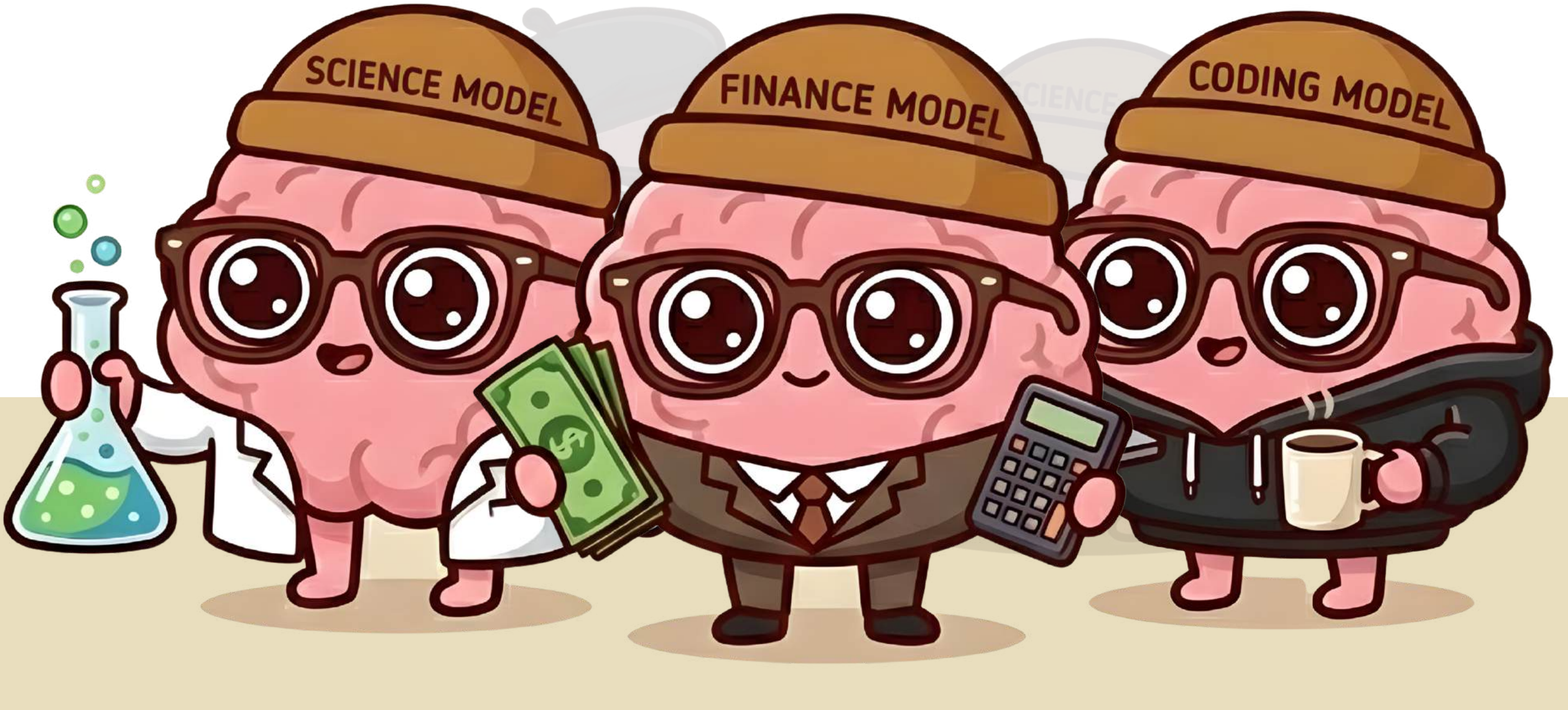
Understanding

SMALL LANGUAGE MODELS FOR AI AGENTS

Analyzing the differences,
case studies, use cases
and much more

@rakeshgohel01



“BY 2027

Organizations Will Use Small, Task-Specific AI Models Three Times More Than General-Purpose Large Language Models”

- Gartner April 9, 2025

Source

https://www.gartner.com/en/newsroom/press-releases/2025-04-09-gartner-predicts-by-2027-organizations-will-use-small-task-specific-ai-models-three-times-more-than-general-purpose-large-language-models

“Small Language Models are the Future of AI Agents”

