웹 데이터 마이닝을 위한 정보 추출패턴의 기계학습*

김 동석[0] 차 정원 이 근배
포 항 공 과 대 학 교 컴 퓨 터 공 학 과
{dskim,himen,gblee}@postech.ac.kr

Machine Learning of Information Extraction Patterns for Web Data Mining

Dongseok Kim[0] Jeongwon Cha Gary Geunbae Lee
Dept. of CSE, POSTECH

요 약

정보추출 기법을 논의할 때 핵심 역할을 차지하는 것이 추출 패턴(규칙)을 표현하는 종류와 규칙을 만들어 내는 기계학습의 방법이다. 본 논문에서는 mDTD(modified Document Type Definition)라는 새로운 추출패턴을 제안한다. mDTD는 SGML에서 사용되는 DTD를 구문과 해석 방식을 변형하여 일반적인 HTML에서의 정보추출에 활용되도록 설계하였다. 이러한 개념은 DTD가 문서에 나타나는 객체를 지정하는 역할을 하는 것을 역으로 mDTD를 이용하여 문서에 나타는 객체를 식별하는데 사용하는 것이다. mDTD 규칙을 순차기계학습으로 확장시켜서 한국어와 영어로된 인터넷 쇼핑몰 중에서 AV(Audio and Visual product) 도메인에 적용하여 실험하였다. 실험 결과로 정보추출의 평균 정확도은 한국어와 영어에 대해서 각각 91.3%와 81.9%를 얻었다.

1. 서론

일반적인 정의에 따르면 정보추출(IE, Information Extraction)이란 미리 사용되지 않은 텍스트를 입력으로 받아서 데이터베이스 형식과 같이 구조화된 형태의 출력을 산출해내는 기술을 의미한다. 이러한 전통적인 정보추출 기법을 웹 문서를 입력으로 받아서 적용하는 것을 웹 IE라고 한다. 인터넷 정보가 대량으로 양산되면서 웹 IE 기술의 중요성과 필요성이 점증되는 추세이다. 웹 IE의 태스크는 MUC-6(Message Understanding Conference)[6]에서 정의한 것처럼 웹 문서로부터 추출된 텍스트 객체를 다수의 비어있는 슬롯(slot)을 갖는 템플릿(template)을 채우는 작업이다. 템플릿을 구성하는 각각 슬롯의 스펙이 곧 추출목표가 되는 것이다. 웹 데이터 마이닝이란 전통적으로 데이터베이스에서 지식을 추출하는 데이터 마이닝을 인터넷으로부터 지식을 추출하는 개념으로 확장시킨 것이다. 따라서 인터넷으로부터 정보를 추출하여 DB화 시켜주는 웹 IE 기술은 웹 데이터 마이닝에서 핵심적인 역할을 한다. 일반적으로 웹에 나타나는 문서형태를 자유(free)문서, 준구조(semi-structured)문서, 구조(structured)문서로 구분한다. 구조문서란 완전히 테이블 형식으로 구성된 문서를 말하며 자유문서란 신문기사와 같이 형식에 구애됨 없이 자유롭게 기술된 문서를 지칭하고, 준구조문서란 이 둘 사이의 중간 형태를 의미한다. 웹 IE의 문제는 많은 부분문제로 나눠진다. 대표적으로 tagging, 구문 및 의미처리, slot-filling, 대용(anaphora)처리 등이 이에 속한다. 정보추출도 웹 문서를 낮은 단계로 인식한다는 측면에서 자연어처리의 문제와 함께 발생하게 된다. 본 논문에서는 웹 IE에서 가장 기본이 되는 태스크인 slot-filling 문제에 초점을 맞추기로 한다. 대부분의 웹 IE 시스템은 각 도메인 별로 매칭되는 데이터

* 본 연구는 교육부 BK21 project에 의해 수행된 것임.

를 식별하기 위한 추출패턴을 사용한다. 본 논문의 목표중 하나가 새로운 도메인과 태스크에 따라 효율적으로 적응할 수 있는 추출패턴을 제안하는데 있다.

정보추출에서 사용되는 일반적인 방법론은 주석 붙은(annotated)코퍼스로부터 규칙을 만들기 위해 기계학습하고 도메인의 일반 문서를 입력으로 받아서 추출하는 과정을 거친다. 물론, 한계점은 있지만 주석 붙인 코퍼스의 량이 많으면 많을수록 추출성능은 높아지게 마련이다. 여기서 가장 큰 문제점으로 대두되는 것이 숙련된 기술자의 수작업에 의한 주석 붙은 코퍼스를 필요로 한다는 점이다. 즉, 동일한 추출 시스템을 다른 도메인으로 이식할 경우 많은 수작업을 필요로 하기 때문에 이식성(portability)이 떨어지게 된다. 이러한 이식성 과제는 IE에 있어서 본질적인 장벽이고 최근의 연구들은 이식성 문제를 극복하는데 초점이 맞춰지고 있다.

