Text-to-CSEQL: Taking a Step in Natural Language Search for Cyberspace Assets Using Large Language Model

Anonymous ACL submission

Abstract

Cyberspace search engines (CSEs) are systems designed to search and index information about network assets in cyberspace. However, using CSEs to grasp the status of cyberspace assets still encounters challenges in usability, due to the specificity of terminology and the professionalism of cyberspace search engine query language (CSEQL). To improve the usability of CSEs, it is essential to support natural language querying. To this end, we propose a task called Text-to-CSEQL, which aims to translate natural language into CSEQL. We propose a method based on the large language model (LLM) to enable natural language interaction with CSEs. Specifically, we adopt retrieval-augmented generation (RAG) techniques with LLM by constructing a knowledge base. Upon receiving a natural language input, it extracts relevant fields and examples from the knowledge base, crafting a well-formed prompt for LLM. To comprehensively assess the method, we construct a dataset related to Text-to-CSEQL and design a new domain-specific evaluation metric, called Field Match (FM). Extensive experiments demonstrate that our framework is highly effective, outperforming existing methods. In addition, our method is adaptable and can accommodate various CSEs.

1 Introduction

A cyberspace search engine (CSE) (Li et al., 2020) is a specialized information retrieval system designed to search, index, and organize internet-connected resources. CSEs are becoming increasingly vital in areas like cyberspace asset discovery and management, cybersecurity risk assessment and monitoring, etc. Currently, there are numerous CSEs, each with its own version of cyberspace search engine query language (CSEQL). This implies that accurate and efficient construction of query statements requires a high level of specialized knowledge, posing challenges for users.

Background Knowledge:

Field	Example	Description	=	!=	*=
port	port=6379	Query by open port number.	✓	✓	✓
is_honeypot	is_honeypot=true	Filter assets that are honeypots.	✓	–	–

Natural Language Input:

How do I find honeypot network assets with port 3306 open?

FOFA Query:

port=3306 && is_honeypot=True

Figure 1: An example of natural language input is converted into the corresponding CSE query statement for FOFA based on background knowledge. In the example FOFA query, blue represents query fields, red denotes query values, and green is logical symbol.

Therefore, it is essential to enable natural language interaction with CSEs.

Text-to-SQL (Deng et al., 2022; Gao et al., 2024; Mao et al., 2024; Xie et al., 2024; Wretblad et al., 2024) aims to translate natural language into SQL statements, enabling users to query vast amounts of data stored in relational databases without needing to write complex SQL code. Inspired by Text-to-SQL, transforming natural language into CSEQL is a practical approach to improve the convenience of the interactions with CSEs. An example of transformation is presented in Figure 1. Despite both being query languages, SQL and CSEQL serve distinct purposes and exhibit significant differences. In essence, SQL is primarily designed for querying and managing data in relational databases, whereas CSEQL is for querying cyberspace assets in CSEs. These differences are reflected in their syntax: SQL has a unified, standardized syntax, while CSEQL syntax is platform-specific, with each platform employing customized syntactic forms. As a result, directly applying Text-to-SQL methods to Text-to-CSEQL is challenging due to the inherent differences between the two languages.

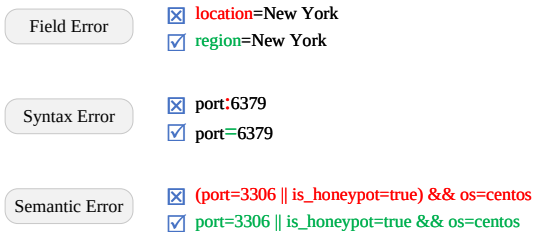

Figure 2: An overview of the errors when LLMs generate FOFA query statements. In the figure, "×" indicates error cases, while "✓" denotes correct answers. The red represents the incorrect parts generated by LLM and the green represents the correct parts of the generated query.

To effectively implement Text-to-CSEQL, the method should be able to accurately capture the relationship between each natural language query and its corresponding CSEQL statement. Large Language Models (LLMs) offer substantial benefits in processing and comprehending natural language. Intuitively, they could be applied to this task. Nevertheless, due to the hallucinations of LLMs, directly using LLMs for this task results in numerous errors. Figure 2 highlights three prevalent errors encountered when utilizing LLMs to generate FOFA query statements: (1) *field error*. In FOFA query language, specific terms like "region" are required for searching assets in particular areas, rather than using synonyms such as "location." (2) *syntax error*. The generated query statements must adhere strictly to syntactic requirements to ensure their validity. (3) *semantic error*. Minor linguistic discrepancies can result in substantial semantic misinterpretations, leading to incorrect query outcomes.

To address these issues, we design a method that incorporates knowledge base construction, retrieval-augmented generation, and prompt design. By harnessing the contextual learning ability of LLMs, our method involves retrieving pertinent query fields via field retrieval and leveraging related few-shot examples through few-shot retrieval from the knowledge base. Furthermore, we craft a meticulously designed prompt, adapted from COSTAR (Teo, 2024), to ensure precise generation of CSE query statements using LLMs. This comprehensive integration of retrieved information and prompt engineering enhances the effectiveness and accuracy of our method.

