2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) | 979-8-3503-8622-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/BIBM62325.2024.10822294

Automatic SQL Query Generation from Code Switched Natural Language Questions on Electronic Medical Records

Haodi Zhang*, Jinyin Nie*, Zeming Liu*, Dong Lei†, Yuanfeng Song‡
*College of Computer and Software Engineering, Shenzhen University, Shenzhen, China
†Microsoft, Beijing, China
‡AI Group, WeBank Co., Ltd, Shenzhen, China

***Abstract*—Electronic Medical Records (EMRs) meticulously document patient information in relational databases, presenting a challenging task for medical professionals to effectively retrieve this data. Natural Language Question to SQL query (NL2SQL), a critical task in natural language processing (NLP), shows promising performance in addressing this challenge. However, existing medical NL2SQL studies often focus on generating SQL queries from monolingual questions in English. None of them have studied medical NL2SQL for the Chinese domain.**

In Chinese EMRs, the questions are primarily in Chinese, while many medical terms, such as drugs and diseases, are commonly described in English. This phenomenon is generally known as code-switching (CS) in the field. Given that underlying systems are typically monolingual, CS has proven to pose significant accuracy challenges, as observed in many tasks such as Automatic Speech Recognition (ASR) and Machine Translation (MT). However, the potential effects on EMRs NL2SQL have not been explored. In this study, we investigate the CS-NL2SQL problem, focusing on CS in the context of Chinese questions for the NL2SQL task on EMRs. To assess the model's performance on this task, we construct the first CS-NL2SQL dataset named CS-MIMICSQL in medical domains. We then explore different CS-NL2SQL architectures along two dimensions: cascaded (translation followed by SQL generation) vs. end-to-end. Our results demonstrate that the proposed end-to-end structure can outperform much better than the cascaded baselines.

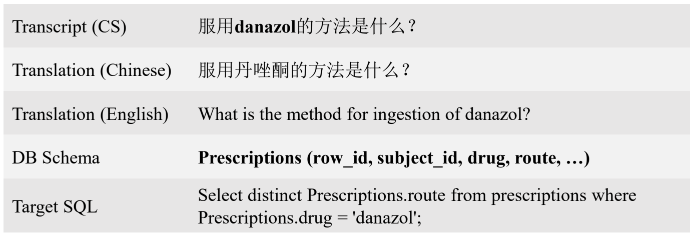
***Index Terms*—Electronic Medical Records Analysis, NL2SQL, Code Switching**

I. Introduction

Electronic Medical Records (EMRs) data [1] serves as a valuable repository, providing a comprehensive longitudinal overview of patients in the field of healthcare informatics. This repository includes essential information such as medical history and allergies, along with clinical documents covering diagnoses, electronic prescriptions, test results, treatment plans, and surgical considerations. The overarching objective of EMRs is to shift from traditional paper-based medical records to electronic recording and storage, introducing a more convenient, efficient, and accurate method of managing patient medical data within a relational database. Structured Query Language (SQL) plays a pivotal role in extracting specific data from EMRs. The widespread integration of EMRs has ushered in an era of digitized and intelligent medical services, ultimately enhancing the efficiency and quality of patient care.

However, the implementation of EMRs has encountered its share of challenges. Retrieving EMRs data from databases presents a challenge for medical professionals, often hindered by a lack of specialized training in executing SQL queries on relational databases [2]. Many hospital staff members are used to accessing the EMRs database through a predefined rule conversion system. To access information beyond these predefined rules, individuals must undergo specialized training to modify and extend the system. This presents a unique set of challenges when navigating the extensive capabilities of EMRs beyond standard protocols [3], [4]. The lack of manpower, expertise, and education has significantly limited the ability to translate EMRs data into actionable insights for improving patient outcomes. The shortage of information technology personnel with the necessary knowledge and training in electronic medical records is a significant obstacle to the widespread use of EMRs. Even for experienced informaticians, dealing with a variety of SQL queries across databases with different structures and application scenarios can be challenging, especially when complex SQL syntax is involved. Information technology personnel often find themselves restricted to managing daily operations and addressing high-priority issues. Their limited understanding of semantic-level knowledge regarding EMRs and their data means that they rarely participate in projects that effectively utilize EMRs data [2]. This further complicates the challenge of capturing and effectively utilizing data to improve healthcare. There is an urgent need for technological solutions that can bridge the gap between healthcare personnel and the effective utilization of EMRs data [5].

An effective solution to overcome these challenges is the development of a system that automatically translates user queries into corresponding SQL queries, commonly known as NL2SQL or Text-to-SQL systems. The NL2SQL generation method solves this problem by first predicting the SQL query for a given question about the database and then executing the query on the database. This innovative approach allows users, including medical professionals, to effortlessly type or verbally express their questions without needing to manually handle the complexities of SQL query execution. To advance NL2SQL tasks in the EMRs domain, several public datasets have been introduced. Notably, the work presented at



Transcript (CS)	服用**danazol**的方法是什么？
Translation (Chinese)	服用丹唑酮的方法是什么？
Translation (English)	What is the method for ingestion of danazol?
DB Schema	**Prescriptions (row_id, subject_id, drug, route, …)**
Target SQL	Select distinct Prescriptions.route from prescriptions where Prescriptions.drug = 'danazol';

Fig. 1. An example of the CS-NL2SQL task on EMRs. The question is mainly using Chinese (the matrix language), and contains English (the embedded language) medical terms.

WWW'20 introduced the MIMICSQL dataset, a substantial dataset of Question-SQL pair [4]. Additionally, datasets such as EHRSQL [3] and emrKBQA [6] are available to support research in this domain.

