CONTINUAL EXPERIENCE LEARNING FROM EXPERT FEEDBACK FOR NL-TO-CYPHER DATA SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Natural-language-to-structured-query systems rely on paired natural-language requests and executable queries for training, prompting, and evaluation. Constructing such data manually is expensive, while LLM-based synthesis can introduce semantic errors that are not detected by syntax or execution alone. Moreover, corrections to individual examples are often discarded rather than reused to improve future synthesis. We study this problem primarily in the setting of natural-language-to-Cypher (NL-to-Cypher), where existing benchmarks remain relatively limited in semantic quality and structural coverage. We present CIRA, a continual experience-learning framework that converts expert corrections into reusable, scoped guidance for subsequent data synthesis. Instead of updating model parameters, CIRA compiles feedback into experiences that capture the failure, corrective guidance, and applicability conditions, then consolidates and selectively retrieves them during future generation. On an unseen CypherBench graph, CIRA generates NL-to-Cypher data with an estimated 95.0% validity based on a manually verified sample. Compared with 1,000 template-generated benchmark examples, CIRA produces 680 versus 503 distinct query structures and reduces structural duplication from 50% to 32%. In downstream fine-tuning, combining CIRA-generated and benchmark data yields the highest overall execution accuracy for both evaluated models. Across five rounds of expert feedback, CIRA increases the valid-pair rate from 0.29 to 0.92, while the same generation pipeline without experience memory ends at 0.21. On identical specifications, retrieving raw feedback reaches 0.67, showing that compiling and consolidating feedback provides additional benefit beyond retrieval alone. We further show that the learned experience bank transfers across a new schema, a different generator, and a new domain, improving the valid-pair rate by 0.21–0.55 over the corresponding memory-free pipeline. Finally, transferring eligible experiences from Cypher to SQL improves validity from 0.58 to 0.79, providing evidence that the learned guidance can extend beyond the query language on which it was acquired. The code is available at `https://anonymous.4open.science/r/CIRA-3814/`.

1 INTRODUCTION

Natural-language-to-structured-query systems rely on paired natural-language requests and executable queries for training, in-context learning, and evaluation Li et al. (2023); Lei et al. (2025). Constructing such data manually requires expertise in database schemas, query languages, and application domains Yu et al. (2018); Chauhan et al. (2025), while data synthesis offers a way to expand datasets across schemas and query capabilities Zhong et al. (2020b); Guo et al. (2025); Li et al. (2025); Jung et al. (2025). In this work, we focus primarily on natural-language-to-Cypher (NL-to-Cypher), where benchmarks are relatively recent and remain less mature than the NL-to-SQL ecosystem in both semantic quality and structural coverage. Our audit of two recent NL-to-Cypher benchmarks illustrates these limitations Chauhan et al. (2025); Feng et al. (2025). Among the audited pairs, 27.76% of Mind-the-Query examples (241/868) and 3.28% of CypherBench examples (30/916) contain incorrect reference queries or ambiguous questions. Coverage is also uneven: neither dataset contains Path Finding queries under our classification scheme, while Multi-step and Compound queries are absent from Mind-the-Query. CypherBench generates queries from pre-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

defined templates, limiting structural coverage to its template inventory. Appendix B details the classification and audit.

LLMs provide a flexible mechanism for synthesizing NL–query pairs conditioned on schemas and generation instructions Zhong et al. (2025); Guo et al. (2025); Li et al. (2025); Pourreza et al. (2025). However, syntactic validity and successful execution do not establish semantic equivalence between a question and its query Zhong et al. (2020a); Li et al. (2026); Wang et al. (2026); Zhu et al. (2025). Expert review can identify failures, but correcting individual examples does not prevent them from recurring. A correction that a query counts traversal paths instead of distinct reachable entities, for example, may provide useful guidance for future queries with similar counting semantics.

More broadly, this places NL–query synthesis in the setting of weak and programmatic supervision, where scalable labeling procedures trade manual annotation effort for potentially noisy supervision Ratner et al. (2017); Song et al. (2023); Zhang et al. (2025a). CIRA can be viewed as an adaptive data-generation function: rather than applying a fixed labeling heuristic or cleaning noisy labels after generation, it uses accumulated expert feedback to improve the procedure that produces future supervision.

The challenge is that such feedback is not universally reusable. Some corrections are schema-, domain-, query-language-, or query-class-specific, while others encode more general principles. Thus, effective reuse requires deciding what guidance to retain, when it applies, and how to consolidate redundant or conflicting corrections. Prior work on agent memory Zhao et al. (2024); Zhang et al. (2025b), feedback-driven refinement Madaan et al. (2023), and context optimization Yang et al. (2024a); Pryzant et al. (2023) provides mechanisms for adapting LLM behavior without updating model parameters. SIPDO further applies iterative prompt optimization to synthetic data generation Yu et al. (2026). However, instance-level feedback may transfer poorly, while overly broad guidance may fail under different schemas or query semantics Zhang et al. (2025b); Wu et al. (2025).

We introduce **CIRA—Clarify, Infer, Repair, Adapt**, a continual experience-learning framework that turns expert corrections into reusable guidance for future data synthesis. CIRA compiles semantic feedback into experiences containing a failure description, corrective guidance, and applicability conditions; assigns each experience an appropriate scope; consolidates accumulated guidance through add, revise, merge, and reject operations; and selectively retrieves applicable experiences during subsequent synthesis. The underlying LLM parameters remain fixed, so adaptation occurs through the evolving experience memory and the generation context constructed from it.

We evaluate CIRA first on its primary objective: producing high-quality, structurally diverse NL-to-Cypher data. On an unseen CypherBench graph, a frozen CIRA experience bank generates 1,000 pairs, of which 115/121 randomly audited examples are valid (95.0%). Compared with 1,000 template-generated benchmark examples, the CIRA data contains 680 versus 503 distinct query structures and reduces structural duplication from 50% to 32%. In downstream fine-tuning, CIRA-generated data complements the benchmark data, and combining the two gives the highest overall execution accuracy for both evaluated models.

