RESTful 웹 서비스에서 시맨틱 온톨로지를 구축하기 위한 클러스터링 및 패턴 분석 기법☆

Clustering and Pattern Analysis for Building Semantic Ontologies in RESTful Web Services

이 용 주*
Yong-Ju Lee

요 약

웹 2.0의 등장과 함께 RESTful 웹 서비스의 활용이 전통적인 SOAP 기반 웹 서비스에 비해 크게 증가되고 있다. 최근 웹상에 이용 가능한 RESTful 웹 서비스들의 수가 급격하게 증가됨에 따라 사용자들이 적합한 웹 서비스를 찾는 것은 매우 중요한 이슈로 대두되었다. 그러나 기존의 키워드 기반 검색 방법은 나쁜 재현율과 나쁜 정확률 때문에 문제가 많다. 본 논문에서는 연관규칙 기반 클러스터링 기법에 패턴 기반 시맨틱 분석 기법을 추가한 하나의 새로운 시맨틱 온톨로지 구축 방법을 제안한다. 이를 통해 온톨로지를 자동 구축하여 시맨틱 정보의 주석처리 부담을 줄일 수 있고, 보다 효율적인 웹 서비스 검색을 지원한다. 본 논문에서 제안된 방법은 ProgrammableWeb 사이트로부터 168개의 RESTful 웹 서비스를 다운로드 받아 실험 분석을 수행한 결과, 기존의 키워드 기반 검색 방법에 비해 재현율과 정확률 두 측면에서 각각 35%, 18%의 성능 향상을 보였다.

ABSTRACT

With the advent of Web 2.0, the use of RESTful web services is expected to overtake that of the traditional SOAP-based web services. Recently, the growing number of RESTful web services available on the web raises the challenging issue of how to locate the desired web services. However, the existing keyword searching method is insufficient for the bad recall and the bad precision. In this paper, we propose a novel building semantic ontology method which employs both the clustering technique based on association rules and the semantic analysis technique based on patterns. From this method, we can generate ontologies automatically, reduce the burden of semantic annotations, and support more efficient web services search. We ran our experiments on the subset of 168 RESTful web services downloaded from the PregrammableWeb site. The experimental results show that our method achieves up to 35% improvement for recall performance, and up to 18% for precision performance compared to the existing keyword searching method.

☞ keyword : RESTful 웹 서비스(RESTful web services), 시맨틱 온톨로지(semantic ontologies), 연관규칙(association rules), 클러스터링(clustering), 패턴 분석(pattern analysis), 계층관계(hierarchical relationships)

1. 서 론

최근 웹 2.0의 등장과 함께 OpenAPI(Application Program Interface)와 매쉬업(mashup)이 발전되면서 기존의 SOAP 기반 웹 서비스에 비해 RESTful 웹 서비스의 활용이 크게 증가하고 있다[1]. 구글(Google), 아마존(Amazon), 야후(Yahoo), 이베이(eBay), 네이버(Naver), 그리고 다음(Daum)과 같은 기업에서는 웹 2.0의 새로운 패러다임에 발맞추어 자사의 정보 자원을 OpenAPI를 통해 외부 사용자에게 적극적으로 개방하고 있는 추세이다. 매쉬업이란 이러한 공개된 OpenAPI를 이용하여 두 가지 이상의 서로 다른 자원을 섞어서 완전히 새로운 가치의 서비스를 만드는 것이다. 근래에

* 정 회 원 : 경북대학교 이공대학 컴퓨터정보학부 교수
yongju@knu.ac.kr
[2011/03/07 투고 - 2011/03/14 심사(2011/04/27 2차) - 2011/05/27 심사완료]
☆ 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임 (No. 2010-0008303).

OpenAPI의 구현 형태는 기존의 SOAP 기반 구현 방식에서 비교적 간단하고 가벼운 REST (Representation State Transfer)[2] 방식으로 바뀌고 있다. 최근 동향을 살펴보면, 아마존의 경우 OpenAPI 중 85%가 REST 방식을 따르고 있으며, 구글은 더 이상 SOAP 방식으로는 웹 서비스를 제공하지 않겠다고 선언하였다.

RESTful 웹 서비스란 HTTP와 REST의 원리를 사용하여 구현된 간단한 웹 서비스로 정의된다[1]. RESTful 웹 서비스는 GET, POST 등 HTTP 기본 기능을 적극 활용하고 XML 형식의 메시지 전송에 의해 보다 의미 있는 데이터셋을 생성하기 위하여 다수의 자원으로부터 데이터를 결합할 수 있게 지원한다. 이는 매쉬업과 비슷한 개념으로, 한편으론 매쉬업을 RESTful 웹 서비스의 조합으로 묘사할 수 있다.

하지만 비록 매쉬업이 RESTful 웹 서비스를 결합하기 위한 최적의 기술로 요즈음 각광받고 있지만, 일반 사용자들에겐 특별한 기술적 훈련 없이 값어치 있는 매쉬업을 생성하기란 현실적으로 어려운 일이다. 매쉬업을 만들기 위해서는 프로그래밍 기술뿐만 아니라 매쉬업에 포함되는 모든 서비스들에 대한 API를 파악해야 하기 때문이다. 이에 최근엔 매쉬업을 자동적으로 생성해 줄 수 있는 여러 가지 툴들(예, 야후의 Pipes[3], 마이크로소프트의 Popfly[4], 인텔의 Mashmaker[5])이 개발되었다. 하지만 이들 제품들은 아직까지 키워드(keyword) 기반 웹 서비스 검색만 지원하고, 공개된 서비스 중심이 아닌 자사에서 개발된 내부의 한정된 서비스들만 취급하고 있다.

오늘날 웹상에 이용 가능한 RESTful 웹 서비스들의 수가 급격하게 증가됨에 따라 사용자가 적합한 웹 서비스를 찾는 것은 매우 중요한 이슈로 대두되고 있다. 그러나 정통적인 키워드 기반 검색 방법은 다음의 두 가지 이유 때문에 문제가 있다. (1) 나쁜 재현율(recall): 키워드 기반 검색에서는 키워드가 정확히 일치하는 웹 서비스일 경우에만 발견이 되므로 사용자가 원하는 웹 서비스일지라도 키워드가 일치되지 않는다는 이유로 검색되지 않은 웹 서비스들이 존재한다. (2) 나쁜 정확률(precision): 키워드 기반 검색에서는 검색 결과 중에 사용자는 원하지 않지만 키워드가 포함된 이유로 많은 관련 없는 웹 서비스들이 포함될 수 있다. 따라서 사용자는 이러한 결과 중에서 다시 원하는 웹 서비스들을 찾아야 하는 불편함이 있다.

이러한 키워드 기반 검색 방법의 한계를 극복하기 위한 기법으로서 시맨틱 정보를 이용한 온톨로지(ontology) 활용 방법이 있을 수 있다[6, 7]. 그렇지만 이러한 온톨로지는 대부분 전문가의 수작업으로 구축되고 있으며, 시간 및 인적 제약 때문에 실용적인 온톨로지를 구축하기가 어렵다. 또한, 시맨틱 정보를 위한 추가적인 레이어는 단순한 REST의 취지와도 맞지 않는다. 따라서 본 논문에서는 연관규칙 기반 클러스터링 기법에 패턴 기반 시맨틱 분석 기법을 추가한 하나의 새로운 시맨틱 온톨로지 구축 방법을 제안한다. 이를 통해 온톨로지를 자동 구축하여 시맨틱 정보의 주석처리(annotation) 부담을 줄일 수 있고, 보다 효율적인 웹 서비스 검색을 지원한다. 즉, 키워드가 정확하게 일치하지 않더라도 사용자가 원하는 웹 서비스를 검색할 수 있고, 반대로 키워드가 일치하지만 사용자가 의도하지 않은 웹 서비스는 검색 결과에서 제거된다.

