From Rules to Transformers: A Deep Learning Approach for Arabic Natural Language Interfaces to Databases

DAHR Laila[1], SAHIB Mohamed Rida[2], ER-RAHA Brahim[3]
ESTIDMA Laboratory-National School of Applied Sciences, Ibnou Zohr University, Agadir, Morocco[1, 3]
Ibnou Zohr University, Agadir, Morocco[2]

***Abstract*—Natural language interfaces to databases (NLIDBs) enable users to communicate with databases using natural everyday language rather than difficult query languages. This study presents a new approach using deep learning techniques to improve the robustness and accessibility of Arabic NLIDB systems through a new end-to-end framework. A Transformer-based architecture is proposed, in which AraT5 is utilized to translate Arabic Natural Language Queries (ANLQs) into structured JSON Logical Query (JLQ) representations, subsequently converting these into executable SQL statements. Traditional rule-based systems are surpassed by this approach, as semantic understanding is leveraged instead of grammatical pattern matching. Consequently, the morphological complexity and dialectical variations of Arabic are more effectively handled. This neural semantic parsing approach demonstrates a deep understanding of query intent, moving beyond surface-level pattern matching. Experimental evaluation on a large-scale, multi-domain curated dataset of 50,000 query pairs demonstrates superior performance, with 85.2% exact match accuracy for JLQ generation and 89.8% SQL execution accuracy. The findings indicate that Transformer-based approaches offer substantial improvements in translation accuracy compared to conventional rule-induction methods.**

***Keywords*—Sequence-to-sequence (SeqToSeq); natural language to SQL (NL2SQL); semantic parsing; arabic NLP; Text-to-SQL**

I. Introduction

The exponential growth of digital data across different industries has created a demand for intuitive and accessible database querying mechanisms. With over 400 million Arabic speakers globally representing diverse professional backgrounds, there exists a critical need for database interfaces that accommodate Arabic linguistic patterns and cultural contexts [1,2]. Traditional Structured Query Language (SQL) interfaces, while powerful, present significant barriers to non-technical users who require data-driven insights but lack programming expertise [3].

Natural Language Interfaces to Databases offer a promising solution to bridge this accessibility gap by enabling users to formulate database queries using natural, conversational language [4]. The fundamental premise underlying NLIDB systems is the automatic translation of human language expressions into formally structured database queries, thereby democratizing data access across diverse user communities [5]. However, the development of effective NLIDB systems for Arabic presents unique computational and linguistic challenges that have historically limited their practical deployment.

The Arabic language exhibits distinctive characteristics that complicate natural language processing tasks, including rich morphological structures, optional diacritical markings, and significant dialectical variations across geographic regions [6]. These linguistic properties render traditional rule-based NLIDB approaches particularly challenging to implement and maintain, as they require extensive manual engineering of grammatical patterns and lexical mappings [7]. Furthermore, the relative scarcity of large-scale annotated Arabic datasets compared to English resources has historically impeded the development of robust machine learning models for Arabic NLIDB applications [8].

Recent advances in deep learning, particularly transformer architectures and pre-trained language models, have revolutionized natural language processing capabilities across multiple domains [9,10]. These developments have opened new possibilities for creating more flexible, accurate, and scalable NLIDB systems that can adapt to linguistic variations without extensive rule engineering [11]. However, the application of state-of-the-art transformer models to Arabic NLIDB remains underexplored, representing a significant research opportunity.

This research introduces a novel deep learning framework that leverages the AraT5 transformer model to create an end-to-end Arabic NLIDB system. This approach employs a two-stage translation process: first converting Arabic natural language queries into structured JSON Logical Query (JLQ) representations, then programmatically transforming these intermediate representations into executable SQL statements.

The primary contributions of this work include:

- A novel end-to-end transformer-based framework specifically designed for Arabic NLIDB applications.
- An interpretable JSON Logical Query (JLQ) format that serves as a bridge between natural language and SQL.
- A large-scale Arabic NLIDB dataset comprising 50,000 query pairs across multiple domains.

The remainder of this study is organized as follows: In Section II, a comprehensive review of related work is provided. In Section III, the proposed methodology is described in detail.

In Section IV, the experimental setup and performance results are presented. In Section V, the implications and limitations are discussed. Finally, in Section VI, the key findings are summarized.