We then examine how this capability emerges from expert feedback. Across five synthesis rounds, CIRA increases expert-validated pair rate from 29% to 92%, while feedback-writing time decreases from 56 to 22 seconds per generated pair. On the same final-round specifications, direct retrieval of raw feedback reaches 67% validity, compared with 92% for CIRA and 21% for the same generation pipeline without memory. The learned experiences also transfer without target-specific feedback to a new schema, a different generator, and a new domain, improving validity by 0.21–0.55 over the corresponding memory-free pipeline. Finally, transferring eligible experiences from Cypher to SQL increases validity from 58% to 79%, providing evidence that some learned guidance extends beyond the query language on which it was acquired.

Our contributions are:

- We identify semantic-quality and structural-coverage limitations in two NL-to-Cypher benchmarks, motivating synthesis methods that target correctness and coverage in addition to scale.
- We introduce CIRA, a continual experience-learning framework that compiles expert corrections into scoped, reusable experiences, consolidates accumulated guidance, and selectively retrieves it across synthesis rounds without updating LLM parameters.

- We show that CIRA generates high-quality and structurally diverse NL-to-Cypher data: on an unseen graph, 115/121 audited examples are valid, and the resulting 1,000-pair dataset contains 680 distinct query structures versus 503 in template-generated benchmark data.
- We show that continual experience reuse raises validity from 29% to 92% while reducing expert feedback effort, outperforms direct retrieval of raw corrections on identical synthesis requests, and transfers across schemas, generator models, domains, and from Cypher to SQL.

2 Related Work

Feedback-driven adaptation with frozen LLMs ranges from revising individual outputs to retaining reusable experience across tasks. Self-Refine iteratively critiques and revises an output Madaan et al. (2023), while CRITIC incorporates feedback from external tools Gou et al. (2024). Reflexion retains verbal reflections Shinn et al. (2023), ExpeL abstracts reusable insights Zhao et al. (2024), Voyager maintains reusable skills Wang et al. (2024), and ReasoningBank accumulates strategies from successful and failed trajectories Ouyang et al. (2026). Related approaches derive experiential guidance from candidate rollouts and feedback on subsequent queries Cai et al. (2025); Bang et al. (2026). Related work on prompt and context optimization similarly adapts future model behavior through evaluated feedback, including OPRO Yang et al. (2024a), ProTeGi Pryzant et al. (2023), GEPA Agrawal et al. (2025), DSPy Khattab et al. (2024), Promptbreeder Fernando et al. (2024), and ACE Zhang et al. (2025b), while RAG and Dynamic Cheatsheet provide related mechanisms for retrieval and persistent reusable guidance Lewis et al. (2020); Suzgun et al. (2026). CIRA builds on these ideas for structured data synthesis, where reusable guidance must remain valid across different schemas, query structures, and generation requests. It represents each experience together with its applicability conditions, consolidates accumulated guidance, and retrieves only experiences compatible with the current synthesis specification.

Existing NL-to-Cypher datasets use a range of synthesis and validation procedures. Neo4j Text2Cypher consolidates existing resources through manual cleaning and `EXPLAIN`-based syntax validation Ozsoy et al. (2025). CypherBench combines template instantiation, LLM-based question rewriting, and manual inspection Feng et al. (2025), while Mind-the-Query uses LLM generation, automated checks, manual review, and iterative prompt refinement Chauhan et al. (2025). SyntheT2C combines prompting and template filling with automated and human validation Zhong et al. (2025), and KG2Cypher constructs executable queries from graph facts before generating NL questions Choi et al. (2026). Cazzaro et al. (2026) similarly study graph-grounded synthesis for text-to-Cypher fine-tuning. These methods establish validation and refinement as important parts of dataset construction, but syntactic validity and successful execution do not by themselves establish semantic alignment between a question and its query. CIRA instead studies whether semantic feedback from reviewed pairs can be compiled and reused to improve later synthesis across targeted query classes.

NL-to-SQL provides a more mature setting for synthetic data generation and semantic evaluation. Prior work addresses schema-grounded and grammar-based synthesis Zhong et al. (2020b); Wu et al. (2021), domain-specific question generation (Duan et al., 2025), retrieval-based linguistic diversification Awasthi et al. (2022), and synthetic data construction for model training Yang et al. (2024b); Guo et al. (2025); Li et al. (2025); Caferoğlu et al. (2025). Semantic evaluation includes test-suite-based approximations of query equivalence Zhong et al. (2020a) and robustness to meaning-preserving linguistic variation Gan et al. (2021). In our work, NL-to-SQL serves as a transfer setting to test whether experiences acquired from NL-to-Cypher feedback remain useful in another query language without updating LLM parameters.

3 The CIRA Architecture

CIRA synthesizes NL–Cypher pairs while accumulating reusable guidance from expert feedback (Figure 1). Each synthesis round has two paths. The *generation path* produces question–query pairs from predefined synthesis specifications. The *feedback loop* converts expert corrections into structured experiences, consolidates them in a persistent memory, and retrieves relevant guidance for later rounds. The underlying LLM parameters remain fixed; only the experience memory evolves across rounds.