Since there is no existing dataset for evaluation, we have created a new dataset. The dataset contains natural language inputs and corresponding CSE query statements of different CSEs. All query statements are collected from real world, written by users to meet legitimate information needs. We leverage LLMs to generate natural language descriptions of query statements, which are subsequently verified by experts. Furthermore, we introduce a domain-specific metric called field match (FM) to evaluate our method comprehensively. This metric assesses the accuracy of the fields in the generated query statements.

Our main contributions are summarized as follows:

- We propose the Text-to-CSEQL task with the motivation of enhancing usability of CSEs with natural language interactions.
- We introduce a method to generate CSEQL based on LLMs. This method leverages RAG to alleviate hallucination through few-shot retrieval and field retrieval.
- We construct a dataset for evaluation. The dataset contains natural language inputs and real-world CSE query statements across multiple platforms, including Shodan, ZoomEye, Censys, and FOFA. Moreover, a domain-specific metric called FM is introduced to evaluate our method comprehensively.
- Extensive experiments on the dataset show that our method outperforms existing methods and can adapt to the syntax variations of different platforms. Ablation studies validate the contribution of each module in our method.

2 Background

2.1 Cyberspace Search Engine

Cyberspace search engines (CSEs) have emerged as powerful tools for exploring, analyzing, and understanding the vast array of devices, services, and data residing within cyberspace. CSEs differ fundamentally from traditional web search engines such as Google, Baidu, or Bing. While traditional search engines primarily index and retrieve web pages, CSEs are designed to actively probe and detect entities and services within cyberspace. They utilize a variety of protocols and techniques to discover open ports, services, and devices, and they provide detailed information about these entities, including their location, configuration, and security

Difference	Cyberspace Search Engines			
	Shodan	ZoomEye	Censys	FOFA
Matching symbol	:	:	:	=
Boolean logic symbol	space , -	+ space -	and or not	&& || !=
Field names,such as port	port	port	services.port	port
Number of fields	81	23	2406	77

Table 1: Comparison of CSEQL syntax differences across four CSEs.

posture. Several notable CSEs are designed in recent years, including Shodan (Sho), Censys (Cen), ZoomEye (Zoo) and FOFA (FOF).

2.2 Cyberspace Search Engine Query Language and Syntax

Cyberspace search engine query language (CSEQL) is a language used to retrieve cyberspace assets information that meet specific conditions on CSEs. Each CSE usually has its customized implementation of CSEQL, featuring distinct syntax rules. Table 1 shows the comparison of the difference in CSEQL syntax across four CSEs. When using CSEs, users have to construct the query statements that statisfy the corresponding syntax rules. A query statement typically consists of query conditions and logical connectors. Query conditions are composed of three parts: query field, matching symbol, and query value. Logical connectors include three types of logic operators: AND, OR and NOT. Query fields are divided into simple and nested fields based on their level of granularity. Matching symbols are categorized by precision into exact match, match, wildcard match, and mismatch. The query value depends on the query field, and the number of available choices can be classified as binary values, finite numbers greater than two, or uncertain content. For instance, a query statement of FOFA is as follows: is_honeypot=true && port=3306, where "is_honeypot" and "port" are query fields, "=" is the matching symbol, and "true" and "3306" are query values. The entire query statement is designed to search for honeypot assets with port 3306 open.

3 Methods

To describe the task accurately, we formulate it as a conditional probability problem. For a query task $\mathcal{T}$ described in natural language and its corresponding cyberspace search engine $\mathcal{S}$, the objective of LLM is to predict the CSE query statement from $\mathcal{T}$. The probability that LLM predict CSE query statement(q) can be defined as a conditional probability distribution. $\mathcal{I}(\mathcal{S})$ represents the relevant syntax of the search engine $\mathcal{S}$, $\mathcal{P}$ represents the prompt template, and |q| denotes the length of the predicted CSE query. q_i and $q_{<i}$ represent the i-th token and the prefix of q_i, respectively. The parameters of LLM $\mathcal{M}$ are denoted by $\mathbf{\Phi}$. The conversion from natural language to CSE query statements using the LLMs is represented as follows:

$$\mathbb{P}_{\mathbf{\Phi}}\left(q|\mathcal{P}(\mathcal{T},\mathcal{I}(\mathcal{S}))\right)=\prod_{i=1}^{|q|}\mathbb{P}_{\mathbf{\Phi}}\left(q_i|\mathcal{P}(\mathcal{T},\mathcal{I}(\mathcal{S})),q_{<i}\right). \quad (1)$$

3.1 Overview

The overview of our proposed method is illustrated in Figure 3. The LLM predicts CSE query statements based on natural language input. The first step involves the construction of the knowledge base. It encompasses CSE query statements along with their corresponding natural language inputs. Additionally it, contains information about CSE query fields. Building on the success of RAG in enhancing LLMs for domain-specific and complex NLP tasks (Lewis et al., 2020), we utilize the knowledge base as an external resource. Relevant CSE query statements and question pairs are retrieved and integrated into the prompts. Related fields are also retrieved to refine the prompt. Modified from COSTAR (Teo, 2024) prompt template, we redesign a prompt template that introduces a supplementary data section (for further details, see Appendix A).