본 논문의 구성은 다음과 같다. 2장에서 본 논문과 연관된 관련 연구들을 살펴보고 3장에서 RESTful 웹 서비스의 기본 개념과 기술언어를 간단히 기술한다. 4장에서 RESTful 시맨틱 온톨로지 구축 방법을 제안하고 5장에서 실험 분석을 수행한다. 그리고 6장에서 결론을 내린다.

2. 관련 연구

온톨로지 구축 방법에 관한 연구는 크게 두 가지 방향으로 추진되고 있다. 첫 번째 접근 방법은 전문가의 수작업으로 웹 서비스 저장소

(registry)에 추가적인 시맨틱 정보를 주석처리하여 온톨로지를 구축하는 방법이다. 전통적인 SOAP 기반 웹 서비스에서는 OWL-S[8], WSMO [9], 그리고 SAWSDL[10] 방법이 있다. OWL-S는 OWL(Web Ontology Language)을 기반으로, 서비스 개요를 기술하는 서비스 프로파일(profile), 서비스 작업 프로세스를 기술하는 서비스 모델(model), 그리고 서비스를 엑세스할 수 있도록 자세한 사항을 기술하고 있는 서비스 그라운딩(grounding)을 기술하고 있다. WSMO(Web Service Modeling Ontology)는 시맨틱 웹 서비스의 다양한 측면을 서술하기 위한 개념 모델로서, 4가지 시맨틱 웹 서비스의 핵심요소(즉, ontologies, goals, web services, mediators)를 온톨로지 형태로 정의한 명세서이다. 그리고 WSDL-S로부터 파생된 SAWSDL(Semantic Annotations for WSDL)은 WSDL(Web Service Description Language)에 시맨틱 개념을 주석처리한 W3C 시맨틱 웹 서비스 권고안이다. 한편, RESTful 웹 서비스에서는 SA-REST[6]와 SBWS(Semantic Bridge for Web Services)[11] 방법이 있다. SA-REST는 SAWSDL과 같은 개념으로 RESTful 웹 서비스에 시맨틱 개념을 주석처리한 것이다. 그러나 SA-REST에는 SAWSDL과는 다르게 WSDL 파일이 없기 때문에 서비스를 묘사하는 HTML 웹 페이지에 직접 주석처리를 첨가한다. SBWS는 WADL(Web Application Description Language) 문서에 주석처리하는데 SAWSDL 방식과 비슷하다. 이러한 시맨틱 정보 주석처리 방법의 문제점은 전문가의 수작업으로 모든 것이 처리되어야 하므로 시간 및 인적 제약으로 인해 온톨로지를 구축하기가 쉽지 않다. 또한 현시점에서 웹 서비스 전체에 대한 주석을 다시 단다는 것은 거의 불가능하게 보인다.

두 번째 접근방법은 수작업으로 온톨로지를 구축하기가 어렵기 때문에 온톨로지 학습(learning) 방법에 의해 자동 구축하는 방법이다. Hess[12]는 웹서비스들을 자동 분류하기 위해 Naive Bayes와 SVM 머신 러닝 방법을 제안하였다. 그렇지만 이 논문에서는 WSDL로부터 추출된 모든 용어(term)들을 단지 단어의 백(bag)으로만 취급할 뿐 계층관계와 같은 시맨틱 개념은 없다. Dong[13]은 연관성이 높은 웹 서비스 매개변수들을 같은 개념으로 묶는 클러스터링 메카니즘을 제안하였다. 이 방법에서는 재현율은 향상시킬 수 있으나 이와 비례하여 원하지 않은 결과도 증가하므로 정확률의 향상은 기대하기 어렵다. Sabou[14]는 온톨로지 학습을 위해 프로그램 소스, 도큐멘테이션, UML 다이어그램 등 다양한 소스를 고려하였고, 자연언어처리 기법을 이용한 웹 서비스 온톨로지 학습 프레임워크를 제안하였다. 그러나 이 논문에서는 WSDL은 취급하지 않았고 웹 서비스 검색에 관한 연구는 없다. Guo[15]는 WSDL로부터 추출된 단어를 백으로 취급하지 않고 이들 간의 상관관계를 고려한 웹 서비스 매칭 방법을 제안하였으나, 이 방법은 실제 웹 서비스 환경에 적용했을 때 단지 오퍼레이션 1:1 부분 매칭에만 적용될 수 있다. 실세계 응용에서 웹 서비스 메타데이터는 이보다 더욱 복잡하고 다양한 구조로 구성되어 있다.

한편, 이러한 온톨로지 학습 방법은 대부분 SOAP 기반 웹 서비스를 위한 방법이었고 RESTful 웹 서비스를 위해서는 이들이 잘 맞지 않는다. 왜냐하면, SOAP 기반 웹 서비스에서는 다양한 오퍼레이션들에 대한 시맨틱 처리가 중요한 반면에, RESTful 웹 서비스에서는 이들 오퍼레이션 대신에 체계적으로 구성된 URI에 대한 HTTP 기본 메소드만 수행되기 때문이다. 예를 들면, SOAP 기반 웹 서비스에서는 고객 정보, 주문 정보 등을 검색, 변경, 삭제하는 오퍼레이션이 각기 존재하여 필요한 기능을 수행할 때 해당 오퍼레이션(예, getOrders( ))을 호출하는 방식으로 일반적인 프로그래밍 개념과 동일한 반면에, RESTful 웹 서비스에서는 총주문(/orders), id를 가지는 특정 주문(/order/{id}), 총고객(/customers), 고객 한 명(/customer/{id})을 모두 리소스로 정의하고, 각

리소스에 URI를 할당한 후 URI에 대해 GET이나 POST 메소드(예, http://korea.com/orders)를 수행한다.

3. RESTful 웹 서비스

3.1 기본 개념

REST는 웹의 창시자 중 한 사람인 Roy Fielding의 박사학위 논문[2]에 의해 소개되었다. 그는 현재의 웹 아키텍처가 웹이 지닌 본래의 설계 우수성을 충분히 활용하지 못하고 있다고 판단하고, 웹의 장점을 최대한 활용할 수 있는 네트워크 기반의 아키텍처를 제안했는데 그것이 바로 REST다. 이런 REST 아키텍처 스타일에 따라 정의되고 이용되는 서비스나 응용을 RESTful 웹 서비스라 한다. RESTful 웹 서비스의 기본 개념은 다음과 같다.

