Symptoms to Hospital Bill: A Multi-Agent System for Diagnosis, Procedures, and Charge Estimation from Natural Language Input

Arnav Gurudatt
arnavgurudatt@uchicago.edu
USA

Sarah Katz
sarahkatz@uchicago.edu
USA

Sarah Song
sasong@uchicago.edu
USA

Abstract

Patients enter the healthcare system with symptoms but little insight into how those symptoms will translate into downstream diagnoses, procedures, or billing codes. Existing clinical decision-support tools do not model this full administrative pipeline, leaving patients effectively "blind" to the financial consequences of their care. In particular, there is currently no reliable system that can take free-text descriptions of a patient's symptoms, demographics, and risk factors and translate them into an end-to-end estimate of likely diagnoses, procedures, and associated charges. To address this gap, we built a modular multi-agent system that simulates the end-to-end clinical administrative process from symptoms to diagnosis, procedure selection, medical coding, billing, and insurance documentation. By allowing uncertainty to accumulate across each step – mirroring real hospital and insurance workflows – we aim to evaluate how predictable, transparent, and aligned these outcomes are from the patient's perspective. A link to our code is available here.

Keywords

health informatics, medical coding, large language models, multi-agent systems, clinical decision support, healthcare billing, patient cost transparency

1 Literature Review and Motivation

Recent benchmarking work has shown that off-the-shelf large language models are surprisingly unreliable medical coders, even when they are provided with well-structured problem lists and detailed code descriptions [15]. These evaluations highlight systematic mismatches between model outputs and gold-standard billing codes, raising concerns about the safety and robustness of naively deploying generic systems in production coding workflows. Related investigations in specialized settings, such as surgical billing and coding, similarly emphasize how prompt design, evaluation methodology, and deployment constraints can strongly influence performance, and that careful benchmarking is essential before integrating these systems into real-world revenue-cycle operations [14].

At the same time, there is growing evidence that domain-specific adaptation and tooling can substantially improve coding performance. Research on fine-tuning large language models on medical coding corpora finds gains in efficiency and accuracy, highlighting the value of tailoring models to the coding domain rather than relying solely on general-purpose pretraining [10]. Complementary approaches show that equipping models with retrieval and tool-use capabilities can partially overcome the limitations of pure next-token prediction and make them "good" medical coders when they can interact with external resources such as code browsers and knowledge bases [11]. To further improve reliability, recent methods propose verification and lightweight adaptation pipelines that treat coding as a process of generating, checking, and revising candidate codes, thereby reducing hallucinations and enforcing consistency with structured clinical inputs [16]. More broadly, survey work on large language models in healthcare situates these technical advances within a wider ecosystem of clinical documentation, decision support, and administrative applications, emphasizing both the promise and the risks of LLMs across the healthcare operations stack [12]. Beyond back-end automation, there is also increasing interest in using language models to make billing workflows more legible to patients themselves. Production-oriented systems have begun to use LLMs to personalize healthcare billing support, explain charges, and help patients understand their financial obligations, reflecting a demand for tools that bridge clinical workflows and patient-facing billing experiences [13].

Notably absent from existing literature, however, is any system that combines these capabilities into a single, end-to-end pipeline that starts from a layperson's symptom descriptions, and propagates all the way through diagnosis, procedure selection, coding, and cost estimation. Prior work on benchmarking, adaptation, tool integration, verification, and patient-facing interfaces [10–16] points to a fragmented landscape: existing systems typically focus on isolated sub-tasks (e.g., code suggestion, verification, or billing explanation) rather than modeling the full pipeline that connects patients' symptom narratives to downstream diagnoses, procedures, codes, and charges. Our work explicitly targets this gap by framing coding and billing as downstream agents in a modular multi-agent architecture, allowing us to study how uncertainty, errors, and biases in earlier clinical reasoning stages propagate into financial outcomes for patients, and to evaluate whether these pipelines can be made more transparent and predictable from the patient's perspective.

2 Methodology

Our system models the administrative and clinical reasoning pipeline that transforms a patient's symptoms into diagnoses, procedures, billing codes, and financial costs. To do this, we designed a modular multi-agent architecture in which each stage produces structured outputs that serve as inputs to the next. This allows uncertainty, error, and interpretive variation to accumulate across each stage, reflecting real healthcare workflows and enabling us to study how upstream reasoning affects downstream financial outcomes. A link to our code is available here.

2.1 Diagnostic Agent

The pipeline begins with structured patient inputs derived from free-text symptoms and optional demographic and clinical context, including age, sex, and care setting. These inputs are passed to a

Diagnostic Agent, a large-language-model–driven reasoning module that infers a ranked set of likely clinical diagnoses and assigns validated ICD-10 codes.

Each patient case is represented as a `PatientCase` object containing age, sex, clinical setting, symptoms, and optional history, medications, and allergies. In the first stage, the agent performs LLM-based clinical reasoning to generate diagnostic hypotheses. A curated set of few-shot symptom–diagnosis examples is injected into a system prompt to guide model behavior and stabilize output. The patient case is serialized as JSON and passed to the LLM with enforced structured output formatting. The model returns a list of candidate diagnoses, each with an estimated probability and clinical rationale. These results are parsed programmatically and sorted by confidence score.

In the second stage, each proposed diagnosis is mapped to ICD-10 codes using a hybrid retrieval–generation framework. The diagnosis string is first passed to an `ICD10Index`, which retrieves a small set of candidate codes using semantic similarity search. These candidates are then provided to a second LLM call that selects the most appropriate codes based on the diagnosis and a condensed patient summary. To ensure coding safety and prevent hallucinated outputs, the agent strictly filters the model's selected codes against the retrieved candidate set, rejecting any codes not present in the index results. The final output is a ranked list of `DiagnosisCandidate` objects containing the diagnosis name, confidence score, rationale, and validated ICD-10 code assignments, which is then serialized for downstream procedural and billing agents.

