Reinforcement Learning for Hallucination Prevention

Agaaz Singhal, Krishnav Mahansaria, Mudit Surana
CSAI & Plaksha University, India
CSAI & Plaksha University, India
CSAI & Plaksha University, India
agaaz.singhal@plaksha.edu.in; krishnav.mahansaria@plaksha.edu.in; mudit.surana@plaksha.edu.in

Abstract*— In high-stakes domains such as medical question answering, large language models (LLMs) often "hallucinate," producing confident yet incorrect or misleading responses. Relying on a single model risks patient safety and undermines trust. In this work, we cast the problem of selecting among multiple LLMs—each with different strengths across medical topics—as a contextual-bandit-based recommender system. This lightweight reinforcement-learning framework balances exploration and exploitation on a per-question basis. For each incoming query, we encode its text and topic into a fixed-length context vector; we then compute an upper-confidence-bound (LinUCB) score for each candidate model, combining a predicted accuracy estimate with an uncertainty bonus that encourages testing lesser-tried models. The model with the highest score is recommended and invoked, its answer compared to a vetted ground-truth, and the resulting similarity score used to update the bandit's parameters. Over time, this approach automatically recommends the LLM most likely to answer each question correctly, maximizing cumulative answer quality, minimizing hallucinations, and saving API-call costs by avoiding poorly performing models. We demonstrate the system's ability to adapt per topic, outperform random or fixed-model baselines, and provide a safe, efficient mechanism for dynamic LLM recommendation in critical applications.

***Keywords*— "Contextual Bandits", "Recommender System", "LinUCB", "Medical Question Answering", "LLM Hallucination Mitigation", "Exploration–Exploitation", "Dynamic Model Selection", "API Cost Reduction"**

I. INTRODUCTION

Large Large language models (LLMs) have shown remarkable success in medical question answering by leveraging extensive pre-trained knowledge. However, in critical healthcare applications, "hallucinations"—confidently generated yet incorrect answers—can have severe consequences for patient safety. Relying solely on a single LLM increases the risk of inaccuracies in specific medical topics where the model may underperform.

To tackle this issue, we propose a lightweight, data-driven recommender system that dynamically selects among multiple LLMs based on each medical query's characteristics. Our approach frames model selection as a contextual bandit problem, encoding each question's text and topic label and employing an Upper-Confidence Bound (LinUCB) algorithm to balance exploration (trying less-tested models) with exploitation (using the best-known models). The system outputs the recommended LLM expected to answer the query accurately and with minimal hallucination. This method reduces hallucination risk, adapts automatically without manual intervention, and lowers API call costs by avoiding unsuitable models.

II. PROBLEM STATEMENT

In medical question answering systems, large language models (LLMs) frequently produce "hallucinations"—responses that appear confident but are factually incorrect. Relying exclusively on a single LLM amplifies the risk of inaccuracies, particularly when that model struggles with certain medical topics. Thus, there is a critical need for a system that dynamically identifies and recommends the most accurate LLM for each specific medical query, minimizing hallucinations and ensuring patient safety.

