Can LLMs Narrate Tabular Data? An Evaluation Framework for Natural Language Representations of Text-to-SQL System Outputs

Jyotika Singh, Weiyi Sun, Amit Agarwal, Viji Krishnamurthy, Yassine Benajiba, Sujith Ravi, Dan Roth

Oracle AI

Correspondence: jyotika.s.singh@oracle.com

Abstract

In modern industry systems like multi-turn chat agents, Text-to-SQL technology bridges natural language (NL) questions and database (DB) querying. The conversion of tabular DB results into NL representations (NLRs) enables the chat-based interaction. Currently, NLR generation is typically handled by large language models (LLMs), but information loss or errors in presenting tabular results in NL remains largely unexplored. This paper introduces a novel evaluation method - Combo-Eval - for judgment of LLM-generated NLRs that combines the benefits of multiple existing methods, optimizing evaluation fidelity and achieving a significant reduction in LLM calls by 25-61%. Accompanying our method is NLR-BIRD, the first dedicated dataset for NLR benchmarking. Through human evaluations, we demonstrate the superior alignment of Combo-Eval with human judgments, applicable across scenarios with and without ground truth references.

1 Introduction

In real-world Natural Language Processing (NLP) applications built around Large Language Models (LLMs) (Khurana et al., 2022; Vaswani et al., 2017), there has been a growing prominence of communicating using plain text across diverse data types (Soudani et al., 2024; Duan et al., 2024; Liu et al., 2022). Natural language (NL) interfaces to databases (DBs) are increasingly becoming integral in industry applications (Singh, 2023) and agent workflows, leveraging Text-to-SQL systems to convert user questions into structured SQL queries and subsequently presenting query execution results (tables) in an NL format (Developer, 2024; Vijay Venugopal, 2024). This transition from raw tables to NL responses is critical, transforming the impersonal "Computer says no" into user-friendly interactions that enhance accessibility and engagement (Farheen, 2025).

To build conversational applications for DBs, two core components are essential: the generation of SQL queries based on user questions, and the vernacular presentation of the DB outputs, the latter being the focus of our study. Traditionally, the emphasis has been on enhancing SQL generation accuracy (Iacob et al., 2020), leaving a gap in methodologies for evaluating natural language representations of SQL execution results (NLRs). Our research addresses this gap by evaluating LLMs in the task of transforming SQL execution result-sets into natural language but also by proposing enhanced methodology for such evaluations.

Existing evaluations, such as those discussed by Wang et al. (2024), highlight limitations of metrics alone in assessing complex NL tasks. While metrics are not entirely sufficient, our findings suggest they offer valuable signals when paired with LLM-based evaluations. We propose a composite method, Combo-Eval, which combines metrics with LLM-as-a-judge to enhance agreement with human evaluations while significantly reducing computational overhead.

Our contributions include:

- **Combo-Eval Method:** An evaluation technique that synergizes metrics with LLM-assessment, reducing LLM calls while maintaining high evaluation fidelity.
- **NLR-BIRD Dataset:** A new NLR-BIRD dataset spanning 11 domains, enabling evaluation of NLR generation in context of Text-to-SQL systems.
- **Comprehensive Benchmarking:** Evaluation of 15 judge LLMs and multiple scenarios on the task of NLR judging. We also share results across multiple LLMs on the task of NLR generation and share challenges.
- **Evaluation Scenarios:** Comparative analysis with and without ground-truth references

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 883–902

to emulate real-world industry applications where references may be unavailable.

2 Background and Related Work

Research in natural language interfaces to DBs has predominantly focused on two areas: Text-to-SQL conversion (Iacob et al., 2020) and long-form table question answering (LFTQA) (Roychowdhury et al., 2024; Zhao et al., 2024; Nan et al., 2022).

Text-to-SQL targets SQL generation complexities while neglecting the NLR phase critical for user-interactive systems. Emerging studies have begun to explore NL renditions from DB outputs, yet they only scratch the surface with no mentions of evaluation of these NLRs (Asim Biswal, 2025).

Unlike LFTQA, which emphasizes gleaning answers directly from tables, where answers may be a part of the available table, our work centers on narrating complete tabular datasets following SQL query execution. This narrative task demands accurately conveying full table data in natural language, a requirement particularly relevant for Text-to-SQL implementations in conversational systems.

Methods for evaluating data similar to NLRs have been explored in LFTQA (Wang et al., 2024), indicating metrics as a poor judge that fail the evaluation task, and showing a preference for LLM-based evaluation. In our work, we developed the Combo-Eval method that outperforms singular LLM-judges by enhancing utility of metrics and integrating metric-based thresholds.

Despite assumptions that simpler tasks like table-to-NL conversion would be straightforward, our findings reveal that LLMs face significant hurdles even in this domain. Our primary focus in this paper is on evaluation of NLRs generated from DB results across evaluation techniques and models.

More details on related work is in Appendix A.1

3 NLR Dataset

We introduce NLR-BIRD[1] dataset which contains NLRs across questions present in the BIRD-dev dataset (Li et al., 2023), presenting a new data point for NLR, thereby enabling users to test components of DB interaction systems end-to-end, across 11 domains – financial, debit card specializing, formula 1, codebase community, European football 2, student club, California schools, card games, superhero, toxicology, and thrombosis prediction. The data collection process is shared in Appendix A.2.

[1] https://sites.google.com/view/nlr-bird/home

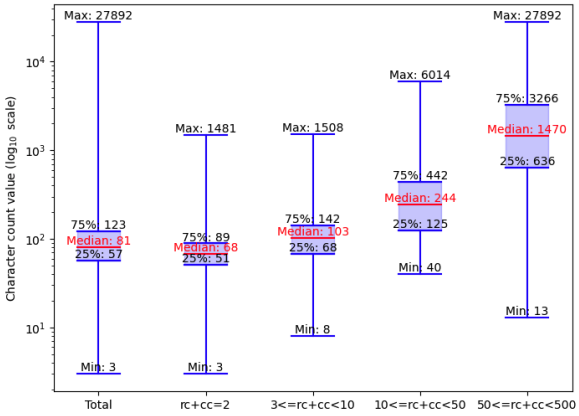

Figure 1: Box plot of percentiles for NLRs in the dataset representing character counts across result-set sizes.

The BIRD dataset contains 1534 NL questions paired with ground truth SQL queries and a BIRD DB. We used row count (rc) and column count (cc) to measure the result size after executing these queries on the DB. Large result sets may benefit from a summarization strategy to formulate the NLR for meaningful rendering and readability. The summary method may depend on the application: some needing a general summary, others requiring partial data representation. To keep a consistent style, large result sets (rc+cc$\geq$500), comprising 4.3% of samples (Figure 7 in Appendix A.3 shows the distribution of result size in BIRD dataset), were excluded. The NLR-BIRD dataset includes 1468 samples across 11 domains, containing human-labeled ground truth NLR for rc+cc<500 that narrates the tabular response generated by SQL execution.

Figure 1 depicts the correlation between the length of NLRs and the result size, indicating a clear upward trend in character count as rc+cc increases. More comprehensive statistics, including word counts, are available in Appendix A.3.

The maximum length of any word in the NLRs is 92 characters, which corresponds to a URL. The NLRs contain 19601 characters corresponding to numbers and 62140 characters corresponding to alphabets.

4 Methods

This section describes the three evaluation methods and the two scenarios in which the methods are applied. One scenario assumes the presence of ground truth (GT) and uses it as a reference to assess the evaluation strategies. The other scenario handles cases where GT is unavailable, employing

source information as a proxy for ground truth.

Given the straightforward nature of transforming simple result-sets (rc=1 and cc=1; total rc+cc=2) into NLRs due to limited information context, our experiments and evaluations emphasize result-sets with rc+cc$\geq$3. We sampled our dataset to represent the different result size categories uniformly. Human evaluators assessed the correctness (labeled 0 or 1) of NLRs (using the same process described for dataset curation in appendix A.2) generated by four LLMs for each sample: Phind-CodeLLma-34B-v2, Llama-3.1-70B-Instruct, Llama-3.1-405B-Instruct (Touvron et al., 2023), and GPT-4o (OpenAI et al., 2024a). This evaluation yielded 660 labeled NLRs corresponding to 165 unique questions, identifying class 0 as incorrect and class 1 as correct.

The above LLM-generated human-assessed NLRs serve as benchmarks to judge different evaluation methods (described in Section 4.2). We assess per-class recall, precision, F1 score, as well as macro recall, precision, F1, and overall accuracy. In scenarios with significant class imbalances, macro-averaging is favored to aptly capture the minority class 0, crucial for industry applications where inaccuracies can detrimentally impact system utility. We present the F1 scores in the main paper and include other scores in the Appendix.

While human-assessed NLRs served as benchmarks to evaluate different methods, we designated 25% of the samples (generated by GPT-4o) as the dev set, which informed our threshold calibration, prompt engineering, and inference parameters. The remaining samples, generated by other models, formed the test set. Results presented hereafter pertain to evaluations on the test set. Details about the experimental setup are in Appendix A.4 and A.5.

4.1 Evaluation Scenarios

We present two scenarios in which our evaluation methods are applied.

1) Model Generated NLR compared to Ground-Truth NLR (**GT**): This approach uses the NLRs from the NLR-BIRD dataset as the reference, assessing the model-generated NLRs against annotated ground-truth NLRs.

2) Model Generated NLR compared to User Question and DB Result-Set (**UQDB**): This is derived by appending the user's question and DB result-set table for the reference text. This does not rely on annotated ground truth NLRs and relies on source information instead.