2.2 Procedure Agent

After generating a set of plausible ICD-10-CM diagnosis codes, the patient's baseline characteristics (age, sex), medical circumstances (emergency setting, symptoms, risk factors), and agent-generated diagnoses scored by the Diagnosis Agent with a rationale are ingested by the pipeline's Procedure Agent. The overarching purpose of the Procedure Agent is two-fold:

- To generate a set of clinically realistic procedure code recommendations that could reasonably accompany the given patient and diagnosis information
- To accurately map said procedure codes between different coding protocols

Typically, hospital procedures are billed according to Current Procedural Terminology (CPT) codes, which are standardized, physician-maintained codes published by the American Medical Association (AMA) to describe medical, surgical, and procedural services in a way that payers can consistently interpret for documentation, billing, and reimbursement. Each CPT code corresponds to a specific service or bundle of services (for example, an evaluation-and-management visit, a minor procedure, or an imaging study), and payers use these codes together with diagnosis codes to determine coverage and payment amounts. In the absence of a service covered by a CPT code, a Healthcare Common Procedure Coding System (HCPCS) code can be assigned to a procedure or supply for billing purposes, particularly for items such as durable medical equipment, ambulance services, and certain drugs or injections that fall outside the core CPT code set. Frequently, HCPCS codes (particularly Level II codes) are used by Medicare and other public payers to capture services, supplies, and non-physician items that are critical for reimbursement under federal programs but not fully represented in CPT [2], [6].

One challenge, however, is that CPT codes and HCPCS Level I codes are copyrighted by the AMA, and as a result are not available in the public domain for use by non-medical professionals. To circumvent this barrier, we instead opt for another common billing protocol, the ICD-10's Procedure Coding System (ICD-10-PCS) [7], which are free and publicly available.

Our Procedure Agent pipeline queries ICD-10-PCS from a data table consisting of a valid code, its short description, and its long description, e.g.:

- `code`: X2JAX47
- `short_desc`: Inspection of Heart using TTE Comp-aid Guid, New Tech 7
- `long_desc`: Inspection of Heart using Transthoracic Echocardiography, Computer-aided Guidance, New Technology Group 7

At a high-level, the agent evaluates the patient's diagnosis and characteristics, searches through the ICD-10-PCS codes table for plausible procedures that could accompany the diagnosis, and selects the most appropriate for the patient (or, may not recommend any procedures if not deemed necessary). However, this seemingly simple approach has several practical challenges during implementation. First, there are 80,029 ICD-10-PCS codes in the 2026 data release, and even a linear search over every code for reasonable candidates would be prohibitively expensive in terms of search-time. Second, there is no actual "ground-truth" for which procedure codes correspond to which diagnoses. The decision to recommend a procedure for a patient is up to the physician's discretion, and there are many possible procedure codes that could correspond to a single diagnosis code. (In fact, the relationship is many-to-many, since a single diagnosis can legitimately lead to multiple alternative or sequential procedures, and the same procedure code can be appropriate for a wide range of underlying conditions.) The lack of a ground-truth not only makes searching for an appropriate procedure code for a given diagnosis difficult, but it also makes downstream evaluation of the system difficult as well. While "cross-walks" between coding protocols are available, these are usually proprietary and/or pay-walled by organizations like the AAPC (formerly American Academy of Procedural Coders) due to their inclusion of copyrighted CPT codes [1].

Due to the complexity at each step going from diagnosis to procedure, we opted to break down the Procedure Agent into three smaller sub-agents that work together. First, we use a **Planner Agent**, which reads the full patient case (symptoms, risk factors, setting, and ICD-10 diagnosis codes) and proposes 3 to 6 natural-language *procedure intents* that describe the key diagnostic and therapeutic procedures likely to occur during the encounter. For example, if a patient presents to the emergency department with chest pain, ST-elevation on ECG, and an ICD-10 diagnosis consistent with acute myocardial infarction, the Planner Agent might propose intents such as "urgent coronary reperfusion," "diagnostic coronary angiography," and "post-procedure hemodynamic monitoring." In other words, instead of jumping directly to ICD-10-PCS codes, this agent first articulates what the clinical team is trying to accomplish in human-readable terms that are specific to the case.

Next, we use a **Keyword Agent** that leverages lexical search to turn each intent into short phrases and character stems, which are then used to query the ICD-10-PCS table and retrieve a manageable set of plausible candidate procedure codes, significantly reducing the search space. For example, if the diagnosis and planner intent had to do with coronary angiography and percutaneous coronary intervention for a blocked heart artery, then the suggested keywords to search for would be a combination of stems, short words, and phrases such as *"angiography," "angioplasty," "stent," "coronary,"* and *"artery,"* and stems like *"cardio", "angi," "coron," "stent,"* and *"percu."* These lexical features are then matched against ICD-10-PCS short and long descriptions to filter down from tens of thousands of procedures to a focused candidate set that is plausibly relevant to that intent.

Lastly, we use a **Scoring Agent**, which conditions on the patient case, the specific intent, and the candidate ICD-10-PCS descriptions to assign each code a suitability score in $[0, 1]$ together with a brief clinical rationale. For example, given an intent of "urgent percutaneous coronary reperfusion" for a patient with an acute myocardial infarction, the Scoring Agent would assign high scores to codes describing percutaneous transluminal coronary angioplasty with stent placement in a coronary artery, moderate scores to related but less specific coronary procedures, and near-zero scores to procedures involving non-cardiac anatomy (e.g., renal artery dilation) or approaches that are inconsistent with the clinical context (e.g., open cardiac surgery in a low-acuity outpatient setting). After collating a set of candidate procedures ranked by confidence, with each candidate being assigned a rationale as well, the pipeline recommends codes based on a simple thresholded top-K selection over these scores, de-duplicating across intents to produce a small set of high-confidence procedure recommendations for each case. Concretely, we retain at most the top $K = 3$ procedures whose scores exceed a minimum confidence threshold of 0.40; if no candidate code meets this threshold for a given case, the pipeline abstains and recommends no procedures rather than returning low-confidence or clinically implausible codes.