III. RL FORMULATION

Concept	Explanation
State	Before choosing a model, we summarize everything we know about the new question in one fixed "vector" of numbers. This vector has two parts: • **Question embedding**: we run the question text through a sentence-encoder that outputs 384 numbers capturing its meaning. • **Topic one-hot**: we convert the question's topic (like "Anatomy" or "Pharmacology") into a 21-element list where exactly one position is 1 and the rest are 0. By gluing these together, we get a single fingerprint of the question that our bandit can use to distinguish contexts.
Action (Models)	These are the choices available for each question. In our case we have three "actions," meaning three large-language models**: Google's Gemini 2.0 Flash**, **Microsoft's Phi-3-mini**, and **Alibaba's Qwen-1.5-4B**. Pulling an arm means "call that model on the question and get its answer." The bandit's job is to learn, for each type of question, which arm tends to give the best answer.
Reward (r)	After calling a model, we compare its answer to our trusted ground-truth answer. We compute a cosine similarity score between [0,1] (where 1 means perfect match, 0 means no overlap). That score is the "reward." A high reward means the model answered accurately (no hallucination); a low reward means the answer was poor or misleading. Over time, the bandit uses these rewards to learn which models to trust in which situations.
Estimate (θ_a)	For each model (arm) we maintain a set of parameters θ_a that represent our current best guess of how that model's reward depends on the context vector x. As we collect rewards, θ_a shifts toward values that predict high reward for contexts where the model did well. Conceptually, θ_a is a "preference profile" for model a over all possible question-features.
Uncertainty Bonus	Whenever we predict a model's reward for a new question, we add a "bonus" that reflects how uncertain we are about that prediction. If we have rarely tried model a on questions like this one, the bonus is large—pushing us to explore and gather data. If we have tried it many times, the bonus is small—letting us exploit what we already know. Mathematically this bonus comes from measuring how "far" the new question's vector lies in parts of feature-space we haven't seen for that model.
Upper-Confidence Bound	We combine the two pieces—the current estimate ($\theta_a \cdot x$) and the uncertainty bonus—to form one score called the UCB score for each model. This score represents an optimistic prediction of how well the model might do, balancing what we know (estimate) against what we don't (uncertainty). We then pick the model with the highest UCB score. This single decision rule automatically handles exploration (via the bonus) and exploitation (via the estimate) in a principled way.
Update Mechanism	Once we see the actual reward ***r*** for the chosen model on that question, we update two internal summaries for that model: • We record that we have now seen this context one more time. • We add the observed reward to our running total for contexts like this. These updates refine both the estimate θ_a (so it better predicts future rewards) and shrink the uncertainty bonus for similar contexts (since we've now gathered more data there).
Objective	Our ultimate goal is to make the sum of all rewards as large as possible across many questions. In practice this means we want to route each question to the model that is most likely to answer it accurately, thereby minimizing hallucinations. Because each question is independent, we simply add up the individual rewards—no discounting or long-term planning is needed.
Why This Learns	- **Early on**, uncertainty bonuses dominate, so we try all models on each topic to gather data. - **As data accumulates**, estimates become more reliable and bonuses shrink, so we gradually favor the model with the best track record for each topic. - **If performance changes** (a model improves), its uncertainty in new contexts grows again, triggering fresh exploration. This dynamic ensures we never get stuck with a sub-optimal model and continually adapt to real-world accuracy.

IV. METHODOLOGY USED

A. Database

We used the MedMCQA dataset (link) for this project. Specifically, we utilized the train.json file, available in JSON format. Each entry (row) in the original dataset consists of the following fields:

- **question**: The medical question text.
- **cop**: The correct option (e.g., "a", "b", "c", "d").
- **opa, opb, opc, opd**: Text for options a, b, c, and d, respectively.
- **exp**: An explanation for why the correct option is correct.
- **subject_name**: The subject category of the question (e.g., Anatomy, Pharmacology).
- **topic_name**: The more specific topic within the subject.

To prepare the dataset for our model, we created a new, refined database by:

- Combining the original **Question** and all four options (**opa, opb, opc, opd**) into a single comprehensive **question** variable.
- Creating an **answer** variable by merging the **exp** and the text of the correct option, forming a complete and informative answer.
- Retaining **subject_name** and **topic_name** as separate fields for use as context information.

This refined database served as the foundation for training our contextual-bandit recommender model, as detailed in subsequent sections of the report.

B. Model Training and Selection

After the refined database was ready, we proceeded with training our contextual-bandit recommender model. We structured the training process into the following detailed steps:

1. *Generating Answers from AI Models:*
Each medical query from our refined database was fed separately into each of the three large language models (Gemini 2.0 Flash, Phi-3-mini, and Qwen-1.5-4B) using their respective APIs. Each model produced its own independent answer to each query.

2. Evaluating Model Answers (Reward Calculation):
To objectively measure how accurate each model's answer was, we calculated a reward score by comparing the model-generated answers to our stored ground-truth answers. The specific metric we employed was the **Cosine Similarity Score**, scaled between 0 and 1:

- **Cosine Similarity:** Measures how similar two text embeddings are, returning a value from -1 (completely dissimilar) to 1 (identical meaning).
- We used a sentence embedding model (all-MiniLM-L6-v2) to convert the answers into numeric vectors.
- To ensure rewards were non-negative and easy to interpret, we scaled the original cosine similarity from [-1,1] to [0,1] using the formula:

$$Reward = (CosineSimilarity (model\_answer, ground\_truth) + 1) / 2$$

Thus, a reward close to 1 meant a highly accurate answer (minimal hallucination), while a reward near 0 indicated poor accuracy.