3.2 Few-shot Retrieval

Few-shot retrieval aims to identify relevant few-shot for the natural language input. For an input query Q, the retrieval process proceeds as follows: 1. All natural language questions are encoded by the encoding model and their representations are stored in the vector database. 2. The input query Q is also encoded by the encoding model. 3. During the retrieval process, the cosine similarity algorithm is applied.

3.3 Field Retrieval

The objective of field retrieval is to identify the fields associated with the natural language input. In the CSEQL syntax manual, query fields typically comprise their values and corresponding explanations. The value of the fields are individual and isolated, containing weak semantic information. Each query field has a corresponding explanation that contains some semantic information. To effectively

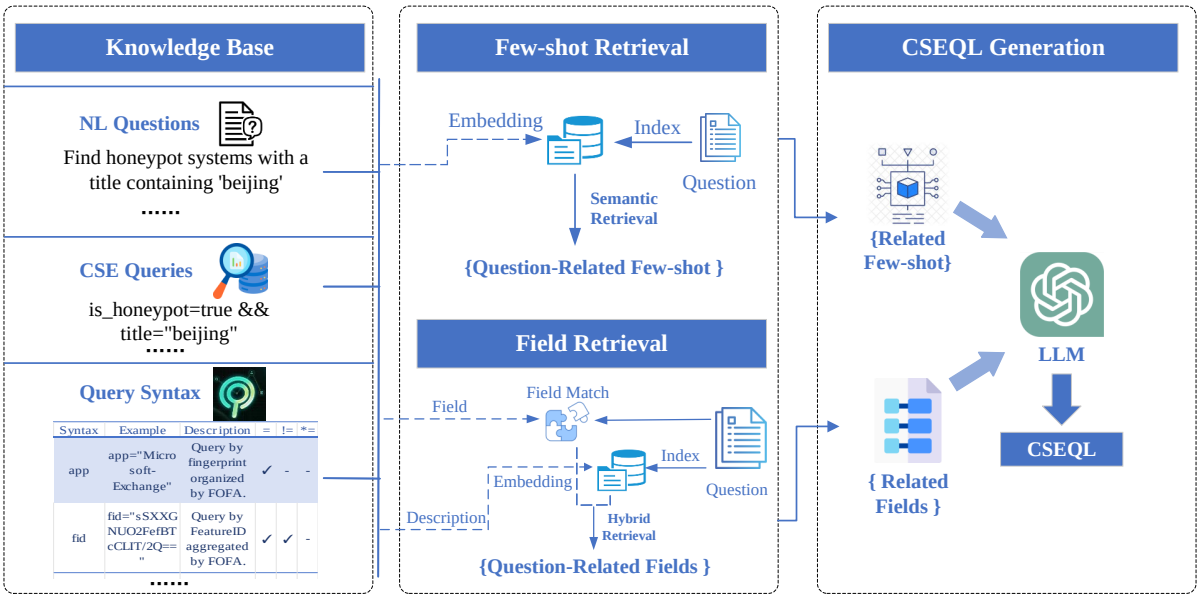

Figure 3: An overview of our proposed method. It consists of the knowledge base construction stage, the retrieval stage, and the generation stage. The knowledge base construction stage involves the creation of three types of knowledge: natural language (NL) questions, CSE queries, and query syntax. The retrieval stage involves few-shot retrieval and field retrieval. The generation stage integrates the results from retrieval stage to prompt the LLM to generate CSEQL.

retrieve fields related to the input, we propose a method that combines keyword retrieval with semantic retrieval. If a field appears in the input, it is considered relevant to the query. When the number of fields through keyword retrieval number does not meet the set threshold, semantic retrieval supplements keyword retrieval. The number of fields to be retrieved during field retrieval is set to K. When keyword retrieval returns N fields (N < K), semantic retrieval calculates the semantic similarity between the input and field explanation, selecting the top M=K-N fields as the result.

3.4 Dataset

This dataset is not only part of the knowledge base construction, but also serves as test data to assess the effectiveness of our method. The collected CSE query statements are real-world data. A portion of the CSE query data is gathered from the official website. The remaining portion is collected from "awesome-search-quries" repository by Project-Discovery (Pro). Additionally, we collect syntax and field information from the official websites of Shodan, ZoomEye, Censys, and FOFA. After gathering the data, We clean and reorganize it.

Annotation. We employ a powerful LLM as annotator. The annotation model we use is GPT-4o. Its task is to generate natural language descriptions for CSE query statements. We provide the LLM with field descriptions for each query, ensuring that its output is faithful to the query's purpose. To ensure high-quality annotation, the data annotated by the model are verified by experts.