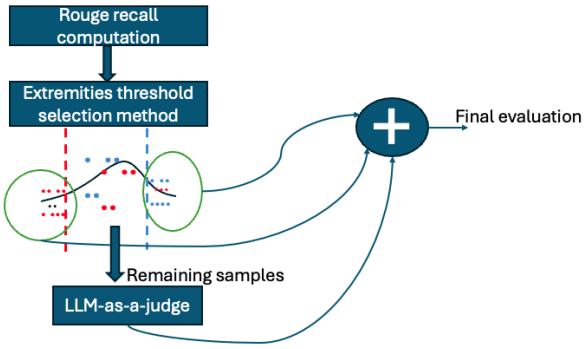

Figure 2: The Combo-Eval method flow combining metrics-based evaluation and LLM-as-a-judge.

4.2 Evaluation Methods

1) **Metrics-as-a-Judge**: Metrics including cosine similarity, BERTScore (Zhang* et al., 2020), and ROUGE scores (Lin, 2004) were considered. A threshold was determined on the dev set to make the decision boundary. Further details can be found in Appendix A.6 and A.7.

2) **LLM-as-a-judge**: Automated evaluation using LLM-judge (Gu et al., 2024; Raina et al., 2024):

3) **Combo-Eval** method combines the above two approaches in an attempt to retain benefits of both to attain superior evaluation along with efficiency advantages by limiting the number of LLM calls.

We started with computing metrics and determined upper and lower thresholds for each class (more details in Appendix A.7). The samples that didn't pass through the extremity threshold were sent to LLM-judge for a finer diagnostic. Figure 2 shows the flow of this method. To represent this mathematically, let's denote:

R (ROUGE-1 recall); C (class label 0 or 1)

L (Output of LLM-as-a-judge – True for class 1 and False for class 0)

th_0l - th_0u are lower and upper thresholds for class 0, and th_1l - th_1u are for class 1.

The classification based on ROUGE score can be represented as:

$C = 1$ if $th_1 1 < R =< th_1u$

$C = 0$ if $th_0l < R =< th_0u$

$C = pending$ otherwise

Then, the final evaluation can be represented

Result size	Phind	L3.1 70	L3.1 405	GPT-4o
3-9	0.80	0.87	0.89	0.91
10-49	0.60	0.77	0.83	0.62
50-499	0.52	0.57	0.59	0.30
Total	0.65	0.75	0.78	0.63

Table 1: Human evaluation of NLRs generated by LLMs - percentage of questions where NLR generated by LLM is rated as acceptable by human evaluators. Phind=Phind-CodeLlama-34B-v2; L3.170=Llama 3.1-70B-instruct; L3.1405=Llama 3.1-405B-instruct.

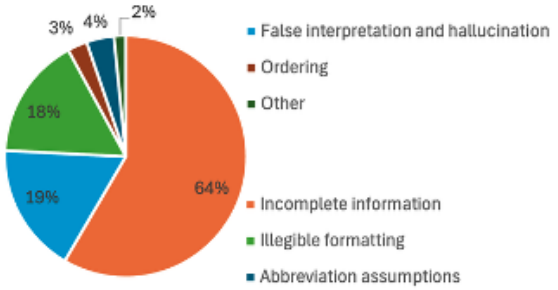

Figure 3: Reasons for incorrect NLRs based on human assessment of model-produced NLRs.

using mathematical notation as

$$C = \begin{cases} 1 & \text{if } th_1 l < R \leq th_1 u \\ & \quad \vee ((th_0 u < R \leq th_1 l) \\ & \quad \vee (R \leq th_0 l) \vee (R > th_1 u)) \wedge L \\ 0 & \text{if } th_0 l < R \leq th_0 u \\ & \quad \vee ((th_0 u < R \leq th_1 l) \\ & \quad \vee (R \leq th_0 l) \vee (R > th_1 u)) \wedge \neg L \end{cases}$$

where $\vee$ represents logical OR, $\wedge$ represents logical AND, and $\neg$ represents logical NOT operation.

5 Results and Discussions

5.1 NLR Generation Model Performance

Quality of LLM-generated NLRs decreases significantly as the result size increases. Table 1 details human evaluation of LLM-generated NLRs, segmented by result size. It reveals a consistent trend where LLMs perform better with smaller result-sets, with performance decreasing 10-30% as result-set size increases from <10 to ≥10. Specifically, larger models like Llama-3.1-405B-instruct generally outperform Phind-CodeLlama-34B-v2, Llama-3.1-70B-instruct, and GPT-4o, particularly on challenging result-set sizes >=10. Notably, GPT-4o excels in the smallest size category of <10 but underperforms with larger sizes.

The most common cause of incorrect NLRs was incomplete information. As indicated in Figure 3, common reasons for incorrect NLRs included missing elements from the result-set rendering the outcome incomplete, hallucinations, rendering results out of order for questions where the order was important, skipping nulls rather than indicating the value was unavailable, formatting inconsistencies impacting readability, and more. Incorrect NLRs at times had inconsistent behavior in de-duplicating result-set values, and inconsistent behavior in interpreting ambiguities in user question or result-set. All identified inaccuracies stem from the LLM generation of NLR, further explained in Appendix A.8. A breakdown of issues across different LLMs and examples of correct and incorrect NLRs are shared in Appendix A.8.

Formatting inconsistencies, particularly in GPT-4o outputs, were prominent with increased result sizes. Discrepancies at the beginning versus the end of NLR text and spacing issues were pronounced, as illustrated in a detailed example in Appendix A.9.

While smaller, structured result-sets are more amenable to LLM processing, larger complex tables pose more challenges. These may exceed LLM context windows, resulting in verbose or impractical NLR outputs for users.

5.2 Comparison Between Evaluation Methods

We evaluated model generated NLRs using three methods (metric-based, LLM-as-a-judge, and Combo-Eval) to analyze how well these methods align with human judgments of NLR completeness/comphrehensiveness, faithfulness, and readability. Our results indicate that Combo-Eval exhibits highest alignment with human judgment.

Metrics-as-a-Judge: Recall-based measures, such as ROUGE, measure stronger differences between correct and incorrect NLRs, compared to other metrics. We calculated various automated metrics between model-generated NLRs against GT and UQDB. As shown in Figure 4, there is a notable differentiation in median metric scores for class 1 (correct NLRs) and class 0 (incorrect NLRs). For a problem of this nature, we anticipate observing a higher frequency of word/n-gram

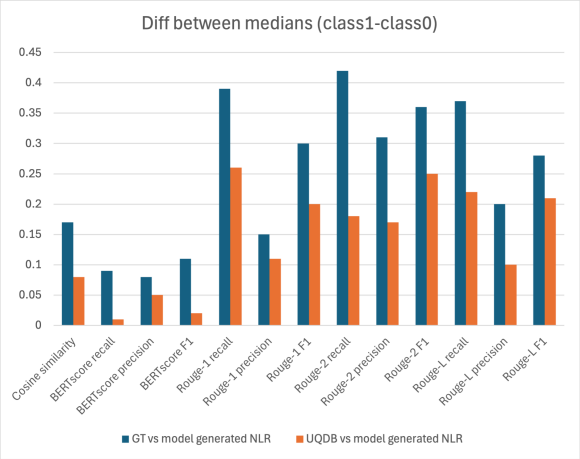

Figure 4: Difference between median scores of class 1 and class 0. Scores are computed between model generated NLRs and (GT (blue) & UQDB (orange).)

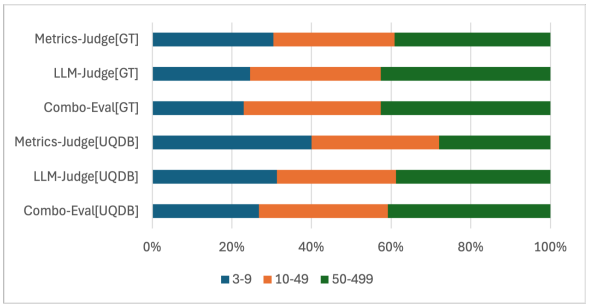

Figure 5: Breakdown of incorrect judgments by result size across evaluation methods (Metrics-based, LLM-judge, and Combo-Eval) for GT and UQDB scenarios, showing higher misjudgments by LLM-as-a-judge on higher result sizes.

matches for effective NLRs. This is because tabular data retrieved from a query often reflects specific values within tables, which may be more appropriately rendered as-is in the NLR, rather than substituting them with semantically similar alternative words. This is likely why BERTscore seems to be comparatively less prominent.

The model-generated NLRs with GT exhibit the most difference in medians between correct and incorrect NLRs, indicating a trend that could aid in evaluating correct versus incorrect NLRs. UQDB follows a similar pattern but with a relatively smaller difference.

We use ROUGE-1-recall metric with a threshold as decision boundary to classify NLRs into correct vs. incorrect based on general applicability of this metric and the maximum median difference between class 1 and 0. Decision boundaries and results for all the metrics are shared in Appendix A.6.

Table 2 reveals that metrics can help in evaluating NLR correctness, although identifying incorrect NLRs remains challenging based on per-class scores. In real-world systems, correctly identifying incorrect NLRs is important and opens opportunities to put correction/flagging measures in place. More statistics and scores are in Appendix A.10.

LLM-as-a-Judge: Table 2 reveals that LLM-based judgment outperforms metric-based methods in both GT and UQDB scenarios. Notably, performance is more robust for class 1, though identifying class 0 instances remains more challenging.

Dissecting false judgments by result size, both UQDB and GT show higher evaluation inaccuracies with larger sizes, as seen in Figure 5. Although LLM-judge yields fewer incorrect judgments than Metrics-judge, it exhibits a higher proportion of false judgments at larger result sizes, as detailed in Appendix A.11. This shows that at larger result sizes, not only is it harder for LLMs to generate correct NLRs (Table 1), but it is also more difficult for LLM-based judges to accurately evaluate model-generated NLRs.