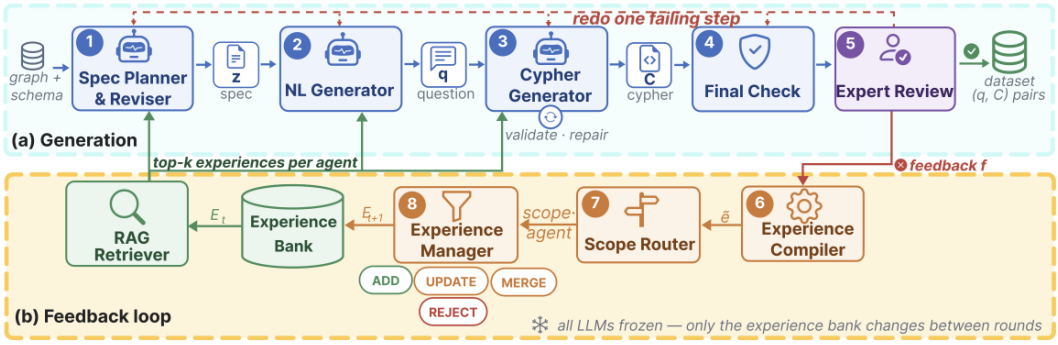

Figure 1: CIRA architecture. The generation path produces NL–Cypher pairs for expert review; the feedback loop converts corrections into reusable experiences, guiding subsequent synthesis rounds.

Let $\mathcal{S} = (\Lambda, \Gamma, P)$ denote a property-graph schema with node labels Λ, relationship types Γ, and properties P. Each synthesis request is defined by a specification $z = (c, \delta, L, R, \rho, \Phi, o)$, which identifies the target query class c, difficulty δ, required schema elements (L, R), reasoning operation ρ, property constraints Φ, and expected output o. CIRA aims to generate an NL question q and Cypher query C that express the semantics specified by z. Full notation and the formal round-to-round learning formulation are given in Appendix A.

3.1 Generation Path

Planning. A Coverage Controller generates synthesis specifications across predefined query-class and difficulty combinations. Retrieved property values ground constraints in the graph (Talaei et al., 2024; Pourreza et al., 2025), while coverage hints encourage use of schema elements not yet represented in the current batch. Each proposed specification is checked deterministically to ensure that its labels, relationships, and constrained property values are compatible with the graph and learned lessons. Invalid proposals are regenerated before entering the generation pipeline. The complete specification validation procedure is given in Appendix A.1.

Question and query generation. For each valid specification, the NL Generator receives a relevant *schema slice*, containing the specified labels and relationships and their local neighborhood, together with up to two grounding subgraphs and relevant experiences. It produces a question that expresses the requested query class, constraints, reasoning operation, and output. The Cypher Generator then receives the question, specification, schema slice, grounding evidence, and any retrieved experiences. The specification helps disambiguate the intended semantics but does not override the NL question. Both generators retrieve from the same experience memory, but each uses its own retrieval query and only receives experiences applicable to its role. Retrieved experiences are rendered as short trigger–guideline pairs describing when a lesson applies and what corrective action to take.

Validation and repair. Generated queries pass through a bounded validate-and-repair loop following the general query-repair strategy used in prior work (Pourreza et al., 2025; Shinn et al., 2023). Deterministic checks verify that the query parses, is consistent with the schema, and executes successfully against the graph. Failed checks are returned to the Cypher Generator for repair, with at most three generation attempts. Empty results are handled separately and trigger regeneration of the question followed by another repair cycle. Execution success alone does not establish semantic correctness. Thus, CIRA performs a final pair-level check that asks whether the question and query express the intended specification and whether the query preserves the meaning of the question. If a problem is detected, the system identifies the earliest faulty module and re-runs the pipeline once from that point with targeted corrective feedback.

CIRA additionally performs a *specification round trip* to detect semantic drift. The final question and query are independently used to reconstruct the intended specification, and the pair is retained only when the two reconstructions are judged semantically equivalent. Retained pairs proceed to expert review together with their specifications, validation evidence, and the identifiers of experiences used during generation. Exact validator definitions, repair rules, and the round-trip procedure are provided in Appendix A.1.

3.2 Learning from Expert Feedback

Each generated pair is reviewed together with its specification, grounding evidence, and query results. The expert records whether the pair is valid, separately evaluates the NL question and Cypher query, and provides free-text feedback describing any identified problem and its expected correction. Valid pairs create no new experience; invalid pairs with written feedback enter the experience-learning pipeline.

Rather than storing the correction verbatim, CIRA compiles it into a reusable experience. Conceptually, an experience records three central elements: $e = (\text{failure}, \text{applicability}, \text{guidance})$, together with metadata describing where the guidance should be used. For example, feedback that a query counts traversal paths rather than distinct reachable entities can be generalized into guidance to deduplicate endpoints when multiple paths reach the same entity.

A **Scope Router** determines how broadly an experience should apply. CIRA distinguishes five levels of scope: *general*, *structured-query*, *language*, *domain*, and *schema*. Language-, domain-, and schema-specific experiences are anchored to their source context, while broader experiences may transfer across contexts. When applicability is uncertain, the router favors the narrower scope. It also identifies which generation module could have prevented the observed failure.

The **Experience Manager** then compares the candidate with related experiences already in memory. It can *add* a new lesson, *update* an incomplete existing experience, *merge* compatible experiences, or *reject* unsupported, redundant, or conflicting guidance. Updates and merges require compatible scope and equivalent lessons rather than topical similarity alone. The experience bank retains provenance and version history linking its contents back to the expert-reviewed pairs from which they were derived. Appendix A.3 gives the complete experience representation, scope definitions, and memory-update algorithm.

3.3 Experience Retrieval

During later synthesis rounds, CIRA retrieves experiences that are applicable to the current request and target module. Retrieval first filters the active memory according to module and scope. Schema-, domain-, and language-specific experiences are restricted to matching contexts, while general and structured-query experiences can transfer more broadly.