CSE	Field Counts	NL-CSEQL Pairs
Shodan	81	1792
ZoomEye	23	100
Censys	2405	175
FOFA	65	4068

Table 2: Detailed information regarding the dataset. It includes an overview of the field counts and the number of NL-CSEQL pairs.

Dataset Statistics. We analyze the number of fields and queries for four CSEs. The detailed information is shown in Table 2. We analyze FOFA data in detail including the number of fields, the average number of logical operators of per query statement, the average number of fields per query statement, the average length of natural language questions, and the average length of query statements. The statistical data are presented in Table 3. The data analysis shows that frequently used query fields follow a power-law distribution, with the top ten most frequently used fields accounting for 84.15% of all query statements.

Quantity	Value
# of question number	4068
# of field number	65
average logical operator/query	1.39
average field/query	1.22
question length/query	89.17
query length(char)/query	32.47

Table 3: Summary statistics of the FOFA dataset. For the sample statistics, average values are reported.

4 Experiment

To evaluate our method, we have summarized the following three research questions:

- **RQ1:** How does our method perform compare to existing work?
- **RQ2:** How does our method perform across different CSEs?
- **RQ3:** How do the components of our method impact performance?

4.1 Experimental Setup

Baselines. We compare our method with existing tools developed by CSEs. CensysGPT Beta (Censys) is a tool developed by Censys to simplify query construction. Another tool is ZoomeyeGPT (Knownsec) introduced by ZoomEye. Due to the lack of tools and methods for generating query statements for FOFA and Shodan, the method of using simple prompt is selected as the baseline.

Large Language Models. We employ several leading LLMs in the experiment. These include GPT-3.5 Turbo (Brown et al., 2020b) and GPT-4o mini (GPT) of OpenAI, Gemini 1.5 Pro (Team et al., 2024) of Google, Claude 3.5 Sonnet (Anthropic, 2024) of Anthropic and Kimi (AI., 2023) of Moonshot.

Implementation of RAG. For few-shot retrieval, we employ semantic retrieval as the primary method. The encoding model, bge-large-en-v1.5 (Xiao et al., 2024), sourced from Hugging Face, has been fine-tuned to enhance retrieval performance for large-scale generation tasks. We use FAISS (Douze et al., 2024) as the vector database, with the number of retrieved few-shot is set to five. In field retrieval, we adopt a hybrid approach that combines keyword-based and dense retrieval methods. The implementation of dense retrieval is identical to that of few-shot retrieval. The number of fields is set to four.

Evaluation Metrics. For evaluation, we follow the prior study (Zhong et al., 2020; Deng et al., 2022; Staniek et al., 2024) to use **E**xact **M**atch (EM) score. Specifically, the EM metric measures whether the generated CSE query statement exactly matches the ground truth. It is note that EM evaluates the performance of our method through strict comparison. Furthermore, to assess whether field errors in the generated CSE query statements have been resolved, a domain-specific evaluation metric, **F**ield **M**atch (FM), is introduced. This metric evaluates the accuracy of field generation in the CSE query statement by measuring the match between the fields of the generated CSE query statement and ground truth. The formula for calculating FM score is as follows:

$$\mathrm{FM} = \frac{\Sigma_{n=1}^{N} \mathrm{score}(\hat{F}_n, F_n)}{N}, \tag{2}$$

$$\mathrm{score}(\hat{F}_n, F_n) = \begin{cases} 1, & \text{if } \hat{F}_n = F_n \\ 0, & \text{if } \hat{F}_n \neq F_n, \end{cases} \tag{3}$$

where N denotes the total number of samples. $\hat{F} = \{\hat{f}_n, n \in (1, m)\}$ and $F = \{f_n, n \in (1, m)\}$ represent the set of fields from the generated CSE query statement and the ground truth.

4.2 Experimental Results

4.2.1 Main Results

We compare our method with existing tools, including CensysGPT Beta (Censys) and ZoomeyeGPT (Knownsec) in terms of the EM and FM score. Table 4 and 5 show the experimental results.

Since the model of CensysGPT Beta cannot be changed and its method is closed-source, we only obtain a set of experimental data. As displayed in Table 4, our method achieves a 9.1% higher EM score and a 24.4% higher FM score on average compared to CensysGPT Beta. These results demonstrate that our method is effective in the test case. Compared to ZoomeyeGPT, our method exhibits an improvement of 26.2% EM score and 31.4% FM score averagely on the dataset of ZoomEye. These results underscore the superior performance of our method in improving the quality of generated query statements.

