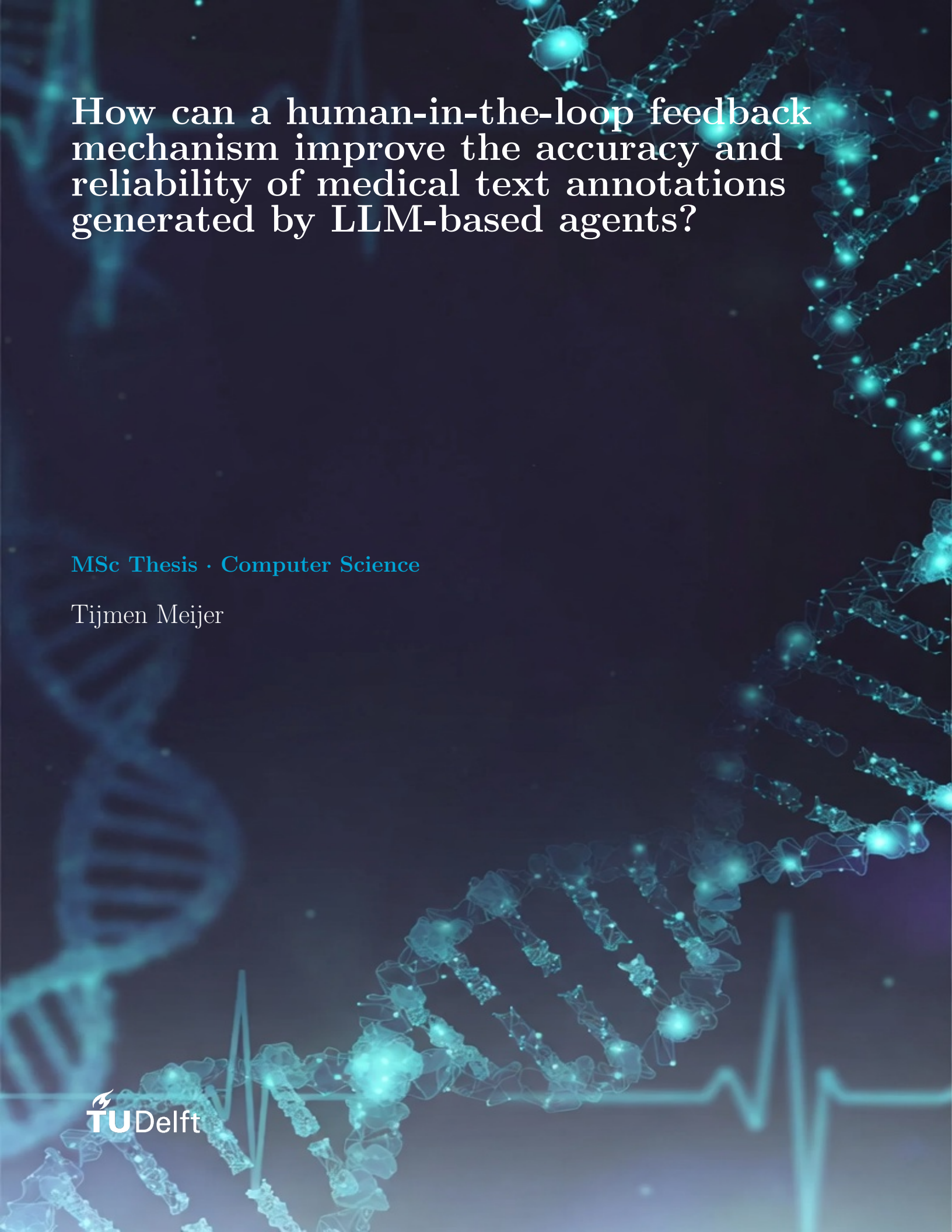

How can a human-in-the-loop feedback mechanism improve the accuracy and reliability of medical text annotations generated by LLM-based agents?

MSc Thesis · Computer Science

Tijmen Meijer

TU Delft

How can a human-in-the-loop feedback mechanism improve the accuracy and reliability of medical text annotations generated by LLM-based agents?

by

Tijmen Meijer
Student number: 5488311

in partial fulfilment of the requirements for the degree of

Master of Science
in Computer Science

at the Delft University of Technology,
to be defended publicly on Friday, 12 June 2026.

Thesis committee chair:	Prof. dr. ir. M. J. T. Reinders, TU Delft, Pattern Recognition & Bioinformatics
Daily supervisor:	D. Selani, TU Delft, Pattern Recognition & Bioinformatics
Committee member:	Dr. C. Lofi, TU Delft, Web Information Systems
Faculty:	Faculty of Electrical Engineering, Mathematics and Computer Science, Delft

Cover image generated with xAI Grok (2026).
An electronic version of this thesis is available at
`https://github.com/surftijmen/agentic-hitl-medical-annotation`.

TU Delft

Contents

List of Figures

List of Tables

How can a human-in-the-loop feedback mechanism improve the accuracy and reliability of medical text annotations generated by LLM-based agents?

Tijmen Meijer (5488311)
Delft University of Technology

June 2026

Abstract

Large language models evaluated on medical text annotation tasks are typically scored against a single reference label, such as the primary billed ICD-9 code in MIMIC-III. This standard imposes a measurement ceiling on strict accuracy: it conflates genuine clinical extraction failure with cases where the model extracted a valid clinical comorbidity rather than the administrative billing-convention primary. To address this, we propose a dual-metric evaluation protocol that pairs strict accuracy (matching the primary billed code) with lenient accuracy (matching any billed code for the admission), scoped to clinical ICD-9 chapters, to separate model capability from billing-convention noise. Exercising the protocol through a human-in-the-loop (HITL) framework, we demonstrate an 18 percentage-point gap between strict (0.67) and lenient (0.85) accuracy on a human-reviewed baseline ($n = 100$); direct on-corpus analysis further shows that approximately 10% of discharge summaries (up to 14% in early iterations) contain gold-label-ambiguous coding artefacts rather than recoverable errors. On the same HITL framework, lenient accuracy converges to a tight $[0.86, 0.90]$ band by iteration 3 across five prompt-engineering conditions, a cross-condition convergence we read as a framework-level reliability gain, and a human-reviewer trajectory shows strict trending $0.62 \rightarrow 0.78$ under rich free-text feedback (a directional signal, confounded with batch difficulty at this sample size, that motivates the higher-powered replication we propose). Future LLM-based ICD-coding evaluations on MIMIC-III should adopt the dual-metric protocol so that model capability and billing-convention noise are reported separately.

1 Introduction

The rapid advancement of large language models (LLMs)[1] has created significant opportunities to automate medical text annotation tasks such as extracting diagnoses, adverse drug reactions, and clinical relationships from electronic health records. Despite impressive performance on general-domain tasks, LLMs often struggle with the specialised requirements of medical annotation: domain-specific terminology, complex clinical reasoning, and the high reliability standards required in healthcare applications.

Traditional approaches rely on fully supervised machine learning with large volumes of manually annotated training data. These approaches are resource-intensive and struggle to adapt when LLMs make systematic errors due to ambiguous task instructions or insufficient domain context. Simply adding more training examples does not address the underlying causes of annotation failure.

This paper investigates an iterative human-in-the-loop (HITL) framework that leverages qualitative human feedback to systematically improve LLM-based medical text annotation through prompt refinement. Rather than retraining models, our approach recognises that many annotation failures stem from suboptimal task specification and can be addressed through iterative clarification of the annotation instructions themselves.

We report this work as a **pilot study**: the HITL trajectories are run at $n = 50$ samples per iteration, which is deliberately chosen to characterise protocol-level signals (the strict/lenient gold-label ceiling, the dominant failure-mode budget, the on-corpus gold-label-ambiguity rate) rather than to resolve per-technique differences statistically. The sample size is the principal limitation of the present study, and a higher-powered follow-up (at $n = 300$ per iteration with a fixed held-

[1] A full list of abbreviations used in this paper is provided in Appendix A.

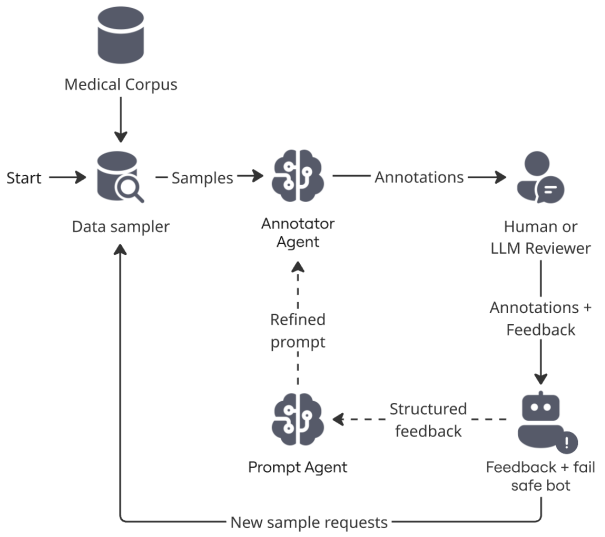

Figure 1: Two-agent HITL framework: the universal core shared by all five conditions. The Annotator extracts the primary diagnosis from a sampled discharge summary; a reviewer (human or LLM-as-judge) produces a verdict, a failure-mode label, and a free-text rationale; recoverable failure modes drive PromptAgent instruction-level prompt edits while ambiguous-case failures are excluded. The refined prompt drives the next iteration.

out test set) is named explicitly in §6.1. We flag this upfront so the reader can read quantitative comparisons across techniques as qualitative trends rather than statistical inferences.

The proposed framework (Figure 1) consists of two primary LLM-based agents working with human domain experts. The Annotator Agent performs extractions using a structured prompt. Human experts, or, at evaluation scale, a stronger LLM acting as a surrogate reviewer, then review annotations, identifying not only errors but diagnosing their root causes (for example, "the model confuses symptoms with diagnoses"). Categorised feedback is passed to a PromptAgent that translates it into refined prompts. Human experts review and approve prompt updates before deployment, maintaining a closed feedback loop aligned with clinical expertise. We evaluate the framework on English-language discharge summaries from the MIMIC-III database [7] with primary ICD-9 codes as gold reference.

This work makes three contributions. First, a two-agent HITL framework combining an Annotator with a PromptAgent, organised around a six-class failure-mode taxonomy that turns the reviewer's categorised rationales into instruction-level prompt patches and holds gold-label-ambiguous cases out of the loop, complemented by a retrieval-augmented memory of validated and corrected past cases. Second, a dual-metric evaluation protocol that pairs strict accuracy (against the primary billed ICD-9) with lenient accuracy (against any billed ICD-9 for the admission), scoped to the clinical ICD-9 chapters, isolating model capability from billing-convention artefacts. Third, an empirical study on this corpus that exposes a measurable upper bound on strict-accuracy evaluation: a +18pp clinical strict/lenient gap at a human-reviewed single-pass baseline ($n = 100$); an HITL trajectory whose lenient accuracy converges to $[0.86, 0.90]$ by iteration 3 across five prompt-engineering techniques while per-technique strict differences are qualitative at this sample size; and a researcher-led human-in-the-loop validation trajectory at $n = 50 \times 3$ in which $\approx$10% of stratified discharge summaries (up to 14% in the early iterations) were marked as gold-label-ambiguous coding artefacts rather than recoverable clinical errors. Failure-mode analysis identifies missed entity as the modal failure mode in 14 of 15 iterations (mean 67%, range 42–91%) and instruction-level prompt patching as the dominant feedback lever.

Strict-accuracy gains are bounded by gold-label noise rather than by prompt-engineering or model capacity, so the methodological contribution is the dual-metric protocol that surfaces and quantifies the ceiling; the framework itself is the test bed that exposes it.

Our overall research question is: *how can a human-in-the-loop feedback mechanism improve the accuracy and reliability of medical text annotations generated by LLM-based agents?* We decompose it into four sub-questions:

- **SQ1:** which types of human feedback are most effective for prompt refinement?
- **SQ2:** which prompt-engineering techniques (few-shot, chain-of-thought, self-consistency, retrieval-augmented generation) benefit most from HITL integration?
- **SQ3:** how does prompt complexity (length, instruction count, structural composition) evolve across HITL iterations, and how does it relate to accuracy?
- **SQ4:** how many iteration cycles are required for convergence, and does the convergence behaviour generalise across clinical text types?

SQ1 is addressed by the failure-mode routing experiment (§3.3.3, §4.3); SQ2 by the per-technique HITL longitudinal study (§4.2); SQ3 and SQ4 by the within-corpus prompt-complexity and convergence analyses in §4.5. SQ3 is answered descriptively across the two Baseline trajectories whose per-iteration prompts are

authoritatively recoverable; a formal complexity-vs-accuracy regression is not statistically supportable at $n = 50$ per iteration. SQ4 answers the within-corpus half (lenient converges in three iterations; strict does not settle within that window) and defers cross-corpus generalisation to §6.1.

2 Related Work

This work intersects several research areas: active learning in medical contexts, agentic AI systems with feedback loops, prompt engineering methodologies, and human-in-the-loop machine learning.

