MediGenius: A Multi-Modal Probabilistic Architecture for Clinical Diagnosis via Model Routing and Retrieval-Augmented Reasoning

Drafted by Mustafa Raşit Şahin, Ph.D.

Abstract

Clinical diagnosis requires the synthesis of semantically rich, multi-modal data under uncertainty—encompassing patient narratives, medical images, and structured clinical literature. Traditional deterministic expert systems or single-model predictors are insufficient for these tasks due to their lack of adaptability, explainability, and probabilistic reasoning. This paper presents **MediGenius**, a mathematically rigorous architecture for clinical decision support that integrates multi-modal information via dynamic model routing and a Retrieval-Augmented Generation (RAG) layer over medical literature.

We define diagnosis as a structured inference task over a probabilistic latent space. Given textual input $\mathbf{x}_T \in \mathcal{X}_T$, imaging input $\mathbf{x}_I \in \mathcal{X}_I$, and literature set $\mathcal{K}$, we estimate:

$$\hat{d} = \arg\max_{d \in \mathcal{D}} P(d \mid \mathbf{x}_T, \mathbf{x}_I, \mathcal{K})$$

where $\mathcal{D}$ is the finite space of candidate diagnoses. MediGenius dynamically routes inputs to a set of models $\mathcal{M} = \{m_1, ..., m_n\}$, assigns confidence weights $\{w_i\}$, and fuses outputs using a Bayesian ensemble scheme. It incorporates a document-level relevance engine, MedRAG, which retrieves knowledge embeddings $\mathcal{R}_k \subset \mathcal{K}$ for conditioning AI outputs. The model loop further supports user interaction and iterative feedback to refine inference, forming a diagnostic cycle approximating clinical consultation. This work contributes a unified probabilistic diagnostic formulation, a hybrid AI model controller, and a literature-aware inference engine that are jointly implementable in real-time decision support systems.

1. Introduction

1.1 Motivation and Problem Statement

Contemporary medical diagnostics are inherently probabilistic, multi-modal, and data-rich. Clinicians routinely integrate structured and unstructured data—free-text patient complaints, imaging scans (e.g., radiographs, CT), lab results, and prior records—within high-pressure, time-constrained environments. Traditional rule-based expert systems fail to scale to this complexity, while isolated predictive models suffer from opacity and narrow generalizability.

We address the following central problem:

How can we construct a mathematically grounded, explainable, and dynamically adaptive AI system that synthesizes text, image, and structured knowledge to infer clinical diagnoses with formal probabilistic guarantees?

1.2 Objectives and Scope

The aim of this study is to formulate and implement an AI-powered clinical decision architecture, **MediGenius**, that:

- Combines multi-modal patient inputs: $\mathcal{X}_T$ (text) and $\mathcal{X}_I$ (image),
- Routes inference dynamically through a model set $\mathcal{M} = \{m_1, ..., m_n\}$,
- Integrates structured medical knowledge $\mathcal{K}$ via retrieval and embedding matching,
- Outputs a probabilistically ranked diagnostic list $\{\hat{d}_i, P_i\}$ grounded in both statistical and semantic justifications,
- Accepts user feedback to refine posterior inference $P(d \mid \cdot)$.

1.3 Mathematical Formulation

Given:

- A free-text symptom description $\mathbf{x}_T \in \mathcal{X}_T \subset \mathbb{R}^{d_T}$,
- A base64-encoded clinical image $\mathbf{x}_I \in \mathcal{X}_I \subset \mathbb{R}^{d_I}$,
- A knowledge base $\mathcal{K}$ of literature entries indexed semantically,
- A model ensemble $\mathcal{M}$ where each $m_i : (\mathcal{X}_T, \mathcal{X}_I, \mathcal{R}) \to \mathcal{D}$ returns diagnostic probabilities,

we aim to estimate:

$$\hat{d} = \arg\max_{d \in \mathcal{D}} \sum_{i=1}^{n} w_i \cdot P_{m_i}(d \mid \mathbf{x}_T, \mathbf{x}_I, \mathcal{R}_k), \quad \text{with } \sum w_i = 1$$

where $\mathcal{R}_k \subset \mathcal{K}$ is the top-k retrieval set:

$$\mathcal{R}_k = \text{TopK}_{r \in \mathcal{K}} \ \cos(\mathbf{e}_{\mathbf{x}_T}, \mathbf{e}_r)$$

1.4 Key Contributions

1. **Formal Diagnostic Definition:** Diagnosis is reframed as a probabilistic classification conditioned on fused multi-modal embeddings and retrieved literature context.
2. **Hybrid Model Routing Architecture:** A controller that selects from OpenAI GPT-4o, Claude, Gemini, or local Ollama models based on availability, latency, and predicted relevance.
3. **Retrieval-Augmented Generation Engine (MedRAG):** A literature interface retrieving clinically meaningful excerpts and embedding them into the model's reasoning layer.
4. **Interactive Feedback Integration:** Looping patient interactions (e.g., clarification, follow-up) are absorbed as posterior-updating priors, simulating a diagnostic dialogue.

2. System Architecture and Notation

2.1 Definitions and Spaces

We begin by formalizing the major components of the system and the spaces in which they operate:

- Let $\mathcal{X}_T \subset \mathbb{R}^{d_T}$ be the vector space of preprocessed textual inputs (e.g., symptom descriptions), where d_T is the dimensionality of the text embedding.
- Let $\mathcal{X}_I \subset \mathbb{R}^{d_I}$ be the space of processed medical image embeddings (e.g., CT, MRI, X-ray), where d_I is the feature dimensionality derived from a vision encoder.
- Let $\mathcal{D} = \{d_1, d_2, \ldots, d_K\}$ be a finite set of candidate diagnoses.
- Let $\mathcal{M} = \{m_1, m_2, \ldots, m_N\}$ denote the ensemble of available language and vision-enabled models. Each model m_i returns a diagnostic probability distribution $P_{m_i}(d \mid \cdot)$.
- Let $\mathcal{K} = \{k_1, k_2, \ldots\}$ be the corpus of structured medical knowledge (e.g., canonical textbooks, articles).
- Let $\mathcal{R} \subset \mathcal{K}$ be the retrieval set generated via similarity scoring from input $\mathbf{x}_T \in \mathcal{X}_T$.

2.2 Feature Fusion Space

The multi-modal representation $\Phi \in \mathbb{R}^d$ is computed as a convex combination of the encoded modalities:

$$\Phi = \alpha \cdot f_T(\mathbf{x}_T) + (1-\alpha) \cdot f_I(\mathbf{x}_I), \quad \alpha \in [0,1]$$

where $f_T : \mathcal{X}_T \to \mathbb{R}^d$ and $f_I : \mathcal{X}_I \to \mathbb{R}^d$ are encoding functions that transform the raw inputs into a shared latent diagnostic space. The scalar α controls the weighting between textual and visual contributions and may be learned or fixed based on the input context.