Below, we provide a diagram illustrating the Procedure Agent's subroutines.

We also provide an input-output pair example to illustrate the Procedure Agent's behavior.

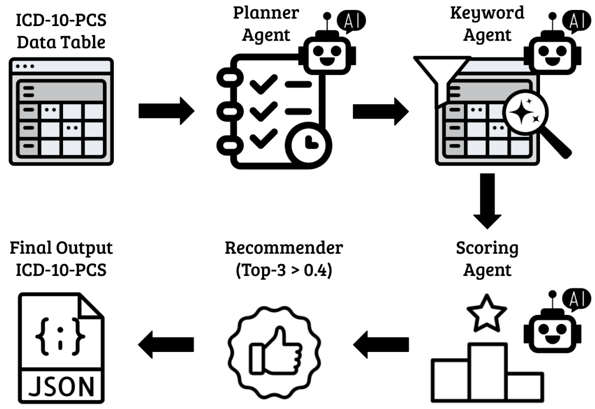


Figure 1: Overview of the Procedure Agent pipeline. The Planner Agent proposes natural-language procedure intents, the Keyword Agent turns each intent into lexical queries over ICD-10-PCS descriptions, and the Scoring Agent ranks candidate procedure codes before a final top-K selection step.

Example INPUT.

```
{
  "case_id": 1,
 "label": "Acute STEMI with unstable angina differential",
  "patient": {
    "age": 56,
    "sex": "F",
    "setting": "emergency_department",
    "symptoms": [
    "crushing substernal chest pain radiating to left arm",
      "shortness of breath",
      "nausea"
    ],
    "risk_factors": [
      "smoking",
      "hypertension",
      "hyperlipidemia"
    ]
  },
  "diagnoses": [
    {
    "name": "Acute ST elevation myocardial infarction",
      "icd10_codes": [
        "I21.3"
      ],
      "confidence": 0.86,
    "rationale": "Classic crushing substernal chest pain
     radiating to the left arm in a high-risk patient,
    consistent with acute ST elevation MI in the inferior
      wall."
    },
    {
      "name": "Unstable angina",
      "icd10_codes": [
        "I20.0"
      ],
      "confidence": 0.62,
    "rationale": "Considered as part of the acute coronary
    syndrome spectrum if ECG/troponin are non-diagnostic,
    but current presentation is more consistent with STEMI."
    }
  ]
}
```

Example OUTPUT.

```
{
  "case_id": 1,
 "label": "Acute STEMI with unstable angina differential",
  "result": {
    "recommended_procedures": [
      {
        "code": "4A12XFZ",
        "score": 1.0,
        "intent": "Continuous cardiac telemetry",
      "reason": "Monitoring of cardiac rhythm, external
     approach - exactly matches continuous ECG telemetry in
        the ED."
      },
      {
        "code": "4A133B1",
        "score": 1.0,
        "intent": "Invasive arterial blood pressure
        monitoring",
       "reason": "Exact match: monitoring of arterial
        pressure, peripheral, percutaneous approach -
     corresponds to placement of a radial arterial line."
      },
      {
        "code": "02Q03ZZ",
        "score": 0.5,
     "intent": "Percutaneous coronary intervention with
        stent placement",
     "reason": "Repair of one coronary artery, percutaneous
     approach; closest match to PCI, though device (stent)
        not specified."
      }
    ],
   "notes": "Selected top 3 procedures with score >= 0.4."
  }
}
```

2.2.1 Mapping ICD-10-PCS Codes to CPT/HCPCS Codes.

– The second goal of our two-fold Procedure Agent is a reliable mapping from ICD-10-PCS codes to CPT/HCPCS codes. Most publicly available billing estimates and Medicare fee schedules are expressed in terms of HCPCS (including CPT Level I codes), rather than ICD-10-PCS, so obtaining these mappings is essential for linking our simulated procedures to real-world cost data.

In an actual deployment environment, it's reasonable to believe that our agentic system would either have access to CPT/HCPCS codes directly – replacing the ICD-10-PCS data table in the Procedure Agent Pipeline – or have access to an ICD-10/CPT cross-walk that allows for easy look-ups between coding protocols. However, because the full list of CPT codes and cross-walks are unavailable to the public, it is difficult to give our agent access to a system that does this mapping with high fidelity. Additionally, the same many-to-many relationship that exists between diagnosis and procedure codes also exists between ICD-10 and CPT/HCPCS codes. There are many plausible CPT/HCPCS codes that could reasonably correspond to a single ICD-10-PCS procedure, and vice versa, making a hard ground-truth conversion between systems inherently ambiguous and highly dependent on clinical context and payer-specific billing conventions. This presents complications for evaluation of how well the agent can do these mappings.

In lieu of a comprehensive public cross-walk, we experimented with several progressively more informed mapping strategies. First, we tried a naive one-shot prompting approach as a baseline in which a single LLM was given the ICD-10-PCS code and description and asked to directly propose one or more CPT/HCPCS codes. A slightly more structured variant asked the model to first reason in natural language about the implied procedure (e.g., an intermediate "planning" step) and then guess the corresponding CPT/HCPCS codes. Both approaches relied purely on the model's pretraining and quickly revealed their limitations: the agent frequently hallucinated non-existent codes, proposed codes outside the relevant anatomical or procedural domain, and showed poor top-K accuracy when evaluated against health-provider-verified mappings.

