creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

[UCI]I804:24011-200000266886
조선대학교
CHOSUN UNIVERSITY

2018년 8월
석사학위 논문

구문 패턴과 개체명사전을 이용한 자연어 질의문장의 SQL 자동 변환 방법

조선대학교 산업기술융합대학원

소프트웨어융합공학과

신 희 철

조선대학교
CHOSUN UNIVERSITY

2018년 8월 석사학위논문

구문 패턴과 개체명사전을 이용한 자연어 질의문장의 SQL 자동 변환 방법

신 희 철

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

구문 패턴과 개체명사전을 이용한 자연어 질의문장의 SQL 자동 변환 방법

Automatic SQL conversion of natural language query sentences using syntax pattern and named entity dictionary

2018년 8월 24일

조선대학교 산업기술융합대학원

소프트웨어융합공학과

신 희 철

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

구문 패턴과 개체명사전을 이용한 자연어 질의문장의 SQL 자동 변환 방법

지도교수 김 판 구

이 논문을 공학석사학위신청 논문으로 제출함

2018년 6월

조선대학교 산업기술융합대학원

소프트웨어융합공학과

신 희 철

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

신희철의 석사학위논문을 인준함

위원장 조선대학교 교수 반 성 범 (인)

위 원 조선대학교 교수 최 준 호 (인)

위 원 조선대학교 교수 김 판 구 (인)

2018년 6월

조선대학교 산업기술융합대학원

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

목 차

ABSTRACT

I. 서론 ····· 1
A. 연구 배경 및 목적 ····· 1
B. 연구 내용 및 구성 ····· 2

II. 관련연구 ····· 3
A. 자연어 처리 ····· 3
1. 자연어 처리 개념 ····· 3
2. N-gram ····· 4
B. NLIDB(Natural Language Interface for Database) ····· 5
1. NLIDB 개념 ····· 5
2. 기존연구 ····· 5
C. SEQ2SQL ····· 7

III. 자연어 질의문장 SQL 변환 ····· 9
A. 전체 시스템 개요 ····· 9
B. 자연어 질의문장 전처리 과정 ····· 12
C. 의문사구와 명사구 추출 ····· 15
1. 의문사구 추출 ····· 16
2. 명사구 추출 ····· 18
D. SQL 변환 ····· 19
1. 의문사구 SQL 변환 ····· 19
2. 명사구 SQL 변환 ····· 20
E. 개체명 인식(Named Entity Recognition) ····· 21
1. 개체명 사전 ····· 21
F. SQL 생성 ····· 22

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

IV. 실험 및 평가 ····· 23
A. 평가 ····· 28

V. 결론 및 제언 ····· 30

참고문헌 ····· 31

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

표 목 차

[표 3-1] POS Tag list ····· 13
[표 3-2] POS Tagging 예시 ····· 14
[표 3-3] 의문사구 Chunking rule 예시 ····· 17
[표 3-4] 명사구 Chunking rule 예시 ····· 18
[표 3-5] WH- 의문문 개체명 예시 ····· 20
[표 3-6] 조건절 매칭 예시 ····· 21
[표 3-7] SQL 생성 예시 ····· 22
[표 4-1] 실험 환경 ····· 23
[표 4-2] WikiSQL 파일 예시 ····· 23
[표 4-3] 예시 문장 ····· 25
[표 4-4] Select절 miss 매칭 예시 ····· 26
[표 4-5] 조건절 miss 매칭 예시 ····· 27
[표 4-6] 매칭 정확도 ····· 28

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

그 림 목 차

[그림 2-1] 어절 N-gram 예시 ····· 4
[그림 2-2] WikiSQL 예시 ····· 8
[그림 3-1] 시스템 구성도 ····· 10
[그림 3-2] Chunking 예시 ····· 15
[그림 3-3] WikiSQL 질문 분포 ····· 16
[그림 5-1] NLIDB 예시 ····· 29

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

ABSTRACT

Automatic SQL conversion of natural language query sentences using syntax pattern and named entity dictionary

HeeChul Shin
Advisor : Prof. Pankoo Kim, Ph.D.
Department of Software Convergence Engineering
Graduate School of Industry Technology Convergence, Chosun University

Information society can be realized based on efficient collection, management and retrieval of large - scale information. Since the database (DB) is an important and important field in the information society, a system that facilitates DB access to general users can be regarded as an important core technology in the future. However, there is a restriction that general users should have knowledge of DB and business knowledge such as SQL in order to use DB directly. In order to solve the above limitations, researches on NLIDB (Natural Language Interfaces to Database System) are under way. To construct NLIDB, natural language processing technology and processing technology for DBMS (Database Management System) query language should precede. In this paper, I propose a method to convert to SQL statement so that general users without DB and SQL knowledge can search DB through natural language processing of query for the purpose of searching and using DB with general computing performance. The proposed method generates SQL by matching the columns using N-gram and named entity dictionary through natural language query. We analyze the experiment through the proposed method and describe the experimental result.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

Ⅰ. 서론

A. 연구 배경 및 목적

인터넷의 발달로 인하여 온라인상에는 수없이 많은 데이터가 존재하게 되었고, 이로 인해 빅데이터라는 용어가 생기고 데이터분석이라는 작업이 대중화가 되었다. 빅데이터 분석이란 다양하고 큰 데이터에서 새로운 가치를 발견하고 그 가치를 통해 미래를 예측하고 분석하는 과정을 의미한다. 양적인 개념에서 출발한 빅데이터가 최근에는 질적인 의미를 중요시 하는 방향으로 나아가고 있다[1]. 빅데이터 혹은 데이터 분석은 전문가들의 전유물이 아닌 일반인들도 수많은 경험적인 데이터를 분석하여 미래를 예측할 수 있다. 하지만 데이터에 기반 하지 않고 미래를 예측할 경우 선입견에 빠질 수 있고, 한두 가지 사례로 섣부르게 예단하게 되는 오류에 빠질 가능성이 높다. 이러한 오류에 빠지지 않기 위한 방법으로 데이터가 저장되어있는 데이터베이스(DB)를 이용하여 데이터를 분석할 수 있다. 정보화 사회는 대량의 정보를 효율적인 수집과 관리, 검색을 기반으로 실현할 수 있다.

하지만 일반 사용자들이 DB를 직접 이용하기 위해서는 DB에 대한 지식과 SQL등의 업무지식을 갖고 있어야 한다는 제약사항이 존재한다. 위와 같은 제약사항을 해결하기 위해 NLIDB (Natural Language Interfaces to Database System)에 관한 연구가 진행되고 있다. DB는 정보화 사회에서 활용도가 높고 중요한 핵심 분야이므로 일반 사용자들에게 DB 접근을 용이하게 하는 시스템은 향후 중요한 핵심기술로 평가될 수 있다. NLIDB 구축을 위해서는 자연어 처리 기술, DBMS(Database Management System) 질의어에 대한 처리 기술 등의 연구가 선행되어야 한다.

본 연구에서는 학습에 필요한 질의문이나 SQL문등의 데이터를 필요로 하지 않고, 학습시키는 과정을 거치지 않고도 여러 DB에 적용이 가능하고 검색할 수 있는 이식성이 높은 질의문의 자연어 처리를 통한 SQL변환방법을 제안한다. 제안하는 방법은 DB와 SQL등의 지식이 없는 일반인도 DB의 검색이 가능하도록 하기 위한 목적으로 높은 컴퓨팅 성능을 필요로 하지 않고, 간단한 질의문으로 DB의 데이터를 검색하여 이용할 수 있게 하는데 초점을 두고 연구를 진행하였다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

B. 연구 내용 및 구성

본 연구는 DB검색을 위한 질의문을 자연어 처리를 통해 SQL문으로의 변환 방법에 대해 제안하고, 실험과 결과에 대해 기술한다. 본 논문의 구성은 다음과 같다. 2장에서는 자연어 처리에 관한 설명과 NLIDB기술 등 관련연구를 살펴본다. 3장에서는 질의문장의 SQL 변환 필요성과 제안하는 방법에 관한 시스템 구성도와 함께 본 연구의 내용을 기술한다. 4장에서는 제안하는 방법을 통한 실험 분석과 분석 후 실험에 관한 내용과 결과에 대해 기술한다. 마지막으로 5장에서는 본 연구의 결론과 향후 연구에 관해 서술하며, 마무리한다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

Ⅱ. 관련연구

A. 자연어 처리

1. 자연어 처리 개념

자연어 처리란 이러한 인간의 언어인 자연어를 컴퓨터가 이해할 수 있도록 여러 기술과 지식들을 접목시켜 처리하는 분야이다. 자연어로 구성된 문장은 보통 여러 개의 단어로 이루어져 있으며, 각 단어 사이에는 문장 내의 단어가 맡은 역할에 따른 연결 관계가 존재한다. 자연어 처리를 위해서는 문장구성의 성분과 성분들 간의 관계를 분석할 필요가 있다. 이러한 관계를 분석하기 위해 사용되는 것이 형태소 분석기와 구문 분석기이다. 형태소란 문장에서 의미를 갖는 최소 단위를 말하며, 형태소를 분석할 때는 사전뿐만 아닌 문법 규칙, 언어 사용 통계 등의 정보를 활용한다. 구문 분석기는 문장의 형태소 분석 결과를 바탕으로 문장을 이루고 있는 (명사, 동사, 부사)구 등의 구문들을 묶어주는 것뿐만 아닌, 문장의 구성성분을 밝혀내고 문장의 문법적 구조를 결정하는 기술이다.