우리의 연구 목적도 웹 IE 태스크에 대한 이식성을 갖춘 시스템을 개발하는 것이다. 수작업을 최대한 경감시켜서 이식성 높은 시스템을 만들기 위해 두 가지 아이디어를 제안한다. 첫째, SGML의 DTD 개념을 확장시켜서 HTML 기반의 웹 문서 정보추출에 적합하도록 mDTD 추출패턴을 도입한다. 두번째는 기계학습에 입력될 예제를 수작업으로 태깅하는 것이 아니고 웹의 구조문서로부터 자동으로 추출하여 도메인 이식성을 높이도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구 사례를 살펴보고, 3장은 mDTD 패턴표현에 대해 기술하고, 4장은 mDTD 규칙을 확장시키기 위한 순차 기계학습 알고리즘을 설명하고, 5장에서는 AV 도메인에 대하여 한국어와 영어에 대한 실험 내용 및 결과를 분석한 다음 6장에서 결론을 맺도록 한다.

2. 관련 연구

정보추출 기술은 영어권을 중심으로 이미 많은 연구가 진행되어 왔다. 채택하고 있는 추출패턴의 형태에 따라서 기존 연구를 살펴보면 대략 세가지로 정도로 구분할 수 있다. 첫째는 정규표현 형식의 규칙을 기반으로 하는 시스템이다. 여기에는 WHISK[10], AutoSlog[9], CRYSTAL[11] 시스템이 포함된다. WHISK는 자유문서에서 구조문서까지 모든 형태의 웹 문서를 대상으로 하고, Covering 알고리즘 계통의 기계학습으로 약간 변형된 정규표현 규칙을 산출한다. 하지만 학습용 예제를 수작업으로 태깅해야 되는 교사(supervised)학습 방법을 사용한다는 단점을 갖는다. AutoSlog와 CRYSTAL은 격(case) 프레임 스타일의 패턴을 갖는다. 프레임의 각각 노드는 명칭과 구문/의미 또는 어휘적 제약을 갖도록 설계되었다. 이러한 시스템의 단점으로는 구문분석기나 의미 태거가 필요하고, WHISK와 마찬가지로 수작업에 의한 태깅이 필요하다는 점이다. 두번째 연구방향은 SRV[4]와 같이 FOL(First-Order Logic) 추출패턴을 사용하는 부류이다. SRV는 top-down 관계형 규칙학습기로써 HTML의 태그 특성에 대한 술어(predicate)는 기본적으로 제공하고 도메인 관련 규칙은 학습을 통해서 산출해낸다. 기계학습은 FOIL[8]과 유사한 방식을 사용한다. SRV도 학습을 위해서는 사람의 수작업으로 태그된 문서를 필요로 한다. 셋째는 웹 기반의 DB 소스로부터 데이터를 추출하고 통합할 필요성에서 제안된 Wrapper Induction 시스템[7]이 있다. 이 시스템은 웹 페이지 대한 wrapper를 귀납 알고리즘으로 구축한다. 추출패턴은 WHISK와 비슷한 delimiter 기반의 규칙을 사용하지만 언어적 제약은 사용하지 않는다. Wrapper induction을 활용하는 시스템은 많으나 대표적으로 finite-state tranducer로 규칙을 산출하는 SoftMealy와 계층적 추출구조를 갖는 STALKER가 있다.

그러나 지금까지 언급된 모든 시스템은 주석달린 코퍼스를 필요로 하는 교사학습 방식에 근간을 두고 있다. 따라서 실제 응용에 적용하려면 도메인 이식성이 어렵다는 문제점을 내포하고 있다. Bootstrapping 방법을 채택한 DIPRE[1] 시스템은 수작업으로 주석을 붙인 소량의 씨앗(seed) 코퍼스에서 학습을 시작해서 다량의 인터넷 문서로부터 유사한 relation을 자동으로 찾아내어 기계학습에 입력시키는 방법론을 사용한다. [12]의 연구도 명칭 분류에 자동 bootstrapping방법을 사용한다는 점에서 이와 비슷한 방법론이라고 볼 수 있겠다.

