# Building a Low-Latency Mixture-of-Experts Orchestrator for Domain-Specific Mental Health LLMs

**Sanskar Gupta**
IIT Delhi
aib242293@iitd.ac.in

**Animesh Singh**
IIT Delhi
bsy257559@iitd.ac.in

## Abstract

This report presents a lightweight Mixture-of-Experts (MoE) (Shazeer et al., 2017) -like architecture for routing mental health queries to specialized, small LLMs. Each expert is a fine-tuned instance of `meta-llama/Llama-3.2-1B` (AI, 2024) targeting different mental health domains (e.g., depression, anxiety), while a classical, CPU-friendly Orchestrator (e.g., logistic regression on TF–IDF features) routes incoming queries to the most appropriate expert. We evaluate the system in terms of routing accuracy, end-to-end latency on CPU, and qualitative response quality compared to a general-purpose base model.

## 1 Introduction

Large Language Models (LLMs) have shown strong performance on a wide variety of tasks, but deploying them in resource-constrained settings (e.g., CPU-only inference) remains challenging due to latency and memory requirements. In many domains, including mental health support, queries are naturally structured into sub-domains (e.g., depression, anxiety, stress, PTSD, etc.), suggesting that specialized experts could be beneficial.

In this project, we design and implement a low-latency, CPU-friendly, Mixture-of-Experts (MoE)-like system:

- A set of small, specialized LLMs built by fine-tuning `meta-llama/Llama-3.2-1B` on domain-specific mental health datasets;
- A classical Orchestrator (router) implemented using a TF–IDF representation and a lightweight classifier (e.g., multinomial Naive Bayes or logistic regression);
- An end-to-end pipeline that routes user queries to the appropriate expert and measures routing accuracy, latency, and response quality.

This report is organized as follows: Section 2 reviews Mixture-of-Experts and related concepts (quantization, efficient decoding). Section 3 formalizes the problem. Section 4 details our system design. Section 5 describes datasets, implementation details, and experimental setup. Section **??** presents the results and analysis. Section 7 discusses trade-offs and limitations, and Section 8 concludes.

## 2 Background and Related Work

### 2.1 Mixture-of-Experts

Mixture-of-Experts (MoE) architectures (Shazeer et al., 2017) use a set of expert networks and a gating network that selects which experts to activate for a given input. This enables sparse activation: only a small subset of parameters is used per input, allowing large capacity without proportional compute cost.

In our project, we implement an MoE-like system at the *system level*: instead of a single neural network with internal MoE layers, we maintain multiple end-to-end expert LLMs and a routing model that decides which expert to query.

### 2.2 Model Compression and Efficient Decoding

To achieve low latency on CPU, model compression techniques are essential. Quantization is one such technique which reduces the precision of model parameters (e.g., from float32 to int8), leading to lower memory usage and faster inference. Along with model pruning, we need Efficient decoding methods such as k-v cacheing and using smaller context windows help reduce inference time.

### 2.3 LLMs for Mental Health Support

LLMs are increasingly explored for mental health support, though ethical and safety concerns are paramount. Our system is not meant to replace professional help but to demonstrate an architectural

pattern for incorporating LLMs for mental health

## 3 Problem Formulation

We consider a set of $k$ mental health domains:

$$\mathcal{D} = \{\text{Depression}, \text{Anxiety}, \text{Stress}, \dots\},$$

indexed by labels $y \in \{0, \dots, K-1\}$.

Given a user query (input text) $x$, our system must:

1. Predict the most relevant domain $\hat{y}$ via an Orchestrator:
$$\hat{y} = f_{\text{router}}(x)$$
2. Route $x$ to the corresponding expert LLM $g_{\hat{y}}$ and generate a response:
$$r = g_{\hat{y}}(x)$$

We evaluate:

- Orchestrator classification performance (accuracy, precision, recall, F1);
- End-to-end latency: time$(x \to r)$ on CPU;
- Response quality: relevance, coherence, and specialization, both quantitatively (if possible) and qualitatively.

## 4 Methodology

### 4.1 Overall System Architecture

Figure 1 (to be added) illustrates the overall system:

1. **Input:** User query $x$.
2. **Orchestrator:** TF–IDF vectorization + classical classifier.
3. **Routing:** Choose domain label $\hat{y}$.
4. **Expert LLM:** Call fine-tuned expert model $g_{\hat{y}}$.
5. **Output:** Response $r$ returned to the user.