자연어 처리 기술은 인간과 컴퓨터가 상호 작용하기 위해 필요한 핵심적인 기술이며, 정보 검색, 분류, 음성 인식 등의 여러 분야에 사용되고 있다. 본 연구에서는 Tokenizing, POS Tagging, Chunking, Stemming등의 자연어 처리 기술을 이용하여 문자열 비교에 사용하고, 제안하는 방법을 이용하여 질의문에 대한 SQL문을 생성하였다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

2. N-Gram

N-gram이란 N개의 어절/음절을 연쇄적으로 분류해 그 빈도를 따지는 알고리즘이다. 텍스트, 바이너리 등 전체 문자열을 N값으로 나누어 통계학적으로 사용한 방법을 말하며 N=1일 경우 unigram, 2일 경우 bigram, 3일 경우 trigram으로 불린다. N-gram 알고리즘은 확률과 통계를 바탕으로 색인 분석 등에 널리 쓰이는 방식이며, 초기 검색엔진에서 쓰인 검색 알고리즘의 하나이다. 검색 시스템에서 키워드 추출, 생명 공학 분야의 염기서열이나 분자구조 분석, 악성코드의 API 추출 등에서 쓰일 수 있는 알고리즘이다. 이러한 N-gram 알고리즘은 단순하고, 검색 누락이 발생하지 않는다는 장점이 있다. 하지만 N-gram 색인 방식은 검색자가 의도하지 않은 Web 페이지가 검색 결과에 섞여버리게 되어 검색 노이즈가 커지는 점, 인덱스 크기가 커져 데이터베이스 용량이 커지는 등의 단점이 있다. N-gram을 두 문장의 텍스트 비교에 적용할 경우 각 문장을 어절/음절 단위 등으로 분류하여 비교가 가능하다. 다음 [그림 2-1]은 “지능형 컴퓨팅 연구실”이라는 예시 문장에 대해 어절 단위의 Uni-gram과 Bi-gram의 결과를 나타낸다.

지능형 컴퓨팅 연구실

Uni - gram : [“지능형”] [“컴퓨팅”] [“연구실”]

Bi - gram : [“지능형”, “컴퓨팅”] [“컴퓨팅”, “연구실”]

[그림 2-1] 어절 N-gram 예시

[그림 2-1]에서 Uni-gram은 예시 문장을 어절 단위로 쪼개어 한 어절씩 나눈 결과이며, Bi-gram은 어절 단위로 쪼갠 문장을 두 어절씩 나눈 결과를 의미한다. 본 연구에서는 N-gram 결과를 이용하여 문자열을 비교하였고, 문자열을 구로 나누어 검색의 정확도와 속도를 향상시켰다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

B. NLIDB(Natural Language Inerface for Database)

1. NLIDB 개념

데이터베이스(DB)에서 필요한 정보를 얻기 위해서는 각각의 DB에 맞는 SQL (Structured Query Language)을 생성해 주어야 한다. NLIDB는 사용자가 입력한 자연어 질의를 DB에 최적화된 SQL 문장으로 자동 변환하여 DB에 포함된 내용을 검색하는 기술이다. NLIDB는 사용자가 입력하는 자연어 질의 요청을 결과물로 출력이 가능한 SQL로 번역을 해주기 때문에 SQL에 익숙하지 않은 사용자도 DB에 접근하여 사용할 수 있고, 보다 쉽게 사용자가 원하는 정보를 찾을 수 있다.

2. 기존연구

사용자가 원하는 데이터를 자연어처리(NLP)를 통해 질의하고 원하는 결과를 획득하기 위한 NLIDB(Natural Language interfaces to Database system)에 대한 연구가 진행되고 있다. 의미 문법 시스템인 IRUS[2], DELPHI[3], PLANES[4]은 입력을 분석하고 구문 분석 트리를 DB query에 매핑 함으로써 질의응답을 수행한다. 의미 문법 시스템의 IRUS(Information Retrieval Using the RUS parser)는 영국 Cambridge대학의 BBN Lab.에서 만든 DB를 위한 NLI(Natural Language Interface)시스템이다. 이 시스템은 단순히 질문의 대답을 찾기보다 질문의 의미 파악을 시도했고, 특정 분야와 특정 DB의 구조에 상관없이 사용자 질의를 처리할 수 있게 구성되어 있는 지식기반 시스템이다. 이 시스템에서 사용하고 있는 지식의 형태로는 의미 규칙의 집합, 영어 문법, 영어 사전, 의미 표현 언어로 쓰인 사용했던 질의 등이 있다.

NLIDB는 중간언어(Intermediate language)가 사용될 때 먼저 자연어 질문을 중간 논리 쿼리(Intermediate Logical Query)로 변환하여 내부의 의미 표현 언어로 표현한다. 중간 논리 쿼리는 DB 구조와 독립적인 고급 개념에서 질문의 의미를 표현하고, 논리적 쿼리가 DB의 쿼리 언어로 변환하여 표현한다. 이러한 시스템으로는 미국의 J. Burger가 개발한 DB 시스템용 Table driven시스템인 EUFID[5], SQUIRREL[6], TEAM[7], DATALOG[8], EXACT[9]등이 있다. TEAM(Transportable English

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

database Access Medium) 시스템의 경우 특정 영역에 제한된 질문만을 처리, 하나의 데이터베이스 내에서 모든 정보를 유지하는 것을 가정하고, 하나의 작업에 국한된 질문만을 처리하도록 하는 등의 다른 NLIDB 시스템이 가지고 있는 단점을 극복하기 위해 구현된 시스템이다. TEAM의 하나의 큰 가정은 NLI가 잘 구성된다면 새로운 환경에 시스템을 적응시키기 위한 정보는 데이터베이스에 대해 얻거나, 작업에 대해 많은 경험을 가진 사용자들과의 대화를 통해 얻을 수 있다는 것이다. TEAM에 주어진 사용자 질의는 논리형태로 바뀌고 DB 질의형태로 변환 후 파서, 문법, 사전, 의미 변환 함수 등으로 구성된 모듈을 거쳐 Query를 생성한다.

NLIDB 기술은 통계시스템에도 적용될 수 있다. NEIS(Natural language processing Engine for Intelligent Statistics)의 경우 통합통계시스템 DB와 KTPIN 시스템 DB의 정보조회용 통계정보를 활용하여 통계정보 검색 시스템을 개발하였다[10]. NEIS 시스템은 질의문이 서버에 입력되면 자연어 처리를 통해 질의문을 분석 후 구축된 메타정보를 활용하여 질의 유형과 통계메뉴 리스트를 추출하여 나타내준다. 나타난 질의 유형을 선택하면 해당되는 쿼리문을 실행하여 통계정보를 조회하여 제공한다. 이를 통해 불필요한 통계메뉴 탐색 시간을 없애고, 통계 조회시간을 줄일 수 있다.

