Unearth-AI: A Proof-of-concept System for Human-in-the-loop Prediction Task Extraction from Natural Language for Health AI

Anonymous Authors[1]

Abstract

Defining and extracting clinical prediction tasks from electronic health record (EHR) data is technically demanding and time-consuming, requiring substantial domain expertise and dataset-specific tooling. We present Unearth-AI, a proof-of-concept interactive, agentic large language model (LLM) system that translates natural language task descriptions into formal task specifications in the ACES framework (Xu et al., 2025), combining LLM generation with automated validation, feasibility detection, and optional human-in-the-loop refinement while requiring only dataset schema information rather than patient-level data. In a feasibility study spanning 14 tasks over the INSPIRE perioperative EHR dataset, Unearth-AI generated specifications judged consistent with the provided free-text inputs for all 8 feasible tasks and correctly identified 4 of 6 infeasible tasks. For feasible tasks, end-to-end generation took under 4 minutes and cost under $0.25 on average. These results suggest the practical feasibility of an interactive LLM-mediated task extraction workflow and motivate future work on systematic evaluation and user-centered tooling for clinical AI development.

1. Introduction

Extracting labeled cohorts for clinical prediction tasks from electronic health record (EHR) data is a core but time-consuming step in health AI research, and discrepancies in cohort definitions contribute to irreproducibility (Johnson et al., 2017; McDermott et al., 2021; McDermott, 2025). Standardized task languages such as (Xu et al., 2025), paired with common data models such as MEDS (Arnrich et al., 2024), lower the barrier to defining transparent, reproducible cohorts, but dataset-to-dataset variability and lack of familiarity with programming can limit use among health researchers.

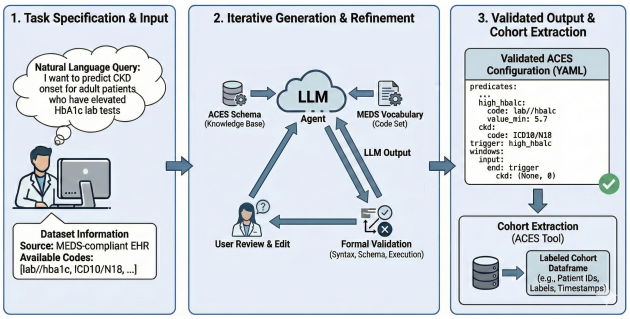

Figure 1. Overview of Unearth-AI: from natural language criteria and dataset schema to a validated ACES configuration via iterative generation, validation, and optional user refinement. Figure created with Nano Banana (`https://deepmind.google/models/gemini-image/`).

We present UNified EHR Aggregation, Retrieval, Transformation, and Harmonization (Unearth-AI), a proof-of-concept interactive, agentic large language model (LLM) system that translates a natural language task description into an executable ACES configuration using only dataset schema information (available codes) rather than patient-level data. Importantly, MEDS and ACES enable the easy extraction of cohorts under temporal constraints (windows). Unearth-AI combines LLM generation with automated validation, feasibility detection, and optional human-in-the-loop refinement.

In a feasibility study of 14 tasks over the INSPIRE perioperative dataset (Lim et al., 2024), Unearth-AI generated configurations judged consistent with user intent for all 8 feasible tasks and correctly identified 4 of 6 infeasible tasks. For feasible tasks, end-to-end generation took under 4 minutes and cost under $0.25 on average.

2. Method

Unearth-AI maps a natural language task description to an ACES configuration through an iterative loop. At each iteration, an LLM proposes either (i) a candidate YAML

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

configuration or (ii) an infeasibility assessment, conditioned on (a) the user-provided criteria and (b) non-sensitive dataset metadata (the set of available MEDS codes). Prompt is then passed to the OpenAI API utilizing the GPT-5 model configured with a fixed temperature of 1.0 and a maximum limit of 12,000 completion tokens per response, and the LLM is instructed to return either an assessment of the overall feasibility of the task via a structured JSON element or a YAML configuration file conforming to the ACES standard.We describe the full prompt construction in Appendix A.