An error analysis on the generated queries of ZoomEye is conducted and detailed cases are

model	CeneysGPT Beta		ours	
	EM	FM	EM	FM
Kimi	\	\	0.491	0.777
GPT-4o mini	\	\	0.543	0.743
GPT-3.5 Turbo	0.434	0.526	0.469	0.749
Claude 3.5 Sonnet	\	\	0.606	0.851
Gemini 1.5 Pro	\	\	0.514	0.731

Table 4: Comparison of the EM and FM score between our method and CensysGPT Beta.

model	ZoomeyeGPT		ours	
	EM	FM	EM	FM
Kimi	0.34	0.55	**0.59**	**0.82**
GPT-4o mini	0.35	0.53	**0.61**	**0.83**
GPT-3.5 Turbo	0.32	0.49	**0.66**	**0.89**
Claude 3.5 Sonnet	0.36	0.52	**0.61**	**0.86**
Gemini 1.5 Pro	0.44	0.60	**0.65**	**0.86**

Table 5: EM and FM score of our method and ZoomeyeGPT across different LLMs.

shown in Table 6. Our method outperforms ZoomeyeGPT, possibly due to the redundant field information provided by ZoomeyeGPT. This suggests that excessive fields should be excluded from the prompt. Moreover, the absence of some official fields in the prompt, such as "app," contributes to the poor performance of ZoomeyeGPT.

Case 1	
Input	Find all assets associated with '北京大学'.
Ground truth / Our method	org:"北京大学"
ZoomeyeGPT	org:"北京大学" organization:"北京大学"
Case 2	
Input	Search for all assets that utilize the '用友GRP-U8' application.
Ground truth / Our method	app:"用友GRP-U8"
ZoomeyeGPT	title:"用友GRP-U8" ssl:"用友GRP-U8" hostname:"用友GRP-U8" org:"用友GRP-U8" isp:"用友GRP-U8"

Table 6: Case study on ZoomEye data. The red content highlights the error compared to the green ground truth.

4.2.2 Analysis

Performance on Different CSEs. To evaluate the performance of our method on different CSEs, we compared it with a simple prompt method on four CESs. Table 7 shows that our method significantly outperforms simple prompt on different CSEs, particularly on Censys and Shodan. This performance indicates the following: (1) Our method not only reduces hallucination of LLMs but also can accommodate various CSE syntaxes. (2) In addition, different models achieve a low EM score on Censys under simple prompt with an average value of 0.024. Nested fields are the primary cause of LLMs' poor performance. This result highlights a key principle of field setting: prioritize simple fields over nested ones whenever possible. (3) Different models exhibit varying abilities in generating query statements across different CSEs. Under the same method, no single LLM is universally optimal for the Text-to-CSEQL task, necessitating the use of different models for different CSEs.

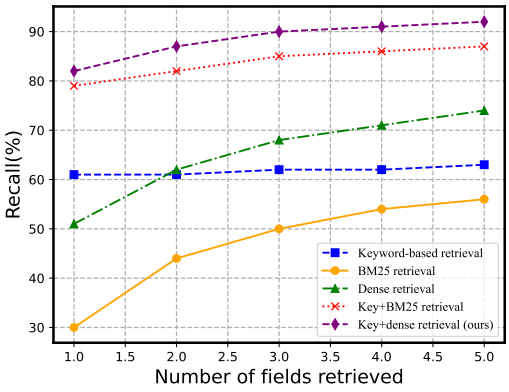

Figure 4: Comparison between different field retrieval methods and retrieval numbers based on the FOFA dataset.

model	Shodan		ZoomEye		Censys		FOFA	
	simple	ours	simple	ours	simple	ours	simple	ours
Kimi	0.104	0.821	0.320	0.590	0.023	0.491	0.424	0.672
GPT-4o mini	0.101	**0.845**	0.310	0.610	0.006	0.543	0.429	0.687
GPT-3.5 Turbo	0.276	0.825	0.390	**0.660**	0.011	0.469	0.372	0.600
Claude 3.5 Sonnet	0.322	0.842	0.510	0.610	0.057	**0.606**	0.548	0.688
Gemini 1.5 Pro	0.188	0.706	0.460	0.650	0.023	0.514	0.545	**0.690**

Table 7: Experimental results of EM score on different CSEs. Simple denotes simple prompt method.

Ablation Study. To evaluate the impact of each component in our proposed method, we conduct ablation studies on the FOFA portion of the dataset. Table 8 shows the results of the ablation study. We remove field retrieval preventing LLMs from accessing field information. As shown in Table 8, the average EM and FM score drop 0.036 and 0.005 without field retrieval respectively. These results suggest that incorporating field retrieval enhances our method's effectiveness. Furthermore, removing

Method		Kimi		GPT-4o mini		GPT-3.5 Turbo		Claude 3.5 Sonnet		Gemini 1.5 Pro	
		EM	FM	EM	FM	EM	FM	EM	FM	EM	FM
Our method	full method	**0.672**	**0.94**	**0.687**	**0.957**	**0.767**	**0.939**	**0.688**	**0.959**	**0.69**	**0.959**
	r.m. field retrieval	**0.672**	0.938	0.681	0.948	0.605	0.931	0.684	0.956	0.683	0.954
	r.m. few-shot retrieval	0.620	0.879	0.624	0.874	0.481	0.846	0.648	0.921	0.640	0.913
Other methods	v.s. all_fields	0.407	0.856	0.369	0.887	0.465	0.870	0.511	0.902	0.514	0.902
	v.s. advanced	0.412	0.599	0.411	0.560	0.301	0.472	0.481	0.723	0.483	0.730