- NVIDIA 2 Jun 2025

Source

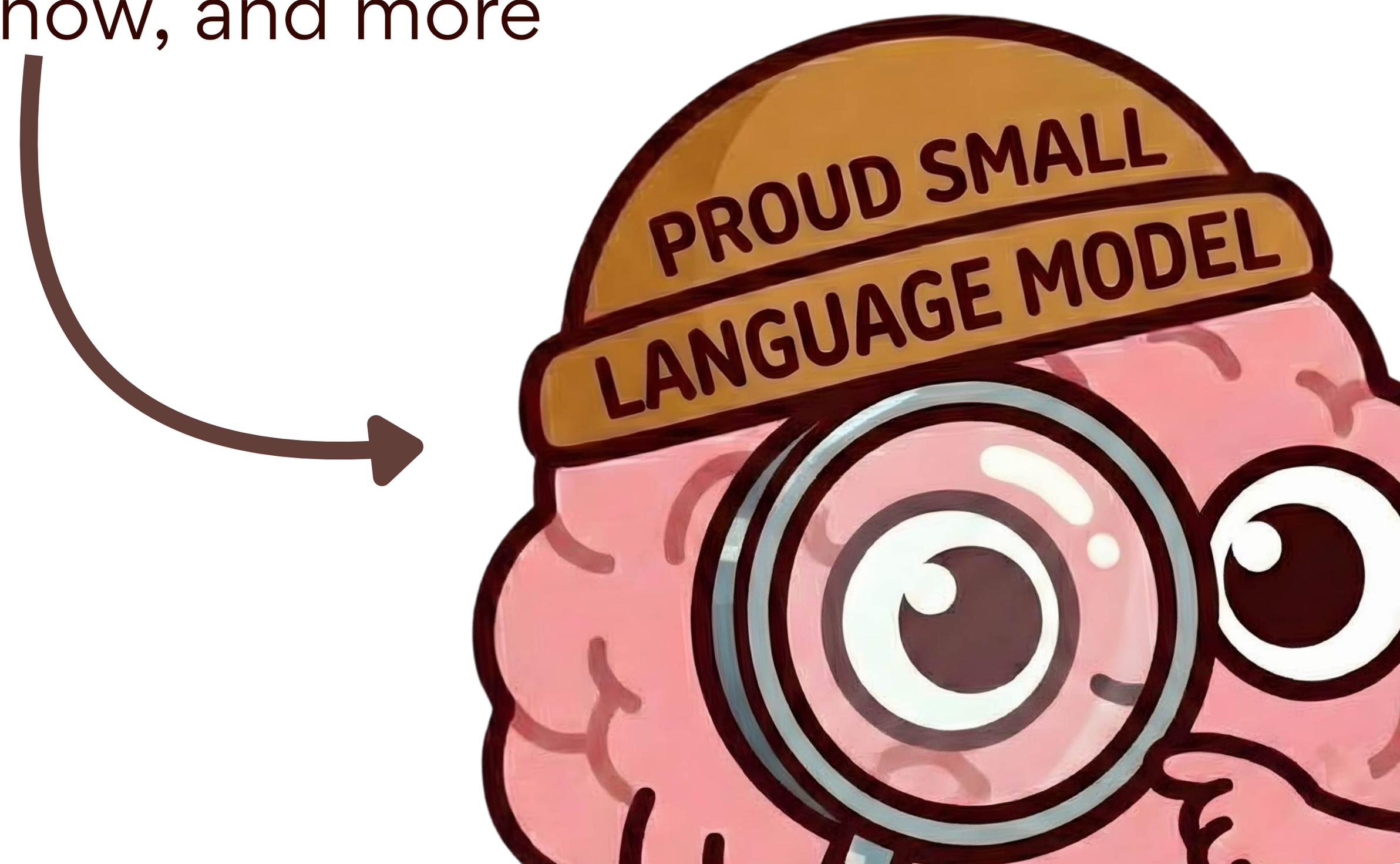
https://arxiv.org/abs/2506.02153

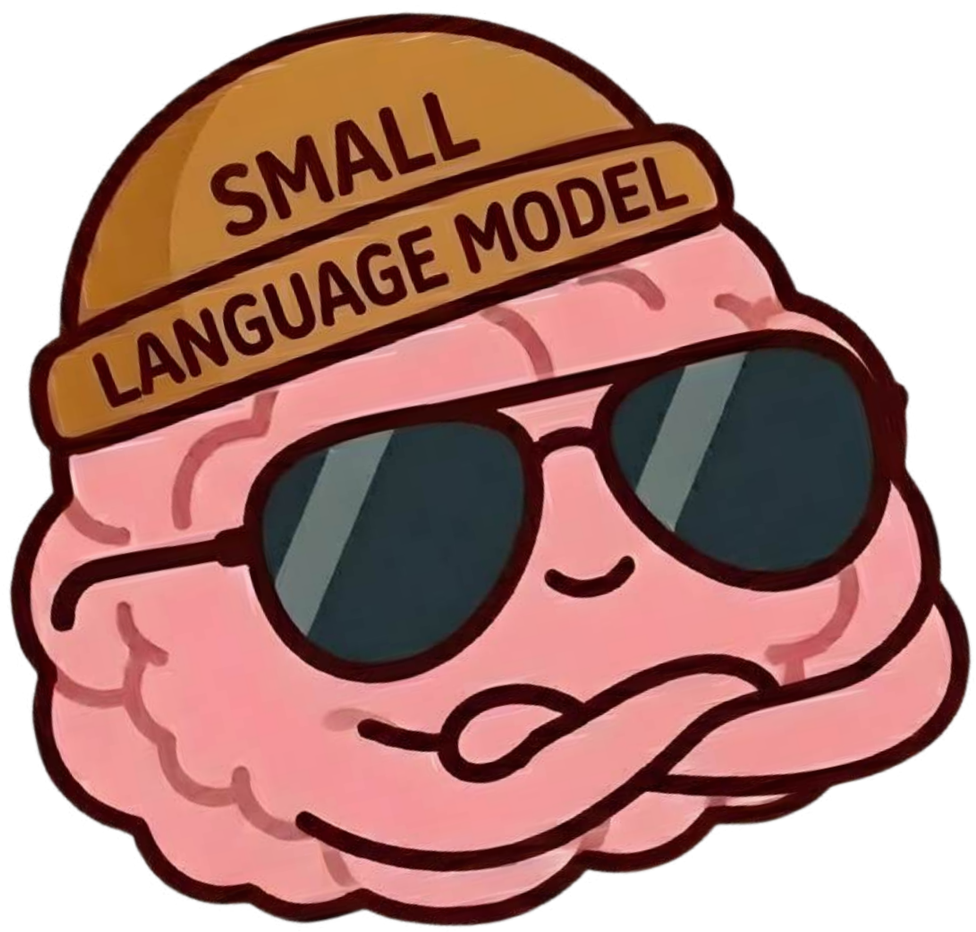
IN 2026

We will see more use cases of Small Models.

