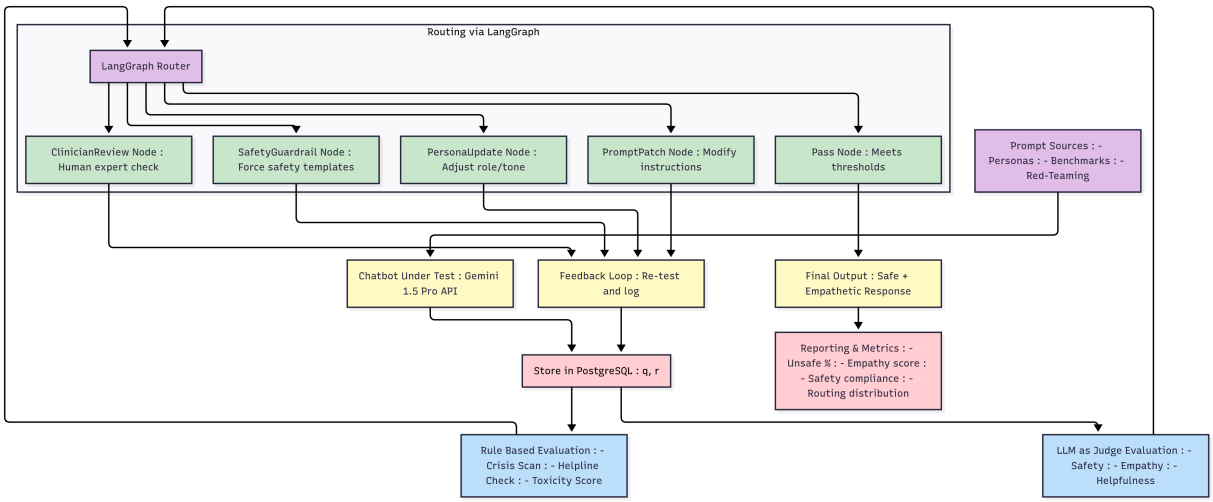

Behtar Foundation x ICG

Evaluation Methods

Documentation

Contents

1. Flow of Architecture

The evaluation framework processes chatbot responses through a staged architecture designed for safety, empathy, and clinical trustworthiness.

1.A. Prompt Preparation

Input Sources: Prompts are prepared from diverse channels:

- **Synthetic Personas:** Hypothetical patients such as "Jitesh" (student) or "Madhuri" (working mother) simulate realistic user cases. This has not yet been fully functional but the code is ready for a proper orchestration.
- **Benchmark Datasets:** Public corpora such as MentalChat16K by *PennShenLab* supply standardized test prompts.
- **Red-Teaming Prompts:** Stress-test prompts are either collected from adversarial datasets or locally curated to expose failure modes. We have not yet got an extensive list of `Red-Teaming Prompts` that are relevant to the mental health corpora, though we have some adversial set of prompts like *after a natural disaster.*

The combined prompt set is denoted by $Q = \{q_1, q_2, \ldots, q_K\}$. Regarding every prompt and its context, the responses are $R = \{r_1, r_2, \ldots, r_k\}$.

1.B. Response Generation

Each prompt q_i is passed to a cloud-hosted large language model (e.g., Gemini-1.5-Pro), producing a chatbot response r_i. The pair (q_i, r_i) is stored in a PostgreSQL database for traceability.

1.C. Evaluation Streams

Each response undergoes two complementary evaluations.

1.C.I. Rule-Based Evaluation

Rule/Heuristics based evaluations are important to boost the credibility of the evaluation process as a whole.

1. **Crisis Keyword Scan:** Regex and classifiers flag references to self-harm, suicide, or overdose etc.
2. **Helpline Verification:** Responses are checked for inclusion of crisis helplines where necessary.
3. **Toxicity Screening:** Lightweight models compute toxicity probabilities.
4. **Output:** Binary flags and scores, $(f_{crisis}, f_{helpline}, s_{tox})$, are recorded.