Table 8: Ablation experiment results on FOFA. EM and FM score are shown in the table under different method and model conditions. The "all_fields" method provides all field information in the prompt. The "advanced" method provides two related examples in the prompt. r.m. stands for "remove" and vs. stands for "versus".

few-shot setting	Kimi		GPT-4o mini		GPT-3.5 Turbo		Claude 3.5 Sonnet		Gemini 1.5 Pro	
	EM	FM	EM	FM	EM	FM	EM	FM	EM	FM
few-shot@1	0.623	0.882	0.633	0.892	0.533	0.879	**0.656**	**0.922**	**0.656**	**0.922**
few-shot@2	0.651	0.914	0.660	0.926	0.598	0.909	0.667	0.944	**0.668**	**0.945**
few-shot@3	0.665	0.930	0.668	0.934	0.608	0.916	**0.676**	**0.950**	**0.676**	**0.950**
few-shot@4	0.668	0.935	0.674	0.940	0.613	0.930	0.676	**0.953**	**0.677**	0.952
few-shot@5	0.672	0.938	0.681	0.948	0.605	0.931	**0.684**	**0.956**	0.683	0.954

Table 9: Experimental results of few-shot retrieval on FOFA. The table shows the EM and FM score under different few-shot numbers and LLMs. The best performances of LLMs under each few-shot number is in bold. The few-shot@1/2/3/4/5 indicates the use of the first 1,2,3,4, or 5 similar examples to enhance the generation capability of LLM.

few-shot retrieval decreases the average EM and FM scores by 0.098 and 0.064, respectively. These findings highlight the necessity of the few-shot retrieval module.

Field Retrieval. As shown in Figure 4, our proposed field retrieval method achieves the highest recall rate among five methods. The comparison results demonstrate the effectiveness of integrating keyword matching with dense retrieval. This superior performance can be attributed to the complementary strengths of both methods. Keyword matching ensure high accuracy but lacks flexibility in handling diverse query expressions. In contrast, semantic similarity retrieval is more flexible, accommodating a broader range of query formulations, but it may not be as accurate as keyword matching. Notably, BM25 (Robertson and Zaragoza, 2009) retrieval exhibits poor performance. This is because the description of field is relatively short, and BM25 is prone to errors in queries and documents that are relatively short. Regarding retrieval performance, we compared the recall rates for different numbers of retrieved fields. As shown in the Figure 4, the recall rate of dense retrieval is lower than that of keyword-based retrieval when the number of retrieved fields is fewer than two. This result suggests that keyword-based retrieval is more effective when retrieving a small number of fields.

Few-shot Retrieval. Few-shot retrieval leverages the contextual learning ability of LLMs. Research by (Brown et al., 2020a) has confirmed the existence of contextual learning capabilities in LLMs. We evaluate the performance of our method under various few-shot settings. As described in Table 9, it can be observed that as the number of few-shot increases, both the EM and FM score of the generated queries grow. This result suggests that increasing the number of few-shot can enhance the performance of our method. The increase in both EM and FM score can be attributed to the inclusion of both field and syntax information in the provided few-shot. We can also observe that the growth rate of the EM and FM score is decreasing, indicating that as the number of few-shot examples increases, the impact of few-shot learning diminishes.

5 Related Work

Natural Language Search for Cyberspace Assets. With the aim of improving the usability

of CSEs, previous work has explored the use of LLMs as a translator to convert natural language inputs into query statements. Censys has developed CensysGPT Beta (Censys) using an API provided by OpenAI. Users can utilize it to generate query statements from natural language for Censys. Unfortunately, the method used by CensysGPT Beta is closed-source and CensysGPT Beta is limited to generating queries for the Censys. ZoomEye addresses this by developing a plugin tool called ZoomeyeGPT (Knownsec). It employs prompt engineering to enable LLMs to convert natural language into ZoomEye query statements.

Text-to-SQL. The Text-to-SQL task is closely related to the Text-to-CSEQL task we propose. It maps natural language questions on the given relational database into SQL queries. The development of Text-to-SQL has evolved from early rule-based models to advanced LLMs. Research in Text-to-SQL is categorized into three types: 1) rule-based methods; 2) deep learning-based methods; 3) LLM-based methods. Early systems (Zelle and Mooney, 1996; Saha et al., 2016) used handcrafted grammar rules and heuristic methods to convert natural language queries into SQL statements but faced limitations when handling complex queries and varied database schemas. With the development of deep learning, neural network-based models emerged, capable of learning patterns directly from the training data instead of relying solely on predefined rules. The Text-to-SQL task is commonly treated as a sequence-to-sequence problem (Guo et al., 2019; Choi et al., 2021; Wang et al., 2020; Cao et al., 2021). Some methods explicitly encode the database schema, while (Scholak et al., 2021) shows that fine-tuning a pretrained T5 model (Raffel et al., 2020) could significantly improve the performance of Text-to-SQL. In recent years, LLMs have emerged as a new paradigm for the Text-to-SQL task. Different from deep learning, LLM-based Text-to-SQL methods primarily focus on prompt LLMs to generate correct SQL queries. This approach, known as prompt engineering, includes question representations (Chang and Fosler-Lussier, 2023; Pourreza and Rafiei, 2023), example selection (Nan et al., 2023), and example organization (Gao et al., 2024).