Active learning addresses the challenge of acquiring labelled training data by strategically selecting the most informative samples for annotation. Budd et al. [2] provide a comprehensive survey of deep active learning in medical image analysis, demonstrating that uncertainty sampling, diversity sampling, and hybrid approaches can substantially reduce annotation burden while maintaining model performance. While their focus is on medical imaging, these principles inform the Data Sampler component in our architecture. A key distinction is that active learning traditionally selects samples for model training, whereas our approach uses strategic sampling to identify annotation errors that inform prompt refinement, a more interpretable and computationally efficient improvement mechanism.

Recent work has demonstrated the effectiveness of multi-agent architectures where specialised AI agents collaborate through feedback loops. Research on multi-AI agent systems has shown that iterative refinement through LLM-driven feedback can significantly improve solution quality by employing distinct agents with complementary roles [6, 22]. This architectural principle underpins our dual-agent design, in which an Annotator Agent performs extractions while a separate PromptAgent specialises in translating human feedback into prompt improvements. The separation of concerns allows each agent to optimise for its specific function without conflicting objectives.

Effective prompt design is critical for LLM performance on specialised tasks. White et al. [21] catalogue reusable prompt patterns (persona, output automater, cognitive verifier, template, and others) drawn from software-engineering pattern-language methodology, codifying recurring prompt structures that practitioners apply across tasks. Our framework treats the prompt itself as the primary object of optimisation: rather than selecting patterns up front, it uses categorised human feedback to iteratively refine task specifications, clarify ambiguities, and add domain context within an existing prompt, so the prompt's structure is shaped by observed failure modes rather than a fixed template. The concept of self-improving agents that learn at test time with human guidance is directly realised in our HITL framework.

A closely related line of work automates prompt refinement directly. Automatic Prompt Optimization [15] treats natural-language critiques of model errors as textual "gradients" and edits the prompt in their direction, while PromptAgent [18] casts prompt search as Monte-Carlo tree planning over error-reflection steps. Both are *autonomous* optimisers: they run without a human in the loop and improve a prompt against a fixed scalar metric on general-purpose benchmarks. Our framework's prompt-refinement agent (which we likewise term a PromptAgent) shares their central move, turning observed failures into instruction-level prompt edits, but differs in three ways that matter for the clinical setting: the edits are driven by *human* (or human-proxy) review rather than self-generated critique, they are routed by a domain-specific failure-mode taxonomy rather than a single scalar reward, and they consume free-text clinical rationales as a first-class signal. The retrieval-augmented memory condition (§4.2) builds on retrieval-augmented generation [10], here retrieving over a growing store of past validated and corrected note-to-diagnosis pairs rather than a static knowledge corpus.

Direct application of LLMs to clinical IE without task-specific fine-tuning has been studied across several axes. Agrawal et al. [1] demonstrate that frontier LLMs are competitive few-shot clinical information extractors on structured extraction tasks, including medication, dosage, and adverse-event extraction, often within a few percentage points of fully-supervised encoder baselines. More recently, Maarseveen et al. [12] report that carefully engineered prompts let general-purpose LLMs reach expert-level rheumatic-disease diagnosis extraction from clinical records; the four-field prompt scaffold and three-key diagnosis output schema of our v1 prompt (Appendix B.2) are adapted from their released artefacts. The dominant alternative for ICD-coding-from-notes specifically is task-fine-tuned encoder pipelines: the PLM-ICD and CAML family [4] treat coding as multi-label classification and currently hold the published micro-F1 leaderboard on MIMIC-III (~ 0.598 micro-F1 in the best replicated setting). Our framework occupies a different operating point: we keep the underlying LLM frozen and treat the prompt as the only object of optimisation, trading the encoder pipeline's raw F1 ceiling for (i) zero-retraining adaptation when annotation guidelines change, (ii) human-inspectable instruction-level patches as the unit of improvement, and (iii) free-text rationales as a first-class feedback channel that supervised classification cannot consume.

Three specific gaps in the prior literature are addressed here. First, existing MIMIC-III ICD-

coding evaluations ([4] and the leaderboard tradition it summarises) report a single match metric against the billing-primary code and so inherit the billing-convention noise as model error; the strict+lenient dual-metric protocol we contribute decomposes that conflation. Agrawal et al. [1] demonstrate few-shot clinical IE feasibility but do not address the gold-label ambiguity that we surface empirically at ≈10% on this corpus. Second, automatic prompt-optimisation methods [15, 18] and multi-agent refinement frameworks [6, 22] apply iterative LLM-driven prompt editing to general-purpose tasks but optimise autonomously against a fixed metric: they neither condition routing on a domain-specific failure-mode taxonomy nor relate feedback richness to prompt growth as we do in §4.5, where a human reviewer writing far longer rationales drives *larger* prompt patches than the auto-reviewer (+63% vs +38% char growth over the same three iterations), consistent with the prompt-bloat narrative we develop on the context-expanding techniques. Third, LLM-as-judge evaluation [23] is typically validated only at the binary-verdict level; we validate our judge's binary verdict against author hand-review (§3.4) and treat its failure-mode labels as provisional and unvalidated, a caution for any downstream use that conditions on those labels.

3 Methods

3.1 Problem framing and solution overview

An LLM annotator extracting a primary diagnosis from a discharge summary makes *systematic* errors: the same prompt repeats the same class of mistake across admissions. Fully supervised fine-tuning addresses this only by gathering more labels and retraining the model, which is expensive, slow, and discards the already-available signal that clinicians produce when they review LLM output. We frame the problem as: given a stream of LLM annotations and structured per-error feedback, how should that feedback be routed back into the system so that the next iteration produces fewer errors *without retraining the model*?

The architecture provides two complementary correction tools. Recall failures (the model missed or mislabelled an entity present in the note) are, in principle, best addressed by adding positive examples of the target pattern (the role of the retrieval-augmented memory), whereas reasoning failures (the model inferred beyond the text, or confused symptoms with diagnoses) call for tightening instruction-level constraints, the role of prompt patching. A six-class failure-mode taxonomy structures the feedback (Table 1): ambiguous gold-label artefacts are held out of correction, and every other (recoverable) failure is aggregated by mode and drives a PromptAgent instruction patch. The retrieval memory accumulates all validated and corrected cases and is exercised by the dedicated RAG condition. The Annotator, PromptAgent, retrieval store, and judge form a closed loop that operates at the prompt level only, so the model weights remain unchanged and the cost per iteration is dominated by API inference rather than training.

Beyond this annotation problem, the same MIMIC-III surface poses a measurement problem that motivates a second contribution: a dual-metric evaluation protocol (§3.4) that separates billing-priority noise from extraction failure. The HITL framework above doubles as the experimental apparatus for exercising this protocol across iterations and prompt-engineering techniques without retraining.

The rest of this section specifies each component in enough detail to reproduce the system: the corpus and sampling (§3.2); the Annotator and its prompt-engineering techniques (§3.3); the LLM-as-judge and the failure-mode routing taxonomy (§3.3.3); the retrieval-augmented memory; the PromptAgent; and the evaluation protocol (§3.4). Verbatim prompts and hyperparameters are listed in Appendix B.

3.2 Data and admission-scope filtering

We use MIMIC-III v1.4 [7], a freely-available database of de-identified ICU admissions from the Beth Israel Deaconess Medical Center. We restrict to discharge summaries, drop error-flagged rows, and retain one note per hospital admission, choosing the most recent chart date when multiple are available. MIMIC-III is purely ICD-9 and remains the dominant ICD-coding-from-notes benchmark.

For each admission we take the primary billed ICD-9 diagnosis, the *principal diagnosis* under the UHDDS (Uniform Hospital Discharge Data Set) definition, that is, the condition chiefly responsible for the admission, taken as the first-listed entry in the admission's billed diagnosis sequence, and use its long-title as the gold string. Each admission has a median of 11 billed ICD-9 codes (max 39). Principal-diagnosis assignment is partly driven by DRG (Diagnosis-Related Group) reimbursement conventions, under which hospitals are paid by category according to which diagnosis is flagged primary, rather than purely clinical primacy. This induces the central measurement artefact the paper addresses with the paired *lenient* metric (§3.4). MIMIC-III's own documentation cautions that this billing-priority ordering of an admission's diagnoses is an imperfect ranking of clinical importance, with "few incentives" to order it correctly [13]; a sepsis admission, for instance, is billed with the infectious agent rather than sepsis as the

principal diagnosis, so an annotator that extracts "sepsis" is clinically correct yet scored strict-wrong. ICD-9 codes are 3–5 character strings whose leading digits encode the chapter (e.g. 41071 falls in Chapter 7, Circulatory).

The annotation task is the extraction of a *clinical diagnosis* from the note body. We scope every reported metric to the 15 clinical ICD-9 chapters, excluding the three administrative groups (V-codes, Perinatal, and Symptoms/ill-defined) whose billed primary is a coding-convention label rather than a clinical diagnosis extractable from the note. The in-scope clinical pool contains 47,567 admissions.

Samples are drawn stratified by ICD-9 chapter (Circulatory alone covers 37% of admissions without stratification, and small chapters must appear in every batch): $n = 100$ for the single-pass baseline and $n = 50$ per HITL iteration. Fixed seed 20260414. A 200-row held-out set is carved off and never sampled, reserved for the higher-powered follow-up study (§6.1); the numbers reported here use only the remaining pool.

3.3 System architecture

Figure 1 illustrates the full pipeline. The system has four principal components (the Annotator, the LLM-as-judge evaluator, the retrieval-augmented memory, and the PromptAgent), connected by the failure-mode routing taxonomy described in §3.3.3.

3.3.1 Annotator Agent

The Annotator is a single-pass LLM call that takes a redacted discharge summary and returns a JSON object containing the extracted diagnosis, a binary flag indicating whether a diagnosis was given, a confidence score, and optional chain-of-thought reasoning. Four prompt-engineering techniques are independently toggleable; what each does and which failure-mode class it targets are listed with the experimental conditions in Table 3 (§3.5).

The four techniques span the recall-style / reasoning-style split that the routing table tests, are well-studied in the LLM-as-annotator literature [19, 20], and operate at the prompt level, consistent with the thesis's no-retraining constraint. Gradient-based techniques (instruction tuning, RLHF) are out of scope.

Before annotation, the note is cleaned (MIMIC de-identification tokens stripped) and the discharge-diagnosis sections are redacted so the gold label is not directly visible to the model.

3.3.2 LLM-as-judge evaluator

At evaluation scale we cannot manually review thousands of annotations, so we employ an LLM-as-judge paradigm [23]: a Gemini 2.5-flash model (temperature 0, enforced response schema) receives the full note, the annotator's output (including any CoT reasoning), and the gold ICD-9 long_title, and returns {`correct: bool, failure_mode: enum, comment: string, confidence: int`}. Rule-based alternatives (exact match, closed-list classification, embedding similarity) were rejected for two reasons: clinical synonymy and abbreviation defeat string-level matching, and none of these alternatives produces the structured failure-mode signal that drives the PromptAgent's routing. Judge reliability is quantified by the Cohen's κ study (§3.4, §5.1).

3.3.3 Failure-mode taxonomy and routing

When the judge labels an annotation incorrect, the verdict carries a failure-mode label (Table 1). The five *recoverable* modes are pooled by mode at end-of-iteration, together with the reviewer's free-text rationales, and passed to the PromptAgent, which builds a meta-prompt describing each pattern and emits a versioned instruction patch for the next iteration. Ambiguous cases are held out of both the PromptAgent and the retrieval store and flagged for human data-quality review, so contestable gold never drives a prompt change. The taxonomy thus separates recoverable errors, which drive instruction patches, from gold-label artefacts, which are excluded; the per-failure-mode decomposition in §4 tests how the loop reduces each recoverable mode. The retrieval-augmented memory (detailed below) is a separate, always-on store of validated and corrected cases from other admissions; it is consulted only by the RAG condition and is not selected per failure mode.

3.3.4 Retrieval-augmented memory

The retrieval-augmented memory is the architectural component underlying the RAG condition (Table 3, §3.5); it is active in that condition only and is inert for Baseline, CoT, Few-shot, and Self-consistency. After each annotation the judge's verdict is recorded. Reviewed cases (the medical note, the model's diagnosis, the judge's verdict, and metadata) are indexed in a cloud-hosted vector store and become available as grounding context for future inferences: at generation time the Annotator is given a File Search tool over the store, and the model retrieves and grounds on semantically similar prior cases server-side. Both validated correct cases (the model's diagnosis is used as the retrieval target) and corrected cases (the gold long-title is used instead) are stored; ambiguous-case failures are excluded so contestable gold does not contaminate retrieval. Implementation details (embedding model,

Table 1: Failure-mode taxonomy, definitions, and how each mode is handled in the loop. The five recoverable modes are aggregated by mode (with the reviewer's rationales) and drive PromptAgent instruction patches; ambiguous cases are held out of correction because the gold label itself is unreliable. The retrieval memory (detailed below) is a separate, always-on store of validated and corrected cases exercised by the RAG condition, not a per-mode destination.

Failure mode	Definition	Handling in the HITL loop
Missed entity	A confirmed diagnosis present in the note was not returned.	Aggregated by mode with the reviewer's free-text rationales and passed to the PromptAgent, which proposes instruction-level patches applied at the next iteration.
Terminology gap	The returned diagnosis is the right entity but expressed in non-standard or non-billable terms.	
Hallucination	The returned diagnosis is not supported by the note.	
Unsupported inference	The model returned a more specific subtype or cause than the note explicitly supports.	
Symptom–diagnosis confusion	The model returned a symptom, sign, procedure, or complication instead of the underlying confirmed diagnosis.	
Ambiguous case	The gold label itself is wrong, vague, or a coding-convention artefact rather than a clinical error the annotator could have avoided.	Excluded from prompt patching and from the retrieval store; flagged for human data-quality review.

document format, deduplication key, the specific FileSearch backend) appear in Appendix B.