Combo-Eval: Several experiments were conducted including combining metrics and LLM-judge output as features and training a machine learning model which didn't yield a robust model and the overall performance did not improve. We also tried injecting extra knowledge of the metrics into the prompt or LLM-as-a-judge method which did not improve the performance either.

Using a single threshold on ROUGE-recall does not suffice, but the difference between class 1 and class 0 tends to be more informative on the extreme values (more information is shared in Appendix A.7). Combo-Eval effectively transcends limitations associated with single thresholds by dynamically applying metric thresholds and subsequent LLM evaluation. This method demonstrated superior performance, validating its utility across varied judge models, as demonstrated in Figure 6.

For our test set, ***Combo-Eval reduced the LLM calls by 61.43% in GT and 24.48% for UQDB***. This method optimizes the evaluation by passing data through a low-complexity calculation first, and passing only a subset of data through an LLM.

Combo-Eval across Models: As seen in Figure 6, small judge models (e.g., Grok-3-mini, GPT-4.1 nano, GPT-4o mini) exhibit a prominent improvement with the Combo-Eval frame-

GT						UQDB					
Metrics-Judge		LLM-judge		Combo-Eval		Metrics-Judge		LLM-judge		Combo-Eval	
69.73		80.72±0.8		80.88±0.4		68.06		75.82±0.4		76.98±0.4	
C0	C1	C0	C1	C0	C1	C0	C1	C0	C1	C0	C1
53.27 ±0	86.19 ±0	70.79 ±1.1	90.65 ±0.4	73.22 ±0.7	88.55 ±0.2	52.05 ±0	84.06 ±0	61.77 ±0.6	89.87 ±0.3	64.75 ±0.5	89.22 ±0.3

Table 2: F1 average (macro) across classes and F1 score per class, with standard deviation across 10 runs, for evaluation methods - metrics (ROUGE1-recall with threshold to determine decision), LLM-as-a-judge, and Combo-Eval method, across GT and UQDB scenarios. class 0 (C0); class 1 (C1). Judge LLM is GPT-4o.

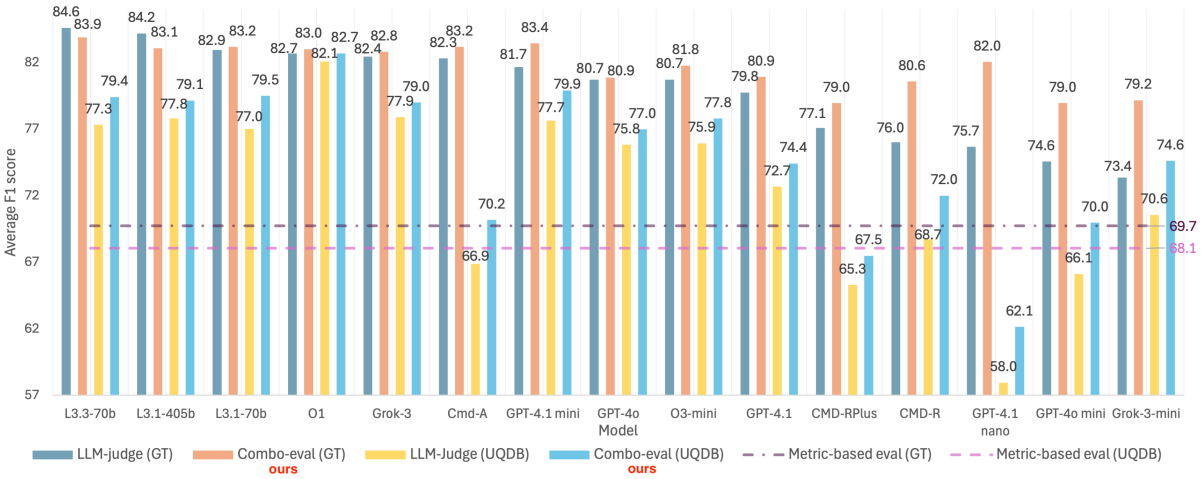

Figure 6: Macro F1 scores across judge LLMs and evaluation methods for GT and UQDB scenarios. This figure shows that Combo-Eval surpasses LLM-judge across different judge LLMs and highlights trends across judge models. LLMs are ordered by their LLM-judge scores on GT, highest to lowest. Results in tabular format are in Appendix A.12

work, demonstrating effective performance at reduced computational costs. Larger models maintain smaller benefits, underscoring Combo-Eval as a strategic choice to balance accuracy and resource consumption. This pattern is seen in both GT and UQDB scenarios. Generally, Combo-Eval outperforms, signifying its robustness in varied scenarios. Also, Combo-Eval's reduced standard deviation in results (Table 2) stems from fewer samples requiring LLM evaluation, stabilizing evaluation consistency.

GT vs. UQDB: Unlike GT, UQDB isn't the ground truth NLR, but the source information that makes up the contents of the NLR. Thus, we see models with stronger reasoning capabilities, such as O1, exhibit consistent performance across GT and UQDB scenarios, doing better in the UQDB scenario compared to other models. On the contrary, most other models do better with GT than UQDB by an average of 7.27% for LLM-judge method and 6.71% for Combo-Eval (Appendix A.12). However, with GT references available, newer Llama models and Grok-3 are formidable contenders, indicating various models' adaptability.

Consistently, GT-based evaluations outperform UQDB, though UQDB offers a viable alternative when GT is inaccessible. UQDB results are much worse for Cmd-A, Cmd-R+, Cmd-R, GPT-4.1 nano, and GPT-4o mini, some even underperforming compared to metrics-judge, indicating potential limitation in inherent reasoning capabilities compared to other models evaluated.

Model Ranking for NLR Generation: Appendix A.13 elucidates rankings among NLR generation models using the three evaluation methods. Our evaluations establish that irrespective of methods, LLM rankings remain consistent, affirming metrics as practical ranking tools for model performance analysis.

Temperature Influence: While low temperature settings are known to keep LLMs judgment closer

to the factual content (Samuylova; Salinas et al., 2025), our experiments show that a judge model's temperature parameter change only minimally enhanced or deteriorated the judgment in different directions for different judge models, showing lack of any notable impact or trend. Combo-Eval had similar improvements over LLM-Judge across temperatures. The trend between LLM-judge and Combo-Eval remained unchanged with temperature changes. Experiment details and results are in Appendix A.14.

Extension and Future Directions: This work lays the foundation for systematic evaluation of existing and emerging LLMs on NLR generation and judgment tasks. Future studies can leverage the benchmark as a new dimension for model comparison and assessment under controlled conditions.

Beyond Text-to-SQL systems, the framework can be extended to other settings that require narrating structured or tabular data. For instance, NLRs can support schema enrichment, expanding database metadata with descriptive language that helps users understand available information and formulate better queries. In interactive systems, such narrations can bridge the gap between user intent and schema representation (e.g., mapping "men's department" to the canonical entity MENS_DEPT), enhancing question understanding.

Another future direction includes comparing Combo-Eval with hard metrics in reasoning-aware datasets such as SciGen (Moosavi et al., 2021), which feature expert-annotated table descriptions requiring arithmetic reasoning.

Insights from our benchmarking results open up opportunities to improve LLM capabilities. Future work may explore improved LLM training strategies or data augmentation methods to improve models' ability to handle large result sets and NLR generation performance.

6 Conclusion

Using NL to communicate with DBs is a valuable industry application. Our study identifies incomplete information as a predominant source of error in NLRs, with model performance deteriorating as result size increases. We introduce Combo-Eval method, a new evaluation framework that aligns closely with human judgment of NLR correctness compared to traditional metrics-based and LLM-based methods. Notably, Combo-Eval achieves this alignment while significantly reducing computational demands by minimizing LLM calls, making it especially advantageous in large-scale or resource-constrained environments. Combo-Eval's differentiated improvements are particularly evident when using smaller judge models, offering a cost-effective solution without compromising accuracy. Our contribution also includes the release of the NLR-BIRD dataset, providing a valuable resource for benchmarking and further research in this domain. Our experiments demonstrate that the presence of ground truth data enhances evaluation accuracy across methods. In scenarios where ground truth is unavailable, UQDB proves to be a viable alternative.

Limitations

Our study, while advancing NLR evaluation methodologies, does present certain limitations. While the NLR-BIRD dataset provides a robust foundation, its domain coverage may not fully capture the diversity of all industrial applications. These aspects highlight the need for continual dataset expansion and further exploration of context-specific evaluation strategies. Additionally, improving NLR generation by LLMs is not the paper's focus. Future LLMs could benefit from training in these areas to enhance their table narration abilities and reduce information loss— the number one reason behind incorrect NLRs as uncovered by our work. Emphasizing completeness in data-based narration can inherently improve LLM performance in this task. Moreover, our analysis shows varied performance based on result-set size which may inform potential next steps in this area. For large result-sets, there is a significant gap in performance, indicating that summarization or rule-based approaches might be needed to ensure NLR completeness.

References

Amit Agarwal, Hansa Meghwani, Hitesh Laxmichand Patel, Tao Sheng, Sujith Ravi, and Dan Roth. 2025a. Aligning llms for multilingual consistency in enterprise applications. *Preprint*, arXiv:2509.23659.

Amit Agarwal, Kulbhushan Pachauri, Iman Zadeh, and Jun Qian. 2024a. Techniques for graph data structure augmentation. U.S. Patent. Issued May 21, 2024.

Amit Agarwal, Srikant Panda, and Kulbhushan Pachauri. 2024b. Synthetic document generation pipeline for training artificial intelligence models. U.S. Patent. Issued January 4, 2024.