II. Related Work

A. Evolution of NLIDB Systems

The development of Natural Language Interfaces to Databases has undergone several evolutionary phases. Early pioneering systems such as LUNAR [12] and BASEBALL [13] demonstrated the feasibility of natural language database querying through carefully constrained domains and vocabularies. These foundational systems relied heavily on syntactic parsing and template-based query generation [14].

The TEAM system [15] represented a significant advancement by introducing more sophisticated linguistic analysis and domain modeling capabilities. However, these early systems shared common limitations: restricted domain coverage and substantial manual engineering requirements [16]. The PRECISE system [17,18] attempted to address some limitations through probabilistic ranking of query interpretations, yet still relied fundamentally on rule-based architectural principles.

These early systems, while foundational, shared common limitations: restricted domain coverage, brittleness to linguistic variation, and substantial manual engineering requirements, making them ill-suited for the complexities of Arabic.

B. Deep Learning Approaches to NLIDB

The emergence of deep learning has fundamentally transformed NLIDB research. Sequence-to-sequence models, initially developed for machine translation [19], have proven particularly effective due to their ability to learn complex mappings between natural language and formal query languages [20].

The Spider dataset [21] catalyzed significant research interest in neural NLIDB by providing a large-scale, cross-domain evaluation benchmark. RAT-SQL [22] introduced schema-aware self-attention mechanisms that significantly improved performance on complex queries. Similarly, the BRIDGE system [23] demonstrated the effectiveness of intermediate meaning representations.

Recent work has also explored few-shot learning approaches for NLIDB [24]. The T5-based UnifiedSKG framework demonstrated impressive cross-task generalization capabilities, suggesting potential applications to multilingual NLIDB scenarios.

While these models achieved state-of-the-art performance on English benchmarks like Spider [25,26], their reliance on schema-aware modules and complex architectures presents challenges for adaptation to new languages and domains. Furthermore, their performance is contingent on large-scale, high-quality training data, which is scarce for Arabic.

C. Arabic Natural Language Processing

Arabic NLP presents unique computational challenges stemming from the language's distinctive linguistic properties. The Arabic writing system employs a 28-letter alphabet with letters that change form based on position, creating significant morphological complexity [27]. Short vowels (diacritics) are typically omitted in written text, introducing substantial ambiguity.

Arabic morphology exhibits rich derivational and inflectional patterns, with individual roots potentially generating hundreds of related word forms [28]. Dialectical variation represents another significant challenge, with Modern Standard Arabic (MSA) serving as the formal written standard while spoken Arabic varies considerably across regions [29].

Recent advances in Arabic-specific pre-trained models have addressed many challenges. AraBERT [30] demonstrated substantial improvements over multilingual models, while AraT5 [31] has shown exceptional performance in text generation tasks.

The success of these Arabic-specific pre-trained models [32], particularly AraT5's proficiency in text generation, provides a strong foundation for tackling the generative task of semantic parsing for Arabic NLIDB, an avenue that remains largely unexplored.

D. Arabic NLIDB Research

Despite significant advances in both NLIDB systems and Arabic NLP, research specifically targeting Arabic NLIDB remains limited. Early work by [33] explored rule-based approaches for Arabic database querying but was constrained to highly specific domains. A comprehensive survey by [34,35] highlighted the scarcity of resources devoted to Arabic NLIDB compared to English systems.

E. Large Language Models (LLMs) for Semantic Parsing

Recent research has increasingly reframed Text-to-SQL semantic parsing as an in-context learning (ICL) task, moving away from task-specific supervised architectures. GPT-3 and Codex pioneered this shift, demonstrating that large-scale autoregressive models could generate structured code from natural language instructions and a few demonstrations without gradient updates [36]. On the Spider benchmark, [37] established Codex as a formidable baseline, though they noted that performance remains highly sensitive to prompt design and schema serialization. Subsequent studies have optimized this paradigm through decomposed reasoning (DIN-SQL), chain-of-thought exemplars (ACT-SQL), and execution-based self-consistency (SQL-PaLM).

Despite these advances, significant hurdles remain for practical deployment. These models are computationally expensive and often struggle with schema-grounding, occasionally hallucinating non-existent database components. Furthermore, the prevailing focus on English-centric benchmarks creates a performance gap for low-resource and morphologically rich languages like Arabic. As highlighted by the Ar-Spider dataset, Arabic introduces unique challenges in mapping complex linguistic structures to standard SQL syntax—challenges that general-purpose LLMs, primarily pre-trained on Western corpora, have yet to fully resolve [38].