- 리소스의 URI 설정: RESTful 웹 서비스의 가장 큰 특징 중의 하나는 모든 대상을 리소스(resource), 즉, 자원으로 표현한다는 것이다. 이 리소스는 HTTP URI(Uniform Resource Identifier)에 의해 표현되며, 웹 사이트, 블로그, 이미지, 음악, 이용자, 지도, 검색결과 등 웹에서 다른 이들과 공유하고자 개방된 모든 자원을 의미한다.
- HTTP 메소드 사용: REST 구조에서의 리소스는 HTTP의 기본 메소드(method)인 POST, GET, PUT, DELETE만으로 접근할 수 있다. 리소스에 접근하기 위한 이러한 4개의 HTTP 메소드는 일반 CRUD(Create, Read, Update, Delete) 오퍼레이션에 각각 대응될 수 있다.
- 다양한 표현 방식: HTTP의 기본 메소드로 전달되는 리소스는 다양한 방식으로 표현(representation)되는데, 이는 XML, JSON, HTML, 텍스트, 이미지 등이 가능하며 클라이언트에서 원하는 형식으로 표현하면 서버에서 이를 처리하게 된다. 리소스의 다양한 표현 방식은 HTTP의 accept 헤더값 또는 URI 파라미터로 지정할 수 있다.
- 스테이트리스: HTTP의 특성을 상속하여 RESTful 웹 서비스 역시 스테이트리스(stateless) 특성을 가지게 되는데, 스테이트리스란 웹 서비스 제공 서버 측에서 클라이언트의 상태(state) 정보를 저장, 관리하지 않는 것을 의미한다. 즉, 모든 HTTP 요청은 완벽한 고립 상태에서 발생한다. 클라이언트가 HTTP 요청(request)을 할 때 서버에 그 요청을 수행할 수 있는 모든 정보를 주어야 하며 이전의 요청에 의존해서는 안 된다. 이런 특성은 상태를 저장 하지 않으므로 확장성이 좋아지고 캐쉬(cache)하기에 적합한 구조를 만든다.

이상과 같이 RESTful 웹 서비스는 리소스의 URI만 알면 SOAP 기반 웹 서비스와 같은 부가적인 전송 레이어 없이 HTTP 프로토콜만으로 접근 가능한 아주 간단한 서비스라 할 수 있다. 이러한 단순 명료한 접근 방식 때문에 구글, 야후, 트위터 등에서 제공하는 대부분의 웹 2.0 OpenAPI들이 RESTful 웹 서비스로 작성되어 있으며, 이는 위젯(widget)을 이용한 매쉬업을 활성화시킨 원동력이 되었다. 또한, 기존에 제공되던 SOAP 기반 웹 서비스 조차도 RESTful 웹 서비스를 추가 개발하여 동시에 제공하는 추세이다.

3.2 기술 언어

SOAP 기반 웹 서비스에 비해 가볍고 구현하기 쉬운 RESTful 웹 서비스가 많은 주목을 받고 있음에 따라, RESTful 웹 서비스의 인터페이스를 기술하기 위한 다양한 방법들이 제안되고 있다. 현재 RESTful 웹 서비스를 기술하기 위한 여러 가지 언어들이 제안되고 있으나 아직까지 주류는 형성되지 않은 상황이다. 기존의 WSDL에서도 2.0 버전에서는 RESTful 웹 서비스를 기술할 수 있는 HTTP 바인딩(binding) 확장 스펙이 제안

```
 1 <?xml version="1.0"?>
 2 <application xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 3  xsi:schemaLocation="http://wadl.dev.java.net/2009/02 wadl.xsd"
 4  xmlns:tns="urn:yahoo:yn"
 5  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 6  xmlns:yn="urn:yahoo:yn"
 7  xmlns:ya="urn:yahoo:api"
 8  xmlns="http://wadl.dev.java.net/2009/02">
 9   <grammars>
10     <include href="NewsSearchResponse.xsd"/>
11     <include href="Error.xsd"/>
12   </grammars>
13   <resources base="http://news.search.yahoo.com/">
14     <resource path="newsSearch">
15       <method name="GET" id="search">
16         <request>
17           <param name="appId" type="xsd:string" style="query" required="true"/>
18           <param name="query" type="xsd:string" style="query" required="true"/>
19           <param name="type" style="query" default="all">
20             <option value="all"/>
21             <option value="any"/>
22             <option value="phrase"/>
23           </param>
24           <param name="results" style="query" type="xsd:int" default="10"/>
25           <param name="start" style="query" type="xsd:int" default="1"/>
26           <param name="sort" style="query" default="rank">
27             <option value="rank"/>
28             <option value="date"/>
29           </param>
30           <param name="language" style="query" type="xsd:string"/>
31         </request>
32         <response status="200">
33           <representation mediaType="application/xml" element="yn:ResultSet"/>
34         </response>
35         <response status="400">
36           <representation mediaType="application/xml" element="ya:Error"/>
37         </response>
38       </method>
39     </resource>
40   </resources>
41 </application>
```

(그림 1) Yahoo News Search 애플리케이션을 위한 WADL 문서

되었지만 너무 복잡하다는 이유로 많이 쓰이지는 못하고 있다. 이에 썬마이크로시스템은 간략하면서도 범용성이 뛰어난 WADL[16]을 발표하게 되었고, 간단하다는 장점 때문에 개발자들 사이에서 WADL의 사용은 점차 증가되고 있는 실정이다.

WADL은 HTTP 기반 웹 애플리케이션(예를 들면, RESTful 웹 서비스)의 기계 판독이 가능한 기술을 제공하는 XML 기반의 문서 형식이다[16]. WADL의 목적은 보다 쉽게 웹 2.0 형태의 애플리케이션들을 생성하고 동적 서비스 생성 및 관리 방법을 제공하기 위하여, 인터넷 상의 서비스들을 기계가 처리할 수 있는 방법으로 기술할 수 있는 방안을 제공하는 것이다. WADL은 현존하는 웹 구조에 기반 한 애플리케이션들을 위하여 고안되었다. 즉, WSDL처럼 플랫폼과 언어에 독립적이며 웹 브라우저의 기본적인 사용 외에 애플리케이션의 재사용을 촉진시키는 것을 목표로 한다. 또한 WADL에서는 서비스에 의해 제공되는 자원들과 그들 사이의 관계를 모델링할 수 있다.

WADL에서 서비스는 resource 엘리멘트들로 기술되며, 이들 각각에는 request와 response를 기술하는 method 엘리멘터가 있다. request 엘리멘트에는 어떻게 입력을 표현할 것인가를 기술하고 있고, response 엘리멘트에는 서비스 결과의 representation과 상태 정보를 기술하고 있다. 그리고 request와 response에는 매개변수를 나타내는 param 엘리멘트들이 있을 수 있다. 이러한 WADL 문서의 예는 (그림 1)과 같다.

(그림 1)에서 줄 2-8은 application 태그 시작과 XML 네임스페이스 정의를 보여주고 있다. 9-12는 서비스에서 사용되는 XML 문법을 정의하였

는데, 이 경우에는 2개의 W3C XML Schema 파일을 포함한다. 14-39는 Yahoo News Search 웹 리소스와 지원되는 HTTP 메소드를 기술하고 있고, 15-38은 "search" GET 메소드, 16-31은 입력, 32-37은 출력을 기술하고 있다. 여기서, 만일 appId와 query 매개변수 값이 각각 '1234'와 'abc'라면 HTTP GET이 수행될 URI는 다음과 같다.

http://news.search.yahoo.com/newsSearch?appId=1234&query=abc

4. RESTful 시맨틱 온톨로지 구축

RESTful 웹 서비스를 위한 시맨틱 온톨로지의 구축은 많은 이점을 줄 수 있으며, REST 서비스와 관련된 수많은 문제점들을 해결할 수 있다. 비록 기존의 SOAP 기반 웹 서비스에 대한 시맨틱 온톨로지는 OWL-S, WSMO, 그리고 SAWSDL과 같은 많은 플랫폼들이 제안되어 있지만, RESTful 웹 서비스에 대한 시맨틱 온톨로지는 아직까지 구체적인 연구 결과가 없는 상황이다. 이는 REST 본래의 목적이 단순함이었기 때문에 RESTful 웹 서비스에서는 애초에 WSDL과 같은 기술 언어를 요구하지도 않았고, 이러한 기술 언어의 부재는 RESTful 시맨틱 온톨로지의 구축을 힘들게 만들었다.