To better reflect a realistic deployment scenario, we then constructed a structured billing-code lexicon from a publicly available datasheet released by Quality Health Associates of North Dakota. The datasheet lists CPT procedure codes and corresponding ICD-10-PCS prefixes for selected surgeries.[1] From this resource, we derived 95 provider-verified ICD-10-PCS → CPT groupings, expanded any CPT ranges into discrete codes, and used them to build a compact catalog of candidate CPT codes with associated category and block-level descriptions. Rather than treating this as "answer key" supervision, we view it as a realistic form of data augmentation: in any production billing environment, an agent would almost certainly have programmatic access to both ICD-10-PCS and CPT/HCPCS code sets (or even proprietary cross-walks), and its job would be to select among these known codes, not to hallucinate them from scratch.

Operationally, our final mapping procedure works as follows. For each ICD-10-PCS code, we extract short lexical phrases and character stems from its short and long descriptions, and use them to perform lexical filtering over the billing-code lexicon, yielding a small set of plausible CPT candidates (typically tens of codes rather than thousands).[2] The ICD-to-CPT **Mapping Agent** is then asked to score these candidates rather than the entire CPT universe, optionally using an external search tool (e.g., Google) to verify that each proposed CPT code's description matches the intended procedure. The agent returns a short list of codes with confidence scores, and we take the top-K predictions for evaluation.

To test whether this setup is robust in the presence of misleading but plausible alternatives, we additionally ran a "decoy" evaluation. Starting from the same provider-derived CPT catalog, we constructed an augmented lexicon that includes all ground-truth CPT codes plus a fixed number of real but irrelevant CPT codes and an equal number of synthetic decoy codes whose identifiers and block-level descriptions mimic the format of genuine CPT entries but are guaranteed not to appear in the Quality Health Associates mapping. The Mapping Agent is again restricted to choosing its top-K predictions from this augmented catalog, and we evaluate it along two dimensions: (i) standard top-K accuracy against the provider-verified mappings, and (ii) a *decoy hit rate* that measures how often any of the top-K predictions falls on a synthetic (invalid) code. This robustness test more closely approximates a realistic deployment setting—where many superficially appropriate but incorrect billing codes are available—and directly probes the model's

[1] "ICD-10-PCS Procedure Codes and Corresponding CPT Procedure Codes," Quality Health Associates of North Dakota.

[2] Implemented via simple phrase/stem matching and heuristic scoring as in `filter_candidates` and `derive_phrases_and_stems_from_text`.

tendency to hallucinate non-existent or semantically mismatched codes when operating over a larger, noisier search space.

We report our findings in the Section 3.

2.3 Billing Agent

The final stage of the pipeline is the Billing Agent, which translates predicted procedures into estimated financial charges under a simplified model of U.S. Medicare Part B reimbursement policy [4]. Whereas the Diagnostic and Procedure Agents focus on clinical reasoning and code selection, the Billing Agent uses structured billing taxonomies, fee schedules, and insurance rules. Its goal is twofold: (1) to determine whether each predicted procedure corresponds to a real, billable HCPCS/CPT code with an associated Medicare payment rate, and (2) to compute the downstream financial cost for a patient, including deductible application, coinsurance, and total out-of-pocket (OOP) responsibility. Since the Procedure Agent returns the top HCPCS/CPT codes for each ICD-10-PCS procedure, the Billing Agent selects the code with the highest confidence score for the cost estimate.

A practical billing system must operate using real procedure codes and map cleanly into payer-specific fee schedules. As mentioned previously, CPT codes are copyrighted and not publicly available, so we rely on HCPCS Level II codes and the publicly available Medicare Physician Fee Schedule (PFS) and Clinical Laboratory Fee Schedule (CLFS) datasets [8], [5]. The Billing Agent assumes access to these datasets and applies Medicare's reimbursement logic directly rather than relying on freeform LLM estimation, except in fallback cases where the fee schedules do not price a given procedure.

For each predicted procedure code, the Billing Agent attempts pricing through a deterministic, tiered hierarchy. First, the agent looks up the HCPCS/CPT code through the Medicare PFS. Most procedures, particularly physician-performed services, are in this dataset. For each code, the agent retrieves an allowed amount using locality, carrier number, facility vs. non-facility setting, and other optional modifiers. The agent then applies Medicare Part B cost-sharing rules, where the remaining deductible is applied first, then the residual amount is split into 80% Medicare payment and 20% coinsurance. If the patient has Medigap coverage, the coinsurance portion is waived. This model then outputs the allowed charge, deductible applied, coinsurance, Medicare payment, and patient OOP.

If a code does not appear in the PFS, the agent then checks the Clinical Laboratory Fee Schedule (CLFS). Lab procedures, such as diagnostic tests, fall under these laboratory pricing rules. Medicare pays for the CLFS rate, so these services have no deductible or coinsurance. If the code appears in the CLFS table, the agent returns a complete pricing breakdown using that data.

If neither PFS nor CLFS contains a given procedure, the agent resorts to a tightly controlled LLM-based fallback. Instead of directly asking the LLM for a price and risking hallucination or inconsistent outputs, the agent takes the procedure, diagnoses, and clinical context information and asks the LLM to determine a severity (low, medium, high, critical). Each tier maps to a fixed base cost. This method helps preserve interpretability and prevent LLM overreach.

For each procedure, the Billing Agent ultimately returns the input metadata and details on the unit price, total price, deductible, coinsurance calculations, Medicare payment, and a short rationale indicating whether the price was derived from PFS, CLFS, or the fallback tiered prices. These outputs are added to the system's final patient-facing summary, which includes information and reasoning from the previous two stages. This helps users trace the reasoning from the symptoms to the final financial costs.