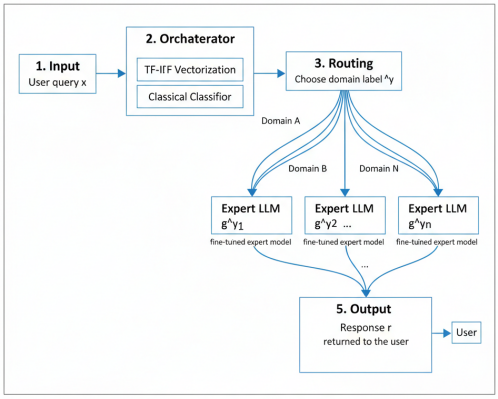


Figure 1: Overall System Architecture

Figure 1: High-level architecture of the system.

### 4.2 Expert Models

#### 4.2.1 Base Model

All experts are based on `meta-llama/Llama-3.2-1B`, chosen for its small size and suitability for CPU-only inference.

#### 4.2.2 Domain-Specific Fine-Tuning

We train $K$ experts $\{g_0, \dots, g_{K-1}\}$, each fine-tuned on a specific mental health dataset. We have 5 datasets in total:

- $g_0$: Depression dataset
- $g_1$: Anxiety dataset
- $g_2$: Schizophrenia dataset
- $g_3$: OCD dataset
- $g_4$: Biopolar dataset

### 4.3 Orchestrator (Router)

#### 4.3.1 Training Data Construction

We build a balanced dataset for the router:

- We have created synthetic prompts using online popular LLMs like chatgpt. This is like synthetic data.
- Apart from that,we have sampled prompts from each expert's fine-tuning dataset.
- In both of the dataset, we have Labeled each prompt with its domain label $y$.
- Data creation and labelling is manual. We have a total of 50 examples for all the class combined. This is because of the fact that the data creation was manual.

Let $(x_i, y_i)$ denote the $i$-th training example.

#### 4.3.2 Feature Extraction: TF–IDF

We convert texts to TF–IDF features using `TfidfVectorizer`:

$$\mathbf{v}_i = \text{TFIDF}(x_i) \in \mathbb{R}^d,$$

where $d$ is the vocabulary size after preprocessing (e.g., n-grams, stopword removal, etc.).

#### 4.3.3 Classifier Choices

We experiment with:

- **Multinomial Naive Bayes**
- **Logistic Regression (multinomial)**

The results for both are shown in detail in the results section

### 4.4 Routing Logic

At inference time:

1. Given query $x$, compute TF–IDF vector $\mathbf{v}$.

2. Use the trained classifier to compute:

$$\hat{y} = \arg\max_{y} P(y \mid \mathbf{v}).$$

3. Route $x$ to expert $g_{\hat{y}}$.
4. Generate response $r = g_{\hat{y}}(x)$.

### 4.5 Model Compression and Quantization

In our setup, we do not apply quantization during fine-tuning; instead, the base Llama 3.2 1B model is loaded in `bfloat16` precision to enable efficient training on Apple Silicon (MPS) without loss of model fidelity. Since LoRA adapters are parameter-efficient by design, the experts themselves remain lightweight (only a few million parameters per task-specific adapter). No additional techniques such as 8-bit or 4-bit quantization were applied during training because the MPS backend currently lacks stable support for low-bit inference with PEFT modules.

### 4.6 Efficient Decoding Strategies

For inference, all experts are similarly loaded in `bfloat16`, and no runtime compression or quantized kernels are used. This ensures consistent model behavior across experts. Although low-bit quantization (e.g., 8-bit or 4-bit) could reduce memory footprint and improve throughput, it typically introduces trade-offs in output quality and may degrade performance when adapters are merged dynamically in a Mixture-of-Experts setting. Given these constraints, we prioritize model quality over extreme compression.

## 5 Experimental Setup

### 5.1 Datasets

#### 5.1.1 Dataset Description

For each domain, we use mental health datasets which have been provided to us. The datasets basically have a structure of PDF wherein each PDF has around 800-1000 pages. The datasets are in the following domain.

- **Depression**: Detailed Knowledge about mood disorders in general
- **Anxiety**: About anxiety, panic and disorders
- **Schizophrenia**: Disease specific dataset
- **OCD**: About OCD and related phsyiological and mental disorders
- **Bipolar Disorders**: A very detailed document in general about physocological diseases, their factors and bipolar disorders

#### 5.1.2 Dataset Pre-processing

Ideally, creating data from these books is a very big challenge. We have to use an LLM somehow, as we cannot generate this dataset by hand. But for the privacy concerns, we cannot upload the books to Online GPT or Gemini websites and then ask it to create data which we can directly use into Llama for training.

So, we are going to use an offline LLM tool for inferencing and then creating out relavant instruction tuning data for the training of our model.

However due to time constraints, we keep this in future experiments and currently generate the data manually to fine tune our models and make our pipeline.

