GraphDx: A Cost-Aware Knowledge-Enhanced Multi-Agent Framework for Sequential Diagnosis

Shaoting Tan[1], Ning Liu[1], Yuntao Du[1], Shuyue Wei[1], Wu Shuai[2], Qian Li[1], Yanyu Xu[1], Wei Zhang[3], Lizhen Cui[1], Haitao Yuan[4]

[1] School of Software, Shandong University [2] College of Computer Science, Nankai University
[3] School of Computer Science and Technology, East China Normal University
[4] Nanyang Technological University, Singapore

Correspondence: liun21cs@sdu.edu.cn

Abstract

Sequential diagnosis requires balancing diagnostic accuracy against resource costs through iterative information gathering. Existing Large Language Model (LLM) approaches exhibit a critical *knowledge-reasoning gap*: despite encoding extensive medical knowledge, they struggle to reason systematically under cost constraints, often resorting to excessive testing. We propose `GraphDx`, a knowledge-enhanced framework with two core innovations. First, we design an automated pipeline that leverages LLMs to construct Medical Diagnosis Knowledge Graphs (MDKGs) with quantized typicality, action-centric topology, and dual-objective attributes for both diagnostic relevance and cost-sensitivity. Second, we introduce three collaborative agents (Perception, Reasoning, and Decision) where the Perception and Decision Agents handle language understanding and generation, while the Reasoning Agent performs deterministic evidence scoring and cost-aware planning on the MDKG. Experiments on MedQA and MIMIC-IV across three LLM backbones (DeepSeek-V3, Kimi-k2, Llama-3.3) show that `GraphDx` improves diagnostic success rates from 50–68% to 79–93% while reducing test costs by 20–54%, providing a robust, economical, and interpretable solution for automated clinical diagnosis.

1 Introduction

Large Language Models (LLMs) have achieved remarkable success on medical examinations, with models like GPT-4 surpassing human performance on USMLE (Nori et al., 2023; Singhal et al., 2023), demonstrating that LLMs encode extensive medical knowledge within their parameters. However, translating this success to real clinical practice, particularly **sequential diagnosis**, remains a significant challenge.

Unlike static question answering, sequential diagnosis mirrors actual clinical workflows: physicians iteratively gather information through symptom inquiries and diagnostic tests, making decisions under uncertainty while balancing diagnostic accuracy against resource costs including monetary expense, patient discomfort, and time. This multi-turn process requires not merely knowledge recall, but systematic reasoning that coordinates information acquisition with hypothesis refinement.

Current LLM-based approaches exhibit a critical limitation we term the *knowledge-reasoning gap*: despite possessing substantial medical knowledge, LLMs struggle to reason systematically under cost constraints. Recent studies (Nori et al., 2025; Jia et al., 2025) reveal that LLM agents tend toward "defensive medicine," ordering excessive tests to compensate for reasoning uncertainty while failing to leverage discriminative features for efficient differential diagnosis. This behavior stems from the implicit, unstructured nature of LLM reasoning, which lacks explicit domain models enabling systematic hypothesis testing. We identify two key challenges underlying this gap:

Challenge 1: Diagnosis-Ready Knowledge Graph Construction. Sequential diagnosis requires *quantitative* attributes beyond semantic relations, including symptom typicality and test utilities. Existing knowledge graphs like UMLS lack these attributes, while current automated methods (Chen and Bertozzi, 2023; Anuyah et al., 2025) extract only semantic triples. Moreover, medical concepts exhibit context-dependent semantics (e.g., "heart failure" as diagnosis vs. complication), requiring dynamic role modeling that static ontologies cannot provide.

Challenge 2: Effective Multi-Agent Collaboration. Bridging neural language understanding with symbolic graph reasoning requires orchestrating observation extraction, graph grounding, evidence scoring, and response generation. Recent approaches like MAI-DxO (Nori et al., 2025) rely

Findings of the Association for Computational Linguistics: ACL 2026, pages 21721–21736

on prompt-based coordination, but their complexity demands strong instruction-following from base models, causing performance degradation on less capable LLMs.

To address these challenges, we propose GraphDx, a knowledge-enhanced framework comprising two integrated components (Figure 1):

(1) Automated MDKG Construction (§3.2) addresses **Challenge 1**. We design an automated pipeline where specialized agents collaborate to construct MDKGs enriched with *quantitative diagnostic attributes* including typicality weights and test cost-effectiveness metrics that existing knowledge graphs lack. The pipeline employs a three-stage process: parallel knowledge extraction with quantized typicality, hybrid entity alignment with dynamic node upgrade for handling semantic role transitions, and attribute enrichment for cost-sensitive planning. This enables complex causal chains to emerge automatically without manual annotation.

(2) Collaborative Diagnostic Agents (§3.3) addresses **Challenge 2**. We introduce three specialized agents: a *Perception Agent* that extracts observations from patient responses and grounds them to graph nodes, a *Reasoning Agent* that performs deterministic evidence scoring and cost-sensitive planning on the MDKG, and a *Decision Agent* that generates clinically appropriate responses. This architecture offloads complex reasoning to the graph, reducing dependence on base model capabilities.

We evaluate GraphDx on MedQA-Extended and MIMIC-IV datasets across three LLM backbones. Results demonstrate that GraphDx improves diagnostic success rates from 50% to over 88% while reducing test costs by 20% to 50%, with consistent gains across all tested models, including scenarios where sophisticated multi-agent baselines fail. To support reproducibility and further research, we open-source our implementation, including the automated graph construction pipeline and the ClinicSim environment at https://github.com/ossiver-GEL/GraphDx.

2 Related Work

We categorize related work into three categories and compare them in Table 1.

Table 1: Comparison of Recent Works and Ours. ✓=Yes, ✗=No, △=Weak/partial, —=Not applicable.

Method	Explicit KG	Auto Construction	Sequential Diagnosis	Cost-Sensitive
End-to-End LLM (Singhal et al., 2023)	✗	–	△	✗
Multi-Agent LLM (Nori et al., 2025)	✗	–	✓	△
KG-Enhanced LLM (Gao et al., 2025)	✓	✗	△	✗
AutoKG (Chen and Bertozzi, 2023)	✓	✓	✗	✗
GraphDx (Ours)	✓	✓	✓	✓

Sequential Diagnosis Agents. Sequential diagnosis requires agents to balance information acquisition and diagnostic costs over multiple interactions. Recently, Nori et al. (2025) proposed SD-Bench and MAI-DxO, achieving diagnostic accuracy and cost control surpassing human doctors through multi-agent orchestration. Jia et al. (2025) proposed the DDO framework, using Reinforcement Learning (RL) to optimize symptom inquiry strategies. Schmidgall et al. (2025) and Qiu et al. (2025) built AgentClinic and DiagGym simulation environments respectively for training and evaluating multimodal diagnostic agents. Additionally, ACTMED proposed by Estévez et al. (2025) combines Bayesian experimental design with LLMs to actively select tests. However, most of these methods rely on the implicit reasoning of LLMs or RL with low sample efficiency, lacking an explicit, interpretable structured domain model to guide decisions.

Automated Medical Knowledge Graph Construction. Traditional MKG construction relies on expert annotation and is costly. Recent research has begun to explore using LLMs for automated graph construction. Chen and Bertozzi (2023) proposed AutoKG, using LLMs to extract keywords and calculate graph Laplacian weights. CoDe-KG by Anuyah et al. (2025) uses sentence complexity modeling to extract triples. Sarabadani et al. (2025) proposed DKG-LLM, integrating dynamic knowledge graphs with the Grok 3 model. Although these works achieve automation, the constructed graphs are mostly used for semantic retrieval and lack the fine-grained typicality parameters (e.g., symptom weights) and decision utility attributes (e.g., test cost, invasiveness) required for diagnostic reasoning.