Despite recent progress, research on NL2SQL in the field of EMRs has predominantly focused on generating SQL queries from single-language problems, like plain English. It is important to acknowledge the prevalence of CS, particularly in the Chinese EMRs NL2SQL scenario. The challenge arises because the issue is primarily presented in Chinese (matrix language), while many medical terms, the database schema, and content are primarily in English (embedded language). It is worth mentioning that matrix languages are the languages that play the dominant role in CS, and are usually the more proficient and commonly used languages by speakers. These languages typically exhibit a higher level of proficiency and widespread usage within the field. In the context of CS, matrix languages provide the exhibit syntactic structure and framework. Embedded languages, on the other hand, refer to languages that are integrated into a matrix language to convey a more precise expression of a specific concept or emotion within sentence construction.

Figure 1 provides a concrete example of this CS phenomenon. The NL2SQL task aims to generate the corresponding target *SQL* based on the transcript and *DB Schema*. In the field of Chinese medicine, it is common for individuals to primarily use the English names when referring to medications. This results in sentences featuring both Chinese and English, creating a CS situation. In this case, the name of the drug in question is "danazol", which is expressed in English. This linguistic variation adds an additional layer of complexity that requires attention and exploration in the development of NL2SQL models for the EMRs domain. While CS has been shown to compromise accuracy and pose challenges in various NLP tasks like Automatic Speech Recognition (ASR) [7] and Machine Translation (MT) [8], its potential impact on EMRs NL2SQL has not been explored before.

In this work, we focus on the *CS-NL2SQL* task in the EMRs field within the context of Chinese/English questions. To evaluate the model performance on this task, we initially develop and release an EMRs CS-NL2SQL corpus named ***CS-MIMICSQL*** derived from MIMICSQL [4]. CS-MIMICSQL is built using GPT-4 [9]. It involves modifying the English EMRs NL2SQL dataset and then generating CS questions for each instance. Then, we systematically explore two CS-NL2SQL architectures: cascaded (translation then SQL generation) versus end-to-end. More specifically, in the cascaded architecture, we attempt to (i) translate the text from English to Chinese, and then attempt Chinese NL2SQL conversion. (ii) translate from Chinese to English, and then proceed with the standard NL2SQL conversion. For the end-to-end approach, we try to (iii) modify advanced monolingual NL2SQL models to enable their multilingual capabilities by utilizing multilingual pre-trained language models (i.e., multilingual BERT [10]). Extensive experiments on our proposed dataset validate that the end-to-end architecture performs much better than the baselines. To sum up, the main contributions of this work are threefold:

- To the best of our knowledge, we are the first to study the effects of CS in the popular EMRs NL2SQL domain. We propose the first CS-NL2SQL dataset in EMRs named CS-MIMICSQL. We hope that this work will fill the gap in CS research for EMRs NL2SQL and, at the same time, inspire more research to focus on CS scenarios in other fields.
- We systematically explore two approaches, namely Cascaded and End-to-end, to tackle this task. The former first converts the problem to NL2SQL in a single language through translation, while the latter's entire network model has only one optimization goal. The End-to-end approach can adapt any existing neural network structure using the neural NL2SQL model enhance its ability to address CS problems.
- We conduct extensive experiments on the proposed CS-MIMICSQL datasets, which validate the feasibility of this task. The end-to-end method is significantly superior to the existing advanced model and the cascaded method in processing CS-NL2SQL tasks. For example, we improve the overall accuracy by 32.4% over the TREQS model, which demonstrates the necessity of enhancing the NL2SQL model's capability to process CS.

The subsequent sections of this paper are structured as follows: in Section II, we begin by introducing related work. Moving on to Section III, we delve into the details of foundational concepts and present our newly proposed dataset, CS-MIMICSQL. Section IV is dedicated to introducing the NL2SQL model, followed by the presentation of two viable CS-NL2SQL methods. We further provide specific experiments and performance analysis within Section V. The concluding remarks for this work can be found in Section VI.

II. Related work

Our study is closely related to two fields: NL2SQL in EMRs and CS in NLP. In this section, we will briefly summarize recent progress in these two related areas.

A. NL2SQL on EMRs

Currently, data query interfaces can be broadly categorized into two types: form-based interfaces, which are userfriendly but possess limited query functionality, and low-level tools that enable users to query using database query languages like SQL. These tools are primarily accessible to a select group of individuals, such as SQL experts. In order to facilitate universal access, utilization, and value extraction, it is imperative to eliminate technical barriers and reduce dependence on IT specialists. Ongoing research trends emphasize the development of Natural Language Interfaces for Databases (NLIDB), enabling users to formulate queries in natural language and then translating them into the underlying database query language. Notably, NL2SQL systems focus on converting natural language queries into SQL queries. While the parse-based approach employed by the database community in the past involves parsing the input question to comprehend its syntactic structure and mapping it to the desired SQL query structure [11], [12], this method, while viable, necessitates substantial human engineering and user interaction to design SQL templates for various scenarios or domains.

In recent times, there has been a surge of interest in neural network generation models [13]–[15]. A prevalent approach adopts a sequence-to-sequence (Seq2Seq) [15] model, featuring an encoder-decoder structure that autonomously learns mapping functions from input natural language (NL) problems to structured query SQL outputs. Essentially, this method transforms an NLSQL challenge into a language translation problem by training the neural network with a substantial dataset of NL query and SQL pairs. The fundamental concept revolves around constructing an encoder capable of understanding the NL problem and the associated database schema, and then predicting the target SQL using a syntax-based decoder. Seq2Seq-based models have emerged as the predominant technique for NL2SQL parsing, primarily due to their ability to be trained in an end-to-end fashion, reducing the need for extensive domain knowledge. Numerous neural network generation models have been introduced to date, with a primary focus on enhancing the performance of both decoders and encoders.