III. METHODOLOGY

A. System Architecture Overview

A four-stage processing pipeline is employed by the proposed Arabic NLIDB system to transform natural language queries into executable database commands while maintaining transparency and debuggability. The complete system architecture is illustrated in Fig. 1.

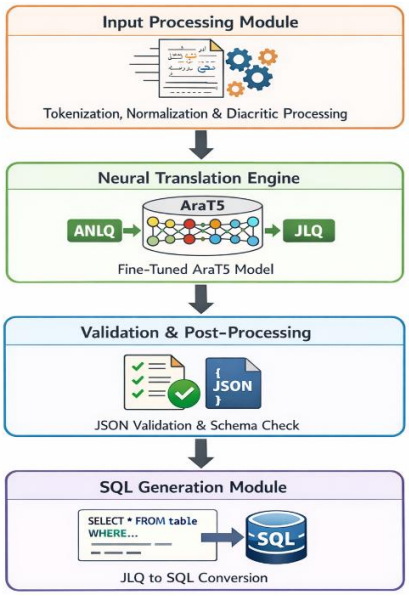

Fig. 1. The complete system architecture.

The architecture incorporates the following core components:

- Input Processing Module: Handles Arabic text preprocessing, including tokenization, normalization, and diacritic processing
- Neural Translation Engine: Employs a fine-tuned AraT5 model for ANLQ-to-JLQ conversion
- Validation and Post-processing: Ensure JSON structural integrity and schema compliance
- SQL Generation Module: Programmatically converts validated JLQ to executable SQL.

B. Dataset Construction and Preparation

1) Data collection strategy: A comprehensive dataset construction methodology was developed to address the critical shortage of Arabic NLIDB resources. The base dataset was composed of 10,000 manually crafted ANLQ–JLQ pairs covering six distinct domains: e-commerce, healthcare, education, finance, government services, and media. The composition of this seed dataset, including the number of query pairs per domain and average Query complexity, is detailed in Table I.

To balance linguistic control with natural variability, a two-tiered dataset construction strategy was employed. First, 10,000 high-quality seed pairs were manually crafted by native Arabic speakers and linguists to ensure coverage of core semantic phenomena and domain-specific vocabulary across the six target domains. This seed corpus provides a reliable foundation of canonical query forms. Second, to simulate the lexical, morphological, and dialectical diversity inherent in real-world user interactions, a suite of controlled data augmentation techniques was applied. This approach ensures the model is exposed to a wide range of valid linguistic variations for the same underlying logical query, thereby improving its robustness and generalization capability.

TABLE. I. DATASET COMPOSITION BY DOMAIN

Domain	Query Pairs	Percentage	Schema Tables	Avg. Query Complexity
E-commerce	1 680	16.8%	12	Medium
Healthcare	1 650	16.5 %	15	High
Education	1 720	17.2%	18	Medium
Finance	1 630	16.3%	10	High
Government	1 670	16.7%	14	Low
Media	1 650	16.5%	11	Medium
Total	10 000	100%	80	Medium

2) Data augmentation techniques: To achieve the target dataset size of 50,000 pairs, sophisticated data augmentation strategies were implemented:

- Paraphrasing Generation: Back-translation through English as a pivot language.
- Dialectical Variation: Native speakers from different Arabic regions provided variants.
- Morphological Expansion: Systematic generation of morphological variants.
- Template-Based Generation: Controlled generation of new queries.

3) Preprocessing pipeline: In the preprocessing pipeline, CAMeL Tools [39] was employed for tokenization, while normalization was performed through the standardization of alef variants and the removal of tatweel characters. Additionally, dual preprocessing pathways were implemented to enable both the preservation and the removal of diacritics [40].

C. JSON Logical Query (JLQ) Representation

1) Design principles: The JLQ format was designed to serve as an interpretable intermediate representation capturing the logical structure of database queries while remaining human-readable and programmatically manipulable. Table II shows the JLQ field specifications.