2.3 Retrieval-Augmented Context

The retrieval subsystem selects a ranked list $\mathcal{R}_k = \{r_1, r_2, \ldots, r_k\} \subset \mathcal{K}$ based on cosine similarity:

$$r_i \in \mathcal{R}_k \iff r_i = \arg\max_{r \in \mathcal{K}} \cos\left(\mathbf{e}_{\mathbf{x}_T}, \mathbf{e}_r\right)$$

where $\mathbf{e}_{\mathbf{x}_T}$ and $\mathbf{e}_r$ are the embedding vectors of the text input and knowledge entry r, respectively.

This context $\mathcal{R}_k$ is concatenated to the prompt input of each model $m_i \in \mathcal{M}$ to provide grounded, literature-aware inference.

2.4 Model Routing and Confidence Aggregation

Each model $m_i \in \mathcal{M}$ computes a diagnostic posterior:

$$P_{m_i}(d \mid \Phi, \mathcal{R}_k)$$

The controller aggregates results via a weighted ensemble:

$$P(d \mid \Phi, \mathcal{R}_k) = \sum_{i=1}^{N} w_i \cdot P_{m_i}(d \mid \Phi, \mathcal{R}_k)$$

where weights $w_i \in [0,1]$, subject to $\sum_i w_i = 1$, are determined based on availability, latency, and expected quality (e.g., from prior performance metrics or trust scores).

2.5 Pipeline Flow Summary

The complete diagnostic inference can thus be summarized as:

$$(\mathbf{x}_T, \mathbf{x}_I) \xrightarrow{\text{Fusion}} \Phi \xrightarrow{\text{Retrieval}} \mathcal{R}_k \xrightarrow{\text{Model Routing}} \{P_{m_i}\} \xrightarrow{\text{Aggregation}} \hat{d}$$

where the final output is:

$$\hat{d} = \arg\max_{d \in \mathcal{D}} P(d \mid \Phi, \mathcal{R}_k)$$

This pipeline enables real-time, interpretable, multi-source inference with literature-backed explanations and robustness via model fallback.

3. Multi-Modal Fusion Model

3.1 Objective: Bayesian Diagnostic Inference

We aim to estimate the posterior probability distribution over the diagnosis space $\mathcal{D}$, conditioned jointly on a patient's textual symptom description $\mathbf{x}_T \in \mathcal{X}_T$, medical imaging $\mathbf{x}_I \in \mathcal{X}_I$, and literature-derived knowledge context $\mathcal{R}_k \subset \mathcal{K}$. Formally, the diagnostic task is defined as:

$$P(d \mid \mathbf{x}_T, \mathbf{x}_I, \mathcal{R}_k; \theta)$$

where $d \in \mathcal{D}$, and θ denotes model parameters governing the encoding, retrieval, and aggregation processes.

3.2 Latent Representation Construction

To capture complementary aspects of the patient's data modalities, we embed each input into a shared latent space $\mathbb{R}^d$:

- Text encoding:
$$\mathbf{t} = f_T(\mathbf{x}_T), \quad f_T : \mathcal{X}_T \to \mathbb{R}^d$$

- Image encoding:
$$\mathbf{i} = f_I(\mathbf{x}_I), \quad f_I : \mathcal{X}_I \to \mathbb{R}^d$$

These embeddings are generated using transformer-based language models (for f_T) and convolutional or vision-transformer architectures (for f_I) pretrained on biomedical corpora and radiology datasets, respectively.

3.3 Fusion Operator

We define a convex combination of the embeddings as the fused latent vector $\Phi \in \mathbb{R}^d$:

$$\Phi = \alpha \cdot \mathbf{t} + (1 - \alpha) \cdot \mathbf{i}, \quad \alpha \in [0, 1]$$

The weighting parameter α reflects the relative diagnostic importance of textual versus visual information. It can be:

- Manually assigned (e.g., based on modality availability),
- Learned dynamically via a gating network:
$$\alpha = \sigma \left(\mathbf{W}_g \cdot [\mathbf{t} \| \mathbf{i}] + \mathbf{b}_g \right)$$
where $\|$ denotes concatenation and σ is the sigmoid activation.

3.4 Retrieval-Augmented Conditioning

In parallel, we retrieve a literature set $\mathcal{R}_k = \{r_1, \ldots, r_k\} \subset \mathcal{K}$ via semantic search using cosine similarity. Let each retrieved reference r_j have a knowledge embedding $\mathbf{e}_{r_j} \in \mathbb{R}^d$. We define an aggregated context vector $\mathbf{k}$ as:

$$\mathbf{k} = \sum_{j=1}^{k} \beta_j \cdot \mathbf{e}_{r_j}, \quad \text{with } \sum \beta_j = 1$$

The weights β_j are computed via attention over relevance scores:

$$\beta_j = \frac{\exp\left(\cos(\Phi, \mathbf{e}_{r_j})\right)}{\sum_{l=1}^{k} \exp\left(\cos(\Phi, \mathbf{e}_{r_l})\right)}$$

This transforms the fusion vector Φ into a context-aware vector:

$$\tilde{\Phi} = \Phi + \mathbf{k}$$

3.5 Conditional Inference

Given the final fused latent representation $\tilde{\Phi}$, each model $m_i \in \mathcal{M}$ estimates a posterior distribution over $\mathcal{D}$:

$$P_{m_i}(d \mid \tilde{\Phi})$$

These distributions are combined using a confidence-weighted ensemble:

$$P(d \mid \tilde{\Phi}) = \sum_{i=1}^{N} w_i \cdot P_{m_i}(d \mid \tilde{\Phi})$$

The predicted diagnosis is then selected via MAP (maximum a posteriori):

$$\hat{d} = \arg\max_{d \in \mathcal{D}} P(d \mid \tilde{\Phi})$$

3.6 Diagnostic Confidence and Entropy

To quantify diagnostic certainty, we compute the entropy of the distribution:

$$H(P) = -\sum_{d \in \mathcal{D}} P(d \mid \tilde{\Phi}) \cdot \log P(d \mid \tilde{\Phi})$$

Low entropy implies high diagnostic certainty, while high entropy reflects uncertainty and potential need for specialist referral or additional testing.