Knowledge-Enhanced Clinical Decision. Knowledge-Enhanced AI aims to combine the perceptual capabilities of neural networks with the logical capabilities of symbolic systems. Gao et al. (2025) explored integrating MKG into LLMs for diagnostic prediction. Our method goes a step further, not only achieving automated graph construction but also designing an explicit

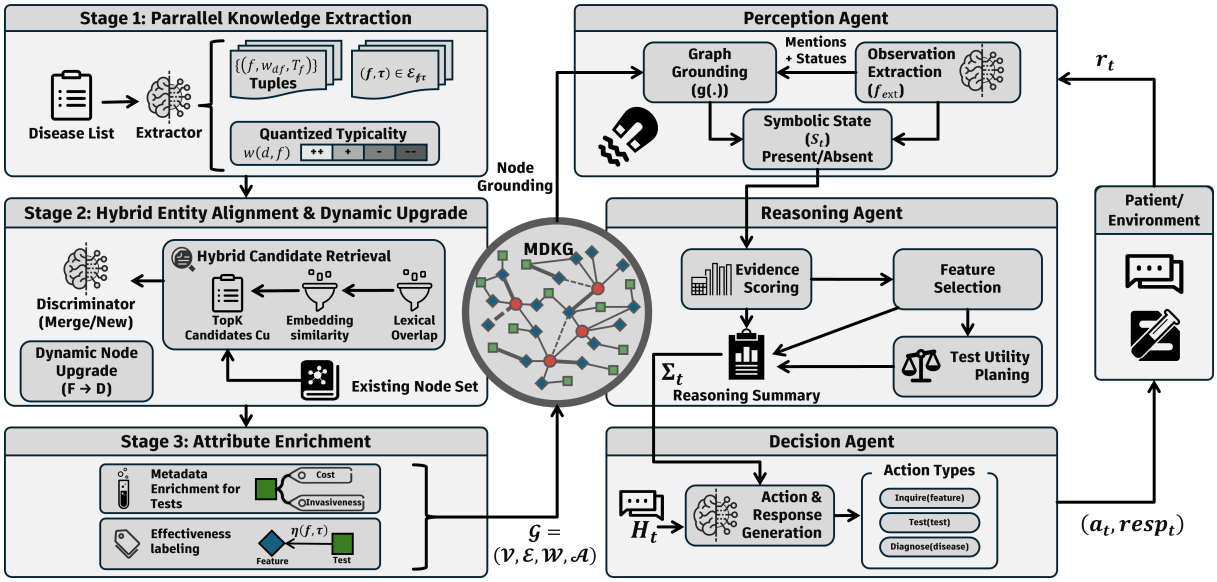


Figure 1: Overview of the GraphDx framework. Left: Automated MDKG Construction Pipeline (Algorithm 1), utilizing parallel extraction and hybrid alignment to build the graph. Right: Graph-Augmented Diagnostic Agent (Algorithm 2), employing a knowledge-enhanced loop of observation extraction, graph grounding, evidence scoring, and utility-based action planning.

evidence scoring engine based on the graph, realizing true "knowledge-guided reasoning."

In summary, our method GraphDx is the only framework that simultaneously possesses automated graph construction capabilities, supports sequential diagnosis, and explicitly models test costs.

3 Methodology

3.1 Problem Formulation

We formulate sequential diagnosis as a partially observable decision process. Let $\mathcal{D}$ be the set of possible diseases. At turn t, the agent observes dialogue history H_t and maintains a belief state over $\mathcal{D}$. The agent selects an action $a_t \in \mathcal{A}_{action}$:

- **Inquire**(f): Ask about symptom or feature f.
- **Test**(τ): Order diagnostic test τ.
- **Diagnose**(d): Provide final diagnosis d.

The objective is to minimize the expected total cost:

$$\min_{\pi} \mathbb{E}\left[\sum_{t=1}^{T} c(a_t) + \lambda_{mis} \cdot \mathbb{I}[\hat{d} \neq d^*]\right] \quad (1)$$

where $c(a_t)$ is the cost of action a_t (including monetary cost, invasiveness, and time), λ_{mis} is the penalty for misdiagnosis, $\hat{d}$ is the predicted diagnosis, and d^* is the ground truth.

3.2 Automated Construction of Medical Diagnosis Knowledge Graph

The core of GraphDx is the Medical Diagnosis Knowledge Graph (MDKG), formally defined as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W}, \mathcal{A})$ where:

- $\mathcal{V} = \mathcal{V}_d \cup \mathcal{V}_f \cup \mathcal{V}_t$: nodes for diseases, features (symptoms/signs), and tests.
- $\mathcal{E} \subseteq (\mathcal{V}_d \times \mathcal{V}_f) \cup (\mathcal{V}_f \times \mathcal{V}_t)$: edges connecting diseases to features and features to tests.
- $\mathcal{W} : \mathcal{E} \to \mathbb{R}$: edge weights encoding association strength.
- $\mathcal{A} : \mathcal{V}_t \to \mathbb{R}^2$: test attributes (cost, invasiveness).

We implement a fully automated pipeline to construct $\mathcal{G}$ from a disease list $\mathcal{D}_{list}$ without manual annotation. The process proceeds in three stages, designed to ensure the graph serves as a robust computational substrate for evidence-based reasoning.

Stage 1: Parallel Knowledge Extraction. The system processes each disease $d \in \mathcal{D}_{list}$ in parallel. For each disease, it prompts the LLM to extract a set of tuples $\{(f, w_{df}, T_f)\}$, where f is a feature, w_{df} is its typicality, and T_f is a list of verifying tests. To mitigate the noise from poorly calibrated continuous probability estimates in LLMs, we employ a **Quantized Typicality** strategy. We map linguistic typicality descriptions to discrete weight buckets $\mathcal{W} : \mathcal{E}_{df} \to \mathbb{R}$:

$$w(d, f) = \begin{cases} w_{++} & \text{if } f \text{ is Hallmark (++)} \\ w_{+} & \text{if } f \text{ is Common (+)} \\ w_{-} & \text{if } f \text{ is Rare (-)} \\ w_{--} & \text{if } f \text{ is Impossible (–)} \end{cases} \quad (2)$$

This quantization acts as a low-pass filter, stabilizing the reasoning process against minor probability

fluctuations in LLM generation.

Additionally, to ensure the graph directly supports diagnostic planning, we enforce an **Action-Centric Topology**. Instead of just declarative knowledge ("Pneumonia causes cough"), the extraction explicitly models *verification edges* $(f, \tau) \in \mathcal{E}_{f\tau}$ connecting features to the tests that can confirm or rule them out. This transforms the graph into a navigation structure: given an uncertain feature, the agent can directly traverse to relevant diagnostic actions.

Stage 2: Hybrid Entity Alignment & Dynamic Upgrade. Medical concepts often suffer from semantic ambiguity and role shifts (e.g., a disease in one context is a symptom in another). To resolve synonymy, we employ a **Hybrid Alignment Mechanism** to align a new term u to the existing node set $\mathcal{V}$. We retrieve a candidate set $\mathcal{C}_u$ using a hybrid retrieval strategy that combines lexical and semantic matching:

$$\mathcal{C}_u = \text{TopK}_{\text{overlap}}(u, \mathcal{V}) \cup \text{TopK}_{\text{embed}}(u, \mathcal{V}) \quad (3)$$

where $\text{TopK}_{\text{overlap}}$ selects candidates based on the Overlap Coefficient (intersection over minimum set size) to capture partial matches (e.g., "Diabetes" $\subseteq$ "Type 2 Diabetes"), and $\text{TopK}_{\text{embed}}$ selects candidates based on cosine similarity of dense embeddings $\phi(\cdot)$ to capture semantic synonyms. An LLM then acts as a discriminator $D(u, \mathcal{C}_u)$ to determine if u should merge with $v^* \in \mathcal{C}_u$ or form a new node.

Furthermore, to handle role shifts without manual intervention, we introduce a **Dynamic Node Upgrade** strategy. When a node initially added as a feature (e.g., "Heart Failure" as a symptom of renal disease) is later identified as an independent disease, the system:

1. Promotes the node from $\mathcal{V}_f$ to $\mathcal{V}_d$: $\mathcal{V}_d \leftarrow \mathcal{V}_d \cup \{v\}, \mathcal{V}_f \leftarrow \mathcal{V}_f \setminus \{v\}$
2. Preserves all existing incoming edges (d', v) (its role as a feature)
3. Enables new outgoing edges (v, f') (its features as a disease)

This allows complex causal chains (e.g., Renal Disease $\rightarrow$ Heart Failure $\rightarrow$ Dyspnea) to emerge automatically without manual ontology restructuring.

Stage 3: Attribute Enrichment. Finally, to support cost-sensitive decision making, the graph requires quantitative metadata often missing from raw extractions. The **Metadata Enrichment Module** infers these attributes for new test nodes. Each test node $\tau \in \mathcal{V}_t$ is associated with attributes $\mathcal{A}(\tau) = (c_\tau, \iota_\tau)$, where c_τ represents the monetary cost level and ι_τ represents the invasiveness multiplier. This ensures the graph is fully populated with the quantitative data required for downstream planning.

3.3 Graph-Augmented Diagnostic Agent

To bridge the gap between unstructured patient interaction and structured medical reasoning, we design a **Graph-Augmented Diagnostic Agent**. This agent operates through a "Think-then-Act" loop (Algorithm 2), where a neural component handles perception and a symbolic component handles reasoning.

3.3.1 Perception Agent

Since patient inputs are unstructured and diverse (ranging from verbal descriptions to medical reports), the system must first normalize them into a structured format. The **Perception Module** acts as this interface. At turn t, it employs an extraction function f_{ext} to parse the input r_t (which may be a patient's verbal response or a returned medical test report) into structured observations:

$$O_t = f_{ext}(r_t | H_{t-1}) = \{(m_i, s_i)\}_{i=1}^{K} \quad (4)$$

where m_i is the mention text (e.g., "fever" or "WBC count elevated") and $s_i \in \{\text{Present}, \text{Absent}\}$ is the status. Crucially, this unified extraction handles both subjective symptoms and objective test findings identically. Subsequently, a grounding function $g : \mathcal{M} \rightarrow \mathcal{V}_f$ maps each mention m_i to a canonical graph node f_i^* using the alignment mechanism from Sec. 3.2. The symbolic state is updated as $\mathcal{S}_t = \mathcal{S}_{t-1} \cup \{(g(m_i), s_i)\}_{i=1}^{K}$.