3.3.5 LLM-proposed prompt patching

At the end of each iteration the PromptAgent aggregates the non-ambiguous failures by mode and submits a structured meta-prompt to a separate LLM call. The meta-prompt holds the system prompt, the user query, the allowed-diagnosis list, and the output JSON schema as read-only and restricts the agent to either adding new instructions or making small refinements to existing ones, with an accompanying rationale; every patch is therefore an instruction-level edit without scope expansion or schema changes (verbatim meta-prompt and output schema in Appendix B.6). The proposed patch is timestamp-versioned, recorded to the prompt log, and approved by the human reviewer through a binary approve-or-reject step before being deployed to the next iteration. Alternatives that touch model weights (fine-tuning, RLHF) are out of scope under the no-retraining constraint.

3.4 Evaluation protocol

The research question concerns *medical text annotation*, which corresponds to the clinical-chapter scope defined in §3.2: admissions whose primary billed code describes a disease or condition, not an encounter type or billing convention. *The clinical-scope metric is therefore the metric of record for the RQ throughout this paper*; every run is generated from a clinical-only sample by construction (the sampler excludes the three administrative chapters by default), so all reported numbers are clinical-scope.

We report two accuracy metrics simultaneously. Under *strict accuracy*, the annotator's free-text diagnosis must be semantically consistent with the primary billed ICD-9 code's long-title for the admission. Under *lenient accuracy*, the annotator's diagnosis is accepted if it is semantically consistent with any of the admission's billed ICD-9 codes (primary or secondary). Both metrics use identical semantic-match rules: synonyms, abbreviations, and strict sub-types are accepted; a different organ system, symptom-as-diagnosis, or unsupported inference is rejected. The lenient metric is implemented as a post-hoc re-judge pass that provides the judge with the full ICD list for the admission; the primary judge remains unchanged.

Accuracy in both regimes is the fraction of cases in a run for which the annotator's diagnosis is accepted under the corresponding match rule. The verdict source varies by phase: the author hand-reviews every case in the $n = 100$ baseline (§4) and in the strict pass of the human-reviewer trajectory (§4.4), while an LLM judge renders the verdicts in the auto-reviewed per-technique trajectories (§4.2); lenient verdicts in the human-reviewer trajectory come from the automatic re-judge pass described above.

We quantify the agreement between the judge's binary verdict and the author's hand-review of the $n = 100$ baseline (§4): Cohen's $\kappa = 0.82$ (strict; 92% raw

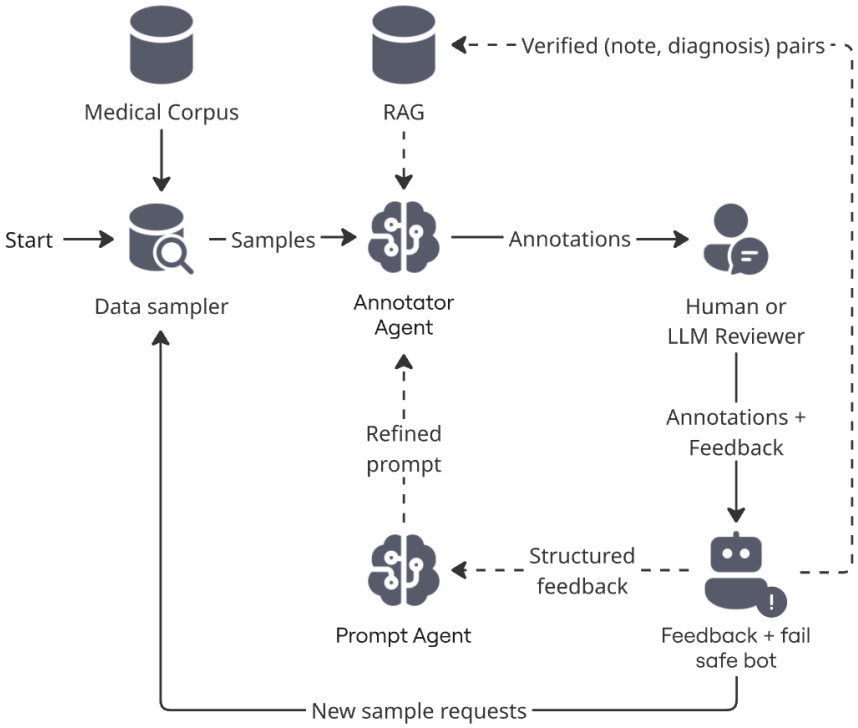

Figure 2: Architectural extension of the core HITL loop (Figure 1), active only in the RAG condition. Reviewed cases are written to a persistent vector store (validated correct cases use the model's diagnosis as the retrieval target; corrected cases use the gold long_title); ambiguous-case failures are excluded so contestable gold does not contaminate retrieval. At generation time the Annotator queries the store via a managed File Search tool and is grounded on semantically similar prior cases server-side. The Baseline, CoT, Few-shot, and Self-consistency conditions do not query the store.

agreement) and $\kappa = 0.77$ (lenient; 95%), both *substantial*, so the verdict underlying the strict-accuracy comparisons is reliable. The agreement values pair the LLM judge's verdicts chart-for-chart with the author's hand verdicts; Cohen's $\kappa = (p_o - p_e)/(1 - p_e)$ is applied with $p_e = p_j p_h + (1 - p_j)(1 - p_h)$, where p_j and p_h are the judge's and the hand-reviewer's marginal positive rates. Table 2 reports the 2×2 cell counts; substituting, $\kappa_{\text{strict}} = (0.92 - 0.56)/(1 - 0.56) = 0.82$ and $\kappa_{\text{lenient}} = (0.95 - 0.78)/(1 - 0.78) = 0.77$. Both values sit at the substantial-to-almost-perfect end of the standard Cohen's κ scale (Landis–Koch convention), so the auto-judged per-technique trajectories in §4.2 can be read without major loss relative to a hand-reviewed counterpart. The lenient row's high p_o but lower κ reflects the prevalence effect on p_e (the first kappa paradox cited in §4).

Validation against an independent practising clinician, and of the judge's failure-mode label rather than the binary verdict, is the principal residual threat to validity and is deferred to §6.1.

Accuracy is the headline metric throughout, reported under both regimes. The HITL trajectories in §4 additionally report per-iteration failure-mode composition (Table 6) so that error-type redistribution is visible alongside aggregate accuracy. The Annotator also returns a commit-or-defer signal and a self-reported confidence score; per-iteration defer rate and confidence calibration are reported in §4.

Table 2: Judge-vs-hand 2×2 contingency on the $n = 100$ baseline. a: both correct; b: both incorrect; c: judge correct, hand incorrect; d: judge incorrect, hand correct.

Metric	a	b	c	d	p_o	p_e
strict	64	28	5	3	0.92	0.56
lenient	85	10	5	0	0.95	0.78

Confidence intervals at the relevant n are reported alongside every headline number so the reader can judge effect vs. sampling noise. The $n = 100$ baseline supports point estimates with a $\sim\pm$9pp 95% CI; per-iteration trajectories at $n = 50$ are underpowered

to resolve small between-technique differences and are reported as qualitative trends (see §5.1). Sample is drawn fresh per iteration, so the trajectories also reflect batch difficulty in addition to prompt change.

All experiments use fixed random seeds, temperature $T = 0.0$ for single-pass inference, response-schema enforcement on the judge, and temperature $T = 0.7$ only for self-consistency sampling (where stochasticity is the mechanism itself). These choices collectively maximise determinism so that observed accuracy differences across iterations and conditions can be attributed to prompt and technique changes rather than decoding noise. Per-run dataframes and aggregate metrics are persisted to disk for post-hoc analysis.

3.5 Experimental design

The empirical evaluation has three phases: a human-reviewed single-pass baseline characterisation, a per-technique HITL longitudinal study at $n = 50 \times 3$ iterations using an LLM-as-judge auto-reviewer, and a human-reviewer validation trajectory at $n = 50 \times 3$ iterations that replicates the Baseline condition with the researcher personally reviewing through the same graphical interface.

3.6 Models, decoding, and prompts

The annotator is Gemini 2.5-flash throughout. The choice of a small/medium-capacity hosted model is deliberate: production deployment of a HITL annotation system inside a hospital will, in practice, run a locally-hosted model (e.g., a 7B–30B open-weights LLM) rather than a frontier API model, for data-residency and cost reasons. Gemini 2.5-flash is a reasonable proxy for that deployment-realistic capacity regime, and findings that depend on running a frontier model would not transfer. We use temperature $T = 0.0$ for all single-shot annotations to make individual runs reproducible; the Self-consistency condition is the sole exception, decoding at $T = 0.7$ (Table 3).

The judge is also Gemini 2.5-flash (held constant across conditions to isolate annotator-side effects) with response-schema enforcement binding the output to four fields: a binary correctness verdict, a failure-mode label, a free-text comment, and a confidence score. The judge runs at $T = 0.0$ to suppress verdict drift across re-runs. Its binary verdict was validated against the author's hand-review of the $n = 100$ baseline (see §3.4).

All HITL conditions start from the same initial prompt, a nine-instruction zero-shot specification asking for the primary diagnosis as a free-text long-title plus a confidence integer. Per-iteration prompts produced by the PromptAgent are version-stamped and persisted; verbatim text appears in Appendix B.

3.7 Phase 1, baseline characterisation ($n = 100$, human-reviewed)

A single non-HITL run with the initial prompt establishes the two reference numbers used throughout, strict and lenient accuracy. The author reviewed all 100 annotations directly against the chart through the same graphical interface used in Phase 3, so the strict verdict is hand-verified rather than judge-assigned. The sample is drawn from the clinical-chapter pool defined in §3 (47,567 admissions across 15 clinical ICD-9 chapters, excluding V-codes, Perinatal, and Symptoms/ill-defined) and stratified by ICD-9 chapter. A fixed random seed is used. The held-out set defined in §3 is identical across conditions but reserved for the higher-powered follow-up; it is not used for the numbers reported here.

3.8 Phase 2, per-technique HITL longitudinal study ($n = 50 \times 3$)

Five HITL conditions are run for three iterations each, summarised in Table 3. All conditions share the same loop structure (annotate, auto-review, parse failures, route signals, patch the prompt, advance to the next iteration) and differ only in the prompt-engineering technique that the annotator wraps around the initial prompt. The correction logic is held constant across conditions: the PromptAgent patches the prompt from all non-ambiguous failure signals, ambiguous cases are held out, and the RAG condition additionally maintains the always-on retrieval store. The failure-mode handling is given in Table 1 (§3).

Each iteration draws a fresh stratified $n = 50$ sample (using the same chapter strata as Phase 1 with the stateful sampler advancing the random state between calls). Drawing different samples each iteration tests whether the patched prompt *generalises*, rather than measuring training-set fit on cases the patch was designed to fix. The RAG store is reset per trajectory so retrieval results reflect only that run's accumulated cases, not state carried over from prior runs.

3.9 Phase 3, human-reviewer validation ($n = 50 \times 3$)

Phase 2's auto-reviewer is a tractable proxy for clinical review, but the central claim of this work is that a *human* in the loop can steer the annotator's prompt through qualitative feedback. To validate that the auto-reviewer trajectory generalises to a real human reviewing the same loop, we replicate the Baseline condition with the researcher personally reviewing through a Tk-based graphical interface (implementation details in Appendix B).

Table 3: The five HITL conditions compared in the per-technique longitudinal study: the technique each wraps around the initial prompt and the failure-mode class it is designed to address. All conditions share the same outer loop and differ only in the wrapped technique; the technique-to-failure-mode mapping is what makes the per-failure-mode decomposition in §4 interpretable.

Condition	**What it does**	**Failure modes targeted**
Baseline	No extra technique; prompt patching and routing only. The reference trajectory the other four are compared against.	none (reference)
CoT	Chain-of-thought scaffold: the model enumerates candidate diagnoses with note-grounded evidence before committing to a primary.	unsupported inference, symptom–diagnosis confusion (reasoning)
Few-shot	Five hand-curated note-to-diagnosis exemplars prepended to the prompt.	missed entity, terminology gap (recall)
Self-consistency	Three independent samples at $T = 0.7$; the majority-vote diagnosis is the final answer [19].	variance / one-off slips (no single mode)
RAG	Top-k retrieval from a persistent store of past validated and corrected note-to-diagnosis pairs, supplied as grounding for the next inference.	missed entity, terminology gap (recall)

The reviewer in Phase 3 is the author, a Master's student in Computer Science, not a clinician. The Phase 3 trajectory therefore primarily exercises (i) the framework's ability to ingest dense free-text feedback (verified against the auto-reviewer baseline, see §4.4) and (ii) on-corpus detection of gold-label artefacts (billing/coding inconsistencies which a careful non-clinician can identify from the ICD-9 long_title and the chart itself), rather than fine-grained clinical adjudication. The auto-reviewer's binary verdict is validated separately against the author's hand-review of the $n = 100$ baseline (§3.4); validation against an independent practising clinician is future work (§6.1). Phase 3 thus validates that human-in-the-loop *operation* is feasible on a real reviewer, while the hand-review validates that the judge's verdict tracks a careful human's. The protocol is identical to Phase 2 (same condition, same starting prompt, the *same* $n = 50 \times 3$ charts as the auto-reviewed Baseline trajectory, same PromptAgent patch-approval gate between iterations), with two deliberate procedural differences. First, for every chart marked incorrect, the reviewer writes a 2–4 sentence free-text rationale in the comment box that (i) points at specific evidence in the note (lab values, temporal cues, phrasing), and (ii) suggests what the prompt should learn from this case. This exercises the contextual-explanation and domain-specific-correction feedback channels named in SQ1, which the auto-reviewer's terse stylised comments do not. Second, the PromptAgent reads these rationales as the example field of the aggregated failure signal, so the patches that result can be qualitatively contrasted with the patches produced under auto-review on the same condition. Phase 3 produces a single trajectory; we use it to (a) report a real human-in-the-loop accuracy trajectory for direct comparison against the auto-reviewer baseline, and (b) qualitatively analyse whether rich textual feedback shifts the PromptAgent's output. Results are reported in §4.4.