본 논문에서는 RESTful 웹 서비스의 기술 언어로써 WADL을 채택하였다. 그러나 RESTful 시멘틱 온톨로지를 구축하기 위해 WADL이 꼭 필요한 것은 아니다. 다만, WADL이 온톨로지 구축 자동화에 도움이 줄 수 있는 수단이 될 수 있다. WADL은 WSDL 처럼 구문(syntactic) 정보는 제공하지만, 시맨틱 웹을 위해 설계되지 않았기 때문에 웹 서비스 리소스에 대한 시맨틱 정보를 정의할 수 있는 틀(placeholder)은 제공하지 않는다. 따라서 본 논문에서는 RESTful 시맨틱 온톨로지의 구축을 가능케 하기 위하여 연관규칙 기반 클러스터링 기법에 패턴 기반 시맨틱 분석 기법을 추가한 새로운 시맨틱 온톨로지 구축 방법을 제안한다.

제안하는 시맨틱 온톨로지 구축 방법의 핵심 내용은 RESTful 웹 서비스의 매개변수들에 대해 의미적으로(semantically) 같은 개념들을 묶고(clustering), 매개변수 내에 있는 단어들 간의 계층관계(hierarchical relationship)를 구축하여 단어들 사이에 숨겨져 있는 시맨틱 개념을 활용하는 것이다. 그러나 이러한 온톨로지 구축 작업은 두 가지 이유 때문에 쉬운 일이 아니다. 첫째, RESTful 웹 서비스 매개변수들은 복합단어, 약어, 개발자의 명명(naming) 습관 등으로 인해 매우 다양해 질 수 있다. 따라서 WordNet과 같은 전자 사전을 바로 적용하기 어렵다. 둘째, RESTful 웹 서비스 리소스에서는 일반적으로 매개변수들이 몇 개 존재하지 않으며, 이에 대한 충분한 설명도 거의 제공하고 있지 않다. 따라서 단어 빈도수를 기반으로 하는 TF/IDF(Term Frequency/Inverse Document Frequency)[17]와 같은 전통적인 정보 검색(information retrieval) 기법들은 잘 적용될 수 없다.

본 연구에서는 제안된 시맨틱 온톨로지 구축 방법의 정확성을 향상시키기 위하여 RESTful 웹 서비스 매개변수들에 대해 다음과 같은 전처리 작업을 먼저 수행한다, (1) RESTful 웹 서비스 매개변수들은 일반적으로 다수의 단어가 연결된 복합단어로 이루어져 있으므로(예, ZipCode) 이들에 대한 토큰화(tokenization)가 요구된다. (2) 올바른 매치를 수행하기 위해 단어 스테밍(stemming) 뿐만 아니라 불용어(stop-word) 필터링이 수행된다. (3) 약어(abbreviation)는 완전한 단어로 확장된 후 동의어를 발견하기 위해 시소러스(thesaurus)가 사용된다.

4.1 연관규칙 기반 클러스터링 기법

RESTful 웹 서비스의 매개변수들을 토큰화하여 용어들로 분리한 후, 관련성이 많은 용어들에 대해 클러스터(cluster)를 형성하면 이 클러스터는

각각의 단어가 아닌 하나의 의미 있는 개념(concept)을 나타낸다. 이러한 클러스터는 "매개변수들이 동시에 자주 나타난다면, 그것들은 같은 개념을 나타내는 경향이 있다"는 가정 하에 하나의 특별한 연관규칙(association rules)[13, 18]에 따라 만들어 진다.

연관규칙은 용어 A가 일어나면 용어 B가 일어난다는 의미로 A ⇒ B로 표현될 수 있으며, 여기서 트랜잭션(transaction)은 웹 서비스 입출력에 나타나는 용어들의 집합으로 볼 수 있다. 그리고 지지도(support)와 신뢰도(confidence)는 해당 규칙이 얼마나 유용한지를 나타내는 지표로서, 지지도는 용어 A와 B를 동시에 포함하는 트랜잭션의 확률을 표현하며, 신뢰도는 용어 A가 주어졌을 때 용어 B가 동시에 나타날 트랜잭션의 확률을 나타낸다. 즉,

$$\text{지지도}(A \Rightarrow B) = P(A \cup B)$$

$$\text{신뢰도}(A \Rightarrow B) = \frac{P(A \cup B)}{P(A)}$$

연관규칙을 찾는 과정은 기본적으로 빈발 용어 집합(frequent termset)을 찾는 단계와 연관규칙을 생성하는 두 단계로 구성된다. 빈발 용어 집합이란 후보 용어 집합(candidate termset) 중 최소 지지도(minimum support) 이상의 값을 가진 용어 집합으로서 데이터 마이닝 기법에서 생성되는 연관규칙의 단위가 된다. 빈발 용어들이 생성되고 나면 이들로부터 최소 신뢰도(minimum confidence)를 만족하는 모든 연관규칙들을 찾는다.

(그림 1)의 WADL 문서에서 트랜잭션은 request나 response 내에 있는 param 엘리먼트(예, <param name="appId"/>)의 집합으로 구성된다. 즉, 트랜잭션 T_i = {application, identification, query, type, results, start, sort, language}로 표현된다. 만일 4개의 트랜잭션이 T_1 = {A, C, D}, T_2 = {B, C, E}, T_3 = {A, B, C, E}, T_4 = {B, E}로 구성되어 있다면, 지지도($A \Rightarrow B$)는 20% (=1/4), 신뢰도($A \Rightarrow B$)는 50% (=1/2)로 계산된다. 이상과 같은 연관규칙 탐사 문제는 Apriori[19] 알고리즘을 사용하면 효율적으로 처리될 수 있다. 이 예에서 최소 지지도와 최소 신뢰도를 각각 30%, 90%라고 가정하면 이 알고리즘의 최종 결과는 (표 1)과 같이 된다.

(표 1) 연관규칙 최종 결과

연관규칙	지지도	신뢰도
A ⇒ C	40%	100%
B ⇒ E	60%	100%
E ⇒ B	60%	100%
B, C ⇒ E	40%	100%
C, E ⇒ B	40%	100%

클러스터링 방법에는 크게 계층적 클러스터링과 비계층적 클러스터링 방법이 존재한다[20]. 계층적 클러스터링은 계층적 관계에 따라 군집화를 시키는 방법으로써 또 다시 하향식(top-down) 방법과 상향식(bottom-up) 방법으로 나눌 수 있다. 하향식 방법은 전체 객체를 하나의 클러스터로 보고 상이한 객체 혹은 클러스터를 분리하는 과정을 수행하고, 상향식 방법은 결합 클러스터링(agglomerative clustering)이라고도 하는데 각 개체가 각각의 클러스터를 형성하는 것에서 시작하여 모든 클러스터가 하나로 합쳐질 때까지 근처에 있는 객체 또는 클러스터를 연속적으로 합치는 방법이다. 본 논문에서는 계층적 클러스터링의 한 종류인 계층적 결합 클러스터링 방법을 사용한다.