3.3.2 Reasoning Agent

Once observations are grounded, the core challenge is to make deterministic and cost-effective decisions. The **Reasoning Engine** addresses this through two key mechanisms:

Evidence Scoring. To avoid the instability of black-box neural reasoning, we implement an explicit **Evidence Scoring** mechanism. It computes a confidence score $S(d)$ for each disease d based on the alignment between the graph knowledge and the observed state $\mathcal{S}_t$. The scoring function

explicitly handles positive and negative evidence:

$$S(d) = \sum_{(f,s)\in\mathcal{S}_t} \text{Score}(f, d, s) \tag{5}$$

where the component score is defined as:

$$\text{Score}(f, d, s) = \begin{cases} w(d, f) & \text{if } s = \text{Present} \\ -\lambda \cdot w(d, f) & \text{if } s = \text{Absent} \end{cases} \tag{6}$$

Here, λ is a penalty factor. Note that if $w(d, f) < 0$ (the disease typically lacks the feature) and $s =$ Present, the term $w(d, f)$ naturally reduces the score. Conversely, if $s =$ Absent and $w(d, f) > 0$ (expected feature missing), the term $-\lambda \cdot w(d, f)$ penalizes the disease.

Cost-Sensitive Action Planning. To balance diagnostic accuracy with cost (mitigating "defensive medicine"), we employ a **Cost-Sensitive Action Planning** module. While full Bayesian Value of Information (VoI) calculation is computationally prohibitive for real-time interaction, we implement a *heuristic utility function* that approximates information gain via feature variance and relevance:

- **Feature Selection.** We identify unobserved features that maximize discriminative power among the top-K candidate diseases D_{top}. The discriminative score $V(f)$ serves as a lightweight proxy for information gain:

$$V(f) = \sum_{d\in D_{top}} |S(d)| \cdot |w(d, f)| + \alpha \cdot \text{Var}_{d\in D_{top}}(w(d, f)) \tag{7}$$

  The first term prioritizes features relevant to high-confidence diseases, while the second term (variance) favors features that can differentiate between them.

- **Test Selection.** For each candidate test τ, we calculate its utility $U(\tau)$ by aggregating the value of the features it can verify:

$$U(\tau) = \frac{\sum_{f\in\text{Verifies}(\tau)} V(f) \cdot \eta(f, \tau)}{1 + c_\tau \cdot \iota_\tau} \tag{8}$$

  where $\eta(f, \tau) \in \{\eta_{low}, \eta_{high}\}$ denotes the effectiveness of test τ for feature f. The denominator penalizes expensive and invasive tests, ensuring the selected actions maximize the information-to-cost ratio.

3.3.3 Decision Agent

Acting as the "right brain," this agent generates the final action a_t and natural language response $resp_t$ by conditioning on the structured reasoning summary Σ_t:

$$(a_t, resp_t) \sim \pi_\theta(\cdot \mid H_t, \Sigma_t) \tag{9}$$

where Σ_t aggregates the symbolic outputs:

$$\Sigma_t = \left\langle \text{Rank}(\mathcal{D}), \text{Top}_K^V(f), \text{Top}_K^U(\tau) \right\rangle \tag{10}$$

This injection of Σ_t constrains the LLM's generation space to medically valid paths derived from the MDKG.

3.4 Design Rationale

The `GraphDx` architecture embodies a key insight: *neural and symbolic components excel at complementary tasks*. LLMs handle the "soft" aspects of diagnosis—understanding colloquial language, maintaining conversational flow, and exercising clinical judgment in ambiguous situations. The symbolic engine handles the "hard" aspects—deterministic state tracking, systematic hypothesis scoring, and principled cost-benefit analysis. This separation provides several advantages: (1) **Interpretability**: Every diagnostic conclusion traces to explicit evidence scores; (2) **Consistency**: Deterministic state tracking prevents contradictory reasoning; (3) **Efficiency**: Symbolic utility computation is faster than LLM deliberation; (4) **Robustness**: Graph structure constrains LLM outputs to valid medical reasoning.

4 Experiments

We conduct extensive experiments to address the following research questions: **Q1**: Can `GraphDx` achieve superior diagnostic accuracy and cost-efficiency compared to current SOTA LLM frameworks? **Q2**: How robust is our method when transferring to real-world, noisy clinical scenarios? **Q3**: What are the individual contributions of the structured MDKG and the explicit inference engine?

Implementation Details. We evaluate `GraphDx` on two datasets: **MedQA** (200 cases) and **MIMIC-IV** (200 real-world cases). We employ `ClinicSim`, a modular simulation environment, to conduct multi-turn diagnostic dialogues. We compare our method against **Standard LLM** prompting and the multi-agent **MAI-DxO** framework (Nori et al., 2025) across three LLM backbones: DeepSeek-V3,

Kimi-k2, and Llama-3.3. Performance is measured by Diagnostic Score (0-10), Total Cost ($), and Dialogue Turns. For detailed descriptions of the datasets, baseline settings, hyperparameter configurations, and the simulation environment, please refer to Appendix D.

4.1 Experimental Results

4.1.1 Main Results

Table 2 and Table 3 present the performance comparison between GraphDx and baselines on MedQA and MIMIC-IV datasets, respectively. The results demonstrate that our method achieves consistent and significant improvements across all metrics and base models.

Diagnostic Accuracy. On the MedQA dataset, GraphDx significantly outperforms the Standard LLM baseline. For instance, with DeepSeek-V3, the success rate (Score ≥ 8) jumps from 67.8% to 88.7%, and Kimi-k2 achieves a remarkable 93.2% success rate (assessment details in Appendix G.3). We also compared our method with the SOTA multi-agent framework MAI-DxO (Nori et al., 2025). While MAI-DxO improves accuracy on DeepSeek-V3 (8.05), it incurs extremely high costs ($2460) and interaction turns (9.51). On the smaller Llama-3.3 model, MAI-DxO suffers from performance collapse (Score 5.38), indicating its high dependency on the base model's capability. In contrast, GraphDx achieves Pareto optimality across all models by offloading reasoning to the graph. This advantage generalizes well to the real-world MIMIC-IV dataset. Despite the increased clinical noise, GraphDx improves the average diagnostic score by approximately 2.0 points (e.g., DeepSeek-V3: 6.60 $\rightarrow$ 8.27) and increases the success rate by 25%–30%. In contrast, the MAI-DxO baseline struggles significantly on this real-world dataset, showing lower scores and high variance, likely due to the complexity of real clinical notes confusing the multi-agent debate mechanism. This proves that structured knowledge effectively guides the model to grasp key features even in noisy environments.

Cost-Effectiveness and Efficiency. GraphDx consistently reduces test costs by 20%–54% across both datasets while improving accuracy (see Appendix G.3.2 for details on the US-based cost estimation methodology). For DeepSeek-V3 on MedQA, the average cost drops from $1062 to $537. This is attributed to the utility-based active planning module, which effectively suppresses the "defensive medicine" behavior—blindly ordering expensive tests (e.g., full-body CT) to compensate for uncertainty—observed in baselines. Notably, on MIMIC-IV, MAI-DxO incurs extremely high costs (e.g., $1497 for DeepSeek-V3) due to uncontrolled exploration, whereas GraphDx maintains low costs ($340) by precisely targeting relevant tests. Figure 2 illustrates this advantage: while Baseline results (purple) are scattered with many high-cost, low-score samples, GraphDx (green) is tightly clustered in the "high score-low cost" region. Regarding interaction efficiency, GraphDx incurs slightly more turns than the Standard LLM (e.g., 5.06 vs. 3.80 for Llama-3.3). This is deliberate: the graph agent performs more thorough differential diagnosis, excluding distractors through multi-turn inquiries rather than jumping to premature conclusions. Given the substantial gains in diagnostic quality, this trade-off is clinically justified.

Robustness and Failure Analysis. On the challenging MIMIC-IV dataset, we observed that baselines often fail due to (1) **Noise Interference**, being misled by irrelevant details in past medical history (e.g., old fractures), and (2) **Defensive Medicine**, leading to soaring costs (e.g., Scenario 66 incurred $1825 in costs yet still resulted in misdiagnosis). GraphDx mitigates these issues by filtering noise through the MDKG's prior knowledge, focusing reasoning on core pathological features and avoiding unnecessary expensive testing.

4.1.2 Performance on Unseen Diseases (Open-Set Evaluation)

To further address the concern regarding the closed-set nature of the MDKG, we conducted an "Open-Set" evaluation using a subset of MIMIC-IV cases where the ground truth disease was **explicitly excluded** from our constructed graph. This setting tests the system's ability to handle "unknown" conditions by relying on the hybrid knowledge-enhanced architecture.

