

Article

Knowledge Graphs vs. SQL over Structured EHR Data

Leonidas Anagnou ^{1,†}, Andreas Vezakis ^{1,†} , Ioannis Vezakis ¹ , Ioannis Kakkos ^{1,2} , Ourania Petropoulou ¹ and George K. Matsopoulos ^{1,*} 

¹ Biomedical Engineering Laboratory, School of Electrical & Computer Engineering, National Technical University of Athens, 15773 Athens, Greece; ikakkos@biomed.ntua.gr (I.K.); rpetro@biomed.ntua.gr (O.P.)

² Department of Biomedical Engineering, University of West Attica, 12243 Athens, Greece

* Correspondence: gmatsopoulos@biomed.ntua.gr

[†] These authors contributed equally to this work.

Abstract

Clinical question answering over electronic health records (EHRs) increasingly relies on large language model (LLM) agents that retrieve structured patient data through external tools. Published benchmarks, however, evaluate these systems at a single patient-population size, and rarely measure the effect of backend representation from that of the retrieval interface design. This paper compares six retrieval configurations that vary along two axes: backend (a property graph database, a relational database and a dense vector index) and interface design (curated domain-specific tool calls, model-generated queries, full-text search, and single-shot dense retrieval). The evaluation covers a 334-question bank spanning six categories (simple lookup, multi-hop, temporal, cohort, reasoning, and unanswerable), instantiated at three nested population scales: 200, 2000, and 20,000 alive patients from a single Synthea cohort. Four models are compared: Claude Haiku 4.5, Qwen 2.5 72B, Llama 3.1 8B, and Llama 3.3 70B, spanning closed-frontier and open-source alternatives. Curated tool-calling configurations improve accuracy over retrieval-augmented baselines for capable models, but reduce accuracy for a small open-source model due to function-calling protocol failures. We report how accuracy, latency, and cost evolve with each approach, model size, and cohort size, supported by paired statistical tests and confidence intervals. All benchmark components, databases, and evaluation code are publicly available.

Keywords: large language models; LLM; SQL; knowledge graphs; structured data; EHR data; question answering



Academic Editor: Gianluigi Ferrari, Christophe Cruz, Hocine Cherifi and Maria Alice Bertolim Nicoleau

Received: 27 May 2026

Revised: 3 July 2026

Accepted: 13 July 2026

Published: 15 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Artificial intelligence in the form of large language models (LLMs) is rapidly transforming healthcare, with applications spanning clinical decision support, diagnostic assistance, and health informatics [1]. A promising frontier is the deployment of LLM-based autonomous agents that interact with structured clinical data sources, most notably electronic health records (EHRs), to answer complex medical questions on demand [2]. The ability to retrieve, reason over, and synthesize information from EHRs at scale holds considerable potential for supporting clinicians, accelerating research, and enabling real-time population-level insight.

The deployment of LLM-based agents over sensitive patient data also raises questions about privacy preservation and secure data access. Recent work has specifically addressed multi-agent architectures designed for privacy-preserving natural language interaction with FHIR-based EHR systems [3]. De Maio et al. proposed a multi-agent system in which

publicly available LLMs construct FHIR resource URIs while a locally deployed model interprets the retrieved data, thereby preventing direct exposure of patient records to external services. An extended version of this work [4] further introduced a dedicated Data Update Agent supporting real-time record modification alongside a dual-pathway design that separates retrieval from update operations, while maintaining regulatory compliance and fault tolerance across diverse clinical scenarios. These architectures demonstrate that, beyond retrieval accuracy, the design of agentic systems for EHR interaction must account for the data governance constraints inherent to real-world clinical deployments.

Clinical question answering (QA) over EHRs is a demanding task, as it requires not only language understanding but also precise interaction with structured databases containing heterogeneous patient data. Three approaches for this task have emerged. Text-to-SQL approaches leverage the familiarity and ubiquity of relational database systems, translating natural language questions into structured query language statements that can be executed against tabular EHR data [5]. Retrieval-augmented generation (RAG) is an approach that offers a simpler alternative by retrieving relevant document chunks or record excerpts and feeding them directly into the language model's context window, bypassing the need for formal query construction [6]. Knowledge graph approaches represent patient data as interconnected entities and relationships, enabling richer multi-hop reasoning, temporal inference, and cohort-level aggregation through specialized retrieval tools [7]. While all three approaches have demonstrated utility in controlled settings, their relative strengths and limitations, particularly with respect to scalability, are under-explored.

Published benchmarks for clinical QA agents have evaluated these systems at a fixed patient-population size [8], limiting insight into how different retrieval architectures behave as the underlying data grows. This is a critical gap: real-world EHR deployments span populations ranging from small clinical units to regional health systems containing tens of thousands of patients, and the computational and accuracy trade-offs of each retrieval approach may diverge substantially across these scales. Furthermore, most existing evaluations rely on a narrow set of question categories, mostly with reference-based lookup, and do not systematically assess performance across clinically relevant query types [5].

The choice of the model adds another dimension of complexity. The rapid increase in both closed-frontier and open-source LLMs means that the performance of any given clinical QA architecture depends on the underlying model. Yet comparative evaluations across models are rarely conducted under identical experimental conditions, the same question bank, the same scoring protocol, and the same population sizes [8], making it difficult to assess the contribution of the retrieval methodology from that of the language model itself.

This study addresses these gaps by implementing and systematically comparing three different methodologies: a graph-based clinical QA agent against text-to-SQL and RAG baselines under controlled, identical conditions. The evaluation employs a 334-question bank spanning six clinically meaningful question categories, instantiated at three population scales (200, 2000, and 20,000 patients) derived from a single Synthea synthetic cohort [9]. Four models, Claude Haiku 4.5, Qwen 2.5 72B, Llama 3.1 8B, and Llama 3.3 70B, are benchmarked across all methodologies, capturing a range from closed-frontier to open-source alternatives. Results show that the choice of method and model has a large effect on accuracy, while increasing the patient population in the range of 200 to 20,000 patients has minimal effect. Accuracy and latency are reported for each combination of method, model, and cohort size, with statistical tests and confidence intervals throughout.

2. Related Work

2.1. Text-to-SQL

Translating natural language questions into SQL over relational EHRs is the most established approach for structured clinical question answering. Early efforts include TREQS [10], an end-to-end attentive translation model trained on MIMIC-III [11], which established the feasibility of question-to-SQL translation on real EHR schemas. Lee et al. introduced EHRSQL [5], a benchmark of approximately 24,000 question–SQL pairs over MIMIC-III and eICU collected from 222 hospital staff that explicitly includes unanswerable questions and reports execution F1 as the primary metric. Their results expose how fragile schema-grounded query generation remains: even strong language models stay well below human accuracy and fail on aggregation queries. Building on this, Shi et al. proposed EHRAgent [2], an LLM agent that emits Python code (rather than raw SQL) to compose multi-step queries over the same MIMIC tables, reporting a 19.9 percentage-point improvement over SQL-only baselines through tool-aware decomposition. Our sql-t2s method follows the same text-to-SQL pattern but evaluates it head-to-head against five different approaches under matched models, matched scoring, and a publicly reproducible Synthea cohort [9].

2.2. Retrieval-Augmented and Full-Context Clinical QA

A common alternative to formal query generation is to retrieve relevant patient text and place it into the LLM's context window. The earliest large-scale dataset for this regime is emrQA [6], a semi-automatically generated one-million-pair question–answer set over the i2b2 clinical notes corpus, which established the feasibility of large-scale QA over unstructured EHR text. Kweon et al. later constructed EHRNoteQA [12], 962 patient-specific questions over MIMIC-IV [13] discharge notes designed to evaluate LLMs directly; they also validated GPT-4 as a clinical judge with Spearman $\rho = 0.78$ against clinician ratings, providing one of the anchors for the LLM-as-judge methodology we adopted in our paper. More recently, FHIR-AgentBench [8] introduced five agent architectures over MIMIC-IV-FHIR, with 2931 clinician-authored QA pairs, explicitly comparing simple retriever variants, retriever-plus-code, and ReAct configurations. Even their strongest system reached only 50% answer correctness, which underlines how difficult grounded patient-data QA remains for state-of-the-art LLMs. Our two retrieval-style baselines, sql-fts (PostgreSQL full-text search with tf-idf-style ranking over the relational tables) and llm-only (the entire patient JSON snapshot placed in context), instantiate the lower and upper bounds of this approach and let us measure where formal query generation actually pays off.

2.3. Knowledge Graph Approaches and Text-to-Cypher

Graph representations of patient data unlock multi-hop reasoning and cohort-level aggregation that flat retrieval struggles with. The recent GraphRAG framework [14] demonstrated that combining knowledge graph extraction with community-summary retrieval improves long-document question answering, motivating a family of medical extensions. Pengcheng Jiang et al. proposed GraphCare [7], the first system to build personalized per-patient knowledge graphs by distilling external KGs and LLM knowledge, with measurable gains on clinical-prediction tasks. Medical-Graph-RAG [15] extended this idea to evidence-based medical QA via community-summary GraphRAG, applying the GraphRAG methodology to a biomedical corpus. On the query-generation side, Ozsoy et al. released a domain-general Text2Cypher benchmark [16]; but clinical applications remain under-explored. Our graph retrieval methodology replaces ad hoc Cypher generation with a small set of curated function calls (tools), while graph-cypher evaluates the alternative of

letting the LLM compose Cypher directly, so the relative overhead of that abstraction can be measured under identical conditions.