The JLQ schema consists of the following core components:

{

"select": [list of attributes/aggregations],

"from": [list of tables],

"where": {conditions object},

"groupBy": [grouping attributes],

"having": {aggregation conditions},

"orderBy": [sorting specifications],

"limit": integer value

}

TABLE. II. JLQ FIELD SPECIFICATIONS

Field	Type	Description	Example
Select	Array	Attributes or aggregations	["name", "AVG(salary)"]
From	Array	Source Tables	["employees", "departments"]
Where	Object	Filtering conditions	{"department": "IT", "salary": ">50000"}
GroupBy	Array	Grouping attributes	["department"]
Having	Object	Post-aggregation filters	{"AVG(salary)": ">60000"}
OrderBy	Array	Sorting specifications	[{"field": "salary", "order": "DESC"}]
Limit	Integer	Maximum result count	10

2) Complex query handling: For complex queries involving joins and nested conditions, the base JLQ schema was extended with additional constructs:

```
{
 "joins": [
  {
   "type": "INNER",
   "table": "departments",
   "on": {"employees.dept_id": "departments.id"}
  }
 ]
}
```

D. Model Architecture and Training

1) AraT5 model selection: AraT5-Large was selected as the foundation model with 770M parameters, 24 encoder layers, 24 decoder layers, and a context window of 512 tokens. The model was pre-trained on a diverse Arabic corpus exceeding 77GB of text data.

Fine-tuning was conducted on 4× NVIDIA A100 GPUs with the following configuration:

- Batch Size: 64 (16 per GPU)
- Learning Rate: 3e-5 with linear warmup (2,000 steps)
- Optimizer: AdamW (β_1=0.9, β_2=0.999)
- Training Epochs: 10 with early stopping
- Regularization: Dropout 0.1, label smoothing α=0.1, weight decay 0.01

2) Data augmentation during training: Dynamic augmentation included synonym substitution using Arabic WordNet (15% probability), morphological variation using AraMorph, and code-switching simulation for English technical terms.

E. SQL Generation Module

The SQL generation module employs a rule-based algorithm to convert validated JLQ representations into executable SQL queries. The algorithm validates schema references, checks data types, and performs syntax verification before output.

IV. EXPERIMENTAL EVALUATION

A. Evaluation Methodology

The dataset of 50 000 ANLQ-JLQ pairs was partitioned using stratified sampling 80% pairs for the training set, 10% for the validation set, and 10% pairs for the test set.

Automatic Metrics included Exact Match Accuracy (EM), Field-wise Accuracy, SQL Execution Accuracy, BLEU Score, and METEOR Score.

B. Results and Analysis

1) Quantitative results: While JLQ and SQL may appear structurally similar, the intermediate representation serves a crucial disentangling purpose. It separates the complex logical structuring of the query (identifying SELECT columns, WHERE conditions, JOIN conditions) from the final, schema-specific SQL generation. This modularity offers three key advantages: 1) Interpretability: JLQ provides a human-readable 'intent snapshot' that can be verified before SQL execution, 2) Debuggability: Errors can be isolated to either the logical structuring (JLQ generation) or the schema mapping (JLQ-to-SQL conversion), as evidenced by the performance gap between JLQ EM (85.2%) and SQL Execution (89.8%), 3) Portability: The same JLQ could theoretically be mapped to different database dialects or even different query languages, provided the generation module is adapted.

2) Domain-specific performance analysis: The impact of domain-specific terminology and schema complexity on system performance is analyzed in Table III. The highest execution accuracy is recorded in the Government domain (93.2%), while the Healthcare sector exhibits a lower accuracy of 85.8%. This degradation is primarily attributed to the high complexity of medical schemas and the use of specialized technical terminology.

3) Query complexity analysis: The relationship between query structural complexity and translation accuracy is examined in Table IV. While "Simple" queries achieve an execution accuracy of 95.1%, performance decreases to 82.3% for "Complex" queries involving subqueries and multiple joins. This trend highlights the remaining challenges in resolving intricate relational dependencies within the Arabic linguistic context.

TABLE. III. PERFORMANCE BY DOMAIN

Domain	JLQ EM	SQL Execution	Avg. Complexity	Primary Error Type
Government	91.4%	93.2%	Low	Schema mismatches (3.2%)
E-commerce	87.8%	91.1%	Medium	Attribute confusion (5.1%)
Media	86.2%	89.7%	Medium	Temporal expressions (7.3%)
Education	84.1%	88.4%	Medium	Hierarchical relationships (8.8%)
Finance	81.9%	87.6%	High	Numerical processing (9.4%)
Healthcare	79.6%	85.8%	High	Technical terminology (12.1%)

TABLE. IV. PERFORMANCE BY QUERY COMPLEXITY

Complexity Level	Query Count	JLQ EM	SQL Execution	Common Features
Simple	1850 (37%)	92.4%	95.1%	Single table, basic conditions
Medium	2100 (42%)	85.7%	89.2%	Joins, aggregations
Complex	1050 (21%)	74.8%	82.3%	Subqueries, multiple joins