Results on the DeepSeek-V3 backbone show that even without specific disease nodes, GraphDx outperforms the Standard LLM baseline. GraphDx achieved a mean score of **7.65** compared to the baseline's **6.66**, a significant improvement of **+1.0 point**. The average test cost was reduced from **$743.48** (Baseline) to **$574.88** (GraphDx), a saving of $\sim$**23%**.

Table 2: Diagnostic Performance Comparison on MedQA Dataset

Base Model	Method	Score (0-10) ↑	Cost ($) ↓	Turns	Success Rate ↑	Harmful Rate ↓
DeepSeek-V3	Standard LLM	7.25 ± 1.34	1062.69 ± 655.20	4.66 ± 0.77	67.8%	22.5%
	MAI-DxO	8.05 ± 0.98	2460.20 ± 1114.69	9.51 ± 2.74	75.0%	13.2%
	GraphDx (Ours)	**9.05 ± 0.47**	**537.13 ± 221.24**	4.76 ± 0.94	**88.7%**	**7.7%**
Kimi-k2	Standard LLM	7.66 ± 1.24	1456.88 ± 575.35	3.62 ± 0.51	66.8%	12.2%
	MAI-DxO	6.92 ± 1.61	2220.57 ± 1026.20	11.99 ± 3.54	64.0%	24.5%
	GraphDx (Ours)	**9.50 ± 0.29**	**1156.02 ± 407.83**	4.17 ± 0.73	**93.2%**	**1.3%**
Llama-3.3	Standard LLM	7.51 ± 1.30	1857.47 ± 773.05	3.80 ± 0.68	65.7%	12.5%
	MAI-DxO	5.38 ± 2.06	3029.19 ± 1236.24	16.86 ± 2.01	50.3%	42.3%
	GraphDx (Ours)	**9.24 ± 0.57**	**1065.78 ± 469.34**	5.06 ± 1.51	**88.3%**	**4.3%**

Table 3: Diagnostic Performance Comparison on MIMIC-IV Dataset

Base Model	Method	Score (0-10) ↑	Cost ($) ↓	Turns	Success Rate ↑	Harmful Rate ↓
DeepSeek-V3	Standard LLM	6.60 ± 0.93	737.43 ± 267.38	5.41 ± 0.87	56.8%	18.8%
	MAI-DxO	5.57 ± 1.43	1497.13 ± 844.00	11.28 ± 3.53	45.0%	30.5%
	GraphDx (Ours)	**8.27 ± 0.51**	**340.98 ± 129.84**	5.19 ± 0.93	**83.0%**	**5.7%**
Kimi-k2	Standard LLM	6.24 ± 1.08	1040.60 ± 539.29	4.02 ± 0.57	54.2%	23.2%
	MAI-DxO	5.19 ± 1.45	956.93 ± 559.68	11.97 ± 3.71	45.0%	36.5%
	GraphDx (Ours)	**8.06 ± 0.65**	**658.48 ± 246.00**	5.30 ± 1.26	**79.3%**	**7.5%**
Llama-3.3	Standard LLM	6.14 ± 1.00	1164.27 ± 521.71	4.34 ± 0.66	50.3%	22.0%
	MAI-DxO	3.56 ± 1.43	1820.21 ± 803.68	17.53 ± 1.65	30.8%	57.2%
	GraphDx (Ours)	**8.55 ± 0.67**	**711.93 ± 331.54**	6.60 ± 1.85	**82.2%**	**5.7%**

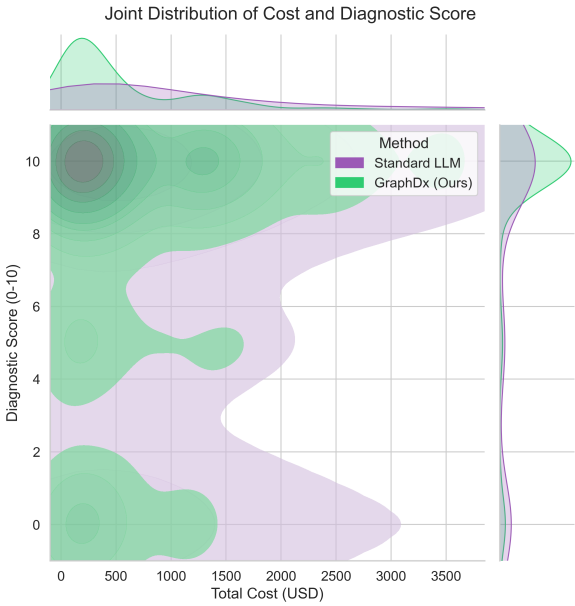


Figure 2: Joint distribution of Cost vs. Diagnostic Score for DeepSeek-V3 model.

These findings confirm the robustness of our hybrid design. Although the specific disease node is missing, the "Unaligned Features" mechanism (Algorithm 2) ensures that clinical observations are still captured and passed to the LLM. Furthermore, the graph's general structure of feature-test relationships remains valid (e.g., "Chest Pain" suggests "ECG" regardless of the specific underlying pathology), allowing the agent to conduct efficient workups before falling back on the LLM's parametric knowledge for the final diagnosis.

4.1.3 Human Evaluation of Simulation Quality

To address concerns regarding the reliance on simulation and LLM-based evaluation, we conducted a rigorous validation study to ensure clinical validity. Medical experts reviewed 365 diagnostic scenarios and 143 medical test cost estimates generated by our system. We analyzed the consistency between human experts and our automated LLM judge. The Pearson correlation coefficient for **Diagnostic Score** was **0.89** (n=365), with a Mean Absolute Error (MAE) of **0.58** (on a 0-10 scale) and a **94.8%** agreement rate on success/failure classification (Score ≥ 8). For **Test Cost**, the correlation was **0.89** (n=143) with an MAE of **$226.14**. These results provide strong empirical evidence that our automated evaluation pipeline is a reliable proxy for human expert judgment.

4.1.4 Visualization of Reasoning Process

Figure 3 illustrates the evolution of the Ground Truth (GT) disease rank and Top-k hit rate over dialogue turns using DeepSeek-V3. We observe two key trends: (1) **Rapid Convergence:** In early

turns (1-3), the GT rank drops sharply and Top-5 hit rate exceeds 50%, showing the graph's ability to quickly lock onto relevant diseases. (2) **Effective Differential Diagnosis:** In later turns (4-8), the Top-1 hit rate rises above 40%, confirming that our utility-based strategy effectively pinpoints the correct diagnosis through discriminative feature collection.

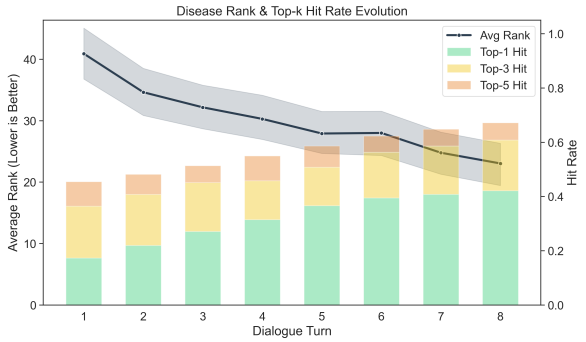


Figure 3: Evolution of GT Disease Rank and Top-k Hit Rate. The blue line (left) shows the decline in average rank, while bars (right) show the increasing probability of GT entering the Top-1/3/5 list.

4.1.5 Ablation Study

To quantify the contribution of each component (MDKG knowledge base and inference engine) in GraphDx, we conducted an ablation study on the Llama-3.3 model using the MedQA-Extended dataset. We designed two variants:

- **w/o Graph:** Removes the real MDKG and instead uses the LLM to "hallucinate" a graph summary based on dialogue history. This tests the value of structured knowledge itself.
- **w/o Inference:** Retains the real MDKG for retrieving neighbor nodes but removes evidence scoring and utility-based planning, relying entirely on the LLM for decision-making. This tests the value of the explicit inference engine.

The experimental results are shown in Table 4.

Table 4: Detailed Ablation Study Results on Llama-3.3

Method	Score ↑	Cost ($) ↓	Success Rate ↑	Harmful Rate ↓
Baseline	7.51 ± 1.30	1857 ± 773	65.7%	12.5%
w/o Graph	8.00 ± 1.21	1710 ± 739	71.5%	10.5%
w/o Inference	8.39 ± 1.08	1286 ± 642	77.5%	11.2%
GraphDx (Full)	**9.24 ± 0.57**	**1066 ± 469**	**88.3%**	**4.3%**

The analysis is as follows:

1. **Necessity of Structured Knowledge:** Compared to Baseline, **w/o Graph** shows a slight performance improvement (Score 7.51 → 8.00), indicating that even a hallucinated structured summary helps the LLM organize its thoughts. However, **w/o Inference** (using the real graph) further improves performance (Score → 8.39, Cost ↓ 25%), proving that accurate domain knowledge (MDKG) is far superior to the LLM's parametric memory.
2. **Critical Role of Explicit Reasoning:** The full GraphDx achieves the largest performance leap compared to **w/o Inference** (Score → 9.24, Cost ↓ 17%). This demonstrates that knowledge alone is not enough; the explicit inference engine based on evidence scoring and utility effectively guides the agent to make optimal decisions in complex uncertain environments, avoiding the LLM's blind trial-and-error.