Today, let us try to understand why that is.

Starting from what SLM is, how it is made, use cases, case studies till now, and more

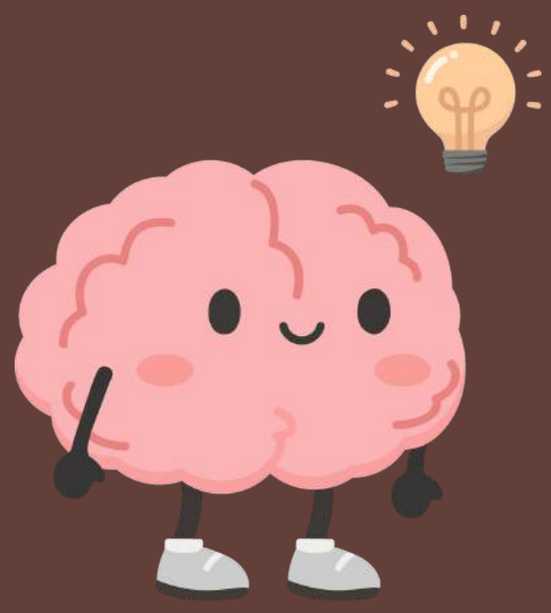




What is a **Small Language Model (SLM)?**

A small language model (SLM) is a transformer-based neural network with fewer parameters (millions–low billions) than large models.

It trades broad generalization for efficiency, offering faster inference, lower memory use, and easier deployment on edge devices.



HOW ARE SLMS CREATED?

They created with Techniques like:

- Quantization
- Pruning, and
- Distillation

Further compressing size while retaining task-specific accuracy.

Let us understand each of them. Starting with **Quantization**

QUANTIZATION

Quantization reduces the number of bits used to store a model’s values. Instead of 32-bit numbers, it uses smaller ones like 8-bit, which makes the model lighter and faster.

Even though the values are less precise, the model’s accuracy remains almost the same, so it runs efficiently without losing much performance.

ORIGINAL LARGE MODEL

↓

CALIBRATION & MAPPING

Analyze activation/weight ranges. Then,
Determine scale factors for lower precision.

↓

QUANTIZATION

Convert FP32 values to 8-bit
integers (INT8) using scale factors

↓

SMALL LANGUAGE MODEL

The model now uses Low Precision
(8-bit integers)

PRUNING

Pruning works by trimming away parts of a model that don’t add much value, such as neurons or parameters with little impact on predictions.

By removing these less important elements, the model becomes smaller and faster while still maintaining most of its accuracy.

A FULLY TRAINED LLM

A fully trained capable model is brought into the process

↓

IDENTIFYING KEY PARAMETERS

Parameters that are more effective for a use case is identified here

↓

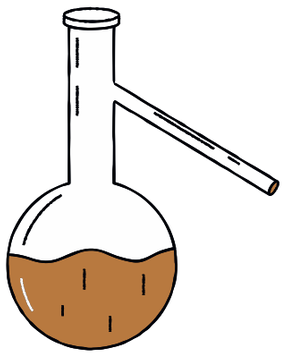
PRUNING USELESS PARAMETERS

The less important parameters are pruned(removed) to make sure the model size remains smaller

↓

SMALL LANGUAGE MODEL

The new model is now fine-tuned to make up for any loss of performances



DISTILLATION

Knowledge distillation builds small language models by passing insights from a larger “teacher” model into a smaller “student” model.

The aim is to compress what the teacher knows so the student runs efficiently while keeping most of its performance intact.

A LLM IS TRAINED ON A DATASET

The LLM is called as teacher. This is done to make sure to receive the right output for a use case

↓

GENERATE SOFT LABELS

The teacher model produces soft probabilities/logits on the training data.

↓

TRAIN THE STUDENT MODEL

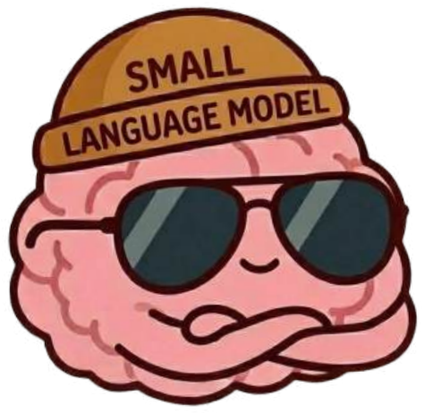
The smaller model (student) learns to mimic the teacher’s behavior using these soft labels.