Text-to-DSL. A domain-specific language (DSL) is a programming language designed for a particular application domain. Due to the excellent natural language processing capabilities of LLMs, Text-to-DSL has attracted growing interest. (Wang et al., 2023) proposes a method called "Grammar Prompting" to leverage the performance of LLMs in DSL generation tasks, particularly in generating highly structured language strings from a small number of examples. In the domain of geographic information, Text-to-OverpassQL (Staniek et al., 2024) develops a natural language interface that enabled users to query complex geographic data from the OpenStreetMap database in natural language. In the healthcare field, (Ziletti and DAmbrosi, 2024) introduces a method that integrates Text-to-SQL generation with RAG to answer epidemiological questions. In the field of linguistic research, the first Text-to-CQL task is introduced by (Lu et al., 2024) and aims to automatically convert natural language into corpus query language (CQL).

6 Conclusion

In this work, we propose a new task called Text-to-CSEQL. Text-to-CSEQL aims to generate query statements for CSEs from natural language. We then propose a retrieval-augmented CSEQL generation method based on LLM. Our method leverages few-shot retrieval and field retrieval to alleviate hallucinations of LLMs. Due to the lack of publicly available datasets for evaluating our method, we have construct a new dataset. Furthermore, we propose a novel domain-specific evaluation metric to comprehensively assess the proposed approach. Extensive experiments demonstrate that our method improves the overall EM and FM score compared to existing tools. Additionally, our method can adapt to the syntax variations of different platforms.. Looking ahead, we hope that our work can lay the groundwork for further research in Text-to-CSEQL and promote the usability of CSEs.

Limitations

We have conducted our experiments with some leading LLMs such as GPT-3.5 Turbo, Claude 3.5 Sonnet and others. Given our budget limitations, we will defer exploring our method alongside other advanced LLMs to future research efforts. Moreover, our work on understanding the intent of natural language inputs and optimizing the output CSEQL is still in the early stage. Further in-depth research can be conducted in these two areas to enhance the performance of LLMs in generating CSEQL.

Ethics Statement

We provide a detailed description about ethics considerations. (1) Our research is centered on methodological studies and experiments. The methods and findings in our paper will not be used to harm or deceive any individuals or groups. (2) All data is sourced exclusively from publicly accessible channels, with no collection or processing of user privacy. (3) Financial sponsorships and institutional affiliations are transparently disclosed and are free from conflicts of interest.

References

Censys Search. https://search.censys.io/.

FOFA Search Engine. https://fofa.info.

GPT-4o mini: Advancing cost-efficient intelligence.

Projectdiscovery/awesome-search-queries.

Shodan. https://www.shodan.io.

ZoomEye. https://www.zoomeye.org/.

Moonshot AI. 2023. Kimi. https://kimi.moonshot.cn/.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020a. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020b. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ruisheng Cao, Lu Chen, Zhi Chen, Yanbin Zhao, Su Zhu, and Kai Yu. 2021. LGESQL: Line graph enhanced text-to-SQL model with mixed local and non-local relations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2541–2555, Online. Association for Computational Linguistics.

Censys. CensysGPT Beta.

Shuaichen Chang and Eric Fosler-Lussier. 2023. How to prompt LLMs for text-to-SQL: A study in zero-shot, single-domain, and cross-domain settings. In *NeurIPS 2023 Second Table Representation Learning Workshop*.

DongHyun Choi, Myeong Cheol Shin, EungGyun Kim, and Dong Ryeol Shin. 2021. RYANSQL: Recursively applying sketch-based slot fillings for complex text-to-SQL in cross-domain databases. *Computational Linguistics*, 47(2):309–332.

Naihao Deng, Yulong Chen, and Yue Zhang. 2022. Recent advances in text-to-SQL: A survey of what we have and what we expect. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2166–2187, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2024. Text-to-sql empowered by large language models: A benchmark evaluation. *Proc. VLDB Endow.*, 17(5):1132–1145.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4524–4535, Florence, Italy. Association for Computational Linguistics.

Knownsec. Knownsec/zoomeyegpt.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Ruiguang Li, Meng Shen, Hao Yu, Chao Li, Pengyu Duan, and Lihuang Zhu. 2020. A survey on cyberspace search engines. In *Cyber Security*, pages 206–214, Singapore. Springer Singapore.

Luming Lu, Jiyuan An, Yujie Wang, Liner yang, Cunliang Kong, Zhenghao Liu, Shuo Wang, Haozhe Lin, Mingwei Fang, Yaping Huang, and Erhong Yang. 2024. From text to cql: Bridging natural language and corpus search engine. *Preprint*, arXiv:2402.13740.