Concerning encoders, certain general-purpose neural networks have gained widespread usage for global reasoning in natural language queries and database schemas. For example, IRNet [13] uses Bi-LSTM [16] and self-attention mechanisms [17] to encode the problem and table schema. TypeSQL [18] is structured similarly to SQLNet [19], utilizing Bi-LSTM and column attention. However, except that TypeSQL incorporates type information from questions to enhance the model's comprehension of uncommon entities and numerical values in natural language. With the advancement of deep neural networks and pre-trained language models, there has been a significant increase in the number of advanced neural-based models published each year in top NLP and speech venues to address the NL2SQL problem. SQLOVA [20] first proposed the method of utilizing pre-trained language models like Bert [21] to leverage table contextualization. They found that using Bert with a Seq2Seq decoder led to poor performance, highlighting the significance of meticulous design when utilizing such a substantial pre-trained model.

With recent advances in deep learning and natural language processing (NLP), the datasets used to train NL2SQL systems are constantly evolving. Currently, NL2SQL has been used and developed in various applications, attracting significant attention. These applications include cross-domain context-free WikiSQL [19], ATIS [22] for air ticket booking, Yelp, IMDB, and the Internet Movie Database [23] and GeoQuery [24] for US geography. However, while cross-domain-based NL2SQL methods can be applied to the EMRs domain, NL2SQL in healthcare is still under-researched. The MIMICSQL dataset, published at WWW '2020, is the first EMRs NL2SQL dataset for EMRs to cater to healthcare databases containing multiple relational tables. The emrKBQA is another dataset in MIMIC-III [25] that is derived from emrQA [26], a clinical reading comprehension dataset. Sequence models are used for semantic parsing to map questions to logical forms that are independent of the query language. EHRSQL [3] is based on real voting results on questions and addresses various practical inquiries that often arise concerning structured EMRs data. However, these studies often focus on generating SQL queries from single-language problems and do not take into account the CS phenomenon that is very common in the field of EMRs. The CS-MIMICSQL dataset we created using GPT-4 takes this situation into consideration.

B. Code-Switching in NLP

The phenomenon of CS is pervasive in the field of Natural Language Processing (NLP), where multiple languages intermingle. While this tendency is commonly observed in multilingual communities, the ubiquity of the Internet has facilitated its occurrence in numerous formal settings. Given the prevalence of CS in human languages, several tasks within the domain of CS have been explored.

In CS language modeling [27], the focus lies on addressing language modeling tasks in code-switched text. The goal is to predict both the next word and the language of the subsequent word. Within Machine Translation (MT), CSP [28] introduces a novel neural machine translation pre-training method called Code-Switching pre-training. CSP adopts an encoder-decoder framework, with the encoder handling mixed sentences as input and the decoder predicting replacement fragments of the input sentences. In the realm of speech, CS scenarios are more commonplace, especially in spoken expressions where multilingual language use is frequent. Concentrating on speech translation (ST) tasks in English/Spanish conversations, this paper [29] creates a new ST corpus from existing public datasets. It explores various ST frameworks along two dimensions: pipeline approaches (transcribe then translate) versus end-to-end methods (jointly transcribe and translate), and unidirectional (source $\longrightarrow$ target) versus bidirectional (source $\longleftrightarrow$ target). In speech recognition tasks, additional speech recognition models are often employed as auxiliary modules to enhance recognition ability through language information [30], [31]. However, this approach increases computational costs. In recent years, there has been a growing focus on end-

TABLE I
THE STATISTICS OF THE PROPOSED CS-MIMICSQL DATASET

	# Q.	#DB	#Table/DB	Avg. Q. Len.	Avg. SQL Len.
Train	8000	1	5	26.92	20.91
Dev	1000	1	5	27.48	21.24
Test	1000	1	5	23.17	16.93
All	10000	1	5	26.60	20.55

to-end frameworks [7], [32], which combine acoustic models, language models, and other components into a single model for joint training. Nevertheless, the absence of code-switching training data limits the performance of these methods.

III. CS-NL2SQL TASK AND DATASET

In this section, we will begin by presenting key foundational concepts and task descriptions that will enhance the comprehension of the subsequent work. Subsequently, we will unveil our newly proposed dataset, CS-MIMICSQL.

A. Preliminary and Task Description

Code-switching natural language questions (CS-NLQ). CS-NLQ is a common phenomenon in the Chinese medical domain, typically occurring when an ordinary user poses a database-related query. In Chinese EMRs, the questions are mainly in Chinese, and a large number of medical terms, such as drugs and diseases, are commonly expressed in English. At the same time, data stored in English or represented by acronyms are often found in the database, which poses challenges for processing NL2SQL tasks.

SQL query. An SQL query is a series of specific keywords used by developers to manipulate data stored in relational databases. These commands are commonly categorized into Data Definition Language (DDL), Data Query Language (DQL), Data Manipulation Language (DML), and Data Control Language (DCL). DDL and DML commands, which involve modifying the database, are typically accessible only to database administrators with the relevant permissions. DQL, used for retrieving data stored in relational databases, is the most commonly used SQL type in our dataset.

Database schema. In this context, a database schema refers to schema objects within the database, including information such as table names, column names, data types, primary keys, and other database details. The table name is used to identify the basic unit of data storage, while the column name represents the attributes or fields in the table. The data types and constraint rules of the columns define the structure and specifications of the data. The design of database schema is crucial to ensure the consistency, integrity, and maintainability of data. It also establishes the fundamental framework for database operations.