Automated Validation and Feedback Each candidate configuration is checked deterministically, and any error message is fed back to the model for revision. We apply three validations:

1. **Syntax validation** (ACES parser)
2. **Predicate verification** (referenced codes exist)
3. **Execution testing** (run on a single shard)

This loop terminates when validation succeeds or the system concludes the task is infeasible under the available data and ACES constraints.

Human-in-the-loop Refinement Users can review intermediate outputs, provide clarifications, and request changes, positioning Unearth-AI as an assistive authoring tool.

Privacy Unearth-AI does not require patient-level data: generation can be grounded in schema information (available codes) and optional aggregate statistics. Practitioners should still treat aggregates with caution in small cohorts.

3. Feasibility Study

We evaluate Unearth-AI on 14 clinical prediction tasks over the INSPIRE perioperative dataset (Lim et al., 2024). Tasks span diverse targets and include both feasible and infeasible specifications; the full task list and outcomes are provided in Table 2.

Metrics We measure (i) success on feasible tasks, (ii) correctness of infeasibility detection, and (iii) efficiency (time, iterations, and token cost).

4. Results

Unearth-AI successfully generated valid configurations for all feasible tasks (8/8) and correctly flagged 4/6 infeasible tasks. The two failure cases corresponded to complex, underspecified endpoints that the system attempted to encode rather than rejecting.

Table 1 summarizes average efficiency across categories.

Table 1. Average performance metrics for Unearth-AI. Time is reported in minutes and cost in dollars, calculated at the appropriate API cost of $1.25/1M input tokens and $10/1M output tokens.

Category	N	Time	Cost	Steps
Feasible	8	3.76	0.24	2
& Correct	8	3.76	0.24	2
Infeasible	6	8.58	0.53	3.67
& Correct	4	1.20	0.11	1

```yaml
predicates:
  icu_admission:
    code: ICU_ADMISSION
  icu_discharge:
    code: ICU_DISCHARGE
  icu_out:
    code: ICU_OUT
  hospital_discharge:
    code: HOSPITAL_DISCHARGE
  death:
    code: MEDS_DEATH
  exit_event:
    expr: "or(icu_discharge, icu_out, hospital_discharge,
death)"
trigger: icu_admission

windows:
  input:
    start: trigger
    end: start + 24h
    start_inclusive: True
    end_inclusive: True
    has:
      death: "(None, 0)"
      hospital_discharge: "(None, 0)"
      icu_out: "(None, 0)"
    index_timestamp: end
  target:
    start: trigger
    end: start + 3 days
    start_inclusive: False
    end_inclusive: True
    label: exit_event
```

Figure 2. Example ACES configuration file generated by Unearth-AI for a clinical prediction task.

Efficiency For feasible tasks, generation completes in under 4 minutes and under $0.25 on average. Distributions of iterations and wall-clock time are shown in Figure 3.

Task Outcomes Table 2 lists all tasks evaluated in the feasibility study and their outcomes.

Example Generated Configuration An example generated configuration is shown in Figure 2.

Efficiency Distributions Efficiency distributions are shown in Figure 3.

Table 2. Task Evaluation Results Across Predefined and Paper-Derived Tasks

Source	Task Description	Feasibility	Result	Iter.
Predefined	Flu diagnosis after annual physical[a]	Infeasible	Pass	1
Predefined	In-hospital mortality (48h input, 24h gap)	Feasible	Pass	3
Predefined	30-day hospital readmission	Feasible	Pass	2
Predefined	Re-ventilation within 24 hours (ICU)	Feasible	Pass	1
Predefined	ICU length of stay >3 days	Feasible	Pass	2
Predefined	Sepsis development after 24h (ICU)	Feasible	Pass	2
Paper-derived	MACCE 90d post non-cardiac surgery	Feasible	Pass	1
Paper-derived	MACCE 30d post surgery (SHR)	Feasible	Pass	2
Paper-derived	Postop renal comp. (abdominal surgery)	Feasible	Pass	2
Paper-derived	In-hospital mortality (valvular)[b]	Infeasible	Pass	1
Paper-derived	AKI risk (creatinine delta)[c]	Infeasible	Pass	1
Paper-derived	Postop comp. (intestinal obstruction)[d]	Infeasible	Pass	1
Paper-derived	$ETCO_2$ and cardiac surgery mortality[e]	Infeasible	Fail	8
Paper-derived	MELD score in valve surgery[f]	Infeasible	Fail	10