3.10 Reproducibility

All experiments use fixed random seeds, version-stamped prompt files, and persisted per-case dataframes. Each run records, for every case, the note, the model annotation, the judge verdict, the parsed failure signal, and the prompt version used. A separate lenient-rescoring pass produces a paired record per case with the judge re-judging against the full ICD-9 list. Multi-run summaries aggregate these per-case files. The figures and tables in §4 are produced by deterministic scripts that consume only the persisted logs (no re-running of API calls). The complete codebase, log layout, prompt files, and figure-generation scripts are released publicly; see Appendix B for paths and the project's source-code repository.

4 Results

4.1 Baseline performance and the strict/lenient gap

Our reference baseline is a human-reviewed single-pass run at $n = 100$ (Gemini 2.5-flash, v1 prompt, no HITL), in which the author validated every annotation directly against the chart rather than relying on the LLM judge. Scored on the clinical ICD-9 chapters

(§3.2), it reaches 0.67 strict accuracy (95% CI $\approx \pm 9$pp) and 0.85 lenient accuracy, a +18pp strict/lenient gap. We take this hand-verified run as the reference baseline precisely because every label was checked by hand; the gap it exposes is the measurement-ceiling signal that the rest of the paper investigates. These hand verdicts also agree with the LLM judge's automatic verdicts on the same 100 charts at $\kappa = 0.82$ (strict) and $\kappa = 0.77$ (lenient; 92% and 95% raw agreement),[2] an independent re-validation of the auto-reviewer that underwrites the auto-judged trajectories in §4.2.

The gap is driven by cases where the annotator extracted a real billed comorbidity rather than the single code designated primary under billing convention, a clinically valid extraction scored strict-wrong. This is not a failure of clinical reading but of the gold label's billing-primacy assumption, which the lenient metric isolates. Figure 3 shows the gap is present across most clinical chapters rather than confined to a few, consistent with a corpus-wide billing-primacy artefact. The remainder of the paper operates on the clinical-chapter scope throughout.

4.2 HITL per prompt-engineering technique (SQ2)

To answer SQ2 directly, we ran the five HITL conditions defined in Table 3 (§3.5) for three iterations each at $n = 50$ samples per iteration. Each iteration draws a fresh stratified sample so the trajectory measures generalisation of the patched prompt, not training-set fit. Apart from Self-consistency (which decodes at $T = 0.7$ by design), all conditions run at temperature $T = 0.0$ with a fixed seed, so the iteration-to-iteration movements in the trajectories below reflect prompt and technique changes plus per-iteration sampling variation, not decoding randomness. Table 4 reports strict and lenient accuracy at every iteration.

The headline numbers in Table 4 and Fig. 4 come from a single pilot trajectory at $n = 50$ per condition per iteration, with an iteration-to-iteration noise floor of approximately ±17pp (§5.1). All per-technique observations in the remainder of this section are therefore read as *qualitative trend directions*, not statistical inferences. Three observations stand out.

First, lenient accuracy converges to roughly 0.86–0.90 regardless of technique: iteration-3 lenient values across all five conditions sit in a tight 0.86–0.90 band. Conditions that start *high* on iter 1 (Few-shot, RAG at 0.92–0.94) drift downward toward this band; conditions that start *low* (Baseline at 0.80) drift upward. This is consistent with the band being a measurement ceiling imposed by the gold-label regime (see §5) rather than a technique-specific outcome. In the terms of the research question, this convergence is the sense in which the HITL loop improves *reliability*: repeated iteration drives outputs to a consistent lenient-accuracy band independent of the starting technique.

Second, the per-technique strict differences are qualitative. The published literature on these techniques (chain-of-thought [20], few-shot prompting [1], self-consistency [19], and retrieval-augmented generation [10]) reports modest gains over zero-shot baselines on a range of tasks, so the prior expectation is that each technique sits at or slightly above Baseline. The observed direction in Table 4 is the opposite: Baseline is the only condition with a positive strict trend across iterations; RAG and Few-shot decline monotonically across the three iterations ($0.82 \rightarrow 0.68$ and $0.76 \rightarrow 0.68$ respectively), plausibly attributable to prompt bloat (see §5). Self-consistency, which adds no fixed wrapper to the prompt and only re-decodes the same prompt three times at $T = 0.7$, stays essentially flat across the trajectory ($0.74 \rightarrow 0.70 \rightarrow 0.72$); its lack of decline is consistent with the prompt-bloat reading, since there is no extra context to compete with the patches for attention.

Third, the strict/lenient gap widens slightly with iteration: the average gap (lenient minus strict) grows from +14pp at iter 1 to +16pp at iter 3 across the five conditions. The HITL loop systematically shifts probability mass toward *clinically valid* diagnoses faster than toward the *billing-primary* one; stakeholder-specific deployment implications are discussed in §5.

Two distinct sources inflate prompt length, and we separate them explicitly rather than reporting a single aggregate. The first is the fixed *technique wrapper* applied at iteration 1 before any feedback, measured directly from the prompt builder (Table 5). Relative to the shared 1078-character, nine-instruction baseline block that every condition starts from, Few-shot adds 1552 characters of static worked examples (+144%), CoT adds a net 620 characters of reasoning scaffold, Self-consistency adds *nothing* to the prompt (it re-decodes the identical prompt three times at $T = 0.7$ and takes a majority vote), and RAG adds nothing to the *authored* prompt but attaches a server-side retrieval tool whose injected text we cannot measure (§5). The second source is HITL patching, and it runs in *all five* conditions: every condition enables the PromptAgent, so Self-consistency and RAG accumulate instruction patches across iterations exactly as Baseline does, even though they add no *fixed* wrapper text. For Baseline this patching grows the instruction block from 1078 to 1490 characters over the three iterations (in-

[2] The lenient pair has *higher* raw agreement (95%) yet *lower* κ than the strict pair because lenient verdicts are heavily skewed toward "correct" (∼88%), inflating chance agreement to ≈0.78 and leaving little room above it: the prevalence effect, or Feinstein and Cicchetti's first kappa paradox [5]. We therefore report raw agreement alongside κ throughout.

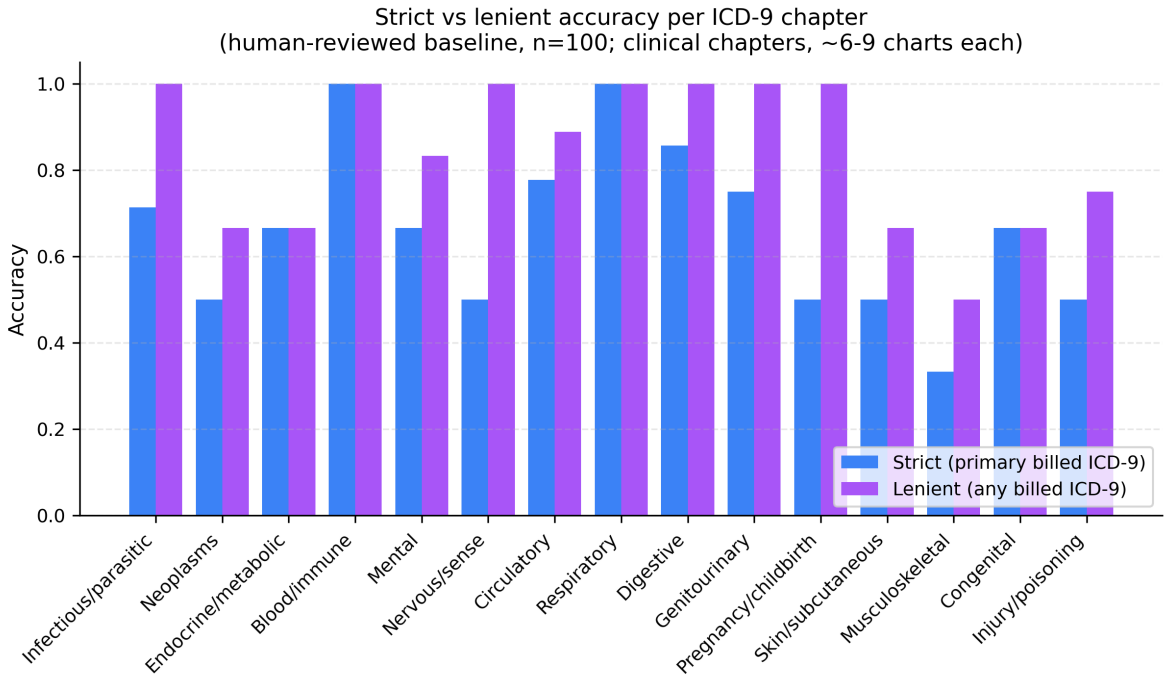

Figure 3: Strict (primary billed ICD-9) vs. lenient (any billed ICD-9) accuracy per clinical ICD-9 chapter on the human-reviewed single-pass baseline ($n = 100$). The strict/lenient gap appears in most chapters rather than being confined to a few, consistent with a corpus-wide billing-primacy artefact rather than a handful of problematic chapters. Per-chapter counts are small (∼6–9 charts each), so individual bars are illustrative rather than precise estimates; the chapter-wide pattern*, not any single chapter's value, is the point.*

struction count holding at 9 → 10; full per-iteration trajectory in Table 7, §4.5). The fixed wrapper in Table 5 is constant across iterations; the patch growth is not.

4.3 Feedback-type effectiveness (SQ1)

SQ1 asks which categories of human feedback are most effective for prompt refinement. Each incorrect case is assigned one of six failure modes (hallucination, missed entity, terminology gap, unsupported inference, symptom–diagnosis confusion, ambiguous case). Ambiguous cases are held out of correction (the gold label is unreliable); the five recoverable modes are aggregated by mode and drive the PromptAgent's instruction patches (§3, Table 1). Table 6 reports the failure-mode breakdown per iteration across the five HITL conditions.

Three points follow from Table 6.

First, missed entity is the modal failure mode in 14 of 15 iterations: across the five conditions and three iterations it is the most-frequent failure mode in every iteration except Few-shot iter 1, where it tied with unsupported inference at 5 cases each (42% of that iteration's 12 failures). In the remaining fourteen iterations missed entity accounts for 56–91% of the failure budget (mean 67% across all 15). The Few-shot iter 1 exception is consistent with the prompt-bloat narrative (§5): the static few-shot demonstrations include a deliberate hard-negative (*indicators-without-name → return none*) whose effect early in HITL is to push the model toward over-rejection, redistributing mass from missed entity to unsupported inference. The remaining failure modes contribute small absolute counts that fluctuate within the noise band. SQ1 therefore collapses, in practice, to a single question: how effective is the HITL loop at reducing missed-entity errors?

Second, only Baseline shows a monotonic missed-entity reduction (11 → 10 → 7). The technique-enriched conditions either hold roughly flat (Self-consistency: 10 → 10 → 8) or trend upward (Few-shot: 5 → 11 → 12; RAG: 5 → 11 → 9). This is consistent with prompt bloat: instruction patches that target missed-entity cases compose poorly with in-context examples (few-shot) or retrieved passages (RAG), which themselves expand the prompt and dilute the patches' signal.

Third, instruction patching, not retrieval, is the corrective lever. Missed-entity errors are addressed in every condition by the PromptAgent, the only feedback-driven prompt mechanism in the loop; baseline, which adds no retrieval at all, is the single condition that

Table 4: HITL trajectory per prompt-engineering technique, $n = 50$ samples per iteration, three iterations. Strict matches the primary billed ICD-9 code; lenient matches any billed ICD-9 for the admission. Different samples are drawn per iteration (test of patch generalisation, not training-set fit). Bold marks the per-condition best iteration on each metric.