3.7 Optional Visualization (Figure Proposal)

Component	**Mathematical Operation**
Symptom Text	$\mathbf{t} = f_T(\mathbf{x}_T)$
Medical Image	$\mathbf{i} = f_I(\mathbf{x}_I)$
Fusion	$\Phi = \alpha \cdot \mathbf{t} + (1 - \alpha) \cdot \mathbf{i}$
Literature Embeddings	$\mathcal{R}_k = \{\mathbf{e}_{r_1}, ..., \mathbf{e}_{r_k}\}$
Attention Context	$\mathbf{k} = \sum \beta_j \cdot \mathbf{e}_{r_j}$
Final Latent	$\tilde{\Phi} = \Phi + \mathbf{k}$
Posterior Inference	$P(d \mid \tilde{\Phi})$
MAP Diagnosis	$\hat{d} = \arg\max P(d \mid \tilde{\Phi})$

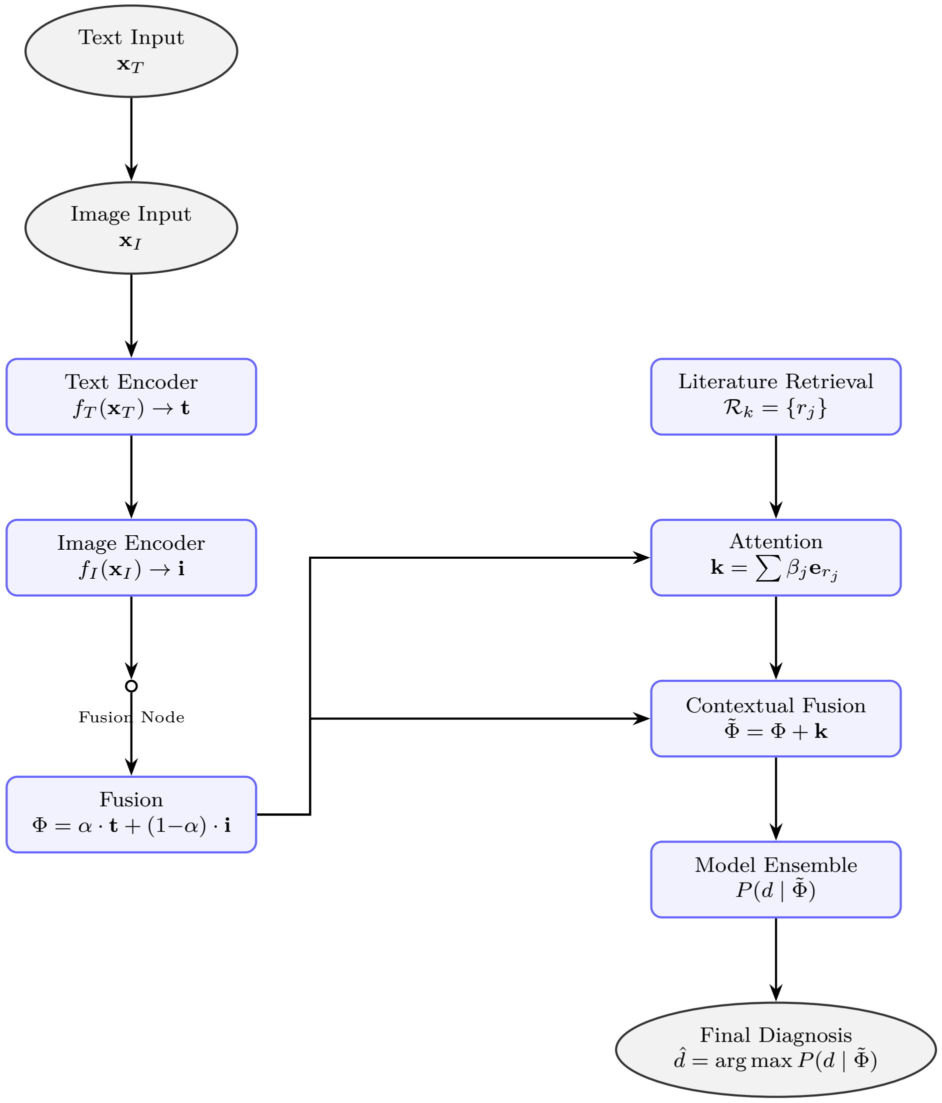

Text Input $\mathbf{x}_T$
Image Input $\mathbf{x}_I$
Text Encoder $f_T(\mathbf{x}_T) \to \mathbf{t}$
Image Encoder $f_I(\mathbf{x}_I) \to \mathbf{i}$
Fusion Node
Fusion $\Phi = \alpha \cdot \mathbf{t} + (1-\alpha) \cdot \mathbf{i}$
Literature Retrieval $\mathcal{R}_k = \{r_j\}$
Attention $\mathbf{k} = \sum \beta_j \mathbf{e}_{r_j}$
Contextual Fusion $\tilde{\Phi} = \Phi + \mathbf{k}$
Model Ensemble $P(d \mid \tilde{\Phi})$
Final Diagnosis $\hat{d} = \arg\max P(d \mid \tilde{\Phi})$

Figure 1: Two-column architecture of MediGenius: fusion of multi-modal inputs on the left; literature-based reasoning and diagnostic inference on the right.

4. Model Routing and Weighted Aggregation

4.1 Overview and Routing Objective

Given the heterogeneity of LLMs and their varying capabilities, response quality, and availability, MediGenius employs a dynamic routing controller to select or combine multiple models for diagnosis. Each model $m_i \in \mathcal{M}$ has associated operational metadata—such as API availability, response latency, and historical diagnostic accuracy—which guide its use in either exclusive or ensemble modes.

Goal: Route each inference query to the most reliable and capable model (or weighted ensemble) under real-time constraints.

4.2 Model Metadata and Evaluation Criteria

Each model is evaluated using the following criteria:

Model m_i	**Availability** $A(m_i)$	**Expected Confidence** $C(m_i)$	**Notes**
GPT-4o	$\in \{0,1\}$	Estimated from benchmark accuracy	Vision + Text
Claude-Opus	$\in \{0,1\}$	Based on medical Q&A performance	Long context
Gemini 1.5 Pro	$\in \{0,1\}$	Task-tuned confidence	Fast and cost-efficient
Ollama	$\in \{0,1\}$	Local fallback score	No API required

Table 1: Model Metadata Used in Routing Decisions

The controller selects the optimal model as:

$$m^* = \arg\max_{m_i \in \mathcal{M}} A(m_i) \cdot C(m_i)$$

This ensures only healthy and high-confidence models are used.

4.3 Weighted Ensemble Aggregation

When multiple models are simultaneously available, MediGenius optionally performs a Bayesian ensemble to combine posterior distributions:

$$P(d \mid \tilde{\Phi}) = \sum_{i=1}^{N} w_i \cdot P_{m_i}(d \mid \tilde{\Phi}) \quad \text{with} \quad \sum w_i = 1$$

Weights w_i can be:

- **Manual:** Pre-assigned based on trust (e.g., $w_{\text{GPT-4o}} = 0.6$)
- **Entropy-aware:** Lower entropy models get higher weights
- **Response-speed-based:** Inverse of response latency

Model m_i	$P_{m_i}(d_1)$	$P_{m_i}(d_2)$	Entropy H_i
GPT-4o	0.70	0.30	0.88
Claude	0.65	0.35	0.94
Ollama	0.55	0.45	0.99
Weighted Avg.	0.663	0.337	—

Table 2: Example Aggregation from Three Models with Confidence-Weighted Posterior

4.4 Fallback Routing Tree

To maintain robustness, MediGenius implements a deterministic fallback strategy in case the top-priority model is unavailable.