4.1.6 Parameter Sensitivity Analysis

To assess the robustness of our mechanism, we conducted an offline sensitivity analysis on the interaction logs. We re-evaluated the disease ranking logic by varying key hyperparameters. The results, detailed in Appendix H.2, show that the system maintains stable performance across a wide range of parameters, demonstrating robustness.

4.1.7 Case Study

To deeply understand the decision-making advantages of GraphDx in different clinical contexts, we selected representative cases for detailed comparative analysis. These cases reveal typical failure modes of baseline models in **rare disease reasoning**, **anatomical logic**, and **cost-effective diagnosis**, while GraphDx successfully avoids these pitfalls through structured knowledge. Detailed analysis and visualization are provided in Appendix H.3.

5 Conclusion

This paper proposes GraphDx, which effectively solves the "Knowledge-Reasoning Mismatch" problem of LLMs in sequential diagnosis by automatically constructing a medical knowledge graph and combining it with knowledge-enhanced reasoning. Experiments demonstrate that this method not only significantly improves the accuracy and economy of diagnosis on standard medical exam datasets but also exhibits excellent generalization and robustness on real-world MIMIC-IV clinical data, providing a feasible path for the interpretability of medical AI. Future work will focus on extending this framework to multi-modal data fusion and larger-scale clinical deployment validation.

Limitations

Despite the excellent performance of GraphDx in experiments, there are still some limitations: (1) **Graph Construction Depends on Base Model Capability**: The quality of MDKG is limited by the medical knowledge reserve of the LLM used for graph construction (DeepSeek-V3 in this paper). If the base model has knowledge blind spots, the graph may miss key nodes or edges. (2) **Limited Multi-modal Processing Capability**: The current framework mainly processes textual information. The understanding of medical imaging (CT/MRI) and waveform data (ECG) still relies on the text descriptions from LLMs, and end-to-end pixel-level feature extraction has not yet been achieved. (3) **Inference Speed**: Due to the introduction of explicit evidence scoring and utility calculation, the single-step decision latency of GraphDx is slightly higher than that of pure end-to-end LLMs. Future work needs to further optimize the computational efficiency of the inference engine. (4) **Simulation Gap**: While our simulator is designed to be realistic, simulated patients may exhibit more consistent logic than real patients, who might provide vague or misleading descriptions. (5) **LLM-based Evaluation Bias**: Our simulator, judge, and cost estimator are LLM-based components and may inherit biases (e.g., stylistic preferences or calibration drift). While we mitigate this via a fixed evaluation protocol and human validation (Section 4), future work should incorporate multi-judge ensembles and real-world clinical evaluation when feasible.

Ethics Statement

This research involves the development of an automated medical diagnosis system. We emphasize that GraphDx is intended for research and educational simulation purposes only and must not be used for direct clinical decision-making without human oversight. The datasets used (MedQA and MIMIC-IV) are de-identified public datasets, ensuring patient privacy. While the automated graph construction reduces manual effort, it may inherit biases or outdated knowledge from the base LLM; therefore, any deployment in real-world settings requires rigorous human-in-the-loop verification. Furthermore, while our system optimizes for cost-efficiency, we acknowledge the ethical trade-off between reducing costs and the potential risk of missing necessary tests, which warrants further investigation.

Acknowledgement

We thank the reviewers and the area chair for constructive feedback. This work was supported by the National Key Research and Development Program of China under Grant No.2024YFE0212000, National Natural Science Foundation of China under Grant No.62402294.

References

Sydney Anuyah, Mehedi Mahmud Kaushik, Sri Rama Krishna Reddy Dwarampudi, Rakesh Shiradkar, Arjan Durresi, and Sunandan Chakraborty. 2025. Automated knowledge graph construction using large language models and sentence complexity modelling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 15526–15550, Suzhou, China. Association for Computational Linguistics.

Bohan Chen and Andrea L. Bertozzi. 2023. Autokg: Efficient automated knowledge graph generation for language models. *Preprint*, arXiv:2311.14740.

Silas Ruhrberg Estévez, Nicolás Astorga, and Mihaela van der Schaar. 2025. Timely clinical diagnosis through active test selection. *Preprint*, arXiv:2510.18988.

Y. Gao, R. Li, E. Croxford, J. Caskey, B. Patterson, M. Churpek, T. Miller, D. Dligach, and M. Afshar. 2025. Leveraging medical knowledge graphs into large language models for diagnosis prediction: Design and application study. *JMIR AI*, 4:e58670.

Zhihao Jia, Mingyi Jia, Junwen Duan, and Jianxin Wang. 2025. Ddo: Dual-decision optimization for llm-based medical consultation via multi-agent collaboration. *Preprint*, arXiv:2505.18630.

Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, Liwei H. Lehman, Leo A. Celi, and Roger G. Mark. 2023. MIMIC-IV, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1.

Harsha Nori, Mayank Daswani, Christopher Kelly, Scott Lundberg, Marco Tulio Ribeiro, Marc Wilson, Xiaoxuan Liu, Viknesh Sounderajah, Jonathan Carlson, Matthew P Lungren, Bay Gross, Peter Hames, Mustafa Suleyman, Dominic King, and Eric Horvitz. 2025. Sequential diagnosis with language models. *Preprint*, arXiv:2506.22405.

Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. 2023. Capabilities of gpt-4 on medical challenge problems. *Preprint*, arXiv:2303.13375.

Pengcheng Qiu, Chaoyi Wu, Junwei Liu, Qiaoyu Zheng, Yusheng Liao, Haowen Wang, Yun Yue, Qianrui Fan,

Shuai Zhen, Jian Wang, Jinjie Gu, Yanfeng Wang, Ya Zhang, and Weidi Xie. 2025. Evolving diagnostic agents in a virtual clinical environment. *Preprint*, arXiv:2510.24654.

Ali Sarabadani, Maryam Abdollahi Shamami, Hamidreza Sadeghsalehi, Borhan Asadi, and Saba Hesaraki. 2025. Dkg-llm : A framework for medical diagnosis and personalized treatment recommendations via dynamic knowledge graph and large language model integration. *Preprint*, arXiv:2508.06186.

Samuel Schmidgall, Rojin Ziaei, Carl Harris, Eduardo Reis, Jeffrey Jopling, and Michael Moor. 2025. Agentclinic: a multimodal agent benchmark to evaluate ai in simulated clinical environments. *Preprint*, arXiv:2405.07960.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Abubakr Babiker, Nathanael Schärli, Aakanksha Chowdhery, Philip Mansfield, Dina Demner-Fushman, and 13 others. 2023. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180.

A Prompts for Automated Graph Construction

We provide the key system prompts used in the MDKG construction pipeline. These prompts are designed to guide the LLM in extracting structured medical knowledge from its internal parameters.

A.1 Disease Knowledge Extraction

This prompt instructs the LLM to extract key diagnostic features (symptoms, signs, risk factors) for a given disease and estimate their typicality.

System Prompt:
You are a physician building a diagnostic knowledge graph. Input: ONE disease name (and optional reference text). Output: ONE JSON object with: - "disease": canonical name of this disease. - "features": an array of feature objects: - "name": short, canonical, self-explanatory term. No extra adjectives, no explanations. E.g. "fever", "chest pain", "elevated troponin", "chest X-ray consolidation". - "category": one of: "symptom", "test_result", "risk_factor", "other". - "typicality": one of "++", "+", "-", "–", meaning P(F|D): * "++": almost all patients with this disease have this feature (P(F|D) ≈ 1). Missing it makes the disease very atypical. * "+": often present in this disease (P(F|D) clearly higher than baseline). * "-": often absent in this disease (P(F|D) clearly lower than baseline). * "–": almost never present (P(F|D) ≈ 0). Seeing it strongly suggests other diseases. Important: "typicality" means frequency in this disease, NOT severity. - "tests": list of test names that DIRECTLY help observe or confirm THIS FEATURE, not "all tests for this disease". If none, use [].
Rules: - Focus on important diagnostic features, not every trivial detail. - Prefer guideline/textbook-style canonical terminology. - You may use your medical knowledge beyond the given text but stay consistent.
Return ONLY the JSON, no extra text.

A.2 Term Alignment

This prompt is used to decide whether a new term should be merged with an existing concept in the graph.