Eligible experiences are then ranked primarily by semantic similarity to the current synthesis request. CIRA also records how often each experience has been used and how often its use is associated with a valid pair. This history provides a utility signal favoring experiences that have repeatedly accompanied successful generations. Because several experiences may be retrieved for the same pair, this utility represents association with successful outcomes rather than causal contribution. At most five experiences are supplied to a generation module. Thus, the experience memory may grow across synthesis rounds without causing the learned guidance included in each prompt to grow proportionally. Appendix A.4 gives the exact eligibility conditions, utility definition, retrieval score, and module-specific retrieval queries.

4 Experiments

We evaluate CIRA around four questions. First, we ask whether a learned experience bank can produce high-quality and diverse NL–Cypher data, which is the primary goal of the framework. We then examine how synthesis improves as expert feedback accumulates, whether compiling feedback into scoped experiences provides benefits beyond retaining raw corrections, and whether the learned experiences remain useful in new settings.

4.1 Experimental Setup

Data. The primary learning study uses a Neo4j database constructed from Earth System Grid Federation (ESGF) climate-science data Cinquini et al. (2014), containing 49 node labels, 65 relationship types, 373.7K nodes, and 2.2M relationships. Transfer experiments use a second climate graph derived from NASA's Global Change Master Directory (GCMD) NASA, the biomedical Hetionet graph Himmelstein et al. (2017), and the `formula_1` relational database from BIRD Li et al.

Table 1: Dataset-level comparison of two 1,000-pair sets on the CypherBench `soccer` graph.

	CB-train	$\mathcal{D}_{\mathrm{CIRA}}$
Pairs	1000	1000
Distinct query structures	503	**680**
Structural duplicate rate	50%	**32%**
Distinct question bigrams (type/token)	0.17	**0.23**
Most common 3-word opening	55%	**14%**
Labels / rel. types covered	100% / 100%	100% / 100%

(2023). Dataset-quality and downstream experiments use the `soccer` and `politics` graphs from CypherBench Feng et al. (2025).

Experimental arms. We compare CIRA against three conditions. *Naive* generates an NL–query pair in a single LLM call using the specification and schema, without repair or memory. *Static* uses the complete generation pipeline but does not update experience memory. *Raw-feedback RAG* stores expert comments verbatim and retrieves them using CIRA's dense retriever, but without experience compilation, scope assignment, or memory consolidation. Unless otherwise noted, all conditions use frozen Gemini 3.7 Flash (Gemini Team, Google, 2023); one transfer experiment replaces the generator with GPT (OpenAI, 2025).

Protocol. In the learning study, all methods receive the same synthesis specifications. Generated pairs are pooled, shuffled, and reviewed by experts blinded to their source. Reviewers assess semantic validity and question quality before providing written feedback; feedback from round t becomes available only in round $t+1$. We report valid-pair rate, expert time per valid pair, and, where relevant, structural and lexical diversity. Between-method comparisons are paired by synthesis specification. Appendices A.2 and D.3 give the annotation procedure and statistical tests. Implementation settings are in Appendix A.

4.2 RQ1: Can CIRA Produce High-Quality and Diverse NL-to-Cypher Data?

We first evaluate the end goal of CIRA: using a learned experience bank to construct a new NL–Cypher dataset. After the learning and transfer stages, we freeze the bank, disable expert feedback, and generate 1,000 pairs on the previously unseen `soccer` graph from CypherBench. We compare this dataset, $\mathcal{D}_{\mathrm{CIRA}}$, with 1,000 questions sampled from the benchmark's template-generated training data (CB-train). Human experts reviewed a random sample of 121 CIRA pairs, of which 115 were valid, corresponding to a validity rate of 95.0%. The 95% Wilson confidence interval is 89.6%–97.7%. Both datasets cover every node label and relationship type of the graph, but $\mathcal{D}_{\mathrm{CIRA}}$ exhibits substantially greater structural and lexical diversity. Its 1,000 queries contain 680 distinct canonical query structures, compared with 503 for CB-train, reducing the structural duplicate rate from 50% to 32%. The most common three-word question opening accounts for 14% of CIRA questions compared with 55% of CB-train questions.

The structural difference is not limited to duplicate counts. $\mathcal{D}_{\mathrm{CIRA}}$ contains variable-length paths in 10% of its queries and `shortestPath` in 5%, while neither occurs in CB-train. Conversely, CB-train uses some constructs, such as `WITH` and subqueries, more frequently. Thus, the CIRA dataset expands structural coverage rather than simply reproducing the benchmark distribution.

Beyond diversity, we ask whether the generated supervision is useful for model training. This downstream evaluation tests whether CIRA's higher-quality, broader-coverage data translates into gains beyond the synthesis metrics. Following the CypherBench protocol, we fine-tune Llama-3.1-8B (Dubey et al., 2024) and Qwen2.5-Coder-1.5B (Hui et al., 2024) on CB-train, $\mathcal{D}_{\mathrm{CIRA}}$, or their union and evaluate on the unseen `politics` graph. CIRA-generated data is complementary rather than a replacement for the benchmark distribution: CB-train performs best on the benchmark's own test questions, while CIRA training performs better on the broader CIRA-generated test set. Combining the two training sets gives the highest overall execution accuracy for both models.

The complementarity is most visible on query classes weakly represented by the benchmark templates. On CIRA-politics, the CB-trained Llama model answers no Pathfinding questions correctly $(0/23)$ and no Complex Aggregation questions correctly $(0/28)$, whereas training on $\mathcal{D}_{\mathrm{CIRA}}$ raises

Table 2: Execution accuracy (%) on the unseen `politics` graph after fine-tuning on CB-train, $\mathcal{D}_{\text{CIRA}}$, or their union.