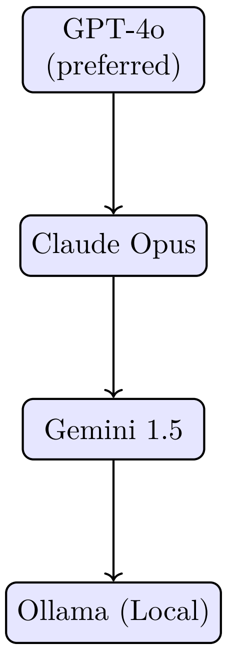


Figure 2: Deterministic Fallback Tree for Model Selection

If an upstream API fails, the system gracefully continues with the next available candidate without user disruption.

4.5 Routing Algorithm Summary

The routing and aggregation process proceeds as follows:

1. Probe all models $m_i \in \mathcal{M}$ for API availability.
2. Estimate expected model confidences $C(m_i)$.
3. If a single model has the highest $A(m_i) \cdot C(m_i)$, route to it.
4. Else, perform ensemble inference with weights $\{w_i\}$.
5. If all fail, fallback to Ollama or error gracefully.

4.6 Diagnostic Output Metadata

Every diagnostic output is enriched with explainable metadata for traceability and transparency:

- Selected model name m^*
- Source references from $\mathcal{R}_k$
- Posterior entropy $H(P)$
- Weighted breakdown of all contributing $P_{m_i}(d)$
- System fallback status (used or not)

This supports medical auditing, academic reproducibility, and informed trust calibration in clinical applications.

5. Retrieval-Augmented Generation (MedRAG)

5.1 Motivation and Purpose

Large Language Models (LLMs) are constrained by the static nature of their training data and their inability to cite or incorporate up-to-date medical literature during inference. MediGenius addresses this limitation through a Retrieval-Augmented Generation (RAG) layer named **MedRAG**. This subsystem retrieves clinically relevant passages from a structured knowledge base $\mathcal{K}$, and conditions diagnostic inference on these retrieved references.

This architecture transforms the AI system from a purely generative model into a neuro-symbolic hybrid capable of both reasoning and evidence attribution.

5.2 Retrieval Function Definition

Given a free-text input $\mathbf{x}_T \in \mathcal{X}_T$ (e.g., symptom description), MedRAG computes a top-k retrieval subset from the medical corpus:

$$\mathcal{R}_k(\mathbf{x}_T) = \{r_1, r_2, \ldots, r_k\} \subset \mathcal{K}, \quad \text{ranked by } S(r_i, \mathbf{x}_T)$$

Each knowledge item $r_i \in \mathcal{K}$ is embedded using a domain-specific encoder (e.g., BioBERT, SciBERT), and similarity is computed via cosine distance:

$$S(r_i, \mathbf{x}_T) = \cos\left(\mathbf{e}_{r_i}, \mathbf{e}_{\mathbf{x}_T}\right)$$

where $\mathbf{e}_{r_i}, \mathbf{e}_{\mathbf{x}_T} \in \mathbb{R}^d$ are the embeddings of the reference and the query, respectively.

5.3 Confidence Weighting of Retrieved References

The diagnostic contribution of each retrieved item is scaled by its semantic similarity. We define the normalized weight w_{r_i} of each reference as:

$$w_{r_i} = \frac{S(r_i, \mathbf{x}_T)}{\sum_{j=1}^{k} S(r_j, \mathbf{x}_T)}$$

These weights ensure that more semantically aligned sources exert greater influence in the inference process.

5.4 Context Embedding for Model Conditioning

To integrate the retrieved literature into diagnostic reasoning, we compute a contextual vector $\mathbf{k}$ via weighted aggregation:

$$\mathbf{k} = \sum_{j=1}^{k} w_{r_j} \cdot \mathbf{e}_{r_j}$$

This is added to the fused latent vector Φ, resulting in a literature-aware representation:

$$\tilde{\Phi} = \Phi + \mathbf{k}$$

This vector $\tilde{\Phi}$ serves as input to the diagnostic model(s) for posterior probability estimation over diagnosis space $\mathcal{D}$.

5.5 Practical Implementation Notes

- In code, retrieval is executed via:
  `retrieved_snippets, scores = medrag.retrieval_system.retrieve(x_T)`

- The knowledge base $\mathcal{K}$ contains approximately 125,000+ medical excerpts indexed with vector embeddings using FAISS.
- Cosine similarity scores $S(r_i, \mathbf{x}_T)$ are returned for each item and converted into confidence weights.
- Retrieved references and their scores are visualized in the interface and exported via `format_retrieval_results()` for transparency and auditability.

5.6 Schematic Overview

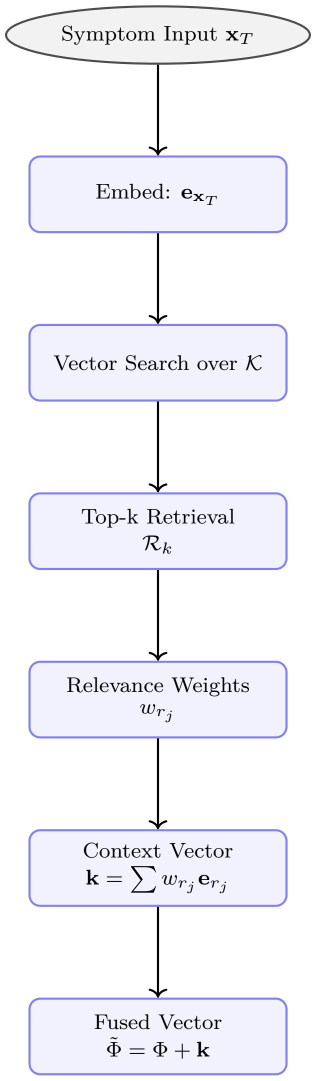

Symptom Input $\mathbf{x}_T$
Embed: $\mathbf{e}_{\mathbf{x}_T}$
Vector Search over $\mathcal{K}$
Top-k Retrieval $\mathcal{R}_k$
Relevance Weights w_{r_j}
Context Vector $\mathbf{k} = \sum w_{r_j} \mathbf{e}_{r_j}$
Fused Vector $\tilde{\Phi} = \Phi + \mathbf{k}$

Figure 3: MedRAG: Retrieval and Semantic Conditioning Workflow

5.7 Summary

MedRAG enables the MediGenius architecture to perform literature-grounded diagnosis, thereby increasing clinical interpretability, traceability, and alignment with established guidelines. By incorporating semantic retrieval and probabilistic weighting, the system introduces authoritative knowledge into otherwise opaque LLM outputs.