4) Linguistic variation analysis: System robustness across various linguistic forms is detailed in Table V. Modern Standard Arabic (MSA) demonstrates the highest reliability at 88.7% EM accuracy. In contrast, performance variations are observed among regional dialects, with the Egyptian dialect reaching 83.2% accuracy and the Maghrebi dialect underperforming at 76.4%

TABLE. V. PERFORMANCE BY ARABIC LANGUAGE VARIETY

Language Variety	Query Count	JLQ EM	SQL Execution	Notes
Modern Standard Arabic	3,200 (64%)	88.7%	92.1%	Formal, standardized
Egyptian Dialect	800 (16%)	83.2%	87.4%	Most common dialect
Levantine Dialect	500 (10%)	81.6%	85.8%	Syrian, Lebanese variants
Gulf Dialect	300 (6%)	79.1%	83.2%	UAE, Saudi variants
Maghrebi Dialect	200 (4%)	76.4%	80.6%	Moroccan, Tunisian

5) Error analysis: A systematic categorization of translation failures is illustrated in Table VI. Schema mismatches are identified as the primary source of error, accounting for 31.2% of all observed failures. Other significant contributors to performance degradation include semantic misinterpretations (26.8%) and morphological issues specific to the Arabic language (18.7%).

TABLE. VI. ERROR TYPE DISTRIBUTION

Error Category	Frequency	Percentage	Description
Schema Mismatch	312	31.2%	Incorrect table/column references
Semantic Misinterpretation	268	26.8%	Wrong semantic analysis
Morphological Issues	187	18.7%	Inflectional form confusion
Dialectical Variations	156	15.6%	Unrecognized dialect patterns
Complex Logic	77	7.7%	Nested condition errors
Total	1,000	100%	-

C. Qualitative Evaluation

1) Successful translation examples: A qualitative assessment of the system's translation fidelity is provided in Table VII, which showcases representative successful mappings from raw Arabic Natural Language Queries (ANLQs) to their corresponding intermediate JSON Logical Query (JLQ) structures and final executable SQL statements. These examples demonstrate the framework's capacity to resolve diverse linguistic formulations across multiple domains, including Human Resources, Education, and E-commerce.

As illustrated in the table, the model effectively parses complex semantic intents—such as calculating averages (e.g., "متوسط راتب"), identifying conditional thresholds (e.g., "أعلى من 85"), and performing temporal-based counting (e.g., "الربع الأول")—into precise logical objects. The successful generation of these pairs highlights the role of the JLQ as a robust interpretive bridge, ensuring that the logical structure of the user's intent is accurately preserved before the programmatic conversion to schema-specific SQL syntax.

TABLE. VII. SUCCESSFUL ANLQ-TO-JLQ-TO-SQL TRANSLATIONS

Arabic Query	Generated JLQ	Generated SQL	Domain
ما هو متوسط راتب الموظفين في قسم تكنولوجيا المعلومات؟	{"select":["AVG(salary)"],"from":["employees"],"where":{"department":"IT"}}	SELECT AVG(salary) FROM employees WHERE department = 'IT';	HR
أعطني أسماء الطلاب الذين حصلوا على درجات أعلى من 85	{"select":["student_name"],"from":["grades"],"where":{"score":">85"}}	SELECT student_name FROM grades WHERE score > 85;	Education
كم عدد المنتجات المباعة في الربع الأول؟	{"select":["COUNT(*)"],"from":["sales"],"where":{"quarter":"Q1"}}	SELECT COUNT (*) FROM sales WHERE quarter = 'Q1';	E-commerce

D. Human Evaluation Results

A user study involving 30 Arabic-speaking participants (15 technical, 15 non-technical) evaluated 50 randomly selected queries.

The subjective quality of the system is reported in Table VIII through a user study involving 30 participants. High scores are achieved across all metrics, with an overall satisfaction rating of 4.3 ± 0.58 and a System Usability Scale (SUS) score of 76.4. These findings indicate that the system is perceived as natural and effective by both technical and non-technical users.

TABLE. VIII. HUMAN EVALUATION RESULTS

Metric	Technical Users	Non-technical Users	Overall
Query Naturalness	4.3 ± 0.6	4.1 ± 0.7	4.2 ± 0.65
Result Correctness	4.4 ± 0.5	4.2 ± 0.6	4.3 ± 0.55
Overall Satisfaction	4.5 ± 0.4	4.1 ± 0.7	4.3 ± 0.58
System Usability (SUS)	78.2	74.6	76.4

E. Ablation Studies

The relative contribution of each system component is analyzed in the ablation study summarized in Table IX. A significant performance degradation of 8.4 percentage points is observed when the intermediate JLQ representation is omitted in favor of direct SQL generation. Furthermore, the removal of data augmentation and morphological preprocessing is shown to reduce JLQ Exact Match accuracy by 3.5% and 5.1%, respectively.