Amit Agarwal, Srikant Panda, and Kulbhushan Pachauri. 2025b. FS-DAG: Few shot domain adapting graph networks for visually rich document understanding. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 100–114, Abu Dhabi, UAE. Association for Computational Linguistics.

Siddharth Jha Amog Kamsetty Shu Liu Joseph E Gonzalez Carlos Guestrin Matei Zaharia Asim Biswal, Liana Patel. 2025. Text2sql is not enough: Unifying ai and databases with tag. In *Proceedings of the 15th Annual Conference on Innovative Data Systems Research (CIDR '25)*, Amsterdam, The Netherlands.

Peter Baile Chen, Yi Zhang, and Dan Roth. 2024. Is table retrieval a solved problem? exploring join-aware multi-table retrieval. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2687–2699, Bangkok, Thailand. Association for Computational Linguistics.

Shijie Chen, Ziru Chen, Huan Sun, and Yu Su. 2023a. Error detection for text-to-sql semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, page 11730–11743. Association for Computational Linguistics.

Ziru Chen, Shijie Chen, Michael White, Raymond J. Mooney, Ali Payani, Jayanth Srinivasa, Yu Su, and Huan Sun. 2023b. Text-to-sql error correction with language models of code. In *Annual Meeting of the Association for Computational Linguistics*.

Cohere. a. Cohere's command r model: Command r model details and specifications. https://docs.cohere.com/docs/command-r. Accessed: 2025-07-03.

Cohere. b. Cohere's command r+ model: Command r+ model details and specifications. https://docs.cohere.com/docs/command-r-plus. Accessed: 2025-07-03.

Team Cohere, Aakanksha, Arash Ahmadian, Marwan Ahmed, Jay Alammar, Milad Alizadeh, Yazeed Alnumay, Sophia Althammer, Arkady Arkhangorodsky, Viraat Aryabumi, Dennis Aumiller, Raphaël Avalos, Zahara Aviv, Sammie Bae, Saurabh Baji, Alexandre Barbet, Max Bartolo, Björn Bebensee, Neeral Beladia, and 210 others. 2025. Command a: An enterprise-ready large language model.

Microsoft Developer. 2024. Building the ultimate chatbot on your own data with azure sql and semantic | data exposed. Accessed: 2025-05-17.

Xuemei Dong, Chao Zhang, Yuhang Ge, Yuren Mao, Yunjun Gao, Lu Chen, Jinshu Lin, and Dongfang Lou. 2023. C3: zero-shot text-to-sql with chatgpt. *CoRR*, abs/2307.07306.

Karan Dua, Puneet Mittal, Ranjeet Gupta, and Hitesh Laxmichand Patel. 2025. SpeechWeave: Diverse multilingual synthetic text & audio data generation pipeline for training text to speech models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 718–737, Vienna, Austria. Association for Computational Linguistics.

Haodong Duan, Jueqi Wei, Chonghua Wang, Hongwei Liu, Yixiao Fang, Songyang Zhang, Dahua Lin, and Kai Chen. 2024. BotChat: Evaluating LLMs' capabilities of having multi-turn dialogues. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3184–3200, Mexico City, Mexico. Association for Computational Linguistics.

Riza Farheen. 2025. Building an agentic workflow: Orchestrating a multi-step software engineering interview. Accessed: 2025-05-17.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Yuanzhuo Wang, and Jian Guo. 2024. A survey on llm-as-a-judge. *CoRR*, abs/2411.15594.

Radu Cristian Alexandru Iacob, Florin Brad, Elena-Simona Apostol, Ciprian-Octavian Truică, Ionel Alexandru Hosu, and Traian Rebedea. 2020. Neural approaches for natural language interfaces to databases: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 381–395, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. 2022. Natural language processing: state of the art, current trends and challenges. *Multimedia Tools and Applications*, 82(3):3713–3744.

Haoyang Li, Jing Zhang, Hanbing Liu, Ju Fan, Xiaokang Zhang, Jun Zhu, Renjie Wei, Hongyan Pan, Cuiping Li, and Hong Chen. 2024. Codes: Towards building open-source language models for text-to-sql. *Proceedings of the ACM on Management of Data*, 2(3):1–28.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Ma Chenhao, Guoliang Li, Kevin Chang, Fei Huang, Reynold Cheng, and Yongbin Li. 2023. Can llm already serve as a database interface? a big bench for large-scale database grounded text-to-sqls. In *Advances in Neural Information Processing Systems*, volume 36, pages 42330–42357. Curran Associates, Inc.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Qi Liu, Zihuiwen Ye, Tao Yu, Linfeng Song, and Phil Blunsom. 2022. Augmenting multi-turn text-to-SQL datasets with self-play. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5608–5620, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hansa Meghwani, Amit Agarwal, Priyaranjan Pattnayak, Hitesh Laxmichand Patel, and Srikant Panda. 2025. Hard negative mining for domain-specific retrieval in enterprise systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 1013–1026, Vienna, Austria. Association for Computational Linguistics.

Nafise Sadat Moosavi, Andreas Rücklé, Dan Roth, and Iryna Gurevych. 2021. Scigen: a dataset for reasoning-aware text generation from scientific tables. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, Mutethia Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, Dragomir Radev, and Dragomir Radev. 2022. FeTaQA: Free-form table question answering. *Transactions of the Association for Computational Linguistics*, 10:35–49.

Ali Bou Nassif, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. 2019. Speech recognition using deep neural networks: A systematic review. *IEEE Access*, 7:19143–19165.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, A J Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024a. GPT-4o system card.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, and 244 others. 2024b. Openai o1 system card.

OpenAI. 2025. Openai o3-mini system card. `https://cdn.openai.com/o3-mini-system-card-feb10.pdf`.

Srikant Panda, Amit Agarwal, and Hitesh Laxmichand Patel. 2025a. Accesseval: Benchmarking disability bias in large language models. *Preprint*, arXiv:2509.22703.

Srikant Panda, Vishnu Hari, Kalpana Panda, Amit Agarwal, and Hitesh Laxmichand Patel. 2025b. Who's asking? investigating bias through the lens of disability framed queries in llms. *Preprint*, arXiv:2508.15831.

Srikant Panda, Hitesh Laxmichand Patel, Shahad Al-Khalifa, Amit Agarwal, Hend Al-Khalifa, and Sharefah Al-Ghamdi. 2025c. Daiq: Auditing demographic attribute inference from question in llms. *Preprint*, arXiv:2508.15830.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, Online. Association for Computational Linguistics.

Hitesh Laxmichand Patel, Amit Agarwal, Arion Das, Bhargava Kumar, Srikant Panda, Priyaranjan Pattnayak, Taki Hasan Rafi, Tejaswini Kumar, and Dong-Kyu Chae. 2025. Sweeval: Do llms really swear? a safety benchmark for testing limits for enterprise use. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pages 558–582.

Hitesh Laxmichand Patel, Amit Agarwal, Bhargava Kumar, Karan Gupta, and Priyaranjan Pattnayak. 2024. Llm for barcodes: Generating diverse synthetic data for identity documents. *arXiv preprint arXiv:2411.14962*.

Priyaranjan Pattnayak, Amit Agarwal, Hansa Meghwani, Hitesh Laxmichand Patel, and Srikant Panda. 2025. Hybrid AI for responsive multi-turn online conversations with novel dynamic routing and feedback adaptation. In *Proceedings of the 4th International Workshop on Knowledge-Augmented Methods for Natural Language Processing*, pages 215–229, Albuquerque, New Mexico, USA. Association for Computational Linguistics.

Priyaranjan Pattnayak, Hitesh Laxmichand Patel, Bhargava Kumar, Amit Agarwal, Ishan Banerjee, Srikant Panda, and Tejaswini Kumar. 2024. Survey of large multimodal model datasets, application categories and taxonomy. *arXiv preprint arXiv:2412.17759*.

Mohammadreza Pourreza and Davood Rafiei. 2024. DIN-SQL: decomposed in-context learning of text-to-SQL with self-correction. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23)*, volume 1577, pages 36339–36348, Red Hook, NY, USA, Article. Curran Associates Inc.

Vyas Raina, Adian Liusie, and Mark Gales. 2024. Is LLM-as-a-judge robust? investigating universal adversarial attacks on zero-shot LLM assessment. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7499–7517, Miami, Florida, USA. Association for Computational Linguistics.

Sohini Roychowdhury, Marko Krema, Anvar Mahammad, Brian Moore, Arijit Mukherjee, and Punit Prakashchandra. 2024. Eratta: Extreme rag for table to answers with large language models. *arXiv preprint*.

David Salinas, Omar Swelam, and Frank Hutter. 2025. Tuning LLM judge design decisions for 1/1000 of

the cost. In *Forty-second International Conference on Machine Learning*.

Elena Samuylova. Llm-as-a-judge: a complete guide to using llms for evaluations. Last updated: 23 July 2025, Accessed: 2025-10-08.

Jyotika Singh. 2021. Social media analysis using natural language processing techniques. In *Proceedings of the 20th Python in Science Conference*, SciPy, page 74–80. SciPy.

Jyotika Singh. 2022. pyaudioprocessing: Audio processing, feature extraction, and machine learning modeling. In *Proceedings of the 21st Python in Science Conference*, SciPy, page 152–158. SciPy.

Jyotika Singh. 2023. *Natural Language Processing in the Real World: Text Processing, Analytics, and Classification*. Chapman and Hall/CRC.

Jyotika Singh, Rebecca Bilbro, Michael Avon, Scott Bowen, Dan Jolicoeur, and Serge Matta. 2021. Method for optimizing media and marketing content using cross-platform video intelligence. U.S. Patent. Issued March 16, 2021.