Training data	CB-politics	CIRA-politics	Overall
Llama-3.1-8B			
CB-train	**47.5**	28.5	38.0
$\mathcal{D}_{\text{CIRA}}$	34.5	42.0	38.2
CB-train + $\mathcal{D}_{\text{CIRA}}$	37.0	**48.0**	**42.5**
Qwen2.5-Coder-1.5B			
CB-train	**30.0**	24.0	27.0
$\mathcal{D}_{\text{CIRA}}$	18.0	**33.0**	25.5
CB-train + $\mathcal{D}_{\text{CIRA}}$	29.5	31.0	**30.2**

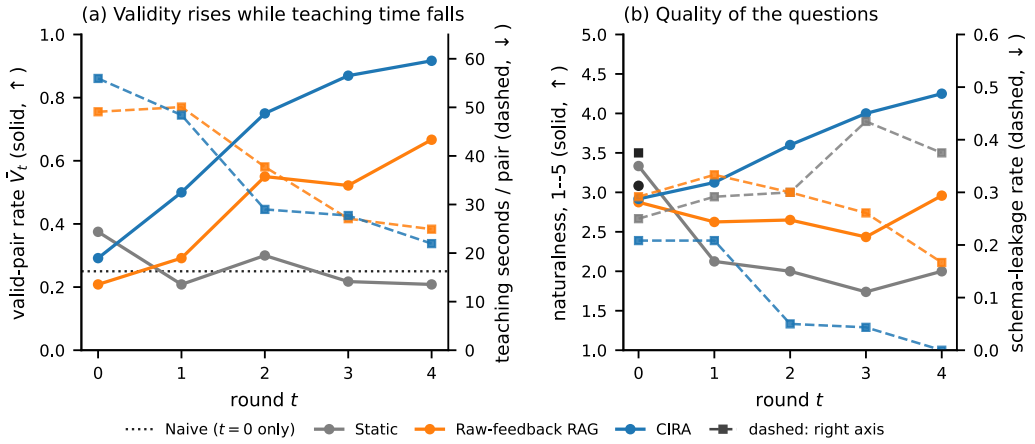

Figure 2: Learning from expert feedback on ESGF. Valid-pair rate increases as CIRA accumulates experiences, while expert feedback effort decreases. Static uses the same generation pipeline without experience memory.

these to $15/23$ and $11/28$, respectively. Appendix C provides additional dataset statistics, validity-audit details, fine-tuning settings, and downstream accuracy and executability results across training sources and sizes.

4.3 RQ2: How Does CIRA Improve Through Expert Feedback?

We next examine how the experience bank is acquired. Starting with an empty memory, CIRA undergoes five rounds of synthesis and expert review on ESGF. Each round uses the same query-class and difficulty schedule across methods, and feedback collected in one round can affect only subsequent rounds.

CIRA's valid-pair rate increases from 0.29 in the first round to 0.92 in the fifth, whereas Static ends at 0.21. The expert burden decreases at the same time: feedback-writing time falls from 56 s to 22 s per generated pair. By the final round, only two CIRA pairs are rejected and no Cypher query is judged semantically incorrect. Total expert time per valid pair is 66 s for CIRA compared with 252 s for Static. The improvement is also broad across query classes. By the final round, CIRA produces at least two valid pairs in every class and all three valid pairs in six of eight classes. Pathfinding, for which no arm produces a valid pair at $t = 0$, reaches three valid pairs at $t = 3$ and two at $t = 4$; Complex Aggregation improves from no valid pairs in rounds 0–1 to all three valid pairs at $t = 4$. Raw-feedback RAG improves several aggregation classes but produces only one valid Pathfinding pair in each of rounds 3 and 4.

Question quality improves alongside query validity. Mean naturalness for CIRA questions rises from 2.9 to 4.2 across the five rounds, while schema leakage falls from 21% to 0%. Static moves in

Table 3: Effect of experience memory on the same 24 ESGF specifications.

Method	Active memory	Valid pairs	Valid-pair rate
Static	0	5/24	0.21
Raw-feedback RAG	58	16/24	0.67
CIRA	32	**22/24**	**0.92**

Table 4: Transfer of accumulated experience. Δ is the difference in valid-pair rate between CIRA and the same generation pipeline without memory.

Target	Static	CIRA	Δ
GCMD: new schema	0.36	**0.92**	+0.55
ESGF: new generator	0.33	**0.88**	+0.54
Hetionet: new domain	0.71	**0.92**	+0.21
`formula_1`: NL-to-SQL	0.58	**0.79**	+0.21

the opposite direction, from 3.3 to 2.0 in naturalness and from 25% to 38% schema leakage. Thus, the increase in valid pairs is not obtained by sacrificing the quality of the generated questions.

Appendices D.1–D.3 report the complete class-by-round trajectories, failure attribution, grading and feedback-time decomposition, memory growth, and paired statistical tests. The paired CIRA–Static gap is significant beginning with the first round in which memory is available and grows to $+0.71$ by the final round.

4.4 RQ3: Does Compiling Feedback Matter?

Improvement across rounds does not establish whether structured experience compilation is necessary: simply retaining previous expert comments might provide similar benefits. Thus, we compare CIRA with Raw-feedback RAG, which uses the same expert-review protocol but stores and retrieves them verbatim. On the same 24 round-4 synthesis specifications, Static produces 5 valid pairs (0.21), Raw-feedback RAG produces 16 (0.67), and CIRA produces 22 (0.92), as shown in Table 3.

The paired comparison reinforces this result. CIRA succeeds on 17 specifications for which Static fails, with no reverse cases ($\Delta = +0.71$, $p < 0.001$), and produces six more valid pairs than Raw-feedback RAG on the same specifications. Thus, retaining feedback is already useful, but compiling and consolidating it provides a further gain beyond retrieval alone.

The experience bank also remains compact. The 51 expert corrections in the learning study produce 46 experience proposals; the manager retains 32 active experiences, rejects 13 redundant or unsupported proposals, and retires one through merging. Of the 32 active experiences, 22 have broad scope (general or structured-query), eight are Cypher-specific, and only two are tied to ESGF. This composition provides a natural basis for the transfer experiments below: most learned guidance is not specific to the source graph.