CS-NL2SQL on EMRs. This paper aims to translate healthcare-related code-switching questions asked by doctors into database queries and then retrieve the answers from medical records. Given a CS-NLQ $\mathbf{x_{cs}}$ which has lexical level language

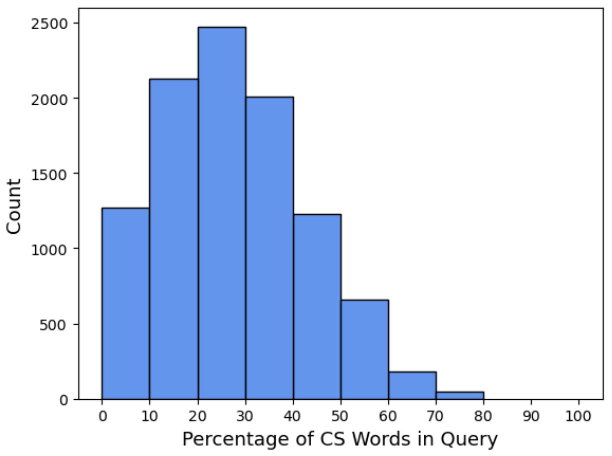
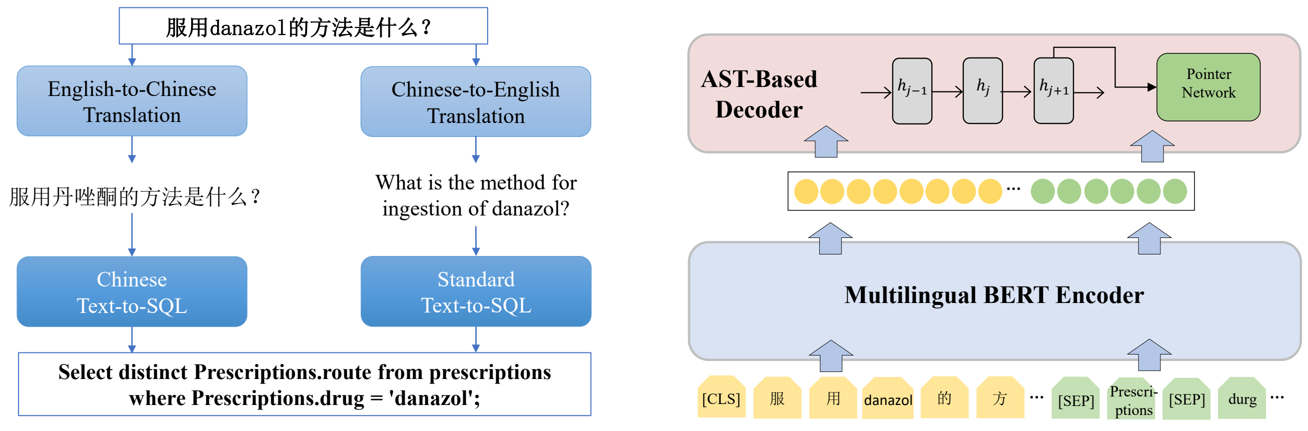


Fig. 2. Histogram illustrating the proportions of CS words in a query for the CS-MIMICSQL dataset. For example, 30 means that 30% of the words in the query are English.

sequence $\mathbf{x_L} = (\mathbf{x_1}, \cdots, \mathbf{x_n})$, and the schema $\mathbf{S}$ of the corresponding database, the *CS-NL2SQL* task aims to learn a model $f(\mathbf{S}, \mathbf{x}_{cs}) \rightarrow \mathbf{sql}$, which could automatically generate the SQL query sql. The schema $\mathbf{S}$ includes N tables' collection T_x, where $T_x = \{t_i\}_{i=1}^{N}$, and the set $C_{x,i}$ of L_i columns where $C_{x,i} = \{c_{i,j}\}_{j=1}^{L_i}$ for each table $t_i \in T_x$, where L_i is the Count of columns in t_i for $i \in [1, N]$. We use C_x to encompass all the columns in the subsequent discussion and $C_x = \{C_{x,i}\}_{i=1}^{N}$. The whole corpus $\mathcal{D}$ contains M instances, denoted as $\mathcal{D} = \{\mathbf{d}^1, \cdots, \mathbf{d}^M\}$, and each instance is in the format of $\mathbf{d} = (\mathbf{x}_{cs}, \mathbf{S}, \mathbf{sql})$.

B. Code-Switched NL2SQL Datasets

Given that CS-NL2SQL on EMRs represents a novel task, there is currently a lack of publicly available datasets suitable for evaluating its performance. The prevalent assumption is that, since CS is primarily observed in spoken language, the most practical approach to data generation involves labeling CS speech. However, this method often requires a significant amount of skilled labor and hours of tedious manual transcription work. An alternative strategy is to generate CS data from existing monolingual text. Unfortunately, predicting code-switching points in a sentence lacks a perfect rule, as individuals tend to code-switch differently. In recent years, efforts have been made to synthesize more CS text using models trained on existing data. Garg et al. [33] employ a generative model to create CS sentences from scratch. In contrast, Chang et al. [34] utilize Generative Adversarial Networks (GAN) [35] coupled with Reinforcement Learning (RL) [36] to automatically generate CS data from monolingual sentences. These days, GPT-4 stands out as one of the most advanced large language models, demonstrating effective understanding of human intent and efficient task execution. Following the practice of automatically generating the CS text/speech from the monolingual text, we utilize GPT-4 [9] to modify the English NL2SQL dataset - MIMICSQL_natural version. To


服用danazol的方法是什么？
English-to-Chinese Translation
Chinese-to-English Translation
服用丹唑酮的方法是什么？
What is the method for ingestion of danazol?
Chinese Text-to-SQL
Standard Text-to-SQL
Select distinct Prescriptions.route from prescriptions where Prescriptions.drug = 'danazol';
AST-Based Decoder
Pointer Network
Multilingual BERT Encoder
[CLS]
服
用
danazol
的
方
[SEP]
Prescri-ptions
[SEP]
durg

(a) Cascaded Approach (b) End-to-end Approach

Fig. 3. Comparison Between the Cascaded and the End-to-End Approach for CS-NL2SQL.

transform initial English questions into CS Chinese/English questions, we create a specific zero-shot prompt for a GPT-4 persona: *"You are an overseas doctor who enjoys conversing in a mix of English and Chinese"*. It is essential for GPT-4 to maintain the correct names of medical terms in English, including drugs, diseases, and medical procedures.