지금까지 살펴본 기존 연구는 다소의 차이는 있지만 근본은 숙련 작업자의 수작업을 상당히 필요로 한다는

단점을 갖고 있다. 이러한 문제점이 바로 이식성과 직결된다. 본 연구에서는 웹 IE의 이식성 문제를 해결하기 위하여 새로운 추출방법론을 제안한다. mDTD라고 하는 서술적(declarative) DTD 스타일의 추출규칙표현 방법과 수동으로 태그붙인 코퍼스가 필요없는 기계학습 방법을 결합시켜서 웹 IE에 적용한 것이다. 기존 방법과 다른 점은 다음과 같다. 웹 문서의 페이지 구성을 잘 반영할 수 있는 DTD 스타일의 선언적 규칙을 추출패턴으로 사용한다는 점이다. 이렇게 함으로써 기계학습의 입력으로 사용될 예제를 수작업이 아닌 테이블 구조를 갖는 웹 문서로부터 직접 추출하여 사용할 수 있다. 즉, 수작업으로 태그해줄 필요가 없어짐으로써 새로운 도메인으로 신속하게 이식될 수 있는 장점도 갖게 된다.

3. mDTD 패턴 표현

3.1 기본 아이디어

SGML, XML, HTML과 같은 markup 언어에서는 DTD (Document Type Definition)가 사용된다. DTD는 해당 문서에 어떠한 요소들이 속해있는지를 나타낸다. 따라서 SGML 문서를 예로 들면, DTD를 이용하여 문서에 나타나는 요소를 인코딩(encoding)하고, 반대로 문서에 나타난 요소를 파싱(parsing)할 때도 그 문서에 대응되는 DTD를 사용해서 요소를 구분해낸다. mDTD를 제안하는 배경 아이디어도 SGML에서의 DTD 역할과 유사한 것이다. 따라서 mDTD는 추출 도메인의 추출목표가 되는 텍스트 객체와 HTML page 구조의 인코딩/디코딩에 적합하도록 설계했다. 모든 도메인에 대하여 HTML의 테이블 구조와 같이 변화가 없는 내용에 대해서는 기본적으로 씨앗 mDTD로 제공된다. 기계학습의 결과로 추출규칙은 증가하게 되고, 도메인별로 학습된 규칙을 이용하여 준구조 문서로부터 정보를 추출하여 데이터베이스에 입력한다.

3.2 mDTD 정의

mDTD 규칙의 문법과 사용되는 연산자는 그림 1과 같으며, 문맥자유문법(context free grammar)의 표기법과 흡사하다. 완전한 mDTD 규칙은 다음과 같이 다섯개의 요소들로 구성된다.

rule => <keyword name opt (content)occurrence_op action>

여기서 <와 >는 규칙을 선언하기 위한 여닫음 심볼이고, *keyword*는 규칙의 타입을 지정한다. 예를 들어서 *keyword*가 TARGET이면 추출목표가 되는 객체를 지정하는 규칙이고, ELEMENT는 일반적인 텍스트 객체를 지정하게 된다. 하나의 TARGET 규칙이 만족되면 추출목표가 되는 하나의 프레임이 채워진 것으로 판단되는 것이다. *name*은 해당 규칙의 식별자가 되며, *opt*은 규칙에 조건이 붙을 경우에 사용된다. 규칙들은 content로 다시 쓰여질 수 있다. content는 다른 규칙의 name으로 구성되며, occurrence_op는 발생 빈도를 나타내는 연산자로 content와 action에 대하여 적용된다. action은 규칙에서 필요로 하는 파라메타나 참조를 표현한다. 각 연산자와 구체적인 문법은 그림 1에 제시하였다.

3.3 씨앗(seed) mDTD

본 논문에서 제안하는 아이디어중 하나가 수작업으로 태그 붙인 코퍼스가 기계학습시 필요없다는 점이다. 이것을 가능하게 해주는 것이 씨앗 mDTD의 역할중에 하나이다. 씨앗이 되는 mDTD는 두 부분으로 나눌 수 있다. 하나는 모든 도메인에 공통적으로 적용될 수 있는 규칙이다. 주로 HTML 문서의 태그 구조를 반영한 것으로 도메인과 상관없이 어느 웹 페이지든지 동일한 형태를 갖기 때문이다. 도메인에 의존적일 수 있는 부분은 추출목표를 해당 도메인에서 일반적으로 어떻게 표현하느냐에 따라서 차이가 난다. 전자상거래 분야를 예로 들어본다. 추출목표가 {상품명, 모델명, 가격, 제조회사, 크기, 특징}이라고 한다면 본 논문에서 실험 대상으로 하고 있는 AV 도메인이든 아니면 다른 전자제품 도메인 이든 추출목표가 같으면 추출객체를 지칭하는 대표명이 같기 때문에 씨앗 mDTD는 이정도의 도메인이 바뀌더라도 대폭적인 수정이 필요 없게 된다. 동일한 전자상거래 도메인이라 하더라도 추출목표가 전혀 다르게 바뀌면 씨앗 mDTD도 변화에 상응하게 수정해야된다.

```
rule=><keyword name opt (content) occurence_op action>
        <  = rule open delimiter
        > = rule close delimiter
keyword =>TARGET | ELEMENT | INSTANCE | ENTITY
        target = represent the extraction target
        element = represent the element
        instance = represent the instance of object
        entity = represent the lexical data
opt =>O | L
        O = omissible tag ( start-tag or end-tag )
        L = learnable content
content =>sym_cnt | cnt_name | cnt_sequence
sym_cnt =>PCDATA | SDATA
cnt_name =>name | name connect_op name
cnt_sequence =>cnt_name connect_op connect_name
        PCDATA = raw data (parsed character data)
        SDATA = symbolic data
occurence_op =>null | ? | * | +
        ? = element appears once or not
        * = repeatable - zero or more
        + = repeatable - once or more
connect_op =>, | & | |
        , = sequenal order
        & = both appears in any order
        | = any one of the elements must appear
action =>null | action_para name
action_para =># | %
        # = reference specifier
        % = parameter specifier
```