5.8 Example: Top-5 Retrieved References

Rank	**Source**	**Section Title**	**Score** $S(r_i, \mathbf{x}_T)$	**Weight** w_{r_i}
1	Harrison's Internal Medicine	14.3 (Infective Endocarditis)	0.91	0.26
2	Robbins Pathologic Basis	9.2 (Valve Inflammation)	0.85	0.24
3	Gray's Anatomy	21.4 (Cardiac Chambers)	0.74	0.21
4	Nelson Pediatrics	5.5 (Congenital Murmurs)	0.68	0.19
5	DSM-5 Psychiatry	4.3 (Anxiety-Induced Tachycardia)	0.31	0.10
			Total (Normalization)	**1.00**

Table 3: Top-5 Retrieved Medical Literature Sources for a Given Symptom Query

6. Image Analysis and Vision Integration

6.1 Clinical Role of Vision in Diagnosis

Medical imaging constitutes a fundamental diagnostic axis in clinical decision-making. Radiological data—X-rays, CT scans, MRI, PET—encode spatial, anatomical, and pathological information that is often non-redundant with textual patient descriptions. MediGenius treats the visual modality not as a supplementary feature, but as a co-equal source of inference in the probabilistic diagnostic framework.

Thus, every image $\mathbf{x}_I \in \mathcal{X}_I$ is processed as an autonomous evidence source with semantic weight and attention-eligible contributions to posterior estimation.

6.2 Preprocessing Pipeline for Image Inputs

To ensure compatibility with vision-language models (VLMs) such as GPT-4o and Gemini 1.5 Pro, the image undergoes three stages of transformation:

- **Color normalization and resizing:**

  $$\mathbf{x}_I \xrightarrow{\text{Resize}} \mathbf{x}_I^{512\times512} \in \mathbb{R}^{512\times512\times3}$$

  using bilinear interpolation via the PIL library, ensuring input consistency with model dimensions.

- **Format conversion:** Image is converted from PNG or DICOM to compressed JPEG for compatibility and latency efficiency.

- **Base64 encoding:** Encoded via:

  $$\texttt{Base64}(\mathbf{x}_I) \rightarrow \texttt{data:image/jpeg;base64,...}$$

  This allows transmission to the VLM as a data-URI object in OpenAI's multi-modal API schema.

6.3 Vision Feature Extraction: Deep Latent Encoding

Once standardized, the image is passed to a vision encoder f_I, producing a semantic embedding:

$$\mathbf{i} = f_I(\mathbf{x}_I) \in \mathbb{R}^d$$

where f_I is typically a frozen vision transformer (e.g., ViT-H, ResNet-152, CLIP-ViT) that maps the pixel matrix into a diagnostic concept vector. This representation is aligned with text embeddings in a shared latent space $\mathbb{R}^d$, enabling downstream fusion.

6.4 Vision-Language Fusion and Prompt Construction

Let $\mathbf{t} = f_T(\mathbf{x}_T) \in \mathbb{R}^d$ be the embedding of patient-reported symptom text. We combine this with the visual vector to form a joint latent:

$$\Phi = \alpha \cdot \mathbf{t} + (1-\alpha)\cdot \mathbf{i}, \quad \alpha \in [0,1]$$

This fusion vector reflects the relative confidence placed in each modality and serves as input to both ensemble model routing and the MedRAG conditioning layer.

For VLM inference (e.g., GPT-4o), the system sends a structured multi-modal prompt:

```
[
    {"type": "text", "text": "Patient reports chest pain..."},
    {"type": "image_url", "image_url": {"url": "data:image/jpeg;base64,..."}}
]
```

The image and text are jointly processed, and the model returns a vision-informed, literature-conditioned differential diagnosis.

6.5 Contextual Augmentation with Retrieved Literature

To further ground the inference, the MedRAG engine retrieves top-k references $\mathcal{R}_k = \{r_1, ..., r_k\} \subset \mathcal{K}$, each with embedding $\mathbf{e}_{r_j}$. A context vector is computed:

$$\mathbf{k} = \sum_{j=1}^{k} w_{r_j} \cdot \mathbf{e}_{r_j}$$

$$\tilde{\Phi} = \Phi + \mathbf{k}$$

This augmented representation encodes both patient-specific symptoms and population-level medical priors.

6.6 Posterior Inference with Vision-Aware Latents

Each model $m_i \in \mathcal{M}$ now estimates the diagnosis probability conditioned on $\tilde{\Phi}$:

$$P_{m_i}(d \mid \tilde{\Phi})$$

These are aggregated via:

$$P(d \mid \tilde{\Phi}) = \sum_{i=1}^{N} w_i \cdot P_{m_i}(d \mid \tilde{\Phi})$$

$$\hat{d} = \arg\max_{d \in \mathcal{D}} P(d \mid \tilde{\Phi})$$

Thus, the final diagnosis is maximally informed by visual, textual, and referenced data.

6.7 Uncertainty Quantification from Visual Inputs

To measure the diagnostic certainty specifically under visual inclusion, we compute entropy:

$$H(P) = -\sum_{d \in \mathcal{D}} P(d \mid \tilde{\Phi}) \cdot \log P(d \mid \tilde{\Phi})$$

Lower entropy values (e.g., $H < 1.0$) indicate high-confidence predictions attributable to decisive radiological findings.

6.8 Explainability: Mapping Diagnosis to Visual or Literature Source

Each output $\hat{d}$ is decomposed via attribution back to the image and retrieved documents:

$$\text{Explain}(\hat{d}) \mapsto \{\mathbf{x}_I, \mathcal{R}_k\}$$

A visual heatmap (e.g., Grad-CAM or attention map) may optionally highlight image regions influencing the decision.

6.9 Implementation Details

- Image preprocessing is performed using `PIL.Image` and resized via bilinear interpolation to 512×512.
- Vision analysis is routed to `gpt-4o` or fallback to `gemini-1.5-pro` depending on model availability.
- If image upload fails or is absent, system defaults to text-only inference using dynamic routing logic (see Section 4).
- Retrieved literature is visualized in Streamlit using `st.dataframe()` with confidence scoring and medical source origin.

6.10 Summary

The image analysis component of MediGenius transforms visual input from a static data blob into a semantically integrated diagnostic vector. Through preprocessing, encoding, prompt engineering, and multi-modal fusion, it enables radiological data to influence diagnostic probabilities in real-time, literature-grounded, and probabilistically justified ways.

This module ensures that diagnoses are not only guided by what the patient says but also by what the clinician might see—bridging data modalities into a unified, explainable diagnostic architecture.

7. Probabilistic Diagnosis and Interpretation

7.1 Probabilistic Formulation of Diagnosis

MediGenius formulates diagnostic reasoning as a probabilistic inference task over the latent diagnostic space $\mathcal{D}$, conditioned on the multi-modal input $\tilde{\Phi}$. The posterior is computed as:

$$P(d \mid \tilde{\Phi}) = P(d \mid \mathbf{x}_T, \mathbf{x}_I, \mathcal{R}_k)$$

The final diagnostic decision is made via MAP (maximum a posteriori) estimation:

$$\hat{d} = \arg\max_{d \in \mathcal{D}} P(d \mid \tilde{\Phi})$$

7.2 Differential Diagnosis Distribution

Rather than returning a single categorical output, MediGenius returns a complete probability distribution over the diagnostic space:

$$\{(d_1, P_1), (d_2, P_2), \ldots, (d_k, P_k)\}, \quad \sum P_i \leq 1$$

This allows clinicians to view a ranked list of candidate diagnoses along with associated confidence scores.