Example Final Summary.

```
PATIENT INPUT
-------------
  Symptoms : Chest pain, shortness of breath
  Age      : 64
  Sex      : M
  Setting  : ER

DIAGNOSIS (top candidates)
--------------------------
  1. Acute myocardial infarction
     ICD-10-CM : I219
     Confidence: 55.0%
  2. Pulmonary embolism
     ICD-10-CM : I2699
     Confidence: 25.0%
  3. Aortic dissection
     ICD-10-CM : I71011
     Confidence: 10.0%

PROCEDURES & LINE-ITEM COSTS
----------------------------
  1. HCPCS/CPT: 93000  |  Intent: 12-lead
  Electrocardiogram
     ICD-10-PCS source : 4A02XFZ (score 1.00)
     Unit price        : $19.31
     Patient OOP       : $19.31
     Medicare payment  : $0.00
     Deductible applied: $19.31
     Coinsurance       : $0.00

 2. HCPCS/CPT: 93784  |  Intent: Invasive Arterial Blood
  Pressure Monitoring
     ICD-10-PCS source : 4A13XB1 (score 1.00)
     Unit price        : $62.78
     Patient OOP       : $62.78
     Medicare payment  : $0.00
     Deductible applied: $62.78
     Coinsurance       : $0.00

  3. HCPCS/CPT: 93312  |  Intent: Transesophageal
  Echocardiography (TEE)
     ICD-10-PCS source : B245ZZ4 (score 1.00)
     Unit price        : $312.37
     Patient OOP       : $222.47
     Medicare payment  : $89.90
     Deductible applied: $200.00
     Coinsurance       : $22.47
```

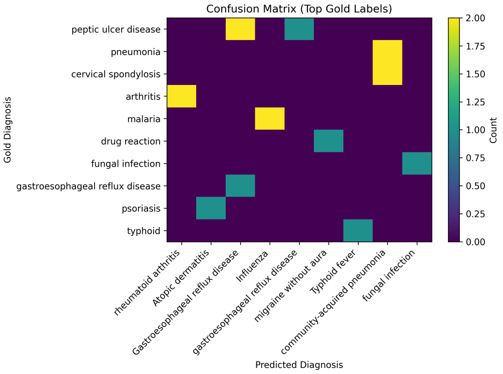
3 Results

3.1 Diagnostic Agent Evaluation

Since our goal was to evaluate how well our overall agentic system performs, we focused on scoring the performance of our various agents on specific subtasks, with the goal of isolating where errors arise, quantifying how much each component contributes to end-to-end performance, and identifying which modules offer the highest leverage for future improvement.

We evaluated the Diagnostic Agent on a held-out test set of 212 symptom–diagnosis pairs. Under strict exact-match scoring, the model achieved a Top-1 diagnostic accuracy of 23.1%, increasing to 28.8% when considering the Top-3 ranked diagnoses. Embedding-based semantic evaluation revealed substantially stronger conceptual alignment: the average cosine similarity between the gold label and the Top-1 prediction was 0.584, rising to 0.677 for the best match among the Top-3. Using a cosine threshold of 0.8, Top-1 semantic accuracy reached 25.9%, and Top-3 semantic accuracy increased to 34.4%, reinforcing that the model frequently proposes clinically relevant hypotheses even when it does not exactly match the dataset label.

To show this, we looked at the confusion matrix on 20 test cases.



Confusion Matrix on 20 cases: The confusion matrix demonstrates that the model's diagnostic errors are largely clinically interpretable rather than arbitrary. Several incorrect predictions reflect meaningful real-world diagnostic overlap, including peptic ulcer disease being misclassified as gastroesophageal reflux disease, psoriasis being confused with atopic dermatitis, and typhoid overlapping with typhoid fever. Similarly, pneumonia is occasionally predicted as community-acquired pneumonia, reflecting reasonable refinement rather than conceptual misunderstanding. In multiple cases, the model exhibits a tendency toward increased diagnostic specificity, such as refining "arthritis" to "rheumatoid arthritis." Importantly, the matrix shows very few clinically incoherent misclassifications, indicating that even when the model's top prediction does not exactly match the gold label, it consistently remains within a medically plausible diagnostic neighborhood. Below, we report our findings with a confusion matrix of gold-standard disease labels (rows) versus the model's Top-1 predicted diagnoses (columns), with color intensity proportional to the number of test cases in each cell.

To assess downstream coding behavior, we measured semantic alignment between the model's predicted ICD-10 codes and the underlying disease labels. The model produced at least one correct ICD-10 code in 198 of 212 cases, indicating near-complete procedural coverage. Across these cases, the average maximum ICD–disease semantic similarity was 0.518, and 61.6% of cases achieved meaningful alignment (cosine ≥ 0.5). These results suggest that even when exact diagnostic agreement is limited, the model's generated billing codes are often conceptually consistent with the intended disease, supporting the system's end-to-end clinical plausibility under uncertainty.

3.2 Procedure Agent Evaluation

We evaluate the Procedure Agent by the fidelity of its conversions from ICD-10-PCS codes to the equivalent range of CPT/HCPCS codes given by the Quality Health Associates of North Dakota datasheet. As described in our methodology, we treat this datasheet as a small gold standard: starting from its ICD-10-PCS prefix-to-CPT range table, we expand the CPT ranges into discrete codes and obtain 95 provider-verified ICD-10-PCS→CPT/HCPCS mappings. For each full ICD-10-PCS code in this set, the `ICD_to_CPT_Mapping` agent receives the code and its description, queries a retrieval-augmented billing-code lexicon built from the HCPCS schedule, and returns a ranked list of candidate CPT/HCPCS codes. We then compare the agent's Top-K predictions against the expanded ground-truth set and report Top-K accuracy, counting a hit whenever at least one of the K proposed codes falls within the mapped CPT range.

