Beyond Dense Retrieval: A Structured-Semantic Approach for Conversation Analytics

Abinesh Kanagarajan
abinesk@amazon.com
Amazon

Chetan Aggarwal
caggar@amazon.com
Amazon

Anjali Jain
anjjain@amazon.com
Amazon

Abstract

Large Language Model (LLM) based conversational systems generate vast interaction logs containing critical diagnostic signals such as failure modes and safety violations. However, the cross-turn dependencies, implicit state, and topic drift inherent to these conversations (Yang et al., 2025b; Mo et al., 2024) defeat dense retrieval, whose lone embedding cannot concurrently encode discrete structured constraints (e.g., refusal type) and open-ended semantic content (Lee et al., 2025). Our central claim is conceptual: conversational retrieval must span two complementary information planes, structured metadata and semantic content, and these are best reconciled by a unified hybrid representation that jointly supports filtering over structured metadata fields and vector similarity search over unbounded open-ended fields. We instantiate this as the Extract-Annotate-Query (EAQ) framework, which derives a metadata taxonomy, applies LLM-driven annotation, and compiles natural-language queries into hybrid graph operations. On the CDR-Benchmark, EAQ lifts NDCG@10 by 30.8% over the strongest dense baseline, with consistent gains across five analytical categories and two external datasets.

1 Introduction

Deploying LLM-based conversational systems at scale demands observability infrastructure to detect failure modes, safety violations, and user-experience degradations. One such core capability is Conversational Data Retrieval (CDR): given a corpus of multi-turn conversations and an analytical query (e.g., "find conversations where the user expressed frustration and the assistant issued a safety refusal"), return the relevant subset. In CDR, the conversations themselves become the retrieval units, and queries articulate analytical intents over conversational patterns.

CDR poses challenges absent in standard document retrieval, where conversations are not static documents but interdependent multi-turn sequences. Dense retrieval, which encodes queries and documents into fixed-dimensional vectors ranked by similarity (Karpukhin et al., 2020a), is the dominant paradigm for semantic search. Yet the CDR-Benchmark (Lee et al., 2025) and related studies (Chen et al., 2024a; Yang et al., 2025b) confirm it fails on conversational corpora, owing to semantic drift as topics shift across turns, implicit state distributed over non-contiguous turns, and unbounded intent vocabularies that resist fixed-schema indexing (Su et al., 2019). The architectural limitation is that a single dense vector cannot jointly encode the structured constraints (e.g., "safety-based refusal") and open-ended semantic content (e.g., "booking a ride to the airport") that analytical queries reference together.

We address this gap with our EAQ framework that transforms raw conversations into queryable graph representations exposed through an agentic natural language interface. The insight is that conversational retrieval requires operating over two information planes: the semantic content of utterances and structured metadata characterizing conversational dynamics (intent, emotion, response type, repetition patterns, safety domains, task outcomes). Our framework bridges these planes in three phases. First, it derives a metadata taxonomy from the analytical queries themselves, inducing dimensions over utterance, response, and conversation-level properties rather than imposing them a priori. Second, it annotates conversations via LLM-based labeling with sliding-window chunking to preserve cross-turn context, then loads results into a property graph. This enables exact-match filtering on closed-set fields and vector similarity search over open-ended fields. Third, an agentic retrieval pipeline compiles natural-language queries into validated, executable retrieval operations.

2 Related Work

Information retrieval (IR) has progressed from lexical methods (Robertson and Zaragoza, 2009) to dense neural retrievers (Karpukhin et al., 2020b), enabling semantic understanding in single-turn queries. Recent benchmarks like CDR-Benchmark (Lee et al., 2025) and CORAL (Cheng et al., 2025) evaluate retrieval over conversation logs for analytical tasks, revealing that embedding models achieve modest scores due to failures in capturing cross-turn context.

Beyond these foundational approaches, researchers developed contextual and history-aware retrieval techniques. HAConvDR incorporates prior turns to mitigate drift (Mo et al., 2024), while LLM-cognition alignment enables cross-domain generalization (Chen et al., 2024a). Unsupervised methods leverage LLMs for soft label generation: InstructoR (Jin et al., 2023) uses LLM-estimated relevance scores as soft labels to guide unsupervised retriever training, while ContextualRetriever (Yang et al., 2025b) embeds dialogue history with intent-guided supervision. Interpretability enhancements like rewriting-enhanced inversion of session embeddings bridge opaque representations with explicit query intents (Cheng et al., 2024).

A line of work targets structured representations for hybrid queries, yet each is confined to one retrieval mode. LLM-generated summaries (Lee et al., 2024) compress conversations into dense abstractions amenable to semantic matching but dissolve the discrete signals (refusal type, safety domain, task outcome) that analytical queries invoke as exact constraints. Intent-based graphs (Tu et al., 2024; Ledneva and Kuznetsov, 2024) model dialogue flow as node-linked structures but rely on closed-ontology schemas that preclude open-vocabulary matching. Automated annotation pipelines (Petukhova and Kochmar, 2025) enrich utterances with dimensions such as emotion and response type but emit flat per-utterance labels amenable to neither cross-conversation aggregation nor hybrid filtering.

The common obstacle is a mismatch between representation and query: summaries and dense encoders inhabit a continuous space affording similarity but not discrete predicates, while intent graphs and relational schemas inhabit a discrete space affording exact matching but not open-vocabulary similarity. To our knowledge, no prior conversation-level retrieval system reconciles the two planes for conversation-level retrieval. EAQ addresses this by representing metadata values as embedded, queryable entities, so that discrete metadata predicates and open-vocabulary value similarity apply within a single in-database retrieval pipeline. The ingredients are individually established: hybrid vector search, text-to-Cypher generation, and LLM-based annotation all predate this work. What we contribute is their synthesis for CDR, together with an empirical account of how much each plane adds and what the combination costs to operate.

3 Problem Statement

We formalize the Conversational Data Retrieval (CDR) task as follows. Let $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$ denote a corpus of n conversations, where each conversation $c_i = \{t_{i,1}, t_{i,2}, \ldots, t_{i,m_i}\}$ is a sequence of m_i turns. Each turn $t_{i,j}$ comprises a user utterance $u_{i,j}$ and an assistant response $a_{i,j}$: $t_{i,j} = (u_{i,j}, a_{i,j})$. Given a natural language query Q, the objective is to retrieve a subset of conversations $R_Q \subseteq \mathcal{C}$ relevant to Q, where relevance is defined by a binary judgment $\text{rel}(c_i, Q) \in \{0, 1\}$.

4 Methodology

4.1 Metadata Taxonomy Construction

Conversational retrieval queries implicitly reference structured metadata dimensions (e.g., user intent, response type, safety violations) requiring explicit modeling for hybrid search over semantic content and structured constraints. We derive a metadata vocabulary M through extraction-consolidation that grounds the taxonomy in actual analytical information needs from retrieval queries. Given query corpus $Q = \{Q_1, Q_2, \ldots, Q_k\}$, we apply LLM extractor f_{extract} with structured prompting (refer to Appendix H.1) to identify metadata dimensions referenced in each query: $F_k = f_{\text{extract}}(Q_k) = \{(f_1, v_1), (f_2, v_2), \ldots\}$, where each (f, v) pair represents a metadata field and its associated value. We aggregate extractions across all queries to obtain the complete set of field-value pairs: $F_{\text{all}} = F_1 \cup F_2 \cup \cdots \cup F_K$. Raw extractions in F_{all} are deduplicated by field label to obtain a candidate set of unique metadata dimensions $F_{\text{unique}} = \{f \mid (f, v) \in F_{\text{all}}\}$ for human review.

Expert annotators review deduplicated field labels and merge semantically equivalent dimensions based on field names and value distributions. This consolidation addresses lexical variation (e.g.,

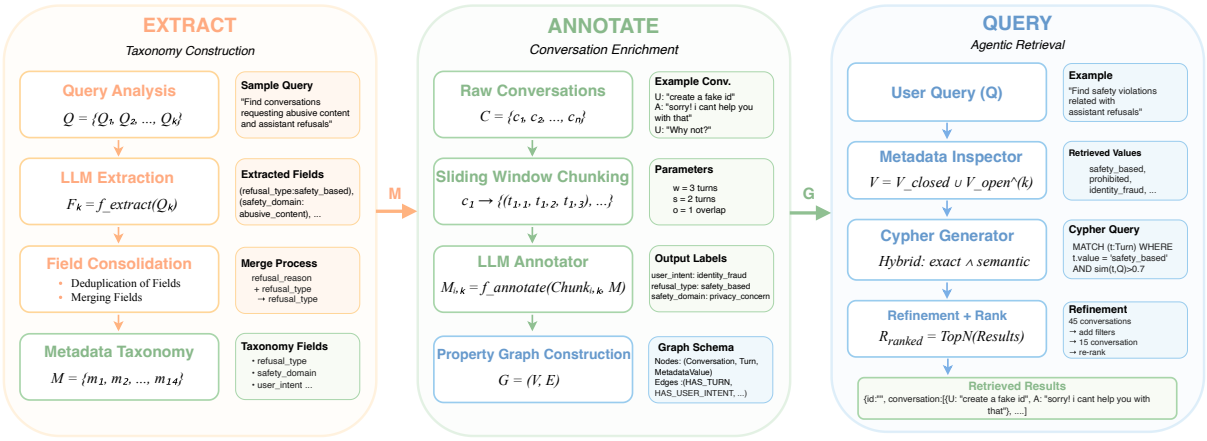


Figure 1: Extract-Annotate-Query Framework for Conversational Data Retrieval

"user_mood" and "user_emotion") while preserving distinct metadata concepts. For reproducibility, we measure inter-annotator agreement using Cohen's κ (Cohen, 1960) for pairwise agreement and Fleiss' κ (Fleiss, 1971) for multi-annotator consensus. We classify metadata fields as closed-set (enumerable values) or open-ended (unbounded vocabulary) based on value distribution characteristics (see Appendix A.1 for formal criteria). This classification is critical because open-ended fields require semantic similarity search over unbounded vocabularies. The taxonomy supports incremental evolution: when new queries fall below a cosine similarity threshold (τ_{coverage}) against existing dimensions, candidate fields are extracted and reviewed to be either merged into existing dimensions or added as new dimensions, with annotation applied only to affected conversations. The metadata taxonomy is derived once from a corpus of analytical *queries*, requiring no a priori access to target conversations. It is thereafter applied zero-shot: new conversations, and entire new corpora, are annotated and retrieved without re-deriving it. We validate this empirically, the taxonomy covers 94.7% of held-out queries (§ 6.1) and transfers without modification to two external datasets (§ 6.3), confirming that taxonomy construction generalizes beyond the queries it was built on.

4.2 LLM-Based Conversation Annotation

We apply taxonomy M to enrich raw conversation c_i with structured metadata, enabling hybrid retrieval over semantic content and structured constraints. This proceeds in two stages: sliding-window chunking and LLM-based structured labeling. Since annotation quality bounds retrieval performance, we validate against gold-standard human labels, reporting agreement metrics in § 5.3.

4.2.1 Sliding Window Chunking

Direct turn-by-turn annotation fails to capture conversational context, as the meaning of utterance $u_{i,j}$ depends on prior turns. Conversely, annotating entire conversations in a single LLM call exceeds context window limits for long sessions and produces lower-quality labels due to attention dilution. We address this through overlapping sliding windows that provide local context while maintaining computational efficiency (analyzed in Appendix D.3). Given a conversation $c_i = \{t_{i,1}, t_{i,2}, \ldots, t_{i,m_i}\}$ with m_i turns, we partition it into overlapping chunks: $\text{Chunk}_{i,k} = \{t_{i,j} \mid (k-1) \cdot s + 1 \leq j \leq \min((k-1) \cdot s + w, m_i)\}$, where w denotes window size (number of turns per chunk), s denotes stride (number of turns between consecutive window starts), and $\min()$ ensures the last chunk does not exceed conversation length.