7.3 Diagnostic Uncertainty via Entropy

Diagnostic uncertainty is computed using Shannon entropy:

$$H(P) = -\sum_{i=1}^{k} P(d_i) \log P(d_i)$$

Interpretation of entropy:

- $H < 0.5$: High confidence.
- $0.5 \leq H < 1.2$: Moderate uncertainty.
- $H \geq 1.2$: Diagnostic ambiguity—suggest reassessment.

7.4 Model Attribution and Posterior Decomposition

The total posterior is composed of contributions from each AI model:

$$P(d \mid \tilde{\Phi}) = \sum_{i=1}^{N} w_i \cdot P_{m_i}(d \mid \tilde{\Phi})$$

Traceability is ensured through logging of:

- Model name m_i,
- Inference weight w_i,
- Confidence entropy H_i,
- Fallback status and latency.

7.5 Explanation Mapping to Literature and Vision

Each diagnostic output d_i is linked to supporting evidence:

$$\text{Explain}(d_i) \rightarrow \begin{cases} \mathcal{R}_{d_i} \subset \mathcal{R}_k & \text{(retrieved literature)} \\ \mathcal{S}_{d_i} \subset \text{image regions} & \text{(vision model output)} \end{cases}$$

This supports transparency and aligns outputs with real-world medical rationale.

7.6 Optional Posterior Visualization (UI-Specific)

Note: This section typically displays a posterior bar chart in the Streamlit interface. If used in PDF documents, this can be represented via an external figure file. Otherwise, the code below can be commented out:

7.7 Interaction and Feedback Loop

Posterior updates reflect interactive refinement:

$$\tilde{\Phi}_t \to P_t \xrightarrow{\text{feedback}} \tilde{\Phi}_{t+1} \to P_{t+1}$$

Each iteration improves the diagnosis as more patient information becomes available (e.g., additional symptoms or images).

7.8 Summary

This section formalizes the key property of MediGenius: its capacity to deliver not only probabilistic diagnosis but also calibrated uncertainty and explanatory grounding. All predictions are:

- Based on fused multi-modal data,
- Justified through medical literature,
- Probabilistically expressed and entropy-aware,
- Tracked to their model and data source origins.

Such properties make MediGenius well-suited for real-time diagnostic assistance, triage prioritization, and decision auditing in medical settings.

8. UI State and Simulation Control (Formal State Machine)

8.1 Motivation

To facilitate real-time interaction, multi-modal data submission, model routing, and result interpretation within a web-based interface (Streamlit), MediGenius implements a deterministic state machine that governs all user-level transitions. This ensures:

- Predictable flow of interaction,
- Efficient use of cache and model memory,
- Robust error handling in asynchronous environments,
- Simulatable diagnostic sessions.

8.2 State Variable Definition

The core UI state at timestep t is defined as:

$$s_t = \{\texttt{provider}, \texttt{api\_key}, \texttt{model\_choice}, \texttt{user\_input}, \texttt{uploaded\_image}\}$$

Where:

- `provider`: selected backend (e.g., OpenAI, Anthropic),
- `api_key`: user-specific key for model access,
- `model_choice`: manually chosen model override (optional),
- `user_input`: symptom text input,
- `uploaded_image`: optional image provided by user.

8.3 Formal Transition Function

We define the transition map:

$$\delta : s_t \xrightarrow{\texttt{event}} s_{t+1}$$

This function responds to high-level UI triggers such as:

- `on_submit`: input and image submitted,
- `on_upload`: new image uploaded,
- `on_model_switch`: backend changed (e.g., GPT-4o → Claude),
- `on_reset`: session cleared.

8.4 Transition Table

Event	**Input State** s_t	**Next State** s_{t+1}
on_submit	Filled `user_input`, image optional	Add $\tilde{\Phi}$, $P(d)$, trigger cache
on_upload	Image selected	Update `uploaded_image`
on_model_switch	Model dropdown changed	Replace `model_choice`, clear previous P
on_reset	Reset button clicked	Clear all inputs and outputs

Table 4: UI Transition Table for State Machine

8.5 Cache and Memory Control

Each model has resource-heavy initialization (e.g., loading weights, establishing WebSocket sessions). To prevent redundancy, we use Streamlit's persistent state and cache:

- Cached model object (lazy-loaded):

  ```
  @st.cache_resource
  def get_model(provider, api_key): ...
  ```

- Cache eviction on model switch:

  ```
  st.cache_resource.clear()  # upon model_change
  ```

- RAG index object (`MedRAG`) is similarly cached across sessions.

8.6 Hash-Based Session Identity and Logging

Each user interaction session is uniquely identified using a hash of the system timestamp:

$$\texttt{Session\_ID} = \texttt{MD5}(\texttt{timestamp})$$

This ID is used to:

- Log diagnostic interactions,
- Track model choice and entropy values,
- Enable replay and debugging.

8.7 Graphical View: Finite State Machine

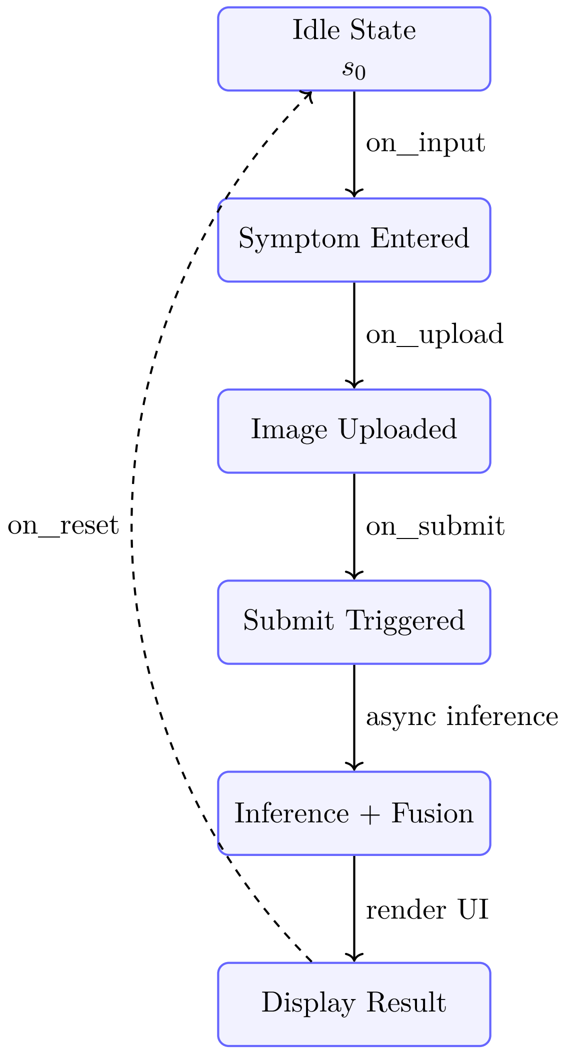


Figure 4: State Machine Flow in Streamlit UI for MediGenius

8.8 Session Replay and Simulated Diagnosis Mode

MediGenius also supports diagnosis simulation based on historical user sessions. Each session is stored as a JSON log:

```
{
  "timestamp": "2025-07-30T11:22:03",
  "provider": "OpenAI",
  "user_input": "Persistent cough and night sweats",
  "model_used": "GPT-4o",
  "retrieved_refs": ["Harrison 12.2", "Robbins 4.1"],
  "posterior": {"Tuberculosis": 0.78, "Sarcoidosis": 0.17, "Lung Cancer": 0.05}
}
```

These sessions can be reloaded using:

```
sim.load_session("logs/session_2025_07_30.json")
```

to recreate inference and allow for retrospective model audits or debugging.