그림 1. mDTD 규칙의 문법과 연산자

```
<TARGET ItemKind - - (tItem | vItem | aItem)>
<ELEMENT tblHtml - - (startHtag, endHtag)+ >
<ELEMENT startHtag - O (stblTag | startHrcTag)>
<ELEMENT startHrcTag - O (stblRow | stblCell)>
<ELEMENT endHtag - O (etblTag | endHrcTag)>
<ELEMENT endHrcTag - O (etblRow | etblCell)>
<ENTITY sTblTag SDATA "<table>" >
<ELEMENT RefName - O ((startHecTag)*,nDic,
  (endHrcTag)*)>
<ELEMENT nDic - O (nrDic| mrDic| prDic| srDic| crDic)>
<TARGET tItem - - ((startHtag)*, Tv, (endHtag)*)>
<TARGET Tv - - (tName & tModel & tPrice)|(tManuf
  |tSpec|tSize)*>
<ELEMENT tName - - (tName1 | tName2 | tName3)+ >
<ELEMENT tName1 - O (tNameInst)>
<ELEMENT tName2 - O ((tNameInst)&tModel)>
<ELEMENT tName3 - O (tnDic, (sCooccur)*, tNameInst)
  >
<ENTITY tNameInst - L SYSTEM tName.dtd %nrDic>
<ENTITY sCooccur SDATA ": " >
```

그림 2. AV 도메인에 대한 씨앗 mDTD 규칙

그림 2에 AV 도메인에 적합하도록 작성된 씨앗 mDTD 중 일부를 제시하였다. 상단부분이 HTML 문서에서 table 구조의 태그를 표현하는 규칙들이다. 이러한 부류에 속하는 아래 규칙를 설명하면 *tblHtml* 객체 (table 객체를 의미)는 *startHtag*와 *endHtag*가 한번이상 나타나는 것으로 구성됨을 의미한다.

<ELEMENT tblHtml - - (startHtag, endHtag)+ >

최종적으로 실제 태그의 이름은 심볼릭 데이터(SDATA)로 표현된다. 추출타깃을 표현하는 규칙은 다음과 같다.

<TARGET Tv - - (tName & tModel & tPrice)|(tManuf |tSpec|tSize)>*

위의 규칙은 Tv 객체를 추출하기 위해서는 반드시 제품명(*tName*), 모델명(*tModel*), 가격(*tPrice*)이 추출되고, 제조회사(*tManuf*), 특징(*tSpec*), 크기(*tSize*)는 각각 하나의 객체로 나올 수도 있고 그렇지 않을 수도 있음을 의미한다.

지금까지 살펴본 씨앗 mDTD는 웹 로봇이 모집해온 도메인의 구조문서로부터 데이터를 추출하는데 사용되며 수작업으로 작성된다. 여기서 추출된 데이터가 간단한 휴리스틱의 검증과정을 거쳐 기계학습의 긍정(positive)입력과 부정(negative)입력이 된다.

4. mDTD 순차 기계학습

본 논문에서 활용한 기계학습은 순차적 전략에 기반한 귀납적(inductive) 학습 알고리즘이다. 즉, 하나의 규칙을 학습한 다음 해당 규칙으로 커버가 되는 입력

데이터를 모두 삭제한다. 이와 같은 과정을 one-rule-learning-discarding 절차라고 한다. 이러한 절차를 입력된 데이터가 모두 커버(cover)될 때까지 반복시켜서 학습하는 방법을 sequential covering 알고리즘이라고 한다. 본 논문은 위의 알고리즘 계열에 속하는 CN2[3] 알고리즘을 변형한 SmL(Sequential mDTD Learner)을 학습기로 개발하였다. 전체 알고리즘은 그림 3에 제시하였다.