The detailed statistics of the generated CS-MIMICSQL dataset are provided in the Table I. Our dataset is built on a database called MIMICSQL [4], which contains five tables. The dataset is divided into train/dev/test. The average length of each NLQ is 26.6, and the average length of each SQL statement is 20.55. In Figure 2, we observe the proportion of CS words in a sentence for the CS-MIMICSQL. Most sentences contain no more than 50% of CS words. While GPT-4 can understand and execute instructions within a specified role, its nature as a black-box generative model may still result in problems like hallucinations and truncation. After generating CS-NLQs, we perform an automated inspection to resolve issues such as truncation during generation and the inclusion of parentheses after English words and their Chinese translations. This step aims to ensure clarity and alignment with the CS scenario in spoken language.

IV. CS-NL2SQL Methodology

In this section, we will first introduce the general structure and basic input and output of the NL2SQL neural model, and then we will delve into the specific structure of the two proposed methods.

A. The NL2SQL Model

We employ a neural network-based Sequence-to-Sequence (Seq2Seq) [15] framework for the NL2SQL generation task. Our Seq2Seq framework consists of a question encoder and an SQL decoder. The question encoder utilizes a pre-trained deep bidirectional transformer model for encoding, while the SQL decoder employs the Abstract Structure Tree-based Decoder [37]. We leverage the BERT [10] encoder to encode questions and schema tokens, as well as to perform schema linking. The decoder part of the network remains the same as the original IRNet model. When generating the encoder input, we concatenate the questions $\mathbf{x_{cs}}$ and schema $\mathbf{S}$ tokens with a special token *[SEP]*. This special token is also used to concatenate schema tokens, which separate all distinct column names in the database schema. We denote the tokens sequence of a question and its schema by $\mathbf{x} = [\mathbf{x_1}, \mathbf{x_2} \cdots, \mathbf{x_L}, [\mathbf{SEP}], \mathbf{S}]$ where $\mathbf{L}$ is the length of $\mathbf{x_l}$ and $\mathbf{x_i}$ is the i^{th} token of $\mathbf{x_l}$. Taking $\mathbf{x}$ as the input of a problem, each $\mathbf{x_{cs}}$ and its corresponding $\mathbf{S}$ can be transformed into a corresponding embedding vector $\mathbf{e_x}$ by BERT encoder:

$$e_x = BERT\_Encoder\,(x) \tag{1}$$

Similar to the grammar model constructed in [13], the probability of generating an AST $\mathbf{y}$ can be expressed as:

$$p\,(y \mid e_x) = \prod_{t=1}^{T} p\,(a_t \mid e_x, a_{<t}) \tag{2}$$

Where $\mathbf{a_t}$ is the action taken at time step $\mathbf{t}$, and $\mathbf{a_{<t}}$ is the sequence of actions taken before time $\mathbf{t}$. $\mathbf{T}$ is the total number of time steps for the whole action sequence.

B. Two CS-NL2SQL Methods

We could roughly divide the potential methodologies for tackling CS-NL2SQL into two categories: the cascaded approach and the end-to-end approach.

1) The Cascaded Method: Our cascaded pipeline is shown in Fig. 3(a). Inspired by common practices in other CS tasks such as sentiment analysis [38], the approach first adopts a MT component to translate the CS-NLQ into a monolingual question. It then employs neural-based NL2SQL models, such as Seq2SQL [15], IRNet [13], and TypeSQL [18]. In terms of implementation, we are trying two different approaches: (i) English-to-Chinese translation followed by Chinese NL2SQL Conversion; (ii) Chinese-to-English translation followed by English NL2SQL conversion. In our CS scenario, the matrix language is Chinese, while the embedded language is English. Opting for the first approach, English-to-Chinese

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT CS-NL2SQL METHODS

Method		Validation Set		Testing Set	
		Sketch Acc.	Overall Acc.	Sketch Acc.	Overall Acc.
Advanced	TREQS	0.757	0.611	0.746	0.602
Cascaded	Chinese-English MT + English NL2SQL	0.946	0.737	0.942	0.704
	English-Chinese MT + Chinese NL2SQL	0.943	0.845	0.920	0.826
End-to-end	Chinese-BERT NL2SQL	0.943	0.845	0.919	0.837
	English-BERT NL2SQL	0.943	0.734	0.946	0.724
	mBERT NL2SQL	**0.979**	**0.935**	**0.973**	**0.913**

translation proves to be more straightforward, facilitating the acquisition of accurate Chinese-based questions. However, the subsequent Chinese NL2SQL conversion presents challenges, as potential models must navigate schema linking in a cross-lingual fashion. This involves answering questions in Chinese while the database schema remains in English. Conversely, in the second approach, the initial Chinese-to-English translation step poses greater difficulty because the majority of words in a CS question are in Chinese. Nonetheless, the subsequent NL2SQL step becomes more manageable because both the question and the schema are expressed in the English language. This trade-off between translation challenges and compatibility with schema language highlights the intricacies involved in selecting an optimal approach for Code-Switching NL2SQL tasks.

2) The End-to-End Method: In the pursuit of an end-to-end approach, we enhance the adaptability of neural-based NL2SQL models to accommodate CS questions. Illustrated in Fig 3(b), our modification involves augmenting the neural network structure of an existing NL2SQL model with a multilingual version of a pre-training language model, such as mBERT [10] in our implementation. mBERT, a pre-trained language model, proves instrumental in meeting our requirements, as it seamlessly supports both Chinese and English words. The model structure of mBERT mirrors that of the original BERT, featuring a 12-layer transformer. Notably, mBERT is pre-trained on multilingual Wikipedia corpora encompassing 104 languages, utilizing a shared word vocabulary rather than monolingual corpora. This unique characteristic enhances its applicability in handling CS scenarios, as researchers have empirically demonstrated mBERT's proficiency in such supervised scenarios. The versatility introduced by mBERT's multilingual pre-training makes it a fitting choice for seamlessly integrating code-switching capabilities into NL2SQL models, thereby contributing to enhanced performance and adaptability in diverse linguistic contexts.