[a]Correctly identified as infeasible: INSPIRE contains ICU/hospital data only; annual physical visits unavailable.
[b]Multiple concurrent outcomes (7/28-day mortality, ECMO/IABP/CRRT) not encodable in single ACES config.
[c]ACES cannot compute cross-event deltas (baseline vs. follow-up creatinine).
[d]POSSUM score unavailable; cannot link surgery to intestinal obstruction indication.
[e]Task infeasible due to complex cohort criteria; system generated config instead of identifying infeasibility.
[f]Task required multiple outcomes; system generated config instead of identifying infeasibility.

5. Discussion

These results suggest that an interactive LLM-mediated workflow can make standardized task authoring more accessible while retaining the auditability of a structured intermediate representation (ACES). Remaining failures highlight the need for better feasibility detection for composite endpoints and missing capabilities (e.g., cross-event computations).

6. Conclusion

Unearth-AI demonstrates the practical feasibility of translating natural language task descriptions into executable ACES configurations with automated validation and low per-task cost.

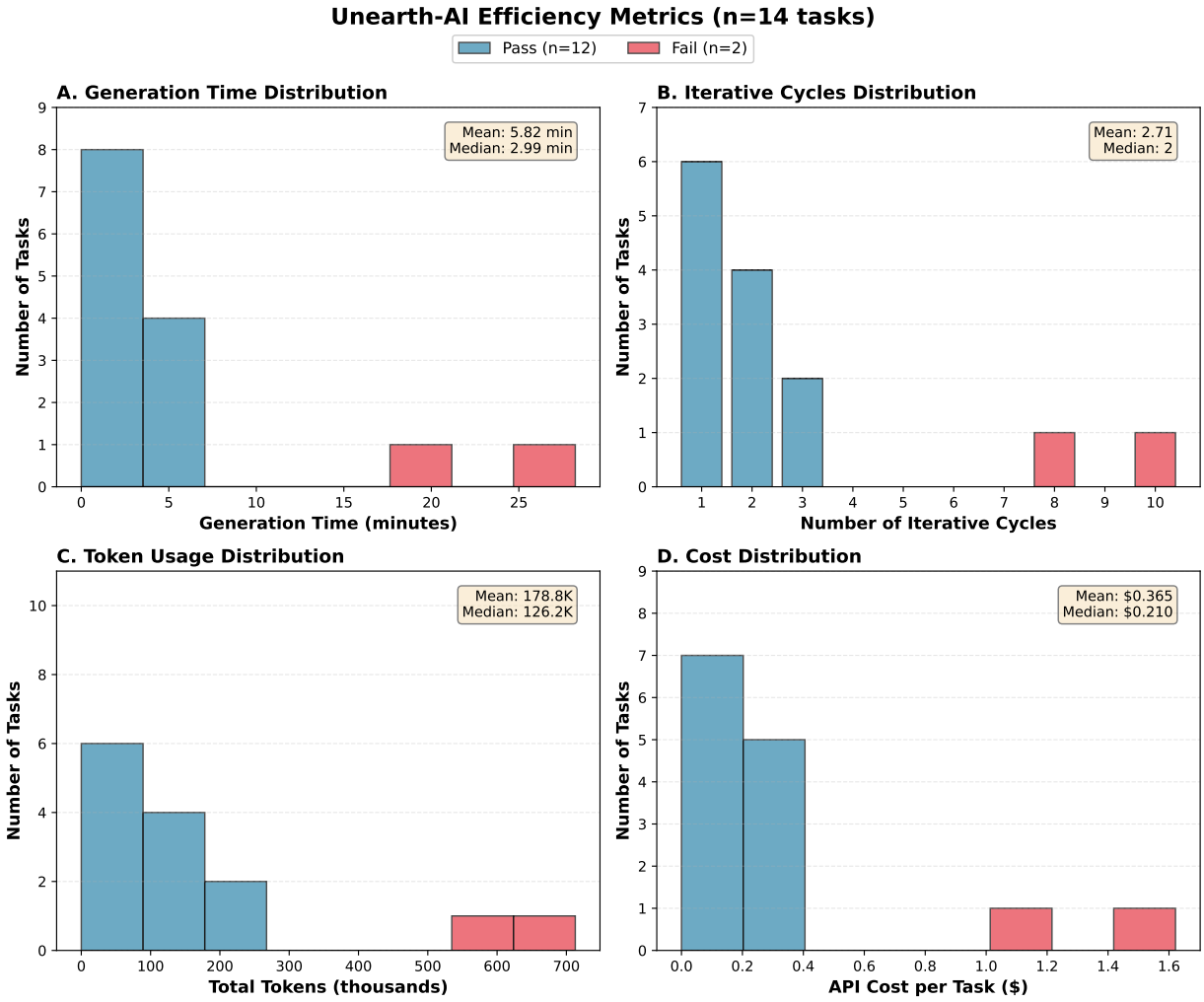

Figure 3. Efficiency metrics for Unearth-AI across 14 tasks. Blue bars indicate correctly processed tasks (n=12), while red bars show incorrectly classified tasks (n=2).

References

Arnrich, B., Choi, E., Fries, J. A., McDermott, M. B., Oh, J., Pollard, T., Shah, N., Steinberg, E., Wornow, M., and van de Water, R. Medical event data standard (MEDS): Facilitating machine learning for health. In *ICLR 2024 Workshop on Learning from Time Series For Health*, 2024. URL https://openreview.net/forum?id=IsHy2ebjIG.

Johnson, A. E. W., Pollard, T. J., and Mark, R. G. Reproducibility in critical care: a mortality prediction case study. In Doshi-Velez, F., Fackler, J., Kale, D., Ranganath, R., Wallace, B., and Wiens, J. (eds.), *Proceedings of the 2nd Machine Learning for Healthcare Conference*, volume 68 of *Proceedings of Machine Learning Research*, pp. 361–376. PMLR, 18–19 Aug 2017. URL https://proceedings.mlr.press/v68/johnson17a.html.