그러나 기존의 계층적 결합 클러스터링 방법은 대부분 저차원 데이터를 위해 설계되어 있어서, 데이터의 차원이 매우 큰(예, 수천 개의 상이한 용어들을 가지는 텍스트 문서) 경우 차원의 저주(curse of dimensionality) 문제가 대두된다[20]. 이러한 문제를 해결하는 하나의 방법으로써 연관규칙이 적용될 수 있다. 이 아이디어는 연관규칙 탐사 과정에서 발견된 빈발 용어 집합이 초기 클러스터를 의미하기도 한다는 것이다. 즉, 고차원 용어 벡터 공간을 클러스터링하는 대신에

저차원의 빈발 용어 집합만을 클러스터 후보로 간주하는 것이다. 따라서 본 논문에서는 연관규칙 탐사 과정에서 생성된 최종 연관규칙으로부터 먼저 신뢰도를 내림차순으로 정렬한 다음 지지도를 내림차순으로 정렬한 후, 각 단계에서 가장 최상위에 있는 규칙을 조사하여 만일 두 용어가 다른 클러스터에 속하면 이들을 결합한다. 표 1의 예에서는 {B E}, {A C}, {B C E}와 같은 클러스터가 형성된다.

결합하는 과정에서 우리는 가장 이상적인 클러스터를 형성하기 위하여 다음 두 가지 클러스터 평가 특성을 고려한다. (1) 응집력(cohesion): 한 클러스터 내의 용어들과의 결합력, (2) 연관성(correlation): 다른 클러스터 용어들 간의 상호관계, 여기서 클러스터의 응집력은 높게 하고, 클러스터 간의 연관성은 낮게 한다. 하나의 클러스터 Cst_1이 주어졌을 때 *응집력*{Cst_1}은 한 클러스터 내에 서로 밀접하게 연관되어 있는 용어 쌍들의 분포 확률로 정의된다. 예를 들면, {B E}의 경우 2개의 쌍들 중 B ⇒ E와 E ⇒ B가 연관규칙 최종 결과에 포함되어 있으므로

$$응집력\{B\ E\}=\frac{\| 연관규칙\ 최종\ 결과 \|}{\| Cst_1 \| \ \| (Cst_1-1) \|}=\frac{2}{2\times 1}=1$$

이 되고, 비슷하게 응집력{*A C*}=$\frac{1}{2\times 1}=\frac{1}{2}$ 응집력{*B C E*}=$\frac{2}{3\times 2}=\frac{1}{3}$이 된다. 한편, 두 개의 클러스터 Cst_1과 Cst_2가 주어졌을 때 *연관성*{Cst_1}{Cst_2}는 클러스터 간 서로 밀접하게 연관되어 있는 용어 쌍들의 분포 확률로 정의된다. 예를 들면, {B E}, {A C}의 경우 총 8개의 쌍들 (B ⇒ A, B ⇒ C, E ⇒ A, E ⇒ C, A ⇒ B, A ⇒ E, C ⇒ B, C ⇒ E) 중 연관규칙 최종 결과에 포함되어 있는 것이 하나도 없으므로

$$연관성\{B\ E\}\{A\ C\}=\frac{\| 연관규칙\ 최종\ 결과 \|}{2 \| Cst_1 \| \ \| Cst_2 \|}=\frac{0}{2\times 2\times 2}=0$$

이 된다. 비슷하게 연관성{*B E*}{*B C E*}=1/3, 연관성{*A C*}{*B E*}=0, 연관성{*A C*}{*B C E*}=1/12, 연관성{*B C E*}{*B E*}=1/3, 그리고 연관성{*B C E*}{*A C*}=1/12이 된다.

최종적으로, 전체적인 클러스터 Cst의 품질을 측정하기 위한 클러스터링 점수(score)는

$$점수(Cst)=\frac{응집력의\ 평균}{연관성의\ 평균}$$

으로 정의된다. 따라서 예제에 대한

$$점수(Cst)=\frac{(1+1/2+1/3)/3}{(0+1/3+0+1/12+1/3+1/12)/6}=2.2$$

로 계산된다. 이러한 과정에서 우리의 최종 목표는 가장 높은 점수를 갖도록 클러스터를 형성하는 것이므로, 만일 위의 6개 연관성 점수 중 연관성이 가장 높은 {B E}와 {B C E}에 대해 클러스터링하면 새로운 클러스터 {A C}, {B C E}가 생성된다. 이러한 새로운 클러스터가 이전보다 더 높은 점수를 갖는지 조사하기 위해 다시 점수를 계산하면

$$점수(Cst)=\frac{(1/2+1/3)/2}{(1/12+1/12)/2}=5.0$$

으로 되어 이전보다 더 큰 점수를 보인다. 이젠 더 이상 높은 점수를 얻을 수 없으므로 이것을 최적 클러스터로 설정한다.

이상의 연관규칙 기반 클러스터링 기법은 각 용어들 간의 상호 연관성을 이용해 관련된 단어들끼리 클러스터링 함으로써 보다 효과적인 웹 서비스 검색을 가능하게 한다. 그러나 이 기법은 연관성 높은 단어들을 한 클러스터에 묶어 단지

동일한 개념처럼 취급할 뿐 계층관계에 따라 사용자의 요구사항을 정확하게 표현하는 온톨로지 기능은 제공하지 못하고 있다. 예를 들면, companyCode와 countryCode에서 companyCode는 company에 관한 코드이고 countryCode는 country에 관한 코드이므로 이들은 매치되지 말아야 하는데 이 기법에서는 하나의 클러스터로 형성될 수 있다. 이러한 문제점을 해결하기 위해 본 논문에서는 다음의 패턴 기반 시맨틱 분석 기법을 추가로 제안한다.

4.2 패턴 기반 시맨틱 분석 기법

패턴 기반 시맨틱 분석 기법의 주된 목표는 매개변수 내의 각 단어들 사이의 상관관계를 취득하고, 비교되는 단어들이 서로 유사하고 상관관계가 조건에 일치한다면 그 비교를 매치하는 것이다. 이러한 기법은 “사람들이 단어를 조합하여 복합단어로 된 매개변수를 만들 때 일반적으로 비슷한 패턴을 사용한다”는 관찰로부터 시작한다[15, 21]. 일반적으로 사람들이 어떤 한 개념을 표현할 때 다양한 방법들이 있을 수 있지만 공간적인 제약 하에서는 비슷한 패턴을 사용하는 경향이 있다.

RESTful 웹 서비스에서의 이러한 패턴들을 조사하기 위해 본 논문에서는 기존의 OpenAPI 및 매쉬업 저장소인 ProgrammableWeb 사이트[22]로부터 RESTful 웹 서비스 매개변수들을 다운로드 받아 실험 분석을 수행하였다. 이 사이트는 2005년 9월부터 OpenAPI가 등록되기 시작하였으며 본 논문의 실험 시점에 2904개의 OpenAPI가 존재하고 있었으며, 이들 중 REST 방식으로 구현된 웹 서비스는 2142개였다. ProgrammableWeb에 등록된 OpenAPI 중 상위 10개 API가 차지하는 매쉬업 이용률이 약 94%이며 GoogleMaps API가 41%로 절대적인 위치에 있다. 또한 이 사이트에서는 사용자 편의를 위해 54개의 카테고리별로 OpenAPI가 정리되어 있는데, 본 조사에서는 이들 모든 웹 서비스들을 다 이용하지 않고 mapping, travel, weather 카테고리에 있는 168개의 RESTful 웹 서비스에 대해서만 실험을 수행하였다.

수집된 실험 데이터는 총 8209개의 매개변수들로 구성되었으며, 이에 대해 CRFTagger POS (part-of-speech) 형태소 분석기[23]를 적용시킨 결과 다음과 같은 결과가 나타났다.