CN2에서는 기계학습 결과로 산출된 규칙의 성능척도(performance measure)로 정보이득(information gain)을 사용하는데 이것은 FOIL과 동일한 것이다. 하지만 SmL에서는 규칙의 성능척도로 가장 많은 입력 예제를 커버하는 비율과 어휘 유사도(similarity)로 결정하였다. 이렇게 결정한 이유는 본 논문에서 상정하고 있는 주요 대상 도메인인 전자상거래 분야에서 발생하는 데이터의 특성이 대부분 유사성을 갖고 있기 때문이다.

```
SmLearning( all_examples )
  let   P = part_of_speech_tagger( all_examples )
  let   rule_set = {}

  until P is empty do
    generate class by SmLForOneClass( P )
    find common part of speech tag sequence, postag, from
class
    transform class and postag into mDTD rules
    add rules to rule_set
    remove from P all examples covered by class
  return   rule_set

SmLForOneClass( P )
  let   max_class = {}
  let   n  =  | P|
  for ( n by n )
    class = FindMaxSimilarity(P), find max similarity count
    max_class = MAX( max_class, class)
  return max_class
```

그림 3. SmL 학습 알고리즘

학습기에 입력되는 예제는 씨앗 규칙을 이용하여 구조문서로부터 추출된 객체를 사용한다. 하지만 객체들이 모두 정확한 것이 아니기 때문에 간단한 휴리스틱을 이용하여 추출된 예제가 기계학습 규칙에 커버될 예제(positive example)인지 제외시켜야될 예제(negative example)인지를 판단하여 기계학습 알고리즘에 입력된다. 학습 결과로 산출되는 추출패턴은 도메인 mDTD에 추가되는데 하나의 결과로 연결결과와 함께 두개의 규칙 노드가 발생된다.

```
<!ELEMENT mInst1 - - mInstsymbol1>
<!ENTITY mInstsymbol1 - - SDATA "DTQ-">
<!ELEMENT mInst2 - - mInstsymbol2>
<!ENTITY mInstsymbol2 - - SDATA "CN-">
<!ELEMENT mInst3 - - mInstsymbol3>
<!ENTITY mInstsymbol3 - - SDATA "CT-">
<!ELEMENT mInst4 - - mInstsymbol4>
<!ENTITY mInstsymbol4 - - SDATA "CK-">
<!ENTITY sInstsymbol1 - - SDATA "Stereo and Multilingual">
<!ELEMENT sInst2 - - sInstsymbol2>
<!ENTITY sInstsymbol2 - - SDATA "English Caption">
<!ELEMENT sInst3 - - sInstsymbol3>
<!ENTITY sInstsymbol3 - - SDATA "Sony FlatTV Trinitron">
<!ELEMENT nInst1 - - nInstsymbol1>
<!ENTITY nInstsymbol1 - - SDATA "SamsungTV">
<!ELEMENT nInst2 - - nInstsymbol2>
<!ENTITY nInstsymbol2 - - SDATA "AnamTV">
<!ELEMENT nInst3 - - nInstsymbol3>
<!ENTITY nInstsymbol3 - - SDATA "Samsung Minimini">
<!ELEMENT nInst4 - - nInstsymbol4>
<!ENTITY nInstsymbol4 - - SDATA "LG TV">
<!ELEMENT nInst5 - - nInstsymbol5>
<!ENTITY nInstsymbol5 - - SDATA "DaewooTV">
<!ELEMENT nInst6 - - nInstsymbol6>
<!ENTITY nInstsymbol6 - - SDATA "AIWA Walkman">
<!ENTITY cInstsymbol1 - - SDATA "Samsung">
<!ELEMENT cInst2 - - cInstsymbol2>
<!ENTITY cInstsymbol2 - - SDATA "Anam">
<!ELEMENT cInst3 - - cInstsymbol3>
<!ENTITY cInstsymbol3 - - SDATA "LG">
<!ELEMENT cInst4 - - cInstsymbol4>
<!ENTITY cInstsymbol4 - - SDATA "Anam Electronics">
<!ELEMENT cInst5 - - cInstsymbol5>
<!ENTITY cInstsymbol5 - - SDATA "LG Electronics">
```