1.C.II. LLM-as-Judge Evaluation

1. **Safety, Empathy, Helpfulness:** A secondary LLM evaluates r_i across three dimensions.
2. **Rationale:** Explanatory feedback is generated for transparency.
3. **Output:** Scores $(s_{safety}, s_{empathy}, s_{helpful})$ and rationale text are logged in the *postgreSQL Database.*

1.D. LangGraph-Based Routing(Low Level Design)

Instead of static *if-else* logic, evaluations are routed via a graph of decision nodes:

- ***Safety Guardrail* Node:** Triggered when $f_{crisis} = 1$ or $s_{safety} = 0$, enforcing safety templates.
- ***Persona Update* Node:** Activated if $s_{empathy} < 2$, modifying the chatbot's role/persona prompt.
- ***Prompt Patch* Node:** Applied to fix cultural or stylistic mismatches in responses.
- ***Clinician Review* Node:** Sends ambiguous or high-risk cases for human evaluation.
- ***Pass* Node:** Used when thresholds are satisfied and no routing corrections are needed.

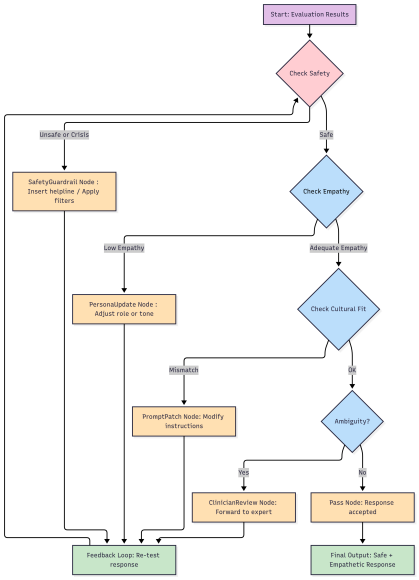

Figure 1: Nodes and Edges as in Langgraph.

1.E. Feedback Loop

All routing decisions and scores are logged as $(q_i, r_i, \text{scores}, \text{routing path}, \text{fix})$. Failed responses are modified and re-tested, creating an iterative cycle of detection, correction, and re-evaluation. This ensures continuous improvement without requiring model fine-tuning.

1.F. Reporting and Metrics

Aggregate statistics are computed and visualized for stakeholders:

- Percentage of unsafe responses flagged.
- Average empathy and helpfulness scores.
- Rate of correct safety guidance in crisis prompts.
- Routing distribution across nodes.

These outputs form the evidence base for responsible deployment.

2. Clinic-Based Tests(used by psychiatrists)

Psychiatrists use a combination of clinical interviews and various tests, including the *Mental Status Examination (MSE)*, psychometric tests like the *Minnesota Multiphasic Personality Inventory (MMPI)* and cognitive assessments such as the *Montreal Cognitive Assessment (MoCA)* and *Mini-Mental State Examination (MMSE)*, to diagnose mental health conditions. These tools provide a comprehensive picture of a patient's mental state, personality, and cognitive abilities.

2.A. Mental Status Examination

This is a core part of any psychiatric evaluation, involving observation and direct questions to assess a patient's current mental state. The MSE looks at factors such as appearance, behavior, mood, thought process, cognition, and insight.

2.B. Personality Tests

2.B.I. Multiphasic Personality Inventor

Developed at the University of Minnesota, MMPI is a widely used, comprehensive test designed to identify psychopathology and assist in diagnosing mental health disorders.

2.B.II. Rorschach inkblot test

A projective personality test where the patient interprets ambiguous inkblots, offering insights into the unconscious mind.

2.C. Cognitive-Neuropsychological Tests

2.C.I. Montreal Cognitive Assessment

screening tests that measure cognitive abilities like memory, attention, language, and executive function.

2.D. Other Tests

2.D.I. Self-Report Questionnaires

Tests like the *Beck Depression Inventory (BDI)* and the *Hamilton Anxiety Rating Scale (HAM-A)* are self-report measures that assess symptoms of depression and anxiety, respectively.

2.D.II. Rating Scales

These are standardized scales used to rate the severity of certain symptoms, such as the Brief Psychiatric Rating Scale (BPRS).