	Strict accuracy			Lenient accuracy		
Condition	iter 1	iter 2	iter 3	iter 1	iter 2	iter 3
Baseline	0.68	**0.78**	0.76	0.80	0.84	**0.90**
CoT	0.72	0.70	**0.74**	0.86	0.84	**0.90**
Few-shot	**0.76**	0.72	0.68	**0.92**	0.88	0.86
Self-consistency	**0.74**	0.70	0.72	**0.90**	**0.90**	0.86
RAG	**0.82**	0.72	0.68	**0.94**	0.92	0.88

Table 5: Fixed prompt overhead of each technique, measured on the iteration-1 prompt before any HITL patching. "Added to authored prompt" is what the prompt builder appends beyond the shared nine-instruction, 1078*-character baseline block (identical across all five conditions at iteration 1). Self-consistency adds zero characters because it re-decodes the same prompt and votes; RAG adds zero* authored *characters but injects retrieved cases server-side, so its true context growth is not recoverable from our logs (§5).*

Condition	Added to authored prompt (iteration 1)	Δ chars	Auditable
Baseline	instruction block only	0	yes
CoT	reasoning scaffold + JSON format rule	+620	yes
Few-shot	5 static worked examples	+1552	yes
Self-consistency	nothing (3× re-decode, majority vote)	0	yes
RAG	retrieved cases, injected server-side	n/a	no

monotonically reduces them ($11 \to 10 \to 7$). Retrieval is active only in the RAG condition, where it is stacked on top of the same prompt patches, and there it coincides with the steepest strict decline (Table 4), not a gain.

Read against the three feedback types SQ1 names (error categorization, contextual explanations, and domain-specific corrections), this gives a graded answer rather than a single lever. Error categorization, the failure-mode tag, is the *enabling* type: it is what routes a correction to the PromptAgent at all, and on this corpus it concentrates the loop on the modal missed-entity mode, but a tag plus the auto-reviewer's terse comment reduces that mode monotonically only in the simplest (Baseline) condition. The reviewer's *contextual explanations* and *domain-specific corrections* are the higher-value type: the longer free-text rationales (mean ≈305 vs 99 characters, §4.4) surfaced an instruction-level refinement absent from the auto-reviewer's stylised comments. The reviewer's rationales produced a patch that sharpened the principal-diagnosis rule rather than just expanding existing instructions, and the patched rule generalised across charts by iterations 2–3 (the qualitative trace is in §4.4); rich rationales thus change the *kind* of patch produced and not merely its length. The retrieval memory, by contrast, adds no measurable gain and, stacked on the prompt patches, coincides with the largest strict declines (Table 4). This ordering is qualitative: the auto-reviewer (Phase 2) and human-reviewer (Phase 3) trajectories were judged by different reviewers and run once each at $n = 50$, so we can compare the kind of patch each feedback type yields but cannot attribute a strict-accuracy difference to the feedback type itself (§4.4).

Beyond commit-accuracy, the annotator also exposes a commit-or-defer signal: the model returns the string "none" when it judges no primary diagnosis to be extractable from the note. Aggregating across the five conditions × three iterations ($n = 750$ cases), the model deferred on 63 cases (8.4% of the corpus); the judge agreed the defer was justified on 25 of those (40%), with the remaining 38 flipping to missed entity under the judge's review. Defer-precision of 0.40 is a behavioural signal independent of strict accuracy: when the model abstains, the abstention is wrong more often than right. This complements the failure-mode analysis above: the *missed-entity-via-defer* subset (the 38 cases where the model returned "none" but the judge found a confirmed diagnosis in the note) is the largest single mechanism feeding the missed-entity budget, and a recall-oriented deployment would benefit more from improving the model's defer calibration than from squeezing additional points out of the commit-accuracy axis.

The annotator also emits a self-reported confidence score, and it is systematically overconfident: 88% of an-

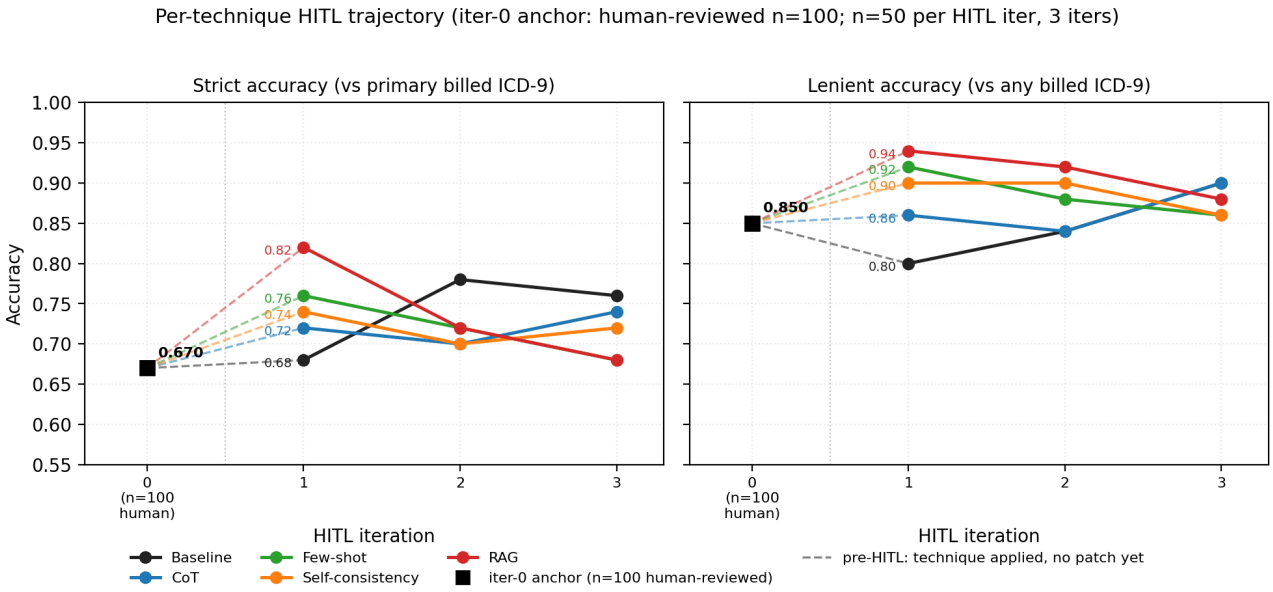

Figure 4: Per-technique HITL trajectory across iterations 1–3 ($n = 50$ per HITL iteration). Left: strict accuracy; right: lenient accuracy. The black square at iteration 0 is the shared single-pass baseline ($n = 100$, human-reviewed by the author: strict 0.67; lenient 0.85, both hand-verified against the billed ICD-9 codes): every condition begins from the same initial prompt with no technique wrapper and no HITL patching. The dashed segments between iteration 0 and iteration 1 represent the effect of layering each technique on top of the initial prompt, before *any HITL feedback has been applied; solid segments from iteration 1 onward are HITL iterations proper. Iter-1 differences across conditions reflect the technique-prior plus $n = 50$ sampling variation on independently drawn stratified samples, not HITL feedback. Lenient convergence around $[0.86, 0.90]$ is robust across techniques; strict differences across techniques are within the per-iteration noise floor.*

notations carry a confidence of 90 or above, yet those high-confidence outputs are correct only 74% of the time, and the gap does not shrink across iterations. The score is correspondingly weak for triage: routing the lowest-confidence decile to the reviewer first surfaces only ∼13% of all errors, barely above the 10% a random review of the same size would catch. Because the loop patches instructions rather than the model's self-assessment, neither the overconfidence nor its triage value improves across iterations; reliability must therefore come from the human review gate, not from trusting (or thresholding on) the model's own confidence.

4.4 Human-reviewer validation trajectory

To exercise the *contextual explanations* and *domain-specific corrections* feedback channels named in SQ1 on a real reviewer, we ran a Baseline trajectory at $n = 50 \times 3$ iterations with the researcher personally reviewing through a Tk-based graphical interface (Phase 3 protocol, §3.9), on the *same* charts as the auto-reviewed Baseline trajectory (§4.2). The researcher wrote a free-text rationale for every incorrect case, averaging ≈305 characters per comment against 99 for the auto-reviewer on the matched charts.

The headline result from this trajectory is ≈10% on-corpus gold-label ambiguity. The reviewer marked 7/7/1 charts as ambiguous case across the three iterations (mean 10%; 14% in iterations 1–2, falling to 2% in iteration 3). The ambiguous case label is reserved for charts where the MIMIC primary ICD-9 code is itself wrong, vague (e.g. "Other specified disorder"), or a coding-convention artefact rather than a clinical error the annotator could have avoided. This is direct, on-corpus evidence for the strict-accuracy measurement ceiling that the discussion (§5) advances on the basis of published literature ([3, 16, 17]) alone. At ≈10% of stratified discharge summaries, gold-label ambiguity is a non-trivial fraction of the residual strict-error budget, and is a direct reason the per-technique strict accuracies in §4.2 do not improve monotonically.

Figure 5 traces the trajectory. Strict accuracy, scored on the researcher's own verdicts, rises across the three iterations ($0.62 \rightarrow 0.64 \rightarrow 0.78$), while lenient accuracy, the model's diagnosis automatically re-scored against any billed ICD-9 for the admission (the lenient criterion of Table 4), holds a tight band

Table 6: Failure-mode counts per HITL iteration ($n = 50$ each). Only the three most frequent modes are shown; the long tail (terminology gap, ambiguous case, hallucination) contributed 0–2 cases per iter and is omitted for clarity. ME = missed entity, UI = unsupported inference, SDC = symptom–diagnosis confusion.

	iter 1			iter 2			iter 3		
Condition	ME	UI	SDC	ME	UI	SDC	ME	UI	SDC
Baseline	11	3	0	10	1	0	**7**	2	2
CoT	8	5	0	11	2	0	9	4	0
Few-shot	5	5	1	11	3	0	12	1	1
Self-consistency	10	2	0	10	2	1	8	3	2
RAG	5	3	1	11	1	1	9	2	3

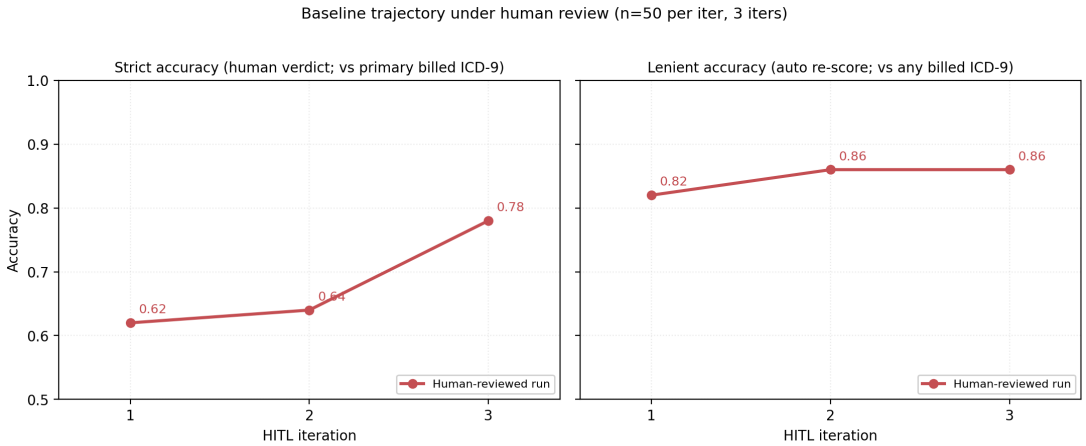

Figure 5: Baseline HITL trajectory under human review ($n = 50$ per iteration, three iterations, on the same charts as the auto-reviewed Baseline). Left: strict accuracy from the researcher's own verdicts; right: lenient accuracy from the automatic re-score against any billed ICD-9. Strict rises 0.62 → 0.78 while lenient holds ≈0.86, narrowing the gap from +20 to +8pp. The iteration-3 strict gain coincides with a different chart batch, so the trend is read as directional, not confirmed.

(0.82/0.86/0.86); the strict/lenient gap therefore narrows from +20 to +8pp as strict accuracy climbs toward the lenient band. We read the upward strict trend as a *potential* improvement signal rather than a confirmed one: the iteration-3 gain falls on a different stratified chart batch, on which the auto-reviewed Baseline also scored 0.76. Prompt improvement and batch difficulty therefore cannot be separated here; the higher-powered replication with a fixed held-out set proposed in §6.1 is needed to confirm the direction.

Inspecting the prompt patches produced under human review, the free-text rationales sharpened an instruction-level rule that the auto-reviewer's terse comments did not. The researcher repeatedly flagged cases where the annotator returned a *focal* or proximate finding instead of the *principal* condition that drove the admission: "Cholangitis" where the gold codes *Unspecified septicemia* (*"the AI captured the focal infection rather than the systemic septic state the gold codes as primary"*), "Clostridioides difficile infection" where the gold codes *Acute kidney failure*, and a steroid-withdrawal pneumonitis where the gold codes *HIV disease*. Aggregated, these rationales sharpened the principal-diagnosis instruction so that the model is now told to return the principal diagnosis

> *"...rather than a secondary diagnosis, a chronic comorbidity, or a contributing cause of the principal diagnosis,"*

the final clause (*contributing cause*) being exactly the focal-versus-systemic distinction the cholangitis rationale named. By iterations 2–3, on separate stratified samples, the model more often returned the systemic principal state rather than a focal contributor: e.g. "Sepsis due to RLL pneumonia and UTI" and "Septic shock" on admissions whose gold primary was *septicemia*, where a focal "pneumonia" or "UTI" answer would have scored strict-wrong. These are qualitative evidence (with the batch caveat above) that the SQ1 *contextual explanation* channel produces patches that generalise across charts. The full chart-through-the-loop trace for the cholangitis case (annotator out-

put, reviewer rationale, PromptAgent patch, and the iteration-2 sepsis chart it generalised to) is reproduced verbatim in Appendix B.9.