A broader body of KGQA and KG-RAG work provides the methodological context for these clinical extensions. Hu et al. [17] conducted a systematic empirical comparison of nine pre-trained language models on simple KGQA tasks, finding that knowledge distillation and entity-level knowledge enhancement are the most effective strategies for PLM-based graph QA—a finding that motivates our choice to treat model capability as an independent experimental axis. For multi-hop settings, Tan et al. proposed CLRN [18], a cross-lingual reasoning network that decomposes multi-hop KG traversal into sequential one-hop steps without requiring a pre-fused graph; the sequential one-hop decomposition principle is directly reflected in our curated tool-call design for multi-hop clinical queries. On the retrieval side, Talk2Doc [19] showed that combining RAG with a weighted knowledge graph substantially improves patient question answering, reporting strong retrieval metrics on the MedQuAD dataset—lending further support to graph-grounded retrieval over flat document retrieval in clinical settings. Pythia-RAG [20] extends this paradigm to multimodal settings by building a unified knowledge graph from both text and images and retrieving relevant subgraphs via the Prize-Collecting Steiner Tree algorithm, achieving accuracy gains over unstructured RAG baselines and illustrating the generality of structured graph retrieval beyond the unimodal, structured-EHR setting studied here.

2.4. Tool-Calling LLM Agents

Tool-calling agents generalize the patterns described above, in which the model orchestrates external tools (queries, retrievers, validators) to ground its answers. The methodological foundation for this approach is the ReAct framework [21], which interleaves chain-of-thought reasoning with tool-call actions and has become the de facto template for agentic LLM systems. Yixing Jiang and colleagues formalized this for medicine with MedAgentBench [22], comprising 300 tasks over 100 patients in a FHIR-compliant interactive environment. Su et al. introduced KGAREvion [23], a tool-calling agent that verifies LLM-generated triplets against a concept KG. FHIR-AgentBench, compares five tool-calling agent architectures and exposes how brittle multi-turn agentic loops can be on patient-grounded questions. However, the effectiveness of fundamentally different retrieval approaches under matched models and matched scoring has received limited attention, and few studies examine how ranking outcomes vary when the same comparison is conducted across different size cohorts.

2.5. LLM-as-Judge Scoring

LLM-based scoring of free-form responses was established by G-Eval [24] and MT-Bench [25], which showed strong agreement between model judges and human raters on open-ended tasks. In clinical QA, GPT-4-class judges match clinician inter-rater agreement on benchmarks like MedQA [26]. Structured EHR retrieval adds further challenges: the answers may be correct but wrong in units, time-point, or patient ID.

This work (i) compares retrieval methodologies at fixed model and scoring; (ii) evaluates across three cohort sizes (200, 2000, 20,000 patients); and (iii) reports results for three model classes spanning closed-frontier and open-source LLMs.

3. Materials and Methods

Figure 1 summarizes the evaluation pipeline. A 334-question bank is created, at three cohort sizes, with six retrieval approaches operated by four language models. Each (question, approach, model, tier) cell is scored by a primary LLM judge and a secondary deterministic metric before being aggregated and submitted to statistical analysis. All

experiments were orchestrated on an Apple MacBook Pro 14-inch (Apple Inc., Cupertino, CA, USA) with an Apple M4 Pro chip and 48 GB of unified memory, running macOS Tahoe 26.3.1. The relational and graph databases ran locally on this machine; all language-model inference was performed through hosted APIs. Synthetic patients were generated with Synthea 3.4.0 (The MITRE Corporation, Bedford, MA, USA). Structured data were stored in PostgreSQL 16 (PostgreSQL Global Development Group) with GIN tsvector full-text indexes, and in Kùzu 0.11.3 (Kùzu Inc., Waterloo, ON, Canada) for the property graph. The evaluation harness was implemented in TypeScript 5.7 on Node.js 22, using the Model Context Protocol SDK for tool calls. Statistical analyses were performed in Python 3.14 with NumPy 2.4.4, pandas 3.0.2, SciPy 1.17.1, statsmodels 0.14.6, scikit-learn 1.8.0, seaborn 0.13.2, and Matplotlib 3.10.8.

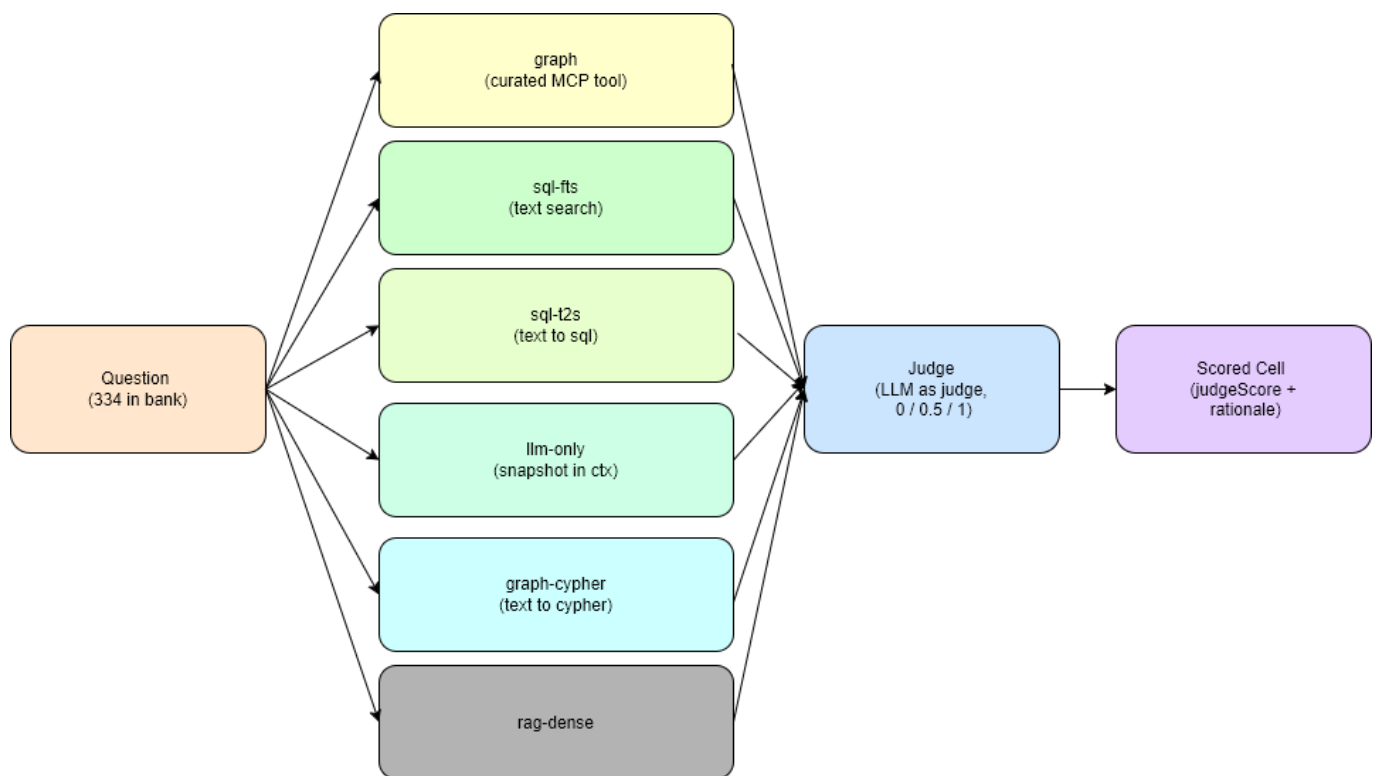


Figure 1. Overview of the evaluation pipeline. The same question bank, the same model, and the same scoring procedure are applied to every retrieval approach and dataset size, isolating the approach and model as the only experimental variables.

3.1. Datasets and Data Representations

All experiments were conducted on synthetic electronic health records produced by Synthea, an open-source generator that simulates the longitudinal life histories of synthetic individuals and emits records in FHIR, C-CDA, and CSV formats. Synthea was chosen over real clinical datasets such as MIMIC because it has no access restrictions, it produces clinically realistic disease modules from published incidence statistics, and its deterministic seeding allows the entire cohort to be regenerated from a single configuration file.

We generated 20,000 synthetic patients with complete clinical histories. From this pool, three subsets with 200, 2000, and 20,000 patients were extracted, with the smaller datasets strictly contained in the larger ones. This ensures that every question whose ground truth is valid at the 200-patient subset remains valid at the 2000-patient and 20,000-patient datasets, so any difference in measured accuracy across scales reflects retrieval behaviour rather than a change in the underlying answer. Table 1 summarizes the datasets created.

Table 1. Counts of clinical resources in the nested Synthea cohort.

Resource	200 Patients	2000 Patients	20,000 Patients
Patients	200	2000	20,000
Encounters	9028	97,091	969,499
Conditions	6297	67,253	657,619
Medications	7196	74,788	696,078
Observations	110,349	1,250,228	11,977,238
Procedures	24,981	276,189	2,739,587
Providers	1183	1183	1183
Organizations	1183	1183	1183

Each dataset was materialized in three storage backends, so that all retrieval methodologies operate over identical scenarios. The three representations are a property graph database, a normalized relational database, and a collection of per-patient JSON snapshots.

3.1.1. Knowledge Graph

The graph representation is implemented in Kuzu [27], an embedded property-graph database with a Cypher-compatible query language. Rather than instantiating every clinical concept on a per-patient basis, the schema separates shared concepts from patient-specific occurrences. Each distinct condition, medication, observation type, and procedure appears once as a concept node carrying its SNOMED, RxNorm, or LOINC code, and per-patient instances of that concept link to it through dedicated relationships. This shared-concept design reduces total node count by roughly two orders of magnitude relative to a naïve per-patient duplication, and lets the LLM reason about populations by traversing edges from a concept rather than aggregating across redundant copies. At the 20,000-patient dataset the graph contains 305 condition concepts, 356 medication concepts, 297 observation concepts, and 412 procedure concepts, against more than 11.9 million observation instances and 657,619 condition instances.