V. EXPERIMENTS

In this section, the performance of our proposed framework is evaluated in detail based on quantitative indicators. We will first introduce the evaluation measures and provide some additional details. Using the previously proposed dataset CS-MIMICSQL, we split the dataset into training, development and test sets using a 80-10-10 split. Then, we will demonstrate the validity of the proposed framework by comparing the results obtained from different methods.

A. Dataset

Following the segmentation of the MIMICSQL datasets, our CS-MIMICSQL dataset is similarly partitioned into training set, validation set, and test set, comprising 8000, 1000, and 1000 samples, respectively.

B. Evaluation Metrics

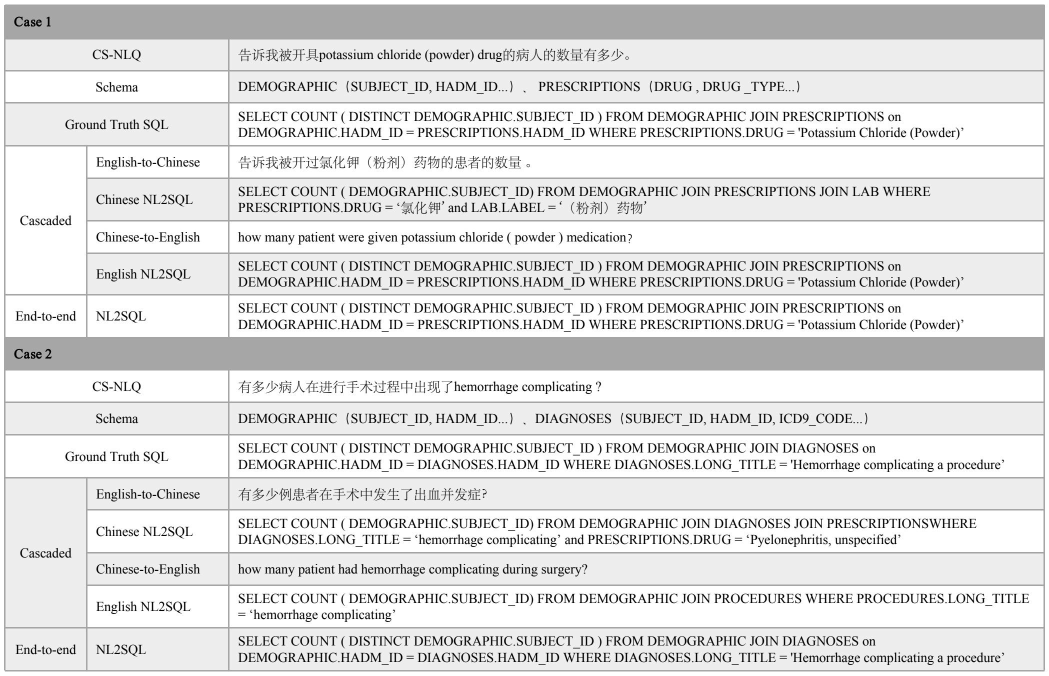
In our experiments, we utilize two commonly adopted metrics for SQL generation to validate the effectiveness of the proposed models.

- **Sketch Accuracy**: This metric evaluates the percentage of the matches between the skeleton of generated SQL and the skeleton of the ground truth. Also, an abstract syntax tree obtained from an SQL query is used to represent the basic structure of an SQL statement to calculate the accuracy of the SQL skeleton.
- **Overall Accuracy**: This metric assesses the proportion of matches between the generated SQL and the reference SQL. To reduce the impact of condition order, we split each SQL statement into different segments: "select column", "aggregator", and "condition". Rather than comparing the SQL strings directly, we handle the condition part as a set and compare each element within that set.

C. Implementation Details

To ensure a fair comparison, we employ the advanced IRNet [13] as the foundational NL2SQL model for both the cascaded and end-to-end approaches. For different methods, we utilize three different pre-trained language models from Hugging Face[1] as encoders: *bert-base-uncased* as the English encoder, *bert-base-chinese* as the Chinese encoder, and *bert-base-multilingual-cased* as the multilingual encoder. For the cascaded approach, we primarily use the translation service provided by Google Translate[2]. It is noteworthy that we also assess the accuracy of the advanced model, TREQS [4], in generating SQL queries for CS questions on CS-MIMICSQL.

