대형 언어 모델의 한국어 Text-to-SQL 변환 능력 평가

최주영°, 민경구, 심묘섭, 정해민, 박민준, 최정규
LG AI Research
{jooyoung.choi,kyungkoo.min,myoseop.sim,haemin.jung,minjun.park,stanleyjk.choi}@lgresearch.ai

Evaluation of Large Language Models' Korean-Text to SQL Capability

Jooyoung Choi°, Kyungkoo Min, Myoseop Sim, Haemin Jung, Minjun Park, Stanley Jungkyu Choi
LG AI Research

요 약

최근 등장한 대규모 데이터로 사전학습된 자연어 생성 모델들은 대화 능력 및 코드 생성 태스크등에서 인상적인 성능을 보여주고 있어, 본 논문에서는 대형 언어 모델 (LLM)의 한국어 질문을 SQL 쿼리 (Text-to-SQL) 변환하는 성능을 평가하고자 한다. 먼저, 영어 Text-to-SQL 벤치마크 데이터셋을 활용하여 영어 질의문을 한국어 질의문으로 번역하여 한국어 Text-to-SQL 데이터셋으로 만들었다. 대형 생성형 모델 (GPT-3 davinci, GPT-3 turbo) 의 few-shot 세팅에서 성능 평가를 진행하며, fine-tuning 없이도 대형 언어 모델들의 경쟁력있는 한국어 Text-to-SQL 변환 성능을 확인한다. 또한, 에러 분석을 수행하여 한국어 문장을 데이터베이스 쿼리문으로 변환하는 과정에서 발생하는 다양한 문제와 프롬프트 기법을 활용한 가능한 해결책을 제시한다.

주제어: Text-to-SQL, 데이터베이스 질의 생성, SQL 쿼리 변환, 프롬프트, Prompt

1. 서론

최근 대규모 코퍼스로 사전학습된 생성형 대형 언어 모델 (Large language model, LLM)에 대한 관심이 높아지고 있다. 이러한 언어 모델들은 파라미터 규모의 증가와 함께 자연어 처리 분야에서 더욱 필수적인 역할을 하며, 사용법 또한 점차 점차 변화하고 있다. 특히 GPT-3 [1] 언어 모델은 fine-tuning 학습 없이도 프롬프트 디자인을 통해 원하는 결과물을 생성한다. 최근에 등장한 ChatGPT[1] 모델은 Reinforcement Learning for Human Feedback (RLHF) [2] 방법론을 활용하여 사전훈련된 모델의 zero-shot 성능을 더 효과적으로 활용하며, 이에 따라 GPT-3 모델의 zero-shot, few-shot 성능을 다양한 자연어 처리 작업에서 평가하고 분석하는 연구가 시작되었다. 이전에는 모델이 우수한 성능을 달성하기 위해 대량의 어노테이션 데이터가 필요하며, 이러한 데이터를 구축하기 위해 많은 시간과 비용이 소요되었다. 이러한 제약 사항을 고려할 때, zero-shot 코드 생성 모델은 매우 중요한 역할을 한다.

Text-to-SQL 태스크는 사용자의 입력 텍스트를 데이터베이스에서 실행 가능한 SQL 쿼리로 변환하는 작업이다. 이 작업은 데이터베이스에 대한 전문 지식이 없는 비전문가 사용자들이 데이터베이스에 쉽게 접근할 수 있도록 도와주는 업무적으로 중요한 작업으로 인식되며, 이에 관한 연구 및 비즈니스 관심이 꾸준히 증가하고 있다.이와 관련하여 해외에서는 다양한 챌린지와 연구가 계속 진행되고 있다 [3, 4, 5, 6]. 전통적인 Text-to-SQL 방법은 주로 인코더-디코더 모델을 fine-tuning하는 방식을 채택해왔다. 그러나 이러한 fine-tuning 학습은 텍스트-SQL 쌍을 포함한 학습 데이터셋을 필요로 하는데, 데이터 구축하는 비용이 많이 드는 단점이 있습니다. 또한, 이러한 fine-tuning 학습은 학습 데이터에 대한 모델의 오버피팅을 유발할 수도 있다. 이러한 한계를 극복하기 위해 본 연구에서는 GPT-3 모델의 zero-shot 및 few-shot 학습 방법을 활용하여 Text-to-SQL 태스크를 수행하며, 성능 평가 및 분석을 통해 다양한 프롬프트 기법을 연구하고자 한다.