3.1.2. Relational Database

The relational representation is implemented in PostgreSQL and consists of nine normalized tables that mirror the entities of the graph. Foreign keys connect every clinical record to its patient and, where applicable, to the encounter in which it was recorded, and B-tree indexes are defined on the join columns most frequently used by analytic queries. Two variants of the database were prepared from the same ingest pipeline, with identical row counts and identical content. The first variant supports the text-to-SQL methodology and exposes only the base tables and indexes described above. The second approach additionally provisions PostgreSQL full-text-search machinery for the keyword-retrieval approach: a tsvector column is added to each clinical table, populated from the relevant free-text fields such as description, code, and name, indexed with a GIN index. Both variants are loaded by a common ingest routine that derives canonical numeric values for laboratory observations using a curated LOINC normalization registry, so that range-based questions can be answered consistently across methodologies.

3.1.3. Patient Snapshots

The third representation collects, for each patient, the entirety of that patient's record into a single JSON document. The document contains the patient demographics, all encounters in chronological order, and, nested under each encounter, the conditions, medications, observations, and procedures recorded during it. This constitutes the input to the context-only methodology, in which the relevant patient or patient set is appended in full to the

language model's context window without any intermediate retrieval, and it is also used by the question-generation pipeline to derive answers and support record identifiers.

3.1.4. Question Bank

A set of 334 questions was created to test retrieval behaviour across the clinically relevant scenarios. The bank covers six categories: simple-lookup questions that require returning a single fact about a single patient (72 items); multi-hop questions that require chaining two or more clinical relationships (58 items); temporal questions that require ordering of events or interval reasoning (56 items); cohort questions that require identifying or counting patients matching combined criteria (60 items); reasoning questions that require integration of clinical context beyond a single field lookup (60 items); and unanswerable questions whose requested information is absent from the cohort (28 items). Questions were generated programmatically from per-category templates instantiated against the JSON ground truth, then filtered manually to remove items whose phrasing was ambiguous, or whose answer space was trivial. Each question is anchored to one or more patient identifiers and carries the supporting record identifiers used to derive its expected answer, which simplifies both manual curation and automated scoring. The same set is administered at every tier; because the smaller ones are strict subsets of the larger one, the expected answer for a given question is identical across tiers when the patients it references are present. Full details of the bank's design, provenance, category definitions, and clinician review are provided in Supplementary S2.

3.2. Question Answering

Each of the 334 questions in the set is administered to six retrieval approaches operated by four language models, resulting in $334 \times 4 \times 3 \times 6 = 24,048$ question instances. The retrieval approaches differ exclusively in the interface through which the language model can access the patient data, while the data themselves, the retrieval method, and the generation settings are held constant.

3.2.1. Retrieval Methodologies

The graph approach equips the language model with eighteen tools that expose the patient knowledge graph through clinically meaningful operations, such as patient lookup, domain-scoped record retrieval, aggregation, and temporal comparison, rather than requiring the model to write raw graph queries.

The graph-cypher methodology uses a single tool that executes Cypher queries and returns the result rows. The system prompt for this approach describes the graph schema, the available node and relationship labels, and a small number of query patterns; the language model is responsible for planning and executing every query itself.

Two of the approaches query a standard relational database (PostgreSQL). The first, sql-t2s, gives the model the database schema and asks it to write SQL queries to answer each question. The second, sql-fts, is nearly identical but uses a version of the database optimized for full-text search, and the model is explicitly encouraged to use keyword-style queries where appropriate. Both approaches share the same safety rules: write operations are blocked, results are capped at a maximum row count, and queries that run too long are cancelled.

The fifth approach, llm-only, removes all retrieval tools and instead pastes the relevant patient data directly into the prompt as text. For questions about a single patient, that patient's full record is included. For questions about a group of patients, records are concatenated until the context window is full. This represents the baseline case where the model reads patient data directly, with no structured retrieval.

As a dense-retrieval baseline we add rag-dense, a single-shot vector RAG approach. Each patient record is split into typed section chunks (demographics, conditions, medications, observations grouped by laboratory code with the most recent values, procedures, and encounters). Every chunk is then embedded into a 768-dimensional vector with the general-purpose nomic-embed-text model and L2-normalized in a per-tier index. At query time the question is embedded with the same model, cosine similarity is computed against the chunks of the anchored patient, and the eight most similar chunks are concatenated into the prompt with the question; the model then answers in a single pass with no tools. Scoping is identical to the lexical sql-fts baseline: patient-specific questions retrieve only within the anchored patient, and cohort questions, which have no single patient, receive the same cohort listing that sql-fts uses. rag-dense therefore differs from sql-fts principally in the retrieval mechanism (dense vector similarity rather than lexical full-text search), though it also retrieves a fixed set of chunks in a single pass rather than issuing model-driven queries; retrieval therefore remains the primary experimental variable.

Several design choices bound this baseline, and we state them so the result is read correctly. The retrieval depth ($k = 8$ chunks) is matched to the sql-fts context size for a controlled comparison rather than tuned; the embedder is general-purpose rather than clinical; and observations are chunked by code with only the most recent values. To separate the contribution of retrieval from that of the language model, we additionally evaluate a language-model-free retrieval baseline, a BM25 ranker and a BM25–dense reciprocal-rank-fusion hybrid, reported in Supplementary S1; Figure S1 shows retrieval-level answer containment by question type.

3.2.2. Models

Four language models were used: Claude Haiku 4.5 (Anthropic) [28], Qwen 2.5 72B, [29] Llama 3.1 8B and Llama 3.3 70B [30]. The four models span a closed-frontier small model, two large open-source models, and a small open-source model, and were chosen so that observed differences across methodologies could be attributed to the language model's capability for structured retrieval rather than to incidental differences in availability or deployment.

3.2.3. Generation and Agent Settings

All experiments used identical generation settings across models and retrieval approaches. The maximum response length was set to 4096 tokens. For the four tool-based approaches, the model was allowed up to 20 back-and-forth tool calls per question before being forced to give a final answer. Questions rarely needed more than 10 tool calls. In the llm-only setting, the model answers in a single turn since it has no tools to call.

Every approach had its own system prompt describing the tools, schema, and expected answer format, but these prompts were kept identical across models and dataset sizes.

3.3. Evaluation

Every (question, model, cohort, approach) cell produces a structured record that pairs the model's natural language answer with the question's expected answer, together with the latency of the run and the number of language-model turns. These records are then scored along three complementary axes: a correctness score produced by a large language model acting as a judge, a secondary token-overlap score retained for transparency, and a set of operational metrics describing the latency and effort of obtaining each answer.

3.3.1. LLM-as-Judge Agreement

The primary correctness metric is a three-level graded score assigned by a language-model judge. A score of 1.0 indicates that every key fact in the reference answer is also

present in the model's answer, regardless of phrasing, date format, or surrounding clinical context. A score of 0.5 indicates partial correctness, applied in three recurring situations: the key data are present but the qualitative assessment or trend descriptor is incorrect, some items of a list-valued answer are correct while others are missing or wrong, or the answer is verbose but ultimately lands on the right facts. A score of 0.0 indicates that the answer is wrong, hallucinated, empty, unparsable, or refuses an answerable question.

Two special flags can be raised during evaluation. The first is set when the judge believes the reference answer itself is wrong, most often because a question is ambiguous about whether it asks for a record count or a patient count. These cases are excluded from the main accuracy figures and flagged for human review. The second is set when the model's answer contains the correct information but also includes extra content that, while not contradictory, goes beyond what was asked. These answers still receive full marks since the rubric only penalizes contradictions, but the flag is retained for qualitative error analysis.

The judge model throughout was Claude Haiku 4.5. Its system prompt is stored in a single authoritative document and loaded automatically at evaluation time, ensuring the prompt in use can never silently drift from the documented version. The rubric was validated by working through seventeen borderline cases in which the human authors and the model agreed on the correct score; these cases are released alongside the rubric so reviewers can audit the calibration decisions directly (Supplementary S3). To guard against self-preference bias, the scoring prompt was designed to be blind to system identity: it presents only the question, the reference answer, and the candidate response, with no information about which retrieval approach or model generated the response. A sample of responses scored as incorrect were also reviewed manually by the authors to confirm the judgements, and no systematic pattern of false negatives was found.

3.3.2. Deterministic Reference Scoring

A secondary score is computed independently of the language-model judge using deterministic word and number matching, with the method varying by answer type. This score is not used as the primary outcome since it cannot handle paraphrasing or clinically equivalent reformulations, but its independence from the judge makes it a useful cross-check on the judge's reliability.

3.3.3. Statistical Analysis

In order to test whether the six approaches differ in accuracy, two statistical tests are run on the scores: Friedman's test [31] on the raw judge scores, and Cochran's Q [27] test on a binarized pass/fail version of the same scores. The binarization is applied at two cutoffs: strict (only fully correct answers pass) and lenient (partially correct answers also pass), to check whether conclusions are sensitive to where the threshold is drawn.

When a test finds significant differences across approaches, all pairs of approaches are compared directly using Wilcoxon signed-rank tests on the raw scores and McNemar's test [28] on the binarized scores, with Cliff's δ [29] effect sizes reported alongside p -values to indicate the magnitude of each difference. McNemar's test is switched to its exact binomial form when the number of discordant pairs is small. All pairwise p -values are adjusted for multiple comparisons using Bonferroni correction [32].