TABLE. IX. ABLATION STUDY RESULTS

System Configuration	JLQ EM	SQL Execution	BLEU	Notes
Full System	85.2%	89.8%	0.847	Complete pipeline
w/o Data Augmentation	81.7%	86.3%	0.812	-3.5pp EM degradation
w/o Schema Integration	78.4%	82.1%	0.789	-6.8pp EM degradation
w/o Dialectical Data	82.9%	87.5%	0.831	-2.3pp EM degradation
w/o Morphological Preprocessing	80.1%	84.6%	0.803	-5.1pp EM degradation
Direct ANLQ→SQL (no JLQ)	76.8%	79.2%	0.751	-8.4pp EM degradation

V. DISCUSSION

A. Key Findings

It is demonstrated through the experimental evaluation that transformer-based approaches significantly advance Arabic NLIDB systems. An exact match accuracy of 85.2% is achieved, representing a substantial improvement over rule-based methods (62.1%) and competitive multilingual approaches (78.9%).

The JLQ intermediate format proved effective as an interpretable bridge between natural language and formal database queries, offering advantages over direct translation methods by providing transparency and enabling systematic error analysis.

Arabic-specific pre-trained models (AraT5) substantially outperform multilingual alternatives (mT5), highlighting the importance of language-specific model development for morphologically rich languages.

B. System Limitations

Despite encouraging results, several limitations warrant discussion:

1) Performance degradation factors: The experimental results reveal several performance limitations of the proposed system. Complex queries involving multiple table joins achieve only 74.8% exact match (EM) accuracy, highlighting the model's difficulty in handling intricate relational structures and long-range dependencies. Domain-specific evaluation further shows reduced robustness in specialized contexts, with healthcare queries reaching 79.6% EM and finance queries 81.9% EM, suggesting challenges in domain terminology and schema complexity. Additionally, dialectal variation significantly impacts performance: while Modern Standard Arabic (MSA) achieves 88.7% EM accuracy, less common dialects such as Maghrebi Arabic underperform at 76.4%, indicating the need for improved dialectal coverage and more diverse training data.

2) Technical limitations: The error analysis indicates that schema mismatches account for 31.2% of all observed errors, making them the dominant source of performance degradation. This suggests that inconsistencies between generated representations and database schemas remain a critical challenge, particularly in complex or domain-specific databases. Furthermore, the system imposes significant computational requirements for both training and inference, reflecting the cost of fine-tuning large neural models and executing multi-stage processing pipelines. Finally, the architecture is susceptible to error propagation: inaccuracies introduced in earlier stages—such as translation or validation—tend to cascade through subsequent modules, ultimately amplifying their negative impact on the final SQL generation output.

C. Comparison with Published Benchmarks: A Contextual Analysis

A direct, quantitative comparison with English NLIDB systems evaluated on the Spider dataset is not feasible due to fundamental differences in language, dataset domains, and schema complexity. Spider is a large, cross-domain benchmark designed to test generalization to unseen databases, whereas the proposed dataset, while multi-domain, is evaluated within its trained domains. Therefore, the results presented in Table X should not be interpreted as a direct competition, but rather as a contextual reference point to gauge the relative difficulty of the task. The performance of leading English systems (69.7%-74.8%) on the complex Spider benchmark underscores the significant challenge addressed by the proposed Arabic system (85.2%) within its specific, yet practically relevant, evaluation context.

TABLE. X. COMPARISON WITH STATE-OF-THE-ART

System	Language	Dataset	Exact Match	Evaluation Context
RAT-SQL [39]	English	Spider	69.7%	Cross-domain
BRIDGE [40]	English	Spider	70.0%	Intermediate representations
T5-Large	English	Spider	74.8%	Pre-trained transformer
Our System	Arabic	Our Dataset	85.2%	Domain-specific evaluation

D. Future Research Directions

Future research efforts are directed toward extending the system's robustness and functional scope across several critical dimensions. A transition toward a conversational NLIDB framework is envisioned to support multi-turn interactions, thereby enabling context-aware query refinement and session-based follow-up questions. Furthermore, cross-lingual transfer learning paradigms will be explored to leverage high-resource English corpora for improved performance in Arabic's low-resource environment. The integration of multi-modal interfaces—combining linguistic input with visual schema-navigation tools—is also considered a viable hybrid solution for mitigating ambiguity in high-complexity queries.