- 분석 결과 단지 하나의 토큰으로 구성된 매개변수(예, city)가 3574개로 전체의 44%를 차지하였다. 이는 패턴 기반 시맨틱 분석 기법과는 관련 없는 매개변수들이다.
- 명사$_1$+명사$_2$($Noun_1$+$Noun_2$) 형태의 매개변수(예, companyCode)가 2435개로 전체의 30%를 차지하였다. 이는 하나의 토큰 매개변수와 합치면 전체의 74%를 차지하는 가장 많이 나타나는 패턴이다.
- 형용사+명사(Adjective+Noun) 형태의 매개변수(예, availableCredit)가 752개로 전체의 9%를 차지하였다.
- 동사+명사(Verb+Noun) 형태의 매개변수(예, arriveTime), 명사$_1$+명사$_2$+명사$_3$($Noun_1$+$Noun_2$+$Noun_3$) 형태의 매개변수(예, telephoneAreaCode), 명사$_1$+전치사+명사$_2$($Noun_1$+Preposition+$Noun_2$) 형태의 매개변수(예, dateOfBirth)가 각각 608개(7%), 472개(6%), 368개(5%)로 거의 비슷한 분포를 보였다.
- 그 외 26개(0.3%)는 어떠한 패턴을 찾을 수 없는 매개변수들로 나타났다.

본 기법의 첫 번째 단계는 매개변수 내의 각 단어들 사이의 상관관계를 취득하여 그들을 온톨로지에 저장하는 것이다. 시맨틱 웹의 논리적 기반이 되는 온톨로지는 “특정분야에 대한 공유할 수 있는 개념들의 정형화된 기술”로 정의되며, 개념은 그것의 특징을 설명하기 위한 속성(property)과 개념 사이의 상-하위(subclass) 관계로 표현된다. 따라서 본 연구에서는 각 단어들 간의 상관관계로써 속성과 상-하위 관계를 중점적으로

(표 2) 온톨로지 변환 규칙

규칙	패턴	상관관계	예(example)
1	명사$_1$+명사$_2$ (Noun$_1$+Noun$_2$)	매개변수 **propertyOf** 명사$_1$	`companyCode` → `companyCode` **propertyOf** `Company`
2	형용사+명사 (Adjective+Noun)	매개변수 **subClassOf** 명사	`availableCredit` → `availableCredit` **subClassOf** `Credit`
3	동사+명사 (Verb+Noun)	매개변수 **subClassOf** 명사	`arriveTime` → `arriveTime` **subClassOf** `Time`
4	명사$_1$+명사$_2$+명사$_3$ (Noun$_1$+Noun$_2$+Noun$_3$)	매개변수 **propertyOf** 명사$_1$	`telephoneAreaCode` → `telephoneAreaCode` **propertyOf** `Telephone`
5	명사$_1$+전치사+명사$_2$ (Noun$_1$+Preposition+ Noun$_2$)	매개변수 **propertyOf** 명사$_2$	`dateOfBirth` → `dateOfBirth` **propertyOf** `Birth`

취급한다. 위에 서술된 패턴 조사 결과로부터 RESTful 웹 서비스 매개변수에 대한 온톨로지 변환 규칙(transformation rules)은 (표 2)와 같다.

(표 2)와 같은 변환 규칙의 생성은 다음과 같은 관찰로부터 유도되었다.

- 일반적으로 기존의 개념 X와 X에 새로운 특징을 추가하여 생긴 개념 Y 사이에는 상-하위 관계가 성립된다[24]. 즉 X는 Y의 상위 개념이다. 한편 웹 서비스 매개변수에서 '중심어'란 복합단어에서 가장 강조되는 단어를 의미하며, 보통 한 단어로 구성된 중심어가 복합단어보다 더 넓은 개념을 표현한다.
- 명사$_1$+...+명사$_n$의 형태에 있어서 중심어는 주로 명사$_1$로 표현이 되며, 중심어와 전체 단어 사이의 관계는 일종의 소유 개념과 많이 닮았다[25]. 이러한 관찰로부터 두 개념들 간의 사이는 **propertyOf** 관계가 설정될 수 있다.
- 형용사+명사, 동사+명사의 형태에 있어서 중심어는 뒤에 있는 명사로 표현이 되고[26], 복합단어는 중심어로부터 대부분의 시맨틱을 상속받는다. 이로부터 중심어의 개념은 전체 단어를 일반화(generalization)하는 개념으로 규칙이 설정될 수 있다.
- 명사$_1$+전치사+명사$_2$의 형태에서는 중심어가 명사$_2$가 되고, 중심어와 복합단어 간의 관계는 명사$_1$+...+명사$_n$과 같이 **propertyOf** 관계가 설정된다.

위와 같은 관찰로부터 우리는 다음의 온톨로지 변환 규칙을 생성할 수 있다.

● **명사$_1$+...+명사$_n$ 형태:** 규칙-1, 규칙-4, 규칙-5
매개변수가 중심어(명사$_1$)의 속성(property)이 된다. 즉 '매개변수 **propertyOf** 명사$_1$'의 규칙이 생성된다. 예를 들면, companyCode는 규칙-1을 따르며 다음과 같은 규칙이 온톨로지에 첨가된다.

- companyCode → companyCode **propertyOf** Company

하지만 규칙-5의 경우에는 전치사의 역할로부터 중심어가 명사$_2$로 바뀌어 '매개변수 **propertyOf** 명사$_2$'의 규칙이 생성된다. 예를 들면, dateOfBirth는 규칙-5를 따르며 다음과 같은 규칙이 온톨로지에 첨가된다.

- dateOfBirth → dateOfBirth **propertyOf** Birth

● **형용사/동사+명사 형태:** 규칙-2, 규칙-3
매개변수가 중심어(명사)의 자식관계(subclass)

가 된다. 즉 '매개변수 **subClassOf** 명사'의 규칙이 생성된다. 예를 들면, availableCredit는 규칙-2를 따르며 다음과 같은 규칙이 온톨로지에 첨가된다.

- availableCredit → availableCredit **subClassOf** Credit

위와 같은 규칙들을 사용하여 온톨로지가 구축되고 나면, 두 번째 단계는 두 개념 간 매칭을 시키는 것이다. 하나의 온톨로지는 원천(source) 웹 서비스 집합으로부터 취득하고, 다른 온톨로지는 목표(target) 웹 서비스 집합으로부터 취득한다. 그러고 나서 두 개의 온톨로지는 다음 조건을 만족하면 매치된다.

- 어떤 개념이 다른 개념의 속성일 경우
  예를 들면, companyCode **propertyOf** Company
- 어떤 개념이 다른 개념의 자식관계인 경우
  예를 들면, availableCredit **subClassOf** Credit
- 두 개념이 동의어인 경우(시소러스에 의해 발견)
  예를 들면, State **equivalent** Province

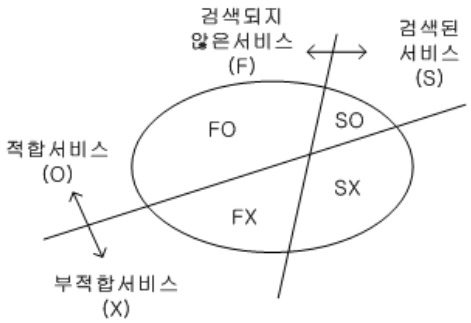
이 알고리즘은 Greedy 방식으로 진행되는데, 만일 두 개의 개념이 조건에 만족되면 유사도 점수는 1이 되고, 만일 두 개의 개념이 조건에 만족되지 않으면 유사도 점수는 0이 되며 이들은 결과로부터 제거된다. 이러한 조건을 적용함에 따라 관련 없는 개념들 사이의 매치를 피할 수 있다. 예를 들면, companyCode와 countryCode는 각각 다른 개념의 속성이므로 매치에서 배제된다. 패턴 기반 시맨틱 분석 기법은 관련 없는 개념들의 매치를 피할 수 있으므로, 매치되는 후보 집합들은 연관규칙 기반 클러스터링 기법에 의해 생성되는 결과보다 더욱 정확한 매치를 얻을 수 있다.