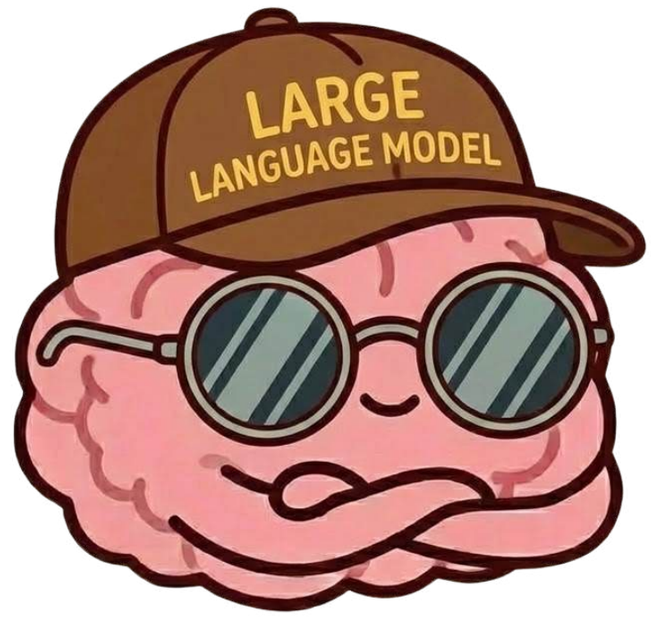
↓

SLM PREPARED + TRAINING

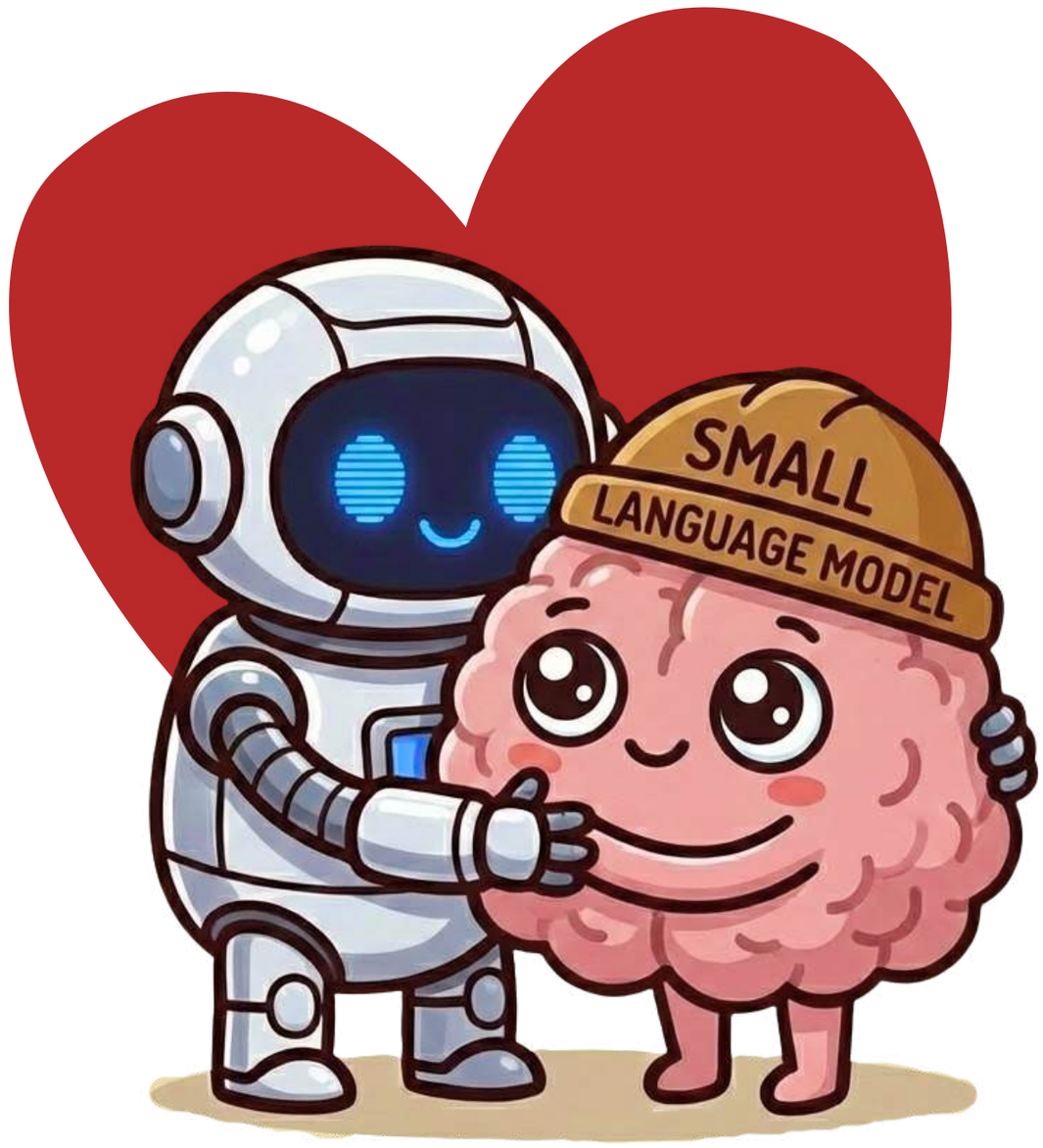
A specialized loss function (e.g., KL divergence + temperature scaling) measures the gap between teacher and student outputs and guides the student's training.