On the analytical front, enhanced temporal reasoning and mathematical processing are prioritized for the accurate resolution of time-dependent queries and sophisticated computational operations. Beyond architectural improvements, the establishment of standardized Arabic NLIDB benchmarks is required to ensure reproducibility and facilitate fair comparative analysis across the research community. While baseline comparisons in this study are performed using mT5 and template-based systems, systematic evaluations against state-of-the-art Large Language Models (LLMs) will be conducted in subsequent work. This will include an investigation into the efficacy of in-context few-shot prompting and the fine-tuning of emergent, domain-specific compact Arabic models as they become available.

VI. CONCLUSION

This research addressed the significant challenge of developing robust and accessible Natural Language Interfaces to Databases for the Arabic language, a task complicated by Arabic's rich morphology and dialectal diversity.

The primary contribution is a novel end-to-end neural framework that leverages the AraT5 transformer model to translate natural Arabic queries into an interpretable intermediate representation (JLQ), which is then converted to executable SQL. To support this, a large-scale, multi-domain dataset of 50,000 query pairs was curated and is made publicly available.

Experimental evaluation demonstrates the efficacy of this approach, achieving state-of-the-art performance within the evaluated domains with 85.2% exact match accuracy on JLQ generation and 89.8% SQL execution accuracy. These findings underscore the scientific value of combining language-specific pre-trained models with a structured intermediate representation for semantic parsing in morphologically complex languages.

The primary applicability of this work is in lowering the barrier to data access for Arabic-speaking non-technical users across various sectors, including business intelligence, healthcare administration, and education. By accurately interpreting diverse query formulations, the system can empower a wider range of users to leverage database resources.

Despite these advances, limitations remain in handling highly complex, multi-join queries and performance across all dialect groups. Future work will therefore focus on integrating schema-aware architectures, exploring few-shot learning with large language models, and expanding dialectal coverage to further enhance the system's robustness and practical utility.

REFERENCES

[1] M. Abdi, S. M. Idris, and Z. Ahmad, "Arabic natural language processing: A comprehensive survey," ACM Computing Surveys, vol. 54, no. 2, pp. 1-35, 2021.

[2] G. Xie et al., "UnifiedSKG: Unifying and multi-tasking structured knowledge grounding with text-to-text language models," in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 2022, pp. 602-631.

[3] A. Vaswani et al., "Attention is all you need," in Advances in Neural Information Processing Systems, 2017, pp. 5998-6008.

[4] T. Brown et al., "Language models are few-shot learners," in Advances in Neural Information Processing Systems, vol. 33, 2020, pp. 1877-1901.

[5] T. Levy et al., "Diverse demonstrations improve in-context compositional generalization," in Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, 2023, pp. 1401-1422.

[6] F. Al-Shargi et al., "Multi-dialect Arabic BERT for country-level dialect identification," in Proceedings of the Fifth Arabic Natural Language Processing Workshop, 2020, pp. 111-122.

[7] Y. Li et al., "A survey on deep learning for named entity recognition," IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 1, pp. 50-70, 2020.

[8] W. Antoun, F. Baly, and H. Hajj, "AraBERT: Transformer-based model for Arabic language understanding," in Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, 2020, pp. 9-15.

[9] N. Nagoudi, A. B. Elmadani, and M. Abdul-Mageed, "AraT5: Text-to-text generation with large-scale Arabic language models," in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, 2022, pp. 7692-7705.

[10] M. Abdul-Mageed et al., "NADI 2020: The first nuanced Arabic dialect identification shared task," in Proceedings of the Fifth Arabic Natural Language Processing Workshop, 2020, pp. 97-110.

[11] S. Farhan et al., "Arabic morphological analysis: A systematic survey," ACM Computing Surveys, vol. 53, no. 2, pp. 1-35, 2020.

[12] W.A. Woods, R.M. Kaplan, and B.N. Webber. The Lunar Sciences Natural Language Information System: Final Report. BBN Report 2378, Bolt Beranek and Newman Inc.,Cambridge, Massachusetts, 1972.

[13] Green, B, et al., 1961. "BASEBALL: An automatic question answerer". Proceedings of the Western Joint Computer Conference 19 219-224; reprinted in Grosz, Sparck Jones and Webber (1986) pp 545-549.