본 연구에서는 GPT-3 모델의 zero-shot, few-shot 세팅에서 한국어 질문을 SQL 쿼리문으로 변환하는 성능을 평가한다. 국내에는 한국어 관련 Text-to-SQL 벤치마크 데이터셋이 공개되어 있지 않기 때문에[2], 영어 벤치마크 데이터셋 Spider [3]을 활용한다. 이를 위해 영어 질문만 한국어로 번역하여 한국어 질문-SQL 쿼리문 쌍으로 학습 (train) 및 검증 (dev) 데이터셋을 생성한다. 이러한 접근은 테이블 이름, 필드, 데이터 유형, 엔티티 간의 관계 및 제약조건 등을 나타내는 데이터베이스 스키마가 영어로 표시되어 있기 때문이다. 본 실험에서는 GPT-3 davinci, GPT-3 turbo 모델을 사용하여 한국어 질의를 SQL 쿼리문으로 변환하는 성능을 평가하며, 이들보다 규모가 작은 본 연구원에서 자체 개발한 1.7B 언어 모델을 학습 데이터셋을 사용하여 fine-tuning하고 이를 통해 성능을 비교한다.

[1] https://chat.openai.com

[2] 2023년 4월에 AIHub (https://www.aihub.or.kr/) 에서 자연어 기반 질의(NL2SQL) 검색 생성 데이터를 한시적으로 공개하였으나 데이터 활용성 검토를 위해 현재 (2023년 9월) 비공개 되어 있다.

2. 관련 연구

기존의 Text-to-SQL 방법론으로는 규칙 기반 (Rule-based)과 fine-tuning 학습 방법으로 나눌 수 있다. 규칙 기반은 잘 설계된 템플릿을 활용하여 SQL 쿼리를 생성하는 방법론으로서, 특정한 경우에는 좋은 성능을 나타내지만 수동으로 설계된 규칙들에 크게 의존하여 다른 도메인 적용이 어렵거나 모델의 일반화에 한계가 있다 [7, 8]. 이러한 문제점들을 해결하기 위해 LSTM [9] 및 CNN [10] 기반의 Text-to-SQL 방법론이 연구되었다. 이러한 방법론들은 규칙 기반보다 더 높은 성능을 얻었지만 데이터베이스 구조 정보를 이해하는데 어려움이 있다. 이를 위해 데이터베이스 스키마를 그래프화 하여 그래프 신경망 모델 (Graph neural network)을 활용한 모델들도 연구되었다 [11, 12]. 또한, T5 언어 모델 [13]을 활용한 연구들도 진행되었는데 텍스트를 SQL 쿼리로 변환하는 태스크에서 기존의 모델들보다 더 좋은 성능을 나타낸다 [14, 15]. 하지만 fine-tuning 방법론은 해당 태스크에 맞는 대량의 어노테이션된 학습 데이터가 필요하며, 이러한 학습 데이터에 모델이 오버피팅되기 쉬운 한계점이 존재한다.

최근에 등장한 대형 언어 모델(LLM)은 Text-to-SQL 태스크에 대한 새로운 방법론을 제시하고 있다. 이는 in-context learning 가능한 GPT 계열의 생성형 언어 모델이 일부 자연어 처리 태스크에서 zero-shot, few-shot 세팅에서 fine-tuning 성능을 능가하고 있음을 보여주었기 때문이다 [1]. 일부 연구에서는 다양한 프롬프트 활용하여 zero-shot 세팅에서 Text-to-SQL 성능을 평가하지만, 기존의 fine-tuning 성능에 못 미친다 [16, 17]. 하지만, 특정 태스크에 특화된 프롬프트를 정교하게 설계함으로써 대형 언어 모델은 더 우수한 성능을 발휘할 수 있는 잠재력을 지니고 있다. 최근에는 zero-shot 성능의 오류 분석을 통해 GPT-4 모델을 활용하여 특정 태스크에 특화된 프롬프트를 개발하고 few-shot 연구를 수행하였다. 이 방법은 Spider 챌린지에서 최고 성능 (SOTA)을 달성하였다[18]. 본 연구에서는 한국어 Text-to-SQL 변환 태스크에 대한 대형 언어 모델의 성능을 평가하고, 또한 에러 분석을 통해 Text-to-SQL 변환 작업에서 발생하는 문제들를 심층적으로 조사한다. 이 작업에서 발생하는 다양한 문제와 이를 해결하기 위한 프롬프트 디자인에 대한 통찰을 제시한다.