4.5 Prompt complexity (SQ3) and convergence (SQ4)

SQ3 and SQ4 are partially answered on the discharge-summary corpus, with the scope of the claim bounded by the single-condition design of the analysis.

For prompt complexity (SQ3), we trace Baseline along two three-iteration trajectories on the same charts, differing only in reviewer: Phase 2 (auto-reviewed, §4.2) and Phase 3 (human-reviewed, §4.4), with per-iteration prompts recoverable from the logs. Table 7 reports two complexity metrics (instruction count, total instruction-text character count) alongside strict accuracy for both trajectories.

Table 7: Prompt complexity vs strict accuracy across the Baseline trajectories under two reviewer regimes (Phase 2 auto-reviewed and Phase 3 human-reviewed), $n = 50$ per iteration. Char count is the total instruction-text character count. Char growth is iter 3 vs iter 1. The two strict-accuracy figures are judged by different reviewers (auto vs human) and are not directly comparable; only the within-trajectory growth-vs-accuracy reading is intended.

	Phase 2 (auto)		Phase 3 (human)	
Iter	#ins.	Chars	#ins.	Chars
1	9	1078	9	1078
2	10	1268	9	1316
3	10	1490	10	1762
Char growth	+38%		+63%	
Strict acc	0.68 → 0.76		0.62 → 0.78	

Three observations follow. First, *instruction count is a misleading complexity proxy*: at iter 3 of Phase 2 the count stayed at 10 but the total instruction-text character count grew by +18% (iteration 2 to iteration 3: 1268 → 1490 chars), because the PromptAgent refined existing instructions rather than adding new ones. Structural composition (refine vs. add) is a separate axis from the headline count. A concrete trace makes this explicit: after iteration 1, whose errors were dominated by missed entity, the PromptAgent added a single recall-oriented rule generated from the aggregated failure signal, *"thoroughly scan the entire text to identify all explicitly confirmed diagnoses before determining the single primary diagnosis"*; subsequent iterations mostly re-expanded existing rules with examples and qualifiers rather than introducing new constraints (the symptom-distinction rule, for instance, grew from a single illustrative example at iteration 1, "chest pain is a symptom", to four by the final iteration, "chest pain, fever, hypotension, or hypercalcemia"). We describe these patches as generated *in response to* the prior iteration's failures, the direction the routing supports; we do not claim a specific added line *caused* a downstream accuracy change. Verbatim prompts for every iteration are released in Appendix B. Second, *the human reviewer's richer rationales drive larger prompt patches*: writing far longer rationales (mean 305 vs 99 chars/comment, §4.4) translated into a +63% char-growth trajectory across three iterations, *more* than Phase 2's +38% on the same charts. Strict accuracy nonetheless rose across the trajectory (Table 7), so patch verbosity is not what drives the gains; the direction is consistent with the prompt-bloat reading developed for the context-expanding techniques in §5, where extra context-length costs the small-model annotator's attention budget rather than buying capability. Third, the prompt-complexity-vs-strict-accuracy relationship is not monotonic; Phase 2 strict peaks at iter 2 and plateaus while its prompt keeps growing, and Phase 3 reaches its best strict value at iteration 3 even as the prompt continues to expand. This is consistent with the §5 reading that instruction-level patching shifts probability mass toward clinically-valid diagnoses faster than toward the strict billing-primary, and with the broader observation that strict-axis gains are bounded by the measurement ceiling. A formal prompt-complexity-vs-accuracy regression is not statistically supportable here; the trajectories are reported descriptively.

SQ4 asks two things: (a) how many iterations are required for convergence on a given corpus, and (b) whether the convergence behaviour generalises across clinical text types. We answer (a) on discharge summaries and explicitly defer (b) to §6.1. On the discharge-summary corpus, lenient accuracy converges to the $[0.86, 0.90]$ band by iteration 3 across all five techniques in Table 4, a pattern confirmed by the matched human-reviewer trajectory in §4.4. Strict accuracy, by contrast, does *not* converge within the three iterations observed in any of the five conditions; its within-trajectory swings remain qualitative.

The take-away for SQ4 within-corpus is that on MIMIC-III discharge summaries the loop reaches a steady lenient-accuracy band within three iterations, while strict accuracy does not settle within that window; the lenient band is a property of the gold-label regime rather than of any technique. Generalisation to other clinical text types is left to §6.1.

4.6 Qualitative analysis

The LLM judge accepts legitimate clinical synonymy and sub-typing. Representative correct cases (clinical chapters) include: *Mental (subj. 45054)*, annotator "Acute alcohol intoxication with altered mental status" vs gold "Alcohol abuse, unspecified"; *Endocrine (subj. 58965)*, annotator "Hyponatremia" vs gold "Hyposmolality and/or hyponatremia"; *Respiratory (subj. 50911)*, annotator "Community-acquired pneumonia with empyema and sepsis" vs gold "Bacterial pneumonia, unspecified". In each case the judge correctly identifies the annotator's answer as a valid specialisation or synonym.

Incorrect cases illustrate distinct failure modes: *Missed entity (subj. 23257)*, annotator "Hyperkalemia" vs gold "Acute kidney failure, unspecified"; the annotator extracted a complication rather than the underlying primary, but under lenient scoring this flips to correct (hyperkalemia is one of the admission's billed secondary codes). *Unsupported inference (subj. 27657)*, annotator "Non-epileptic seizures" vs gold "Epilepsy"; the annotator inferred "non-epileptic" from a negative EEG but the patient was discharged on anti-epileptic medications.

5 Discussion

The robust finding from §4.2 is cross-condition lenient convergence to $[0.86, 0.90]$ by iteration 3, consistent with the band being a property of the gold-label regime rather than of any prompt-engineering strategy. On strict accuracy, no per-technique difference is statistically resolvable, and the ceiling visible at +18pp is best read as a boundary condition on what HITL can achieve under this gold-label regime rather than as a HITL failure; Baseline is the only condition with an upward strict trend within that ceiling, the direction expected if the HITL loop improves the bare prompt.

The SQ1 dominant failure mode is missed entity, and only Baseline shows a monotonic reduction in it across iterations. We attribute the lack of reduction in the technique-enriched conditions to *prompt bloat*: PromptAgent patches compete with the techniques' own context expansions (few-shot exemplars, CoT scaffolding, retrieved cases) for the small-model annotator's attention budget [11, 9], and the steepest strict declines (Few-shot, RAG, and to a lesser extent CoT) occur on the conditions whose prompts grow fastest.

RAG inherits this dynamic with two specific asymmetries. Retrieval was available for the modal missed-entity failures (the store is populated by all validated and corrected non-ambiguous cases, queried only by RAG), so the failure to help is not a coverage shortage. Rather, the technique's context growth is unauditable (File Search exposes neither embeddings, top-k, nor retrieved text, only chunk counts), and the system prompt instructs the model to "prioritize [the retrieved] logic over your general knowledge," so when retrieval fires the authored patches are by design *superseded* by the retrieved case. Both effects predict the same sign and cannot be separated at this scale.

The deployment-relevant implication is that "run until convergence" is the wrong target on strict for the context-expanding techniques: their strict trajectories peak at iteration 1 and decline thereafter, so principled stopping rules or context-budget management belong in the higher-powered follow-up (§6.1). Self-consistency, which does not expand the prompt, stays roughly flat over the trajectory and shows no analogous peak-then-decline.

A natural follow-up question is whether the system performs better than a human coder. The gold labels are themselves human-produced (MIMIC-III primary ICD-9 codes are assigned by hospital coders following billing conventions), so every strict-accuracy number in this paper is implicitly a system-vs-human comparison: the system reproduces the original coder's primary choice on 67% of clinical-chapter admissions (the hand-verified $n = 100$ baseline). Three external reference points place this in context.

First, published audits of routine ICD coding report a median diagnostic-coding accuracy of 80.3% with interquartile range 63.3–94.1% across 32 studies [3], and primary-diagnosis change rates of 8.7–16.8% on re-audit [14]. Inter-rater agreement is similarly bounded: $\kappa = 0.77$ for primary-diagnosis ICD-9 coding of stroke discharges [8], and $\kappa = 0.42$ at terminal-code level rising to $\kappa = 0.72$ at chapter level across 13 specialist coders [17]. Second, MIMIC-III's labels are themselves demonstrably incomplete: the most frequent codes are under-coded by up to 35% [16]. Third, the best replicated automated medical coder on this corpus reaches micro-F1 ≈ 0.598 [4].

Our 0.67 strict and 0.85 lenient (human-reviewed baseline) therefore sit inside the published professional-coder interquartile range (63.3–94.1%) and above prior automated-coder baselines, while still bounded above by the billing-convention measurement artefact we documented.

The findings carry different implications for different audiences. For *ICD-coding ML researchers*, the dual-metric protocol suggests that published MIMIC-III leaderboard numbers (e.g. [4] above) may systematically under-state model capability by conflating billing-primary mismatch with clinical extraction failure; re-reporting against any-billed-ICD lenient accuracy would put prior results on a more comparable footing. For *LLM-as-judge methodology researchers*, our judge's binary verdict validates well against hand-

review (§3.4) but its failure-mode labels are not validated, so any downstream analysis that conditions on the judge's failure-mode label should treat that surface as provisional. For *clinicians and clinical reviewers*, the human-validation trajectory (§4.4) shows that free-text rationales translate into instruction-level prompt patches that generalise beyond their originating cases (the principal-diagnosis rule example), supporting the practical viability of a clinician-in-the-loop deployment where the clinician's time is spent on rationale-rich review rather than case-by-case correction. For *hospital coding and billing operations*, the 18pp clinical strict/lenient gap is a direct quantification of how much of an automated coder's apparent error budget is in fact "correct clinical extraction, wrong billing-priority" rather than a failure of clinical understanding; this is the operationally relevant decomposition when scoping where automation can offload work versus where human judgement on DRG-driven primacy is still required. For *patients and broader healthcare-system stakeholders*, the gold-label-ambiguity finding (≈10% on-corpus, up to 14%) implies that even the human-produced reference labels used to evaluate medical NLP systems carry non-trivial inconsistency, with implications for any downstream care or billing decision conditioned on a single "primary" diagnosis.

Three foreseeable risks accompany the contributions reported here. First, an *over-reliance risk* for clinical reviewers: if a HITL system of this kind is deployed at scale and clinicians are asked only to validate (rather than originate) extractions, sustained exposure to mostly-correct LLM annotations may erode the critical-reading skill the validation step depends on, a deskilling pattern documented in adjacent automation contexts. Mitigation requires periodic blind review of cases without an LLM suggestion present and explicit measurement of reviewer agreement drift over time. Second, a *downstream-billing risk* for hospital coding operations: the same 18pp clinical/lenient gap that motivates the dual-metric protocol also means that an LLM-extracted diagnosis can be clinically valid yet billing-inappropriate, and naive deployment without a human-in-the-loop step before billing submission could systematically mis-prioritise comorbidities under DRG reimbursement schemes. Third, a *methodological-overclaim risk* for the research literature: the dual-metric protocol exposes a measurement ceiling but does not eliminate it, and re-reporting prior MIMIC-III leaderboard results under lenient matching should be accompanied by both metrics rather than used to revise headline F1 numbers upward in isolation. That the judge's failure-mode labels are unvalidated (§5.1) compounds this: any meta-analysis that conditions on those labels inherits that uncertainty, so downstream uses of this framework's logs should treat the binary verdict as substantive and the per-mode label as provisional.

5.1 Limitations

The empirical findings reported here should be read as a *pilot characterisation*: every HITL trajectory in the paper is a single run at $n = 50$ per iteration, every Phase 3 finding rests on the author's reviews of 150 charts, and no human reviewer in this study is a practising clinician. The four caveats below elaborate these constraints along the classical validity axes (internal, construct, statistical conclusion, and external).

Internal validity. Each HITL iteration draws a fresh stratified $n = 50$ sample (§3.5). This design tests whether the patched prompt generalises across charts rather than measuring training-set fit on cases the patch was written for, but as a consequence iteration-to-iteration accuracy differences confound prompt improvement with between-batch variation in chart difficulty. A fixed held-out set evaluated at every iteration would separate these and is part of the higher-powered follow-up proposed in §6.1.

Construct validity. The headline accuracy numbers depend on an LLM judge. Its binary verdict is validated by the author's hand-review of the $n = 100$ baseline (§4): the hand and judge verdicts agree at Cohen's $\kappa = 0.82$ on the strict verdict (92% raw agreement) and $\kappa = 0.77$ on the lenient verdict (95%), both in the *substantial*-to-*almost-perfect* range, so the binary verdict supporting the strict-accuracy comparisons is reliable at this sample size. Two caveats remain. First, this is an *author* hand-review, not an independent practising clinician; validation against a recruited clinician is future work (§6.1). Second, the judge's categorical failure-mode label is *not* separately validated: we report and act on the binary correct/incorrect verdict and treat the failure-mode label as provisional rather than a validated measurement. Accuracy differences smaller than the ∼5–8% implied per-verdict disagreement rate should not be treated as separable from judge stochasticity at single-condition resolution. A related construct-side threat is the gold-label regime: our gold labels inherit MIMIC-III's billing conventions, and the dual strict/lenient metric and the admission-scope filter address the most extreme artefacts, but ICD-9 codes within clinical chapters can still be chosen by DRG considerations rather than clinical primacy, imposing a residual measurement ceiling on strict accuracy.