3. Contextual Bandit Formation and Implementation (LinUCB):
We used the contextual-bandit framework with the LinUCB (Linear Upper Confidence Bound) algorithm to dynamically select the best-performing model based on context. LinUCB allowed us to continuously learn from previous observations and handle uncertainty by balancing exploration and exploitation:

- *Exploration:*
    - If a certain model was rarely tested on a specific topic, the LinUCB algorithm added a large uncertainty bonus to its estimated reward score.

    - This uncertainty bonus encouraged our recommender to occasionally choose less-tested models, ensuring that potentially superior models were not overlooked due to lack of historical data.
- *Exploitation:*
    - As more data accumulated, the uncertainty bonus naturally decreased. Thus, when sufficient data was available, the model with the highest demonstrated reward (accuracy) was chosen.
    - This allowed our recommender to confidently rely on models proven to deliver accurate answers for specific topics.

LinUCB Decision Rule: For each query context (represented by question embeddings and topic embeddings), the algorithm calculated the Upper Confidence Bound score for each model:

$$\textbf{\textit{UCB Score} = \textit{(Estimated Reward)} + } \alpha \times \textbf{\textit{(Uncertainty Bonus)}}$$

- *Estimated Reward:* Based on previously observed rewards for similar contexts.
- *Uncertainty Bonus:* Calculated as the measure of uncertainty or unfamiliarity with a particular context, ensuring proper exploration.
- *α (alpha)***:** A parameter we tuned experimentally (chosen value: α=1.5) to control the balance between exploration and exploitation.

4. Model Training Procedure:
In each iteration of training:

- The model selected one AI model (arm) using the LinUCB decision rule.
- It obtained the actual answer from the selected AI model.
- The calculated reward (accuracy score) was used to update LinUCB's internal statistics (A matrix and b vector), thereby refining the estimates for future predictions.

5. Final Model Recommendation:
After sufficient training, our recommender model could reliably predict the most accurate LLM for answering new medical queries. Given a new query characterized by its topic and textual embedding, the trained LinUCB model computed the UCB scores and recommended the best-performing LLM—the one that previously exhibited minimal hallucinations and maximal accuracy for similar contexts.

V. CONTRIBUTIONS

1. Agaaz Singhal:

- Proposed and finalized the idea of using the Contextual Bandit approach, identifying it as the most suitable reinforcement-learning technique to dynamically select accurate LLMs.
- Designed and implemented the core LinUCB algorithm, managing the crucial exploration–exploitation balance to optimize model selection.
- Developed and optimized the embedding-based reward calculation methodology using Cosine Similarity scores.
- Led comprehensive parameter tuning and testing procedures, especially fine-tuning critical parameters like alpha to achieve optimal performance.

This positioned Agaaz as the technical lead for the project, taking responsibility for the core algorithmic design and ensuring the system's adaptability, accuracy, and robustness in minimizing hallucinations.

2. Krishnav Mahansaria:

- Located and accessed the MedMCQA dataset (link), reviewing multiple options and choosing a reliable and comprehensive source for the project.
- Worked extensively on initial dataset exploration, cleaning, and validation processes to ensure data quality and integrity before training.
- Worked with Mudit in the Integration of APIs.

- Worked closely with Agaaz on the design and implementation of the LinUCB algorithm, contributing to the exploration–exploitation mechanism.
- Significantly worked on report structuring and documentation and the Final Presentation.

3. Mudit Surana:

- Conducted significant refinement and preprocessing of the database, merging question texts, options, and answers into a structured format suitable for effective model training.
- Managed the integration of APIs, ensuring effective communication and querying of the three AI models (Gemini-Pro, Phi-3-mini, and Qwen-1.5-4B).
- Played a central role in interpreting initial results, analyzing accuracy, and significantly contributed to report structuring and documentation.
- Worked closely with Agaaz on developing and optimizing the reward calculation mechanism using sentence embeddings and cosine similarity.