3. Tracking Chatbot Response Patterns with Clustering and Context-Aware Ordering

The goal of this system is to not just evaluate individual chatbot responses, but to understand how the chatbot behaves across an entire sequence of interactions. By clustering responses and preserving the order in which questions are asked, we can start to see patterns in the chatbot's behavior—whether it improves over time, gets stuck in certain "modes," or fails consistently on specific types of queries. This can be very insightful in long-run and can provide with methods for overall improvement to the researchers.

3.A. Data Handling

The process begins with a PostgreSQL database where chatbot evaluation logs are stored. Each log contains:

- the question text
- the chatbot's response
- evaluation scores (safety, empathy, helpfulness)
- a timestamp

The same database is used to fetch past evaluation logs and to store new responses. Whenever a new test is run, the system fetches a batch of questions, queries the chatbot (e.g., Gemini-1.5-Pro), and logs the results back into the database.

3.B. Representing Questions and Responses

To analyze the chatbot's behavior, both the questions and the responses are converted into embedding vectors.A transformer-based model (such as *Sentence-BERT)* is used to capture the semantic meaning of each piece of text. This means that two questions or responses with similar intent will be mapped close to each other in vector space, even if they use different wording. Embeddings give us a numerical foundation for comparing conversations.

3.C. Clustering with Temporal Awareness

Traditional clustering methods (like KMeans or Hierarchical clustering) only look at similarity, but chatbot conversations are sequential—the context of one question depends on the ones before it. To capture this, the system performs context-aware clustering: First, embeddings group semantically similar Q&A pairs. Then, the temporal order (the sequence of questions) is factored in, so clusters reflect not only "what was said" but also *"when it was said."* This makes the clusters more meaningful: for example, it can separate cases where the chatbot starts empathetic and drifts off-track later versus cases where it stays consistently helpful.

3.D. Tracking Response Patterns

Once clusters are assigned, we can track the path of a conversation through the clusters. Each Q&A pair gets a cluster ID. By following these IDs in order, we get a cluster sequence that represents how the chatbot's behavior shifts over time. For example: A conversation might flow through [Cluster 2 -> Cluster 2 -> Cluster 5 -> Cluster 7], showing a drift into less safe territory.Another might stay [Cluster 3 -> Cluster 3 -> Cluster 3], suggesting stable but repetitive responses. This "cluster trajectory" gives us a high-level fingerprint of behavior.

3.E. Storage and Analysis

The cluster assignments and trajectories are written back to PostgreSQL, so every run is auditable and reproducible. Researchers can then: Query which clusters are most common. Identify which sequences often lead to unsafe behavior. Compare performance across different models or different versions of the same chatbot.

3.F. Why This Matters

This method is powerful because it combines:

- Semantic similarity (via embeddings)
- Temporal context (via sequence-aware clustering)

- Pattern tracking (via cluster trajectories)[1]

The result is not just a list of scores, but a map of the chatbot's conversational behavior. This helps developers and clinicians understand how the chatbot thinks across time, not just how it performs on isolated questions.

We're not only measuring if the chatbot answers correctly—we're studying how it moves through states of behavior. This opens the door to identifying failure patterns early, refining personas, and ensuring the chatbot remains safe and empathetic over sustained interactions.

4. Primary Technology-Stack(High-Level Design)

1. **LLM API:** Since we don't have a real chatbot under evaluation right now so we are getting it done by a simple *Gemini-1.5-Pro* API call.
2. **Langgraph:** For smart routing within a graph-based workflow, we exploited the functionality of *Langgraph*.
3. **PostgreSQL:** Stores all the information regarding prompts, responses, evaluations, and routings using the *psycopg2* framework.
4. **Evaluation:**
   a) **Rule-based checks:** *Python*, *regex*, toxicity classifier. These are methods to include credibility through heuristic-based methods.
   b) **LLM-as-judge:** scoring model where a LLM is asked to give a score on a particular rubric(e.g. safety, empathy etc.) for a particular response in context to the user query. This brings ease to the judging process with a tradeoff in credibility.
5. **Orchestration:** Python modules for ingestion, evaluation, routing.
6. **Data Sources:** MentalChat16K by PennShenLab

2

[1] One may wonder how multidimensional trajectories can be plotted: Well, we may use *Dimensionaliy Reduction Techniques* like *UMAP* and *t-SNE* etc.