그림 4. SmL 기계학습 결과로 산출된 규칙

AV 도메인에 대하여 SmL 기계학습기로 실행시킨 결과로 산출된 규칙 예제를 그림 4에 제시하였다. SmL 알고리즘은 추출목표가 되는 템플릿의 각 필드별로 학습을 수행하도록 되어있다. 학습된 결과에서 mInst로 분류되는 것은 제품의 모델명을 의미하게 되고 cInst는 제조회사를, nInst는 제품의 명칭에 해당되는 규칙이다. 가격 필드 같은 경우는 거의 모든 데이터가 숫자로 표현되어 있고 아주 일부만 문자열이므로 학습 결과도 많은 규칙이 생성되지는 않는다. 학습된 결과로 산출된 규칙들은

심볼릭 데이터 노드를 표현하는 SDATA로 표현되며 규칙의 타입은 ENTITY를 갖는다. 학습된 규칙의 수는 입력되는 구조문서의 양에 따라서 가변적이지만 대략 60개의 문서가 초과되면 학습된 패턴수의 증가량이 미약함을 실험으로 확인할 수 있었다.

5. 실험 및 결과

5.1 실험 시스템 구현

웹 전자상거래 도메인을 대상으로 mDTD와 SmL 학습기를 적용한 실험 시스템을 구현하였다. 시스템 전체 구성도는 그림 5와 같다. 시스템은 학습단계(learning phase)와 추출단계(extracting phase)로 구분된다. 학습단계는 웹로봇, 웹 문서 전처리, 예제 추출, POS(Part-of-Speech) 태깅 및 SmL 모듈로 구성된다. 웹 로봇은 씨앗 URL 리스트로부터 해당 도메인의 구조문서를 모집해온다. 모집된 HTML 문서를 웹 문서 전처리 모듈에서 토큰 형태로 출력해준다. 여기서 추출작업과 관련없는 HTML 태그를 제거하고, 띄어쓰기와 같은 문장 오류를 수정하고, 예제 추출 모듈에서 필요로 하는 토큰 형태로 출력해준다. 예제추출 모듈에서는 특정 도메인용으로 작성된 씨앗 mDTD를 기반으로 기계학습에 입력될 예제를 추출한다. 추출된 예제는 기계학습에 입력되기 전에 POS 태깅과정을 거친다. 태거는 포항공대 자연어처리연구실에서 자체 개발한 한국어용 POSTAG를 사용했고 영어는 태그정보를 사용하지 않았다. 학습단계에서 출력되는 최종 결과는 씨앗 mDTD규칙에 학습된 규칙이 추가된 도메인용 추출규칙이다.

학습단계에서와 마찬가지로 추출단계에서도 웹 로봇으로 도메인과 관련된 추출대상이 되는 문서를 모집해온다. 여기에는 구조문서와 준구조 문서를 모두 포함하고 있다. 모집된 문서는 웹 문서 전처리 모듈에서 토큰 단위로 전처리된다. 토큰은 POS 태깅된 다음 학습단계에서 학습된 mDTD 규칙으로 매칭 테스트를 거친다. 반복적으로 매칭 실험을 거쳐 최종적으로 TARGET 규칙이 매칭되면 하나의 템플릿이 산출되는 것이다. 템플릿의 슬롯에 채워진 데이타는 실제로 캐릭터 스트링 포맷을 갖는다. 이것을 DB 형식에 맞도록 DB 엔트리 입력 모듈에서 변환시킨 후 DB에 추가된다.