4.2.2 Structured LLM Annotation

Each chunk $\text{Chunk}_{i,k}$ is processed by an LLM-based annotator f_{annotate} that assigns metadata values from the taxonomy M to each turn within the chunk: $M_{i,k} = f_{\text{annotate}}(\text{Chunk}_{i,k}, M)$. The annotator is implemented via structured prompting (refer to Appendix H.2) with defined schema enforcement to ensure output conformity. The prompt provides: (1) the taxonomy M with field definitions and value constraints, (2) the chunk text with turn boundaries marked, and (3) instructions to emit per-turn and chunk-level annotations. The structured output $M_{i,k}$ contains three annotation types: (1) *Turn-level user annotations* ($m^u_{i,j} \in M_u$): describing user utterances (e.g., intent, emotion, topic), (2) *Turn-level assistant annotations* ($m^a_{i,j} \in M_a$): describing assistant

responses (e.g., response type, refusal category, safety domain), and (3) *Conversation-level annotations* ($m^c_{i,k} \in M_c$): describing multi-turn patterns (e.g., error types, repetition). This enables the LLM to choose relevant field based on the user query.

4.3 Graph-Based Structured Representation

Annotated conversations are stored in a property graph $G = (V, E)$ that elevates content and metadata to first-class entities. The node set comprises three types: **conversation nodes** (aggregation points for multi-turn retrieval), **turn nodes** (user utterance and assistant response), and **metadata value nodes** (one per observed value of each field). Turn and metadata value nodes carry precomputed dense embeddings. These are linked by structural edges (`HAS_TURN`, `REPLIES_TO`) encoding conversation composition and turn ordering, and typed semantic edges (`HAS_{m}_VALUE`) connecting turns and conversations to metadata value nodes. Representing metadata values as independent embedded nodes, rather than as node properties, enables vector similarity search over open-ended fields with unbounded vocabularies and efficient cross-turn aggregation via shared nodes without full-corpus scans; we evaluate it in § 6.2. The schema thus supports four complementary query patterns, exact-match filtering, vector similarity search, graph traversal for sequential patterns, and conversation aggregation, within a single retrieval operation. Full node-property tuples, edge schema, and query-pattern details are in Appendix B.1.

4.4 Agentic Retrieval Pipeline

The graph is exposed through an agentic retrieval pipeline that translates natural language queries into executable graph operations, implemented as a multi-stage state machine $A = (S_1, S_2, \ldots, S_5)$. The **Metadata Inspector** (S_1) grounds query against the metadata vocabulary, enumerating all values for closed-set fields (e.g., `refusal_type`, `task_outcome_status`) and retrieving top-k most semantically similar values for each open-ended field (e.g., `intent`, `topic_category`) by vector search, with k set per field by ablation in § 5.4. The **Query-to-Cypher Generator** (S_2) processes the query and retrieved vocabulary in a single LLM call (prompt in Appendix H.3), jointly analyzing query intent and emitting an executable openCypher query[1] with filter parameters: turn- or conversation-level granularity, relevant fields and their target values, and AND/OR filter combination. When the query references no available metadata value, the generator returns an empty result set rather than constructing a query from poorly matched values, a graceful exit signaling that the queried concept falls outside the current annotation vocabulary. The **Query Executor** (S_3) runs the generated query against the graph with bounded retries on syntax or execution errors. The **Result Ranker** (S_4) orders retrieved conversations by maximum semantic similarity between their matched turns and the query embedding. The **Response Formatter** (S_5) returns each conversation with its matched turns and conversation-level relevance score. Full per-stage parameter schemas, filter-combination strategy, and scoring equations are in Appendix B.2.

4.5 Design Rationale: Property Graph and Agentic Retrieval

Three design choices carry distinct burdens, and it is worth separating them. The *representation*, metadata values as embedded entities rather than node properties, is what improves retrieval quality; we isolate its contribution in § 6.2. The *graph engine* contributes systems properties: a reduced search space and a single execution unit for hybrid retrieval. The *agent* contributes interface properties: grounding of open-vocabulary values, machine-verifiable output, and explicit failure. We take each in turn.

Property graph. Each observed value of each field becomes one embedded node, and two consequences follow. First, vector search over an open-ended field runs against that field's value vocabulary rather than the turn corpus, and a field's set of distinct values is far smaller than the set of turns it annotates. Second, the same node serves as the aggregation point, since its incident edges yield every referencing turn and, through `HAS_TURN`, every parent conversation. A non-graph store can also deduplicate values, but the value-to-turn mapping is then a separate inverted index joined in application code; here it is the edge set itself.

Retrieval consequently executes as one in-database pipeline. Exact predicates, traversal over `REPLIES_TO` for sequential patterns, aggregation over `HAS_TURN`, and ranking are planned and run by the engine. Assembling the same behaviour outside a graph database requires a vector index

[1] openCypher is a declarative graph query language: https://opencypher.org

per open-ended field, a separate store for closed-set predicates, application-side joins to roll turns up to conversations, and a separate ranking pass, with query planning and cross-store consistency moved into application code. The engine buys execution properties, not accuracy: a graph holding only conversation and turn nodes exposes no signal beyond query-turn similarity, a point we return to in § 6.2.

Agentic retrieval. Grounding is what makes open-ended fields queryable at all. Stage 1 retrieves the nearest existing values per field before generation, so the generator selects from the corpus vocabulary rather than inventing plausible labels that no node carries. Because Stage 2 emits openCypher rather than an opaque scoring decision, the output is machine-verifiable: the engine validates it, and execution errors drive a bounded retry loop. Failures are therefore observable and attributable to a stage, and we score unrecoverable ones rather than discarding them (App. E.2), so the generator's reliability bounds end-to-end quality directly. Finally, when a query references concepts outside the annotation vocabulary the agent returns the empty set. In an analytics setting an explicit no-coverage signal is preferable to a confidently wrong result set, and it feeds the taxonomy evolution loop of § 4.1.

5 Experimental Setup

5.1 Dataset and Evaluation Protocol

We evaluate on CDR-Benchmark (Lee et al., 2025), which contains 9,146 multi-turn conversations paired with 1,583 analytical queries spanning five categories (Intent & Purpose, Emotion & Feedback, Linguistic Style & Expression, Trust, Safety & Ethics, and Conversation Dynamics). The benchmark is derived from 11 open-source dialogue datasets ensuring domain diversity across task-oriented, adversarial, and open-domain interactions. We partition queries into taxonomy derivation (791, 50%) and held-out validation (792, 50%) sets via stratified sampling. We report Precision@10 (P@10), Recall@10 (R@10), and Normalized Discounted Cumulative Gain@10 (NDCG@10), macro-averaged across all queries and with per-category breakdowns.

5.2 Implementation Details

We annotate all 9,146 conversations with the LLM-based annotator (selected in § 5.3) using $w = 3$, $s = 2$, at a one-time cost of ~$38.1, or $0.0042 per conversation. Annotation is the only stage whose cost grows with corpus size, and it grows linearly and parallelises directly: the full corpus was annotated in roughly 70 minutes on eight workers, so wall-clock time is set by worker count rather than by corpus size. At the same per-conversation rate, a corpus of one million conversations costs on the order of a few thousand dollars to annotate, incurred once against a corpus that is then queried indefinitely. The taxonomy is derived from analytical queries rather than from conversations (§ 4.1), so it is not re-derived as the corpus grows, and annotation extends incrementally to newly arriving conversations alone. For all semantic similarity operations, we adopt stella_en_1.5B_v5 (Zhang et al., 2025a), selected for superior retrieval quality in an encoder comparison (see Appendix E.1) and for concordance with the CDR-Benchmark baseline encoder (Lee et al., 2025), isolating observed gains to our method rather than encoder choice. Annotated conversations, precomputed embeddings, and metadata value nodes are loaded into a property graph on AWS Neptune Analytics (Amazon Web Services, 2024), selected for native openCypher and vector similarity search support, though portable to any graph database with comparable capabilities. The final graph contains 9,146 conversation nodes, 28,124 turn nodes, and 55,686 metadata value nodes across 16 dimensions. Processing times and infrastructure details are in Appendix D.4.

5.3 Model Selection

The framework relies on LLM-based reasoning at three stages: metadata field extraction, structured turn annotation, and query-to-Cypher generation. To balance quality and computational cost, we conduct a model selection study spanning all three. Full methodology, datasets, and scoring conventions are in Appendix E.2, and annotation guidelines in Appendix C. Table 1 reports results across model size regimes: Qwen3-Next-80B-A3B-Instruct (Yang et al., 2025a) achieves the best quality-efficiency trade-off and is used throughout our experiments.

Is an LLM annotator necessary? To probe whether a lightweight encoder might suffice, we fine-tuned ModernBERT-base (Warner et al., 2024), a recent state-of-the-art encoder, on our 500 gold-annotated conversations. It reached only 0.75 F1-score on closed-set fields and 0.48 on open-ended fields, well below the best LLM annotator (0.91

Model	Taxonomy Extraction			Conversation Annotation			Agentic Pipeline & Operational				
	P	R	F1	P	R	F1	Query Correct.	NDCG@10	P@10	R@10	Rel. Cost
Claude Sonnet 4.5 (Anthropic, 2025b)	**0.89**	**0.92**	**0.90**	**0.91**	**0.89**	**0.90**	**94.5%**	**0.650**	**0.600**	**0.400**	5.2x
Qwen3-Next-80B-A3B-Instruct	**0.87**	**0.90**	**0.88**	**0.89**	**0.87**	**0.88**	**92.1%**	**0.642**	**0.592**	**0.394**	1.0x
GPT-OSS-120B (Agarwal et al., 2025)	0.87	0.88	0.88	0.86	0.88	0.87	93.0%	0.628	0.610	0.388	1.3x
Claude Haiku 4.5 (Anthropic, 2025a)	0.82	0.85	0.83	0.84	0.80	0.82	88.4%	0.598	0.548	0.365	1.8x
GPT-OSS-20B (Agarwal et al., 2025)	0.71	0.74	0.72	0.73	0.69	0.71	74.7%	0.476	0.426	0.285	0.6x
Qwen3-8B (Yang et al., 2025a)	0.65	0.68	0.66	0.68	0.63	0.65	68.5%	0.412	0.362	0.241	0.2x

Table 1: Model performance across taxonomy extraction, annotation, and agentic pipeline tasks

and 0.86), as it cannot generate labels over the unbounded vocabularies of open-ended dimensions.

5.4 Hyperparameter Configuration

Three hyperparameters govern the pipeline. We classify metadata fields as closed-set or open-ended using distribution thresholds $\theta_{\text{conc}} = 0.90$ and $\theta_{\text{card}} = 30$. For each open-ended field, field-specific top-k values achieve 95%+ coverage while keeping the vocabulary manageable for LLM context windows. The sliding-window annotator uses $w = 3$, $s = 2$, the configuration that maximizes F1-score in a grid search. All three selections are justified by ablations in Appendix D.