Lim, L., Lee, H., Jung, C.-W., Sim, D., Borrat, X., Pollard, T. J., Celi, L. A., Mark, R. G., Vistisen, S. T., and Lee, H.-C. Inspire, a publicly available research dataset for perioperative medicine. *Scientific Data*, 11(1):655, 2024.

McDermott, M. The (lack of?) science of machine learning for healthcare. In Hegselmann, S., Zhou, H., Healey, E., Chang, T., Ellington, C., Mhasawade, V., Tonekaboni, S., Argaw, P., and Zhang, H. (eds.), *Proceedings of the 4th Machine Learning for Health Symposium*, volume 259 of *Proceedings of Machine Learning Research*, pp. 19–29. PMLR, 15–16 Dec 2025. URL https://proceedings.mlr.press/v259/mcdermott25a.html.

McDermott, M. B., Wang, S., Marinsek, N., Ranganath, R., Foschini, L., and Ghassemi, M. Reproducibility in machine learning for health research: Still a ways to go. *Science translational medicine*, 13(586):eabb1655, 2021.

Xu, J., Gallifant, J., JOHNSON, A., and McDermott, M. B. ACES: Automatic cohort extraction system for event-stream datasets. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=P4XmKjXTrM.

A. Full Prompt Construction

This section provides the complete prompt construction process used by Unearth-AI to generate ACES configuration files.

A.1. System Message

The system message provides foundational context and constraints to the LLM. It includes:

MEDS dataset note: No explicit age variable exists. Age must be calculated from `MEDS_BIRTH` events.

ACES age constraint instruction: The system cannot directly compute continuous age values (e.g., `Trigger_Date - Birth_Date`). Age must be enforced as a windowed constraint using inclusion by confirmation: define a window from `NULL` to `trigger - 6574 days` and require `has: birth_event: (1, None)` so only verified adults are included. The instruction prefers day-based offsets (e.g., 6574 days for 18 years).