한용진 외 1명은 NLIDB의 연구가 학습 데이터 및 입력 표현에 대한 구문 분석 결과에 의존하게 된다는 단점을 해결하기 위해 주어진 분야에 맞는 의미 표현들을 생성하였고 각 의미 표현에 맞는 하나 이상의 자연어 표현을 생성하는 방법을 제안했다[11]. 온톨로지와 같은 지식 기반이 주어졌을 때, 표현할 수 있는 의미들을 개념 그래프를 이용하여 정의한 후 계산량을 줄이기 위해 공통적인 개념 그래프를 하나의 그룹으로 만들어 표현하였다.

이 외에도 통계적인 기계번역을 이용하여 문장을 분석 후 질의로 변환하는 방법[12], 자연어 처리기술을 이용한 NLIDB 시스템 제안[13], 백과사전의 문장에서 인물의 특징을 나타내는 속성 값을 인식하여 추출하는 방법을 통해 지식베이스를 구축하는 방법 제안[14]등의 연구가 있다.

자연어 기반 인터페이스를 적용하기 위해서는 자연어 질의에서 주제어와 개체명을 효과적으로 인식하는 방법이 필요하다. 이러한 연구로는 구조적인 패턴 정의와 매칭을 통한 개체명 인식[15], 규칙과 통계를 혼합한 개체명 인식 방법 제안[16], 날짜, 시간, 장소 등을 나타내는 개체명을 유한오토마타와 음절 n-gram기반의 HMM을 이용해 인식하는 방법 제안[17,18] 등의 연구가 있다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

NLIDB 분야의 많은 연구에도 불구하고, 대부분의 NLIDB 시스템은 오류가 발생하기 쉬운 특성으로 인해 실시간 어플리케이션에서 요구되는 수용을 달성하지 못했다. Saikrishna Srirampur외 3명은 오류가 없는 NLIDB 시스템을 구축하는 것을 목표로 NLIDB 시스템 구축의 중간 단계에서 오류 발생 가능성이 적은 접근법을 제안하였다 [19].

C. SEQ2SQL

Seq2SQL은 Salesforce AI연구소에서 강화 학습을 사용하여 자연 언어에서 구조적 질의를 생성한 연구이다[20]. 이 연구는 자연어 질문을 SQL로 변환하기 위해 Neural network로 구성된 Seq2SQL을 소개하고 있다. Seq2SQL의 연구진은 Wikipedia의 26,375개의 HTML테이블에서 추출한 자연어 관련 질문, SQL Query 및 테이블에 대해 직접 주석을 첨부한 자료인 WikiSQL을 만들어 연구에 사용하고 Github에 공개하였다.

SQL은 한 가지 질문에도 복수의 정확한 쿼리가 존재할 수 있지만, 이를 강화학습을 적용해 해결하였다. 강화학습이란 머신러닝 시스템의 결과를 다른 머신러닝 시스템이 평가하는 과정을 통해 자기 시스템의 성능을 개선하는 학습방법이다. 연구진들은 SQL을 3개의 component인 aggregation operator와 Select Column, Where clause로 구분하였다. aggregation operator는 classification을 이용하였고, Select Column에 해당하는 Input Table Column을 특정하여 Select Column을 선택하였다. 조건절은 pointer network를 활용하여 생성하였다. 사용한 모델은 aggregation operator와 Select Column의 경우 Cross entropy loss를 활용한 지도학습을 이용하였고, 조건절의 경우 쿼리 조건에 순서 개념이 없는 문제를 해결하기 위해 policy gradient reinforcement learning를 이용하였다.

다음 [그림 2-2]는 Question, query and table ID를 jsonl형식으로 나타낸 WikiSQL 데이터셋에 대한 설명이다[21].

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

Question, query and table ID

These files are contained in the `*.jsonl` files. A line looks like the following:

```
{
   "phase":1,
   "question":"who is the manufacturer for the order year 1998?",
   "sql":{
      "conds":[
         [
            0,
            0,
            "1998"
         ]
      ],
      "sel":1,
      "agg":0
   },
   "table_id":"1-10007452-3"
}
```

The fields represent the following:

- `phase` : the phase in which the dataset was collection. We collected WikiSQL in two phases.
- `question` : the natural language question written by the worker.
- `table_id` : the ID of the table to which this question is addressed.
- `sql` : the SQL query corresponding to the question. This has the following subfields:
  - `sel` : the numerical index of the column that is being selected. You can find the actual column from the table.
  - `agg` : the numerical index of the aggregation operator that is being used. You can find the actual operator from `Query.agg_ops` in `lib/query.py`.
  - `conds` : a list of triplets `(column_index, operator_index, condition)` where:
    - `column_index` : the numerical index of the condition column that is being used. You can find the actual column from the table.
    - `operator_index` : the numerical index of the condition operator that is being used. You can find the actual operator from `Query.cond_ops` in `lib/query.py`.
    - `condition` : the comparison value for the condition, in either `string` or `float` type.