3. 방법론

3.1 Text-to-SQL 태스크 설명

한국어 질문을 SQL 쿼리로 변환하는 태스크에서는 주어진 자연어 질문 $\mathcal{Q}$와 데이터베이스 스키마 $\mathcal{S}$의 집합으로 이루어진 데이터가 주어진다. 데이터베이스 스키마 $\mathcal{S} = \{\mathcal{T}, \mathcal{C}, \mathcal{R}\}$는 여러 개의 테이블 $\mathcal{T}$, 컬럼 $\mathcal{C}$ 및 외래 키 (Foreign key) 관계 $\mathcal{R}$을 포함하고 있다. 주어진 데이터로부터 질문에 해당하는 SQL 쿼리 $\mathcal{Y}$를 생성하는 것을 목표로 한다.

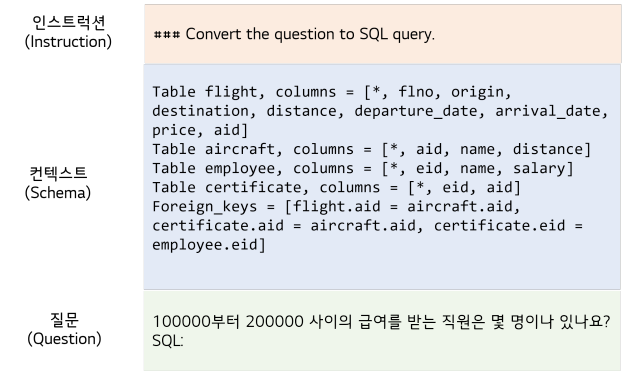

그림 1. 한국어 Text-to-SQL 태스크에 대한 GPT-3 모델 zero-shot 프롬프트 예제

3.2 프롬프트

GPT-3 모델들이 정확한 SQL 쿼리문을 생성할 수 있도록 그림 1과 같이 일반적인 프롬프트를 사용한다. 프롬프트에는 질문과 질문에 관련된 데이터베이스 스키마 정보를 입력한다. Few-shot 세팅에서는 데이터셋 구성 특징을 고려하여 SQL 쿼리 난이도별로 컨텍스트를 구성하고 GPT-3 davinci와 GPT-3 turbo 모델의 입력으로 넣어준다.

4. 실험

본 장에서는 실험에 사용된 데이터셋, 실험 방법, 평가 방법, 그리고 오류 분석을 포함한 실험 결과에 대하여 보고한다.

4.1 데이터셋

이 연구에서는 앞서 설명한 바와 같이 공개된 한국어 Text-to-SQL 데이터셋이 없기에 Spider 데이터셋 [3]을 활용하였다. 해당 데이터셋은 다양한 도메인 데이터베이스에 대한 자연어 인터페이스를 개발을 목표로 하는 Spider 챌린지[3]를 위해 공개된 영어 Text-to-SQL 벤치마크 데이터셋으로, 138개의 다른 도메인을 포함하는 200개의 데이터베이스에서 멀티 테이블을 다루는 10,181개의 자연어 질의문과 5,693개의 고유한 SQL 쿼리가 쌍으로 구성되어 있다.

실험을 위해 먼저 Spider 데이터셋의 영어 질문만 GPT-3 davinci 모델을 사용하여 한국어로 번역한다. 실제 데이터베이스에서 테이블 이름, 데이터 유형, 필드 및 제약조건 등은 영어로 사용되고 있어, SQL문과 데이터베이스 스키마는 영어 그대로 사용한다. 한국어 질의문과 SQL 쿼리문 쌍으로 이루어진 한국어 Text-to-SQL 데이터셋을 새롭게 구성하였고, train

[3]https://yale-lily.github.io/spider

표 1. 각 모델별 한국어 Spider Dev 데이터셋에 대한 execution accuracy (EX), exact match accuracy (EM), test-suite accuracy (TS) 점수 결과 비교

모델	Zero-Shot	Few-shot	Fine-tuning	EX	EM	TS
GPT-3 turbo	√	√		42.6	35.8	**51.1**
GPT-3 davinci	√	√		**45.0**	36.3	48.0
Exaone 1.7B			√	44.4	**42.7**	42.7

표 2. SQL 쿼리 난이도에 따른 각 모델별 실행 정확도 (EX) 성능 비교

모델	쉬움(Easy)	중간(Medium)	어려움(Hard)	매우어려움(Extra-hard)	(Overall)
GPT-3 turbo	62.5	44.0	31.0	16.9	42.1
GPT-3 davinci	**65.3**	**46.0**	34.5	22.3	**45.0**
Exaone 1.7B	64.1	43.7	**37.4**	**24.1**	44.4

8,659 건, dev 1,034 건으로 나눠서 학습 및 모델 성능평가에 사용한다.