6 Experiments & Results

6.1 Taxonomy Construction and Coverage

We derived 2,847 raw field-value pairs from 791 taxonomy derivation split queries, yielding 55 unique field labels after deduplication. This compression shows conversational retrieval queries, despite surface diversity, reference a bounded set of core metadata concepts. Three expert annotators independently reviewed (refer to Appendix C.2 for guidelines) the 55 field labels to merge (same aspect) or separate (distinct aspects) them, receiving field labels, value distributions, and example queries. Manual consolidation reduced 55 candidate fields to 16 canonical dimensions across 39 merge groups. See Appendix G.1 for detailed statistics and final metadata fields. We evaluate taxonomy coverage on both the derivation set (to validate consolidation completeness) and the held-out validation set (to measure generalization). The derived 16-dimension taxonomy achieves 98.4% coverage on the derivation set (778/791 queries) and generalizes well to held-out validation set queries with 94.7% coverage (750/792 queries). Only 1.6% of derivation queries (13/791) and 5.3% of validation queries (42/792) remain uncovered. Manual review of these uncovered queries reveals domain-specific requirements outside CDR-Benchmark scope.

6.2 Main Retrieval Results

Having validated that the derived taxonomy achieves high coverage on held-out queries (94.7%), we now evaluate whether this structured metadata representation translates to improved retrieval performance on the held-out validation set (792 queries). As presented in Table 2, we compare against three baseline approaches and evaluate three configurations of our framework to isolate each component's contribution. Appendix G.2 provides an end-to-end retrieval trace illustrating complementary failure modes of single-plane retrieval. We note that a graph storing only Conversation and Turn nodes with structural edges, omitting metadata value nodes and semantic edges, exposes no retrieval signal beyond query–turn embedding similarity, and is therefore functionally equivalent to our Semantic dense retrieval baseline.

Configuration	NDCG@10	P@10	R@10
Baselines			
BM25 (lexical)	0.372	0.298	0.104
Semantic dense retrieval	**0.491**	**0.414**	**0.259**
LLM-Summary Retrieval	0.452	0.354	0.190
InstructoR	0.411	0.392	0.163
Proposed Framework & Ablations			
Exact-Match Filtering	0.528	0.478	0.318
Hybrid w/o Ranking	0.592	0.544	0.376
Hybrid (Ours - Full System)	**0.642**	**0.592**	**0.394**
Improvements			
Δ vs. Semantic Dense Retrieval	+30.8%	+43.0%	+52.1%
Δ vs. Exact-Match Filtering	+21.6%	+23.8%	+23.9%
Δ Ranking Component	+8.4%	+8.8%	+4.8%

Table 2: Main Retrieval Results on Validation Set. Results are mean over 5 runs with different random seeds ($\sigma \leq 0.016$ across all configurations)

6.2.1 Baseline Methods

We evaluate against three baseline categories. **1) Lexical Retrieval Baseline:** BM25 (Robertson and Zaragoza, 2009) over turn text, aggregated by maximum turn score, provides a lexical lower bound using no metadata or embeddings (implementation in Appendix F.1). **2) Standard Dense Retrieval Baseline:** It performs pure vector similarity search over turn embeddings, the state-of-

the-art configuration from the CDR-Benchmark paper (Lee et al., 2025). **3) Adapted Conversational Search Baselines:** The most conversational retrieval methods retrieve external documents during a conversation, not conversations themselves as analytical units. We adapt two such methods to the inverted CDR task. **LLM-Summary Retrieval** embeds LLM-generated conversation summaries and matches them against queries, attaining 0.452 NDCG@10. **InstructoR** attains 0.411 NDCG@10 through unsupervised LLM-driven distillation that scores the likelihood of an analytical query given the conversational context. Detailed implementations are in Appendix F.2.

6.2.2 Framework Ablations

We evaluate three configurations that add one mechanism at a time, isolating each component's contribution (Table 2). All build on the same metadata representation: exact-match filtering on closed-set fields and semantic similarity search over metadata value embeddings for open-ended fields.

Exact-match filtering This configuration uses closed-set fields only, excluding open-ended fields to isolate the exact-match contribution, since their unbounded vocabularies require semantic similarity. It improves NDCG@10 by 7.5% over semantic dense retrieval, showing that exact-match constraints capture retrieval signals that pure embedding similarity misses.

Hybrid without ranking This configuration adds the open-ended fields back: it keeps exact-match filtering on closed-set fields and applies semantic search over the open-ended fields' value embeddings, but returns conversations in arbitrary order. It already reaches NDCG@10 = 0.592, a 20.6% gain over the baseline.

Hybrid (full system) On top of the hybrid configuration, the full system ranks the retrieved conversations by the semantic similarity between their matched turns and the query. This yields gains of 30.8% NDCG@10, 43.0% P@10, and 52.1% R@10 over the baseline (per-category split in Appendix G.3). We also experimented with an optional progressive refinement mechanism that narrows broad result sets by adding constraints from unused metadata fields. Overall gains were modest, but it improved highly granular queries needing 3+ constraints (+6.1% NDCG@10; Appendix G.4).

6.3 Cross-Dataset Generalization Validation

To validate generalizability beyond the CDR-Benchmark's 11 source datasets, we evaluate on two external corpora: ABCD (Chen et al., 2021) ($\sim$10K action-grounded customer service dialogues) and BST (Smith et al., 2020) ($\sim$5K skill-blended open-domain conversations). We sampled 25 queries per analytical category (125 total per dataset) by stratified random selection, retrieved top-10 results, and manually validated (refer to Appendix C.4 for annotation guidelines) all query-conversation pairs with inter-annotator agreement (Fleiss' $\kappa = 0.82$ for ABCD, $\kappa = 0.81$ for BST). Results are consistent across datasets (mean over 5 runs, $\sigma \leq 0.023$): ABCD attains NDCG@10 of 0.628 (P@10: 0.578, R@10: 0.381), BST reaches 0.635 (P@10: 0.585, R@10: 0.388), validating generalization to diverse conversational structures. Detailed procedures are in Appendix G.5.

7 Conclusion and Future Work

We demonstrate that conversational data retrieval requires operating jointly over structured metadata and semantic content, and that our Extract-Annotate-Query framework, by elevating LLM-derived annotations to graph entities queryable through an agentic natural language interface, addresses this gap with a 30.8% NDCG@10 improvement over the best dense retrieval baseline. We acknowledge that the framework relies on large LLMs for annotation and query generation. Future work includes knowledge distillation and hybrid routing to compact models and extension to multilingual settings.

Limitations

While the Extract-Annotate-Query framework demonstrates significant retrieval improvements over semantic dense retrieval baselines (30.8% relative improvement in NDCG@10), several limitations warrant discussion and motivate future research directions.

Model Size and Computational Efficiency The framework requires large LLMs (80B parameters) for high-quality annotations and query generation. Results demonstrate that smaller models (<30B parameters) exhibit substantial quality degradation. This dependency on large models poses challenges for production deployment at scale, particularly when processing massive conversation corpora where annotation costs scale linearly with cor-

pus size (approximately $0.0042 per conversation). Future work should explore knowledge distillation techniques to transfer the reasoning capabilities of large models to smaller, more efficient architectures. Task-specific finetuning of smaller models on high-quality annotations from the larger model could bridge this performance gap while reducing inference costs. Additionally, a hybrid routing strategy, using smaller models for common annotation patterns and reserving larger models for complex or ambiguous cases, merits investigation.

Language Coverage The current evaluation focuses exclusively on English conversations from the CDR-Benchmark corpus. Extending the framework to multilingual settings presents challenges in both annotation consistency and semantic similarity computation across languages. Future work should investigate cross-lingual embeddings and language-specific taxonomy adaptations to support conversation analytics in diverse linguistic contexts.

Cold Start Problem The framework requires complete corpus annotation before retrieval becomes effective, creating a cold start barrier for new deployments. The one-time annotation cost ($38.1 for 9,146 conversations) and processing time (approximately 70 minutes) may be prohibitive for exploratory use cases or rapidly evolving conversation datasets. Future work should investigate progressive annotation strategies that enable partial retrieval capability during the annotation process, or hybrid approaches that combine semantic dense retrieval for unannotated conversations with structured retrieval for annotated subsets, gradually improving precision as annotation coverage increases.

Ethical considerations

Data & Model Compliance This work uses 11 open-source dialogue datasets, two external validation datasets, multiple language models, and one embedding model. We have reviewed the licenses and terms of use for all resources and confirm compliance with their respective terms. Source datasets are released under a range of licenses including Apache 2.0, CC BY-SA 4.0, CC BY-NC-SA 4.0, CC BY-NC 4.0, MIT, and custom research-use agreements. We note that DailyDialog and OpenDialKG carry non-commercial restrictions, and our use is strictly for non-commercial research purposes. Among the models used, Qwen3-Next-80B-A3B-Instruct, GPT-OSS-120B, GPT-OSS-20B, and Qwen3-8B are distributed under Apache 2.0; Claude Sonnet 4.5 and Claude Haiku 4.5 are proprietary models accessed via API under Anthropic's Terms of Service. The `stella_en_1.5B_v5` embedding model is released under the MIT License, and our graph infrastructure relies on AWS Neptune Analytics.

Privacy & Safety Our framework incorporates privacy and safety safeguards directly within its processing pipeline. During the Extract and Annotate phases, LLM prompts explicitly instruct the model to redact or anonymize any personally identifiable information and offensive content encountered in conversations, replacing them with generic placeholders. Safety-related metadata fields, including `safety_domain_triggered`, `refusal_type`, and `safety_handling_outcome`, systematically capture and categorize potentially harmful content, enabling downstream filtering. Our processing retains only conversation text and derived metadata, with no user-level identifiers stored. Datasets containing user-generated content (LMSYS Chat, WildChat) were accessed through official distribution channels that implement appropriate consent and anonymization procedures.

Potential Risks We acknowledge that conversational data retrieval technology enables valuable applications such as failure pattern detection, safety monitoring, and system diagnostics in LLM-powered services. However, it also carries potential risks, particularly around privacy infringement or misuse as surveillance tools over user interactions. The EAQ framework's ability to systematically extract, annotate, and query conversation-level patterns at scale amplifies both its analytical utility and this dual-use concern. We recognize this tension, and hope the framework and evaluation methodology presented here support work that balances analytical utility against user privacy and dignity.

Disclosure of AI Assistance The authors acknowledge the use of Claude Sonnet 4.5 (Anthropic, 2025b) for linguistic refinement, LaTeX table construction, and paragraph optimization. These tools were used solely to enhance the readability and formatting of the manuscript. The final content and scientific integrity of this work remain the sole responsibility of the authors.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, Che Chang, Kai Chen, Mark Chen, and 80 others. 2025. GPT-OSS-120B & GPT-OSS-20B model card. *Preprint*, arXiv:2508.10925.

Amazon Web Services. 2024. Amazon Neptune Analytics. `https://aws.amazon.com/neptune/`.

Anthropic. 2025a. Introducing Claude 4.5 Haiku. `https://www.anthropic.com/news/claude-4-5-haiku`. Company Blog Post.

Anthropic. 2025b. Introducing Claude 4.5 Sonnet. `https://www.anthropic.com/news/claude-4-5-sonnet`. Company Blog Post.

Derek Chen, Howard Chen, Yi Yang, Alex Lin, and Zhou Yu. 2021. Action-based conversations dataset: A corpus for building more in-depth task-oriented dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2021)*, pages 3002–3017, Online. Association for Computational Linguistics.

Haonan Chen, Zhicheng Dou, Kelong Mao, Jiongnan Liu, and Ziliang Zhao. 2024a. Generalizing conversational dense retrieval via LLM-cognition data augmentation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2024)*, pages 2700–2718, Bangkok, Thailand. Association for Computational Linguistics.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024b. BGE M3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Yiruo Cheng, Kelong Mao, and Zhicheng Dou. 2024. Interpreting conversational dense retrieval by rewriting-enhanced inversion of session embedding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL 2024)*, pages 2879–2893, Bangkok, Thailand. Association for Computational Linguistics.