[그림 2-2] WikiSQL 예시

dCollection @ chosun

조선대학교
CHOSUN UNIVERSITY

Ⅲ. 자연어 질의문장 SQL변환

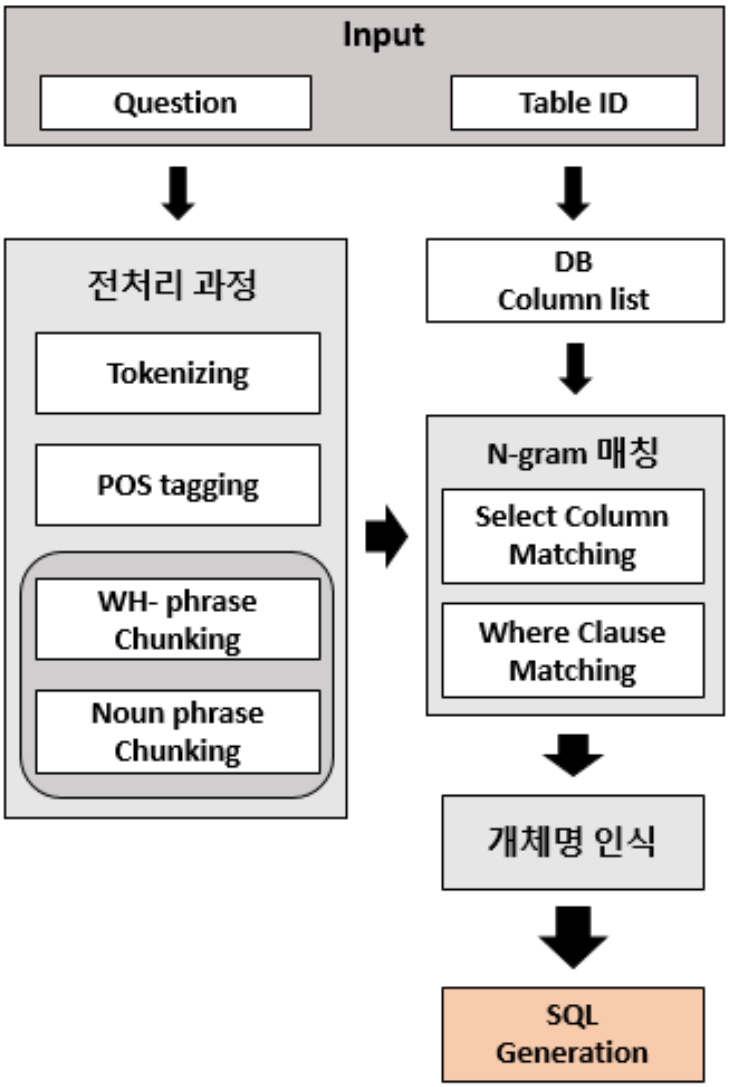
A. 전체 시스템 개요

정보의 가치가 중요하게 인식되고 데이터를 쉽게 접할 수 있는 정보화 사회는 일상에서 많은 정보와 관계를 맺고 있다. 정보화 사회에서 데이터 분석을 통해 가치 있는 정보를 얻는 것은 일상생활에서도 응용하는 중요한 행위이다. 데이터베이스(DB)는 많은 정보 중 필요에 의해서 데이터를 일정한 형태로 저장해 놓은 것을 의미한다. DB 데이터를 이용하기 위해서는 DB의 데이터를 조회, 관리하기 위한 SQL(Structured Query Language)이라는 프로그래밍 언어를 사용해야 한다. DB 사용자들은 SQL을 이용하여 DB에 접근하여 직접 조회할 수 있다. 하지만 DB에 대한 지식과 SQL의 업무 지식을 필요로 하는 제약사항으로 인해 일반 사용자들은 DB 데이터의 접근이 쉽지 않다. DB와 SQL에 익숙하지 않은 일반 사용자들에게는 간단한 SQL문이라도 그것을 배우고 이해하여 적절하게 사용하기 까지는 많은 시간과 노력이 요구된다. 이러한 제약사항을 해결하기 위해 사용자가 입력한 자연어 질의를 DB에 최적화된 SQL 문장으로 자동 변환하여 DB에 포함된 내용을 검색하는 기술인 NLIDB에 대한 연구는 오래전부터 진행되고 있다. 자연어를 이용함으로써 DB 질의어가 가지고 있는 일반 사용자의 제약 사항을 완화시키고, DB접근의 장벽을 낮출 수 있다. 자연어 질의 인터페이스는 SQL에 익숙하지 않은 사용자뿐만 아니라 SQL을 학습하는 이용자를 위한 학습 도구로도 사용될 수 있으며, DB시스템이 가져야 할 유용한 요소가 될 것으로 기대된다. 본 논문에서는 위에서 언급된 제약사항을 완화시키고, 일반 사용자의 유연한 DB접근을 위해 질의문장의 자연어 처리를 통한 SQL변환방법을 기술한다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

본 논문에서는 질의문의 자연어 처리를 통해 DB Column list와 매칭 하여 질의문을 SQL로 변환하는 방법에 대해 제안한다. [그림 3-1]은 질의문의 자연어 처리를 통한 SQL변환방법의 시스템 구성도이다.



[그림 3-1] 시스템 구성도

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

본 논문에서 제안하는 질의문의 SQL변환과정은 다음과 같다. 먼저 입력으로 질의문과 해당 DB Table ID를 사용한다. Input 데이터의 질의문과 Table ID는 각각의 과정을 거친다. 입력으로 쓰인 질의문은 4단계의 전처리 과정을 거친다. 첫 번째로, 질의문을 단어별로 나눠주는 Tokenizing 처리를 한다. 두 번째로, Tokenizing처리를 거친 질의문의 형태소를 분석하는 POS(Part of speech) tagging 과정을 거친다. 세 번째로, 형태소 분석이 완료된 질의문의 WH-phrase(의문사구)를 추출하기 위한 Chunking과정을 거친다. 네 번째로, 질의문의 Noun phrase(명사구)를 추출하기 위한 Chunking과정을 거친다.

입력으로 쓰였던 Table ID는 DB에서 해당되는 ID의 Column list를 가져온다. DB Column list와 전처리 과정에서 추출된 WH-phrase, 명사구를 각각 N-gram으로 매칭시킨 후, WH-phrase와 매칭된 Select절의 Column과 명사구와 매칭된 조건절의 Column, 조건값을 추출한다. 만약 WH-phrase를 통해 Select절의 Column이 매칭 되지 않을 경우 WH- 의문문의 개체명 인식을 통해 Select절의 Column을 매칭 시킨다. 마지막으로 추출된 값을 조합하여 SQL문을 생성함으로써 질의문을 통해 SQL문을 얻을 수 있다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

B. 자연어 질의문장 전처리 과정

전처리 과정은 자연어 처리 분야에서 글이나 문서 등을 분석하기 위한 기초 작업이다. 본 논문에서는 질의문을 단어별로 Token을 나눠주고 형태소 분석을 거친 후, 의문사구와 명사구를 추출하고 DB Column list와 매칭하여 SQL문의 Select절과 조건절에 알맞은 데이터를 추출하여 SQL문을 생성한다. 본 연구의 전처리 과정은 Tokenizing과 POS tagging 과정을 거쳤다.

Token이란 의미를 가지는 문자열을 뜻한다. Token은 형태소나 이보다 상위 개념인 단어까지 포함한다. Tokenizing은 문서나 문장을 분석에 용이하도록 하나의 단어 단위로 세분화시키는 작업이며, 전처리 과정의 초기 과정이다. 본 연구에서는 주어진 질의문에서 띄어쓰기를 기준으로 단어를 Token화 시켰다.

POS(Part Of Speech) tagging은 단어의 품사를 판별하는 작업이다. 모든 자연어 처리 분야에서 기본적으로 필요하며 중요한 것이 그 언어의 형태소 분석이며, 형태소 분석이 끝난 후 구문 분석, 의미 분석 등을 거쳐 자연어 관련 분야에 응용될 수 있다. 본 연구에서는 Python의 영어 형태소 분석기인 NLTK 라이브러리를 사용하여 POS tagging을 진행하였다.

다음 [표 3-1]은 POS Tag의 종류와 함께 설명과 예시를 담은 POS Tag list이다.

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

[표 3-1] POS Tag list

POS Tag	Description	Example
CC	coordinating conjunction	and
CD	cardinal number	1, third
DT	determiner	the
EX	existential there	there is
FW	foreign word	d' hoevre
IN	preposition/subordinating conjunction	in, of, like
JJ	adjective	big
JJR	adjective, comparative	bigger
JJS	adjective, superlative	biggest
LS	list marker	1)
MD	modal	could, will
NN	noun, singular or mass	door
NNS	noun plural	doors
NNP	proper noun, singular	John
NNPS	proper noun, plural	Vikings
PDT	predeterminer	both the boys
POS	possessive ending	friend 's
PRP	personal pronoun	I, he, it
PRP$	possessive pronoun	my, his
RB	adverb	however, usually, here
RBR	adverb, comparative	better
RBS	adverb, superlative	best
RP	particle	give up
TO	to	to go, to him
UH	interjection	oh, wow
VB	verb, base form	take
VBD	verb, past tense	took
VBG	verb, gerund/present participle	taking
VBN	verb, past participle	taken
VBP	verb, sing. present, non-3d	take
VBZ	verb, 3rd person sing. present	takes
WDT	wh-determiner	which
WP	wh-pronoun	who, what
WP$	possessive wh-pronoun	whose
WRB	wh-abverb	where, when

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

[표 3-2]는 문장에 대한 품사 태깅 예시이다.

[표 3-2] POS Tagging 예시

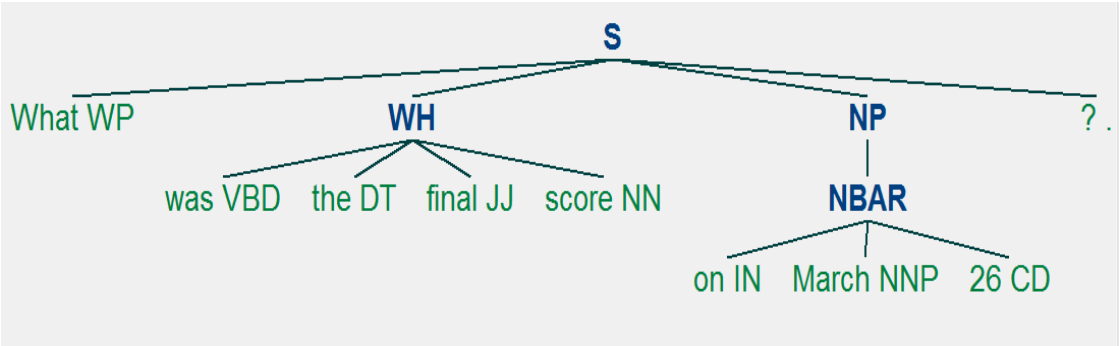
입력문장 예시	품사 태깅 결과
Who is the academic & University affairs when the Human resources & operations is N. Charles Hamilton?	('Who', 'WP'), ('is', 'VBZ'), ('the', 'DT'), ('academic', 'JJ'), ('&', 'CC'), ('University', 'NNP'), ('affairs', 'NNS'), ('when', 'WRB'), ('the', 'DT'), ('Human', 'NNP'), ('resources', 'NNS'), ('&', 'CC'), ('operations', 'NNS'), ('is', 'VBZ'), ('N.', 'NNP'), ('Charles', 'NNP'), ('Hamilton', 'NNP')
What was the final score on March 26?	('What', 'WP'), ('was', 'VBD'), ('the', 'DT'), ('final', 'JJ'), ('score', 'NN'), ('on', 'IN'), ('March', 'NNP'), ('26', 'CD')

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

C. 의문사구와 명사구 추출

본 연구에서는 문장의 Token을 나누어 POS tagging을 한 후 단어를 결합하여 의문사구(WH- phrase)와 명사구(Noun phrase)를 추출하는 Chunking과정을 거친다. 의문사구와 명사구는 각각의 rule을 만들어 Chunk를 생성한다. 다음 [그림 3-2]는 예시 문장인 “What was the final score on March 26?”에 대한 Chunking 예시를 나타낸다.