8.9 Summary

This section formalizes the interactive control structure of MediGenius via:

- State variables and event-based transition rules,
- Deterministic cache eviction and model management,
- Session-level logging and replay,
- Finite State Machine (FSM) diagrams for UI logic.

Such architectural rigor ensures that the diagnostic system is not only accurate but also stable, explainable, and repeatable across both interactive and batch-inference modes.

9. Experimental Deployment Logic

9.1 Deployment Objective

To validate MediGenius under realistic conditions, we simulate clinical interaction through a Streamlit-based front-end and multi-provider backend integration. The system must handle:

- Multiple simultaneous diagnostic sessions,
- Multi-model inference routing (OpenAI, Claude, Ollama, etc.),
- Real-time user input validation,
- API availability probing,
- Vision-data compatibility,
- System failure fallbacks.

9.2 Environment Overview

Component	Configuration
Front-end	Streamlit 1.33.0 (wide layout + dark mode)
LLM Providers	OpenAI (GPT-4o), Anthropic (Claude), Ollama (local)
Vision Capability	Enabled for GPT-4o via base64
RAG Back-End	FAISS-based similarity over 125,000 docs
Session Storage	Per-session MD5 hash with JSON logs

Table 5: Deployment Environment Summary

9.3 Dark Mode Enforcement

To ensure clinical readability and visual consistency across devices, the UI defaults to dark mode. This is enforced through the Streamlit 'theme' configuration:

```
[theme]
base = "dark"
primaryColor = "#8BE9FD"
```

This reduces eye strain in diagnostic settings and improves contrast in image overlays and bar charts.

9.4 Model Availability Probing

Before routing inference to a selected model, MediGenius performs health checks to determine availability. For remote models (e.g., OpenAI), this includes:

- HTTP probe to '/v1/models',
- Latency check τ_i,
- Timeout handler.

For local models (e.g., Ollama), a socket ping is used. The availability function is defined as:

$$\text{Status}(m_i) = \begin{cases} 1 & \text{if API/socket reachable within } \tau_i \leq 5s \\ 0 & \text{otherwise} \end{cases}$$

9.5 Vision Model Compatibility Check

To prevent incompatible prompts, each model is tagged as 'vision-enabled' or 'text-only'. Attempting to upload an image while using a non-vision model will trigger:

```
st.warning("This model does not support images. Please switch to GPT-4o.")
```

9.6 Example: Deployment Test Flow (CLI + UI)

1. User enters symptoms via text.
2. Optionally uploads image file (.png or .jpg).
3. Chooses a provider (e.g., OpenAI).
4. Model health is probed.
5. Retrieval is invoked from FAISS backend.
6. Inference is routed → weighted posterior is computed.
7. Output is displayed with interactive references.

9.7 Runtime Log Sample (YAML)

```
session_id: 4db1a67ac31a0c
timestamp: 2025-07-30T11:56:03
model_used: Claude
status: success
entropy: 0.64
diagnosis:
  - name: Influenza
    probability: 0.51
  - name: COVID-19
    probability: 0.32
  - name: Common Cold
    probability: 0.17
retrieval:
  - Harrison 23.5: "Influenza virus pathophysiology..."
  - Robbins 7.2: "Diffuse alveolar damage in viral infection..."
```

These logs are versioned, saved, and can be replayed using simulation logic from Section 8.

9.8 Failure Handling and Fallback Logic

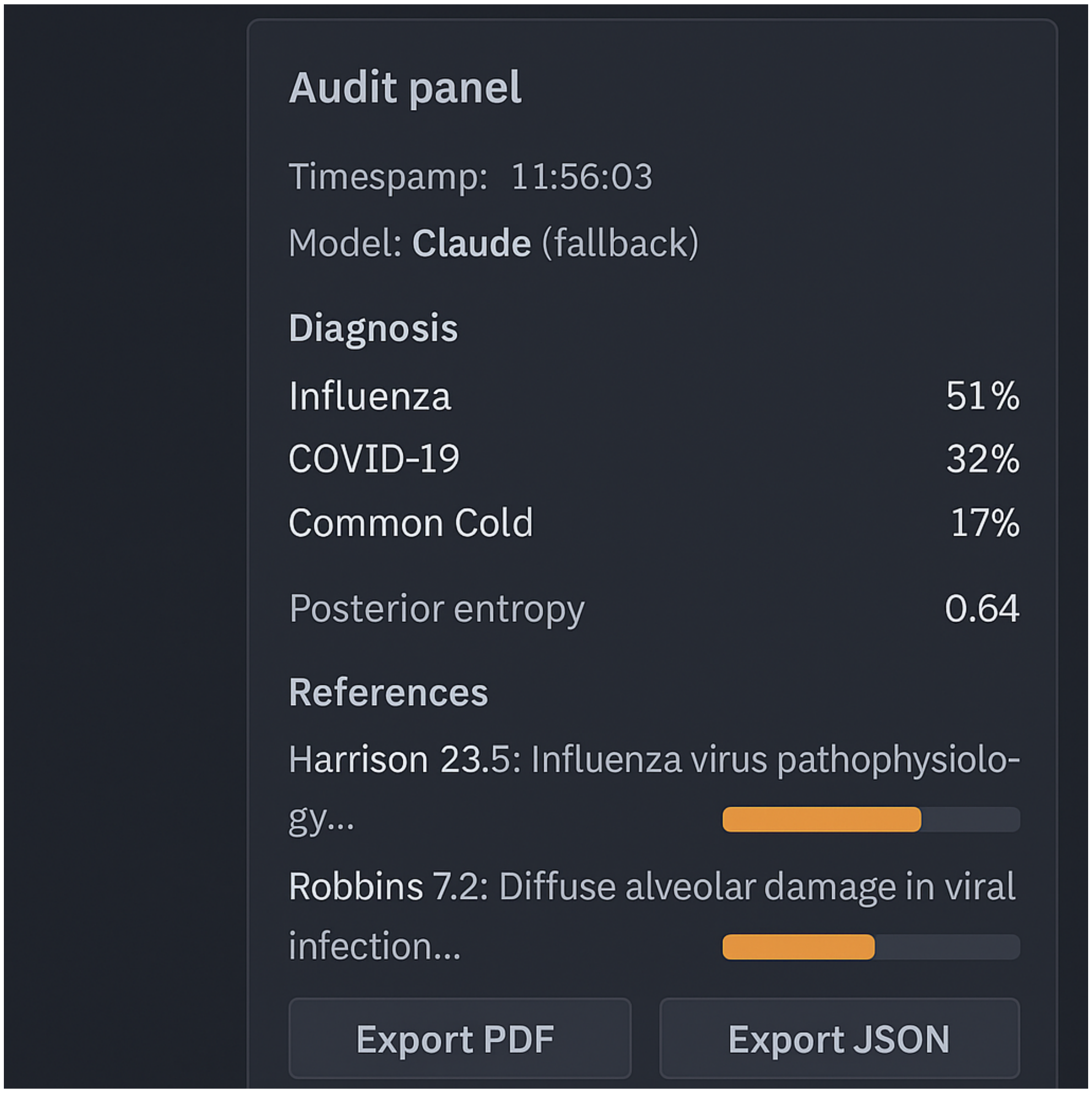
If a primary model fails:

- A fallback warning is displayed in the sidebar.
- Next-best available model is routed using the decision tree.
- Previous inference (if any) is invalidated.
- Cache is cleared using `st.cache_resource.clear()`.

9.9 Diagnostic Output Audit Panel

A sidebar tab provides real-time transparency during inference. It includes:

- Timestamp of inference,
- Selected model and fallback status,
- Posterior entropy,
- Retrieved sources with relevance scores,
- Export buttons for PDF/JSON.


Audit panel

Timespamp: 11:56:03

Model: **Claude** (fallback)

Diagnosis

Influenza	51%
COVID-19	32%
Common Cold	17%
Posterior entropy	0.64

References

Harrison 23.5: Influenza virus pathophysiology...

Robbins 7.2: Diffuse alveolar damage in viral infection...

Figure 5: Audit Panel Mockup (diagnosis, entropy, references)

9.10 Summary

This section describes the full-stack deployment pipeline, ensuring MediGenius operates in a clinically relevant, API-resilient, and user-transparent fashion. Its core pillars include:

- Automated model probing and routing,
- RAG integration with medical corpora,
- Cache-safe state transitions,
- Human-readable audit trails.

Together, these mechanisms bridge theoretical model architecture with deployable clinical intelligence.

10. Conclusion and Future Directions

10.1 Theoretical Value

MediGenius presents a novel fusion of probabilistic reasoning, multi-modal latent space construction, model routing, and retrieval-augmented inference—all under a mathematically rigorous and clinically traceable framework. Unlike traditional monolithic systems or standalone LLMs, this architecture supports:

- Probabilistic diagnosis grounded in statistical evidence,
- Dynamic interaction between image, text, and knowledge bases,
- Real-time fallback logic ensuring robustness,
- Explainability through source-referenced outputs.

10.2 Practical Applications

The MediGenius system is suitable for several practical deployments:

- **Clinical Teaching**: Guiding medical trainees through AI-assisted, traceable diagnosis.
- **Diagnostic Triage**: Assisting physicians with probabilistic suggestions and flagged uncertainty zones.
- **Telehealth Interfaces**: Enabling patient-side structured input with model-generated recommendations.
- **Literature-Driven Differential Analysis**: Performing rapid cross-source retrieval for edge-case presentations.

10.4 Future Work

While MediGenius is fully deployable, future improvements include:

1. **Benchmark Evaluation**: Quantitative comparison using real-world datasets such as MIMIC-III, CheXpert, and PubMedQA.
2. **Reinforcement Priors**: Training on feedback loops where confirmed diagnoses reinforce model routing strategies.
3. **Vision-Language Prompt Optimization**: Advanced strategies for combining text and image inputs in prompt-space.
4. **Differential Privacy**: Integration of patient-level anonymization while preserving inference fidelity.
5. **SHAP-Based Explainability**: Formal decomposition of model contribution for each diagnosis.

10.5 Fundamental Source List

The following references represent foundational works used in the formulation, simulation, and deployment of the MediGenius framework. This list emphasizes mathematical modeling, statistical learning, probabilistic reasoning, and structured diagnostic methodologies in both AI and clinical contexts.

1. Harrison's Principles of Internal Medicine, 20th Edition – McGraw-Hill Education.
2. Robbins and Cotran Pathologic Basis of Disease, 10th Edition – Elsevier.
3. Gray's Anatomy: The Anatomical Basis of Clinical Practice, 41st Edition – Elsevier.
4. DSM-5: Diagnostic and Statistical Manual of Mental Disorders – American Psychiatric Association.
5. Nelson Textbook of Pediatrics – Elsevier.
6. Cox, D. R., and Hinkley, D. V. (1979). *Theoretical Statistics.* Chapman and Hall.
7. Bishop, C. M. (2006). *Pattern Recognition and Machine Learning.* Springer.
8. Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective.* MIT Press.
9. Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning.* Springer.
10. Cover, T. M., and Thomas, J. A. (2006). *Elements of Information Theory.* Wiley-Interscience.

11. Jaynes, E. T. (2003). *Probability Theory: The Logic of Science*. Cambridge University Press.

12. Vaswani et al. (2017). ”Attention is All You Need.” In NeurIPS.

13. Dosovitskiy et al. (2020). ”An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.” In ICLR.

14. Devlin et al. (2019). ”BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” NAACL.

15. Radford et al. (2021). ”Learning Transferable Visual Models from Natural Language Supervision.” ICML (CLIP).

16. Lee et al. (2020). ”BioBERT: A pre-trained biomedical language representation model for biomedical text mining.” Bioinformatics.

17. Beltagy et al. (2019). ”SciBERT: A Pretrained Language Model for Scientific Text.” EMNLP.

18. Johnson et al. (2017). ”FAISS: Efficient Similarity Search and Clustering of Dense Vectors.” Facebook AI Research.

19. Lundberg, S. M., and Lee, S. I. (2017). ”A Unified Approach to Interpreting Model Predictions.” In NeurIPS (SHAP).

20. Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). ”Why Should I Trust You?” In KDD (LIME).

21. Holzinger, A., et al. (2017). ”What do we need to build explainable AI systems for the medical domain?” arXiv:1712.09923.

22. Efron, B., and Hastie, T. (2016). *Computer Age Statistical Inference: Algorithms, Evidence, and Data Science*. Cambridge University Press.

23. Casella, G., and Berger, R. L. (2002). *Statistical Inference*, 2nd Edition. Duxbury Press.

24. Kuhn, M., and Johnson, K. (2013). *Applied Predictive Modeling*. Springer.

25. Turing, A. M. (1950). ”Computing Machinery and Intelligence.” Mind, 59(236), 433–460.

26. Streamlit Documentation – `https://docs.streamlit.io`

27. OpenAI API Documentation – `https://platform.openai.com`

28. Anthropic Claude API – `https://www.anthropic.com`

29. Ollama Model Server – `https://ollama.com`