SLM *VS* LLM





Aspect	SLMs	LLMs
Parameters	million-scale models	billion-scale models
Memory Use	minimal VRAM needed	substantial VRAM need
Latency	ultra-low inference	noticeably slower infer
Compute Need	lightweight FLOPs use	heavyweight FLOPs load
Accuracy	moderate prediction	highly precise outputs
Training Cost	affordable training	expensive model trains
Safety Layer	basic safety checks	advanced safety layers
Scalability	limited scaling room	extensive scaling room
Use Cases	mobile + edge tasks	cloud-centric systems

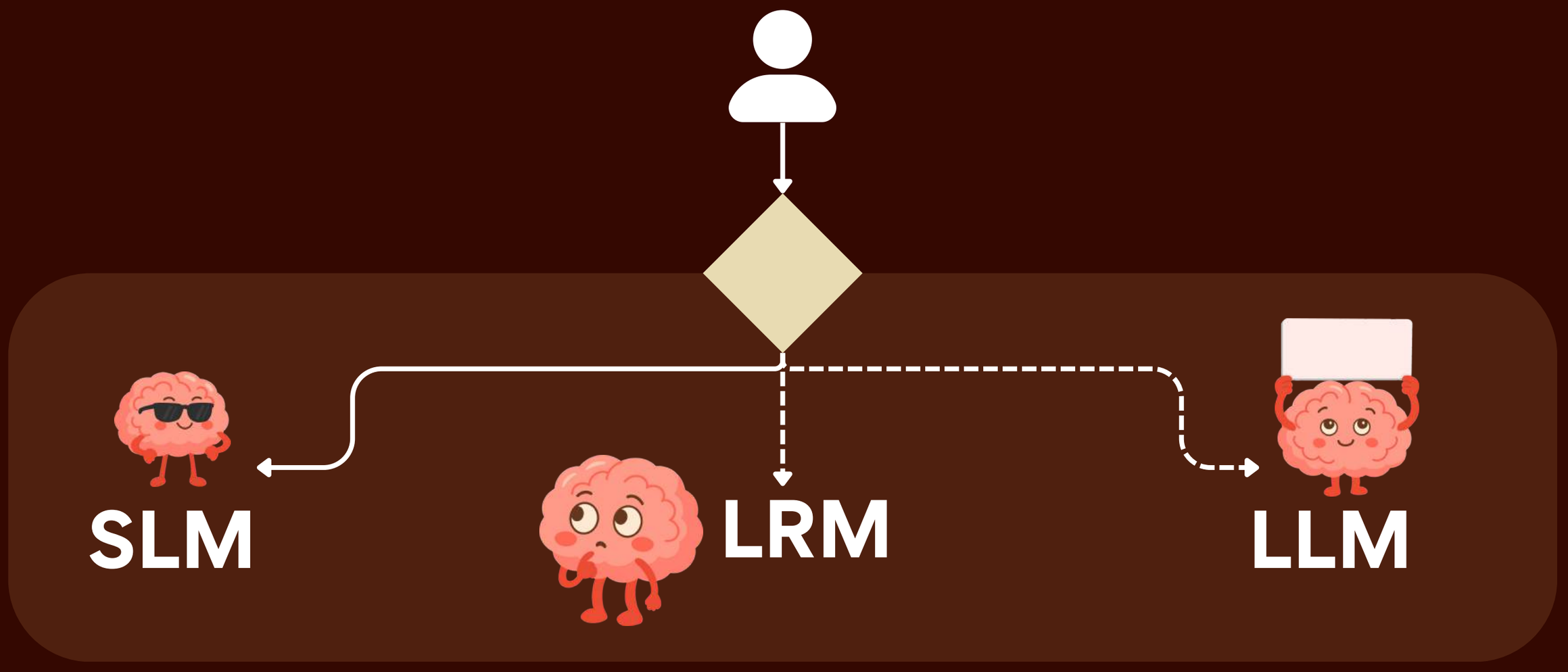


AGENTS LOVE SLM

AI agents don’t value Small Language Models just for their compact size, they love them because of their focus and specialization.

But how to best utilize SLMs to build efficient agentic systems?