5. 실험 분석

실험 분석의 목적은 본 논문에서 제안하는 연관규칙 기반 클러스터링 기법에 패턴 기반 시맨틱 분석 기법을 추가한 새로운 시맨틱 온톨로지 구축 방법의 우수성을 보이는 것이다. 지금까지 RESTful 웹 서비스에 대한 서비스 검색 방법은 전통적인 키워드 기반 검색만 지원하고 있었다. 따라서 이러한 키워드 기반 검색 방법에 비해 시맨틱 온톨로지 구축 방법에 의한 웹 서비스 발견 방법이 얼마나 효율적으로 수행되는지 이들 두 방법을 비교·분석하였다.

본 논문에서 제안하는 시맨틱 온톨로지 구축 방법은 자바 언어(JDK1.5)로 구현되었으며, 기존에 이미 구현되어 배포되고 있는 오픈 소스(open source) 알고리즘들을 적극 활용하였다. 토큰화, 불용어 필터링 등 데이터 전처리 과정은 OpenNLP[27]를 사용하였고, POS 형태소 분석기는 CRFTagger[23]를 이용하였다. 그리고 연관규칙 및 클러스터링 알고리즘은 각각 Apriori[28]와 ClusterLib[29] 알고리즘을 수정·확장하였다.

평가 방법은 정보 검색에서 가장 보편적으로 이용되고 있는 재현율, 정확률, 그리고 F-척도를 사용한다. 재현율은 사용자의 질의에 적합한 웹 서비스를 얼마나 검색했는지를 나타내며, 정확률은 검색 결과 중에서 사용자 질의에 적합한 웹 서비스들이 얼마나 되는지를 나타낸다. 재현율과 정확률은 모두 높을수록 성능이 좋다고 할 수 있으나, 이들은 서로 반비례의 관계가 있어 한쪽을 높이면 다른 한쪽이 내려가는 것이 보통이다. 그래서 F-척도는 재현율과 정확률을 대체하는 하나의 척도로써 재현율과 정확률의 가중치 조화 평균이다. 재현율(recall) R과 정확률(precision) P, 그리고 F-척도(F-measure) F는 (그림 2)와 같이 계산된다.

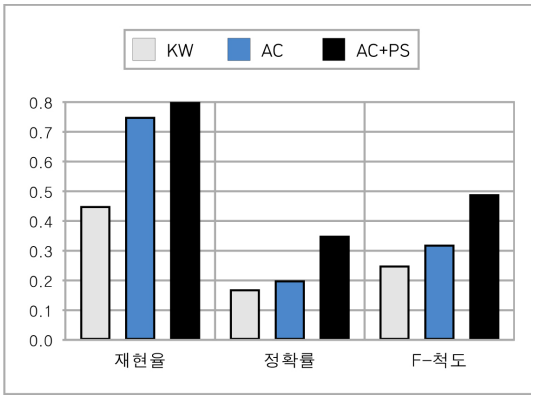


$$R = \frac{SO}{SO+FO} \qquad P = \frac{SO}{SO+SX} \qquad F = \frac{2RP}{R+P}$$

(그림 2) 재현율(R)과 정확률(P), 그리고 F-척도(F)의 계산

실험은 ProgrammableWeb 사이트로부터 다운로드 받은 168개의 RESTful 웹 서비스들에 대해 수행한다. 수집된 데이터는 264개의 request/response 엘리먼트와 전체 501개의 용어로 구성되어 있다. 본 실험에서는 실험 데이터에 대해 최저 지지도와 최저 신뢰도를 변경시켜 가면서 클러스터 결과를 분석하였는데, 그 결과 최저 지지도 3%, 최저 신뢰도 80%에서 관련성 있는 용어들과 관련 없는 용어들을 가장 잘 분리할 수 있는 하나의 적절한 임계값이 될 수 있음을 알 수 있었다. 이로부터 추출된 클러스터와 계층관계 온톨로지에 대한 질의를 수행하여 본 논문에서 제안하는 기법들의 성능을 분석한다. 예를 들어, 제안된 방법에서는 웹 사용자가 zipCode라는 request 매개변수를 사용해 RESTful 웹 서비스들을 검색한다면 zipCode와 관련된 클러스터는 {zip, city, area, state}와 같이 되고, 이와 관련된 계층관계 온톨로지는 zipCode **propertyOf** Zip이 사용된다.

(그림 3)은 기존의 키워드 기반 검색 방법과 비교하여 연관규칙 기반 클러스터링 기법만 사용한 방법, 그리고 연관규칙 기반 클러스터링 기법에 패턴 기반 시맨틱 분석 기법을 추가한 방법이 각각 얼마만큼 성능이 향상되었는지를 보여주고 있다. 여기서 **KW**는 키워드 기반 검색 방법을 의미하고, **AC**는 연관규칙 기반 클러스터링 기법만



(그림 3) 클러스터링과 패턴 분석 기법을 적용한 실험 분석 결과

적용한 검색 방법을 의미하며, **AC+PS**는 연관규칙 기반 클러스터링 기법에 패턴 기반 시맨틱 분석 기법을 추가한 방법을 나타낸다. 이렇게 **AC**와 **PS** 두 기법을 각각 분리하여 성능을 분석한 이유는 각 기법에 대한 성능 향상의 효과를 분석하기 위해서다. 우리는 (그림 3)과 같이 각 방법에 대한 재현율(R), 정확률(P), 그리고 F-척도(F)를 각각 측정하였다.

키워드 기반 검색 방법(**KW**)은 예측된 바와 같이 재현율, 정확률, F-척도 모두 가장 낮으므로 성능은 가장 나쁘다. 연관규칙 기반 클러스터링 기법만 적용 되었을 경우(**AC**)에는 단순 키워드 검색만 사용했을 때보다 재현율은 많이 개선되었으나 정확률은 약간 증가되었다. 이는 클러스터링 검색 결과에서는 적합한 오퍼레이션들이 증가한 만큼 비례적으로 부적합한 서비스들도 증가하기 때문이다. 예를 들면, zipCode는 관련된 클러스터인 {zip, city, area, state}에 있는 모든 용어들에 대해 웹 서비스 검색을 수행하기 때문에 이러한 긍정적인 면과 부정적인 면 두 가지 현상이 동시에 나타날 수 있다. 이는 그림 2에서 SO와 SX 부분이 증가한 것으로 재현율의 결과는 당연히 높아지고, 정확률은 SO와 SX의 영향으로 거의 비슷(조금 상승)하게 되었다. 즉 재현율, 정확률 각각 **KW**에 비해 30%, 3% 향상되었고, F-

척도는 이들 평균 계산에 따라 7% 향상되었다.

다음으로 연관규칙 기반 클러스터링 기법에 패턴 기반 시맨틱 분석 기법을 추가한 경우(**AC+PS**)에는 검색 결과 중 부적합한 웹 서비스들이 급격하게 줄어들어 정확률이 많이 상승하게 된다. 이런 현상을 예를 들어 설명하면, zipCode는 zip의 속성이므로 code만 포함하고 있는 웹 서비스들이 매치에서 배제되기 때문이다. 이는 그림 2에서 SX 부분이 많이 감소되고 FO 부분도 영향을 받아 약간 줄어들어 재현율, 정확률 각각 **AC**에 비해 5%, 15% 향상되었다. F-척도도 이들 변화에 의해 17% 향상되었다. 그림 3에서 F-척도가 전체적인 성능 향상의 효과를 가장 잘 설명하고 있는데, **KW**, **AC**, 그리고 **AC+PS**의 방법이 적용됨에 따라 F-척도의 성능이 향상되고 있음을 보여주고 있다. 결론적으로 실험 분석 결과 본 논문에서 제안한 **AC+PS** 방법이 기존의 **KW** 방법에 비해 재현율, 정확률, F-척도 각각 35%, 18%, 24% 개선된 것을 알 수 있다.