4.2 평가 방법

평가 방법으로는 유효한 SQL (Valid SQL, VA), 실행 정확도 (Execution accuracy, EX), 정확도 (Exact match accuracy, EM), 그리고 테스트-스위트 정확도(Test-suite accuracy, TS) 네 가지 평가 점수를 활용한다. 유효한 SQL (VA)은 성공적으로 실행할 수 있는 SQL 문의 비율을 나타내고, 실행 정확도 (EX)는 실행 결과가 ground-truth SQL 결과와 일치하는 비율을 나타낸다. 정확도 (EM)는 모델 예측값이 ground-truth SQL 쿼리와 정확히 일치하는지를 평가한다. 동일한 질문에 대한 SQL 쿼리는 여러 다른 방식으로 표현될 수 있기 때문에 좀 더 유연한 성능 평가를 위해 ground-truth SQL 쿼리와 모델 예측값이 의미적으로 동일한지를 평가하는 테스트-스위트 정확도 (TS) [19] 평가 방법을 추가하였다.

4.3 실험 방법

대용량 텍스트 데이터로 학습된 강력한 언어 모델인 GPT-3.5 [20] 시리즈 모델인 text-davinci-003과 turbo를 사용하여 dev 데이터셋으로 zero-shot과 few-shot 실험을 수행한다. Zero-shot/few-shot 성능을 fine-tuning 모델과 비교하기 위해, 본 연구원에서 자체 개발한 1.7B개의 학습 파라미터를 가지는 GPT 계열의 한국어\영어 생성형 언어 모델을 train 데이터셋으로 fine-tuning 한다. Deepspeed[4] 와 Huggingface[5]를 이용하여 A100 40GB GPU 8장으로 10 에폭 (epoch) 학습시켰다.

[4]https://github.com/microsoft/DeepSpeed
[5]https://huggingface.co/

4.4 실험 결과

표 2는 한국어 Spider dev 데이터에 대한 few-shot 실험 결과로, GPT-3 turbo 및 GPT-3 davinci 모델과 train 데이터로 fine-tuning 된 Exaone 모델의 성능을 나타낸다. 여기서 fine-tuning 학습된 1.7B 모델은 정확도 (EM) 면에서 GPT-3 모델을 능가하는 성능을 보여주고 있다. 그러나 모델 예측의 의미적 정확도 평가에서는 GPT-3 turbo 모델이 우수한 성능을 나타내고 있다. 이는 175B 개의 거대한 학습 파라미터를 가진 GPT-3 turbo 모델이 내재된 지식을 활용하여 입력 텍스트를 더 잘 이해하고 SQL로 변환하는 데 도움이 되고 있음을 시사한다.

Spider 데이터셋의 SQL 쿼리는 다음과 같은 난이도로 구분된다: 쉬움 (Easy), 중간 (Medium), 어려움 (Hard), 매우 어려움 (Extra-hard). 각 난이도에 따른 모델 성능은 표 2에서 보여준다. GPT-3 davinci 모델은 쉽거나 중간 단계의 SQL 쿼리 생성 성능에서 fine-tuning 1.7B 모델보다 우수한 성능를 보이지만, 특히 어렵거나 매우 어려운 쿼리에 대해서는 fine-tuning 학습된 모델이 더 좋은 SQL 변환 성능을 보여주고 있다.

한국어 Spider 데이터셋에 대한 fine-tuning 및 GPT-3의 few-shot 실험에서 전반적으로 낮은 성능이 관찰되었습니다. 특히, GPT-3 davinci 모델이 few-shot 세팅에서 ground-truth 쿼리문과 다른 쿼리문을 생성하는 이유에 대한 깊은 이해를 위해, 다음 장에서는 테스트-스위트 정확도 (TS) 실패 사례를 세밀하게 분석하고 분류한다.