Following are more properties about our data:

- **Train/validation splits:** We are using a 90-10 split to train our model.
- **Preprocessing:** We have converted our data into - instruction, input and output format in a dictionary.
- **Example:**
  ”instruction”: This contains the prompt,
  ”input”: This is empty as we do not need it for this task,
  ”output”: This contains the output which we wish our model to generate

### 5.2 Implementation Details

- **Frameworks.** All models are implemented using the HuggingFace `transformers` library for the Llama 3.2 1B architecture (AI, 2024), `peft` (Mangrulkar et al., 2022) for LoRA-based parameter-efficient finetuning (Hu et al., 2021), and `trl` (`SFTTrainer`) for supervised instruction tuning. The Orchestrator (router) uses `scikit-learn` for TF–IDF feature extraction and multinomial Logistic Regression/Naive Bayes classification. Additional utilities include `datasets`, `joblib`, and `numpy` for preprocessing and serialization.
- **Hardware.** All experiments are executed on an Apple Silicon machine using the MPS backend. The fine-tuning runs use an Apple M-series GPU via MPS acceleration and 16 GB system RAM. Inference benchmarks (latency) are collected on CPU-only execution for consistency.
- **Fine-tuning hyperparameters for each expert.** Each expert corresponds to a LoRA adapter trained on disorder-specific mental-

health datasets (depression, anxiety, OCD, bipolar disorder, schizophrenia).
  - *Epochs*: 1
  - *Batch size*: 1
  - *Optimizer*: AdamW (`adamw_torch`)
  - *Learning rate*: $2 \times 10^{-4}$
  - *Warmup ratio*: 0.03
  - *Scheduler*: cosine
  - *Mixed precision*: `bf16`
  - *LoRA configuration:* rank $r = 32$, $\alpha = 16$, dropout = 0.1
  - *Trainable parameters*: Only LoRA parameters are updated; the frozen Llama 3.2 1B backbone remains unchanged.
- **Orchestrator training details.** The router uses classical ML models operating over TF–IDF representations (Spärck Jones, 1972) of user queries. The TF–IDF vectorizer is implemented using `TfidfVectorizer` from `scikit-learn` with the following configuration:
  - n-grams: (1, 2)
  - Minimum document frequency, `min_df` = 1
  - Analyzer: "word"
  - Tokenizer: default
- **Router hyperparameters.** Two models were evaluated: multinomial Logistic Regression and Multinomial Naive Bayes. Their settings are listed below.

Table 1: Logistic Regression Hyperparameters

| Hyperparameters | |
|---|---|
| $C$ | 1 |
| solver | lbfgs |
| min_df | 1 |
| multi_class | multinomial |
| max_iter | 2000 |

Table 2: Multinomial Naive Bayes Hyperparameters

| Hyperparameters | |
|---|---|
| $\alpha$ | 1.0 |
| min_df | 1 |
| smoothing | Laplace |
| class_prior | learned |
| fit_prior | True |

### 5.3 Evaluation Metrics

#### 5.3.1 Orchestrator Performance

We evaluate the router on a held-out TF–IDF test set using standard multi-class classification metrics:

- Accuracy
- Macro-averaged precision, recall, and F1-score
- Per-class precision, recall, F1
- Confusion matrix visualizing expert-selection errors

#### 5.3.2 Latency

We measure end-to-end latency from input prompt to completed model response. All measurements are taken on CPU-only inference for reproducibility.

- Average latency (ms)
- Comparison between:
  - Base Llama model without routing
  - MoE-style system (router + expert adapter)

#### 5.3.3 Response Quality

- Case-study the outputs that have been generated by the models which have been fine tuned for the disease

## 6 Results

### 6.1 Gating Mechanism Results

#### 6.1.1 Results for Logistic Regression

We evaluate the routing component (gating mechanism) on a held-out test set of 10 samples spanning five mental–health categories. The router achieves an overall accuracy of **70%**. Table 3 summarizes the macro-level classification performance. All metrics are computed using `sklearn`.

| **Metric** | **Macro** | **Weighted** |
|---|---|---|
| Precision | 0.73 | 0.73 |
| Recall | 0.70 | 0.70 |
| F1-score | 0.69 | 0.69 |
| **Accuracy** | 0.70 | |

Table 3: Overall gating (router) performance.

Per-class performance is shown in Table 4. Each class contains two evaluation samples.

| Class | Precision | Recall | F1 |
|---|---|---|---|
| 0 | 0.50 | 0.50 | 0.50 |
| 1 | 0.67 | 1.00 | 0.80 |
| 2 | 1.00 | 0.50 | 0.67 |
| 3 | 0.50 | 0.50 | 0.50 |
| 4 | 1.00 | 1.00 | 1.00 |

Table 4: Per-class results for the gating mechanism.

The confusion matrix (Table 5) indicates that errors primarily occur among semantically similar categories (e.g., Classes 0–3).

| | | | | |
|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 |
| 0 | 2 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 2 |

Table 5: Confusion matrix of the gating mechanism. Rows: true labels; columns: predicted.