[1]https://huggingface.co/
[2]https://pypi.org/project/translators/



Case 1		
CS-NLQ		告诉我被开具potassium chloride (powder) drug的病人的数量有多少。
Schema		DEMOGRAPHIC（SUBJECT_ID, HADM_ID...）、 PRESCRIPTIONS（DRUG , DRUG _TYPE...）
Ground Truth SQL		SELECT COUNT (DISTINCT DEMOGRAPHIC.SUBJECT_ID) FROM DEMOGRAPHIC JOIN PRESCRIPTIONS on DEMOGRAPHIC.HADM_ID = PRESCRIPTIONS.HADM_ID WHERE PRESCRIPTIONS.DRUG = 'Potassium Chloride (Powder)’
Cascaded	English-to-Chinese	告诉我被开过氯化钾（粉剂）药物的患者的数量 。
	Chinese NL2SQL	SELECT COUNT (DEMOGRAPHIC.SUBJECT_ID) FROM DEMOGRAPHIC JOIN PRESCRIPTIONS JOIN LAB WHERE PRESCRIPTIONS.DRUG = ‘氯化钾’ and LAB.LABEL = ‘（粉剂）药物’
	Chinese-to-English	how many patient were given potassium chloride (powder) medication？
	English NL2SQL	SELECT COUNT (DISTINCT DEMOGRAPHIC.SUBJECT_ID) FROM DEMOGRAPHIC JOIN PRESCRIPTIONS on DEMOGRAPHIC.HADM_ID = PRESCRIPTIONS.HADM_ID WHERE PRESCRIPTIONS.DRUG = 'Potassium Chloride (Powder)’
End-to-end	NL2SQL	SELECT COUNT (DISTINCT DEMOGRAPHIC.SUBJECT_ID) FROM DEMOGRAPHIC JOIN PRESCRIPTIONS on DEMOGRAPHIC.HADM_ID = PRESCRIPTIONS.HADM_ID WHERE PRESCRIPTIONS.DRUG = 'Potassium Chloride (Powder)’
Case 2		
CS-NLQ		有多少病人在进行手术过程中出现了hemorrhage complicating ?
Schema		DEMOGRAPHIC（SUBJECT_ID, HADM_ID...）、DIAGNOSES（SUBJECT_ID, HADM_ID, ICD9_CODE...）
Ground Truth SQL		SELECT COUNT (DISTINCT DEMOGRAPHIC.SUBJECT_ID) FROM DEMOGRAPHIC JOIN DIAGNOSES on DEMOGRAPHIC.HADM_ID = DIAGNOSES.HADM_ID WHERE DIAGNOSES.LONG_TITLE = 'Hemorrhage complicating a procedure’
Cascaded	English-to-Chinese	有多少例患者在手术中发生了出血并发症?
	Chinese NL2SQL	SELECT COUNT (DEMOGRAPHIC.SUBJECT_ID) FROM DEMOGRAPHIC JOIN DIAGNOSES JOIN PRESCRIPTIONSWHERE DIAGNOSES.LONG_TITLE = ‘hemorrhage complicating’ and PRESCRIPTIONS.DRUG = ‘Pyelonephritis, unspecified’
	Chinese-to-English	how many patient had hemorrhage complicating during surgery?
	English NL2SQL	SELECT COUNT (DEMOGRAPHIC.SUBJECT_ID) FROM DEMOGRAPHIC JOIN PROCEDURES WHERE PROCEDURES.LONG_TITLE = ‘hemorrhage complicating’
End-to-end	NL2SQL	SELECT COUNT (DISTINCT DEMOGRAPHIC.SUBJECT_ID) FROM DEMOGRAPHIC JOIN DIAGNOSES on DEMOGRAPHIC.HADM_ID = DIAGNOSES.HADM_ID WHERE DIAGNOSES.LONG_TITLE = 'Hemorrhage complicating a procedure’

Fig. 4. Two Cases Given by Difference CS-NL2SQL Methods.

D. Experimental Results

In this section, we will examine the outcomes of each method on the suggested datasets.

1) Performance Analysis: The results of the cascaded approach and the end-to-end approach are presented in Table II, from which we could make several observations. (i) Generally speaking, the end-to-end method outperforms both the advanced and the cascaded approaches. The reason is obvious: the cascaded approach usually suffers from a well-known problem named *error propagation*, which means that a small error made in the former step results in a much larger error in the following step. However, the end-to-end approach could alleviate this problem because the entire network benefits from a single optimization objective. (ii) In the cascade approach within the EMRs CS scenario, where the matrix language is Chinese and the embedded language is English, translating all words into Chinese (e.g., English-Chinese MT + Chinese NL2SQL) works better than translating them into English. Since a larger proportion of the questions are in Chinese. (iii) We also found that all models achieve sketch accuracy above 0.9, but the overall accuracy varies widely, with the mBERT NL2SQL model achieving up to 11% more. In the medical field, there are numerous proper nouns and abbreviations for drugs and procedures that are prone to errors during translation. Obviously, mBERT assists the decoder in accurately populating the corresponding columns, tables, and other elements in the skeleton during parsing.

2) Case Study: As demonstrated in Fig. 4, we have presented two cases that illustrate the SQL query generation process of the cascaded and the end-to-end methods. The schema in Fig. 4 provides information about the database, including the table names and column names. The first case serves as a positive example, where both methods accurately convert the CS question to the corresponding SQL statement. However, the cascaded approach requires more time and resources due to the need for multi-step translation. In contrast, the second case is more complex, with only the *End-to-End* method and *Chinese-English MT + English NL2SQL* method producing the correct answer. The *English-Chinese MT + Chinese NL2SQL* method, on the other hand, generates an incorrect table that is not present in the database. This indicates the lack of robustness of cascaded methods and highlights the impact of the intermediate translation process on the final SQL statement generation.

VI. Conclusion

In this work, we focus on the common code-switching phenomena in the context of Chinese NL2SQL in EMRs. To evaluate the model's performance on this task, we first create a CS-MIMICSQL corpus derived from existing datasets. Then, we systematically explore various CS-NL2SQL architectures across two dimensions: cascaded (translation followed by SQL generation) versus end-to-end. We demonstrate that our proposed end-to-end model outperforms all baseline models. We hope our work will shed some light on Code-Switching research in EMRs NL2SQL.

Acknowledgments. This work is supported by the Guangdong Basic and Applied Basic Research Foundation (2022A1515010675). Yuanfeng Song is the corresponding author.

References

[1] Faustine Williams and Suzanne Austin Boren. The role of the electronic medical record (emr) in care delivery development in developing countries: a systematic review. *Informatics in primary care*, 16(2), 2008.

[2] Miriam Moerbe and Arpad Kelemen. Turning electronic health record data into meaningful information using sql and nursing informatics. *CIN: Computers, Informatics, Nursing*, 32(8):366–377, 2014.