4.5 에러 분석

그림 2는 GPT-3 davinci 모델의 few-shot 실험에서 테스트-스위트 정확도 (TS)를 실패 사례를 수동으로 분석하고 6개의 범주로 분류한 오류 분석결과를 보여준다. 이는 선행 연구 논

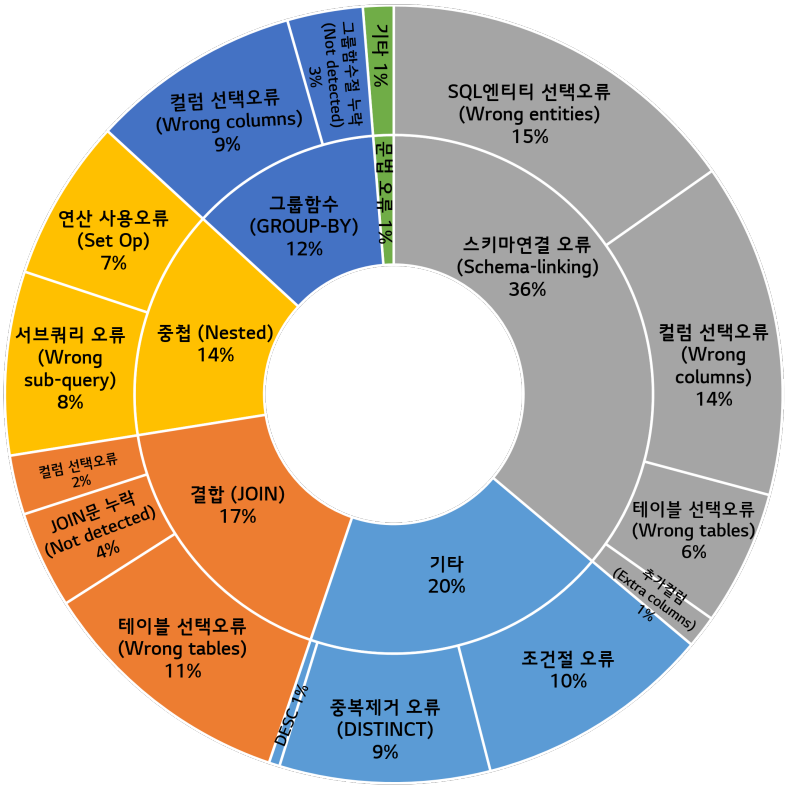

그림 2. 한국어 Text-to-SQL dev 데이터셋에 대한 GPT-3 davinci 모델의 SQL 쿼리 생성 오류 532건에 대한 통계. 본 실험에서는 기존 연구 방법 [18]을 채택하여 심층적인 SQL 쿼리 오류 분석을 수행.

스키마연결 (Schema-linking)	
질문 1	1930년 이전에 설립된 국가들 중에서 공식 언어의 총 개수는 얼마인가요? (For the countries founded before 1930, what is the total number of distinct official languages?)
정답 SQL	SELECT count(DISTINCT T2.Language) FROM country AS T1 JOIN countrylanguage AS T2 ON T1.Code = T2.CountryCode WHERE IndepYear < 1930 AND T2.IsOfficial = "T"
예측 SQL	SELECT SUM(COUNT(DISTINCT language)) FROM countrylanguage AS t1 JOIN country AS t2 ON t1.CountryCode = t2.Code WHERE t2.IndepYear < 1930 AND t1.IsOfficial = 'T'
질문 2	가장 많은 콘서트가 열린 연도는 언제인가요? (What is the year that had the most concerts?)
정답 SQL	SELECT YEAR FROM concert GROUP BY YEAR ORDER BY count(*) DESC LIMIT 1
예측 SQL	SELECT year, COUNT(*) FROM concert GROUP BY year ORDER BY COUNT(*) DESC LIMIT 1
질문 3	직원 수를 세어주세요. (Count the number of employees)
정답 SQL	SELECT count(*) FROM employee
예측 SQL	SELECT COUNT(DISTINCT Employee_ID) FROM hiring
질문 4	USA 국적을 가진 모든 선수들의 이름과 생년월일을 나열해주세요. (List the first name and birth date of all players from the country with code USA.)
정답 SQL	SELECT first_name , birth_date FROM players WHERE country_code = 'USA'
예측 SQL	SELECT first_name, last_name, birth_date FROM players WHERE country_code = 'USA'

그림 3. 스키마연결 (Schema-linking) 오류 예제

문에서 제안한 오류 분석 방법을 따른다 [18].