System Prompt:
You are an ontology curator for a medical knowledge graph. Task: Given one NEW term and a list of existing candidate concepts, decide if the new term: - is the SAME concept (same meaning) as exactly one candidate, or - should be a NEW concept.
Rules: - Treat synonyms, abbreviations, spelling variants, plural/singular, and minor wording changes as the same concept. - If the new term is clearly broader or narrower than all candidates in a clinically important way (e.g. "lung cancer" vs "small cell lung cancer"), treat as NEW. - If multiple candidates could match, choose the single best one; only use NEW when all are clearly wrong.
Output: - ONE JSON object: { "decision": "<candidate_id>" } or { "decision": "NEW" }. - No other keys, no explanations.

A.3 Test Metadata Enrichment

This prompt generates cost and invasiveness metadata for new test nodes.

System Prompt:
You are enriching metadata for diagnostic tests.
Input: ONE test name. Output: ONE JSON object with: - "cost": short qualitative label of typical cost, e.g. "very low", "low", "moderate", "high", or "very high". No numeric prices. - "invasiveness": one of ["low", "moderate", "high"]: * "low" – non-invasive, minimal risk (e.g. X-ray, blood test). * "moderate" – minimally invasive (needle, catheter, contrast...). * "high" – clearly invasive or interventional/surgical. - "applicable_population": brief description of typical use population, e.g. "adults with suspected acute coronary syndrome".
Be approximate but clinically reasonable. Return ONLY the JSON with these three keys.

A.4 Feature-Test Effectiveness Inference

This prompt determines the effectiveness of a test in confirming a specific feature.

System Prompt:
You are linking features to diagnostic tests.
Task: Given ONE feature and ONE test, decide how effective the test is for confirming or excluding that FEATURE itself (not any disease).
Output: JSON with one key: "effectiveness", value in ["++", "+"].
Meaning: - "++": The test directly measures or visualizes this feature with high reliability. It is the main or standard way to determine whether the feature is present. - "+": The test only indirectly supports or weakly relates to this feature. It helps but is not decisive alone.
Guidelines: - Judge the relation TEST ↔ FEATURE, not TEST ↔ disease. - When unsure or the link is not direct, choose "+" rather than "++". - Do NOT use values other than "++" or "+".
Return ONLY: { "effectiveness": "++" } or { "effectiveness": "+" }.

B Prompts for Graph-Augmented Agent

B.1 Information Extraction

This prompt extracts structured observations from the patient-doctor dialogue.

System Prompt:
You are a medical information extraction assistant with expertise in clinical terminology.
Task: Given the latest turn of conversation with a patient (which may include answers to questions, or test results), extract ONLY the NEW observations mentioned in THIS TURN.
CRITICAL: You MUST translate all observations into STANDARD MEDICAL TERMINOLOGY. Do NOT use colloquial or layperson descriptions. The extracted terms will be matched against a medical knowledge graph that uses professional clinical vocabulary.
Output a JSON object with: { "observations": [{ "name": "...", // MUST use standard medical terms "category": "symptom" | "test_result" | "risk_factor" | "other", "status": "present" | "absent" | "unknown", "source": "patient" | "test" | "other" }, ...], "tests_mentioned": [...], "candidate_diseases": [...] }
Guidelines: - ONLY include observations that are NEW or UPDATED in THIS TURN, not previous turns. - ALWAYS convert colloquial descriptions to standard medical terminology. - If the patient clearly denies a symptom (e.g. "No fever"), set status="absent". - If a symptom is clearly present or strongly implied, set status="present". - If it's unclear whether the feature is present, use status="unknown". - Map lab/imaging findings to category="test_result".

B.2 Agent Decision Making

This system prompt guides the final decision-making of the agent, incorporating the graph summary.

System Prompt:
You are an expert diagnostician AI with access to an internal diagnostic knowledge graph. The graph provides: - relationships between diseases and features (symptoms, test results, risk factors) with typicality ("++", "+", "-", "–"), - relationships between features and tests, with a qualitative measure of how directly the test measures the feature ("++" or "+"), - simple metadata about test cost and invasiveness.
You are interacting with a simulated patient to diagnose their condition. Each turn you receive NEW information from the environment (patient answers or test results). The system may also provide you with a GRAPH SUMMARY containing: - a list of structured observations parsed from the conversation, - a graph-based ranking of candidate diseases, - graph-based suggestions of useful follow-up questions and tests.
IMPORTANT: - The knowledge graph may be incomplete or imperfect. Its suggestions are HINTS, not truth. - You must combine the graph suggestions with your own clinical reasoning and the full history. - Do NOT blindly follow the graph if it conflicts with clinical reasoning. - You have a maximum of {max_turns} turns; try to be efficient and accurate.
Available actions: 1. "ask" – Ask the patient a follow-up question to gather more information. 2. "test" – Order a medical test. 3. "diagnose" – Provide a final diagnosis if you think you have enough information.
You MUST respond in JSON with two keys: "action_type" and "action_content".

C Algorithms

We provide the detailed pseudocode for the two core algorithms of `GraphDx`.

D Detailed Experimental Setup

D.1 Environment Setup

We developed `ClinicSim`, a modular simulation environment designed to evaluate sequential diagnostic capabilities. It integrates the dynamic interaction mechanism of `AgentClinic` (Schmidgall et al., 2025) with the cost-sensitive evaluation framework of `SDBench` (Nori et al., 2025). Specifically, `ClinicSim` consists of two core components: (1) **Patient Actor**: An LLM-driven patient agent that conducts realistic multi-turn natural language dialogues based on a complete clinical profile. It is instructed to be "passive," revealing information only when explicitly asked, to mimic the information asymmetry in real consultations; (2) **Test Result Generator**: An independent generation module responsible for returning corresponding medical test results (e.g., blood metrics, imaging descriptions) based on doctor orders. To ensure consistency, it retrieves results directly from the ground truth profile; if a specific value is missing, it returns a "normal" or "unknown" status consistent with the patient's overall condition. The maximum number of dialogue turns is set to 20. This

Algorithm 1: Automated MDKG Construction Pipeline

Input: Disease List $\mathcal{D}_{list}$, LLM $\mathcal{M}$
Output: MDKG $\mathcal{G}$

Initialize empty graph $\mathcal{G}$;
// Stage 1: Concurrent Disease Knowledge Extraction
foreach $d \in \mathcal{D}_{list}$ *(Concurrent)* **do**
  $(F_d, \mathcal{T}_{map}) \leftarrow \mathcal{M}$.extract_knowledge$(d)$ // Extract features & test map
  $P_d \leftarrow \mathcal{M}$.estimate_typicality$(F_d, d)$;
end
// Stage 2: Sequential Entity Alignment & Graph Construction
foreach $d \in \mathcal{D}_{list}$ **do**
  $v_d \leftarrow$ HybridAlign$(d, \mathcal{G}.\mathcal{V})$;
  if $v_d \in \mathcal{G}.\mathcal{V}_f$ **then**
    UpgradeNode(v_d) // Promote Feature to Disease
  end
  foreach $f \in F_d$ **do**
    $v_f \leftarrow$ HybridAlign$(f, \mathcal{G}.\mathcal{V}_f)$ // Hybrid Alignment
    $\mathcal{G}$.add_edge$(v_d, v_f, \text{weight} = P_d[f])$;
    foreach $\tau \in \mathcal{T}_{map}[f]$ **do**
      $v_\tau \leftarrow$ HybridAlign$(\tau, \mathcal{G}.\mathcal{V}_t)$ // Test Alignment
      $\mathcal{G}$.add_edge(v_f, v_τ);
    end
  end
end
// Stage 3: Metadata Enrichment & Edge Inference
foreach $\tau \in \mathcal{G}.\mathcal{V}_t$ *(Concurrent)* **do**
  attr$_\tau \leftarrow \mathcal{M}$.enrich_metadata$(\tau)$ // Cost/Invasiveness
  $\mathcal{G}$.update_node(τ, attr_τ);
end

Algorithm 2: Knowledge-Enhanced Diagnostic Loop

Input: Dialogue History H_t, MDKG $\mathcal{G}$
Output: Action a_t
// 1. Info Extraction & State Tracking
$O_t \leftarrow$ ExtractObservations(H_t);
UpdateState$(\mathcal{G}, O_t)$ // Update Confirmed/Excluded Features
// 2. Evidence Scoring (Heuristic)
foreach $d \in \mathcal{G}.\mathcal{V}_d$ **do**
  $S(d) \leftarrow \sum_{f \in O_t^+} w(d,f) - \lambda \cdot \sum_{f \in O_t^-} w(d,f)$;
end
$D_{top} \leftarrow$ TopK(S);
// 3. Utility-Based Action Suggestion
$A_{suggest} \leftarrow \emptyset$;
$T_{cand} \leftarrow \{\tau \mid \exists f \in \text{Neighbors}(D_{top}), (f,\tau) \in E_{test}\} \setminus T_{history}$;
foreach $\tau \in T_{cand}$ **do**
  $V_{total} \leftarrow \sum_{f \in \text{Verifies}(\tau)} V(f) \cdot \eta(f,\tau)$;
  Utility$(\tau) \leftarrow V_{total}/(1 + \text{Cost}(\tau) \cdot \text{Inv}(\tau))$;
  $A_{suggest}$.add$(\tau, \text{Utility}(\tau))$;
end
// 4. LLM Decision Making
$S_{summary} \leftarrow$ FormatSummary$(O_t, D_{top}, A_{suggest})$;
$a_t \leftarrow$ LLM.decide$(H_t, S_{summary})$;
return a_t;

design allows us to track the economic cost of each step in real-time, enabling a comprehensive evaluation of agent performance in the "Accuracy-Cost-Efficiency" space. The source code for `GraphDx` and the `ClinicSim` environment is available at https://github.com/ossiver-GEL/GraphDx.

