 자가학습형 지식 베이스 및 추론 기술 개발]

참고문헌

[1] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Kleef, S. Auer, C. Bizer, “DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia,” *Semantic Web 1*, 1-5, 2012.

[2] L. Stojanovic, "Methods and tools for ontology evolution," Ph.D. dissertation, Vrije Universiteit in Amsterdam, Netherlands, 2004.

[3] M. Fowler, “*Refactoring: improving the design of existing code*,” Addison-Wesley, 1999.

[4] I. H. Witten, E. Frank, “*Data Mining: Practical machine learning tools and techniques*,” 2nd ed., Morgan Kaufmann, 2005.

[5] J. Srivastava, R. Cooley, M. Deshpande, P. -N. Tan, "Web usage mining: Discovery and applications of usage patterns from web data," in *ACM SIGKDD Explorations Newsletter* Vol.1, Issue.2, pp.12-23, 2000.

[6] E. K. Kim, M. Weidi, K. S. Choi, S. Auer, “Towards a Korean DBpedia and an approach for complementing the Korean Wikipedia based on DBpedia,” in *Proceedings of the 5th Open Knowledge Conference*, 2010.

[7] 한국어 디비피디아 인스턴스 ‘박정희’ 페이지 [Internet], http://ko.dbpedia.org/page/박정희

[8] 한국어 디비피디아 인스턴스 ‘노홍철’ 페이지 [Internet], http://ko.dbpedia.org/page/노홍철

[9] DBpedia Ontology Classes [Internet], http://mappings.dbpedia.org/server/ontology/classes.

[10] DBpedia Mapping_ko [Internet], http://mappings.dbpedia.org/index.php/Mapping_ko.