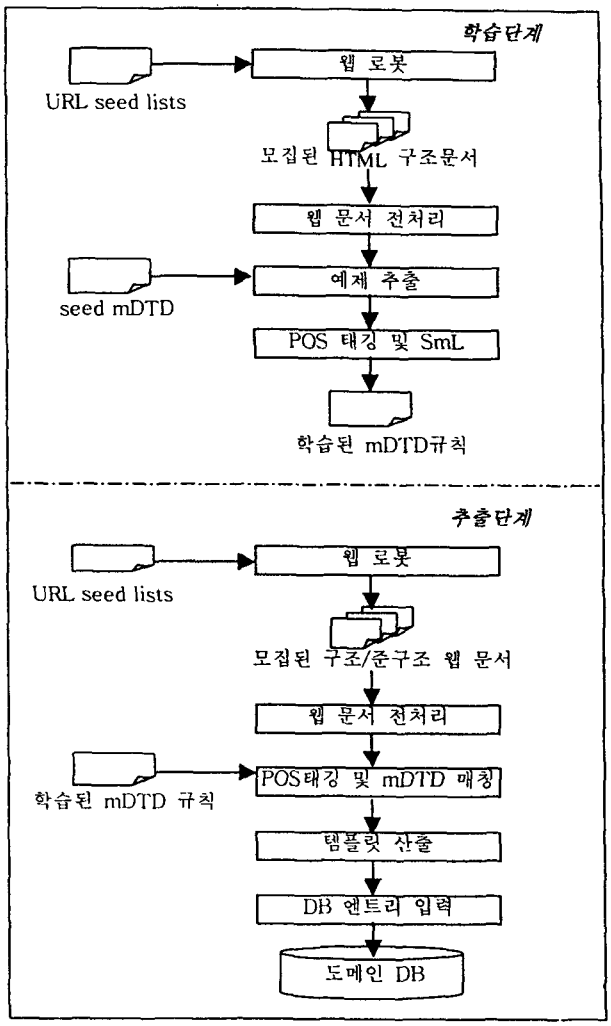

그림 5. 추출 시스템 전체 구성도

이러한 과정을 거쳐서 구축된 데이터베이스는 많은 응용에서 활용될 수 있다. 예를 들어, 웹 쇼핑 정보에 대한 자연어 질의 시스템과 결합해서 사용될 수도 있고, 다양한 웹 쇼핑 사이트의 제품과 가격을 비교하는 서비스를 제공하는 경우에는 back-end에서 웹 마이닝 시스템으로도 활용될 수 있다.

5.2 실험 내용

본 논문에서 제안한 내용의 효용성을 검증하기 위

하여 AV(Audio and Visual Product) 도메인에 대하여 실험을 실행하였다. 추출 목표는 그림 6에서 제시한 것과 같이 여섯개의 아이템을 추출하도록 되어 있다. 그림 6에서 제시한 슬롯 이름에는 영문 이름과 한글 이름이 함께되어 있는데 이것은 영어와 한국어에 대하여 동일한 추출목표를 가지고 실험했기 때문이다. 대상이 되는 사이트와 모집된 문서에 관한 내용을 표1에 제시하였다. 표1에 나타난 구조문서는 모두 기계학습에 쓰여진 문서들이다.

슬랏 이름	내용
제품명(item)	LG tv
제조사(manufacturer)	LG Electronics
모델명(model)	CN-1380T
가격(price)	980,000
스펙(specification)	Fullflat and Superflat Tube
크기(size)	29 "

그림 6. 추출내용으로 채워진 템플릿 예제

	사이트 수	구조문서수	준구조문서수
한국어	50	70	130
영어	30	90	120
전체	80	160	250

표 1. 실험을 위한 웹 문서 상세

실험 결과를 평가하기 위하여 재현율(recall)과 정확도(precision) 기반의 MUC 표준 평가방법을 채택했다[2]. 계산식은 다음과 같다.

재현율 = (출력 템플릿중 정답 슬랏수) /
(총 정답 슬랏수)

정확도 = (출력 템플릿중 정답 슬랏수) /
(출력 템플릿중 채워진 슬랏수)

총 정답 슬롯수는 평가를 위해서 수작업으로 채워진 템플릿의 총 슬롯수를 의미한다. 위에서 정의한 재현율-정확도 척도를 이용하여 테스트 문서집합으로부터 올바르게 추출한 적합정보의 총량을 측정 (재현율로 표현됨) 할 수 있고, 추출된 정보의 신뢰성(reliability)도 (정확도로 표현됨) 측정할 수 있다. 또 다른 평가척도로 F-척도가 있다. 이 평가 방법은 재현율과 정확도의 조화평균(harmonic mean) 값으로 정의된다. 재현율과 정확도에 동일한 가중치를 부여했을 때 F-척도의 계산은 다음과 같다[8].

F-척도 = (2*재현율*정확도) / (재현율+ 정확도)

이러한 평가방법으로 표1에 나타낸 것처럼 한국어 200 문서, 영어 210 문서에 대하여 추출 실험을 하였다.

5.3 실험 결과

한국어와 영어 모두 실험 결과를 템플릿의 각각 슬롯별로 평가하였다. 왜냐하면 슬롯의 특성에 따라서 추출 성능에 차이가 많이 나기 때문에 모두 통합해서 평가하는 것은 의미가 없을 것으로 판단했기 때문이다. 표2에 전체 평가 결과를 제시하였다. 표에서 보는 바와 같이 각 추출목표에 따라서 성능 차이가 많다. 예를 들어서 가격 슬롯은 98.3%, 99.5%의 높은 재현율과 정확도를 보인 반면에, 스펙은 평균 70%대의 저조한 결과를 보인다. 가격과 같은 경우는 숫자로 구성되어 있고 가격임을 확인할 수 있는 정보(한국어의 경우는 '원', 영어의 경우는 '$')가 보편적으로 정확하기 때문이다. 하지만 스펙의 경우는 자연어로 기술되기 때문에 그만큼 난이도가 높다고 하겠다. 모델명의 경우는 한국어와 영어에서 모두 높은 성능을 보였다. 모델명은 대부분 영문과 숫자의 결합이라는 일정한 형식을 갖고 있기 때문에 규칙에 잘 매칭되고, 대부분 제조회사마다 비슷한 모델명을 사용하고 있어서 추출성능이 높게 나오는 것으로 판단된다. 한국어와 영어를 모두 포함한 전체 평균 정확도에서 86.6%를 얻음으로써 본 논문에서 제안한 아이디어가 충분히 활용성이 있음을 확인할 수 있었다. 이것은 boosting 방식을 이용하여 영어권에서 높은 성능을 얻은 BWI[5] 시스템과 비교해도 비슷한 추출목표에서 일부는 우수한 성능을 얻은 것이다.