Heydar Soudani, Roxana Petcu, Evangelos Kanoulas, and Faegheh Hasibi. 2024. A survey on recent advances in conversational data generation. *Preprint*, arXiv:2405.13003.

Ruoxi Sun, Sercan Arik, Rajarishi Sinha, Hootan Nakhost, Hanjun Dai, Pengcheng Yin, and Tomas Pfister. 2023. SQLPrompt: In-context text-to-SQL with minimal labeled data. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 542–550, Singapore. Association for Computational Linguistics.

Shayan Talaei, Mohammadreza Pourreza, Yu-Chen Chang, Azalia Mirhoseini, and Amin Saberi. 2024. Chess: Contextual harnessing for efficient sql synthesis. *arXiv preprint*.

Yuan Tian, Zheng Zhang, Zheng Ning, Toby Li, Jonathan K. Kummerfeld, and Tianyi Zhang. 2023. Interactive text-to-SQL generation via editable step-by-step explanations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16149–16166, Singapore. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30 of *NIPS'17*. Curran Associates, Inc.

Greg Michnikov Vijay Venugopal. 2024. Chat with your business data - conversational analytics comes to gemini in looker. Accessed: 2025-05-17.

Tianshu Wang, Xiaoyang Chen, Hongyu Lin, Xianpei Han, Le Sun, Hao Wang, and Zhenyu Zeng. 2025. Dbcopilot: Natural language querying over massive databases via schema routing. In *EDBT*, pages 707–721.

Yuqi Wang, Lyuhao Chen, Songcheng Cai, Zhijian Xu, and Yilun Zhao. 2024. Revisiting automated evaluation for long-form table question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14696–14706, Miami, Florida, USA. Association for Computational Linguistics.

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.

xAI. Models and pricing. `https://docs.x.ai/docs/models`. Accessed: 2025-07-03.

Wenbo Xu, Liang Yan, Peiyi Han, Haifeng Zhu, Chuanyi Liu, Shaoming Duan, Cuiyun Gao, and Yingwei Liang. 2024. Tcsr-sql: Towards table content-aware text-to-sql with self-retrieval. *arXiv preprint*.

Jiaxi Yang, Binyuan Hui, Min Yang, Jian Yang, Junyang Lin, and Chang Zhou. 2024. Synthesizing text-to-SQL data from weak and strong LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7864–7875, Bangkok, Thailand. Association for Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yilun Zhao, Lyuhao Chen, Arman Cohan, and Chen Zhao. 2024. TaPERA: Enhancing faithfulness and interpretability in long-form table QA by content planning and execution-based reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12824–12840, Bangkok, Thailand. Association for Computational Linguistics.

A Appendix

A.1 Related Work and Background

Large language models (LLMs) have become foundational tools in natural language processing (NLP), a field that enables diverse domain analysis across modalities such as speech (Nassif et al.,

2019; Singh, 2022) and text. LLMs have rapidly transformed real-world applications such as visual question answering (Pattnayak et al., 2024, 2025), product or content search and analysis (Meghwani et al., 2025; Singh et al., 2021; Singh, 2021), interactive assistants, document understanding, (Patel et al., 2024, 2025; Agarwal et al., 2025a,b), accessibility for visually impaired users (Panda et al., 2025a,b,c), and synthetic data-pipelines (Dua et al., 2025; Agarwal et al., 2024a,b). Structured data understanding and connecting language interfaces to structured data has gained more momentum in the last few years.

Traditionally in published studies on natural language interfaces to DBs, the focus has been either on converting text to SQL (Iacob et al., 2020) or long-form table QA which stands for trying to extract answers present in tables directly, without using SQL.

Surrounding text-to-SQL-based applications, a lot of work has been presented such as question/-query decomposition for breaking down the query into sub-queries and combining the results (Pourreza and Rafiei, 2024), SQL correction for self-checking syntactical issues in the generated query (Chen et al., 2023a,b), producing SQL in the presence large schema and multi-table retrievals (Chen et al., 2024; Xu et al., 2024), schema linking (Talaei et al., 2024), noise in SQL generation (Wretblad et al., 2024), SQL generation systems (Tian et al., 2023), scaling (Wang et al., 2025) and more to get better SQL generations from models (Dong et al., 2023; Yang et al., 2024; Sun et al., 2023; Li et al., 2024). However, this primarily targets complexities around the SQL generation part and the focus on industry applications consuming the results of such a system has been missing. In conversational systems with Text-to-SQL backing, the user asks a question in NL and also expects the response back in NL rather than structured table or JSON type of representation. One of the mentions of NLR generation is in (Asim Biswal, 2025), where the work briefly touched upon generating responses from database results in the context of questions requiring a combination of database schema and real-world knowledge. However, there have been no mentions of evaluation of these NL responses. Thus, we find that there has been limited work done on the natural language representations of the result-set fetched after the produced SQL is run against the DB. Existing work (Asim Biswal, 2025) uses language models to handle such conversions and no study presents how well language models are able to represent such data in NL, nor any evaluation methods have been discussed for this specific problem. Errors/information loss induced by this NLR generation step is not studied.

There has also been work in extracting NL answers from tables directly, specifically around short-form and long-form Table QA (LFTQA) (Roychowdhury et al., 2024) (Zhao et al., 2024) (Nan et al., 2022). Most of these tasks have been around answering questions where the answer to the question needs to be looked up in tables. In essence, partial information from the table forms the final answer, combined with other general/analytical reasoning in some of the benchmarks. On the contrary, our work is focussed on the task of narrating a tabular answer to a user question in natural language. The task is not looking for answers in the table or relying on reasoning from LLMs to fetch answers from tables, but in majority of the questions, the full table provided in the actual answer and the job of the model is to correctly describe the information in the table in natural language given the user question's context. In particular, the benchmark we provide is applicable to Text-to-SQL applications in the real world including conversational systems and are different from LFTQA.

The work presented in (Parikh et al., 2020) focuses on obtaining descriptions from highlighted sections of tables. It proposes a controlled generation task where, given a Wikipedia table and highlighted cells, the goal is to generate a descriptive sentence. However, this dataset does not explicitly represent the verbalization of tables in a Text-to-SQL application, and there is currently no benchmark available for this specific task.

While evaluation methods for NLR generation from tabular result-set have not been explored, there has been work in evaluating responses extracted from tabular sources in long-form Table QA. We draw from LFTQA research on evaluation (Wang et al., 2024) where it shows that existing automatic metrics fail in assessing the answers generated by LLM-based systems (e.g.,BLEU, ROUGE, METEOR, and TAPAS-Acc). LLM-based metrics demonstrate a significant improvement over traditional automated metrics in terms of correlation with human evaluation. Even though the task we are discussing here is not LFTQA, it is related to tables and thus we can derive some learning from the field. We evaluate metrics and LLM based eval-

uation methods for NLRs as well. We use similar criteria for evaluating NLRs using LLM-as-a-judge for faithfulness and comprehensiveness. We also take it forward and present a Combo-Eval method that uses both metrics and LLM-judge, and surpasses LLM-judge's performance in addition to reducing number of LLM calls required.

If LLMs can handle long-form table question answering, a task that seems more complex than converting tables to natural language, one might assume that the latter would be a relatively simple task lacking challenges. However, this is not the case. Even long-form question answering on tables remains an unsolved problem, posing numerous challenges. particularly in industrial settings, where accuracy is paramount, limiting its industry adoption. Our study reveals that even for table-to-NL tasks, LLMs struggle to perform effectively. In this paper, our prime focus is on evaluation of these NLRs generated from tables across evaluation techniques and models.

A.2 Data Curation Process

The NLR-BIRD dataset was developed by extracting and processing SQL queries from the BIRD-dev dataset using SQLite. These queries were executed against the database to retrieve the result sets, serving as the basis for creating natural language responses (NLRs).

Labeling involved contributions from four professionals with backgrounds in database management, data science, engineering, and related domains. To streamline the annotation process, a subset of queries was initially processed by various LLMs to create a baseline. Annotators improved upon these by either refining LLM-generated responses or crafting new NLRs from scratch.

For quality assurance, each annotator began with a small set of random samples, both unlabeled and baseline-labeled, to calibrate their understanding of the task. Through iterative discussions, annotators developed shared criteria for assessing NLR correctness:

- An NLR is considered correct if it comprehensive and encapsulates all pertinent information from the result-set relevant to the user's question.

- A correct NLR should exhibit faithfulness. Inclusion of excessive, unverifiable information renders an NLR incorrect.

- A correct NLR should be readable. While a consistent format is encouraged, variations are permissible if readability and completeness are maintained.

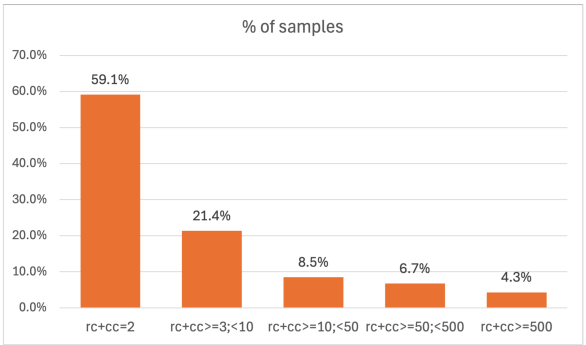

Figure 7: Result size distribution in the BIRD-dev dataset. rc=row count; cc=column count.

Example of handling ambiguous queries include interpreting "Which constructors have been ranked 1?" where the SQL-derived DB results list multiple constructors. Annotators were instructed to treat all listed constructors as sharing rank 1, consistent with the ground-truth logic used to generate the DB results.

A post-annotation review process involved multiple reviewers to cross-check agreements, achieving an 83.3% agreement rate among annotators.