6. 결 론

본 논문에서는 연관규칙 기반 클러스터링 기법에 패턴 기반 시맨틱 분석 기법을 추가한 새로운 시맨틱 온톨로지 구축 방법을 제안하였다. 본 연구의 핵심 내용은 RESTful 매개변수들에 대해 의미적으로 같은 개념들을 클러스터링으로 묶고, 매개변수 내에 있는 각 단어들 간의 계층관계를 형성하여 자동적으로 시맨틱 온톨로지를 구축하는 것이다. 이러한 자동 구축 방법에 따라 기존에 수작업으로 수행되고 있는 온톨로지 구축 작업이 보다 수월하게 진행될 수 있다. 또한, 기존의 SOAP 방식과 같은 복잡한 표준화 과정이 불필요하므로 단순한 REST의 취지와도 부합한다. 제안된 방법은 자바 언어로 구현되었으며, 오픈소스 프로그램인 OpenNLP, CRFTagger, Apriori, ClusterLib 툴들을 적극 활용하였다. 본 논문에서 제안된 방법은 실제 OpenAPI 및 매쉬업 저장소인 ProgrammableWeb 사이트로부터 168개의 RESTful 웹 서비스를 다운로드 받아 실험 분석을 수행하였으며, 그 결과 기존의 키워드 기반 검색 방법에 비해 재현율, 정확률, 그리고 F-척도 각각 35%, 18%, 24%의 성능 향상을 보였다.

향후 연구 과제로는, 본 논문에서 제안되는 패턴 기반 시맨틱 분석 기법은 적용되는 응용 분야에 따라 온톨로지 변환 규칙이 약간 달라질 수 있다. 비록 본 연구에서 제시하고 있는 기본적인 개념과 핵심 규칙에는 큰 변화가 없을 것으로 예상되지만, 다른 문헌상에 나와 있는 구문 패턴 분석 기법이나 통계 기반 패턴 기법과 같은 다양한 기법들에 의한 분석이 추가로 요구된다. 또한, 본 연구에서 표현되는 동일(equivalent), 속성(property), 그리고 상-하위(subclass) 관계 온톨로지 개념은 자연 언어 처리(natural language processing: NLP) 기법을 활용하여 보다 다양한 인과 관계 및 토폴로지(topology) 관계로 일반화시킬 필요가 있다.

참 고 문 헌

[1] L. Richardson and S. Ruby, RESTful Web Services: Web services for the real world, O'Reilly Media, 2007

[2] R. Fielding, Architectural Styles and The Design of Network-based Software Architectures, PhD thesis, University of California, 2000

[3] http://pipes.yahoo.com/pipes

[4] T. Loton, Introduction to Microsoft Popfly, No Programming Required, Lotontech Limited, 2008

[5] http://mashmaker.intel.com/web/learnmore.html

[6] K. Gomadam, A. Ranabahu, and A. Sheth, "SA-REST: Semantic Annotation of Web Resources," http://www.w3.org/Submission/SA-REST/, 2010

[7] O. F. F. Filho and M. A. G. V. Ferreira, "Semantic

Web Services: A RESTful Approach," IADIS International Conference WWW/Internet, 2009

[8] D. Martin, et al., "OWL-S: Semantic Markup for Web Services," http://www.w3.org/Submission/OWL-S/, 2004

[9] J. Bruijin, et al., "Web Service Modeling Ontology (WSMO)," http://www.w3.org/Submission/WSMO/, 2005

[10] J. Farrell, H. Lausen, "Semantic Annotations for WSDL and XML Schema," http://www.w3.org/TR/sawsdl/, 2007

[11] R. Battle and E. Benson, "Bridging the Semantic Web and Web 2.0 with Representational State Transfer (REST)," Journal of Web Semantics, Vol. 6, pp. 61-69, 2008

[12] A. Hess, E. Johnston, and N. Kushmerick, "ASSM: A Tool for Semi-Automatically Annotating Semantic Web Services," Proceedings of the 3rd International Semantic Web Conference, 2004

[13] X. Dong, A. Halevy, J. Madhavan, E. Nemes, and J. Zhang, "Similarity Search for Web Services," Proceedings of VLDB, 2004

[14] M. Sabou, C. Wroe, C. Goble, and H. Stuckenschmidt, "Learning Domain Ontologies for Semantic Web Service Descriptions," Journal of Web Semantics, 3(4), 2005

[15] H. Guo, A. Ivan, R. Akkiraju, and R. Goodwin, "Learning Ontologies to Improve the Quality of Automatic Web Service Matching," Proceedings of IEEE International Conference on Web Services, 2007

[16] M. Hadley, "Web Application Description Language (WADL)," http://www.w3.org/Submission/wadl/, 2009

[17] G. Salton and M. J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, 1983

[18] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," Proceedings of the 1993 ACM-SIGMOD International Conference Management of Data, 1993

[19] R. Agrawal and R. Srikant, "Fast Algorithm for Mining Associations Rules," Proceedings of the 20th VLDB Conference, Santiage, Chile, Sept. 1994

[20] P. Tan, M. Steinbach, and V. Kumar, Introdoction to Data Mining, Addition Wesley, 2005

[21] P. Velardi, P. Fabriani, and M. Missikoff, "Using Text Processing Techniques to Automatically Enrich a Domain Ontology," Proceedings of the ACM International Conference on Formal Ontology in Information Systems, 2001

[22] http://www.programmableweb.com

[23] http://crftagger.sourceforge.net/

[24] 최기선, 류법모, "온톨로지 구축과 학습: 상하위 관계," 정보과학회지, 제24권, 제4호, pp. 24-30, 2006

[25] S. Mokarizadeh, P. Küngas, and M. Matskin, "Ontology Learning for Cost-Effective Large-scale Semantic Annotation of Web Service Interfaces," Proceedings of the 17th International Conference on Knowledge Engineering and Management by the Masses, 2010

[26] 임수연, 송무희, 이상조, "전문용어의 처리에 의한 도메인 온톨로지의 구축," 정보과학회지논문지:소프트웨어 및 응용, 제31권, 제3호, pp. 353-360, 2004

[27] http://opennlp.sourceforge.net/

[28] http://www.cs.uregina.ca/~dbd/cs831/notes/itemsets/dic.java

[29] http://niels.drni.de/s9y/pages/clusterlib.html

◐ 저 자 소 개 ◑

이 용 주(Yong-Ju Lee)

1983년 울산대학교 산업공학과(학사)
1985년 한국과학기술원 산업공학과 정보검색전공(석사)
1997년 한국과학기술원 정보및통신공학과 컴퓨터공학전공(박사)
1985년~1989년 KIST 시스템공학연구소 연구원
1989년~1994년 삼보컴퓨터 근무
1998년~2007년 상주대학교 컴퓨터공학과 부교수
2008년~현재 경북대학교 이공대학 컴퓨터정보학부 교수
관심분야 : 웹 데이터베이스, 정보검색, 공간 데이터베이스
E-mail : yongju@knu.ac.kr