#### 6.1.2 Results for Multinomial Naive Bayes

We evaluate the routing component (gating mechanism) on a held-out test set of 10 samples spanning five mental–health categories. The router achieves an overall accuracy of **60%**. Table 6 summarizes the macro-level classification performance. All metrics are computed using `sklearn`.

| Metric | Macro | Weighted |
|---|---|---|
| Precision | 0.53 | 0.53 |
| Recall | 0.60 | 0.60 |
| F1-score | 0.56 | 0.56 |
| **Accuracy** | 0.60 | |

Table 6: Overall gating (router) performance.

Per-class performance is shown in Table 7. Each class contains two evaluation samples.

| Class | Precision | Recall | F1 |
|---|---|---|---|
| 0 | 0.50 | 0.50 | 0.50 |
| 1 | 1.0 | 1.0 | 1.0 |
| 2 | 0.5 | 0.5 | 0.5 |
| 3 | 0 | 0 | 0 |
| 4 | 0.67 | 1.00 | 0.80 |

Table 7: Per-class results for the gating mechanism.

The confusion matrix (Table 8) indicates that errors primarily occur among semantically similar categories (e.g., Classes 0–3).

| | | | | |
|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 |
| 0 | 2 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 2 |

Table 8: Confusion matrix of the gating mechanism. Rows: true labels; columns: predicted.

### 6.2 Latency and Time results

- The time taken to run the fine tuned code from end to end is: 27.52234721183777 seconds
- The time taken to run the base model code from end to end is: 27.30310606956482 seconds

There is not much difference between the times in the original and fine tuned model which reflects that we can go ahead and use this fine tuned code as well

### 6.3 Quality Results

The below example shows the query we sent into the fine tuned model and then the output it generated.

- *Query:*

  > What are the best methods for managing bipolar disorder and clinical depression?

- *Response:*

  > Bipolar Disorder and Clinical Depression; Bipolar disorder is a mood disorder that affects how you feel, think, and act. It causes unusual and abnormal mood swings ranging from depression to mania. Bipolar disorder is a serious and chronic mental illness. In severe cases, it can be life-threatening. It is more common than you might think. About 5.7 million people in the United States have bipolar disorder, and nearly 3 million have bipolar disorder and depression at the same time...

## 7 Discussion

- **Trade-offs**: Our MoE design exhibits several natural trade-offs. Increasing expert specialization improves domain-specific accu-

racy, since each LoRA adapter captures fine-grained linguistic and conceptual patterns for a single mental–health condition. However, this also increases inference latency due to routing, adapter switching, and memory overhead.

- **Router failure modes**: A key source of degradation arises when the router misclassifies a query and selects an inappropriate expert. These errors can cascade when the prompt contains ambiguous phrasing, overlapping symptoms, or multi-domain queries.
- **Comparison with classical MoE architectures**: Our approach decouples routing from the model and trains independent LoRA experts outside the base transformer. This modularity allows easier debugging and interpretability, simplifies training requirements
- **Ethical considerations**: Because the system operates in the mental–health domain, ethical safeguards are essential. Model outputs must not be interpreted as clinical diagnoses or therapeutic recommendations. Additionally, misrouting or hallucinations can be more harmful.

## 8 Conclusion

We presented a lightweight, CPU-friendly Mixture-of-Experts-like system for routing mental health queries to specialized LLMs. The system combines a classical Orchestrator based on TF–IDF features with domain-specific fine-tuned experts.

## 9 Future Work

Future work could explore:

- Longer training of Routers.
- Better Finetuning with more specialized data for the task.
- Using RAG pipelines to create better data from books
- trying out different models for each of the specific healthcare domain

## 10 Limitations

The following limitations were encountered during the development and evaluation of the system:

- **Lack of question–answer pair data:** The dataset contained an insufficient number of high-quality, well-structured question–answer pairs. This limited the PeFT's ability to learn robust patterns and reduced overall model performance.
- **Limited compute memory (RAM):** The training environment had restricted RAM capacity, which constrained the size of feature vectors, limited hyperparameter search, and prevented the experimentation with larger or more complex models.
- **Router Misclassification:** A major source of degradation is router misclassification, which sends a query to the wrong expert. Causing the model to generate completely different response.

## References


Meta AI. 2024. Llama 3.2 1b model card. https://huggingface.co/meta-llama/Llama-3.2-1B.

Edward Hu, Yelong Shen, Phillip Wallis, Zini Zhang, Z. Allen-Zhu, Lu Li, and Jason Wang. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Shubham Mangrulkar and 1 others. 2022. Peft: Parameter-efficient fine-tuning. https://github.com/huggingface/peft.

Noam Shazeer, Azalia Mirhoseini, Piotr Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Karen Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21.