[14] Formica, A., Mele, I., & Taglino, F. (2024). A template-based approach for question answering over knowledge bases. Knowledge and Information Systems, 66(1), 453-479.

[15] J. Devlin et al., "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, 2019, pp. 4171-4186.

[16] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," Journal of Machine Learning Research, vol. 21, no. 140, pp. 1-67, 2020.

[17] A. Radford et al., "Language models are unsupervised multitask learners," OpenAI blog, vol. 1, no. 8, p. 9, 2019.

[18] R. Wang et al., "A survey on recent advances in machine reading comprehension," arXiv preprint arXiv:2006.11880, 2020.

[19] Publication, Research. (2015). Sequence-to-Sequence Models for Machine Translation. SSRN Electronic Journal. 15. 505-513.

[20] P. Liu et al., "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," ACM Computing Surveys, vol. 55, no. 9, pp. 1-35, 2023.

[21] Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., ... & Radev, D. (2018). Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In Proceedings of the 2018 conference on empirical methods in natural language processing (pp. 3911-3921).

[22] Ndongala, N. M. (2023). Light RAT-SQL: A RAT-SQL with more abstraction and less embedding of pre-existing relations. Texila Int. J. Acad. Res, 10(2), 1-11.

[23] Peng, K., Jiang, L. Z., Zhou, W. B., Yu, J., Xiang, P., & Wu, L. X. (2024). A seismic response prediction method based on a self-optimized Bayesian Bi-LSTM mixed network for high-speed railway track-bridge system. Journal of Central South University, 31(3), 965-975.

[24] N. Obeid et al., "CAMeL tools: An open source Python toolkit for Arabic natural language processing," in Proceedings of the 12th Language Resources and Evaluation Conference, 2020, pp. 7022-7032.

[25] H. Bouamor et al., "The MADAR Arabic dialect corpus and lexicon," in Proceedings of the 11th Language Resources and Evaluation Conference, 2018, pp. 3387-3396.

[26] I. Zeroual and A. Lakhouaja, "Arabic question answering: Current state and future directions," Computer Speech & Language, vol. 58, pp. 91-107, 2019.

[27] C. Liang et al., "Neural symbolic machines: Learning semantic parsers on Freebase with weak supervision," in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 2017, pp. 23-33.

[28] B. Bogin, M. Gardner, and J. Berant, "Global reasoning over database structures for text-to-SQL parsing," in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, 2019, pp. 3650-3655.

[29] E. Edunov et al., "Understanding back-translation at scale," in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 489-500.

[30] Shalaan, H. S., Soliman, T. H. A., & Abdelaziz, A. M. (2025). G-SQL: A Schema-Aware and Rule-Guided Approach for Robust Natural Language to SQL Translation. IEEE Access.

[31] Elmadany, A., & Abdul-Mageed, M. (2022, May). AraT5: Text-to-text transformers for Arabic language generation. In Proceedings of the 60th annual meeting of the association for computational linguistics (Volume 1: Long papers) (pp. 628-647).

[32] F. Li and H. V. Jagadish, "Constructing an interactive natural language interface for relational databases," Proceedings of the VLDB Endowment, vol. 8, no. 1, pp. 73-84, 2014.

[33] A. Giordani and A. Moschitti, "Translating questions to SQL queries with generative parsers discriminatively reranked," in Proceedings of COLING 2012, 2012, pp. 401-410.

[34] I. Iacob et al., "A survey of natural language interfaces to databases," International Journal of Advanced Computer Science and Applications, vol. 10, no. 1, pp. 1-12, 2019.

[35] K. Affolter, K. Stockinger, and A. Bernstein, "A comparative survey of recent natural language interfaces for databases," The VLDB Journal, vol. 28, no. 5, pp. 793-819, 2019.

[36] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in Advances in Neural Information Processing Systems, 2014, pp. 3104-3112.

[37] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in International Conference on Learning Representations, 2015.

[38] K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2014, pp. 1724-1734.

[39] Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL Tools: An Open Source Python Toolkit for Arabic Natural Language Processing. In Proceedings of the Twelfth Language Resources and Evaluation Conference, pages 7022–7032, Marseille, France. European Language Resources Association.

[40] Chen, W. R., Adebara, I., & Abdul-Mageed, M. (2024, June). Interplay of machine translation, diacritics, and diacritization. In Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers) (pp. 7559-7601).