D.2 Datasets

We utilized two datasets for evaluation: (1) **MedQA Dataset:** Derived from the `medqa_extended` dataset (Schmidgall et al., 2025), covering diverse chief complaints, medical histories, and personality traits, totaling 200 test cases. (2) **MIMIC-IV Dataset:** To verify robustness on real-world data, we constructed 200 scenarios from the MIMIC-IV clinical database (Johnson et al., 2023) following the AgentClinic (Schmidgall et al., 2025) methodology. These cases contain real clinical noise, incomplete information, and complex medical histories, posing a greater challenge to the agent's information extraction and reasoning capabilities.

D.3 Parameter Settings

In our experiments, we set the typicality weights as $w_{++} = 4.0$, $w_{+} = 1.5$, $w_{-} = -1.5$, and $w_{--} = -4.0$. The penalty coefficient λ was set to 0.6. For the utility calculation, the verification effectiveness weights were set to $w_{eff} \in \{1.0, 0.5\}$. The cost levels $C(t)$ were mapped to integers $\{1, 2, 3, 4, 5\}$ representing "very low" to "very high" costs, and the invasiveness multipliers $M_{inv}(t)$ were set to $\{1.0, 1.5, 2.0\}$ for low, moderate, and high invasiveness, respectively. The semantic retrieval threshold τ was set to 0.8.

D.4 Baseline Settings

To verify the generality of our method, we evaluated it on three advanced Large Language Models: (1) **DeepSeek-V3**; (2) **Kimi-k2**; (3) **Llama-3.3**. For each model, we compared three settings:

- **Standard LLM (Baseline)**: Standard agent using direct prompting for inquiry and diagnosis.
- **MAI-DxO (Nori et al., 2025)**: The SOTA architecture proposed in SDBench, employing multi-agent collaboration (Dr. Test-Chooser, Dr. Challenger, etc.) for diagnostic orchestration.

- **GraphDx (Ours)**: The proposed graph-augmented knowledge-enhanced agent.

We note that several related sequential diagnosis frameworks (e.g., DDO (Jia et al., 2025), ACTMED (Estévez et al., 2025), and DiagGym (Qiu et al., 2025)) are highly relevant in motivation, but are not included as baselines because they rely on closed-set action spaces (predefined symptom/test lists) incompatible with our open-ended natural language dialogue setting. To ensure a fair comparison, we also enforce a controlled evaluation protocol where the `ClinicSim` environment (Patient Actor, Test Result Generator, and Judge) remains fixed across all methods, isolating the impact of the agent design.

D.5 Evaluation Metrics

We employ three core metrics to evaluate diagnostic performance: (1) **Diagnostic Score (0-10)**: Scored by Llama-3.3 as a judge model based on the clinical equivalence between the agent's final diagnosis and the ground truth (10: Optimal; 5: Incomplete but safe; 0: Incorrect and misleading). (2) **Cost ($)**: The sum of estimated costs for all medical tests ordered during the diagnosis. To ensure fair comparison, costs are estimated in USD by an independent LLM evaluator using a standardized price book prompt, rather than using the agent's internal optimization scores. (3) **Turns**: The number of dialogue turns between the doctor and patient, reflecting diagnostic efficiency. Note that for all metrics, we report the **Mean ± Standard Deviation** across **3 independent runs** for each scenario. The standard deviation measures the stability of the agent's performance, with randomness stemming from the LLM's decoding temperature.

E Hyperparameters and Constants

We list the specific numerical values used for the qualitative attributes in our system.

Table 5: Typicality Weights used in Evidence Scoring

Typicality Label	Symbol	Weight Value
Strongly Positive	++	4.0
Positive	+	1.5
Negative	-	-1.5
Strongly Negative	–	-4.0

Table 6: Cost Scores and Invasiveness Multipliers

Cost Label	Score	Invasiveness	Multiplier
Very Low	1.0	Low	1.0
Low	2.0	Moderate	1.5
Moderate	3.0	High	2.0
High	4.0		
Very High	5.0		

F Baseline Agent Prompts

To ensure fair comparison and reproducibility, we provide the system prompts used for the Baseline Agent. The Baseline Agent uses a standard Chain-of-Thought (CoT) approach without access to the external knowledge graph.

System Prompt:
You are an expert diagnostician AI. You are interacting with a patient to diagnose their condition. You will receive information in turns. Each turn, you will get new information, which could be the patient's response to your questions or the results of a medical test you ordered. Based on the information you have, you must decide on the next action to take. You have three possible actions: 1. **ask**: Ask the patient a follow-up question to gather more information. 2. **test**: Order a medical test to get specific data. 3. **diagnose**: If you have enough information, provide a final diagnosis.
You must respond in JSON format with two keys: - ‘"action_type"‘: One of ‘"ask"‘, ‘"test"‘, or ‘"diagnose"‘. - ‘"action_content"‘: The specific question, test name, or diagnosis.
For example: If you want to ask a question: ‘{"action_type": "ask", "action_content": "Have you experienced any fever?"}‘
If you want to order a test: ‘{"action_type": "test", "action_content": "Complete Blood Count"}‘
If you are ready to diagnose: ‘{"action_type": "diagnose", "action_content": "Common Cold"}‘
You have a maximum of {max_turns} turns to diagnose the patient. Strive to be efficient and accurate.

G Simulation Environment Details

We use a modular simulation environment `ClinicSim` to evaluate the agents. The environment consists of a Patient Actor and a Test Result Generator, both driven by LLMs.

G.1 Patient Actor Prompt

The Patient Actor simulates a real patient based on a detailed clinical profile. It is instructed to reveal information only when asked, mimicking the information asymmetry in real consultations.

System Prompt:
You are role-playing as the PATIENT (or the accompanying family member IF the patient is not capable of making decisions) described below. A doctor is examining you (or your dependent).
Rules: - Your profile (or your dependent's) contains very detailed medical information about you (or your dependent) but NEVER directly reveal it unless asked by the doctor. - NEVER directly reveal the true diagnosis to the doctor. - Answer the doctor's questions naturally and unprofessionally, as a real patient would. - ONLY share symptoms/history based on your profile when asked. - If unsure, say "I don't know" or give a plausible answer.
{profile}

G.2 Test Result Generator Prompt

The Test Result Generator provides realistic test results based on the patient's ground truth data.

System Prompt:
You are a medical test result generator. Given a test name, provide a realistic result for that test based on the patient's profile.
Rules: - Return realistic values CONSISTENT with the patient's condition. - Return EXACTLY the required test result, NOT all results from the profile. - If the test is not in the profile, generate the most CONSISTENT result based on the patient's condition. - Format: brief, factual lab report style. - NO diagnosis, NO interpretation, NO conversation.
{profile}

G.3 Evaluation Prompts

To ensure consistent and scalable evaluation, we utilize LLM-based judges for both diagnostic accuracy scoring and cost estimation.

G.3.1 Diagnostic Accuracy Evaluator

We employ an LLM-based evaluator to grade the diagnostic outcome against the ground truth.

System Prompt:
You are an expert medical evaluator. Compare the agent's diagnosis with the ground truth diagnosis. Provide a score from 0 to 10. Use this scale:

- 10 (Optimal): The agent's final diagnosis is clinically equivalent to the ground truth diagnosis.
- 5 (Acceptable): The agent's diagnosis is incorrect/incomplete, BUT it is safe and does not mislead future treatment (e.g., a correct referral, or a safe differential).
- 0 (Harmful): The agent's diagnosis is incorrect AND misleading, potentially causing harm or treatment delay.

Above descriptions are anchor points; intermediate scores (1-4, 6-9) should reflect gradations in clinical relevance and safety.
You MUST output a JSON object with ONLY the "score" field. No other fields or text allowed.
Example output: {"score": 8}
Full profile of the case: {profile_text}
Agent's Diagnosis: {agent_diagnosis}

G.3.2 Cost Estimation Methodology & Prompt

To evaluate the economic efficiency of diagnostic agents, we calculate the financial cost of each diagnostic session. Since real-world pricing varies continuously, we employ a Large Language Model (specifically `Llama-3.3-70b-instruct`) acting as a "Medical Billing Expert" to estimate the cost of each ordered test. The model is prompted to provide realistic US-based pricing (in USD) for each specific test name.