첫째로, 다빈치 모델은 스키마 연결 (Schema-linking) 오류가 가장 흔하게 발생하는데, 이러한 오류가 모델의 예측에서 주요한 비중을 차지한다. 스키마 연결 오류란 모델이 자연어 질문에서 언급된 테이블 이름, 컬럼 이름 또는 집계 함수 (aggregation function)와 같은 엔터티를 정확하게 구분하지 못한 경우를 나타낸다. 특히, 추가 컬럼 (Extra columns) 오류는 정답 SQL 쿼리보다 더 많은 유용한 컬럼을 쿼리에 포함하는 경우를 의미한다. 이 오류 범주에 속한 모델의 예측 SQL 예제는 그림 3에서 확인할 수 있다. 질문1의 예시는 엔터티 선택 (Wrong entities) 오류를 보여주며, 질문2는 추가 컬럼 (Extra columns) 오류를 나타낸다. 이는 GPT 계열 모델이 특히 수량과 관련된 질문에 대해 질문과 관련된 컬럼을 선택하는 경향이 있음을 보여준다. 질문3은 테이블 및 컬럼 선택 오류에 대한 예시를 제시하고, 질문4 예시는 추가 컬럼 오류에 속하지만 근본적으로 영어 - 한국어 질문 번역 문제로 인해 발생한 것이다. 예를 들어, 영어 질문 “first name” 이 “이름” 으로 번역되어 “first name, last name” 컬럼들이 모델로부터 선택되어 오답 처리된다.

두번째로 빈도가 높은 모델의 SQL 쿼리 생성 오류 범주는 기타로, 조건문이 추가되어 있거나 조건문을 놓친 조건절 오류들과, 중복 제거 (DISTINCT) 키워드가 빠진 경우 또는 중복된 쿼리와 같은 문제들이 포함되어 있다.

결합 (JOIN) 범주는 JOIN이 필요한 쿼리를 포함한다. 모델이 JOIN에 필요한 테이블들을 질문으로부터 제대로 식별하지 못한 테이블 선택 오류 (Wrong tables)가 11%를 나타내며, 또한 JOIN문을 누락한 오류 (Not detected)도 4% 차지한다.

난이도가 어려움 (Hard), 매우 어려움 (Extra-hard)인 SQL 쿼리에는 대게 중첩된 쿼리문이 다수 포함되어 있다. 이러한 중첩 (Nested) 오류로는 하위 쿼리 (sub-query)의 중첩이나, 집합 연산자 (UNION, INTERSECT)가 필요한 경우가 있으며, 모델이 올바른 중첩 구조를 파악하지 못하거나 (Wrong sub-query) 집합 연산을 인식하지 못하는 경우들이 포함된다.

그룹함수 (GROUP-BY) 범주에는 GROUP-BY 절이 필요한 쿼리를 포함하며, 모델이 그룹화의 필요성을 감지하지 못하거나 (Not detected), 또는 그룹화하기 위해 잘못된 컬럼을 사용한 경우 (Wrong columns)가 포함된다.

이 실험에서는 대형 언어 모델의 헌국어 텍스트를 SQL 쿼리로 변환하는 능력을 평가하기 위해 일반적인 프롬프트만을 사용하였다. 위에서 제시된 심층적인 오류 분석을 토대로, 향후 연구에서는 한국어 Text-to-SQL 태스크를 더 효과적으로 수행하기 위한 태스크 특화된 프롬프트를 개발하는 방향으로 연구하고자 한다.

5. 결론

본 연구에서는 한국어 Text-to-SQL 변환 태스크에 대한 다양한 실험을 수행하고 평가하였다. 실험 결과를 통해 GPT-3 모델과 fine-tuning 모델 간의 성능 차이를 확인하였으며, few-shot 세팅의 의미적 정확도 면에서는 GPT-3 turbo 모델이 좋은 성능을 보이고 있음을 보고하였다. 또한 오류 분석을 통해 여러 종류의 SQL 오류를 심층적으로 조사하였으며, 특히 스키마연결 오류와 중첩 오류가 많이 발생하였다. 이를 토대로 향후 연구에서는 더 나은 프롬프트 디자인과 모델 개선을 통해 한국어 Text-to-SQL 작업의 성능을 향상시키고자 한다. 본 연구에서는 영어 데이터인 Spider 벤치마크 데이터셋을 번역하여 실험을 진행하였으나, 한국어 Text-to-SQL 데이터셋이 구축되어 본 연구에서 제시한 향후 개선 방법론이 실제로 적용될 수 있기를 기대한다.