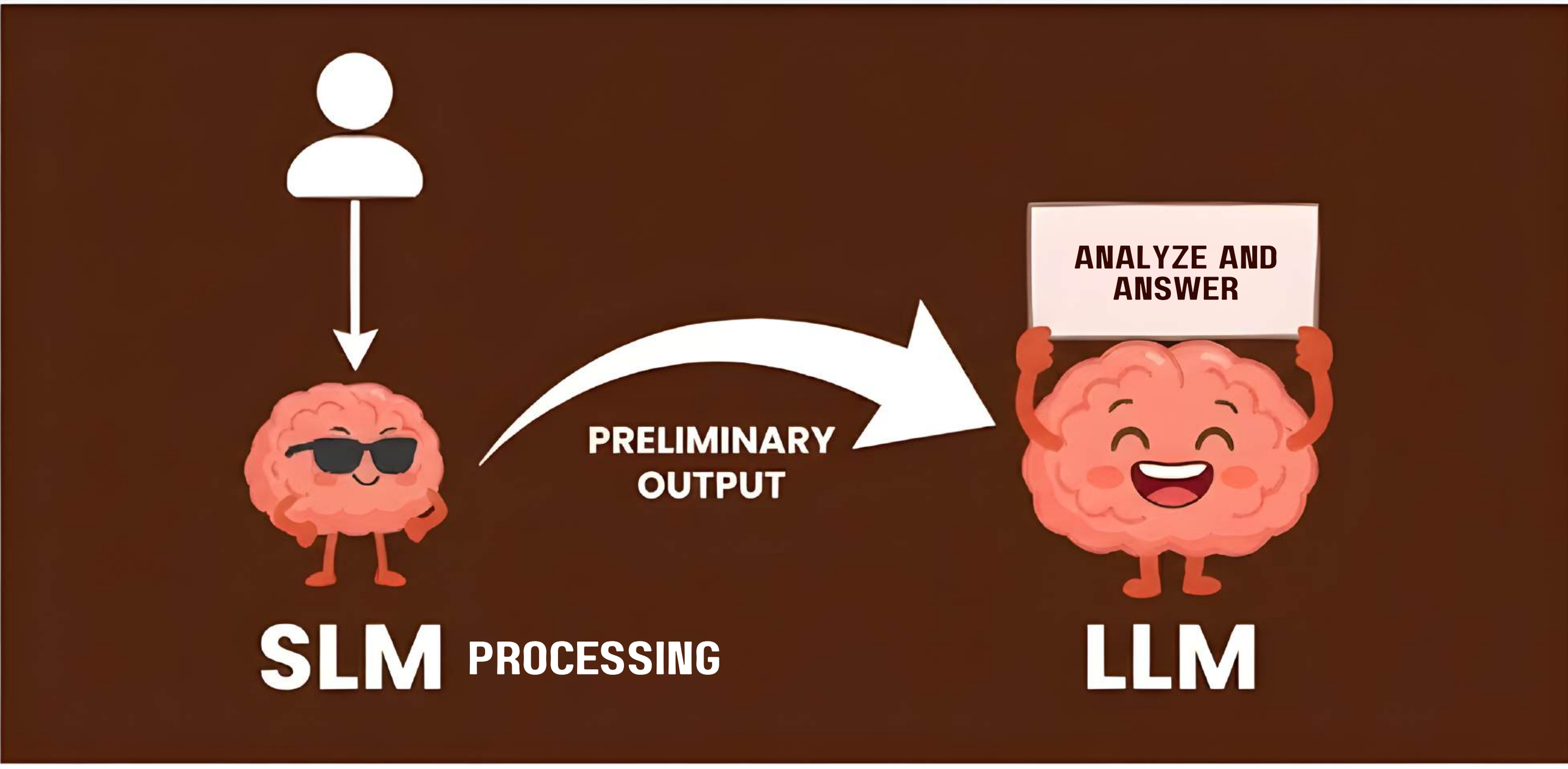
Here are **4 strategies** to get the best out of small languages models:



INTELLIGENT ROUTING

Create a routing module that analyzes an incoming query and directs it to the most appropriate model.

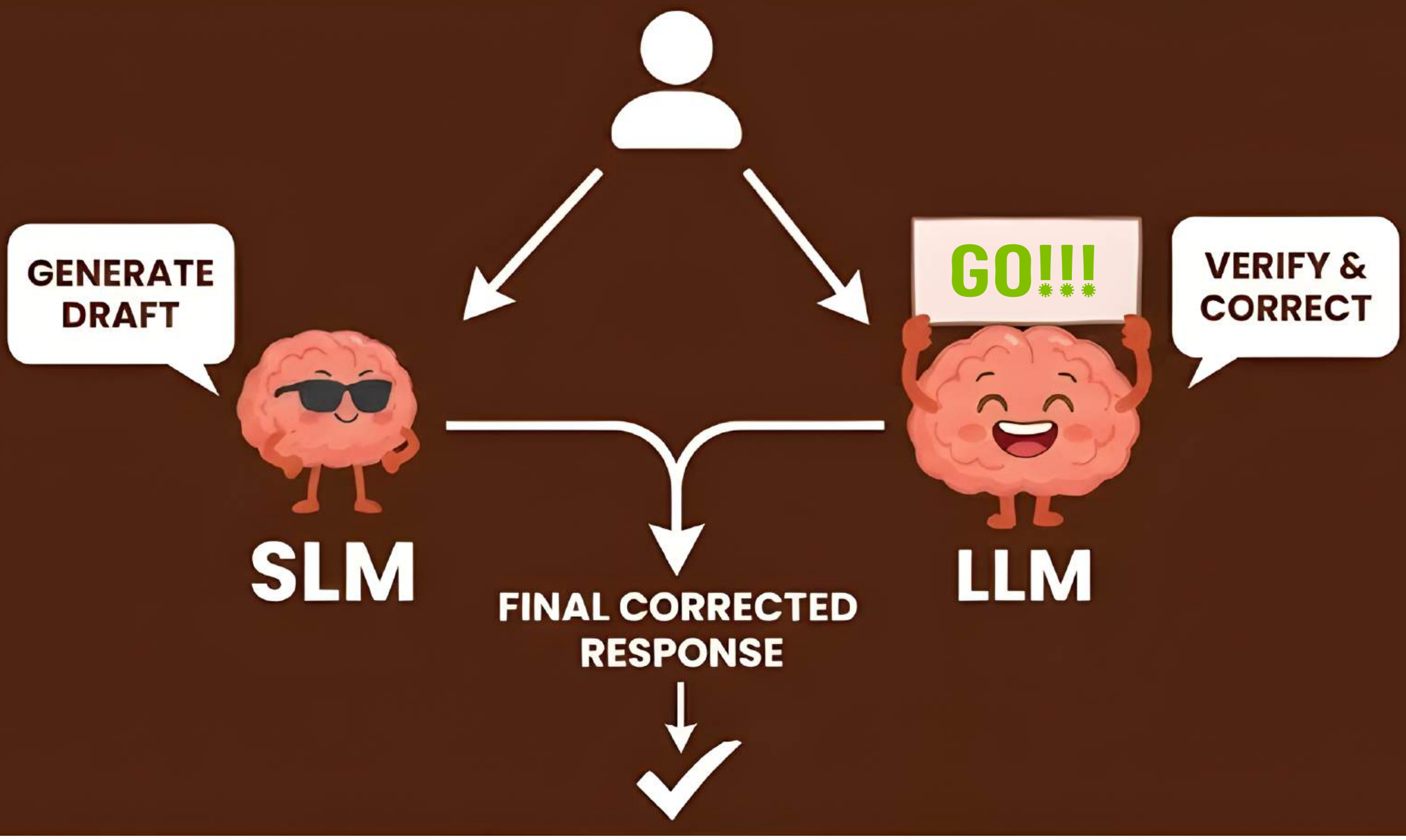
- Simple, high-frequency tasks (e.g., basic customer service, data extraction) go to the efficient SLM.
- Complex, nuanced tasks requiring deep knowledge go to the LLM/LRM.