A.3 Dataset Attributes

Figure 7 shows the result size disctribution in the BIRD-dev dataset. The NLR-BIRD datasets contain NLRs for all result sizes rc+cc<500.

In the NLR-BIRD dataset, he maximum word length is 92 characters, typically representing URLs. NLRs are composed of 19,601 numeric characters and 62,140 alphabetic characters.

Table 3 illustrates the distribution of character and word counts for NLRs across varied result-set sizes.

Figure 8 and Figure 9 depict the word count distribution and character count variation among NLRs, respectively.

A.4 Prompts

Prompt for NLR generation:

```
tbl_str = table.to_json(orient="
    records", lines=True) # table
    is a pandas dataframe

prompt = """For the following
    question, use the Answer
    provided to generate a
    response in plain text. Do not
     make up any information, only
```

	Total	rc+cc = 2	3<= rc+cc <10	10<= rc+cc <50	50<= rc+cc <500
count	1468	907	328	130	103
mean(c)	311	74	120	434	2853
mean(w)	49	13	20	65	444
std(c)	1421	62	112	659	4611
std(w)	236	10	19	100	784
min(c)	3	3	8	40	13
min(w)	1	1	2	5	5
25%(c)	57	51	68	125	636
25%(w)	9	8	12	22	114
50%(c)	81	68	103	244	1470
50%(w)	14	12	18	40	202
75%(c)	123	89	142	442	3266
75%(w)	21	16	23	67	449
max(c)	27892	1481	1508	6014	27892
max(w)	4931	237	252	926	4931

Table 3: NLR statistics: Character (c) and word (w) counts across result sizes (rc+cc) and overall.

```
    use what can be found in the
   table to return the plain text
    response. Do not compute any
   trend. Do not calculate any
   numbers. Do not miss any
   answer row.


Question: {question}

Answer: {tbl_str}

Response:"""
```

LLM judge prompt:

```
judge_prompt = """Question: {q}

Actual answer:{ip}

Model generated answer:{op}

For the above question, you have
   the correct answer (`Actual
   answer`) known and a model
   generated answer. Compare if
   the model generated answer
   contains complete and correct
   information to answer the
   question as the actual answer.
```

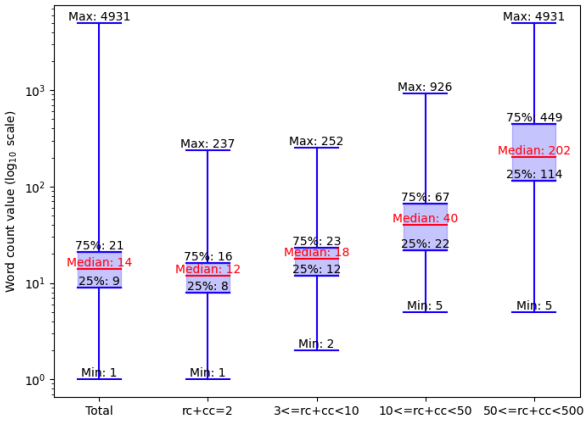

Figure 8: Box plot for word count distribution in NLRs across various result-set sizes.

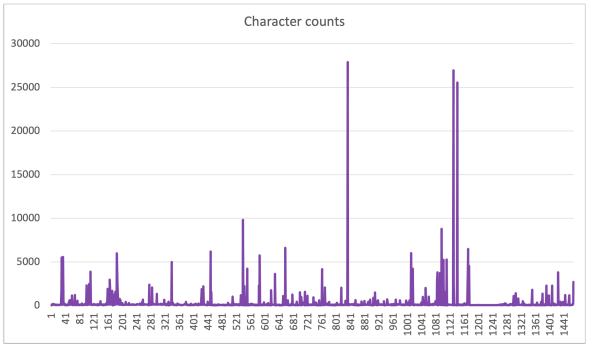

Figure 9: Character count distribution in NLRs of the NLR-BIRD dataset. (x-axis: sample number, y-axis: character count.)

```
It is ok if the format is
   different but core information
    in the model generated answer
    should match the correct
   answer as it relates to the
   question.
Does the model generated answer
   hold the same information as
   the actual answer? Say True or
    False.

Evaluation:"""
```

The value for ip above is the ground truth NLRs for GT and the user question + db results for UQDB.

A.5 Inference Parameter Settings

The following settings were used where applicable.

- max_new_tokens: 2000
- temperature: 0.01
- top_p: 0.95

- top_k: 10
- frequency_penalty: 1.1

A.6 Metrics

Table 4 contains the different metrics considered for metric-based evaluation, along with a threshold for each to determine decision boundary and results representing alignment with human assessment. The thresholds were determined on the dev set and the results shared below reflect the results of the thresholds applied on the test set.

We calculated ROUGE-1, ROUGE-2, ROUGE-L, BERTscore, and cosine similarity for the model generated NLRs (compared to both GT and UQDB) and computed their medians per class Table 5 shows the results.

A.7 Thresholds Used for Experiments

Metrics:

A threshold was determined on the dev set to make the decision boundary using grid search. This was done while optimizing for macro F1 score. We searched for this threshold for ROUGE1-recall metric, which was choosen specifically for its simplicity, general applicability on different types of text, and large difference in median scores between class 0 and class 1.

Threshold of 0.45 for GT and 0.4 for UQDB yielded best macro F1 score (across class 0 and class 1) on the dev set. The Results section shared the results from application of these thresholds on the test set.

Combo-Eval:

As shown in Figure 10, the extreme scores provide a better indication of the accuracy of an NLR. The difference between class 1 and class 0 ROUGE recall scores tends to be more informative on the extreme values, where the difference between % of samples from class 1 and class 0 for the score threshold is not only high but also the class with lower number is samples has a very small presence on those scores.

After conducting a grid search on the lower and upper extremes of ROUGE-1 recall scores using the development set, we established thresholds for both ends. This grid search was optimized for macro F1 score to best align with human assessments of NLRs, where F1 scores were averaged for class 0 (NLRs deemed incorrect by humans) and class 1 (NLRs deemed correct by humans).

The following thresholds were used.

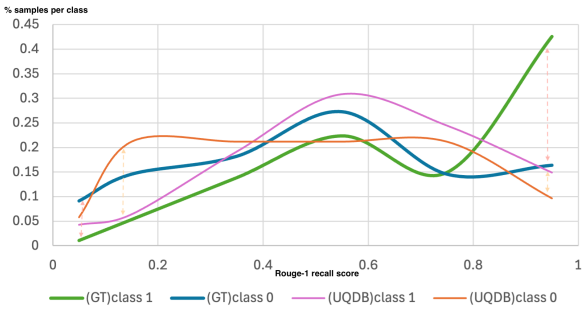

Figure 10: Trend for % of samples with different ROUGE 1 recall scores for class 1 and for class 0 for GT vs. model outputs and UQDB vs. model outputs. The plot shows the extreme scores tend to show more difference between class 1 and class 0, indicating the potential for choosing thresholds that are more likely to contain true positives and true negatives, and minimizing false positives and false negatives.

$th_0l = 0$ for GT; 0.05 for UQDB
$th_0u = 0.1$ for GT; 0.1 for UQDB
$th_1l = 0.9$ for GT; 0.87 for UQDB
$th_1u = 1$ for GT and UQDB

These thresholds were then applied to test set and that represents the numbers shared in the Results section 5.

A.8 LLM Generation Errors

Some errors produced by LLM generated NLRs have common grounds for what went wrong. Figure 11 shows the trends for breakdown of the error types.

Clarifying the source of errors, all identified inaccuracies stem from LLM-generation of NLR, as SQL execution was based on ground-truth SQLs provided in the BIRD dataset, assumed to represent correct data.

Examples of correct and incorrect NLRs by generation models are shared below.

- **Model**: Phind-CodeLLma-34B-v2

  Correct NLR:

  Q: "In which city can you find the school in the state of California with the lowest latitude coordinates and what is its lowest grade? Indicate the school name.

  NLR: "The school with the lowest latitude coordinates in the state of California is Central Middle - RISE located in Oroville. This school offers classes from grade 7 onwards."

  Incorrect NLR:

Metric	Thresholds	Recall	Prec	F1	Recall	Prec	F1
		GT			UQDB		
Cosine similarity	GT=0.7 UQDB=0.65	0.65	0.62	0.63	0.61	0.6	0.6
BERTscore recall	GT=0.3 UQDB=0.3	0.54	0.69	0.6	0.59	0.77	0.67
BERTscore precision	GT=0.7 UQDB=0.6	0.65	0.67	0.66	0.55	0.63	0.59
BERTscore F1	GT=0.45 UQDB=0.41	0.56	0.66	0.6	0.56	0.68	0.62
ROUGE-1 recall	GT=0.45 UQDB=0.4	0.68	0.71	0.7	0.68	0.68	0.68
ROUGE-1 precision	GT=0.8 UQDB=0.65	0.63	0.62	0.63	0.56	0.58	0.57
ROUGE-1 F1	GT=0.7 UQDB=0.5	0.72	0.69	0.71	0.69	0.71	0.7
ROUGE-2 recall	GT=0.4 UQDB=0.3	0.72	0.7	0.71	0.68	0.66	0.67
ROUGE-2 precision	GT=0.7 UQDB=0.43	0.65	0.62	0.64	0.58	0.58	0.58
ROUGE-2 F1	GT=0.45 UQDB=0.43	0.71	0.68	0.69	0.67	0.64	0.66
ROUGE-L recall	GT=0.45 UQDB=0.4	0.69	0.71	0.7	0.67	0.67	0.67
ROUGE-L precision	GT=0.8 UQDB=0.42	0.63	0.61	0.62	0.54	0.61	0.57
ROUGE-L F1	GT=0.7 UQDB=0.5	0.71	0.66	0.69	0.7	0.67	0.68

Table 4: Decision boundary thresholds (determined using the dev set) across various metrics, along with results on test set (macro average recall, precision, and F1 scores) indicating alignment with human assessments of NLR correctness. Thresholding on ROUGE scores aligns more closely with human assessments than cosine similarity and BERT scores.