We compare five variants of the agent: (i) a pure one-shot baseline that guesses CPT codes directly from the ICD-10-PCS description based on data the LLM was trained on, (ii) a "plan-then-guess" variant that first verbalizes a natural-language translation before outputting codes, (iii) our retrieval-augmented system that conditions on a structured CPT/HCPCS lexicon, (iv) a variant of the retrieval-augmented system that incorporates Google Search as an additional verification step, and (v) a retrieval-augmented system that also has completely fabricated but realistic synthetic CPT codes to choose from for the mapping task (the "decoys"). Table 1 shows the results for each variant.

Table 1: Top-K ICD-10-PCS→CPT/HCPCS mapping accuracy for different Procedure Agent variants (evaluation set $n = 95$ ICD-10-PCS codes).

Strategy	Accuracy
One-Shot (Baseline)	0.02
Plan-then-Guess Top-3	0.20
Plan-then-Guess Top-10	0.22
Retrieval-Augmented Lexical	0.74
Retrieval-Augmented + Google Search	0.40
Retrieval-Augmented + Decoys	0.64

Decoy hit rate. In the retrieval-augmented + decoys condition, the agent selected at least one synthetic (invalid) CPT/HCPCS code in 66 of 238 total prediction slots, yielding a decoy hit rate of approximately 27.7%.

The one-shot baseline had absymal performance, yielding just 2% accuracy on the validation set. This shows that without an existing dataset of valid CPT/HCPCS codes to query from, the Mapping Agent either had a tendency to (1) incorrectly select valid codes from the CPT/HCPCS codes it knew of from its training data, or (2) simply hallucinate non-existent codes. Even with an additional planning step to reason through possible conversion intents, the agent could not reconcile the planning step with actual, valid code conversions. Although, even 20 − 22% accuracy is a remarkable improvement from the 2% baseline, especially given that these codes are chosen based only on what the agent may be aware of in its training data.

Of the 95 ICD-10-PCS codes derived from the Quality Health Associates datasheet, our retrieval-augmented setup achieves a top-3 accuracy of approximately 74%, indicating that access to a structured billing-code lexicon and lightweight lexical retrieval substantially improves mapping fidelity over pure pretraining-based guessing.

Notably, retrieval-augmentation with Google Search assistance significantly degraded the accuracy on the mapping validation set, falling to just 40%. The most common failure mode we observed was a form of lazy evaluation; the CPT code 15576 is a generic, all-encompassing code that can be used to categorize any surgical reconstruction procedure on the eyelids, nose, ears, lips, or intraoral areas. Because of this, even for procedures for which this all-purpose code was not valid, since it was broad enough to *plausibly* code the procedure, Google Search would simply validate the agent's guess that it was generic enough to be an appropriate code mapping. Additionally, once the Mapping Agent selected code 15576 and validated that it was appropriate, it would forgo even naming Top-K possible candidates, only suggesting one code.

Expanding the search space by adding synthetic decoy CPT/HCPCS codes also reduced performance, though less dramatically than the Google Search addition, with accuracy falling to 64%. In this decoy setting, the agent selected at least one fake code in 64 of 232 prediction slots (decoy hit rate ≈ 27.6%), suggesting that the synthetic entries were often realistic enough in format and description to fool the model. This behavior shows that in a deployment environment with access to the full CPT/HCPCS universe (and many superficially plausible but incorrect options), an effective mapping system must not only retrieve relevant candidates, but also sufficiently prune the search space and incorporate stronger verification mechanisms to avoid drifting toward spurious or non-existent codes.

The implication for the downstream Billing Agent is that it must expose the Procedure Agent to the same publicly available Medicare CPT/HCPCS fee schedule that it later uses for cost estimation, so that each predicted procedure can be mapped onto a real, billable code whose allowed amount and patient out-of-pocket cost can be consistently retrieved from government pricing data (falling back to heuristic tiered pricing only when no such entry exists). Our evaluation results show that simply asking it to come up with an estimate based on its training data is error-prone and hallucinatory.

We also collected measures of confidence that each Mapping Agent had in their suggested code conversions, so we decided to conduct an analysis to see how well-calibrated the Mapping Agent was using both retrieval-augmented lexical conversions and the decoy-augmented variant.

We evaluate agent confidence based both on (1) forecasting accuracy, measured with the Brier score [3], and (2) calibration error, measured with Expected Calibration Error (ECE) [9]. The former is a forecasting analog to the mean-squared error in one-dimension, given by:

$$\mathrm{BS} \;=\; \frac{1}{N}\sum_{i=1}^{N}\left(p_i - y_i\right)^2,$$

where p_i is the predicted probability (confidence in our setting) and $y_i \in \{0,1\}$ is the realized outcome for event i (whether the code was a valid CPT/HCPCS code for the given ICD-10-PCS). The latter is a measure of the average discrepancy between predicted probabilities and empirical event frequencies across groups of similarly confident predictions.

We report metrics for three different Mapping Agent regimes: (1) the vanilla Retrieval-Augmented lexical converter; (2) the decoy-augmented converter; and (3) confidence on just the decoys in (2). The rationale for (3) is that our Mapping Agent assigning low confidence scores to its suggested decoy codes means that even when it selects a code incorrectly, it does so with high uncertainty.

Table 2: Brier score and Expected Calibration Error (ECE) for three ICD-10-PCS→CPT Mapping Agent regimes.