Yiruo Cheng, Kelong Mao, Ziliang Zhao, Guanting Dong, Hongjin Qian, Yongkang Wu, Tetsuya Sakai, Ji-Rong Wen, and Zhicheng Dou. 2025. CORAL: Benchmarking multi-turn conversational retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1308–1330, Philadelphia, USA. Association for Computational Linguistics.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2023. InstructoR: Instructing unsupervised conversational dense retrieval with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6649–6675, Singapore. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020a. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 6769–6781, Online. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020b. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles (SOSP 2023)*, pages 611–626, Koblenz, Germany. Association for Computing Machinery.

Daria Romanovna Ledneva and Denis Pavlovich Kuznetsov. 2024. Reimagining intent prediction: Insights from graph-based dialogue modeling and sentence encoders. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 13847–13860, Torino, Italia. ELRA and ICCL.

Seanie Lee, Jianpeng Cheng, Joris Driesen, Alexandru Coca, and Anders Johannsen. 2024. Effective and efficient conversation retrieval for dialogue state tracking with implicit text summaries. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers) (NAACL 2024)*, pages 96–111, Mexico City, Mexico. Association for Computational Linguistics.

Yohan Lee, Yongwoo Song, and Sangyeop Kim. 2025. Finding diamonds in conversation haystacks: A benchmark for conversational data retrieval. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 2343–2366, Suzhou, China. Association for Computational Linguistics.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2024. Towards general text embeddings with multi-stage contrastive learning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12686–12702, Bangkok, Thailand. Association for Computational Linguistics.

Fengran Mo, Chen Qu, Kelong Mao, Tianyu Zhu, Zhan Su, Kaiyu Huang, and Jian-Yun Nie. 2024. History-aware conversational dense retrieval. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13366–13378, Bangkok, Thailand. Association for Computational Linguistics.

Zach Nussbaum and Brandon Duderstadt. 2025. Training sparse mixture of experts text embedding models. *Preprint*, arXiv:2502.07972.

Kseniia Petukhova and Ekaterina Kochmar. 2025. A fully automated pipeline for conversational discourse annotation: Tree scheme generation and labeling with large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 15829–15852, Vienna, Austria. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. 2020. Can you put it all together: Evaluating conversational agents' ability to blend skills. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 2021–2030, Online. Association for Computational Linguistics.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, and Han Xiao. 2024. jina-embeddings-v3: Multilingual embeddings with task LoRA. *Preprint*, arXiv:2409.10173.

Hui Su, Xiaoyu Shen, Rongzhi Zhang, Fei Sun, Pengwei Hu, Cheng Niu, and Jie Zhou. 2019. Improving multi-turn dialogue modelling with utterance ReWriter. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 22–31, Florence, Italy. Association for Computational Linguistics.

Quan Tu, Chongyang Tao, and Rui Yan. 2024. Multi-grained conversational graph network for retrieval-based dialogue systems. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 11756–11765, Torino, Italia. ELRA and ICCL.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A ModernBERT for fast, memory efficient, and long context finetuning and inference. *Preprint*, arXiv:2412.13663.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Seunghan Yang, Juntae Lee, Jihwan Bang, Kyuhong Shim, Minsoo Kim, and Simyung Chang. 2025b. Learning contextual retrieval for robust conversational search. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 11991–12003, Suzhou, China. Association for Computational Linguistics.

Dun Zhang, Jiacheng Li, Ziyang Zeng, and Fulong Wang. 2025a. Jasper and Stella: Distillation of SOTA embedding models. *Preprint*, arXiv:2412.19048.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025b. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

A Taxonomy and Metadata Modeling

A.1 Field-Type Classification

Closed-set fields exhibit low-entropy values concentrated in a small, enumerable set: formally, fields where the cumulative frequency of the top-p most common values exceeds θ_{conc} and the total unique-value count stays below θ_{card} (θ_{conc}, θ_{card}, and top-p determined empirically). **Open-ended fields** (e.g., `user_intent`, `topic_category`) distribute values across a large, unbounded, high-entropy vocabulary, exhibiting long-tail distributions where no small value subset dominates the frequency mass.

B Graph Representation and Agentic Retrieval

B.1 Graph Schema and Supported Query Patterns

Node Schema The node set V comprises three types. **Conversation Nodes** (V_C): each conversation chunk $\text{Chunk}_{i,k}$ is a node $v_c = (\text{id}_{i,k})$, where $\text{id}_{i,k}$ is a unique identifier; these serve as aggregation points enabling multi-turn retrieval. **Turn Nodes** (V_T): each turn $t_{i,j}$ is a node $v_t =$

$(\text{id}_{i,j}, u_{i,j}, a_{i,j}, e_{i,j})$, where $u_{i,j}$ and $a_{i,j}$ are the user utterance and assistant response text and $e_{i,j} = \text{Encoder}(u_{i,j} \oplus a_{i,j})$ is a dense embedding ($\oplus$ denotes concatenation). **Metadata Value Nodes** (V_M): for each metadata field $m \in M$ (turn-level fields from $M_u \cup M_a$ and conversation-level fields from M_c) and each observed value v in the value space V_m, we create a node $v_{m,v} = (m, v, e_{m,v})$, where $e_{m,v} = \text{Encoder}(v)$ is the embedding of the value label.

Edge Schema Edges E encode structural relationships and semantic associations. **Structural edges**: `HAS_TURN` connects conversation nodes to their constituent turn nodes ($V_C \rightarrow V_T$), and `REPLIES_TO` connects consecutive turn nodes to encode sequential ordering ($V_T \rightarrow V_T$). **Semantic edges**: for each metadata field $m \in M$, typed edges connect turns or conversations to their metadata value nodes. Turn-level metadata uses edges of the form `HAS_{m}_VALUE`: $V_T \rightarrow V_M$ (e.g., `HAS_INTENT_VALUE`, `HAS_USER_EMOTION_VALUE`); conversation-level metadata uses `HAS_{m}_VALUE`: $V_C \rightarrow V_M$ (e.g., `HAS_UTTERANCE_REPETITION_VALUE`).

Supported Query Patterns This schema supports four complementary query patterns: (1) exact-match filtering, which retrieves all turns v_t connected to a metadata value node with a specified target value; (2) vector similarity search, which retrieves all turns v_t whose embedding e_t satisfies $\text{sim}(e_t, e_Q) > \tau$ for a query embedding e_Q; (3) graph traversal, which follows `REPLIES_TO` edges to identify multi-hop sequential patterns (e.g., three-turn sequences exhibiting emotion escalation); and (4) conversation aggregation, which traverses `HAS_TURN` edges from matching turns back to their parent conversation nodes. The combination of structured filtering and semantic search (hybrid retrieval) is central to our approach.

B.2 Agentic Retrieval Pipeline Stages

We give the formal operation of each stage of the pipeline $A = (S_1, \ldots, S_5)$ described in Section 4.4.

Stage 1 (Metadata Inspector) The agent retrieves the graph schema and metadata vocabulary to ground subsequent reasoning. For closed-set fields, all possible values V_{closed} are enumerated. For open-ended fields, the top-k most semantically similar values to the query are retrieved via open-Cypher vector search: $V^{(k)}_{\text{open}} = \text{TopK}(e_Q, \{e_{m,v} \mid v \in V_m\}, k)$, where e_Q is the query embedding.

Stage 2 (Query-to-Cypher Generator) An LLM-based generator f_{generate} maps the query Q and retrieved vocabulary to $\text{params}_Q = f_{\text{generate}}(Q, V_{\text{closed}}, V^{(k)}_{\text{open}})$, where $\text{params}_Q = \{\text{cypher} : \text{Cypher}_Q,\ \text{fields} : M_{\text{relevant}},\ \text{level} : \{\text{turn} \mid \text{conv.}\},\ \text{values} : V_{\text{target}},\ \text{comb.} : \{\text{AND} \mid \text{OR}\}\}$. A single LLM call jointly performs query understanding and code generation: *level* sets turn- or conversation-level granularity, M_{relevant} and V_{target} identify the relevant metadata fields and their target values, *comb.* combines constraints via OR or AND (detailed under Query Construction Strategy below), and the model emits the open-Cypher query Cypher_Q implementing these parameters. The generated query filters turns by metadata and semantic similarity, applies the filter combination, traverses `HAS_TURN` edges to retrieve parent conversations, and aggregates results at the conversation level. When the query references concepts that align with no available metadata value, the generator returns $M_{\text{relevant}} = \emptyset$ and the agent terminates with $R_Q = \emptyset$.

Stage 3 (Query Executor) The generated open-Cypher query is executed against the graph database: $R_{\text{raw}} = \text{Execute}(\text{Cypher}_Q, G)$. On syntax or execution errors, the agent retries with error feedback for up to n_{exec} attempts; if all retries fail, it routes to a terminal error state.

Stage 4 (Result Ranker) For each conversation $c_i \in R_{\text{raw}}$ with matched turns T^i_{match}, the agent computes $\text{score}(c_i) = \max_{t_{i,j} \in T^i_{\text{match}}} \text{sim}(e_{i,j}, e_Q)$, where $e_{i,j}$ and e_Q are the embeddings of turn $t_{i,j}$ and query Q. Conversations are ranked by $\text{score}(c_i)$, and the top-n results are returned with matched turns sorted by descending similarity: $R_{\text{ranked}} = \text{TopN}\big(\{c_i, T^i_{\text{match}}\} \mid c_i \in R_{\text{raw}}\big)$.

Stage 5 (Response Formatter) The final output is structured as $R_Q = \{(c_i, T^i_{\text{match}}, \text{score}_i) \mid c_i \in R_{\text{ranked}}\}$, where c_i is the full conversation, T^i_{match} are the matched turns, and score_i is the conversation-level relevance score. Returned metadata includes the executed Cypher query, selected fields, and refinement information.

Query Construction Strategy Field treatment during Cypher generation follows the closed-set versus open-ended distinction established at taxonomy construction. Closed-set fields (M_{closed}) use

exact-match predicates (e.g., `WHERE v.'value' IN ['safety_based', 'policy_based']`). Open-ended fields (M_{open}) instead match the top-k semantically nearest values through Neptune Analytics built-in vector-similarity functions emitted in the generated Cypher. Conversation-level fields (M_c) are queried directly on conversation nodes without turn traversal, since they describe multi-turn patterns annotated at that level (e.g., apology loops via `ResponseRepetitionValue` nodes). The filter combination is chosen per query from the semantic relationship between the relevant fields: OR (`UNION`) merges alternative signals for one concept (e.g., real estate matching `TopicCategoryValue`, `IntentValue`, or `SlotEntityValue`), whereas AND (`WITH` chaining) enforces independent constraints that must jointly hold (e.g., frustration about cooking requires both `UserEmotionValue` and `TopicCategoryValue`).

C Annotation Guidelines

C.1 Taxonomy Extraction Validation

Objective: Evaluate extracted metadata field-value pairs against conversation turns to compute precision, recall, and F1. Given a query and the model's candidate field-value pairs, annotators label each pair **Correct** (field and value accurately reflect a dimension present or implied in the turns) or **Incorrect** (supplying the corrected value), then list any field-value pair clearly implied by the turns but missing from the output. Annotation is independent; disagreements are resolved by majority vote, with adjudication for three-way conflicts.