Metric	Median of the metric computed between model generated NLR and			
	GT		UQDB	
	class 0	class 1	class 0	class 1
Cosine similarity	0.58	0.75	0.64	0.72
BERTscore recall	0.70	0.79	0.51	0.52
BERTscore precision	0.74	0.82	0.68	0.73
BERTscore F1	0.70	0.81	0.58	0.60
ROUGE-1 recall	0.47	0.86	0.39	0.65
ROUGE-1 precision	0.78	0.93	0.81	0.92
ROUGE-1 F1	0.53	0.83	0.49	0.69
ROUGE-2 recall	0.27	0.69	0.22	0.4
ROUGE-2 precision	0.48	0.79	0.5	0.67
ROUGE-2 F1	0.31	0.67	0.24	0.49
ROUGE-L recall	0.44	0.81	0.36	0.58
ROUGE-L precision	0.71	0.91	0.73	0.83
ROUGE-L F1	0.48	0.76	0.42	0.63

Table 5: Median values broken down by human evaluation category (0-NLRs deemed incorrect by human; 1-NLRs deemed correct by human) across scores (cosine similarity, BERTscore, and ROUGE) representing computation of model produced NLRs wrt GT NLRs as well as wrt UQDB.

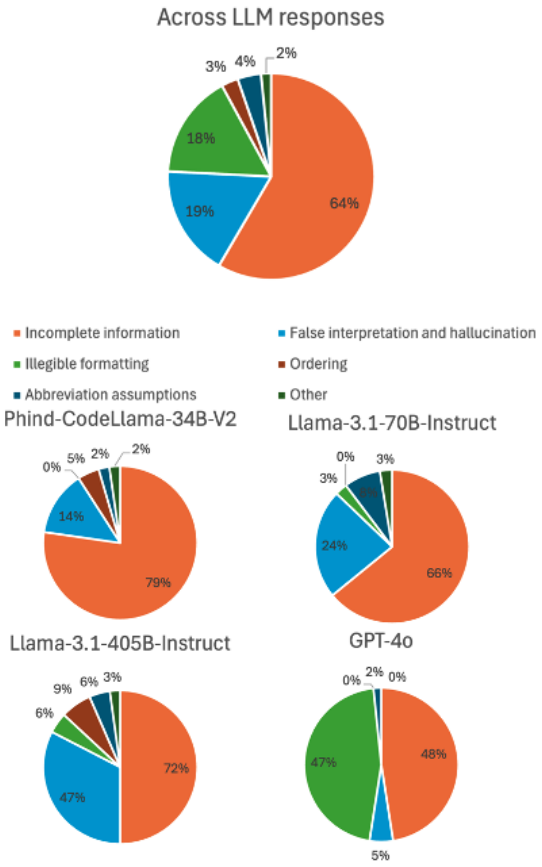

Figure 11: Reasons for inaccurate NLRs - overall and by LLM.

Q: "What are the webpages for the Los Angeles County school that has between 2,000 and 3,000 test takers?"

NLR: "The webpage for the Los Angeles County school with between 2,000 and 3,000 test takers is www.avdistrict.org."

NLR is missing information. The DB result set returned websites 'www.avdistrict.org', 'www.lbschools.net', and 'www.hartdistrict.org'.

- **Model**: Llama3.1-70B-Instruct

  Correct NLR:

  Q: "Give the code of sets have expansion commander type?"

  NLR: "The sets with expansion commander type are C13, C14, C15, C16, C17, CM1, and CMD.

  Incorrect NLR:

  Q: "When was the set of cards with "Ancestor's Chosen" released?"

  NLR: "The set of cards with "Ancestor's Chosen" was released on 2002-05-27."

  This is incorrect because the DB results returned three dates (['2007-07-13'], ['2002-05-27'], ['2018-12-07']), but the LLM assumed only one is correct.

- **Model**: Llama-3.1-405B-Instruct

  Correct NLR:

  Q: "List the football players with a birthyear of 1970 and a birthmonth of October."

  NLR: "Chris Swailes, Dean Kiely, Edwin van der Sar, and Sander Boschker"

  Incorrect NLR:

  Q: "Which set is not available outside of the United States and has foil cards with Japanese writing on them? Please include the set ID in your response."

  NLR: "Commander's Arsenal (50) is not available outside of the United States and has foil cards with Japanese writing on them."

  Names and IDs returned included ["Commander's Arsenal", '50'], ['Premium Deck Series: Fire and Lightning', '307'], ['From the Vault: Exiled', '522'], ['From the Vault: Relics', '523'], ['From the Vault: Legends', '524'], ['From the Vault: Realms', '525'], ['From the Vault: Twenty', '526'], ['From the Vault: Annihilation', '527'], ['From the Vault: Angels', '528'], ['From the Vault: Lore', '529']. The NLR contained incomplete information.

- **Model**: GPT4o

  Correct NLR:

  Q: "What is the sum that client number 4's account has following transaction 851? Who owns this account, a man or a woman?"

  NLR:"The sum that client number 4's account has following transaction 851 is 15096. The owner of this account is a man."

  Incorrect NLR:

  Q: "What is the amount of debt that client number 992 has, and how is this client doing with payments?"

  NLR: "Client number 992 has a debt amount of 53076 and is doing well with payments, as indicated by the status "A"."

Assumption for the meaning of A is considered incorrect. For another example of a bad NLR for this model, see Appendix A.9.

A.9 Format Inconsistency Error Example

While manually reviewing LLM generation of NLRs, we noticed format inconsistencies impacting readability of NLRs, specially in high volumes for GPT-4o as the generation model. An example of such an NLR is as follows.

User question: Provide the IDs and age of the client with high level credit card, which is eligible for loans.

NLR generated based on DB results (21 rows) is shared below. For readability and space, some lines were truncated (represented by symbol <...TRUNC>) where the same format pattern was repeating.

> Account ID: 66, District Name: Rychnov nad Kneznou, District Region: east Bohemia
>
> Account ID: 273, District Name: Karlovy Vary, District Region: west Bohemia
>
> Account ID: 485, District Name: Kutna Hora, District Region: central Bohemia
>
> Account ID: 539, District Name: Rakovnik, District Region: central Bohemia
>
> Account ID: 1050, District Name: Hodonin, District Region: south Moravia
>
> Account ID: 1286, District Name: Zdar nad Sazavou, District Region:
>
> south Moravia
>
> Account ID:
>
> 1527,
>
> District Name:
>
> Tachov,
>
> District Region:
>
> west Bohemia
>
> <...TRUNC>
>
> Account ID:
> 1965,
> District Name:
> Hodonin,
> District Region:
>
> south Moravia
>
> Account ID:
>
> 2137,
>
> District Name:
>
> Kladno,
>
> District Region:
>
> central Bohemia
>
> Account
>
> ID:
>
> 2464,
>
> <...TRUNC>

There were multiple new lines between lines above randomly distributed, which were removed from the above response to enhance readability of the text.

This was seen more predominantly in large result-set sizes.

A.10 Evaluation Results using GPT-4o as the Judge Model

Table 6 presents how well different NLR evaluation methods align with human assessments of NLRs. It includes detailed scores for recall, precision, and F1, both at the class level and averaged (macro level). The evaluation methods covered are Metrics-as-a-Judge, LLM-as-a-Judge, and Combo-Eval. This table offers broader results than Table 2, which is limited to F1 scores, by presenting a wider range of statistical metrics.

With GPT-4o as the judge model, Combo-Eval achieves only modest performance gains compared to the LLM-judge approach. Nonetheless, it maintains its overall performance while requiring fewer LLM calls. Conversely, as shown in Table 8, when other judge models are used, Combo-Eval offers greater performance gains while simultaneously reducing the number of LLM calls. The reduction in LLM calls remains constant regardless of which judge LLM is being used.

A.11 Misjudgment by Result Size

Table 7 presents the distribution of falsely judged NLRs categorized by the size of the results. As demonstrated in Figure 12, both LLM-judge and

	Metrics-judge		LLM-judge		Combo-Eval	
	C0	C1	C0	C1	C0	C1
GT						
Recall	46.9	89.69	65.78±1.4	92.92±0.7	71.29±1.0	89.26±0.2
Prec	61.63	82.95	76.65±1.7	88.49±0.4	75.26±0.5	87.85±0.3
F1	53.27	86.19	70.79±1.1	90.65±0.4	73.22±0.7	88.55±0.2
Rec(macro)	68.30		79.35±0.7		80.27±0.5	
Prec(macro)	72.29		82.57±0.9		81.56±0.4	
F1(macro)	69.73		80.72±0.8		80.88±0.4	
Accuracy	78.52		85.83±0.6		85.07±0.3	
UQDB						
Recall	50.44	84.06	49.56±0.9	96.15±0.7	55.05±0.9	93.72±0.6
Prec	53.77	84.06	82.02±2.4	84.37±0.2	78.65±2	85.14±0.2
F1	52.05	84.06	61.77±0.6	89.87±0.3(SD)	64.75±0.5	89.22±0.3
Rec(macro)	67.25		72.85±0.3		74.39±0.3	
Prec(macro)	68.92		83.19±1.2		81.9±1	
F1(macro)	68.06		75.82±0.4		76.98±0.4	
Accuracy	75.29		83.99±0.4		83.99±0.4	

Table 6: Results (with standard deviation where applicable across 10 runs) from evaluation methods (ROUGE1-recall with threshold to determine classification, LLM as a judge and Combo-Eval method) across class 0 (C0) and class 1 (C1), and average recall, precision, and F1, along with the overall accuracy. Judge LLM is GPT-4o.