Appendix D.4 reports the manager's update and merge decisions and the utility statistics; Table 13 gives the full paired analysis.

4.5 RQ4: Do Learned Experiences Transfer?

Finally, we examine whether experience learned during NL–Cypher synthesis remains useful beyond the setting in which it was acquired. We carry the experience bank forward without resetting it through a sequence of increasingly different targets: a new schema in the same climate domain (GCMD), a different generator LLM, a biomedical graph (Hetionet), and finally NL-to-SQL synthesis on BIRD's `formula_1` database. CIRA is evaluated on each target before receiving targeted feedback, so the reported validity reflects the guidance available when that target is encountered.

The bank remains useful as the schema, generator, and domain change. The gains are largest when the memory-free pipeline is weakest: CIRA improves validity by 0.55 on GCMD and 0.54 after replacing the generator. On GCMD, CIRA reaches 0.92 validity in its first round, the level reached

on ESGF only after several rounds of feedback. With the new generator, CIRA is valid on all eight hard specifications compared with three of eight for Static, while schema leakage drops from 12 questions to 4. Transfer is also visible on difficult query classes. On GCMD, Pathfinding improves from no valid pairs under Static to two of three with CIRA, while Simple Retrieval and Compound improve from none to all three. On Hetionet, where Static already reaches 0.71 overall, CIRA produces all three valid Pathfinding pairs compared with one under Static.

Transfer to NL-to-SQL provides a more demanding test because the query language itself changes. Only experiences whose scope permits cross-language reuse are made available. CIRA increases validity from 0.58 to 0.79, reduces schema leakage from six questions to three, and reduces grading time from 71 to 38 seconds per valid pair. The paired comparison contains five CIRA-only successes and no Static-only successes, but remains just short of the conventional significance threshold ($p = 0.063$). We treat this experiment as suggestive evidence of cross-language transfer rather than a definitive result, and leave a broader study of transfer across structured query languages to future work.

Appendix D.5 describes the sequential transfer protocol, the paired tests on GCMD and Hetionet, the experiences added on GCMD and with the new generator, and the adaptation of the pipeline to NL-to-SQL. GCMD and the new-generator stage contribute three and four retained experiences, respectively.

5 Conclusion

We introduced CIRA to improve natural-language-to-structured-query data synthesis by reusing what experts teach during review. Focusing primarily on NL-to-Cypher, CIRA converts corrections into scoped experiences, consolidates accumulated guidance, and retrieves only experiences relevant to subsequent synthesis requests. Across five feedback rounds, valid-pair rate rises from 29% to 92%, while feedback-writing time decreases from 56 to 22 seconds per generated pair. On identical specifications, CIRA reaches 92% validity compared with 67% for retrieval of raw feedback and 21% for the same generation pipeline without memory. The learned guidance also remains useful beyond its original setting. When transferred to new schemas, a different generator, and a new domain, CIRA improves validity over the corresponding memory-free pipeline, while cross-language transfer from Cypher to SQL provides additional evidence that some experiences capture principles beyond a particular query language. With the experience bank frozen, CIRA generates a dataset on an unseen graph, with an estimated 95.0% validity based on a manually verified sample and substantially more distinct query structures than template-generated benchmark data. Downstream experiments show that CIRA-generated and template-generated data are complementary: their combination yields the highest overall execution accuracy for both evaluated models.

More broadly, CIRA can be viewed as an adaptive data-generation procedure for scalable supervision: instead of treating noise in automatically generated data only after it is produced, expert feedback improves the procedure that generates future examples. This suggests a complementary direction to weak-supervision and noisy-label methods, reducing recurring errors during data generation itself.

These results suggest that the value of expert feedback lies not only in correcting the example that elicited it, but in extracting guidance that can be reused selectively across future synthesis requests.

Limitations. The main limitation is the cost of human feedback, which constrains the size of the learning study. Consequently, the feedback rounds are relatively small, limiting the precision of absolute validity estimates. A natural next step is to test whether an LLM-based critic can provide useful feedback before or in place of human review, and how closely the resulting experiences match those learned from experts. The downstream study uses 1,000 training pairs per source and one run per configuration, which is sufficient to demonstrate the complementarity of CIRA-generated and template-based data. Future work can extend this analysis to larger generated datasets, multiple runs, and additional models. Because a frozen experience bank can generate additional data without further expert feedback, scaling this evaluation is straightforward.

AI USE STATEMENT

Large language models were used to assist with manuscript editing and code development. As components of CIRA, they were also used to synthesize natural-language–query pairs and transform expert feedback into reusable experiences, as described in the methodology. All scientific contributions, experimental designs, analyses, and conclusions were developed and verified by the authors.

ETHICS STATEMENT

This work uses publicly available NL-to-Cypher datasets and expert feedback to study continual data synthesis. Expert judgments are used to evaluate generated pairs and construct reusable guidance. The proposed framework may inherit biases or inaccuracies from its underlying language models and source datasets. We mitigate these risks through semantic validation, expert review, and explicit applicability conditions for retrieved experiences.

REPRODUCIBILITY STATEMENT

We provide the framework architecture, algorithms, experimental protocols, annotation procedures, and statistical analyses in the main paper and appendix. Source code and additional implementation details are available in the GitHub repository linked in the abstract.

REFERENCES

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning, 2025. URL https://arxiv.org/abs/2507.19457.

Abhijeet Awasthi, Ashutosh Sathe, and Sunita Sarawagi. Diverse parallel data synthesis for cross-database adaptation of text-to-SQL parsers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11548–11562. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.emnlp-main.794. URL https://aclanthology.org/2022.emnlp-main.794/.