참고문헌

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, Vol. 33, pp. 1877–1901, 2020.

[2] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, “Deep reinforcement learning from human preferences,” *Advances in neural information processing systems*, Vol. 30, 2017.

[3] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman *et al.*, “Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task,” *arXiv preprint arXiv:1809.08887*, 2018.

[4] T. Yu, R. Zhang, M. Yasunaga, Y. C. Tan, X. V. Lin, S. Li, H. Er, I. Li, B. Pang, T. Chen *et al.*, “Sparc: Cross-domain semantic parsing in context,” *arXiv preprint arXiv:1906.02285*, 2019.

[5] T. Yu, R. Zhang, H. Y. Er, S. Li, E. Xue, B. Pang, X. V. Lin, Y. C. Tan, T. Shi, Z. Li *et al.*, “Cosql: A conversational text-to-sql challenge towards cross-domain natural language interfaces to databases,” *arXiv preprint arXiv:1909.05378*, 2019.

[6] V. Zhong, C. Xiong, and R. Socher, “Seq2sql: Generating structured queries from natural language using reinforcement learning,” *CoRR*, Vol. abs/1709.00103, 2017.

[7] J. M. Zelle and R. J. Mooney, “Learning to parse database queries using inductive logic programming,”

Proceedings of the national conference on artificial intelligence, pp. 1050–1055, 1996.

[8] D. Saha, A. Floratou, K. Sankaranarayanan, U. F. Minhas, A. R. Mittal, and F. Özcan, "Athena: an ontology-driven system for natural language querying over relational data stores," *Proceedings of the VLDB Endowment*, Vol. 9, No. 12, pp. 1209–1220, 2016.

[9] J. Guo, Z. Zhan, Y. Gao, Y. Xiao, J.-G. Lou, T. Liu, and D. Zhang, "Towards complex text-to-sql in cross-domain database with intermediate representation," *arXiv preprint arXiv:1905.08205*, 2019.

[10] D. Choi, M. C. Shin, E. Kim, and D. R. Shin, "Ryansql: Recursively applying sketch-based slot fillings for complex text-to-sql in cross-domain databases," *Computational Linguistics*, Vol. 47, No. 2, pp. 309–332, 2021.

[11] B. Wang, R. Shin, X. Liu, O. Polozov, and M. Richardson, "Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers," *arXiv preprint arXiv:1911.04942*, 2019.

[12] R. Cao, L. Chen, Z. Chen, Y. Zhao, S. Zhu, and K. Yu, "Lgesql: line graph enhanced text-to-sql model with mixed local and non-local relations," *arXiv preprint arXiv:2106.01093*, 2021.

[13] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *The Journal of Machine Learning Research*, Vol. 21, No. 1, pp. 5485–5551, 2020.

[14] T. Scholak, N. Schucher, and D. Bahdanau, "Picard: Parsing incrementally for constrained auto-regressive decoding from language models," *arXiv preprint arXiv:2109.05093*, 2021.

[15] H. Li, J. Zhang, C. Li, and H. Chen, "Resdsql: Decoupling schema linking and skeleton parsing for text-to-sql," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37, No. 11, pp. 13 067–13 075, 2023.

[16] N. Rajkumar, R. Li, and D. Bahdanau, "Evaluating the text-to-sql capabilities of large language models," *arXiv preprint arXiv:2204.00498*, 2022.

[17] A. Liu, X. Hu, L. Wen, and P. S. Yu, "A comprehensive evaluation of chatgpt's zero-shot text-to-sql capability," *arXiv preprint arXiv:2303.13547*, 2023.

[18] M. Pourreza and D. Rafiei, "Din-sql: Decomposed in-context learning of text-to-sql with self-correction," *arXiv preprint arXiv:2304.11015*, 2023.

[19] R. Zhong, T. Yu, and D. Klein, "Semantic evaluation for text-to-sql with distilled test suite," *The 2020 Conference on Empirical Methods in Natural Language Processing*, 2020.

[20] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems*, Vol. 35, pp. 27 730–27 744, 2022.