Cross-model comparisons follow the same template: within each approach and dataset size, Friedman's test is applied across the four models, paired by question, to test whether model choice affects accuracy. Confidence intervals around mean accuracies are estimated by stratified bootstrap resampling with 10,000 replicates. To test whether accuracy changes with cohort size, the per-question score is regressed on the log of the cohort size for each

model and approach combination, and the slope coefficient is reported with its magnitude and significance.

4. Results

4.1. Overall Accuracy

Figure 2 shows the mean judge score of each retrieval methodology under each model, pooled across the three tiers and with 95% bootstrap confidence intervals. Table 2 identifies, for each (model, cohort) pair, the methodology that achieved the highest mean judge score, together with that method's score and the pooled mean across all six approaches in the same block.

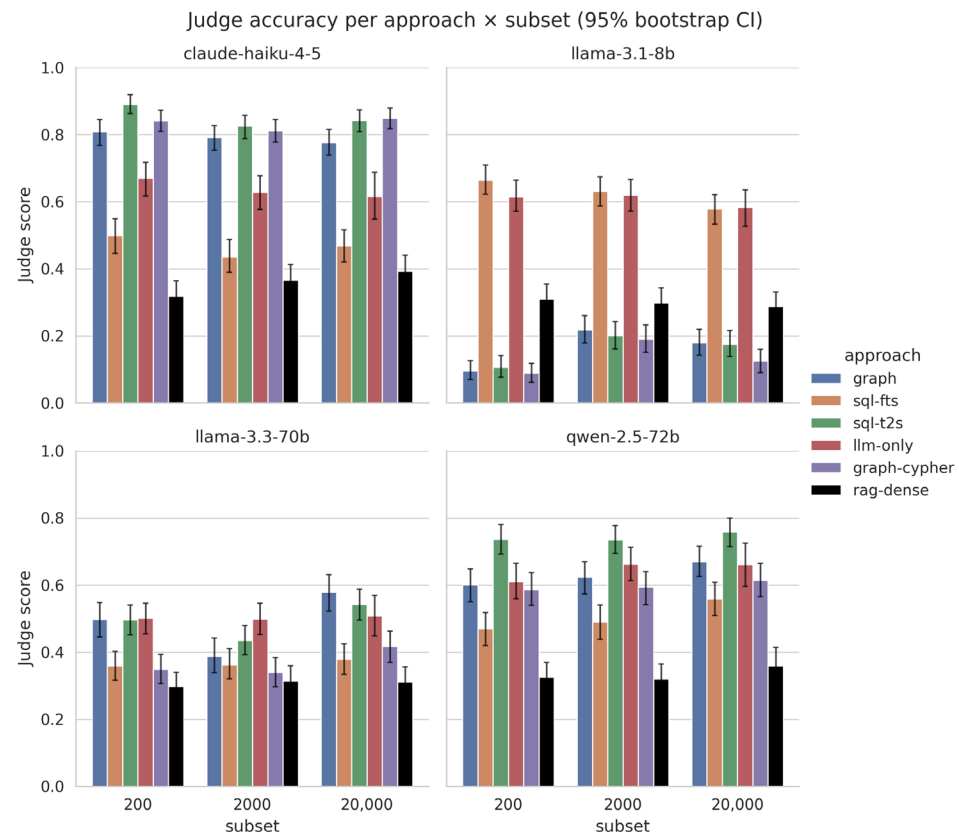


Figure 2. Mean judge score per approach and model, pooled across the three subsets. Error bars are 95% bootstrap confidence intervals stratified by question identifier.

Table 2. Best-performing retrieval approach for each (model, subset) pair.

Model	Subset	Best Approach	Best Score	Pooled Score
Claude Haiku 4.5	200	sql-t2s	0.885	0.736
Claude Haiku 4.5	2000	sql-t2s	0.806	0.690
Claude Haiku 4.5	20,000	graph-cypher	0.837	0.715
Qwen 2.5 72B	200	sql-t2s	0.736	0.600
Qwen 2.5 72B	2000	sql-t2s	0.730	0.619
Qwen 2.5 72B	20,000	sql-t2s	0.766	0.655
Llama 3.1 8B	200	sql-fts	0.676	0.304
Llama 3.1 8B	2000	sql-fts	0.646	0.366
Llama 3.1 8B	20,000	sql-fts	0.592	0.314
Llama 3.3 70B	200	llm-only	0.502	0.436
Llama 3.3 70B	2000	llm-only	0.498	0.404
Llama 3.3 70B	20,000	graph	0.579	0.477

4.2. Statistical Significance of Retrieval Approaches

Figure 3 shows the pairwise Wilcoxon signed-rank tests between every pair of approaches within each (model, tier) block, paired by question identifier. The heatmap cells encode the Bonferroni-corrected p -value.

Pairwise approach differences in mean judge score, pooled across subsets

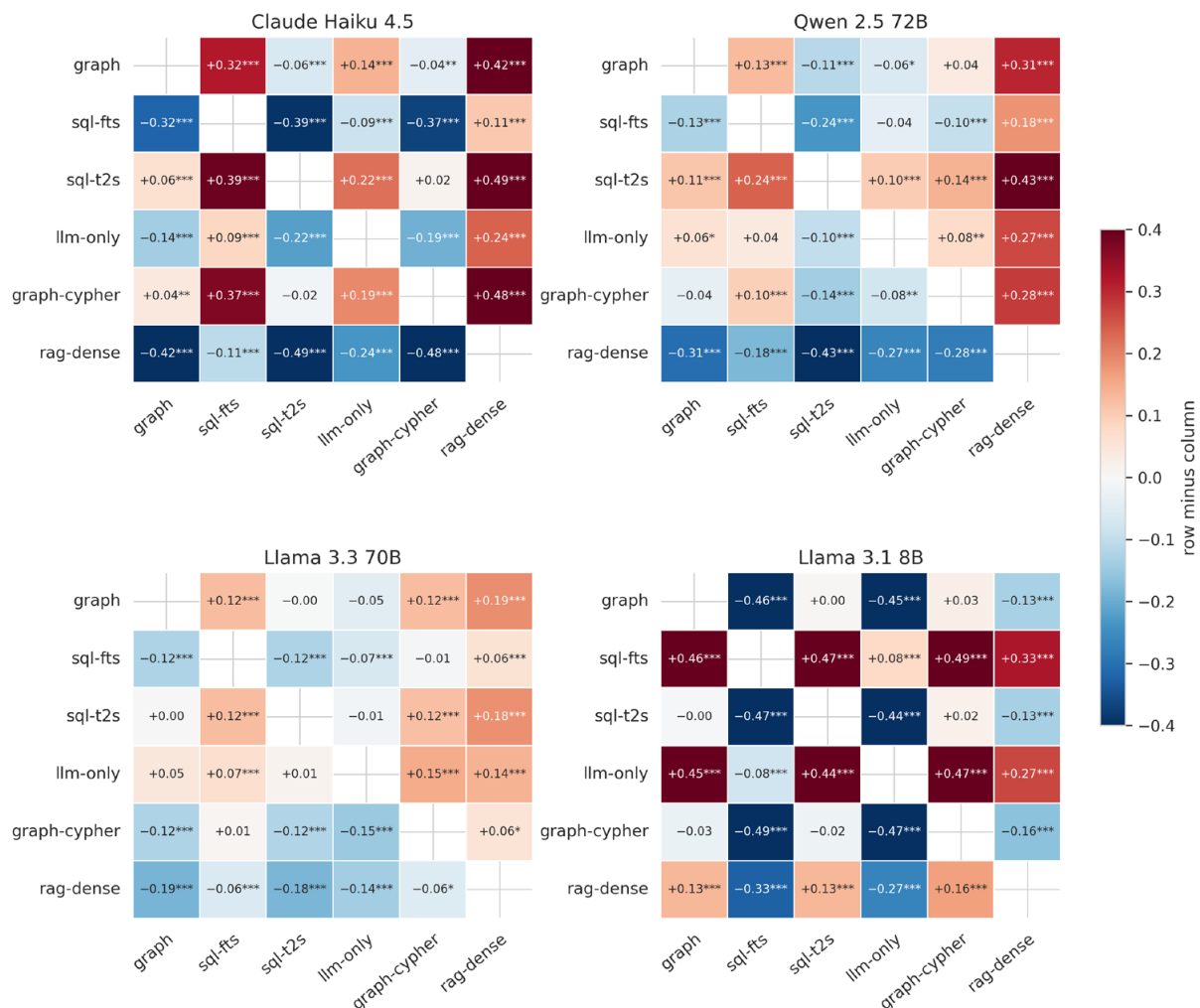


Figure 3. Pairwise approach differences in mean judge score, pooled across the three subsets (200, 2000, 20,000 patients), per model. Cell value is row – col; positive (red) means the row approach scored higher than the column approach on the same questions. Stars denote Bonferroni-adjusted Wilcoxon signed-rank significance at $p < 0.05$ (*), $p < 0.01$ (**), $p < 0.001$ (***)

4.3. Performance by Question Type

Figure 4 presents the mean judge score along the question-type axis, producing one heatmap per model with rows for the six question categories and columns for the six retrieval approaches. Cell colour encodes mean accuracy.