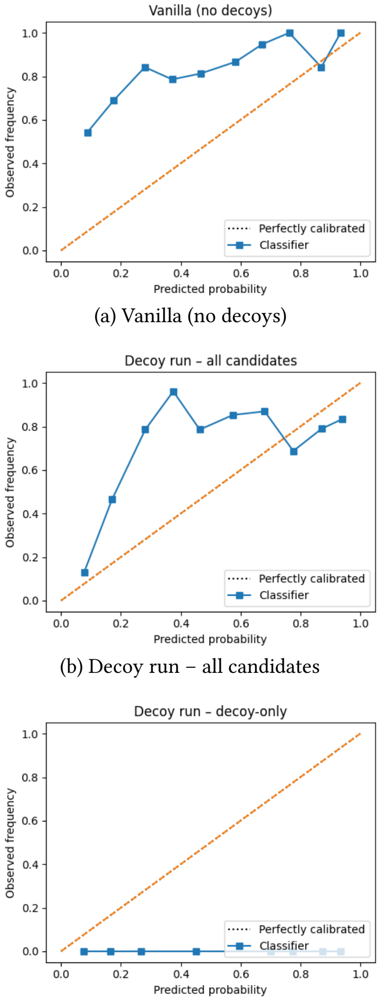
Regime	# Candidates	Brier Score	ECE
Vanilla (no decoys)	225	0.3121	0.3626
Decoys (all candidates)	302	0.2359	0.2439
Decoys (decoys only)	83	0.0761	0.1747

Based on the table and calibration curves, the vanilla retrieval-augmented lexical converter – despite having the highest *accuracy* on a Top-K basis – was also the most systematically underconfident (BS = 0.3121, ECE = 0.3626). In fact, just under 50% (112/225) of its confidence scores for suggested CPT/HCPCS conversions were in the bottom two deciles (i.e., below 30% confidence), despite the fact that among the third decile of confidence scores with a mean confidence of just 28%, the converter suggested a correct CPT/HCPCS conversion 84.2% of the time. Despite also being systematically underconfident like its vanilla no-decoy counterpart, the Mapping Agent run that used decoy CPT/HCPCS codes was better calibrated for conversions it suggested with high-confidence (BS = 0.2359, ECE = 0.2439).

While the calibration curve for the Mapping Agent's confidence on suggested decoys may look deceivingly bad, the agent was actually the most calibrated whenever it suggested a decoy CPT/HCPCS code. Out of 83 suggested candidates,[3] 75 of them (90.4%) were in the bottom three deciles of confidence, with 34 (41%) of those being in the bottom decile alone.

This was a positive sign that even when the Mapping Agent was suggesting decoy codes, it did so with the correct amount of uncertainty, giving it very strong forecasting and calibration scores (BS = 0.0761, ECE = 0.1747).

[3] This count differs from the 66/238 decoy hit rate reported above because the latter only considers decoy codes that appear in the top-K predictions per ICD-10-PCS code (up to three per ICD), whereas the total of 83 here includes *all* suggested decoy candidates across ranks, including those that fell below the top-K cutoff.



(a) Vanilla (no decoys)

(b) Decoy run – all candidates

(c) Decoy run – decoy-only

Figure 2: Calibration curves for Three Procedure Agent Regimes' Confidence in ICD-10-PCS→CPT Mappings.

We compared this distribution to the distribution of confidence scores in the vanilla regime and the decoys with all candidates regime, which had confidence scores much more evenly distributed (see figures 4, 5). On average, the Mapping Agent had a confidence score of 17.5% when it suggested a decoy mapping, compared to an average confidence of 42.8% on the full real codes + decoy dataset and an average confidence of 42.4% on the vanilla no decoys dataset.

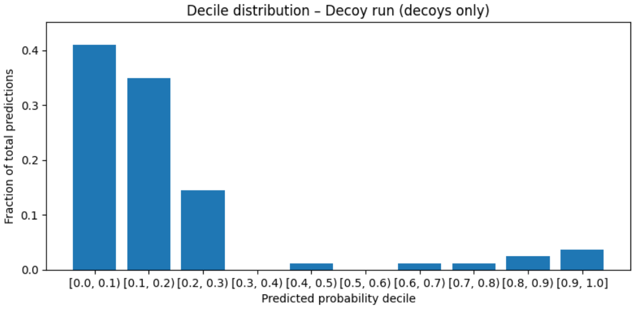


Figure 3: Distribution of suggested decoy candidates by predicted confidence decile for the Retrieval-Augmented + Decoys regime.

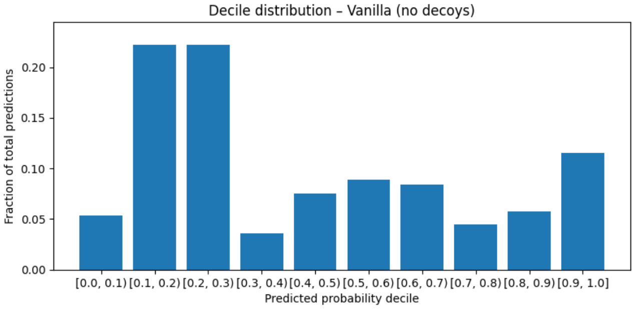


Figure 4: Distribution of all predicted confidence scores by decile for the Vanilla regime.

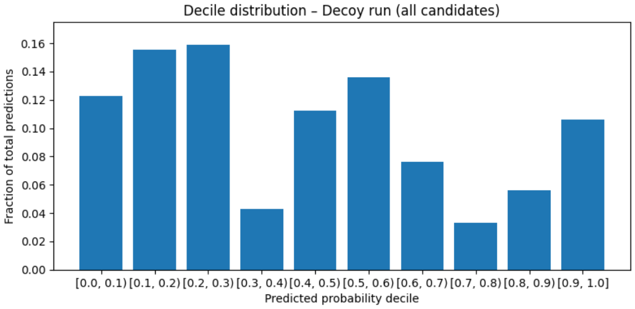


Figure 5: Distribution of all predicted confidence scores by decile for the Retrieval-Augmented + Decoys regime.