[그림 3-2] Chunking 예시

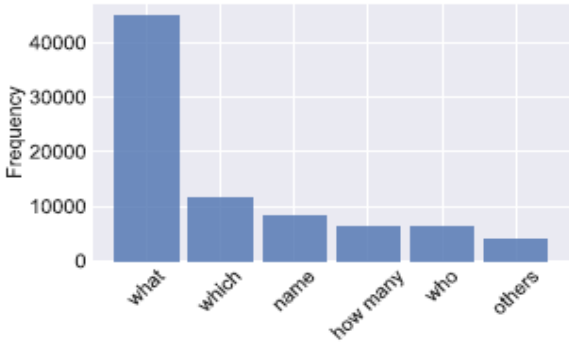
[그림 3-2]의 WH- phrase(의문사구)를 보면 “What”이 포함되어 있지 않은 모습을 볼 수 있다. 이는 의문사구 Chunking rule에 POS Tag가 의문사인 경우를 포함시키지 않는 Chinking(칭킹)을 사용하여 문장에서 Chinking을 찾아 제외함으로써 의문사구를 생성한 결과이다.

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

1. 의문사구 추출

영어의 의문사 중 What, Who, When, Where, Why, How로 시작하는 의문문을 WH-question이라 하며, WH- 의문문은 어떠한 궁금한 것에 대한 대답을 물을 경우 사용하는 의문문이다. 또한 구체적인 정보를 요구하고 내용을 답해야 하는 의문문이므로 Yes나 No로 답할 수 없다는 특징이 있다. 다음 [그림 3-3]는 WikiSQL의 질문 분포를 나타낸다[20].



[그림 3-3] WikiSQL 질문 분포

WikiSQL의 질문 분포인 [그림 3-3]에서도 나타나듯이 많은 질문이 WH- 의문문으로 이루어져 있음을 알 수 있다. 의문사구의 이동에는 구조적 제약이 따른다. Ross의 연구 중 WH-이동의 구조적 제약을 WH-Island Constraint(WH- 섬 제약)이라고 한 연구가 있다[22]. WH-Island Constraint(WH- 섬 제약)는 WH-보문소[1]나 Comp[2] 내에 WH-phrase가 있을 때는 그 절로부터 어떤 요소도 이동시킬 수 없다는 제약이다. 본 연구에서는 이러한 제약으로 인해 자연어 질문을 WH- 의문문으로 시작한다는 조건 하에 실험을 진행하였고, 자연어 질문의 의문사 구를 추출한 후 추출된 의문사구와 DB Column name과의 매칭을 통하여 SQL문의 Select절의 Column을 생성하였다. 영어에서 의문문의 어순은 동사+주어+... 으로 구성되며 의문사가 있는 경우는 의문사+동사+주어+... 으로 구성된다. 또한 앞의 의문문의 어순 사이에는 명사를 꾸며주는 형용사와 동사, 형용사, 부사 등을 꾸며주는 부사 등이 쓰일 수 있다.

1) 문장을 보완해주는 요소를 말하며, Complement는 보어를 뜻한다.
2) that 절을 포함한 for - to, poss -ing, if whether 및 wh-hrases 등

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

다음 [표 3-3]은 의문사구의 Chunking rule example을 나타낸다.

[표 3-3] 의문사구 Chunking rule 예시

의문사구 Chunking rule 예시
<W.*><N.*>
<W.*><VB.*|IN>
<W.*><VB.*|IN><N.*>
<W.*|NNP><VB.*|IN><DT><N.*>
<W.*|NNP><VB.*|IN><DT><CD><N.*><N.*>
<W.*|NNP><VB.*|IN><DT><CD><N.*><POS><N.*>
<W.*|NNP><VB.*|IN><PDT><DT><JJ.*|RB.*><N.*>
<W.*|NNP><VB.*|IN><PDT><DT><JJ.*|RB.*><CD>
<W.*|NNP><VB.*|IN><PDT><DT><JJ.*|RB.*><CD><N.*>
<W.*|NNP><VB.*|IN><PDT><DT><JJ.*|RB.*><CD><N.*><POS><N.*>
⋮

[표 3-3]의 W.*이나 N.*은 W나 N으로 시작하는 POS tag의 모든 것을 의미하고, "|"표시는 or의 의미를 가진다. 또한 <>표시는 <>표시 안의 태그의 순서적 위치를 나타낸다. 본 연구에서는 의문사구에 많이 쓰이는 문장의 패턴을 분석하여 위 예시들의 POS tag 들과 기타 예시들을 포괄적으로 나타내는 의문사구의 Chunking룰을 다음과 같이 사용하였다.

{<W.*|JJ|NNP><VB.*|IN>?<N.*>?<PDT>?<DT>?<VB.*>?<JJ.*|RB.*>?<VB.*>?<CD>?<N.*>?<POS>?<IN>?<N.*>?<N.*>?}

위의 Chunking rule의 "?"표시는 물음표 앞의 POS tag가 해당하는 위치에 있을 경우와 없을 경우를 모두 아우르는 표시이다. 또한 본 연구에서 사용한 의문사구는 <W.*> 태그를 Chinking하여 추출하였다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

2. 명사구 추출

Noun phrase(명사구)는 문장에서 명사의 역할을 하는 구를 뜻한다. 명사구는 명사와 수식어로 구성되어 있으며 명사처럼 문장에서 주어, 목적어, 보어 역할을 한다. 명사구는 명사와 함께 부사, 형용사 전치사구 등이 쓰일 수 있으며, 일반적인 명사구는 관사+부사+형용사+명사+형용사+전치사구 등의 구조를 가지고 있다. 본 연구에서는 추출한 명사구들을 통해 각각 DB Column list와 비교하여 조건절을 구성하였다. [표 3-4]는 명사구의 Chunking rule을 나타낸다.

[표 3-4] 명사구 Chunking rule 예시

명사구 Chunking rule 예시		
<NBAR>	<N.*|CD>	<JJ.*>?<N.*|CD>
	<RB.*>?<N.*|CD>	<IN>?<N.*|CD>
	<DT><JJ.*>?<N.*|CD>	<DT><RB.*>?<N.*|CD>
	<DT><IN>?<N.*|CD>	<P.*><DT><JJ.*>?<N.*|CD>
	<P.*><DT><RB.*>?<N.*|CD>	<P.*><DT><IN>?<N.*|CD>
	<VB.*><VB.*|R.*>	…
	1. {<P.*>?<DT>?<JJ.* | RB.* | IN>?<N.* | CD><CD>?} 2. {<VB.*><VB.* | R.*>}	
Noun phrase	1. {<DT>?<JJ.*>?<VB.*>?<NBAR><NBAR>?<NBAR>?} 2. {<VB.*>?<VB.*>?<IN><NBAR><NBAR>?<NBAR>?} 3. {<NBAR><NBAR>?<NBAR>?<N.* | CD>?} 4. {<JJ>}	

[표 3-4]의 <NBAR>의 경우 명사구 Chunking rule의 일부를 미리 추출하여 따로 <NBAR>라는 태그로 이용할 수 있게 만들었다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

D. SQL 변환

Select절의 Column과 조건절은 입력한 질의문의 전처리 과정에서 나온 의문사구와 명사구를 각각 DB Column list와의 N-gram 매칭을 통하여 SQL을 생성하였다. Select Column과 조건절의 매칭 모두 전처리 과정 후 나온 의문사구, 명사구와 DB Column list 모두 문자열은 모두 소문자로 치환하여 비교하였다. 만약 DB의 Column값이 문자형이 아닌 숫자 형일 경우 Column의 값을 문자형으로 치환 후 비교한 후 SQL문의 생성 시 다시 숫자 형으로 치환한다. 또한 의문사구와 명사구 모두 각각의 phrase를 N-gram으로 나눈 후 각 Token이 보통 문장에서 많이 쓰이는 관사(a, an, the), 대명사(it, he, her), 전치사(of, on, for)와 같은 불용어를 제거한 후 매칭을 하였다.