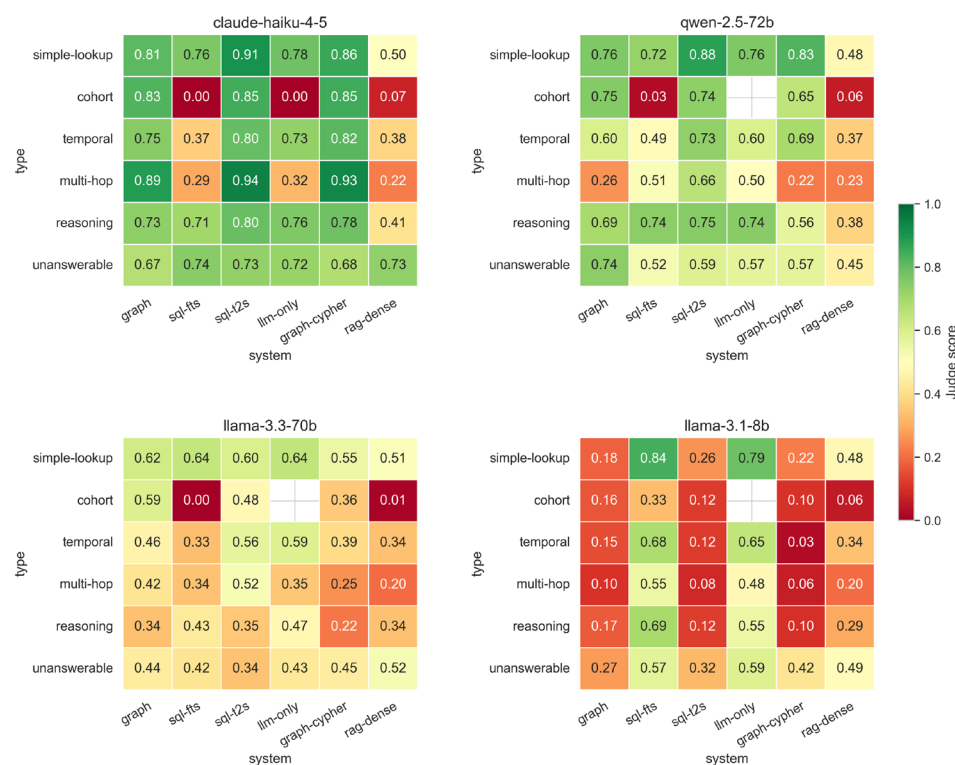


Figure 4. Mean judge score per question type and approach, one panel per model. Cell colour encodes accuracy on the [0, 1] scale.

4.4. Performance Across Cohort Sizes

Figure 5 plots mean judge score against subset size on a logarithmic horizontal axis, with one curve per approach and one panel per model, and 95% bootstrap confidence intervals at each tier.

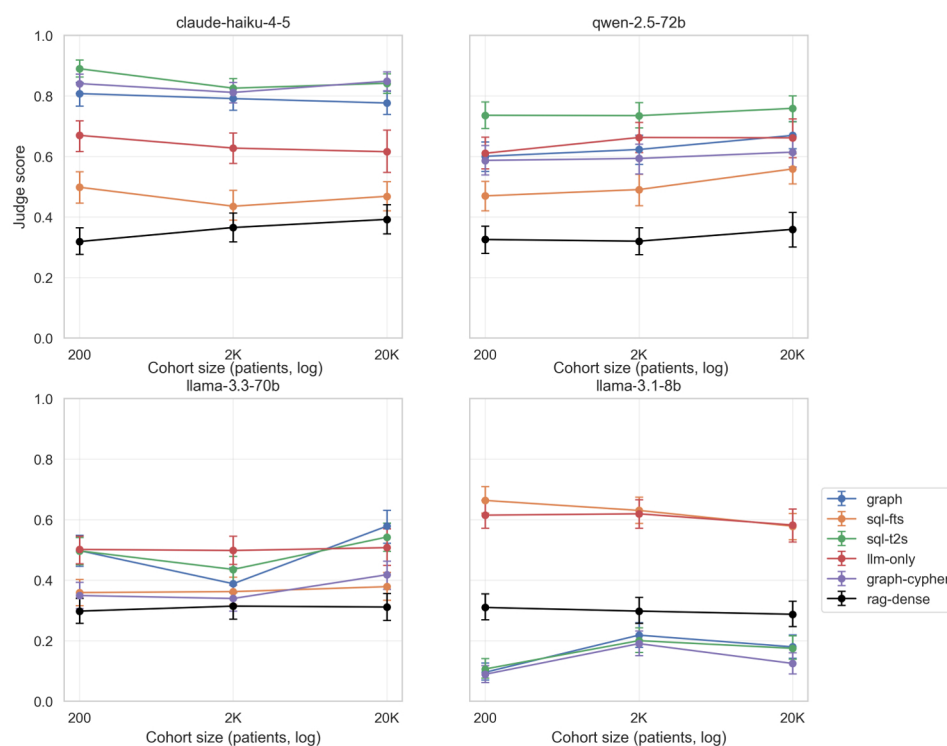


Figure 5. Mean judge score as a function of cohort tier, per approach and model. Cohort size is shown on a logarithmic axis; shaded bands are 95% bootstrap confidence intervals.

The largest absolute slope is 0.042 per log-decade, well below practical concern. Full per-cell regression coefficients are in Table S4.

4.5. Retrieval Gains

Figure 6 compares, for each model, the mean judge score of the tool-using approaches (graph, graph-cypher, sql-t2s) against the context-only baseline (llm-only), quantifying how much the model gains or loses from being given retrieval tools.

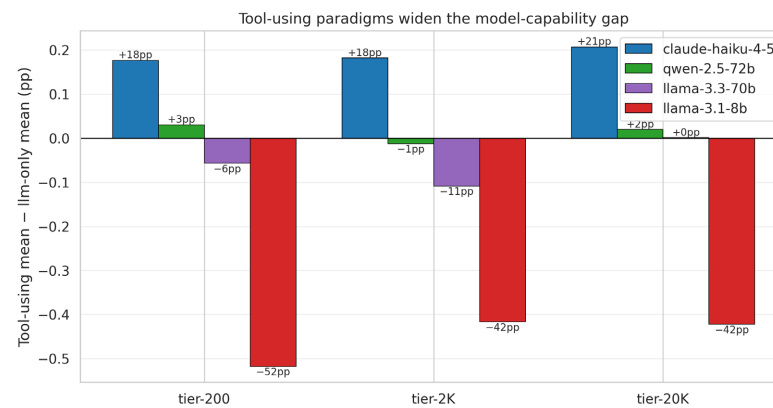


Figure 6. Mean judge score of the best tool-using approach minus the mean judge score of llm-only, per model, expressed in percentage points (pp). Positive bars indicate that tool use improves accuracy; negative bars indicate the opposite.

4.6. Latency and Failures by Approach and Model

Figure 7 shows the distribution of end-to-end latency per question, per (model, method), as a box plot on a logarithmic axis. Figure 8 shows the percentage of attempted cells that failed for each (model, method), split by failure type.

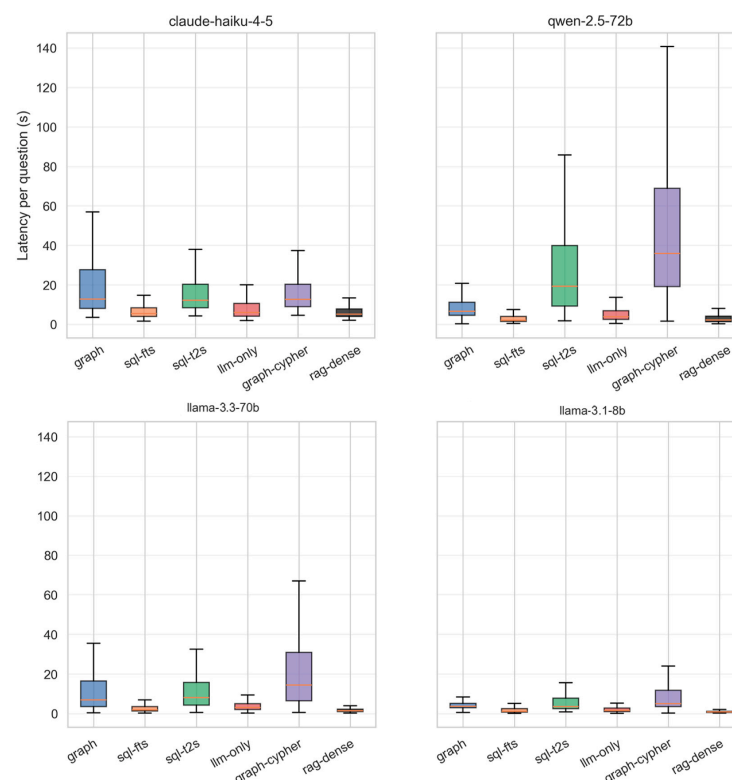


Figure 7. End-to-end latency per question, by approach and model. Boxes mark the interquartile range with the median; whiskers extend to the 5th and 95th percentiles. The horizontal axis is logarithmic.