Wenxin Mao, Ruiqi Wang, Jiyu Guo, Jichuan Zeng, Cuiyun Gao, Peiyi Han, and Chuanyi Liu. 2024. Enhancing Text-to-SQL Parsing through Question Rewriting and Execution-Guided Refinement. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2009–2024, Bangkok, Thailand. Association for Computational Linguistics.

Linyong Nan, Yilun Zhao, Weijin Zou, Narutatsu Ri, Jaesung Tae, Ellen Zhang, Arman Cohan, and Dragomir Radev. 2023. Enhancing Text-to-SQL Capabilities of Large Language Models: A Study on Prompt Design Strategies. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14935–14956, Singapore. Association for Computational Linguistics.

Mohammadreza Pourreza and Davood Rafiei. 2023. DIN-SQL: Decomposed in-context learning of text-to-SQL with self-correction. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Diptikalyan Saha, Avrilia Floratou, Karthik Sankaranarayanan, Umar Farooq Minhas, Ashish R Mittal, and Fatma Özcan. 2016. Athena: an ontology-driven system for natural language querying over relational data stores. *Proceedings of the VLDB Endowment*, 9(12):1209–1220.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9895–9901, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Staniek, Raphael Schumann, Maike Züfle, and Stefan Riezler. 2024. Text-to-OverpassQL: A Natural Language Interface for Complex Geodata Querying of OpenStreetMap. *Transactions of the Association for Computational Linguistics*, 12:562–575.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Sheila Teo. 2024. How I Won Singapore's GPT-4 Prompt Engineering Competition.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Bailin Wang, Zi Wang, Xuezhi Wang, Yuan Cao, Rif A. Saurous, and Yoon Kim. 2023. Grammar prompting for domain-specific language generation with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Niklas Wretblad, Fredrik Riseby, Rahul Biswas, Amin Ahmadi, and Oskar Holmström. 2024. Understanding the Effects of Noise in Text-to-SQL: An Examination of the BIRD-Bench Benchmark. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 356–369, Bangkok, Thailand. Association for Computational Linguistics.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 641–649, New York, NY, USA. Association for Computing Machinery.

Yuanzhen Xie, Xinzhou Jin, Tao Xie, Matrixmxlin Matrixmxlin, Liang Chen, Chenyun Yu, Cheng Lei, Chengxiang Zhuo, Bo Hu, and Zang Li. 2024. Decomposition for Enhancing Attention: Improving LLM-based Text-to-SQL through Workflow Paradigm. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10796–10816, Bangkok, Thailand. Association for Computational Linguistics.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055.

Ruiqi Zhong, Tao Yu, and Dan Klein. 2020. Semantic evaluation for text-to-SQL with distilled test suites. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 396–411, Online. Association for Computational Linguistics.

Angelo Ziletti and Leonardo DAmbrosi. 2024. Retrieval augmented text-to-SQL generation for epidemiological question answering using electronic health records. In *Proceedings of the 6th Clinical Natural Language Processing Workshop*, pages 47–53, Mexico City, Mexico. Association for Computational Linguistics.

A Modified COSTAR Prompt Template

Context
You want to convert a natural language task into a FOFA query. A natural language task might be something like: "Search for all assets that have an IP address of 1.1.1.1."
FOFA queries are used to search cyberspace assets in the FOFA platform.
For instance, the above task would convert into the FOFA query:
ip="1.1.1.1"
{few_shot}
Objective
The GPT should convert natural language tasks into FOFA queries,
following FOFA's syntax and logic operator rules.
The task includes:
1. Strictly generating FOFA queries without additional explanations or context.
2. Adhering to FOFA's syntax, logic operators, and fields, as outlined in the provided tables.
3. Handling varied natural language inputs, such as:
- "Search for all assets that have an IP address of 1.1.1.1.",
- "Find domains using port 80.",
- "List assets associated with example.com.".
Style
The GPT should adopt a technical and precise style, focusing strictly on generating accurate FOFA queries.
It should behave like a FOFA expert, with expertise in cyberspace asset search
and a thorough understanding of FOFA's syntax, operators, and field rules.
Tone
The GPT should use a straightforward and neutral tone, focusing on delivering the FOFA query output
without unnecessary elaboration or additional context.
Audience
The GPT's outputs should be accessible to everyone, including technical users familiar with FOFA syntax
and non-technical users who may need clear and precise FOFA queries without additional jargon.
Supplementary Data
Below are FOFA query fields and their descriptions to help the GPT generate accurate queries:
{fields}
Response
The GPT should return its output in JSON format with two keys:
- text: A natural language summary of the query, echoing the input task in a simplified form.
- query: The corresponding FOFA query string.
Example:
{{
"text": "Search for all assets with the IP address 1.1.1.1.",
"query": "ip=\"1.1.1.1\""
}}

Table 10: The content of modified COSTAR prompt for CSEQL generation