Statistical conclusion validity. The HITL trajectories are scoped as a pilot at $n = 50$ per iteration, which is **underpowered** to resolve small between-technique differences in strict accuracy. The headline human-reviewed baseline ($n = 100$, $p = 0.67$) has a 95%

CI of approximately ±9pp. At $n = 50$ per iteration, the single-iteration CI widens to roughly ±12pp, so an iteration-to-iteration difference carries a 95% CI of approximately ±17pp. We therefore read the per-technique strict-accuracy trajectories in Table 4 and Fig. 4 as *qualitative trend directions*, not statistical inferences, and draw firmer conclusions from (i) the cross-condition lenient convergence pattern, (ii) the failure-mode shifts in Table 6, and (iii) the on-corpus gold-label-ambiguity rate from the human-validation trajectory. Resolving a ∼5pp per-technique difference at 80% power would require approximately $n = 300$ per iteration; that is the principal follow-up experiment named in §6.1.

External validity. All experiments use Gemini 2.5-flash. Generalisation to other model families (GPT-4 class, Claude, Llama) is not evaluated. The choice of a small/medium-capacity hosted model is deliberate: a hospital deploying this framework in practice will run a locally-hosted open-weights model for data-residency and cost reasons, and that target operating point is closer to Gemini-flash than to a frontier API model. The findings reported here (the strict/lenient ceiling, the prompt-bloat trajectory, the modal missed entity failure budget) should therefore transfer more directly to a local deployment than they would if the experiments had been run on a frontier model whose extra capacity might mask long-context degradation [11]. A second external-validity caveat is the corpus: the trajectories were run only on MIMIC-III discharge summaries, and cross-text-type generalisation (e.g. radiology or nursing notes) is the SQ4 completion proposed in §6.1.

6 Conclusion

We can now return to the main research question. On *reliability*, the HITL loop yields convergence: lenient accuracy lands in a consistent inter-technique band by iteration 3 regardless of starting technique. On *accuracy*, gains on strict are bounded above by the gold-label ceiling, and within that ceiling are directionally positive but unconfirmed at this sample size (the human-reviewer trajectory's $0.62 \rightarrow 0.78$ signal, confounded with batch difficulty). On *mechanism*, the operative lever is instruction-level prompt patching driven by free-text human rationales rather than retrieval or technique choice; Appendix B.9 reproduces a single chart-through-the-loop trace.

The headline contribution of this work is methodological: the MIMIC-III primary-ICD-9 label, when used as gold for LLM annotation evaluation, imposes a measurable upper bound on strict accuracy that conflates model failure with billing-convention noise. We proposed a dual-metric protocol that exposes this ceiling quantitatively (both metrics scoped to the clinical ICD-9 chapters) and demonstrated it across a two-agent human-in-the-loop framework with five prompt-engineering conditions.

The framework delivers on reliability and partially on accuracy. It works for reliability because the loop converges all five techniques to the same lenient band by iteration 3, with the convergence reflecting the gold regime more than any prompt-engineering decision. It is bounded on strict accuracy because the gold label encodes billing-priority that prompt engineering alone cannot recover; within that ceiling, instruction-level patches driven by rich free-text rationales are the mechanism that moves the system upward, while context-expanding techniques (Few-shot, RAG) degrade because their context expansions compete with those patches for the small-model annotator's attention budget [11, 9]. Published audits of professional ICD coding (median diagnostic-coding accuracy ≈ 0.80; primary-diagnosis $\kappa \approx 0.77$ on ICD-9 stroke coding) place the system inside the human-coder inter-rater agreement band, suggesting that the residual strict-accuracy gap is the same gap human coders face, not a model-specific shortfall.

We recommend that future LLM-based ICD-coding evaluations on MIMIC-III adopt the dual-metric protocol so that model capability and billing-convention noise are reported separately, and that within-corpus convergence claims include both metrics rather than reporting strict accuracy as a single number.

6.1 Future Work

Several planned experiments extend the present study and will be completed in the remaining project window.

A full longitudinal study would run six conditions (baseline, RAG, chain-of-thought, few-shot, CoT+few-shot, and the full pipeline) across three iterations at $n = 300$ on the clinical-scoped sample, the per-iteration size our power analysis (§5.1) indicates is needed to resolve 5pp HITL effects, to answer SQ2 (which techniques benefit most from HITL) with tighter confidence intervals.

An independent clinician agreement study would address the present study's reliance on the author's hand-review for judge validation (§3.4). Recruiting one or more practising clinicians, blind to the judge verdict, to re-label a chapter-balanced sample would validate the verdict against clinical expertise rather than the author, and would additionally let the judge's failure-mode label, unvalidated here, be estimated with usable precision.

Extended HITL trajectories at higher power would resolve the per-technique trend directions statistically. The per-technique three-iteration trajectories at $n =$

50 reported in §4.2 are preliminary, underpowered evidence (see §5.1) with a fresh-batches-per-iteration design that confounds prompt improvement with batch difficulty. Replicating the headline conditions at $n = 300$ per iteration with a fixed held-out test set across iterations would allow a more definitive convergence answer for discharge summaries.

Cross-text-type convergence (SQ4 completion) would re-run the HITL trajectory on a second corpus, for example radiology reports or nursing notes from MIMIC-III's other note categories, to test whether the convergence behaviour observed on discharge summaries generalises across clinical text types, completing the second half of SQ4.

Per-chapter stratified reporting would provide a primary results figure showing strict and lenient accuracy per ICD-9 chapter across conditions, enabling a finer-grained answer to which admission types benefit most from HITL.

Disease-category-specific prompting follows from the observation that the strict/lenient gap is uneven across ICD-9 chapters (Fig. 3), which suggests that a single global prompt is unlikely to be equally well suited to every disease category. A natural follow-up routes each note through a lightweight upstream classifier that predicts its likely ICD-9 chapter and then applies a prompt specialised for that category rather than one prompt for all admissions. A coarse version groups chapters into broad clinical versus administrative regimes; a finer version learns or hand-tunes a separate prompt per disease category, testing whether category-aware conditional prompting lifts accuracy on the chapters where the current single prompt is weakest.

Dynamic stopping and context-budget management would convert the prompt-bloat finding into a deployment-ready policy. The per-technique strict trajectories peak at iteration 1 for the context-expanding techniques (RAG and Few-shot; §4.2, §5) and decline thereafter, so a uniform iteration count is sub-optimal on strict. A principled stopping rule, or a context budget that caps retrieved or few-shot examples once accuracy stops improving, would address this.

An instruction-count versus prompt-length dose-response study would turn the qualitative prompt-bloat reading into a quantitative threshold usable by the stopping rule above. The present study observes prompt-bloat indirectly, by reading accuracy against the total character count reached at iteration 3 (Table 5). A controlled follow-up would vary the number of patched instructions independently of the surrounding technique scaffolding (few-shot exemplars, CoT preamble, retrieved cases) and measure strict and lenient accuracy as a function of both axes, separating the contribution of *instruction count* from *total prompt length* and identifying whether a saturation point exists beyond which additional patched instructions stop helping or actively hurt.

Code and data availability

The codebase, prompts, hyperparameter configurations, random seeds, and supplementary review spreadsheets used to produce every number reported in this paper are released publicly at `https://github.com/surftijmen/agentic-hitl-medical-annotation`. Per-case experimental records are *not* redistributed because they embed MIMIC-III note text, which the PhysioNet credentialed Data Use Agreement prohibits sharing with third parties; researchers with PhysioNet credentials can reproduce every reported number by running the released code against their own MIMIC-III copy. Single-call inference (Annotator, judge, and PromptAgent calls to Gemini 2.5-flash) was issued under a zero-data-retention configuration so that no MIMIC note text persisted on Google's servers beyond the request lifetime. The RAG condition is an explicit exception: it relies on Gemini File Search, an inherently persistent vector store, and uploads note-derived retrieval documents (∼500 characters per case; see Appendix B.7) that remain on the File Search backend for the trajectory's duration. Each trajectory was run against a freshly created store so no MIMIC-derived content crossed between trajectories within the experiment, and operational deployment of this framework should include a post-experiment teardown step that deletes the store via the File Search API to limit DUA exposure beyond the experiment.

Acknowledgements and use of AI tools

The research design, experimental decisions, analyses, and interpretation of results in this paper are the author's own work. Anthropic's Claude Code (Claude Opus 4.7) was used as a software-engineering assistant during implementation of the codebase, and as a writing assistant during revisions to specific sections of this paper (notably for tightening prose, restructuring paragraphs, and verifying that reported numbers match the underlying experiment logs). All claims, interpretations, and the framing of the methodological contribution were determined by the author; AI tools served as drafting and verification support, not as the source of the scientific content.

References

[1] Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. Large language models are few-shot clinical information extractors. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1998–2022, 2022.

[2] Samuel Budd, Emma C Robinson, and Bernhard Kainz. A survey on active learning and human-in-the-loop deep learning for medical image analysis. *Medical Image Analysis*, 71:102062, 2021.

[3] Edward M Burns, Elaine Rigby, Ravikrishna Mamidanna, Alex Bottle, Paul Aylin, Paul Ziprin, and Omar D Faiz. Systematic review of discharge coding accuracy. *Journal of Public Health*, 34(1):138–148, 2012.

[4] Joakim Edin, Alexander Junge, Jakob Drachmann Havtorn, Lasse Borgholt, Maria Maistro, Tuukka Ruotsalo, and Lars Maaløe. Automated medical coding on MIMIC-III and MIMIC-IV: A critical review and replicability study. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2023.

[5] Alvan R Feinstein and Domenic V Cicchetti. High agreement but low kappa: I. the problems of two paradoxes. *Journal of Clinical Epidemiology*, 43(6):543–549, 1990.

[6] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations (ICLR)*, 2024.

[7] Alistair E W Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 3(1):160035, 2016.

[8] Maurizio A Leone, Paola Gaviani, and Giovannino Ciccone. Inter-coder agreement for ICD-9-CM coding of stroke. *Neurological Sciences*, 27(6):445–448, 2006.

[9] Mosh Levy, Alon Jacoby, and Yoav Goldberg. Same task, more tokens: the impact of input length on the reasoning performance of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

[10] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wentau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474, 2020.

[11] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[12] Tjardo D Maarseveen, Daniyal Selani, Nils Steinz, Robin Ten Brinck, Herman K Glas, Josien Verisvan Dieren, Marcel J T Reinders, Erik B van den Akker, and Rachel Knevel. Work smarter, not harder: achieve expert-level diagnosis extraction from medical records with optimal prompting of large language models. *Annals of the Rheumatic Diseases*, 85(4):778–780, 2026.

[13] MIT Laboratory for Computational Physiology. On the priority ordering of billed diagnoses in the MIMIC database. MIT-LCP mimic-code repository, issue #319, `https://github.com/MIT-LCP/mimic-code/issues/319`. Maintainers note that the billing-priority ordering of an admission's diagnoses is an imperfect ranking of clinical importance (accessed 2026-05-29).

[14] S A R Nouraei, A Hudovsky, A E Frampton, U Mufti, N B White, C G Wathen, G S Sandhu, and A Darzi. A study of clinical coding accuracy in surgery: implications for the use of administrative big data for outcomes management. *Annals of Surgery*, 261(6):1096–1107, 2015.

[15] Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

[16] Thomas Searle, Zina Ibrahim, and Richard Dobson. Experimental evaluation and development of a silver-standard for the MIMIC-III clinical coding dataset. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing (BioNLP)*, 2020. Reports systematic

under-coding of frequent ICD-9 codes in MIMIC-III by up to 35%.

[17] Jürgen Stausberg, Nils Lehmann, Daniela Kaczmarek, and Manfred Stein. Reliability of diagnoses coding with ICD-10. *International Journal of Medical Informatics*, 77(1):50–57, 2008.

[18] Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P Xing, and Zhiting Hu. PromptAgent: Strategic planning with language models enables expert-level prompt optimization. In *International Conference on Learning Representations (ICLR)*, 2024.

[19] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[20] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[21] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. A prompt pattern catalog to enhance prompt engineering with ChatGPT. *arXiv preprint arXiv:2302.11382*, 2023.

[22] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. In *International Conference on Learning Representations (ICLR) Workshop*, 2024.

[23] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

A Abbreviations

CI	Confidence Interval
CoT	Chain-of-Thought (prompting technique)
DRG	Diagnosis-Related Group (reimbursement classification)
EHR	Electronic Health Record
F1	Harmonic mean of precision and recall
HITL	Human-In-The-Loop
ICD	International Classification of Diseases
ICD-9 / ICD-10	9th / 10th revision of ICD
ICU	Intensive Care Unit
LLM	Large Language Model
MIMIC	Medical Information Mart for Intensive Care
NLP	Natural Language Processing
RAG	Retrieval-Augmented Generation
RQ	Research Question
SQ	Sub-Question (SQ1 – SQ4 in this paper)
UHDDS	Uniform Hospital Discharge Data Set
pp	percentage points
κ	Cohen's kappa (inter-rater agreement coefficient)

B Reproducibility specifics

This appendix lists the verbatim prompts, hyperparameters, and implementation details required to reproduce every reported number by re-running the released code on a credentialed MIMIC-III copy. Pointers reference modules in the released codebase.