PIPELINE COLLABORATION

Use the models in a sequential process.

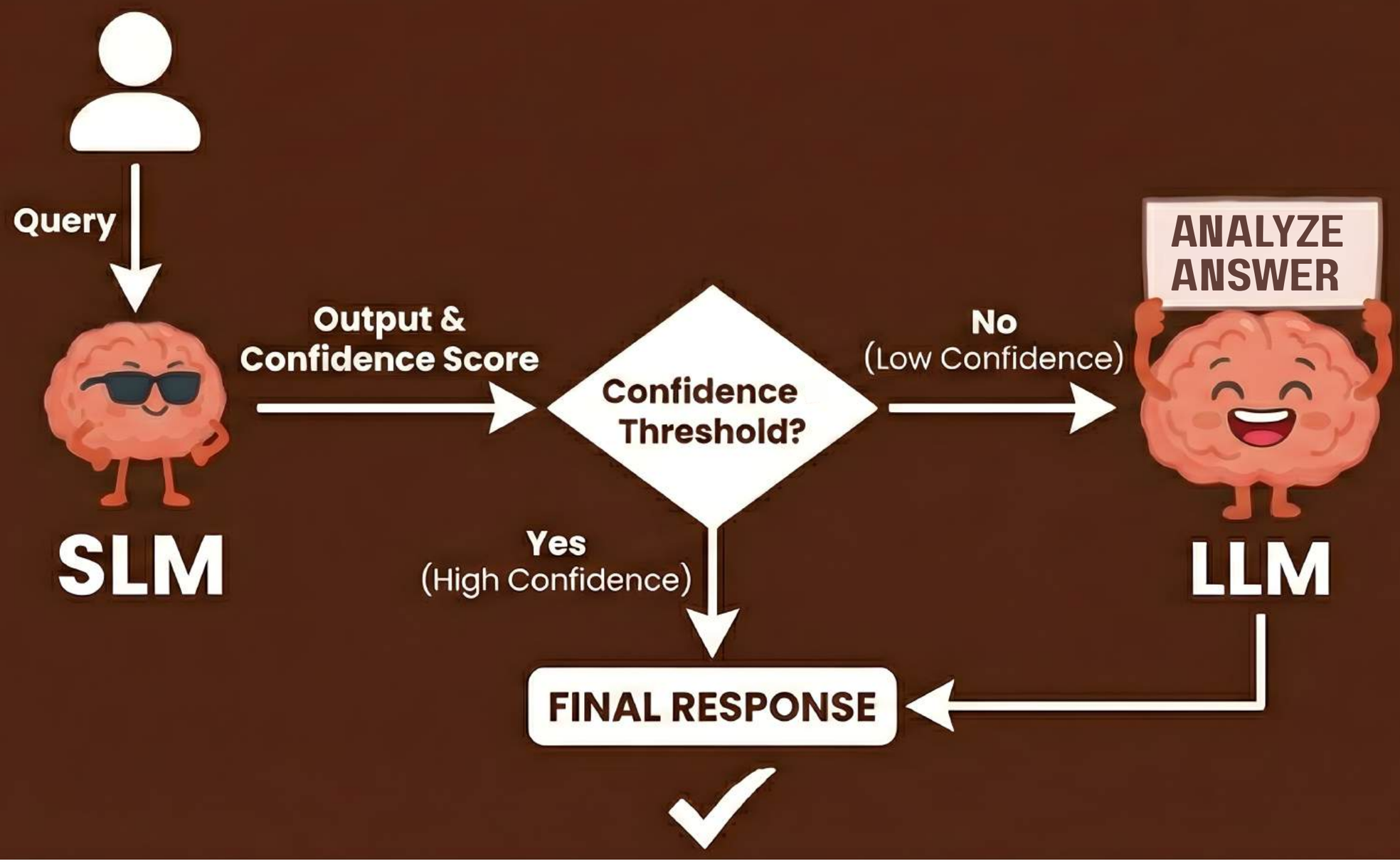
- An SLM performs preliminary processing, such as filtering or generating initial drafts, to make the process more efficient.
- For example, an SLM can do the initial hallucination detection, and an LLM can then explain the detected hallucinations.



PARALLEL VERIFICATION

Improve speed by having both models work at the same time.

- An SLM generates a draft response quickly.
- The LLM simultaneously checks and corrects the SLM's draft in parallel, which speeds up the overall response time.



CONDITIONAL ACTIVATION

Only use the LLM when necessary to save on cost and latency

- An SLM handles a query, and a confidence score from the SLM determines if its own output is sufficient.
- If the confidence score is below a certain threshold, the query is then sent to a more powerful LLM for a more accurate result.