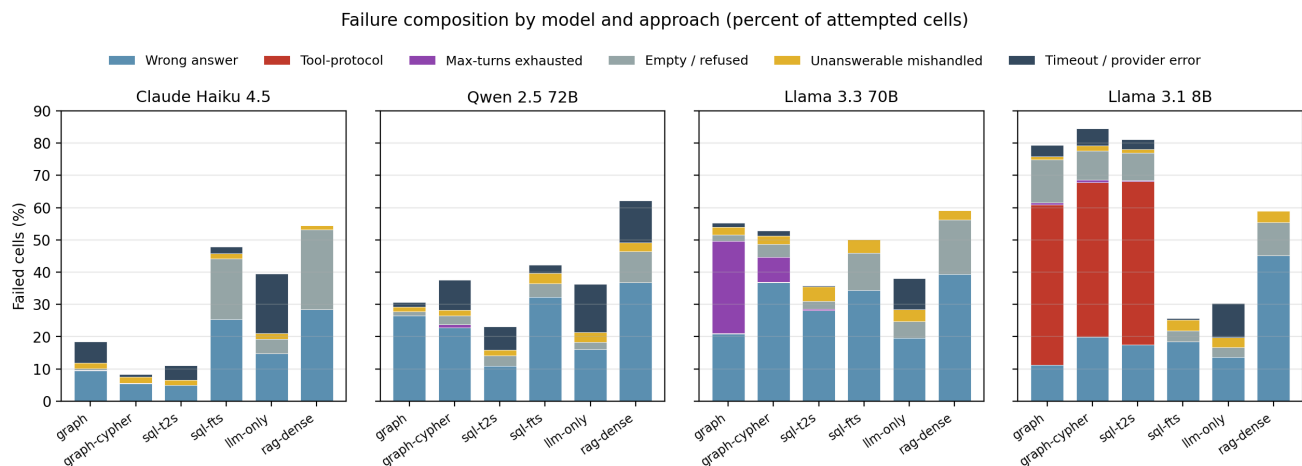


Figure 8. Each bar is the percentage of attempted cells that failed, split by failure type. A cell counts as failed if it errored or received a judge score of zero. Tool-protocol means the model emitted a tool call as its final answer instead of placing it in the structured tool channel. Claude Haiku 4.5 and Qwen 2.5 72B fail mostly by wrong answer, tallest on sql-fts, and never by tool-protocol. Llama 3.1 8B fails by tool-protocol on about half of all agentic cells. Llama 3.3 70B removes the tool-protocol failure but exhausts the turn budget on the graph approach.

4.7. Cost by Approach and Model

Most APIs of the LLMs are not publicly available and require payment per request and tokens spent. We present in Table 3 the cost per 334-question run and in Table 4 the cost per correct answer (in USD). These are bound to change since the cost is changing all the time. Cost per run divided by the mean judge score times 334. Within a model the agentic methods cost more per question, but their higher accuracy keeps the cost per correct answer close to the single-shot methods. For Claude Haiku 4.5 the cheapest method per correct answer is llm-only at 0.023 and the most expensive is graph at 0.064. Llama 3.1 8B is zero because it can be run locally.

Table 3. Cost per 334-question run in USD.

Method	Haiku 4.5	Qwen 72B	Llama 3.3 70B	Llama 3.1 8B
Graph	16.85	1.05	2.17	0
Graph-cypher	14.68	1.45	2.35	0
Sql-t2s	13.31	1.41	1.26	0
Sql-fts	5.54	0.24	0.38	0
llm-only	4.86	0.48	0.75	0
rag-dense (est.)	4.50	0.40	0.60	0

Table 4. Cost per correct answer in USD.

Method	Haiku 4.5	Qwen 72B	Llama 3.3 70B	Llama 3.1 8B
Graph	0.064	0.005	0.013	0
Graph-cypher	0.053	0.007	0.019	0
Sql-t2s	0.047	0.006	0.008	0
Sql-fts	0.036	0.001	0.003	0
llm-only	0.023	0.002	0.004	0
rag-dense (est.)	0.035	0.004	0.006	0

5. Discussion

5.1. Retrieval Benefit Depends on the Model

The most consequential finding of this study is the interaction between model capability and the value of structured retrieval. Figure 6 shows that the gain from equipping

a model with retrieval tools is not a property of the data or the approach used, but of the model's capacity to operate the tool interface reliably. For Claude Haiku 4.5 the best agentic approach exceeds the llm-only baseline by roughly 0.21 in mean judge score (0.84 against 0.64 pooled across tiers), and for Qwen 2.5 72B the gain is smaller but still consistently positive. For Llama 3.1 8B the relationship reverses: the graph and graph-cypher approaches collapse to judge scores between 0.08 and 0.22, well below the same model's 0.62–0.66 llm-only baseline. The Llama 3.1 8B runs on agentic approaches which exhibit unusually long turn distributions terminating in max-turns failures rather than final answers, and inspection of the raw transcripts shows the model emitting raw JSON rather than issuing a structured tool call, hallucinating tool names absent from the schema, or repeatedly invoking the same tool with identical arguments. Llama 3.3 70B occupies an intermediate position: it sustains the agentic approaches without the 8B's collapse (graph and sql-t2s both near 0.48 pooled), yet its gains over its own llm-only baseline are small and category-dependent, with no single approach dominating. The implication is that structured retrieval is not universally beneficial: it amplifies the gap between strong and weak models rather than narrowing it, and a benchmark that fixes the model while varying only the retrieval approach will systematically underestimate the difficulty of small-model deployment. This echoes findings in medical image analysis, where smaller deep neural networks have achieved competitive or superior performance over larger architectures [33,34], suggesting that the relationship between model capacity and task performance is non-trivial across medical AI domains.

5.2. Retrieval Approach Comparison

Among the four tool-using approaches, text-to-SQL emerges as the strongest default for the two capable models. Table 2 shows sql-t2s winning most (model, cohort) scenarios, including all instances with Qwen and the two smaller cohorts with Claude. Graph-cypher performs the best with the 20,000-patient cohort for Claude. Llama 3.1 8B performed the best for all cohorts with the sql-fts approach.

The pairwise Wilcoxon heatmap in Figure 3 confirms that the sql-t2s advantage over graph and llm-only is large and statistically significant when it comes to Claude and Qwen models, while its advantage over graph-cypher is small and significant only in some cells. Figure 4 shows that performance varies meaningfully across question categories. For the two capable models, sql-t2s posts the highest score on most categories—simple lookup, multi-hop, and reasoning, and it ties the curated graph approach on cohort—while graph-cypher leads on temporal and sql-fts on unanswerable.

5.2.1. Simple Lookup

sql-t2s achieves the highest scores for capable models (Claude: 0.91, Qwen: 0.88), reflecting that single-table exact-value retrieval is precisely what relational databases are optimized for. Graph traversal offers no structural advantage here. Llama 3.1 8B's best score is 0.84 (sql-fts), but then accuracy collapses with other agentic approaches (0.18–0.26). The single-shot rag-dense baseline scores 0.48–0.51 uniformly across all four models, and for Llama 3.1 8B it nearly doubles sql-t2s (0.48 vs. 0.26), indicating that dense retrieval is more robust than keyword search when the model cannot drive an agentic loop; it nonetheless trails sql-t2s for the capable models. Llama 3.3 70B clusters tightly across approaches (0.51–0.64), with the context-only llm-only baseline (0.64) matching the best tool-based result, so structured retrieval offers it no advantage here.

5.2.2. Multi-Hop

sql-t2s again leads for the larger models (Claude: 0.94, Qwen: 0.66); multi-table JOINS cover the same relational traversal as graph hops over structured data. The gap between

Claude and Qwen likely reflects stronger SQL generation. Llama 3.1 8B reaches only 0.08, making multi-hop the category where small-model collapse is most severe. rag-dense scores 0.20–0.23 across models—below sql-t2s (Claude 0.94) but, for Llama 3.1 8B, more than double the best agentic result (0.20 vs. 0.08), since single-pass dense retrieval sidesteps the tool-protocol failures that sink the small model on multi-step queries. Llama 3.3 70B follows the capable-model pattern here, with sql-t2s its strongest approach (0.52), well ahead of the single-shot baselines.

5.2.3. Temporal

graph-cypher is the strongest approach for Claude (0.82), as Cypher’s native date-ordering and time-interval syntax maps more directly onto temporal questions than SQL’s equivalent constructs. Qwen reaches 0.73 with sql-t2s. Llama 3.1 8B’s best score is 0.68 with sql-fts. rag-dense scores 0.34–0.38 and, like the other single-shot baseline, varies little across models; a fixed top-k retrieval lacks the explicit date-ordering that makes Cypher effective here, so it trails graph-cypher for the capable models. For Llama 3.3 70B the llm-only baseline (0.59) edges sql-t2s (0.56), and graph-cypher (0.39) shows none of the temporal advantage it gives Claude.

5.2.4. Cohort

sql-t2s and graph approaches are almost tied for capable models (Claude: 0.85/0.83; Qwen: 0.74/0.75), with graph-cypher well below for Qwen. The dedicated population-level functions exposed as graph tools spare the model from writing complex multi-table aggregations, explaining why the gap between approaches is narrower here than on other categories. Llama 3.1 8B scores 0.10–0.33 regardless of approach. rag-dense is the weakest approach in this category, collapsing to 0.01–0.07. The curated graph approach is strongest for Llama 3.3 70B as well (0.59, ahead of sql-t2s at 0.48), reinforcing the value of dedicated population-level tools on cohort questions.

5.2.5. Reasoning

sql-t2s leads for capable models (Claude: 0.80, Qwen: 0.75). The inferential step is handled by the LLM after retrieval, so the deciding factor is which backend delivers the cleanest underlying fact. SQL’s exact matching has the edge. Llama 3.1 8B reaches 0.12 with sql-t2s. rag-dense scores 0.29–0.41 (Claude 0.41), well below sql-t2s but degrading only modestly for the small model (Llama 3.1 8B: 0.29), again reflecting its insensitivity to model capability. Llama 3.3 70B departs from the capable-model pattern on reasoning: its best result is the llm-only baseline (0.47), with sql-t2s reaching only 0.35, indicating the SQL backend does not aid its post-retrieval inference.

5.2.6. Unanswerable

This is the only category where llm-only is on top of the best approaches for both Claude (0.74) and Llama 3.1 8B (0.57). A tool that explicitly returns no results gives the model an unambiguous signal to decline; sql-t2s requires the model to independently interpret a zero-row result, leading to more false positives. These questions require no complex retrieval, only the absence of a result. Here rag-dense matches lexical retrieval (Claude 0.73 versus sql-fts 0.74) and stays comparatively high for the small model (Llama 3.1 8B: 0.49, Llama 3.3 70B: 0.52), confirming that dense and lexical retrieval are essentially equivalent at signalling that no answer exists. Llama 3.3 70B is the one case where dense retrieval leads outright—rag-dense (0.52) tops every other approach, and graph-cypher (0.45) edges lexical sql-fts (0.42).