ACES window linkage rule: Exactly one boundary must reference the other. Use `end: trigger - 6574 days` with `start: NULL`, or use `end: trigger` with `start: end - 6574 days` for explicit pre-trigger windows.

ACES Technical Spec: The complete ACES technical specification is loaded from a configurable file path and included in the system message.

A.2. User Prompt Template

The main user prompt begins with: "Given the following dataset schema and cohort inclusion criteria, provide a full configuration file based on the provided style guide. Generate predicates, trigger, and windows in a format that is directly usable."

The prompt then includes the following sections:

Dataset Schema: A list of all unique medical codes available in the MEDS dataset, extracted from schema files or data shards. May be truncated to the first 1,000 codes for token management.

Dataset Context (optional): Summary statistics including available care settings (ICU, ED, etc.), population demographics (age, sex distributions), common measurement frequencies, and dataset size. May be truncated to the first 120 lines.

Inclusion Criteria: The natural language task description provided by the user.

Feasibility Assessment: Instructions to first evaluate whether the task is possible given the dataset context and schema. If impossible, the LLM should respond with a JSON object containing `feasibility: "IMPOSSIBLE"`, a `reasoning` field explaining why, and a `missing_capability` field describing what is lacking. If feasible, return only the YAML configuration.

Style Guide: Structural guidance for YAML format, either provided directly or generated from example configurations. Includes how predicates, triggers, and windows are defined and linked, common logic expressions, and general formatting conventions.

Basis Configuration File (optional): An existing ACES configuration to modify or build upon for iterative refinement.

Example Configurations (optional): One or more complete example ACES configurations demonstrating common patterns like hospital mortality, readmission, and length of stay.

Partial Configuration (optional): An incomplete configuration to complete.

User Feedback (optional): Accumulated feedback from previous iterations, such as validation errors to fix, semantic corrections needed, or cohort size adjustments requested.

A.3. Schema Generation

The dataset schema is extracted via one of three methods:

1. **Schema file**: Read from a pre-generated text file listing all codes
2. **MEDS shard**: Parse a MEDS Parquet shard and extract unique codes from the `code` column
3. **Codes file**: Read from a dedicated codes Parquet file containing the full code vocabulary

The schema is formatted as a simple list of medical codes (e.g., `HOSPITAL_ADMISSION`, `LAB//mg/dL//glucose`, `DIAGNOSIS//ICD//10//I21`). When the schema exceeds token limits, it is automatically truncated to the first 1,000 entries while retaining the most commonly used codes.

A.4. Dataset Context Generation

Dataset context is optionally generated by analyzing MEDS data to produce summary statistics. A typical context includes:

Dataset Overview: Total patients, total events, and date range.

Available Care Settings: Counts of ICU admissions, emergency department visits, and hospital admissions.

Demographics: Age distribution, sex distribution, and other population characteristics.

Common Measurements: Top measurements by frequency (e.g., `VITAL//mmHg//sbp`, `LAB//mg/dL//glucose`).

This context helps the LLM assess task feasibility before attempting configuration generation, but because summary statistics may carry re-identification risk in small or specialized cohorts, this module is optional and should be used at the practitioner's discretion.

A.5. Style Guide Generation

When example configurations are provided, Unearth-AI can automatically generate a style guide by prompting the LLM to analyze the examples and extract structural patterns. The prompt asks the LLM to summarize the structure, style, and common patterns of multiple example configuration files into a concise style guide covering: (1) how predicates are defined and their common fields, (2) how triggers are defined, (3) how windows are defined and linked, (4) common logic expressions, and (5) general formatting conventions. The prompt instructs the LLM not to repeat examples verbatim, but to produce a short set of rules and guidelines.

Generated style guides are cached to avoid regeneration on subsequent runs.

A.6. Iterative Refinement

When validation errors occur or user feedback is provided, the prompt is reconstructed with the feedback history appended. This enables the LLM to learn from mistakes and converge toward a correct configuration over multiple iterations. Our results show that most tasks converge within 1–3 iterations (median: 2), indicating effective prompt design and LLM comprehension of ACES syntax.