While the above roles outline individual contributions, it is important to highlight that all three team members worked closely together throughout the project. Each person contributed to the best of their ability and was always available to support and guide the others whenever needed. The division of tasks was not rigid—rather, we operated as a collaborative unit, continuously learning from and helping each other. These combined efforts allowed us to complete the project on time and with confidence. The individual breakdown of responsibilities serves to provide structure but is not a complete reflection of the hard work, mutual support, and dedication that each teammate brought to the project.

VI. RESULTS

To evaluate the effectiveness of our contextual-bandit-based LLM recommender system, we conducted experiments on our refined medical QA dataset using the LinUCB algorithm. The goal was to train the model to identify, for each medical query, which language model would produce the most accurate (least hallucinated) answer.

1. Training Performance and Alpha Tuning

To balance exploration and exploitation, we experimented with different values of the exploration parameter **α (alpha)**. A lower alpha encourages exploitation (choosing models known to perform well), while a higher alpha encourages exploration (trying lesser-tested models). The training was executed across four different alpha values: 0.5, 1.0, 1.5, and 2.0.

Alpha (α)	Average Reward
0.5	**0.7102**
1.0	0.6588
1.5	0.6472
2.0	0.6368

The best average reward was achieved at **α = 0.5**, indicating that the model benefited more from confident exploitation than frequent exploration in our dataset. This alpha value was selected as the optimal setting for deployment.

2. Topic-wise Best Performing Models

After training, the model was able to recommend the most suitable AI model for each topic in the dataset. Below is the summary of which model was most frequently recommended per medical topic based on reward performance:

- **Gemini 2.0 Flash** was recommended for a vast majority of topics, indicating its consistency and reliability across disciplines such as Anatomy, Pathology, Pharmacology, and Pediatrics.
- **Phi-3-mini** was favored for specific areas like **Medicine**, **Skin**, and entries marked as **Unknown**.

Topic	Best Model
Anaesthesia	gemini_ans
Anatomy	gemini_ans
Biochemistry	gemini_ans
Dental	gemini_ans
ENT	gemini_ans
Forensic Medicine	gemini_ans
Gynaecology & Obstetrics	gemini_ans
Medicine	phi3mini_ans
Microbiology	gemini_ans
Ophthalmology	gemini_ans
Orthopaedics	gemini_ans
Pathology	gemini_ans
Pediatrics	gemini_ans
Pharmacology	gemini_ans
Physiology	gemini_ans
Psychiatry	gemini_ans
Radiology	gemini_ans
Skin	phi3mini_ans
Social & Preventive Medicine	gemini_ans
Surgery	gemini_ans
Unknown	phi3mini_ans

Table: *Summary of which model was most frequently recommended per medical topic based on reward performance*

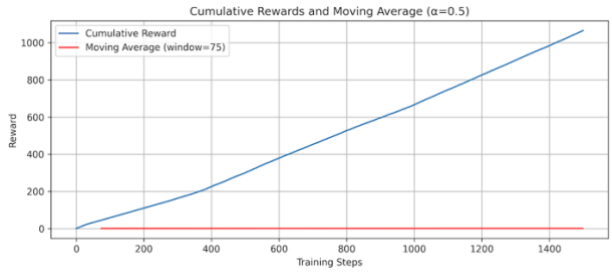

Fig.1: *This graph shows the cumulative sum of rewards (i.e., how accurate the system has been overall) over training steps, along with the moving average to visualize trends in recent performance.*

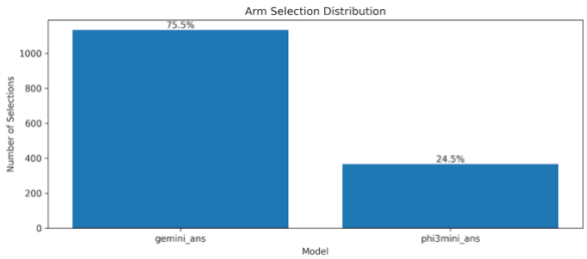

Fig.2: *This bar chart shows how frequently each model (arm) was selected by the LinUCB policy during training. Gemini was chosen ~75.5% of the time, while Phi3mini was chosen ~24.5%, reflecting relative confidence in each model.*