5.3. Cohort-Size Effects

No approach degrades meaningfully as the cohort grows from 200 to 20,000 patients. The few statistically significant slopes are inconsistent in direction with no clear pattern, which suggests they reflect noise rather than a genuine scaling trend. Cohort size has no meaningful effect on accuracy within the range tested (200–20,000 patients). This conclusion should not be extrapolated to the scales typical of regional or national EHR systems, which can span one to ten million patients. At those scales, failure modes absent from this evaluation may emerge: query timeouts on large relational tables, index performance degradation under high-cardinality joins for SQL-based approaches, and memory pressure during multi-hop graph traversals for knowledge graph approaches. Future works should test those approaches at millions of patients.

It is also worth noting that graph-cypher takes the top spot for Claude Haiku 4.5 at the 20,000-patient dataset, while sql-t2s performs better at smaller cohort sizes. This hints that graph-based approaches may become more competitive as the dataset grows.

5.4. Latency and Deployment Tradeoffs

Latency is driven by the choice of approach, not by how many patients are in the cohort. Figure 7 shows that sql-fts is the fastest approach across all four models, with median latencies between 1 and 7 s. The approaches that involve multi-step planning take between 12 and 20 s for Claude and between 6 and 36 s for Qwen. These are acceptable latencies for research and prototyping contexts, but would be too slow for latency-sensitive applications. Qwen on graph-cypher stands out as a problem: a median of 35.9 s and a mean of 53.8 s, the latter driven by the model repeatedly generating Cypher that the database rejects and the agent retrying until it runs out of turns.

Within the scope of this benchmark evaluation, sql-t2s achieves the highest overall accuracy for capable models, with curated graph tools offering a complementary advantage for cohort-level and temporal questions. For smaller models that struggle with tool-calling protocols, sql-fts or llm-only are more reliable choices than agentic approaches. These findings characterize the relative strengths of each configuration as a research benchmark; clinical safety, decision impact, and downstream harm are outside the scope of this evaluation and would need to be assessed before any of these configurations could be considered for use in clinical practice.

5.5. Cost and Accuracy Tradeoffs

The most expensive approaches are the agentic ones (graph, graph-cypher and sql-t2s) and they cost the most on the most capable model. For Claude Haiku 4.5 a single 334-question run on the graph approach costs USD 16.85, more than three times the USD 4.86 of the context-only llm-only baseline (Table 3). These agentic approaches are, however, also the ones that reach the highest accuracy, so once cost is normalized per correct answer the gap narrows considerably: graph costs USD 0.064 per correct answer against USD 0.023 for llm-only (Table 4). The higher per-run price of the agentic approaches therefore buys additional accuracy rather than being spent unproductively. API pricing changes frequently, so the absolute numbers reported here should be read as a snapshot rather than as fixed costs. As expected, the more capable models command the higher prices: Claude Haiku 4.5 is several times more expensive than Qwen 2.5 72B on every approach, while Llama 3.1 8B is effectively free because it runs locally. Cost and capability thus move together, and any deployment decision must weigh the two against each other.

5.6. Limitations

The patient cohort is entirely synthetic. Synthea generates clinically realistic data based on published disease incidence statistics and epidemiological models, but synthetic data is inherently cleaner and more structured than real EHR data. Real clinical records contain inconsistent coding across providers, missing or contradictory entries, free-text notes, and data quality issues that Synthea does not reproduce. The accuracy figures reported here are specific to this well-structured synthetic environment, and performance on real EHR data may differ substantially from the results presented.

The two main structured retrieval approaches are likely to be affected differently by real-world data quality issues. SQL-based approaches are vulnerable to key integrity failures: missing or mismatched foreign keys, duplicate patient identifiers, and inconsistent coding across encounters can cause JOIN operations to return incomplete or incorrect result sets without raising an explicit error, leaving the text-to-SQL agent with no signal that its query succeeded on corrupted data. Knowledge graph approaches face a complementary failure mode: incomplete or inconsistently coded entities result in absent or disconnected nodes, causing graph traversal to terminate early or return partial answers. Because both failure modes are silent, the system returns an answer rather than an error and neither is easily detectable without ground-truth validation. Evaluating these differential effects on a real, noisy EHR dataset is an important direction for future work.

The four models evaluated span a limited portion of the capability spectrum. Larger and newer models would be expected to perform better, particularly on the tool-calling approaches where Llama 3.1 8B struggled. The collapse in accuracy seen when Llama 3.1 8B was paired with agentic approaches is an important finding in itself; it shows that model capability is not just a performance factor but a condition for certain approaches to work at all.

The three cohort sizes tested (200, 2000, and 20,000 patients) cover a useful range, but remain small relative to real-world EHR systems, which can contain several million patients. Some failure modes may only emerge at those scales, and the scaling conclusions reported here should not be extrapolated beyond the range tested.

Finally, the scoring rubric does not penalize verbose answers. As shown in Figure S3, the `over_answered` flag (raised when a model gives the correct answer but adds unrequested information) is highest for Claude sql-t2s and graph-cypher, at 11.3% and 10.7% of responses respectively, compared to 2.7% for graph and 5.2% for sql-fts. In a real clinical setting, accuracy alone is not enough; a clinician asking a simple question expects a concise answer, not additional unrequested clinical context. Future work should consider whether verbosity should be penalized alongside factual correctness in the scoring rubric. At the system level, over-answering could be discouraged through explicit brevity constraints in the system prompt (e.g., instructing the model to respond in a single sentence or to return only the requested value), output-length penalties during fine-tuning, or post-generation filtering that strips content beyond the first direct answer, each of which could be measured using the `over_answered` flag as a direct indicator of conciseness.

A potential source of bias is that Claude Haiku 4.5 serves as both a system under evaluation and the primary scoring judge, raising the possibility that its scores are systematically inflated. As a cross-check, we computed agreement between the LLM judge and the deterministic reference scorer broken down by model, and found no evidence of systematic self-preference (Table S1). Nevertheless, future evaluations should consider using a judge model that is not among the systems being benchmarked, or reporting scores under multiple judges to quantify inter-judge variability.

6. Conclusions

This study compared six retrieval approaches for clinical question answering over electronic health records across four models, three cohort sizes, and 334 questions covering six clinically relevant categories. The main contribution is a controlled, reproducible evaluation that isolates the effect of the retrieval approach and the model from each other and from cohort scale.

The results show that text-to-SQL works best as a general-purpose approach for capable models, with curated graph tools performing better for cohort-level and temporal questions. Interestingly, graph-cypher outperforms text-to-SQL for Claude Haiku 4.5 at the largest cohort size tested, which raises the question of whether graph-based approaches become more competitive as the patient population grows. Testing this at the scale of millions of patients is a direct and important continuation of this work.

More importantly, the results show that adding structured retrieval tools does not help equally across models; it substantially improves accuracy for a capable model but reduces it for a small open-source model that cannot reliably call tools. Cohort size of these ranges, by contrast, has almost no effect on accuracy across the range tested.

Supplementary Materials: The following supporting information can be downloaded at <https://www.mdpi.com/article/10.3390/fi18070365/s1>, Table S1: Self-preference test: mean score per model under each judge; Table S2: Judge–deterministic-scorer agreement per model; Table S3: Friedman omnibus on the continuous judge score, per (model, subset); Table S4: Linear regression of the per-question judge score on log10(subset); Table S5: Cochran’s Q omnibus on the binarized judge score; Table S6: Pairwise McNemar tests with continuity correction; Figure S1: Retrieval-level answer containment by question type (no language model); Figure S2: Per-(model, subset) pairwise Wilcoxon heatmaps; Figure S3: Judge flag rates per (model, approach); Supplementary S1: Retrieval Methods Comparison; Supplementary S2: Question bank: design, provenance, and clinician review; Supplementary S3: Adjudicated calibration cases for the LLM judge rubric.