	Result size		
	3-9	10-49	50-499
Metrics-Judge			
GT	30.44	30.44	39.13
UQDB	40.00	32.00	28.00
LLM-judge			
GT	22.95	34.43	42.62
UQDB	26.76	32.39	40.85
Combo-Eval			
GT	24.59	32.79	42.62
UQDB	31.34	29.85	38.81

Table 7: Percentage of samples that were falsely judged by LLM-judge method broken down by result size.

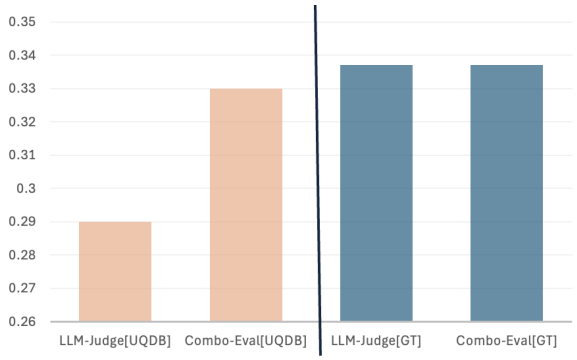

Figure 12: Percentage reduction in incorrect NLR judgments for LLM-judge and Combo-Eval methods compared to Metrics-based evaluation method in UQDB (left) and GT (right) scenarios.

Combo-Eval methods lead to a reduction in the overall number of incorrect judgments compared to the traditional metrics-based evaluation. However, among the incorrect judgments that remain for each method, Table 7 details the percentage breakdown by result size.

The LLM-as-a-judge method consistently encounters greater difficulty in accurately judging larger result sizes compared to smaller ones.

A.12 Judgment by Different Judge Models

Table 8 shows F1 macro scores across 15 judge LLMs for GT and UQDB scenarios for the LLM-as-a-judge and Combo-Eval methods. The judge LLMs include both closed-source and open-source models of different sizes.

- Llama series instruct models (3.3-70B, 3.1-70B, and 3.1-405B)
- OpenAI GPT-4o (OpenAI et al., 2024a), 4o mini, 4.1, 4.1 mini, 4.1 nano
- OpenAI advance reasoning models including O1 (OpenAI et al., 2024b) and O3 mini (OpenAI, 2025)

Judge model	GT		UQDB	
	LLM-as-a-judge	Combo-Eval ***(ours)***	LLM-as-a-judge	Combo-Eval ***(ours)***
L3.3-70b	**84.60**	83.90	77.32	**79.40**
L3.1-405b	**84.18**	83.08	77.79	**79.14**
L3.1-70b	82.93	**83.17**	77.01	**79.5**
O1	82.68	**83.00**	82.08	**82.68**
Grok-3	82.43	**82.80**	77.89	**78.99**
Cmd-A	82.31	**83.17**	66.86	**70.18**
GPT-4.1 mini	81.65	**83.44**	77.65	**79.90**
GPT-4o	80.72	**80.88**	75.82	**76.98**
O3-mini	80.71	**81.76**	75.94	**77.79**
GPT-4.1	79.75	**80.91**	72.67	**74.42**
CMD-RPlus	77.08	**78.95**	65.29	**67.48**
CMD-R	76.02	**80.57**	68.70	**72.0**
GPT-4.1 nano	75.66	**82.04**	57.95	**62.14**
GPT-4o mini	74.56	**78.95**	66.12	**69.97**
Grok-3-mini	73.37	**79.17**	70.56	**74.61**
Average	79.91	**81.72**	72.64	**75.01**

Table 8: F1 macro scores across judge LLMs under GT and UQDB scenarios. Best score amongst different judge LLMs is in blue and worst score is in red. Best score between LLM-judge and Comb-eval for each judge model is in bold. Score for Metric-judge is 69.73% for GT and 68.06% for UQDB. Results are ordered (descending) by scores on LLM-judge with GT reference.

- Cohere Cmd-A (Cohere et al., 2025), R (Cohere, a), and R+ (Cohere, b)
- Grok-3 and Grok-3-mini (xAI)

Combo-Eval outperforms LLM-judge for most of the judge models considered. ***The results show an average improvement of 1.81% using Combo-Eval over LLM-judge when GT is used as the reference, and 2.37% when UQDB is used as the reference.***

When ground truth NLRs are unavailable, relying on the UQDB approach can be a viable alternative. Otherwise, GT scenario exhibits betters alignment with human assessment of NLRs compared to UQDB.

The results indicate that, ***in the UQDB scenario, the alignment with human assessment decreases by 7.27% for the LLM-judge method compared to using GT. For the Combo-Eval method, the alignment reduces by 6.71% under the same conditions.***

A.13 Evaluation Methods for Determining Accuracy Across Different NLR-Generation LLMs

In the paper, we conducted a human evaluation of NLR generations from different LLMs. We previously shared the agreement between human evaluation of LLM-generated NLRs and the NLR evaluation using three automated evaluation methods - Metric-judge, LLM-judge, and Combo-Eval. Now, we take a different route and compare the overall accuracy evaluation we obtain using Metric-judge, LLM-judge, and Combo-Eval for NLRs produced by the different generation LLMs, and compare that to the overall accuracy we obtain from human evaluation of NLRs generated by each of the different LLMs. This shows how well we can rank different LLMs for the task of generating NLR without human evaluation knowledge. For this exercise, we use GPT-4o as our judge model.

We run the evaluation methods against the test data to determine overall accuracy percentage rather than sample-by-sample alignment with human evaluation. In other words, we calculate accuracy solely based on these evaluation methods, without comparing how well they agree with human assessments for each specific sample.

The results from this exercise are shared in Table 9. Our findings reveal that all the evaluation methods rank the performance of generation LLMs similarly for the task of NLR generation in a way that matches the rankings from human evaluations. Therefore, to determine which model performs

Judge	Phind	L3.1 70	L3.1 405
Human	0.65	0.75	0.78
GT			
Metrics-Judge	0.79	0.80	0.81
LLM-Judge	0.72	0.8	0.82
Combo-Eval	0.73	0.80	0.81
UQDB			
Metrics-Judge	0.73	0.76	0.76
LLM-Judge	0.75	0.86	0.89
Combo-Eval	0.68	0.78	0.79

Table 9: Accuracy of NLRs generated across LLMs using Metrics as a judge, LLM as a judge, Combo-Eval, and human evaluation. Phind=Phind-CodeLlama-34B-v2; L3.170=Llama 3.1-70B-instruct; L3.1405=Llama 3.1-405B-instruct.

better at generating NLRs, metrics thresholding can be a practical alternative. Although there is a greater discrepancy between human and metrics-based judgments of individual NLRs, this approach can still effectively identify the general trend or ranking of LLMs for this task.

In an industry setting, this approach can aid in selecting the most suitable model for the task from the available options, thereby supporting critical decision-making and development.

A.14 Temperature Change for Judge LLMs

On a subset of judge LLMs, we ran evaluations using three temperature settings, 0.01, 0.5, and 1. Table 10 shows the F1 macro score (averaged F1 score representing class 0 and class 1 alignment with human evaluation) for LLM-as-a-judge evaluation method and the Combo-Eval method.

For the Llama-3.3-70B-Instruct model, we observe that lower temperature settings yield slightly better scores across both LLM-as-a-judge and Combo-eval methods under GT as well as UQDB scenarios. On the other hand, for GPT-4o, higher temperature settings result in marginally improved scores, with temperature 0.5 showing highest scores for GT scenario and temperature of 1.0 showing highest scores for UQDB scenario. The other judge models don't exhibit any clear trends. Also, the differences between scores across the various temperature settings across the judge models remain very minimal.

In conclusion, no clear trend emerges from this experiment; the evaluation does not significantly worsen or improve with an increase or decrease

Judge	Eval method	t0.01	t0.5	t1.0
GT				
L3.370	LLM-judge	**84.60**	84.25	83.39
L3.370	Combo-Eval	**83.90**	83.53	83.17
Grok-3	LLM-judge	82.43	**83.45**	82.58
Grok-3	Combo-Eval	**82.80**	**82.80**	82.43
GPT-4o	LLM-judge	80.72	**81.14**	80.89
GPT-4o	Combo-Eval	80.88	**82.00**	81.88
G4.1nano	LLM-judge	75.66	**75.88**	74.37
G4.1nano	Combo-Eval	**82.04**	81.69	79.61
UQDB				
L3.370	LLM-judge	**77.32**	76.24	76.18
L3.370	Combo-Eval	**79.40**	78.30	78.28
Grok-3	LLM-judge	**77.89**	77.08	77.71
Grok-3	Combo-Eval	78.99	78.85	**79.40**
GPT-4o	LLM-judge	75.82	76.54	**77.58**
GPT-4o	Combo-Eval	76.98	77.67	**78.25**
G4.1nano	LLM-judge	57.95	**58.17**	54.17
G4.1nano	Combo-Eval	62.14	**62.97**	59.12

Table 10: F1 macro score (averaged score from class 1 and class 0 F1) showing alignment of evaluation methods with human judgement across different temperature settings. L3.370=Llama 3.3-70B-instruct; G4.1nano=GPT4.1-nano

in temperature for the task of judging NLRs. The slight variations observed in the results across different temperature settings indicate varying trends for different judge LLMs, suggesting no consistent trend regarding temperature settings across all LLMs. The results imply that different judge models may react differently to temperature changes, but the impact is minimal. Combo-Eval remains superior to LLM-judge across most judge LLMs, and this trend persists across different temperature settings.