1. 의문사구 SQL 변환

Select Column의 매칭은 전처리 과정에서 추출된 의문사구와 DB Column name들을 N-gram으로 나누어 비교하였다. 다음 수식 (1)은 각 DB Column name과 의문사구의 매칭 점수를 나타낸다.

$$Score = \frac{count}{total} \qquad (1)$$

수식 (1)의 count는 각 DB Column name과 의문사구를 각각의 Uni-gram으로 나눈 후 비교하여 문자열이 같을 때마다 +2를 더한 최종 값을 의미하며, total은 각 DB Column name과 의문사구의 각각의 Uni-gram 총 개수를 의미한다. 의문사구와 각 Column name과의 Score중 가장 높은 Score의 Column name을 질의문의 Select절의 Column으로 선택한다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

2. 명사구 SQL 변환

조건절의 매칭은 질의문에 대해 전처리 과정에서 추출된 명사구와 DB Column list들을 N-gram으로 나누어 비교하였다. 조건절도 Select절의 Column name을 선택하는 것과 같이 수식 (1)을 사용하여 명사구와 각 DB Column list와의 Score중 가장 높은 Score의 Column들을 추출한다. 조건절의 Column의 선택은 각 명사구와 DB의 각 row의 Column마다 비교하여 Score가 높은 Column을 추출한다. 각 명사구와 각 row에서 추출된 Column들 중 각 명사구마다 가장 높은 Score의 Column을 선택한다. 각 명사구에 매칭되는 Column들 중 중복되는 Column은 Score가 더 높은 Column만을 남겨놓고 제외한다.

[표 3-5] 조건절 매칭 예시

예시 문장		
What is the score of the match with opponents Gretchen Magers Kelly Jones?		
명사구	Score	예측한 조건절
the match with opponents Gretchen	0.2857142857142857	Opponents = “Gretchen Magers Kelly Jones”
Magers Kelly Jones	0.8571428571428571	

[표 3-5]는 위에서 설명한 조건절 매칭의 예시를 나타낸다. 예시 문장인 “What is the score of the match with opponents Gretchen Magers Kelly Jones?”에 대한 전처리 과정 후 명사구는 “the match with opponents Gretchen”와 “Magers Kelly Jones”의 예측한 조건절의 값이 같다. 명사구의 “the match with opponents Gretchen”의 경우 불용어인 “the”와 “with”의 값을 제거한 후 Column list와 매칭한 Score이다. 위의 [표 3-3]과 같이 예측한 Column이 중복되는 경우 가장 높은 Score의 값만을 조건절의 예측 값으로 사용한다.

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

E. 개체명 인식(Named Entity Recognition)

개체명이란 지명, 날짜, 이름, 제목, 시간 등을 말하며 고유명사나 미등록 언어인 경우가 많다. 글에서 이러한 개체명을 찾아 해당하는 의미의 범주를 결정하는 것을 개체명 인식이라 한다.

1. 개체명 사전

Select절의 정확도를 높이기 위해 WH- 각 의문사들에 대해 관련된 단어들을 수집하였다. Where의 경우 장소와 관련된 Location, Position, Place, Point 등등, When의 경우 Time, Age, Hour, Day 등등 관련 단어를 수집하였고 수집된 단어들을 정리하여 개체명 사전을 구축하였다. 다음 [표 3-6]은 WH- 의문문에 대한 개체명 사전의 예시를 나타낸다.

[표 3-6] WH- 의문문 개체명 예시

WH- 의문사	의문문에 해당하는 개체명 예시
Who	Person, Someone, Anyone, Anybody …
What	That, It, Something, Anything …
When	Time, Age, Hour, Day, Minute …
Where	Location, Position, Place, Point …
Which	Any, One, Number, Indication …

의문사구에서 Select절의 Column이 매칭 되지 않았을 경우, 의문사구의 WH- 의문사에 대한 개체명 사전을 이용하여 Select절의 Column을 매칭 시킨다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

F. SQL 생성

위의 과정들을 통해 나온 Select절의 Column과 조건절에 매칭 되는 Column값을 가지고 질의문에 대한 SQL을 생성한다. Select절과 조건절의 경우 예측한 값과 SQL 생성 방법은 다음 [표 3-7]과 같다.

[표 3-7] SQL 생성 예시

예측값 표현	"sel": 6, "conds": [[5, 0, "Gretchen Magers Kelly Jones"]]
SQL문	Select Score from 1-1028356-3 where Opponents = "Gretchen Magers Kelly Jones"

예측값 표현은 본 연구에서 사용한 데이터셋인 WikiSQL과 비교를 용이하게 하기 위해 유사한 형태로 만들어 비교하였다. [표 3-6]에서 sel의 값은 예측한 Select절의 Column값을 DB Column name의 인덱스로 나타내었고, conds의 값은 예측한 조건절의 값을 나타내었고, 2장의 관련연구인 SEQ2SQL의 [그림 2-2]의 설명 중 일부와 같다. conds는 조건절의 조건의 개수가 질의문마다 다르므로 배열 형태로 만들어 놓았고, 그 안의 또 다른 배열의 첫 번째인 5는 DB Column name의 인덱스를 뜻하고, 0은 (=)을 뜻하며, "Gretchen Magers Kelly Jones"는 Column 값을 뜻한다.

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

Ⅳ. 실험 및 평가

본 장에서는 제안하는 질의문의 자연어 처리를 통해 SQL문을 생성하기 위한 실험 환경 및 실험 방법, 결과와 분석을 기술한다. 앞에서 기술한 데이터셋을 이용하여 제안하는 방법에 대한 실험을 진행하였다.

본 연구의 실험은 Window 운영체제의 Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz 환경에서 진행되었으며, [표 4-1]은 본 연구의 실험환경을 나타낸다.

[표 4-1] 실험 환경

OS	Window 7
Language	Python 3.6
Tool	Spyder
Database	SQLite
Library	NLTK, Json

실험에 사용된 데이터는 WikiSQL의 WH- 의문문의 형식을 지닌 질의문에 대한 데이터를 실험에 사용하였다. WikiSQL의 -.jsonl 파일의 경우 table_id와 question 그리고 question에 해당하는 sql문을 json형식의 파일로 모아놓았다. 또한 -.tables.jsonl 파일의 경우에는 각 테이블의 Column 이름을 적어놓은 header와 테이블의 Column list들을 모아놓았다. WikiSQL의 json파일들의 예시는 다음 [표 4-2]와 같다.

[표 4-2] WikiSQL 파일 예시

-.jsonl 파일	{“phase”: 1, “table_id”: “1-11959669-7”, “question”: “What was the final score on March 26?”, “sql”: {“sel”: 3, “conds”: [[1, 0, “March 26”]], “agg”: 0}}
-.tables .jsonl 파일	{“header”: [“Game”, “Date”, “Team”, “Score”, “High points”, “High rebounds”, “High assists”, “Location Attendance”, “Record”], “page_title”: “2007\u201308 Boston Celtics season”, “types”: [“real”, “text”, “text”, “text”, “text”, “text”, “text”, “text”, “text”], “id”: “1-11959669-7”, “section_title”: “Game log”, “caption”: “Game log”,