Jihwan Bang, Seunghan Yang, Kyuhong Shim, Simyung Chang, Juntae Lee, and Sungha Choi. Feedback Adaptation for Retrieval-Augmented Generation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 28472–28485, 2026. doi: 10.18653/v1/2026.findings-acl.1419. URL https://aclanthology.org/2026.findings-acl.1419/.

Hasan Alp Caferoğlu, Mehmet Serhat Çelik, and Özgür Ulusoy. SING-SQL: A synthetic data generation framework for in-domain text-to-SQL translation, 2025. URL https://arxiv.org/abs/2509.25672.

Yuzheng Cai, Siqi Cai, Yuchen Shi, Zihan Xu, Lichao Chen, Yulei Qin, Xiaoyu Tan, Gang Li, Zongyi Li, Haojia Lin, Yong Mao, Ke Li, and Xing Sun. Training-free group relative policy optimization, 2025. URL https://arxiv.org/abs/2510.08191.

Francesco Cazzaro, Jessica Lennon, and Ariadna Quattoni. Achieving precise text-to-cypher via grounded knowledge graph data generation, 2026. URL https://arxiv.org/abs/2606.14325.

Vashu Chauhan, Shobhit Raj, Shashank Mujumdar, Avirup Saha, and Anannay Jain. Mind the query: A benchmark dataset towards Text2Cypher task. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 1890–1905. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-industry.133. URL https://aclanthology.org/2025.emnlp-industry.133/.

Minjun Choi, Yerin Kim, Junghyuk Seo, Sujin Mo, Hyemin Lee, and Youngjoong Ko. KG2Cypher: Data-Centric Pipeline for Building Enterprise Text-to-Cypher Systems. *arXiv preprint arXiv:2606.27742*, 2026. URL https://arxiv.org/abs/2606.27742.

Luca Cinquini, Daniel Crichton, Chris Mattmann, John Harney, Galen Shipman, Feiyi Wang, et al. The earth system grid federation: An open infrastructure for access to distributed geospatial data. *Future Generation Computer Systems*, 36:400–417, 2014. doi: 10.1016/j.future.2013.07.002. URL https://doi.org/10.1016/j.future.2013.07.002.

Shaoming Duan, Youxuan Wu, Chuanyi Liu, Yuhao Zhang, Zirui Wang, Peiyi Han, Shengyuan Yu, Liang Yan, and Yingwei Liang. DSQG-syn: Synthesizing high-quality data for text-to-SQL parsing by domain specific question generation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 2971–2989. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-naacl.162. URL https://aclanthology.org/2025.findings-naacl.162/.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Yanlin Feng, Simone Papicchio, and Sajjadur Rahman. CypherBench: Towards precise retrieval over full-scale modern knowledge graphs in the LLM era. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8934–8958. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.438. URL https://aclanthology.org/2025.acl-long.438/.

Chrisantha Fernando, Dylan Sunil Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-Referential Self-Improvement via Prompt Evolution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 13481–13544. PMLR, 2024. URL https://proceedings.mlr.press/v235/fernando24a.html.

Yujian Gan, Xinyun Chen, Qiuping Huang, Matthew Purver, John R. Woodward, Jinxia Xie, and Pengsheng Huang. Towards robustness of text-to-SQL models against synonym substitution. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 2505–2515, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.195. URL https://aclanthology.org/2021.acl-long.195/.

Gemini Team, Google. Gemini: A family of highly capable multimodal models, 2023. URL https://arxiv.org/abs/2312.11805.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2305.11738.

Yu Guo, Dong Jin, Shenghao Ye, Shuangwu Chen, Jian Yang, and Xiaobin Tan. SQLForge: Synthesizing reliable and diverse data to enhance text-to-SQL reasoning in LLMs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8441–8452. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.443. URL https://aclanthology.org/2025.findings-acl.443/.

Daniel Scott Himmelstein, Antoine Lizee, Christine Hessler, Leo Brueggeman, Sabrina L. Chen, Dexter Hadley, Ari Green, Pouya Khankhanian, and Sergio E. Baranzini. Systematic integration of biomedical knowledge prioritizes drugs for repurposing. *eLife*, 6:e26726, 2017. doi: 10.7554/eLife.26726. URL https://doi.org/10.7554/eLife.26726.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, Kai Dang, Yang Fan, Yichang Zhang, An Yang, Rui Men, Fei Huang, Bo Zheng, Yibo Miao, Shanghaoran Quan, Yunlong Feng, Xingzhang Ren, Xuancheng Ren, Jingren Zhou, and Junyang Lin. Qwen2.5-Coder technical report, 2024. URL https://arxiv.org/abs/2409.12186.

Jaehun Jung, Seungju Han, Ximing Lu, Skyler Hallinan, David Acuna, Shrimai Prabhumoye, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, and Yejin Choi. Prismatic synthesis: Gradient-based data diversification boosts generalization in llm reasoning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan A, Saiful Haq, Ashutosh Sharma, Thomas Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling Declarative Language Model Calls into State-of-the-Art Pipelines. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/f1cf02ce09757f57c3b93c0db83181e0-Abstract-Conference.html.

Fangyu Lei, Jixuan Chen, Yuxiao Ye, Ruisheng Cao, Dongchan Shin, Hongjin Su, Zhaoqing Suo, Hongcheng Gao, Wenjing Hu, Pengcheng Yin, Victor Zhong, Caiming Xiong, Ruoxi Sun, Qian Liu, Sida Wang, and Tao Yu. Spider 2.0: Evaluating language models on real-world enterprise text-to-SQL workflows. In *International Conference on Learning Representations*, 2025.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Boyan Li, Ou Ocean Kun Hei, Yue Yu, and Yuyu Luo. DPC: Training-free text-to-SQL candidate selection via dual-paradigm consistency. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6897–6913, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.313. URL https://aclanthology.org/2026.acl-long.313/.