6. 결론

웹 IE 시스템의 도메인 이식성 문제를 해결하기 위해 본 논문에서는 mDTD 추출패턴 개념을 도입하였고 아울러 순차 기계학습 방법을 적용하였다. 이식성 문제를 해결하기 위하여 최근의 연구 동향을 살펴보면 소량의 수작업 코퍼스만을 필요로하는 제한된 비교사학습 방법이 도입되고 있다. 여기에는 active learning, boosting, boostrapping 등의 기계학습 방법이 주류를

이룬다. 하지만 본 논문은 수작업으로 태그붙인 코퍼스를 필요로 하지 않는 안정적인 방법론을 제안했다. mDTD라는 서술적 규칙 표현 방법과 구조문서로부터 자동으로 학습예제를 추출하여 순차적인 기계학습방법으로 도메인 mDTD 규칙을 학습해내는 방법을 제시하였다. 또한 한국어와 영어에 대한 웹 전자상거래정보를 대상으로 실험을 하여 boosting 방식을 사용한 최근의 외국 연구와도 견줄만한 86.6%라는 평균 정확도를 얻을 수 있었다.

Document	slot name								
	Item			Manufacturer			Model		
	R	P	F	R	P	F	R	P	F
Korean	83.2	82.1	82.6	88.6	88.3	88.5	93.7	96.9	95.3
English	72.3	73.5	72.9	83.1	77.9	80.4	89.5	86.2	87.8
Average	77.7	77.8	77.8	85.9	83.1	84.5	91.6	91.6	91.6

slot name									Average		
Price			Specification			Size					
R	P	F	R	P	F	R	P	F	R	P	F
98.3	99.5	98.9	70.2	81.7	75.5	75.6	99.7	85.9	84.9	91.3	87.9
89.8	93.1	91.4	43.6	69.5	53.6	72.1	90.9	80.4	75.1	81.9	78.4
94.1	96.3	95.2	56.9	75.6	64.6	73.9	95.3	83.2	80.0	86.6	83.2

표 2. AV 도메인에 대한 정보추출 성능 (단, 영어는 태그정보를 사용하지 않음)

7. 참고 문헌

[1]S. Brin, "Extracting Patterns and Relations from the World Wide Web", Proc. of International Workshop on the Web and Databases, 1998.

[2]C. Cardie, "Empirical Methods in Information Extraction", AI magazine Vol.18, No.4 , Winter 1997.

[3]P. Clark and T. Niblett. "The CN2 Induction Algorithm, Machine Learning", Vol.3, 1989.

[4]D. Freitag, "Information Extraction from HTML: Application of a General Machine Learning Approach", Proc. of the 15th Conference on AI, AAAI-98, 1998.

[5]D. Freitag and N. Kushmerick, "Boosted Wrapper Induction", 14th European Conference on AI, 2000.

[6]R. Grishman and B. Sundheim, "Message Understanding Conference-6: A Brief History", Proc. of the 6th Message Understanding Conference, 1995.

[7]N. Kushmerick, "Wrapper Induction for Information Extraction", Ph.D thesis, Univ. of Washington, 1997.

[8]U. Nahm and R. J. Mooney, "Using Information Extraction to Aid the Discovery of Prediction Rules from Text", ACM SIGKDD-2000 Workshop on Text Mining, August 2000.

[9]J.R., Quinlan, "Learning Logical Definitions from Relations", Machine Learning, Vol.5, 1990.

[10]E. Riloff, "Automatically Constructing a Dictionary for Information Extraction Tasks", Proc. of the 11th National Conference on AI, 1993.

[11]S. Soderland, "Learning Information Extraction Rules for Semi-structured and Free Text", Machine Learning, Vol. 34., 1999.

[12]R. Yangarber and R. Grishman, "Extraction Pattern Discovery through Corpus Analysis", Proc. of Conference on Applied NLP ANLP-NAACL, 2000.