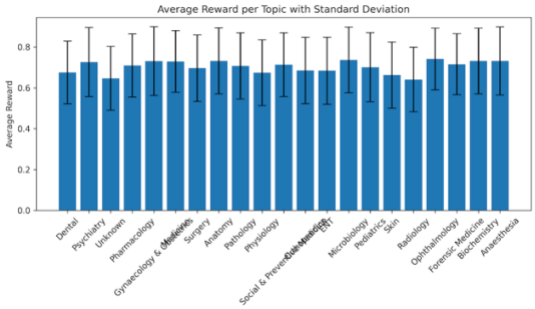

Fig.3: *This chart shows the average reward (accuracy score) obtained per topic, along with error bars representing standard deviation. This helps us understand both average performance and variability across medical domains.*

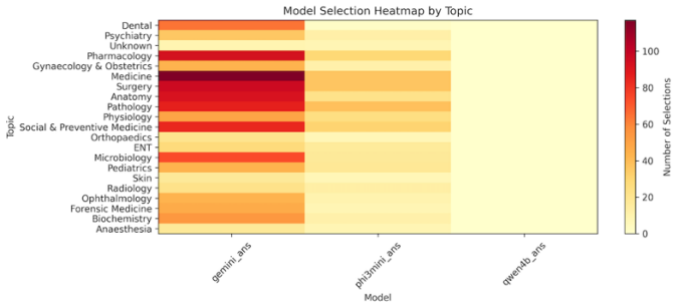

Fig.4: *This heatmap shows how often each model was selected for different medical topics. Darker shades indicate higher selection frequency. It highlights Gemini's dominance across topics, with Phi3mini emerging in select cases like Medicine and Skin.*

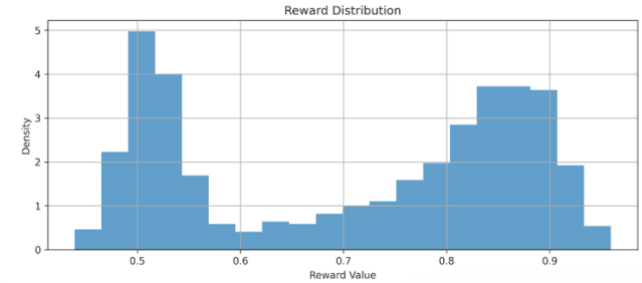

Fig.5: *This histogram shows the distribution of all reward values (i.e., model answer accuracy scores). The peak around 0.85–0.90 suggests the model frequently made high-confidence, accurate recommendations.*

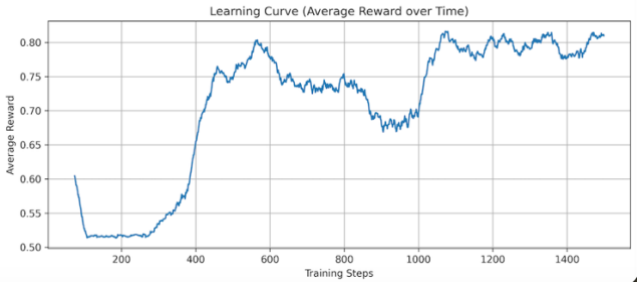

Fig.6: *This graph tracks the average reward (i.e., model answer quality) over time. The trend indicates progressive learning, with phases of improvement followed by plateaus, likely as the system balances exploration and exploitation.*

3. Example Output: Real Query Recommendation

As a real-world test, we entered the question: **"Which nerve supplies the biceps brachii?" Topic:** Surgery

The recommender system suggested: **Gemini 2.0 Flash**, indicating that based on training, Gemini had historically shown the best accuracy and lowest hallucination rate for similar surgical queries.

These results confirm that our model not only learns topic-specific preferences across LLMs but also recommends the most suitable model with measurable confidence. This ability to adapt per topic and maximize output accuracy validates the effectiveness of applying contextual bandits to minimize hallucinations in medical question answering.