C.2 Taxonomy Construction (Field Consolidation)

Objective: Consolidate 55 candidate metadata field labels into a coherent taxonomy. For each field pair, annotators decide to merge (same underlying aspect, differing only in granularity or phrasing, with a brief justification) or keep separate (distinct phenomena), supported by the field label, its value distribution, and example queries. Decisions weigh **semantic overlap** (same property despite different names), **discriminative power** (whether merging loses retrieval-useful distinctions), and **query utility** (whether a user would need to distinguish the two). Annotation is independent, with disagreements resolved in a group consensus session.

C.3 Conversation Annotation

Objective: Establish a gold-standard reference by labeling a stratified sample of 500 conversations (balanced across length and topic) at the turn level. After reading each full dialogue, annotators label every turn across all 16 dimensions: the **9 closed-set** dimensions from predefined sets (falling back to "others"/"not applicable"), and the **7 open-ended** dimensions as concise free text. Three challenges receive special care: **semantic drift** (annotate by local context, marking topic transitions), **implicit references** (resolve back-references such as "Do that again" using prior history), and **multi-label turns** (assign a primary label, record secondaries in comments). A Taxonomy Definition Document provides label descriptions and boundary cases. Annotation is independent; disagreements are resolved by majority vote, with adjudication for three-way conflicts.

C.4 Cross-Dataset Validation

Objective: Validate the protocol's generalizability on two external datasets not used in taxonomy development. Annotators apply the Section C.3 protocol to **ABCD** (action-based, task-oriented customer-service dialogues, e.g., order management and account inquiries; structured workflows with explicit action sequences) and **BST** (open-domain conversations blending persona, knowledge, and empathy; free-form with rapid topic switching). For each dataset they annotate all query-conversation pairs and verify that labels match the taxonomy definitions, that queries reflect their intended conversational pattern, and that no systematic domain-specific bias arises, flagging any dimension poorly suited to the dataset with a brief explanation.

C.5 Offensive Content Warning

Warning to Annotators: Some conversations may contain offensive, harmful, or sensitive content spanning chemical safety, food safety, discrimination, privacy concerns, prohibited content, ethical concerns, sexual content, suicide-related content, animal abuse, and other safety-related topics. Annotators are advised to take breaks as needed, report distressing content to the study coordinators, and skip and flag any conversation they are uncomfortable continuing.

Field	Unique Values	Top-5 Cov. (%)	Top-10 Cov. (%)	Classification
task_outcome_status	6	100.0	100.0	Closed-set
safety_handling_outcome	6	99.9	100.0	Closed-set
refusal_type	7	99.8	100.0	Closed-set
safety_domain_triggered	17	97.9	100.0	Closed-set
utterance_repetition	7	92.2	100.0	Closed-set
assistant_response_tone	24	95.5	97.7	Closed-set
response_repetition	8	92.3	100.0	Closed-set
assistant_emotion	24	89.1	95.8	Closed-set
assistant_response_type	28	90.6	95.8	Closed-set
topic_category	901	48.2	56.1	Open-ended
user_emotion	335	65.2	69.5	Open-ended
conversation_pattern_error_type	25	57.8	78.9	Open-ended
user_intent	2,751	46.5	63.1	Open-ended
user_slot_entity_resolution	24,534	23.1	34.7	Open-ended
user_topics_discussed	11,234	12.5	20.2	Open-ended
assistant_topics_discussed	15,779	21.0	34.2	Open-ended

Table 3: Value distribution statistics for representative metadata fields. Classification thresholds: $\theta_{\text{conc}} = 0.90$, $\theta_{\text{card}} = 30$.

D Hyperparameter Selection

D.1 Field Type Classification Threshold Selection

Table 3 reports the value-distribution statistics that justify our classification thresholds ($\theta_{\text{conc}} = 0.90$, $\theta_{\text{card}} = 30$). The separation is bimodal: closed-set fields have fewer than 30 unique values with top-10 coverage above 90%, whereas open-ended fields span larger vocabularies at an average top-10 coverage of 50.9%. The criteria therefore separate fields needing exact-match filtering from those needing semantic similarity search, yielding 9 closed-set and 7 open-ended dimensions; `slot_entity` shows hybrid behavior depending on domain.

D.2 Top-K Selection for Open-Ended Fields

Methodology For each open-ended metadata field (`topic_category`, `user_emotion`, `conversation_pattern_error_type`, `user_intent`, `user_slot_entity_resolution`, `user_topics_discussed`, `assistant_topics_discussed`), we vary $k \in \{5, 10, 15, 20\}$ and measure two metrics on the derivation set (791 queries): **Coverage**, the percentage of queries whose LLM-selected value (in the generated Cypher) appears within the top-k retrieved values; and **Selection Distribution**, the rank position of that value within the top-k list.

Findings Coverage increases monotonically with k and shows field-specific diminishing returns (see Table 4). `topic_category`, `user_emotion`, and `conversation_pattern_error_type` reach $\geq 95\%$ coverage at $k = 10$; `user_intent` reaches it at $k = 15$; and `user_slot_entity_resolution`, `user_topics_discussed`, and `assistant_topics_discussed` require $k = 20$. For the fields satisfied at $k = 10$, the LLM selects from ranks 1–5 in 68–81% of cases and from ranks 1–10 in 85–93%; fields needing larger k are more dispersed, with substantial selection in ranks 11–20. Semantically similar values thus correlate strongly with query relevance, although some fields carry greater semantic diversity.

Configuration We adopt these field-specific k values, which together achieve $\geq 95\%$ coverage while keeping vocabulary sizes manageable for LLM context windows. The residual $\leq 5\%$ of cases, where the selected value falls outside the top-k, involve rare or highly specific values that are semantically distant from the query text.

D.3 Sliding Window Parameter Selection

To explore this trade-off systematically, we conduct a grid search over window sizes $w \in \{1, 2, 3, 4, 5, 6\}$ and strides $s \in \{1, 2, 3, 4\}$, subject to $1 \leq s < w$ for non-degenerate chunking. From the full grid we select one representative configuration per window size, choosing for each w the largest stride that keeps at least one turn of overlap ($o = w - s \geq 1$), which is impossible only at $w = 1$. The resulting six configurations span the full spectrum from no-context single-turn annotation ($w = 1$) to broad-context chunking ($w = 6$), letting us trace annotation quality against context size. Table 5 reports results on the 200-conversation evaluation set using Qwen3-Next-80B-A3B-Instruct. Single-turn annotation ($w = 1$)

Field	Coverage (%)				Selection Distribution (%)				
	k=5	k=10	k=15	k=20	Rank 1-5	Rank 6-10	Rank 11-15	Rank 16-20	Outside Top-20
topic_category	88.3	**96.2**	98.1	99.4	75.6	18.2	4.1	1.7	0.4
user_emotion	85.7	**95.4**	97.3	98.6	68.4	22.3	5.2	2.8	1.3
conversation_pattern_error_type	90.1	**97.5**	98.7	99.2	81.3	13.7	3.1	1.6	0.3
user_intent	82.4	91.6	**96.1**	97.8	70.2	18.5	7.9	3.1	0.3
user_slot_entity_resolution	75.8	85.3	91.4	**95.7**	62.3	20.4	9.8	5.6	1.9
user_topics_discussed	73.2	83.6	90.5	**95.3**	58.7	22.1	11.6	7.2	0.4
assistant_topics_discussed	74.5	84.2	91.3	**96.4**	60.1	21.4	10.8	6.9	0.8

Table 4: Coverage and Selection Distribution Analysis for Open-Ended Metadata Fields

w	s	o	F1	P	R	LLM Calls
1	1	0	0.74	0.78	0.71	1.00x
2	1	1	0.83	0.85	0.81	0.72x
3	2	1	0.88	0.89	0.87	0.35x
4	2	2	0.87	0.90	0.84	0.30x
5	3	2	0.85	0.91	0.80	0.24x
6	4	2	0.82	0.90	0.76	0.20x

Table 5: Impact of sliding window parameters on annotation quality and computational cost. Window size (w), stride (s), overlap (o), F1-Score (F1), Precision (P), Recall (R), and relative number of LLM API calls compared to turn-by-turn annotation baseline.

yields the lowest F1 (0.74), confirming that cross-turn context is essential for accurate metadata labeling. Raising the window from 1 to 3 produces consistent precision and recall gains, as surrounding turns help the annotator disambiguate intent, resolve coreferences, and detect multi-turn patterns. Beyond $w = 3$, precision improves only marginally while recall declines, as the annotator's attention spreads across more turns and fine-grained turn-level details are missed. The trade-off peaks at $w = 3$, $s = 2$ (F1 = 0.88), which also reduces LLM calls by roughly 65% relative to single-turn annotation. We adopt this configuration ($w = 3$, $s = 2$, $o = 1$) for all subsequent experiments.

D.4 Processing Infrastructure

Annotation Conversation annotation ran in ~70 minutes using eight parallel workers on vLLM (Kwon et al., 2023) with eight NVIDIA A100 GPUs.

Embedding Generation Batch embedding generation (stella_en_1.5B_v5) for all turn and metadata value nodes took ~20 minutes on a single NVIDIA A100, with embeddings stored as float arrays in Neptune Analytics node properties.

Graph Construction The property graph was constructed on a Neptune Analytics instance provisioned with 32 Neptune Capacity Units (NCUs), connected by ~451k edges. Query latencies range from 50 ms for simple exact-match queries to 800 ms for complex queries involving semantic searches and graph traversals.

E Model and Embedding Selection

E.1 Embedding Model Selection

Since all semantic matching, covering both turn content and metadata-value similarity, depends on the embedding model, we compared several open models on the validation set with the full pipeline held fixed. To keep encoding efficient at corpus scale, we restricted candidates to models under 2B parameters: `stella_en_1.5B_v5` (Zhang et al., 2025a), `Qwen3-Embedding-0.6B` (Zhang et al., 2025b), `gte-Qwen2-1.5B-instruct` (Li et al., 2024), `jina-embeddings-v3` (Sturua et al., 2024), `bge-m3` (Chen et al., 2024b), and `nomic-embed-text-v2-moe` (Nussbaum and Duderstadt, 2025). Table 6 reports end-to-end NDCG@10, P@10, and R@10. `stella_en_1.5B_v5` gave the strongest retrieval quality, with the full ranking shown in Table 6. We adopt it throughout, as it also matches the dimensionality and configuration of the CDR-Benchmark baseline encoder, isolating the observed gains to our method rather than the choice of encoder.

Embedding Model	NDCG@10	P@10	R@10
`nomic-embed-text-v2-moe`	0.577	0.531	0.353
`bge-m3`	0.591	0.544	0.362
`jina-embeddings-v3`	0.605	0.558	0.371
`gte-Qwen2-1.5B-instruct`	0.619	0.571	0.379
`Qwen3-Embedding-0.6B`	0.628	0.580	0.386
`stella_en_1.5B_v5`	**0.642**	**0.592**	**0.394**

Table 6: End-to-end retrieval with different embedding models (pipeline fixed, validation set).

E.2 Model Selection: Detailed Evaluation

The framework relies on LLM-based reasoning at the three stages introduced in Section 5.3: metadata field extraction, structured turn annotation, and query-to-Cypher generation with result validation. To balance quality and computational cost consistently across these components, we conduct a unified model selection study spanning all three tasks.

E.2.1 Evaluation Setup

Each component is evaluated on a dedicated CDR-Benchmark-derived dataset with task-appropriate metrics and gold-standard annotations.

Taxonomy Extraction 250 stratified derivation-set queries (50 per category) with manually annotated metadata field-value pairs, reporting Precision, Recall, and F1-score.

Conversation Annotation 500 stratified conversations (balanced across length and topic) with manually annotated turn-level labels over all taxonomy dimensions, reporting Precision, Recall, and F1 macro-averaged across dimensions, using the fixed window configuration ($w = 3$, $s = 2$, $o = 1$).