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

"rows": [[58, "March 2", "Atlanta", "98\u201388", "Pierce (30)", "Garnett (16)", "Rondo (8)", "TD Banknorth Garden 18,624", "46\u201312"], [59, "March 5", "Detroit", "90\u201378", "Garnett (31)", "Perkins (20)", "Allen , Pierce , Rondo (5)", "TD Banknorth Garden 18,624", "47\u201312"], [60, "March 7", "Chicago", "116\u201393", "Pierce (22)", "Perkins (11)", "House , Pierce (5)", "TD Banknorth Garden 18,624", "48\u201312"], [61, "March 8", "@ Memphis", "119\u201389", "Allen (23)", "Perkins (11)", "Rondo (9)", "FedExForum 18,119", "49\u201312"], [62, "March 10", "@ Philadelphia", "100\u201386", "Garnett (26)", "Garnett (12)", "Rondo (8)", "Wachovia Center 20,438", "50\u201312"], [63, "March 12", "Seattle", "111\u201382", "Allen , Garnett (18)", "Allen (7)", "Pierce (11)", "TD Banknorth Garden 18,624", "51\u201312"], [64, "March 14", "Utah", "92\u2013110", "Garnett (15)", "Perkins (7)", "Garnett (6)", "TD Banknorth Garden 18,624", "51\u201313"], [65, "March 15", "@ Milwaukee", "99\u201377", "Garnett (19)", "Brown (9)", "Cassell (5)", "Bradley Center 17,626", "52\u201313"], [66, "March 17", "@ San Antonio", "93\u201391", "Pierce (22)", "Garnett , Pierce (8)", "Pierce (5)", "AT&T Center 18,797", "53\u201313"], [67, "March 18", "@ Houston", "94\u201374", "Garnett (22)", "Garnett (11)", "Pierce (5)", "Toyota Center 18,525", "54\u201313"], [68, "March 20", "@ Dallas", "94\u201390", "Pierce (22)", "Garnett , Pierce (13)", "Cassell (6)", "American Airlines Center 20,582", "55\u201313"], [69, "March 22", "@ New Orleans", "106\u2013113", "Pierce (28)", "Garnett (12)", "Rondo (7)", "New Orleans Arena 18,280", "55\u201314"], [70, "March 24", "Philadelphia", "90\u201395", "Garnett (18)", "Powe (7)", "Rondo (7)", "TD Banknorth Garden 18,624", "55\u201315"], [71, "March 26", "Phoenix", "117\u201397", "Garnett (30)", "Perkins (10)", "Allen (8)", "TD Banknorth Garden 18,624", "56\u201315"], [72, "March 28", "New Orleans", "112\u201392", "Pierce (27)", "Garnett (13)", "Pierce (9)", "TD Banknorth Garden 18,624", "57\u201315"]], "name": "table_11959669_7"}

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

실험은 -.jsonl 파일에서 한 줄씩 읽어들여 question과 table_id를 제안하는 방법의 입력으로 사용한 후 [그림 3-1]의 시스템 구성도와 같이 진행한다. 매칭 단계 중 Select절의 컬럼과 조건절의 매칭결과 값이 없을 경우 의문사구, 명사구와 DB Column list들에 대해 Stemming(스테밍) 작업 후 SQL을 생성한다. Stemming이란 어형이 변형된 단어로부터 접사 등을 제거한 후, 단어를 원형의 형태로 변환시키는 작업을 뜻한다. 영어의 문법은 단수형, 복수형에 따른 수일치 문제나 동사의 시제에 따라 표현이 다르므로 통일성을 위해 단어를 원형으로 변환시킨다.

[표 4-3] 예시 문장

입력문장	What positions do the hartford whalers nhl team have?
의문사구	**positions**
Column Name	“Pick”, “Player”, “**Position**”, “Nationality”, “NHL team”, “College/junior/club team”

[표 4-3]은 Stemming이 필요한 문장의 예시를 나타낸다. 입력문장에 대한 의문사구의 Chunking결과는 “positions”이다. 하지만 Column Name에는 “positions”의 원형인 “Position”이 있는 모습을 알 수 있다. DB에는 어떠한 단어의 복수형과 단수형 모두가 들어가 있을 수 있다. 그러므로 Stemming은 전처리 단계에서 미리 할 수 있지만, 미리 Stemming 과정을 거치게 되면 원하지 않는 결과가 나올 수 있어 전처리 단계에서는 Stemming 과정을 거치지 않는다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

N-gram 매칭 방법의 실험 결과의 Select절의 매칭 정확도는 약 95%가 나왔다. 매칭이 부정확한 데이터를 분석 한 결과 다음 [표 4-4]의 예시 문장 등에서 Select절의 Column을 매칭하지 못하였다.

[표 4-4] Select절 miss 매칭 예시

예시 문장	Select절의 정확한 Column값
Where does Route 24 intersect?	Location
Where can you find Colorado and Southern Railway #9?	Disposition
Who had the challenge of night driving?	Test-taker

[표 4-4]의 예시처럼 질의문에서 궁금한 사항을 의문사 자체가 주어가 되어 물어보는 질문들의 경우 Select절의 miss 매칭이 일어남을 볼 수 있었다. 또한 의문사구의 단어 표현이 DB의 Column name과 전혀 상이할 경우도 Select절의 miss 매칭이 일어났다. 위와 같은 문제를 보완하기 위해 WH- 각 의문사들에 대해 구축한 개체명사전을 기반으로 개체명 인식을 진행하여 정확도를 향상시켰다.

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

다음 [표 4-5]는 조건절의 miss 매칭 예시를 나타낸다.

[표 4-5] 조건절 miss 매칭 예시

예시 문장	
What is the name of the actress that was nominated for best actress in a leading role in the film Chopin: Desire for love?	
조건절 예측 값	정확한 조건절 값
[[0, 0, 'best actress in a leading role'], [2, 0, 'chopin: desire for love']]	[[2, 0, 'chopin: desire for love'], [0, 0, 'best actress in a leading role']]
예시 문장	
What is the UN operation title with the UN operation name, Uncok?	
조건절 예측 값	정확한 조건절 값
[[1, 0, 'un border relief operation'], [0, 0, 'uncok']]	[[0, 0, 'uncok']]
예시 문장	
What date did Adriano Buzaid have the pole position?	
조건절 예측 값	정확한 조건절 값
[[5, 0, 'adriano buzaid']]	[[3, 0, 'adriano buzaid']]

[표 4-5]의 첫 번째 예시 문장의 조건절 예측 값과 정확한 조건절 값을 비교하여 보면, 두 값 모두 일치하지만 순서가 다름을 알 수 있다. 이는 SQL문의 조건절을 생성할 경우 (조건1 and 조건2) 와 (조건2 and 조건1)과 같이 나타나게 되므로 SQL의 실행결과는 동일하지만 생성되는 SQL문은 다르게 나타남을 알 수 있다. 두 번째 예시 문장의 조건절 예측 값과 정확한 조건절 값을 비교하여 보면, 조건절 예측 값이 조건이 2가지가 생성되었지만, 정확한 조건절의 조건은 1가지뿐임을 알 수 있다. 세 번째 예시 문장의 경우는 두 값 모두 Column값은 일치하지만 Column의 인덱스가 다름을 알 수 있다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

A. 평가

실험의 평가는 Select절과 조건절의 각각의 매칭 정확도와 SQL문의 매칭 정확도를 각각 백분율로 측정하였다. 실험은 데이터셋의 질문이 WH-의문사로 시작하는 5000개의 질의문에 대해 각각의 Select절과 조건절을 매칭하여 SQL을 생성하였다. 질의문의 SQL변환의 매칭 정확도는 다음 [표 4-6]과 같다.

[표 4-6] 매칭 정확도 (%)

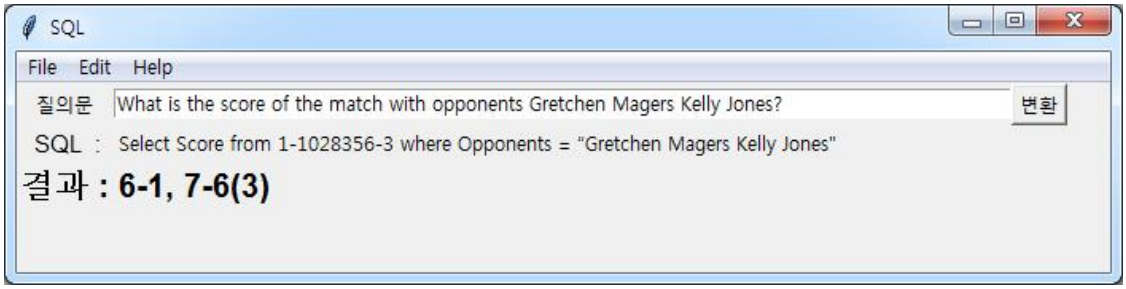
매칭 방법 \ SQL	SQL문	Select절	조건절
N-gram	0.7698	0.9512	0.7748
N-gram + 개체명 사전	0.7708	0.9618	0.7748
N-gram + 개체명 사전 + 조건절 보완	0.787	0.9618	0.812

Select절은 [표 4-4]의 예시와 같이 Select절의 매칭이 이루어지지 않았을 경우, WH- 의문문에 대한 개체명 사전을 통해 해당 질의문의 의문사와 매칭 되는 Column name을 Select절의 Column으로 선택하여 매칭 정확도를 향상시켰다. 조건절의 경우 질의문에서 추출한 명사구 중 Select절의 Column으로 선택된 Column name이 있는 경우 명사구 안의 선택된 Column name을 제거한 후 조건절의 매칭을 진행하여 [표 4-5]의 두 번째 예시와 같은 문제를 보완하였다. 또한 각 명사구와 매칭 되는 Column들 중 Column값이 같은 Column일 경우 Score뿐만이 아닌 Column name과 명사구와의 매칭을 통하여 Column을 선택하여 [표 4-5]의 세 번째 예시와 같은 문제를 보완하였다. 본 연구에서 제안한 방법의 SQL문 자동 생성 방법은 0.787(%)의 정확도를 보였다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

다음 [그림 5-1]은 본 연구를 이용한 NLIDB의 예시를 나타낸다.