Taken together, these results suggest that our Mapping Agent is generally cautious rather than overconfident. The vanilla retrieval-augmented converter achieves strong top-K accuracy but tends to assign probabilities that are too low across the board, especially in regions where it is actually very accurate. Introducing decoys slightly degrades top-K mapping accuracy but improves calibration at higher confidence levels. The decoy-only analysis shows that when the agent does hallucinate a fake code, it typically does so with very low confidence. In other words, the main failure mode is underconfidence in correct mappings, not unjustified certainty in incorrect or synthetic ones.

4 Conclusion

Our findings demonstrate that a modular multi-agent architecture can meaningfully model the end-to-end administrative and clinical reasoning pipeline that transforms a patient's free-text symptoms into diagnoses, procedures, and estimated financial charges. By allowing uncertainty to propagate across each agent, the system models how diagnostic ambiguity, coding variability, and billing complexities combine to shape a patient's financial experience. We observe that retrieval-augmented and structured approaches substantially outperform naive LLM inference, particularly in mapping ICD-10-PCS procedures to billable CPT/HCPCS codes. At the same time, evaluation results reveal persistent calibration challenges, error modes, and sensitivity to the search space. This emphasizes that even advanced agentic systems remain vulnerable to hallucination and overgeneralization without well-defined constraints.

Altogether, these results suggest that while LLM driven agents show promise for improving transparency and predictability in the clinical billing processes, substantial work remains before such systems can support real-world, patient facing systems. Future work should incorporate richer domain knowledge, more robust verification, and more accurate integration with payer-specific billing logic. By modeling the pathways through which patients move across the diagnosis, procedures, and billing stages, the framework we present here is a step toward building AI systems that are not only clinically aligned but also reflective of the informational and financial realities patients face.

References

[1] AAPC. 2025. AAPC: The Business of Healthcare. https://www.aapc.com/. Accessed: 2025-12-08.

[2] American Medical Association. 2024. *Current Procedural Terminology (CPT®) 2025 Professional Edition*. American Medical Association, Chicago, IL.

[3] Glenn W. Brier. 1950. Verification of Forecasts Expressed in Terms of Probability. *Monthly Weather Review* 78, 1 (1950), 1–3.

[4] Centers for Medicare & Medicaid Services. [n. d.]. What Part B Covers. https://www.medicare.gov/providers-services/original-medicare/part-b.

[5] Centers for Medicare & Medicaid Services. 2025. Clinical Laboratory Fee Schedule: CY 2025 Q4 Release. https://www.cms.gov/medicare/payment/fee-schedules/clinical-laboratory-fee-schedule-clfs/files/25clabq4.

[6] Centers for Medicare & Medicaid Services. 2025. *Healthcare Common Procedure Coding System (HCPCS) Level II Codebook*. Centers for Medicare & Medicaid Services, Baltimore, MD.

[7] Centers for Medicare & Medicaid Services 2025. *International Classification of Diseases, Tenth Revision, Procedure Coding System (ICD-10-PCS)*. Centers for Medicare & Medicaid Services, Baltimore, MD.

[8] Centers for Medicare & Medicaid Services. 2025. PFS National Payment Amount File for Jan 2026. https://www.cms.gov/medicare/payment/fee-schedules/physician/national-payment-amount-file/pfrev26a.

[9] Morris H. DeGroot and Stephen E. Fienberg. 1983. The Comparison and Evaluation of Forecasters. *Journal of the Royal Statistical Society: Series D (The Statistician)* 32, 1-2 (1983), 12–22.

[10] Zhen Hou, Hao Liu, Jiang Bian, Xing He, and Yan Zhuang. 2025. Enhancing medical coding efficiency through domain-specific fine-tuned large language models. *npj Health Systems* 2 (2025), 14. doi:10.1038/s44401-025-00018-3

[11] Keith Kwan. 2024. Large language models are good medical coders, if provided with tools. arXiv:2407.12849 [cs.IR] https://arxiv.org/abs/2407.12849

[12] Subhankar Maity and Manob Jyoti Saikia. 2025. Large Language Models in Healthcare and Medical Applications: A Review. *Bioengineering* 12, 6 (2025), 631. doi:10.3390/bioengineering12060631

[13] Sumayah Rahman, Siyu Yang, and Ethan Cha. 2024. Large Language Models for Personalized Healthcare Billing Support. *Cedar* blog. https://www.cedar.com/blog/large-language-models-for-personalized-healthcare-billing-support/

[14] John C. Rollman, Bruce Rogers, Hamed Zaribafzadeh, Daniel Buckland, Ursula Rogers, Jennifer Gagnon, Ozanan Meireles, Lindsay Jennings, Jim Bennett, Jennifer Nicholson, Nandan Lad, Linda Cendales, Andreas Seas, Alessandro Martinino, E. Shelley Hwang, and Allan D. Kirk. 2025. Practical Design and Benchmarking of Generative AI Applications for Surgical Billing and Coding. arXiv:2501.05479 [cs.CL] https://arxiv.org/abs/2501.05479

[15] Ali Soroush, Benjamin S. Glicksberg, Eyal Zimlichman, Yiftach Barash, Robert Freeman, Alexander W. Charney, Girish N Nadkarni, and Eyal Klang. 2024. Large Language Models Are Poor Medical Coders — Benchmarking of Medical Code Querying. *NEJM AI* 1, 5 (2024), AIdbp2300040. arXiv:https://ai.nejm.org/doi/pdf/10.1056/AIdbp2300040 doi:10.1056/AIdbp2300040

[16] Zhangdie Yuan, Han-Chin Shing, Mitch Strong, and Chaitanya Shivade. 2025. Toward Reliable Clinical Coding with Language Models: Verification and Lightweight Adaptation. arXiv:2510.07629 [cs.CL] https://arxiv.org/abs/2510.07629