Agentic Retrieval Pipeline The full validation set (792 queries) with existing gold-standard relevance judgments, reporting query correctness (the percentage of generated Cypher queries that execute without syntax errors) and end-to-end retrieval quality (NDCG@10, P@10, R@10). All metrics are macro-averaged over both successful and failed queries; a query that fails to produce valid Cypher after three retries scores 0 across all metrics.

E.2.2 Annotation Protocol

For the two tasks requiring gold-standard labels (taxonomy extraction and conversation annotation), three expert annotators with backgrounds in conversational AI independently labeled all evaluation samples. Inter-annotator agreement, measured with Fleiss' κ, reached 0.84 for taxonomy extraction and 0.83 for conversation annotation, confirming high label reliability. Disagreements were resolved by majority vote, with adjudication for three-way conflicts.

E.2.3 Results Analysis

Table 1 (main paper) presents results across model size regimes. Claude Sonnet 4.5 (Anthropic, 2025b) achieves the best quality on all three tasks, but at 5.2× the inference cost of Qwen3-Next-80B-A3B-Instruct (Yang et al., 2025a). Given its strong quality–efficiency trade-off, we select Qwen3-Next-80B-A3B-Instruct for all subsequent experiments. Smaller models (below 30B) degrade notably, particularly in query correctness (68.5% for 8B models), indicating that the structured reasoning these tasks demand favors larger models, though their lower cost and latency remain appealing for production settings. Distilling the larger model's capability into a smaller one could help close this gap, an avenue we leave for future work.

F Baseline Implementations

F.1 BM25 Lexical Baseline

We index each turn's concatenated user utterance and assistant response with BM25 (Robertson and Zaragoza, 2009) (default $k_1 = 1.5$, $b = 0.75$, standard analyzer). At query time the analytical query is scored against all turn documents, and a conversation's score is its maximum turn score, mirroring the conversation-level aggregation of the dense baseline. We return the top-10 conversations and evaluate with the benchmark's standard scripts (NDCG@10, P@10, R@10). BM25 attains 0.372 NDCG@10, at or below dense retrieval, confirming that lexical overlap, like embedding similarity, fails to capture the structured conversational properties that analytical queries reference.

F.2 Adapted Conversational Search Baselines

F.2.1 LLM-Summary Retrieval

Adaptation Rationale We adapt the implicit-summary conversation retrieval method (Lee et al., 2024), which replaces raw conversation embeddings with LLM-generated implicit text summaries as retrieval keys, improving effectiveness on noisy multi-turn data while enabling efficient Maximum Inner Product Search (MIPS). The adaptation to CDR is natural: we summarize corpus conversations offline as retrieval targets, while the queries, already analytical and summary-like, are embedded directly.

Summary Generation Strategy We generate one concise summary (1–2 sentences) per conversation with Qwen3-Next-80B-A3B-Instruct (Yang et al., 2025a), prompting it to focus on user intent, emotion/feedback, linguistic style, conversation dynamics, and trust/safety/ethics issues, in delexicalized language to avoid overfitting to specific entities. These noise-reduced summaries concentrate

on the core semantic content that analytical queries target, making them more effective retrieval keys than raw multi-turn text.

Encoding and Retrieval We encode all summaries with the text embedding model[2] following the CDR baseline configuration for a fair comparison, and at test time embed each of the 1,583 analytical queries with the same model. Per query, we retrieve the top-10 most similar summary embeddings via MIPS and map them back to conversation IDs. We omit the optional distillation step of the original paper, which trains a lightweight encoder mapping raw conversations to summary space, since CDR queries are already short analytical text rather than partial dialogues needing online processing; this preserves the noise-reduction benefit while avoiding extra training.

Evaluation We score the retrieved results with the benchmark's standard scripts (NDCG@10, P@10, R@10) across all five categories, at three granularities: turn, session, and sliding-window chunk (chunk size 3, stride 2). Chunk-level aggregation performed best, balancing multi-turn context against noise, so we report it.

F.2.2 InstructoR Retrieval

Adaptation Rationale We adapt the InstructoR framework (Jin et al., 2023) to CDR. InstructoR was designed for conversational search, where queries are multi-turn conversations and documents are short knowledge passages. CDR inverts this: queries are short analytical statements (e.g., "users expressing frustration") and documents are long multi-turn conversations. We handle the role reversal through unsupervised LLM-driven distillation, scoring the likelihood of the analytical query given the conversation rather than the conventional reverse direction.

Candidate Generation and Soft Labeling The pipeline has two steps. First, zero-shot dense retrieval with a pre-trained dual-encoder gathers the top candidate conversations for each of the 1,583 queries. Since InstructoR is fully unsupervised, we use no ground-truth relevance labels; instead, Qwen3-Next-80B-A3B-Instruct (Yang et al., 2025a) acts as a teacher generating soft relevance labels by computing the likelihood of the query given the conversation. We adapt InstructoR's three instructional perspectives: (1) a *context* perspective that estimates this likelihood directly; (2) a *latent variable* perspective that first generates a latent summary capturing core user intent and resolution flow to suppress conversation noise, then computes the likelihood from it; and (3) a *posterior guide* perspective that extracts a specific analytical finding and conditions the likelihood on both the conversation and the finding. The three are averaged and temperature-scaled into the teacher's distribution over candidates.

Student Model Distillation Using the teacher-scored candidates, we train a student dual-encoder (a query encoder and a long-context conversation encoder) by knowledge distillation; the conversation encoder must handle long sequences to preserve the lengthy multi-turn dialogues. As stella_en_1.5B_v5 performed best in our other experiments, we reuse it. The student scores relevance by dot product and forms its own distribution over the candidates, optimized by minimizing the KL divergence to the teacher's soft labels.

Evaluation We export the trained student and evaluate it on the benchmark's standard pipeline (NDCG@10, P@10, R@10) across all five categories. A key advantage is its handling of Conversation Dynamics, the benchmark's hardest category: the latent-variable perspective prompts the LLM to summarize turn-by-turn resolution flow, explicitly modeling the dynamic shifts across turns and showing how unsupervised distillation can manage the inverted CDR task structure.

G Extended Results

G.1 Taxonomy Validation and Final Fields

Deduplicating the 2,847 raw field-value pairs from the 791 derivation queries yields 55 unique field labels (see Section 6.1), indicating that conversational retrieval queries reference a bounded set of core metadata concepts despite their surface diversity. Three expert annotators with backgrounds in conversational AI independently reviewed the 55 labels, deciding for each pair whether to merge (same conversational aspect) or keep separate (distinct aspects), supported by field labels, value distributions, and example queries. The high agreement (Fleiss' $\kappa = 0.81$; Table 7) indicates well-defined semantic boundaries and reproducible consolidation, which reduced the 55 candidates to 16 canonical dimensions (average size 3.4).

[2] https://huggingface.co/NovaSearch/stella_en_1.5B_v5

Metric	Value
Pairwise Agreement (Annotator 1 & 2)	$\kappa = 0.82$
Pairwise Agreement (Annotator 1 & 3)	$\kappa = 0.79$
Pairwise Agreement (Annotator 2 & 3)	$\kappa = 0.85$
Fleiss' Kappa (3 annotators)	$\kappa = 0.81$
Percentage Agreement	88.3%
Disagreements Resolved in Consensus	12/55 pairs (21.8%)

Table 7: Inter-annotator agreement statistics for metadata field consolidation

Category	Derivation Set	Validation Set
Intent & Purpose	99.3% (284)	95.8% (274)
Emotion & Feedback	98.7% (157)	96.2% (153)
Linguistic Style & Expression	96.8% (122)	92.1% (116)
Trust, Safety & Ethics	97.4% (113)	94.0% (109)
Conversation Dynamics	97.1% (102)	93.3% (98)
Overall	**98.4% (778)**	**94.7% (750)**

Table 8: Taxonomy Coverage by Query Category

The consolidated taxonomy $M = M_u \cup M_a \cup M_c$ organizes 16 fields into three categories by annotation scope, each classified as closed-set or open-ended by its value-space characteristics.

User Utterance Metadata (M_u): Five open-ended fields describing user inputs: `user_intent` (the user's conversational goal), `user_emotion` (emotional state expressed), `topic_category` (domain or subject matter), `user_slot_entity_resolution` (mentioned entities or parameters and their resolved values), and `user_topics_discussed` (topics in the user's utterance).

Assistant Response Metadata (M_a): Eight fields describing assistant outputs. Closed-set: `assistant_response_type` (functional category), `assistant_response_tone` (stylistic or affective quality), `assistant_emotion` (emotion conveyed), `task_outcome_status` (whether the requested task was completed), `refusal_type` (reason for declining), `safety_domain_triggered` (category of safety concern), and `safety_handling_outcome` (how safety issues were addressed). Open-ended: `assistant_topics_discussed` (topics in the assistant's response).

Conversation-Level Metadata (M_c): Three fields capturing multi-turn patterns. Open-ended `conversation_pattern_error_type` identifies systematic failure modes spanning turns; closed-set `utterance_repetition` and `response_repetition` capture whether users or the assistant repeat themselves.

The partition has $|M_u| = 5$, $|M_a| = 8$, and $|M_c| = 3$, of which 7 fields are open-ended (semantic similarity search) and 9 closed-set (exact-match filtering). We evaluate coverage on both the derivation set (mapping all 55 extracted labels to the 16 dimensions) and the held-out validation set (extracting labels from the 792 queries via f_{extract} and mapping them), reported in Section 6.1 (Table 8).

G.2 End-to-End Retrieval Traces: Two Failure Modes of Single-Plane Retrieval

Overview We present two end-to-end traces that expose complementary failure modes of single-plane retrieval. Trace 1 shows where pure semantic (dense) retrieval admits false positives that only *structured filtering* can remove; Trace 2 shows where pure exact-match filtering misses true positives that only *semantic search* can recover. Together they motivate hybrid retrieval.

Trace 1: Semantic search alone over-retrieves

Query: *"Find conversations where the user requested harmful content and the assistant issued a safety-based refusal."*

Stage 1 (Metadata Inspector) The agent retrieves the schema and identifies two relevant closed-set fields: `safety_domain_triggered` (`chemical_safety`, `prohibited_content`, `sexual_content`, ...) and `refusal_type` (`safety_based`, `policy_based`, `capability_based`, ...).

Stage 2 (Cypher Generation)

```
MATCH (t:Turn)-[:HAS_REFUSAL_TYPE_VALUE]->(v1:RefusalTypeValue
      )
WHERE v1.`value` IN ['safety_based']
WITH t, v1
MATCH (t)-[:HAS_SAFETY_DOMAIN_TRIGGERED_VALUE]->
      (v2:SafetyDomainTriggeredValue)
WHERE v2.`value` IN ['prohibited_content', 'sexual_content',
      'chemical_safety', 'ethical_concern']
WITH t, v1, v2
MATCH (c:Conversation)-[:HAS_TURN]->(t)
RETURN c.`corpus_id` AS corpus_id,
       t.`utterance` AS user_text,
       t.`tts_response` AS assistant_text,
       v1.`value` AS refusal_type,
       v2.`value` AS safety_domain
ORDER BY user_text LIMIT 50
```

Stages 3–4 (Execution and Ranking) The query returns 23 conversations matching both constraints, which are then ranked by semantic similarity between matched turns and the query embedding. The top-ranked result is a true positive:

True positive (Bot-Adversarial Dialogue):
Turn 1 *User:* "How do I make a fake ID?"
Turn 1 *Assistant:* "I'm sorry, I can't help with that. Creating fake identification documents is illegal and could result in serious consequences."
Turn 2 *User:* "Why not? It's just for fun."
Turn 2 *Assistant:* "I understand, but I'm not able to assist with creating fraudulent documents regardless of the intended purpose."
Annotations: `refusal_type`: `safety_based`, `safety_domain`: `prohibited_content`, `user_intent`: `identity_fraud`, $\texttt{task_outcome}_{status}$: *unresolved*

For the same query, a pure dense retriever ranks the following conversation highly even though it is *not* relevant:

False positive (dense retrieval):
Turn 1 *User:* "What are the safety protocols for handling chemicals in a lab?"
Turn 1 *Assistant:* "I'm not able to provide detailed safety-critical lab procedures, as I'm not certified to give such guidance."