Author Contributions: Conceptualization, L.A., A.V., I.V., I.K. and G.K.M.; methodology, L.A. and A.V.; software, L.A. and A.V.; validation, L.A. and A.V.; formal analysis, L.A. and A.V.; investigation, L.A. and A.V.; resources, L.A. and A.V.; data curation, L.A. and A.V.; writing—original draft preparation, L.A. and A.V.; writing—review and editing, L.A., A.V., I.V., I.K., O.P. and G.K.M.; visualization, L.A. and A.V.; supervision, O.P. and G.K.M.; project administration, G.K.M.; funding acquisition, G.K.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data were synthetically generated. The code to regenerate them is publicly available at <https://github.com/anagnole/EHR-Clinical-Assistant> (accessed on 12 July 2026).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

EHR	Electronic Health Record
LLM	Large Language Model
QA	Question Answering
SQL	Structured Query Language
RAG	Retrieval-Augmented Generation
FHIR	Fast Healthcare Interoperability Resources
MCP	Model Context Protocol
KG	Knowledge Graph
NLP	Natural Language Processing
CI	Confidence Interval

References

1. Du, X.; Zhou, Z.; Wang, Y.; Chuang, Y.-W.; Li, Y.; Yang, R.; Zhang, W.; Wang, X.; Chen, X.; Guan, H.; et al. Testing and Evaluation of Generative Large Language Models in Electronic Health Record Applications: A Systematic Review. *medRxiv* **2025**. [[CrossRef](#)] [[PubMed](#)]
2. Shi, W.; Xu, R.; Zhuang, Y.; Yu, Y.; Zhang, J.; Wu, H.; Zhu, Y.; Ho, J.C.; Yang, C.; Wang, M.D. EHRAgent: Code Empowers Large Language Models for Few-shot Complex Tabular Reasoning on Electronic Health Records. In *Proceedings of the Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*; Al-Onaizan, Y., Bansal, M., Chen, Y.-N., Eds.; Association for Computational Linguistics: Miami, FL, USA, 2024; pp. 22315–22339. [[CrossRef](#)] [[PubMed](#)]
3. De Maio, C.; Fenza, G.; Furno, D.; Grauso, T.; Loia, V. A Multi-Agent Architecture for Privacy-Preserving Natural Language Interaction with FHIR-Based Electronic Health Records. In *Proceedings of the 2024 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*; IEEE: New York, NY, USA, 2024; pp. 1–6. [[CrossRef](#)]
4. De Maio, C.; Fenza, G.; Furno, D.; Grauso, T.; Loia, V. Privacy-Preserving Healthcare Data Interactions: A Multi-Agent Approach Using LLMs. *J. Commun. Softw. Syst.* **2025**, *21*, 13–22. [[CrossRef](#)]
5. Lee, G.; Hwang, H.; Bae, S.; Kwon, Y.; Shin, W.; Yang, S.; Seo, M.; Kim, J.-Y.; Choi, E. EHRSQL: A Practical Text-to-SQL Benchmark for Electronic Health Records. *arXiv* **2026**, arXiv:2301.07695. [[CrossRef](#)]
6. Pampari, A.; Raghavan, P.; Liang, J.; Peng, J. emrQA: A Large Corpus for Question Answering on Electronic Medical Records. *arXiv* **2018**, arXiv:1809.00732. [[CrossRef](#)]
7. Jiang, P.; Xiao, C.; Cross, A.; Sun, J. GraphCare: Enhancing Healthcare Predictions with Personalized Knowledge Graphs. *arXiv* **2024**, arXiv:2305.12788. [[CrossRef](#)]
8. Lee, G.; Bach, E.; Yang, E.; Pollard, T.; Johnson, A.; Choi, E.; Jia, Y.; Lee, J.H. FHIR-AgentBench: Benchmarking LLM Agents for Realistic Interoperable EHR Question Answering. *arXiv* **2025**, arXiv:2509.19319. [[CrossRef](#)]
9. Walonoski, J.; Kramer, M.; Nichols, J.; Quina, A.; Moesel, C.; Hall, D.; Duffett, C.; Dube, K.; Gallagher, T.; McLachlan, S. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. *J. Am. Med. Inform. Assoc.* **2018**, *25*, 230–238. [[CrossRef](#)] [[PubMed](#)]
10. Wang, P.; Shi, T.; Reddy, C.K. Text-to-SQL Generation for Question Answering on Electronic Medical Records. *arXiv* **2020**, arXiv:1908.01839. [[CrossRef](#)]
11. Johnson, A.E.W.; Pollard, T.J.; Shen, L.; Lehman, L.H.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Anthony Celi, L.; Mark, R.G. MIMIC-III, a freely accessible critical care database. *Sci. Data* **2016**, *3*, 160035. [[CrossRef](#)] [[PubMed](#)]
12. Kweon, S.; Kim, J.; Kwak, H.; Cha, D.; Yoon, H.; Kim, K.; Yang, J.; Won, S.; Choi, E. EHRNoteQA: An LLM Benchmark for Real-World Clinical Practice Using Discharge Summaries. *arXiv* **2024**, arXiv:2402.16040. [[CrossRef](#)]
13. Johnson, A.E.W.; Bulgarelli, L.; Shen, L.; Gayles, A.; Shammout, A.; Horng, S.; Pollard, T.J.; Hao, S.; Moody, B.; Gow, B.; et al. MIMIC-IV, a freely accessible electronic health record dataset. *Sci. Data* **2023**, *10*, 1. [[CrossRef](#)] [[PubMed](#)]
14. Edge, D.; Trinh, H.; Cheng, N.; Bradley, J.; Chao, A.; Mody, A.; Truitt, S.; Metropolitan, D.; Ness, R.O.; Larson, J. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. *arXiv* **2025**, arXiv:2404.16130. [[CrossRef](#)]
15. Wu, J.; Zhu, J.; Qi, Y.; Chen, J.; Xu, M.; Menolascina, F.; Jin, Y.; Grau, V. Medical Graph RAG: Evidence-based Medical Large Language Model via Graph Retrieval-Augmented Generation. In *Proceedings of the Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*; Che, W., Nabende, J., Shutova, E., Pilehvar, M.T., Eds.; Association for Computational Linguistics: Vienna, Austria, 2025; pp. 28443–28467. [[CrossRef](#)]
16. Ozsoy, M.G.; Messallem, L.; Besga, J.; Minneci, G. Text2Cypher: Bridging Natural Language and Graph Databases. *arXiv* **2024**, arXiv:2412.10064. [[CrossRef](#)]
17. Hu, N.; Wu, Y.; Qi, G.; Min, D.; Chen, J.; Pan, J.Z.; Ali, Z. An Empirical Study of Pre-trained Language Models in Simple Knowledge Graph Question Answering. *arXiv* **2023**, arXiv:2303.10368. [[CrossRef](#)]
18. Tan, Y.; Zhang, X.; Chen, Y.; Ali, Z.; Hua, Y.; Qi, G. CLRN: A reasoning network for multi-relation question answering over Cross-lingual Knowledge Graphs. *Expert Syst. Appl.* **2023**, *231*, 120721. [[CrossRef](#)]
19. Khan, A.; Ali, Z.; Aziz, A.; Kefalas, P. Talk2Doc: A Patient Q&A system using Retrieval-Augmented Generation with Weighted Knowledge Graphs and LLMs. In *Proceedings of the 21st International Conference on Intelligent Computing (ICIC 2025)*, Ningbo, China, 26–29 July 2025. [[CrossRef](#)]
20. Ali, Z.; Huang, Y.; Khan, A.; Qi, G.; Zhang, Y.; Feng, J.; Deng, C.; Kefalas, P. Pythia-RAG: Retrieval-augmented generation over a unified multimodal knowledge graph for enhanced QA. *Knowl.-Based Syst.* **2026**, *335*, 115200. [[CrossRef](#)]
21. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. ReAct: Synergizing Reasoning and Acting in Language Models. *arXiv* **2023**, arXiv:2210.03629. [[CrossRef](#)]
22. Jiang, Y.; Black, K.C.; Geng, G.; Park, D.; Zou, J.; Ng, A.Y.; Chen, J.H. MedAgentBench: A Realistic Virtual EHR Environment to Benchmark Medical LLM Agents. *arXiv* **2025**, arXiv:2501.14654. [[CrossRef](#)]
23. Su, X.; Wang, Y.; Gao, S.; Liu, X.; Giunchiglia, V.; Clevert, D.-A.; Zitnik, M. KGAREvision: An AI Agent for Knowledge-Intensive Biomedical QA. *arXiv* **2025**, arXiv:2410.04660. [[CrossRef](#)]

24. Liu, Y.; Iter, D.; Xu, Y.; Wang, S.; Xu, R.; Zhu, C. G-Eval: NLG Evaluation using Gpt-4 with Better Human Alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*; Bouamor, H., Pino, J., Bali, K., Eds.; Association for Computational Linguistics: Singapore, 2023; pp. 2511–2522. [\[CrossRef\]](#)
25. Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E.P.; et al. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. *arXiv* **2023**, arXiv:2306.05685. [\[CrossRef\]](#)
26. Jin, D.; Pan, E.; Oufattole, N.; Weng, W.-H.; Fang, H.; Szolovits, P. What Disease does this Patient Have? A Large-scale Open Domain Question Answering Dataset from Medical Exams. *arXiv* **2020**, arXiv:2009.13081. [\[CrossRef\]](#)
27. Feng, X.; Jin, G.; Chen, Z.; Liu, C.; Salihoğlu, S. KÜZU Graph Database Management System. In *Proceedings of the 13th Annual Conference on Innovative Data Systems Research (CIDR'23)*, Amsterdam, The Netherlands, 8–11 January 2023.
28. Introducing Claude Haiku 4.5. Available online: <https://www.anthropic.com/news/claude-haiku-4-5> (accessed on 22 May 2026).
29. Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; et al. Qwen2.5 Technical Report. *arXiv* **2025**, arXiv:2412.15115. [\[CrossRef\]](#)
30. Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. The Llama 3 Herd of Models. *arXiv* **2024**, arXiv:2407.21783. [\[CrossRef\]](#)
31. Friedman, M. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *J. Am. Stat. Assoc.* **1937**, *32*, 675–701. [\[CrossRef\]](#)
32. Cliff, N. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychol. Bull.* **1993**, *114*, 494–509. [\[CrossRef\]](#)
33. Vezakis, I.A.; Lambrou, G.I.; Matsopoulos, G.K. Deep Learning Approaches to Osteosarcoma Diagnosis and Classification: A Comparative Methodological Approach. *Cancers* **2023**, *15*, 2290. [\[CrossRef\]](#) [\[PubMed\]](#)
34. Vezakis, A.; Vezakis, I.; Petropoulou, O.; Miloulis, S.T.; Anastasiou, A.; Kakkos, I.; Matsopoulos, G.K. Comparative Analysis of Deep Neural Networks for Automated Ulcerative Colitis Severity Assessment. *Bioengineering* **2025**, *12*, 413. [\[CrossRef\]](#) [\[PubMed\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.