B.1 Hyperparameter table

B.2 Annotator prompt (v1, baseline)

The v1 prompt is captured as a versioned JSON file in the released codebase. The four-field JSON scaffold (`system_prompt`, `query`, `instructions`, `format_instruction`) and the three-key output schema (`diagnosis`, `is_diagnosis_given`, `confidence_level`) are adapted from the rheumatic-disease diagnosis-extraction prompt scaffold released by Maarseveen et al. [12] (`prompt_rachel_v1.json` in their public repository), shared with the author by co-supervisor D. Selani; the nine task instructions themselves are written for the MIMIC-III primary-ICD-9 setting. The active fields are:

System prompt.

You are a medical information extraction assistant. Extract only information that is explicitly stated in the text. Do not guess or infer. ALWAYS search the provided `file_search` tool for similar validated cases before providing an annotation. If you find a similar case in the 'medical-annotation-rag' store, prioritize that logic over your general knowledge to ensure consistency across the dataset.

Query.

From the following medical text, identify the single primary confirmed diagnosis for this admission.

Instructions (nine items).

1. Return the PRIMARY confirmed diagnosis for this admission, the billed principal diagnosis that best describes why the patient was admitted.
2. Express the diagnosis using standard clinical terminology (e.g. 'Acute myocardial infarction', 'Congestive heart failure'). Common unambiguous abbreviations are permitted (MI, CHF, UTI, DVT, PE, CKD, AKI, COPD, ICH).
3. Be as specific as the note supports (e.g. 'Acute inferior STEMI' rather than just 'MI') but do not invent specificity that isn't in the text.
4. Do NOT include suspected, possible, or rule-out diagnoses unless explicitly confirmed.
5. Ignore negated findings (e.g. 'no evidence of').
6. If multiple confirmed diagnoses are present, return the one that is the principal reason for admission, not a secondary or chronic comorbidity.
7. Distinguish diagnoses from symptoms: 'chest pain' is a symptom, not a diagnosis; return the underlying confirmed diagnosis instead.
8. If no primary diagnosis is confirmed in the note, return 'none'.
9. Do not use external knowledge beyond the text.

Output schema (response_schema-enforced).

```
{"diagnosis":  string or "none",
"is_diagnosis_given":  0 or 1,
"confidence_level":  int in [0,100]}
```

Open-vocabulary extraction. The prompt schema also supports an optional `allowed_diagnoses` closed list, but it is left *empty* in every reported run: the task is open-vocabulary extraction scored against the free-text ICD-9 long-title. The "map confirmed conditions to the allowed-diagnoses list" step in the chain-of-thought variant below and the "do not change the `allowed_diagnoses` list" PromptAgent constraint are therefore inert holdovers from an earlier closed-list prototype and did not constrain the reported experiments.

B.3 Chain-of-thought variant

When chain-of-thought is enabled (the CoT condition, §4.2) the output schema is extended with a `reasoning` field and the following instruction is prepended:

Before answering, reason step-by-step inside the 'reasoning' field: (1) list every medical condition mentioned in the text; (2) for each condition, decide whether it is explicitly confirmed, not 'possible', 'suspected', 'rule out', or negated; (3) map confirmed conditions to the allowed-diagnoses list using synonym mapping where needed; (4) select the most clinically relevant confirmed diagnosis, or 'none'. CRITICAL: only reference words and phrases that literally appear in the text. Do NOT invent section names, quotes, or content that is not present.

B.4 Static few-shot examples

Five examples are statically prepended to the prompt when `use_few_shot` is set. Each example is a triple of (*note excerpt*, *expected JSON*, *expected reasoning trace for the CoT-enabled variant*). The set is chosen to cover four edge cases beyond a single positive: (i) simple explicit confirmation; (ii) rule-out / not yet confirmed (negative example); (iii) clinical indicators present but the diagnosis word absent (must *not* infer); (iv) the same presentation with the diagnosis explicitly named (*must* commit); and (v) multi-comorbidity primary selection. The full text of each example is in `src/modules/annotator_agent.py` (`FEW_SHOT_EXAMPLES`).

B.5 Judge prompt and failure-mode definitions

The judge receives the full note, the annotator's output (including any CoT reasoning), and the gold ICD-9 long_title. The failure-mode definitions in the judge's system prompt are verbatim:

Failure-mode guide (pick the MOST SPECIFIC):

- *hallucination: diagnosis is not supported by anything in the note.*
- *missed_entity: a clearly confirmed diagnosis was ignored (annotator said 'none' or picked something else while the correct concept is plainly stated).*
- *terminology_gap: correct concept, wrong or non-clinical terminology that obscures the match.*
- *unsupported_inference: annotator inferred / assumed a diagnosis not explicitly stated.*
- *symptom_diagnosis_confusion: annotator returned a symptom or sign, not a diagnosis.*
- *ambiguous_case: ONLY when it is genuinely impossible to judge (note too short / corrupted; gold title too vague, e.g. "Other specified disorder").*

The output schema is:

```
{"correct": bool, "failure_mode":
enum|"none", "comment": str,
"confidence": int in [0,100]}
```

with `failure_mode` restricted by the response schema to one of the six enum values (or `"none"` when `correct=true`).

B.6 PromptAgent meta-prompt

At the end of each iteration, the PromptAgent aggregates failures by mode and submits the following meta-prompt to Gemini 2.5-flash. The constraint list is critical: it confines the agent to instruction-level edits and forbids scope expansion or schema changes.

> *You are assisting with IMPROVING an existing medical information extraction prompt.*
>
> *IMPORTANT CONSTRAINTS (DO NOT VIOLATE):*
>
> - *You MUST NOT rewrite or rephrase the task.*
> - *You MUST NOT change the system prompt.*
> - *You MUST NOT change the query.*
> - *You MUST NOT change the allowed_diagnoses list.*
> - *You MUST NOT change the output JSON format.*
> - *You MUST NOT introduce new task scope.*
>
> *You are ONLY allowed to:*
>
> - *Propose ADDITIONAL instructions.*
> - *Propose SMALL refinements to EXISTING instructions.*
>
> *The goal is to reduce the observed failure modes WITHOUT harming recall of explicitly stated diagnoses.*

The current base prompt and the per-mode aggregated failures (`count`, `error_locus`, up to three reviewer example sentences) are appended verbatim. The required JSON output schema is:

```
{"instructions_to_add": [str,
...], "instructions_to_refine":
[{"target_instruction": str,
"refined_version": str}, ...],
"rationale": str}
```

Each patch is timestamped and persisted as a versioned JSON file before deployment, so the entire iteration history is reconstructible from the released prompt sequence.

B.7 Retrieval-augmented memory

RAG is implemented on top of Google's Gemini File Search API (`src/modules/rag_memory.py`). Each validated correct case is serialised as a ∼500-character plain-text retrieval document followed by a JSON metadata block, written to disk, and uploaded to the cloud store; deduplication is by `display_name = case-⟨subject_id⟩-⟨hadm_id⟩`. At inference time the annotator is given a tools list containing `types.Tool(file_search=FileSearch(...))`; the Gemini runtime performs embedding, retrieval, and grounding internally. Grounding metadata returned in the response (number of chunks used) is logged per call for traceability. File Search is a persistent backend: uploaded documents remain on Google's infrastructure for the duration of the trajectory and across all inference calls within it. A fresh store is created per trajectory so no content crosses between trajectories during the experiment; stores from prior trajectories are not implicitly deleted by the API, and operational deployments should add an explicit teardown call to limit residency beyond the experiment.

B.8 Reproducibility

Every reported number is reproducible by running the released code against a credentialed MIMIC-III copy under the seeds and hyperparameters in Table 8. Each experimental run writes a per-case record (note ID, annotator output, judge verdict, failure mode, confidence, and gold reference), a versioned prompt JSON capturing the prompt active at that iteration, and a paired strict-plus-lenient re-judge for every HITL run; the trajectory is reconstructible from this sequence. Because

the per-case records embed MIMIC-III note text, they are not redistributed, but the generating code is.

Figures, tables, and every numerical claim in the paper are regenerated by deterministic post-processing scripts that consume only the per-run JSON records and issue no API calls. A separate verification script independently re-derives each headline number and emits a per-claim pass/fail report.

The per-case `prompt_version` field records the working filename of the prompt applied to that case. Generic working filenames are reused across runs and overwritten by later ones, so the materialised per-iteration prompts of each reported trajectory are archived verbatim under run-specific subdirectories in the released codebase, where iteration-2 and iteration-3 prompts are preserved alongside the shared iteration-1 prompt above.

B.9 Worked example: one chart through the loop

This trace instantiates the SQ1 finding that human rationales drive patches that generalise across charts (§4.4). It follows a single iteration-1 chart from the paired-human Baseline trajectory through the reviewer verdict, the resulting PromptAgent patch, and a different iteration-2 chart on which the patch fired correctly. All stages are reproduced verbatim from the released records, except the source-note descriptions which are paraphrased to one sentence (MIMIC note text is not reproduced; see Code and data availability).

Stage 1, source chart (iteration 1). A 67-year-old male admitted via the surgical service for ERCP after returning with right upper quadrant pain and abnormal liver function tests; the discharge summary documents a stone-obstructed bile duct with pus on extraction, transfusion for coagulopathy, and a return to baseline post-procedure.

Stage 2, annotator output under the v1 prompt (verbatim).

```
{"diagnosis": "Cholangitis",
"is_diagnosis_given": 1,
"confidence_level": 95}
```

Stage 3, gold reference. ICD-9 `038.9`, "Unspecified septicemia".

Stage 4, human-reviewer verdict. Failure mode `symptom_diagnosis_confusion`. Free-text rationale (verbatim):

> *The AI named cholangitis (bile duct infection) while the gold codes unspecified septicemia. Cholangitis could be a source of sepsis, but the AI captured the focal infection rather than the systemic septic state that the gold codes as primary.*

Stage 5, PromptAgent patch (added to v2 prompt, verbatim). The reviewer rationale, aggregated with other iteration-1 failures under the same mode, drove a refinement of the principal-diagnosis instruction. The new instruction text reads:

> *If multiple confirmed diagnoses are present, return the single principal diagnosis that best explains the patient's need for inpatient admission, rather than a secondary diagnosis, a chronic comorbidity, or a contributing cause of the principal diagnosis.*

The clause "a contributing cause of the principal diagnosis" is the generalising element: it abstracts the cholangitis-as-cause-of-sepsis correction into a rule that applies to any focal infection underlying a systemic state.

Stage 6, generalised behaviour at iteration 2 (verbatim). A different chart from the iteration-2 sample (an 80-year-old male admitted via the medical service with dyspnea on a background of CHF and COPD, coded as unspecified septicemia) was annotated under the v2 prompt as:

```
{"diagnosis": "Sepsis due to RLL
pneumonia and UTI", ...}
```

The annotator named the systemic septic state and identified the focal infections (pneumonia, UTI) as causes rather than returning either as the primary diagnosis. The iteration-2 chart shares no entities with the iteration-1 chart, so the patch's effect on this case is instruction-level rule generalisation rather than retrieval, the SQ1 mechanism quantified in §4.3 and discussed in §5.

C Qualitative cases

Three representative cases illustrate the headline error classes discussed in §4.

Correct, sub-typing (subj. 50911): annotator "Community-acquired pneumonia with empyema and sepsis" vs. gold "Bacterial pneumonia, unspecified", the judge accepts the more specific clinical phrasing as a valid sub-type.

Incorrect, missed entity (subj. 23257): annotator "Hyperkalemia" vs. gold "Acute kidney failure", a complication extracted in place of the underlying primary; this flips to correct under lenient scoring (hyperkalemia is billed as a secondary diagnosis for the admission).

Incorrect, unsupported inference (subj. 27657): annotator "Non-epileptic seizures" vs. gold "Epilepsy", the annotator inferred "non-epileptic" from a negative EEG even though the patient was discharged on anti-epileptic medications.

Table 8: Hyperparameters used across every experiment.

Parameter	Value
Annotator model	Gemini 2.5-flash (all reported conditions)
Judge model	Gemini 2.5-flash
PromptAgent model	Gemini 2.5-flash
Annotator temperature (single-pass)	0.0
Annotator temperature (self-consistency)	0.7
Self-consistency samples	$N = 3$
Judge temperature	0.0
PromptAgent temperature	0.2 (max_output_tokens = 5000)
Random seed (sampler)	20260414
Sample size	$n = 100$ human-reviewed for the headline no-HITL baseline; $n = 50$ per iter for the five-condition per-technique HITL longitudinal study; $n = 50$ per iter for the human-reviewer validation trajectory
Stratification	By ICD-9 chapter
Held-out set	200-row set carved off and reserved for the follow-up study (§6.1); not used for the reported numbers
Output-schema enforcement	Gemini structured-output `response_schema` (both annotator and judge)
RAG retrieval backend	Gemini File Search API (cloud vector store)
RAG document format	Plain-text summary "*Patient case diagnosed with X. Key context: ⟨first 400 chars⟩*" followed by a JSON metadata block
RAG deduplication key	`case-`⟨`subject_id`⟩`-`⟨`hadm_id`⟩
RAG store membership	Both validated (correct) cases and corrected (incorrect, non-ambiguous) cases, the latter stored with the gold long_title as the retrieval target
RAG top-k	Set by the Gemini File Search tool; not exposed in the public API (the tool decides at call time)
Embedding model	Gemini File Search internal (not user-selectable)