[3] Gyubok Lee, Hyeonji Hwang, Seongsu Bae, Yeonsu Kwon, Woncheol Shin, Seongjun Yang, Minjoon Seo, Jong-Yeup Kim, and Edward Choi. Ehrsql: A practical text-to-sql benchmark for electronic health records. *Advances in Neural Information Processing Systems*, 35:15589–15601, 2022.

[4] Ping Wang, Tian Shi, and Chandan K Reddy. Text-to-sql generation for question answering on electronic medical records. In *Proceedings of The Web Conference 2020*, pages 350–361, New York, NY, USA, 2020. ACM.

[5] Steven Szydlowski and Christina Smith. Perspectives from nurse leaders and chief information officers on health information technology implementation. *Hospital Topics*, 87(1):3–9, 2009.

[6] Preethi Raghavan, Jennifer J Liang, Diwakar Mahajan, Rachita Chandra, and Peter Szolovits. emrKBQA: A clinical knowledge-base question answering dataset. In Dina Demner-Fushman, Kevin Bretonnel Cohen, Sophia Ananiadou, and Junichi Tsujii, editors, *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 64–73, Online, June 2021. ACL.

[7] Ke Li, Jinyu Li, Guoli Ye, Rui Zhao, and Yifan Gong. Towards code-switching asr for end-to-end ctc models. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6076–6080. IEEE, 2019.

[8] Orion Weller, Matthias Sperber, Telmo Pires, Hendra Setiawan, Christian Gollan, Dominic Telaar, and Matthias Paulik. End-to-end speech translation for code switched speech. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1435–1448, 2022.

[9] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[10] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*, 2019.

[11] John M. Zelle and Raymond J. Mooney. Learning to parse database queries using inductive logic programming. In *AAAI/IAAI, Vol. 2*, 1996.

[12] ZENG Zhong, Lee Mong Li, and Ling Tok Wang. Answering keyword queries involving aggregates and group-bys in relational databases. 2015.

[13] Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. Towards complex text-to-SQL in cross-domain database with intermediate representation. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of ACL*, pages 4524–4535, Florence, Italy, July 2019. Association for Computational Linguistics.

[14] Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. In *Proceedings of the 58th ACL*, pages 7567–7578, 2020.

[15] Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*, 2017.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[18] Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir Radev. TypeSQL: Knowledge-based type-aware neural text-to-SQL generation. In Marilyn Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 NAACL*, pages 588–594, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[19] Xiaojun Xu, Chang Liu, and Dawn Song. Sqlnet: Generating structured queries from natural language without reinforcement learning. *arXiv preprint arXiv:1711.04436*, 2017.

[20] Wonseok Hwang, Jinyeong Yim, Seunghyun Park, and Minjoon Seo. A comprehensive exploration on wikisql with table-aware word contextualization. *arXiv preprint arXiv:1902.01069*, 2019.

[21] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[22] Robin Jia and Percy Liang. Data recombination for neural semantic parsing. In *Proceedings of the 2016 EMNLP*, pages 12–22, Berlin, Germany, August 2016. ACL.

[23] Navid Yaghmazadeh, Yuepeng Wang, Isil Dillig, and Thomas Dillig. Sqlizer: query synthesis from natural language. *Proc. ACM Program. Lang.*, 1(OOPSLA), oct 2017.

[24] Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. Learning a neural semantic parser from user feedback. *arXiv preprint arXiv:1704.08760*, 2017.

[25] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

[26] Anusri Pampari, Preethi Raghavan, Jennifer Liang, and Jian Peng. emrQA: A large corpus for question answering on electronic medical records. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 EMNLP*, pages 2357–2368, Brussels, Belgium, October-November 2018. ACL.

[27] Khyathi Chandu, Thomas Manzini, Sumeet Singh, and Alan W. Black. Language informed modeling of code-switched text. In Gustavo Aguilar, Fahad AlGhamdi, Victor Soto, Thamar Solorio, Mona Diab, and Julia Hirschberg, editors, *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 92–97, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[28] Zhen Yang, Bojie Hu, Ambyera Han, Shen Huang, and Qi Ju. CSP:code-switching pre-training for neural machine translation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2624–2636, Online, November 2020. Association for Computational Linguistics.

[29] Orion Weller, Matthias Sperber, Telmo Pires, Hendra Setiawan, Christian Gollan, Dominic Telaar, and Matthias Paulik. End-to-end speech translation for code switched speech, 2022.

[30] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[31] Shiyu Zhou, Linhao Dong, Shuang Xu, and Bo Xu. A comparison of modeling units in sequence-to-sequence speech recognition with the transformer on mandarin chinese. In *International Conference on Neural Information Processing*, pages 210–220. Springer, 2018.

[32] Bo Li, Yu Zhang, Tara Sainath, Yonghui Wu, and William Chan. Bytes are all you need: End-to-end multilingual speech recognition and synthesis with bytes. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5621–5625. IEEE, 2019.

[33] Saurabh Garg, Tanmay Parekh, and Preethi Jyothi. Code-switched language models using dual RNNs and same-source pretraining. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *2018 EMNLP*, pages 3078–3083, Brussels, Belgium, October-November 2018. ACL.

[34] Ching-Ting Chang, Shun-Po Chuang, and Hung-Yi Lee. Code-Switching Sentence Generation by Generative Adversarial Networks and its Application to Data Augmentation. In *Proc. Interspeech 2019*, pages 554–558, 2019.

[35] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[36] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[37] Pengcheng Yin and Graham Neubig. A syntactic neural model for general-purpose code generation. *arXiv preprint arXiv:1704.01696*, 2017.

[38] Devansh Gautam, Kshitij Gupta, and Manish Shrivastava. Translate and classify: Improving sequence level classification for english-hindi code-mixed data. In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 15–25, 2021.