Haoyang Li, Shang Wu, Xiaokang Zhang, Xinmei Huang, Jing Zhang, Fuxin Jiang, Shuai Wang, Tieying Zhang, Jianjun Chen, Rui Shi, Hong Chen, and Cuiping Li. OmniSQL: Synthesizing high-quality text-to-SQL data at scale. *Proceedings of the VLDB Endowment*, 18(11):4695–4709, 2025. doi: 10.14778/3749646.3749723. URL https://doi.org/10.14778/3749646.3749723.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin Chang, Fei Huang, Reynold Cheng, and Yongbin Li. 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, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2303.17651.

NASA. Global change master directory (gcmd). National Aeronautics and Space Administration. URL https://gcmd.earthdata.nasa.gov/. Accessed: 2026-09-24.

OpenAI. GPT-5 system card, 2025. URL https://openai.com/index/gpt-5-system-card/.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling Agent Self-Evolving with Reasoning Memory. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.25140.

Makbule Gulcin Ozsoy, Leila Messallem, Jon Besga, and Gianandrea Minneci. Text2Cypher: Bridging Natural Language and Graph Databases. In *Proceedings of the Workshop on Generative AI and Knowledge Graphs (GenAIK)*, pp. 100–108. International Committee on Computational Linguistics, 2025. URL https://aclanthology.org/2025.genaik-1.11/.

Mohammadreza Pourreza, Hailong Li, Ruoxi Sun, Yeounoh Chung, Shayan Talaei, Gaurav Tarlok Kakkar, Yu Gan, Amin Saberi, Fatma Ozcan, and Sercan Arik. CHASE-SQL: Multi-path reasoning and preference optimized candidate selection in text-to-SQL. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/974ff7b5bf08dbf9400b5d599a39c77f-Abstract-Conference.html.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. Automatic Prompt Optimization with "Gradient Descent" and Beam Search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7957–7968, 2023. doi: 10.18653/v1/2023.emnlp-main.494. URL https://aclanthology.org/2023.emnlp-main.494/.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment*, 11(3):269–282, 2017.

Mohammadtaher Safarzadeh, Afshin Oroojlooyjadid, and Dan Roth. Evaluating NL2SQL via SQL2NL. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 18954–18968. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.1031. URL https://aclanthology.org/2025.findings-emnlp.1031/.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2303.11366.

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11):8135–8153, 2023. doi: 10.1109/TNNLS.2022.3152527.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic Cheatsheet: Test-Time Learning with Adaptive Memory. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7080–7106, 2026. doi: 10.18653/v1/2026.eacl-long.333. URL https://aclanthology.org/2026.eacl-long.333/.

Shayan Talaei, Mohammadreza Pourreza, Yu-Chen Chang, Azalia Mirhoseini, and Amin Saberi. CHESS: Contextual harnessing for efficient SQL synthesis, 2024. URL https://arxiv.org/abs/2405.16755.

Dongyu Wang, Jingyu Li, Lan Zhang, Ganggang Yu, and Liang Huang. VET: Verifiable execution tracing for reliable text-to-SQL generation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 30867–30883, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.findings-acl.1544. URL https://aclanthology.org/2026.findings-acl.1544/.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An Open-Ended Embodied Agent with Large Language Models. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=ehfRiF0R3a.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/d813d324dbf0598bbdc9c8e79740ed01-Abstract-Conference.html.

Kun Wu, Lijie Wang, Zhenghua Li, Ao Zhang, Xinyan Xiao, Hua Wu, Min Zhang, and Haifeng Wang. Data augmentation with hierarchical SQL-to-question generation for cross-domain text-to-SQL parsing. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 8974–8983, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.707. URL https://aclanthology.org/2021.emnlp-main.707/.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large Language Models as Optimizers. In *International Conference on Learning Representations*, 2024a. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/3339f19c5fcee3ad74502947a32be9e6-Abstract-Conference.html.

Jiaxi Yang, Binyuan Hui, Min Yang, Jian Yang, Junyang Lin, and Chang Zhou. 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)*, pp. 7864–7875. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.acl-long.425. URL https://aclanthology.org/2024.acl-long.425/.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 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. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1425.

Yaoning Yu, Ye Yu, Peiyan Zhang, Kai Wei, Haojing Luo, and Haohan Wang. SIPDO: Closed-loop prompt optimization via synthetic data feedback. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/e6da278cdd692077b7e4a99d55573d9c-Abstract-Conference.html.

Qi Zhang, Huitong Pan, Zhijia Chen, Longin Jan Latecki, Cornelia Caragea, and Eduard Dragut. DynClean: Training dynamics-based label cleaning for distantly-supervised named entity recognition. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 2540–2556, 2025a.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models, 2025b. URL https://arxiv.org/abs/2510.04618.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936.

Ruiqi Zhong, Tao Yu, and Dan Klein. Semantic evaluation for text-to-SQL with distilled test suites. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 396–411, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.29. URL https://aclanthology.org/2020.emnlp-main.29/.

Victor Zhong, Mike Lewis, Sida I. Wang, and Luke Zettlemoyer. Grounded adaptation for zero-shot executable semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6869–6882. Association for Computational Linguistics, 2020b. doi: 10.18653/v1/2020.emnlp-main.558. URL https://aclanthology.org/2020.emnlp-main.558/.

Zijie Zhong, Linqing Zhong, Zhaoze Sun, Qingyun Jin, Zengchang Qin, and Xiaofan Zhang. SyntheT2C: Generating synthetic data for fine-tuning large language models on the Text2Cypher task. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 672–692. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.coling-main.46/.

Yuchang Zhu, Huizhe Zhang, Bingzhe Wu, Jintang Li, Zibin Zheng, Peilin Zhao, Liang Chen, and Yatao Bian. Measuring diversity in synthetic datasets. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 80373–80397. PMLR, 2025.