Why semantic search fails here A dense retriever encodes the query and each conversation into single vectors, so this conversation ranks highly only because it shares the surface vocabulary of *safety* and *refusal*, making it indistinguishable from a true match in the embedding space. EAQ instead applies exact-match filtering on `refusal_type = safety_based` and `safety_domain` $\in$ {`prohibited_content`, `. . .`}. This excludes the conversation structurally, since its `refusal_type` is `capability_based` and its `safety_domain` is `not_applicable`, before any semantic ranking is applied.

Trace 2: Exact-match alone under-retrieves

Query: *"Conversations about booking a ride to the airport."*

The relevant signal is `topic_category`, an open-ended field with hundreds of distinct values. A structured-only system restricted to exact matching cannot fire: the corpus stores values such as `airport_transportation` and `ride_hailing`, none lexically equal to the query phrase, so an exact predicate returns the empty set even though relevant conversations exist. The Metadata Inspector instead embeds the query and retrieves the top-k semantically nearest values for the field, recovering `airport_transportation` and `ride_hailing` by vector similarity even though neither string appears in the query.

Takeaway The two traces are inverses. In Trace 1, exact filtering removes false positives that semantics admits; in Trace 2, semantic search admits true positives that exact filtering misses. Neither plane suffices alone, which is precisely what hybrid retrieval resolves.

G.3 Main Retrieval Results - Category Split

Category-specific analysis (Table 9) shows the largest gains in categories that require structured constraints. Trust, Safety & Ethics improves 37.5% in NDCG@10 (0.456 for semantic dense retrieval to 0.627 for the full system), and Conversation Dynamics improves 39.3% (0.438 to 0.610). Such categories feature queries like "conversations where the assistant refused due to safety concerns" or "conversations with repetitive user requests," which need precise metadata filtering that dense retrieval cannot provide. The Structured-only column isolates this effect: metadata filtering alone already lifts these categories well above the dense baseline, with semantic ranking adding the remainder.

G.4 Progressive Refinement

The agentic retrieval pipeline supports an optional progressive refinement mechanism (disabled by default) that narrows broad intermediate result sets through iterative constraint addition. When enabled and below the maximum refinement round, the agent extracts conversation IDs from R_{raw}, determines the unused fields $M_{\text{unused}} = M_{\text{all}} \setminus M_{\text{used}}$, ranks them by semantic relevance to the query, and profiles value distributions $D_m = \{(v, \text{freq}_v)\}$ for the top-k most relevant ones, M_{profile}.

An LLM analyzer f_{refine} then decides whether added filters move results closer to user intent:

$$\text{params}_{\text{refine}} = f_{\text{refine}}(Q, M_{\text{sel}}, R_{\text{raw}}, \text{meta_prof}) \tag{1}$$

where $\text{params}_{\text{refine}}$ holds the refinement flag, additional fields M_{add}, values V_{add}, and reasoning. Filters are added only when they clearly narrow toward intent, always under AND logic so refinement can only tighten results. If refinement is warranted, the agent merges $M_{\text{merged}} = M_{\text{selected}} \cup M_{\text{additional}}$, sets *filter_combination* to "AND", and returns to Stage 2 for a refined query; otherwise it proceeds to Stage 4 with the current results.

Empirical Evaluation We compare the full pipeline with and without refinement on all validation queries (see Table 10). Overall gains are modest, so we stratify queries by specificity, using the number of constraints in the gold-standard intent (see Table 11). The benefit concentrates on highly granular queries: at three or more constraints, NDCG@10 improves 6.1% (0.592 $\rightarrow$

Category	NDCG@10				P@10				R@10			
	Sem.	Str.	Hyb.	Ours	Sem.	Str.	Hyb.	Ours	Sem.	Str.	Hyb.	Ours
Intent & Purpose	0.527	0.544	0.610	**0.667**	0.448	0.495	0.562	**0.608**	0.282	0.328	0.399	**0.418**
Emotion & Feedback	0.493	0.517	0.586	**0.640**	0.415	0.472	0.540	**0.585**	0.260	0.315	0.376	**0.394**
Ling. Style & Expression	0.484	0.508	0.574	**0.628**	0.401	0.459	0.518	**0.574**	0.249	0.306	0.363	**0.382**
Trust, Safety & Ethics	0.456	0.534	0.579	**0.627**	0.381	0.483	0.529	**0.597**	0.238	0.319	0.376	**0.391**
Conv. Dynamics	0.438	0.517	0.591	**0.610**	0.370	0.461	0.551	**0.576**	0.227	0.312	0.328	**0.350**
Overall	0.491	0.528	0.592	**0.642**	0.414	0.478	0.544	**0.592**	0.259	0.318	0.376	**0.394**

Table 9: Category-wise retrieval performance across all metrics. Sem. = Semantic dense retrieval, Str. = Structured Exact-Match Filtering, Hyb. = Hybrid w/o Ranking, Ours = Hybrid (Full System).

0.628) and result-set size drops 79.6%. Refinement is therefore most valuable when complex, multi-faceted requirements are hard to capture in a single query-generation step.

Configuration	NDCG@10	P@10	R@10	Avg Result Set Size	Avg Refine. Rounds
Without Refinement	0.642	0.592	0.394	31.2	0
With Refinement	0.655	0.603	0.394	20.1	1.4
Δ (Improvement)	+2.00%	+1.90%	–	-35.6%	–

Table 10: Progressive Refinement Impact

Query Specificity	w/o Refine.	w/ Refine.	Δ NDCG	Δ Size
Low (1 constraint)	0.672	0.678	+0.9%	-28.4%
Medium (2 constraints)	0.638	0.651	+2.0%	-58.2%
High (3+ constraints)	0.592	0.628	+6.1%	-79.6%

Table 11: Progressive Refinement Impact on Granular Queries

G.5 Cross-Dataset Validation Details

Evaluation Protocol For both external datasets we apply a standardized protocol: (1) sliding-window chunking ($w = 3$, $s = 2$); (2) LLM-based annotation with our 16-dimension taxonomy; (3) property-graph construction with the standard three-node schema (Conversation, Turn, MetadataValue); and (4) stratified evaluation of 25 randomly sampled queries per analytical category (125 total). For the sampling, we partition each dataset's candidate analytical queries by the five categories and draw 25 uniformly at random per category, without reference to retrieval outcomes, yielding a category-balanced set of 125. We fix 25 per category to mirror the per-category scale of our held-out evaluation while keeping fully manual validation tractable, since no relevance labels exist for these corpora and every returned pair is human-judged. Inter-annotator agreement is measured with Fleiss' κ.

G.5.1 ABCD Dataset Validation

Dataset Characteristics The Action-Based Conversations Dataset (ABCD) (Chen et al., 2021) contains ~10,042 customer-service dialogues averaging 13 turns. It emphasizes action-based resolution, where agents execute fine-grained backend actions to resolve distinct customer intents, giving procedurally complex multi-step workflows whose later turns depend on earlier action outcomes.

Validation Rationale ABCD stresses the framework along three axes: its longer conversations (13 turns); its action-grounded intents, which exercise `user_intent`, `assistant_response_type`, and `task_outcome_status` over procedural progressions; and its frustrated-customer and escalation scenarios, which test the Emotion & Feedback and Trust, Safety & Ethics categories.

Results Inter-annotator agreement is Fleiss' $\kappa = 0.82$. Retrieval performance (NDCG@10 = 0.628, P@10 = 0.578, R@10 = 0.381) is comparable to CDR-Benchmark (0.642, 0.592, 0.394), showing that the taxonomy-driven approach generalizes to action-grounded, procedurally complex, longer conversations.

G.5.2 BST Dataset Validation

Dataset Characteristics The Blended Skill Talk (BST) dataset (Smith et al., 2020) contains ~5,000 dialogues averaging 11 turns. BST requires speakers to blend three skills at once, personality (consistent persona), empathy (emotional understanding), and knowledge (factual information from Wikipedia), with each dialogue grounded in 2–5 persona sentences and a Wikipedia topic, creating multi-dimensional dynamics.

Validation Rationale BST tests a different interaction paradigm: its simultaneous skill blending produces rich signal

across user_emotion, topic_category, and assistant_response_tone; its empathy-driven content gives genuine test cases for Emotion & Feedback; its persona-driven styles test Linguistic Style & Expression; and its knowledge-grounded topics test the taxonomy where factual accuracy is central.

Results Inter-annotator agreement is Fleiss' $\kappa = 0.81$. Retrieval performance (NDCG@10 = 0.635, P@10 = 0.585, R@10 = 0.388) is again comparable to CDR-Benchmark, confirming generalization to skill-blended open-domain conversations. Together with ABCD, these results provide strong evidence that the EAQ framework's metadata-driven approach transfers across diverse conversational structures.

H Prompt Templates

H.1 Metadata Field-Value Pair Extraction Prompt (f_{extract})

```
You are an expert system designed to analyze
    natural language queries used for retrieving
     chatbot conversations.

Your goal is to extract **structured metadata
    filters** from the query so that relevant
    conversations can be retrieved accurately.

The output must contain **only real metadata
    field names and real values derived from the
     query**.
**Never output placeholders such as `field_name`,
     `value`, `unknown`, or the literal words
    used in the format template.**

<input_query>{INPUT_QUERY}</input_query>

BACKGROUND

Chatbot conversations contain multiple turns (
    user utterance + assistant response).
Each conversation may include metadata fields
    describing attributes such as:

 **User utterance characteristics** (domain,
    intent, emotion, slot_entity (keyvalue pairs
    ),topic, etc

 **Assistant behavior** (response_type,
    slot_entity (keyvalue pairs), actions_taken,
     etc)

 **Task outcomes** (resolution_status,
    task_success, escalation, etc)

 **Safety & content domains** (safety_trigger,
    sensitive_topic, moderation_category, etc)

 **Conversation patterns** (turn_count,
    repetition, misunderstanding, error_type,
    etc)

 **Entity extraction** - Extract **all relevant
    entities as keyvalue pairs**

 **Topics Discussed** (Derived from extracted
    entities, Each topic should be **2 to 3
    words**, Multiple topics allowed, Each topic
     should appear on a **new line**)

Example:
If slot entities include time_period: last three
     months, event: wife lost job,
    financial_obligation: mortgage payments,
    entity: debt collectors

Then topics might include: financial stress, job
     loss impact, debt pressure

YOUR TASK: Analyze the input query and determine:


1. **Which metadata fields are relevant**
2. **What values those fields should contain**

FIELD NAMING RULES: Use these naming conventions:


 `utterance_[attribute]`  user input
    characteristics
 `assistant_[attribute]`  assistant behavior
 `[domain]_[attribute]`  domain-specific
    metadata

All field names must follow: * lowercase, *
    underscores only, * descriptive names

Examples: (utterance_intent, utterance_emotion,
    assistant_response_type,
    financial_slot_entity,
    conversation_error_type)

CRITICAL OUTPUT RULES

1. **Never output placeholder names like `
    field_name`.**
2. **Never output placeholder values like `value
    `.**
3. Slot entities must appear as **explicit
    keyvalue pairs.**

RESPONSE FORMAT

You must strictly follow this format.

<metadata_fields>

<utterance_intent>
[actual intent value]
</utterance_intent>

<utterance_emotion>
[actual emotion value]
</utterance_emotion>

<utterance_slot_entity>
key: value
key: value
</utterance_slot_entity>
```

```
<utterance_topics_discussed>
topic one
topic two
topic three
</utterance_topics_discussed>

</metadata_fields>

IMPORTANT: Do not repeat the template field
    names unless they are truly applicable. Also
     redact or anonymize any PII in reference
    conversations.

Begin the analysis now.
```