[그림 5-1] NLIDB 예시

[그림 5-1]과 같이 본 연구를 통해 특별한 학습과정이나 DB에 관한 지식이 없는 일반인도 자연어로 된 질의문을 통해 쉽게 DB를 이용하여 원하는 정보를 검색할 수 있고, SQL을 공부하는 학습자를 위한 학습 도구로 사용할 수 있을 것으로 기대된다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

V. 결론 및 제언

본 논문에서는 DB와 SQL의 지식이 없는 일반인들도 일반적인 컴퓨팅 성능으로 DB를 이용하고 검색할 수 있는 방안을 목적으로 질의문의 자연어 처리를 통해 DB 검색이 가능하도록 SQL문으로의 변환 방법을 제안했다. SQL문의 Select절과 조건절을 보다 정확히 변환하고, 변환 시간을 단축하기 위해 질의문의 WH- phrase(의문사구)와 Noun phrase(명사구)를 추출하였다. 각 의문사구와 명사구에서 Select절의 Column name과 조건절의 Column값을 추출하여 SQL문을 생성하였고, 매칭 정확도와 결과를 분석하였다. N-gram을 통한 컬럼 매칭 정확도를 향상시키기 위해 WH- 의문사들에 대한 개체명 사전을 구축하여 Select절의 정확도를 향상시켰고, 조건절의 경우 중복되는 Column값들의 명사구와 Column name간의 비교를 통해 정확도를 향상 시킨 결과를 볼 수 있었다.

본 연구는 일반인들도 DB를 이용하고 검색할 수 있도록 질의문을 통해 SQL문 변환 방법을 제안했다. 실험 결과 Select절의 Column name 매칭의 경우 질의문의 의문사구에 찾고자 하는 데이터의 값을 정확히 입력 할 경우, 보다 정확한 Select절을 생성할 수 있었다. 조건절의 Column 매칭의 경우 찾고자 하는 데이터의 값들을 문장의 강조나 도치 등을 쓰지 않고 불필요한 용어가 없는 간결한 조건으로 이루어진 질의문으로 이루어 졌을 경우, 보다 정확한 조건절을 생성할 수 있었다.

향후 연구로는 Select절과 조건절을 질의문을 통해 보다 정확하게 생성하기 위한 규칙의 추가 및 수정을 통하여 보강할 예정이며, Select절의 경우 질의문의 의문사구 문법 규칙 등을 정규화하는 방법을 이용한다면 보다 빠르고 정확하게 프로세스 모델링을 할 수 있을 것이다. 또한 질의문의 SQL변환 방법을 이용하여 사용자들이 편리하고 용이하게 DB를 이용하고 검색할 수 있도록 다양한 기능이 추가된 편리한 유저인터페이스를 개발하여 NLIDB기술의 고도화를 목표로 한다.

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

참고문헌

[1] 이충권, “빅데이터 정보시스템의 구축 및 사례에 관한 연구”, 한국스마트미디어학회, Vol.4, No.3, pp.56-61, 2015.

[2] Bates M., R. Bobrow. “Information retrieval using a transportable natural language interface”, SIGIR '83 Proceedings of the 6th annual international ACM SIGIR conference on Research and development in information retrieval, pp.81-86, June 1983.

[3] Bates M., R. Bobrow, R. Ingria, D. Stallard. “The Delphi natural languageunderstanding system”, In Proceedings of the fourth conference on Applied naturallanguage processing, Oct 1994.

[4] Waltz D., “An English Language Question Answering System for a LargeRelational Database”, Communications of the ACM, Vol.21, No.7, July 1978.

[5] Martin P., D. Appelt, B. Grosz, F. Pereira, “TEAM: an experimental transportablenatural-language interface”, In Proceedings of 1986 on fall joint computer conference, Nov 1999.

[6] Barros, F., A. De Roeck, “Resolving Anaphora in a Portable NaturalLanguage Front End to a Database”, In Proceedings of the 4th Conference on AppliedNatural Language Processing, Stuttgart, Germany, pp.119-124, 1994.

[7] Templeton M., J. Burger, “Problems in Natural Language Interface to DBMSwith Examples from EUFID”, In Proceedings of the 1st Conference on Applied NaturalLanguage Processing, Santa Monica, California, pp.3-16, 1983.

[8] Hafner C., K. Godden. “Portability of syntax and semantics in DATALOG”, ACMTransactions on Information Systems (TOIS), Vol.3, No.2, pp.141-164, Apr 1985.

[9] Yates A., O. Etzioni, D. Weld, “A Reliable Natural Language Interface toHousehold Appliances”, In Proceedings of the Conference on IUI'03, Miami, Florida, USA, 2003

[10] 윤상호, 김영대, 황찬규, 유재형. “자연어 처리를 이용한 통계정보 검색 시스템 (NEIS) 개발”, 한국통신학회 종합 학술 발표회 논문집 (하계) 2007, pp.545-548,

Collection @ chosun

조선대학교 CHOSUN UNIVERSITY

2007.

[11] 한용진, 박세영, “질의 생성을 이용한 자연어 인터페이스”, 정보과학회논문지 : 소프트웨어 및 응용, 39(12), pp.998-1004, 2002.

[12] F. Meng and W. W. Chu, “Database Query Formation from Natural Language using Semantic Modeling and Statistical Keyword Meaning Disambiguation", CSD-TR 990003, University of California, June 1999.

[13] I. S. Kang, S. H. Na, J. H. Lee, and G. J. Yang, “Lightweight Natural Language database Interfaces", Proceedings of the 9th International Conference on Application of Natural Language to Information Systems. pp. 76-88, June 2004.

[14] J. H. Wang, H. J. Kim, and M. G. Jang, “Information Extraction Based on Property Patterns to Construct a Knowledgebase for Encyclopedia Person Domain", The 2004 KIISE Autumn Conference, Vol.31, No.2, pp.793-795, Nov 2004.

[15] R. Cooper, S. Ali, and C. Bi, “Extracting Information from Short Messages", NLDB 2005, LNCS 3513, pp.388-391, June 2005.

[16] B. H. Yun, “Natural Language Processing based Information Extraction for Newspapers", Journal of Korean Institute of Information Technology, Vol.6, No.4, pp.188-195, Aug 2008.

[17] C. N. Seon, H. S. Kim, and J. Y. Seo, “Information extraction using finite state automata and syllable n-grams in a mobile environme", Proceedings of the ACL-08: HLT Workshop on Mobile Language Processing, pp.13-18, June 2008.

[18] C. N. Seon, H. S. Kim, and J. Y. Seo, “Efficient appointment information extraction from short messages in mobile devices with limited hardware resources", Pattern Recognition Letters 32, pp.127-133, Jan 2011.

[19] Saikrishna Srirampur, Ravi Chandibhamar, Ashish Palakurthi, Radhika Mamidi. “Concepts identification of an NL query in NLIDB systems”, 2014 International Conference on Asian Language Processing (IALP), pp.230-233, Oct 2014.

[20] Victor Zhong, Caiming Xiong, Richard Socher, “Seq2SQL:Generating Structured Queries from Natural Language using Reinforcement Learning”, arXiv, Nov

Collection @ chosun

조선대학교
CHOSUN UNIVERSITY

2017.

[21] WikSQL[웹사이트], URL: https://github.com/salesforce/WikiSQL

[22] Ross R., “Constraints on Variables in Syntax, Doctoral Dissertation”, MIT (Distributed by the IULC), 1967.

Collection @ chosun