System Prompt:
You are a medical billing expert. For each of the following medical tests, provide an estimated cost in USD. Use realistic pricing based on typical costs in the United States healthcare system.
CRITICAL RULES:

1. You MUST present the output as a valid JSON object where keys are the test names and values are the estimated costs (as numbers).
2. Do not include any other text or explanations.
3. ONLY include tests that are explicitly listed below. Do NOT add any tests that are not in the list.

Example output when tests are ordered: { "Complete Blood Count (CBC)": 50, "Basic Metabolic Panel (BMP)": 75 }
Tests Ordered: {ordered_tests}

H Additional Experimental Analysis

H.1 MDKG Statistics

Before evaluating the diagnostic performance, we assessed the quality of the constructed MDKG through a rigorous manual verification process. We randomly sampled 243 triples (disease-feature associations) from the generated graph and asked medical professionals to classify them into three categories: *Correct* (medically accurate and precise), *Plausible* (reasonable but potentially vague or context-dependent), and *Incorrect* (medically false).

The evaluation results yielded 186 (76.5%) *Correct* triples, 37 (15.2%) *Plausible* triples, and 20 (8.2%) *Incorrect* triples. The overall validity rate (Correct + Plausible) reached 91.8%, demonstrating that our automated pipeline can reliably extract high-quality medical knowledge. The small fraction of incorrect triples mostly involved over-generalized associations (e.g., linking generic symptoms to specific rare diseases), which are often filtered out during the subsequent evidence scoring phase.

Figure 4 shows the distribution of key graph attributes. Figure 4a displays the distribution of fea-

ture typicality ($P(f|d)$), presenting a reasonable skewed distribution, indicating that the graph can distinguish between "hallmark features" (high typicality) and "common features." Figure 4b shows the distribution of test costs, covering a wide range from low-cost (e.g., CBC) to high-cost (e.g., MRI) tests, providing a basis for cost-sensitive planning. Figure 4c illustrates the degree distribution of disease nodes (i.e., number of features per disease), with an average of 15.7 features per disease, providing sufficient evidence for differential diagnosis.

H.2 Parameter Sensitivity Analysis

To assess the robustness of our evidence scoring mechanism while avoiding the prohibitive computational cost of re-running full agent simulations, we conducted an offline sensitivity analysis on the Llama-3.3 interaction logs. We re-evaluated the disease ranking logic by varying two key hyperparameters: the weight of strong positive evidence (w_{++}) and the penalty coefficient for missing expected features (λ).

Figure 5 illustrates the Top-1 accuracy of the ground truth disease across a grid of parameter settings. We observe that:

- **Robustness to Weight Variations:** The system maintains stable performance (Top-1 accuracy $\approx 60\%$, Top-5 accuracy $\approx 88\%$) across a wide range of w_{++} values (2.0–5.0). This indicates that the ranking is primarily driven by the structural correctness of the graph (presence of correct edges) rather than fine-tuned parameters.
- **Impact of Absence Penalty:** A lower absence penalty ($\lambda \in [0.0, 0.3]$) yields slightly better results than higher penalties. This suggests that in natural language dialogue, "missing" features are often simply unmentioned by the patient rather than clinically absent, so the system should be cautious about penalizing them too heavily.

H.3 Case Study

To deeply understand the decision-making advantages of `GraphDx` in different clinical contexts, we selected three representative cases from the MedQA-Extended dataset for detailed comparative analysis. These cases reveal typical failure modes of baseline models in **rare disease reasoning**, **anatomical logic**, and **cost-effective diagnosis**, while `GraphDx` successfully avoids these pitfalls through structured knowledge.

Case 1: Capturing Subtle Clues in Rare Diseases (Scenario 50) As shown in Figure 6, **Case Background:** A 16-year-old female presented with primary amenorrhea and delayed breast development. Notably, her history included "fractures from minor falls" and she was "tall for her age".

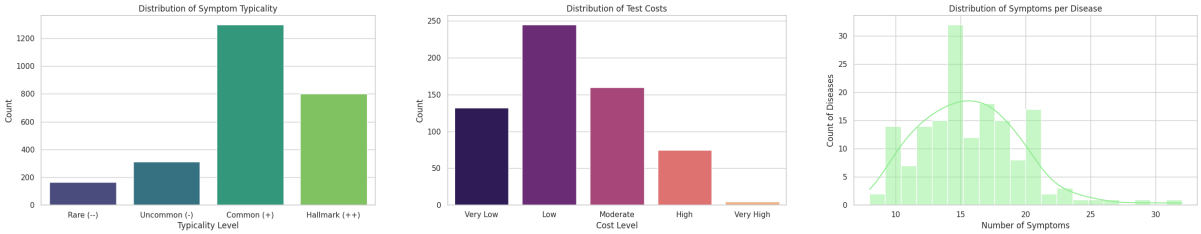
- **Baseline:** Focused solely on amenorrhea and delayed puberty, diagnosing "Turner Syndrome". It failed to recognize that Turner Syndrome typically presents with *short* stature, contradicting the patient's "tall" description. It also ignored the fracture history.
- **`GraphDx`:** The graph reasoning engine successfully linked "fractures" (osteoporosis) and "tall stature" (delayed epiphyseal closure) with "amenorrhea". It identified the rare etiology **"Aromatase Deficiency"** (inability to synthesize estrogen from androgens) and confirmed it via hormone panel and karyotype, achieving a precise diagnosis.

Case 2: Strict Anatomical and Functional Constraints (Scenario 14) **Case Background:** A patient reported a swollen right ring finger after a football injury, stating, "I can't bend the tip of my finger at all when I try to make a fist."

- **Baseline:** Confused the functional description, diagnosing "Mallet Finger" (an extensor tendon injury characterized by inability to *extend* the tip).
- **`GraphDx`:** The MDKG explicitly encodes the functional logic of musculoskeletal anatomy. It correctly interpreted the inability to *flex* the distal joint as a rupture of the Flexor Digitorum Profundus tendon, correctly diagnosing **"Jersey Finger"**.

Case 3: Cost-Effective Decision Making (Scenario 66) **Case Background:** A 1-month-old preterm infant presented with "apnea episodes" during sleep and "pale skin".

- **Baseline:** Reacted defensively to the high-risk symptom "apnea" by ordering an expensive **Polysomnography ($2500)**, diagnosing "Apnea of Prematurity" without investigating the root cause.
- **`GraphDx`:** Integrated the "pale skin" clue with "apnea" in the evidence-weighted graph, hypothesizing anemia as a potential cause. It prioritized



(a) Feature Typicality Distribution (b) Test Cost Distribution (c) Disease-Feature Degree Dist.

Figure 4: Statistical distribution of key attributes in MDKG.

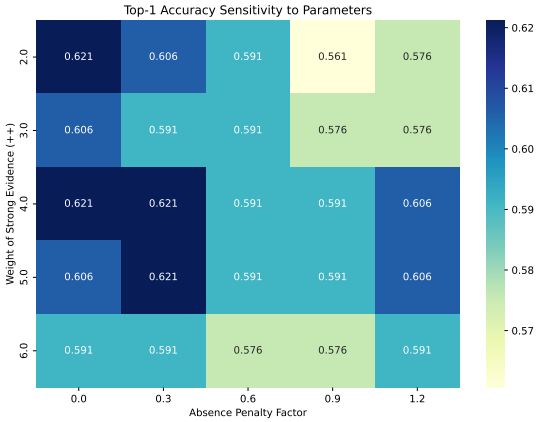


Figure 5: Sensitivity of Top-1 Accuracy to variations in Strong Evidence Weight (w_{++}) and Absence Penalty (λ). The system exhibits a stable performance plateau, demonstrating robustness.

a low-cost **Complete Blood Count ($50)**, revealing low hemoglobin (8.2 g/dL), and correctly diagnosed **"Anemia of Prematurity"**. This demonstrates how utility-based planning avoids unnecessary expensive testing.

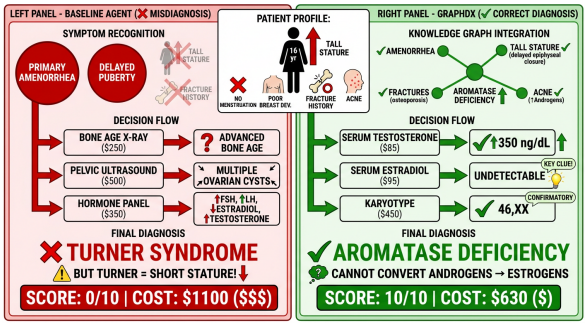


Figure 6: Typical Case Analysis (Scenario 50: Primary Amenorrhea). The Baseline agent (left) misdiagnosed **Turner Syndrome** by focusing on amenorrhea while ignoring the contradictory evidence of **tall stature** and **fractures**. In contrast, `GraphDx` (right) successfully linked these cues to **Aromatase Deficiency** via the structured graph, demonstrating the navigational role of structured knowledge.