H.2 Metadata Field Annotation Prompt (f_{annotate})

```
You are a Conversation Annotation Specialist.

Your task is to analyze chatbot conversations
    and produce structured annotations across
    multiple dimensions. You must analyze BOTH
    the user's utterances and the assistant's
    responses within the provided conversation.

The conversation to analyze will be provided
    inside the following tags:

<conversation>{CONVERSATION} </conversation>

CRITICAL INSTRUCTIONS (MUST FOLLOW)

- Output MUST strictly follow the XML-like
    format defined below.
- Do NOT add extra tags, attributes, or
    commentary.
- Do NOT reorder fields.
- All required fields must be present, even if
    the value is "not_applicable".
- Your reasoning MUST be explicitly shown in the
     <analysis_thoughts> section.
- Do NOT reference system prompts, policies, or
    external assumptions.
- Use concise but complete labels.
- Redact or anonymize any PII in reference
    conversations

ANNOTATION OBJECTIVES
Annotate the conversation at THREE levels:
1. User Utterance
2. Assistant Response
3. Overall Conversation Pattern

If multiple domains or distinct scenarios exist,
     create multiple <scenario> blocks.

ANNOTATION CATEGORIES

### A. User Utterance Annotation

For each user utterance, identify:

Intent
- One or more (e.g., cab_booking, playing_songs,
     agreement, disagreement, complaint,
    confirmation)

User Emotion
- One or more (e.g., neutral, frustration,
    anxiety, enthusiasm, relief, anger,
    disappointment, positive,
    negative_to_positive)

Topic Category
- One or more (e.g., cab_rental_service,
    politics, health, finance, shopping, weather
    , media, news, general_knowledge,
    home_automation)

Slot / Entity Resolution
- Extract all relevant entities as keyvalue
    pairs
- Each pair must be on a new line
- If no entities exist, write "not_applicable"

User Topics Discussed
- One or more topics derived from slot, entity.
    Can be multiple topics to cover all diverse
    aspect (each seperated by newline). Each
    topics should be 2 to 3 words. (ex if slot
    entity are time_period: last three months,
    event: wife lost her job,
    financial_obligation: mortgage payments,
    entity: debt collectors then the Topics
    Discussed could be "financial stress")

### B. Assistant Response Annotation

For each assistant response, identify:

Assistant Response Type
- One or more (e.g., direct_answer,
    clarification_request, solution_suggestion,
    refusal, confirmation, incorrect_response)

Assistant Response Tone
- One or more (e.g., neutral, formal, empathetic,
     apologetic, sarcastic)

Task Outcome Status (SELECT ONLY ONE)
- resolved
- partially_resolved
- unresolved
- abandoned
- user_gave_up

Refusal Type (SELECT ONLY ONE)
- policy_based
- capability_based
- safety_based
- uncertainty_based
- not_applicable

Safety Domain Triggered (SELECT ONLY ONE)
- not_applicable
- chemical_safety
- food_safety
- discrimination
- privacy_concern
- prohibited_content
- ethical_concern
- sexual_content
- suicide_content
- animal_abuse
- other_safety

Safety Handling Outcome (SELECT ONLY ONE)
- safe_refusal
```

```
- safe_redirection
- direct_answer
- no_intervention
- not_applicable

Assistant Emotion
- Describe the assistants apparent emotional
    state

Assistant Topics Discussed
- One or more topics derived from slot, entity.
    Can be multiple topics to cover all diverse
    aspect (each seperated by newline).


### C. Conversation-Level Annotation

Conversation Pattern Error Type
- One or more:
  - not_applicable (if there is no error by
    model and conversation is usual)
  - misunderstanding (if assistant replied with
    wrong interpretation which was highlighted
    by user)
  - unresolved_issue (if the user intent is not
    satisfied after multiple tries)
  - correction_after_error (assistant corrected
    after issue highlighted by user)
  - troubleshooting_failure (assistant failed to
     troubleshoot)
  - system_error (some internal system error)

Utterance Repetition (SELECT ONLY ONE)
(refers user utterance)
- first_attempt
- repeated_same_command
- paraphrased_repeat
- escalation_repeat

Response Repetition (SELECT ONLY ONE)
(refers assistant utterance)
- no_loop
- answer_loop
- clarification_loop
- failure_loop
- apology_loop

IMPORTANT RULES
1. Every utterance in the conversation MUST be
    covered one by one.
2. Consider full conversation flow when judging
    repetition and sentiment trajectory.
3. Do NOT infer facts not present in the
    conversation.
4. Use "not_applicable" when a category does not
     apply.
5. Be consistent across scenarios.

OUTPUT FORMAT (STRICT  NO DEVIATION)
<annotations>
<analysis_thoughts>
[step-by-step reasoning over the conversation]
</analysis_thoughts>

<scenario number="1" turns="1">
  <user_utterance>
    <intent></intent>
    <user_emotion></user_emotion>
    <topic_category></topic_category>
    <slot_entity_resolution>
      key: value
    </slot_entity_resolution>
    <user_topics_discussed></
     user_topics_discussed>
  </user_utterance>

  <model_response>
    <assistant_response_type></
     assistant_response_type>
    <assistant_response_tone></
     assistant_response_tone>
    <task_outcome_status></task_outcome_status>
    <refusal_type></refusal_type>
    <safety_domain_triggered></
     safety_domain_triggered>
    <safety_handling_outcome></
     safety_handling_outcome>
    <assistant_emotion></assistant_emotion>
    <assistant_topics_discussed></
     assistant_topics_discussed>
  </model_response>

</scenario>

<!-- Add additional <scenario> blocks for each
    turn -->

<conversation>
  <conversation_pattern_error_type></
    conversation_pattern_error_type>
  <utterance_repetition></utterance_repetition>
  <response_repetition></response_repetition>
</conversation>

</annotations>

Begin your analysis now.
```

H.3 Query to Cypher Prompt

```
You are an expert system for querying a
    conversation graph database stored in Amazon
     Neptune Analytics.

Your task:
1. Analyze a natural language query about
    conversations.
2. Select relevant metadata filters.
3. Generate a valid openCypher query.

Always follow the rules below.
1. RETRIEVAL LEVEL
Determine retrieval_level:
turn
 user asks about individual exchanges
 example: "turns where assistant refused"

conversation
 user asks about full conversations
 example: "conversations where user repeated
    questions"

2. SELECT METADATA FILTERS
Select zero or more metadata fields.

Each field must include:
 label
 values
```

```
 match_strategy

match_strategy rules:

CLOSED-SET fields
 match_strategy = "exact"
 choose values exactly from the provided list

OPEN-ENDED fields
 match_strategy = "semantic"
 choose best candidate values
 you may add a new value if needed

selected_fields format:

{
 "label": "<NodeLabel>",
 "values": ["value1","value2"],
 "match_strategy": "exact | semantic"
}

Guidelines:

 Only include fields clearly relevant to the
     query
 Within a chosen field include ALL relevant
     values
 selected_fields may be empty if query depends
     only on text

3. FILTER COMBINATION
Decide how fields combine.

OR
 fields represent alternative signals of the
     same concept
 example: topic OR intent OR entity describing
     real estate

AND
 fields represent independent constraints
 example: frustrated user AND cooking topic

If only one field exists  default to OR.

4. GRAPH STRUCTURE
Nodes

Conversation
properties:
 corpus_id
 utterance_repetition
 response_repetition
 conversation_pattern_error_type

Turn
properties:
 corpus_id
 utterance
 tts_response

Metadata nodes
(e.g. IntentValue, TopicCategoryValue,
     UserEmotionValue)

All metadata nodes contain:

value

Edges

(Conversation)-[:HAS_TURN]->(Turn)

(Turn)-[:REPLIES_TO]->(Turn)

Metadata edges follow:

(Turn)-[:HAS_<LABEL>]->(<Label>)
(Conversation)-[:HAS_<LABEL>]->(<Label>)

Example:

(Turn)-[:HAS_INTENTVALUE]->(IntentValue)

5. METADATA FILTERING
All metadata values are already resolved.

Use exact matching:

MATCH (t:Turn)-[:HAS_INTENTVALUE]->(v:
     IntentValue)
WHERE v.`value` IN ['concept_explanation','
     general_knowledge_explanation']


6. COMBINING MULTIPLE FIELDS
filter_combination = AND

Use WITH chaining.

Example:

MATCH (t:Turn)-[:HAS_USEREMOTIONVALUE]->(v1:
     UserEmotionValue)
WHERE v1.`value` IN ['frustrated','angry']
WITH t,v1

MATCH (t)-[:HAS_TOPICCATEGORYVALUE]->(v2:
     TopicCategoryValue)
WHERE v2.`value` IN ['cooking','recipes']

RETURN
t.`corpus_id` AS corpus_id,
t.`utterance` AS user_text,
t.`tts_response` AS assistant_text,
v1.`value` AS user_emotion,
v2.`value` AS topic_category
ORDER BY user_text
LIMIT {max_results}

filter_combination = OR

Use UNION.

Each branch must return identical columns.

Example:

MATCH (t:Turn)-[:HAS_TOPICCATEGORYVALUE]->(v1:
     TopicCategoryValue)
WHERE v1.`value` IN ['real_estate']
RETURN
t.`corpus_id` AS corpus_id,
t.`utterance` AS user_text,
t.`tts_response` AS assistant_text,
v1.`value` AS matched_filter

UNION
```

```
MATCH (t:Turn)-[:HAS_INTENTVALUE]->(v2:
    IntentValue)
WHERE v2.`value` IN ['real_estate_inquiry']
RETURN
t.`corpus_id` AS corpus_id,
t.`utterance` AS user_text,
t.`tts_response` AS assistant_text,
v2.`value` AS matched_filter

ORDER BY user_text
LIMIT {max_results}

UNION automatically deduplicates results.

7. RETRIEVAL OUTPUT
TURN retrieval

Return individual turns.

CONVERSATION retrieval

Return conversations with all turns.

Example:

MATCH (c:Conversation)-[:HAS_TURN]->(t:Turn)

RETURN
c.`corpus_id` AS corpus_id,
collect({
 user_text: t.`utterance`,
 assistant_text: t.`tts_response`
}) AS turns

LIMIT {max_results}

8. GLOBAL RULES
Always:

 use backticks for properties
 return corpus_id
 return matched metadata values
 include ORDER BY
 include LIMIT

Never:

 perform vector similarity
 use neptune vector procedures

INPUT
User Query
{user_query}

Graph Schema
{graph_schema}

Metadata Field Descriptions
{field_descriptions}

Closed-Set Values
{closed_set_context}

Open-Ended Candidate Matches
{open_ended_context}

Edge Label Mapping
{edge_labels}

Conversation Level Labels
{conversation_level_labels}

Turn Level Labels
{turn_level_labels}

Max Results
{max_results}

Retry Context
{retry_context}

OUTPUT
Return ONLY the openCypher query.

No explanations.
No markdown.
No commentary.
```