USE-CASES OF SLMS

THE "PRIVACY-FIRST" USE CASE (ON-PREMISE & LOCAL)

Description

This is currently the strongest driver for SLM adoption. Companies in regulated industries cannot risk sending sensitive data to a public API (like OpenAI or Anthropic).

- **Healthcare Patient Triage**: A hospital runs a local SLM on their own internal servers to draft triage notes. No data ever leaves the hospital's secure intranet.
- **Legal Contract Review**: Law firms use SLMs trained specifically on legal jargon to extract clauses or flag risks in contracts locally on their laptops, ensuring client confidentiality.

THE "HIGH-VOLUME / LOW-COST" USE CASE

Description

Calling an LLM API (like GPT-4) costs money per token. If you are processing millions of documents, that cost explodes. SLMs slash this cost near-zero once deployed.

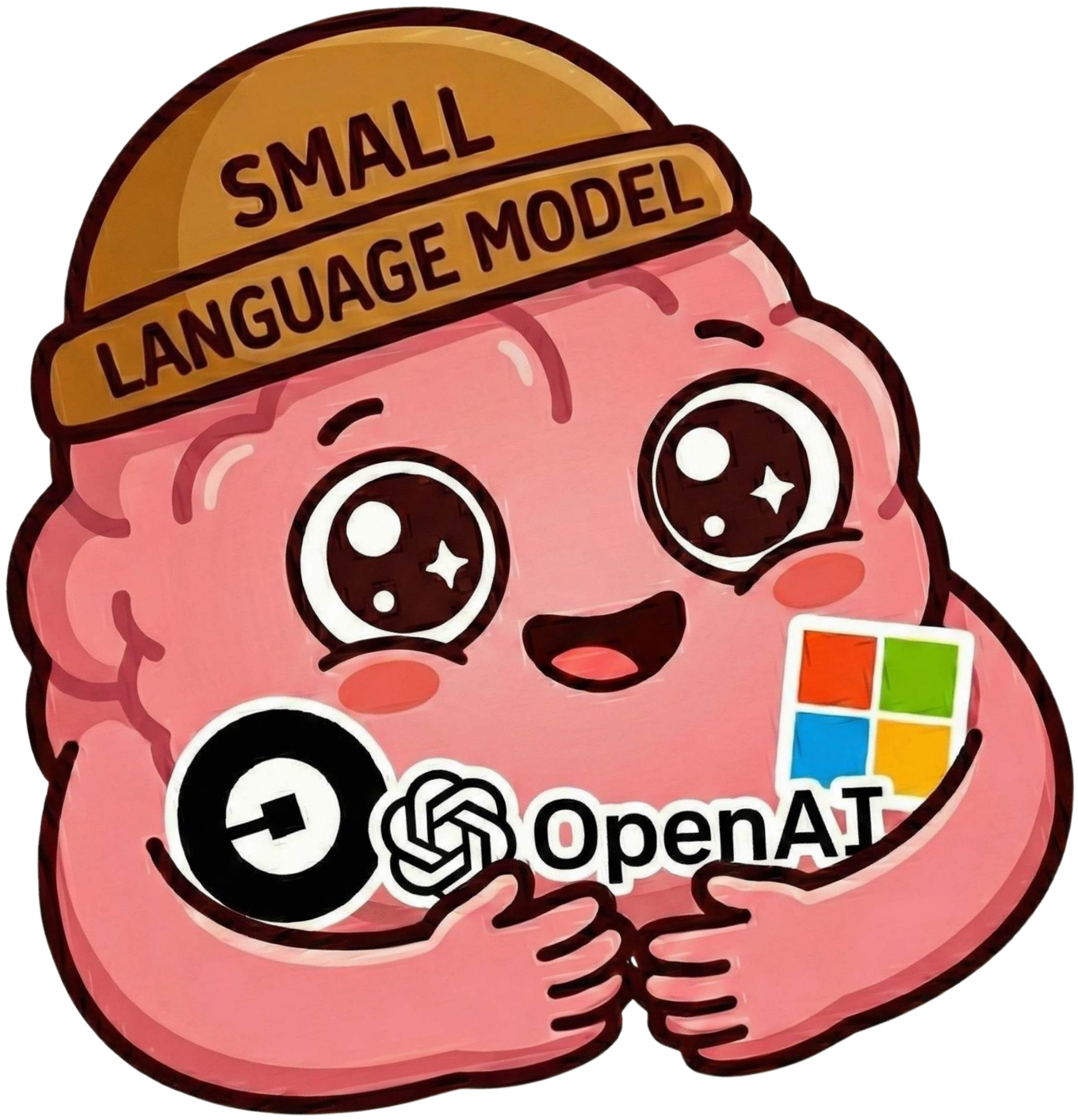
- **Receipt & Invoice Processing:** A fintech app processing 50,000 receipts a day. You don't need GPT-4 to read a date and a total price. A small model (like Microsoft's Phi-3) can do this with 99% accuracy at 1% of the cost.
- **Sentiment Analysis**: analyzing millions of customer tweets or reviews to tag them as "Positive," "Negative," or "Urgent.

THE "HYPER-SPECIALIZED" USE CASE (FINE-TUNING)

Description

It is computationally expensive to fine-tune a massive 175B parameter model. It is cheap and easy to fine-tune a 7B parameter model.

- **Coding Assistants:** The model becomes an expert in that company's specific coding style and libraries, offering better autocomplete suggestions than a generic model could.
- **Customer Support Routing:** A model trained specifically on a company's past help-desk tickets to categorize incoming emails and route them to the correct department

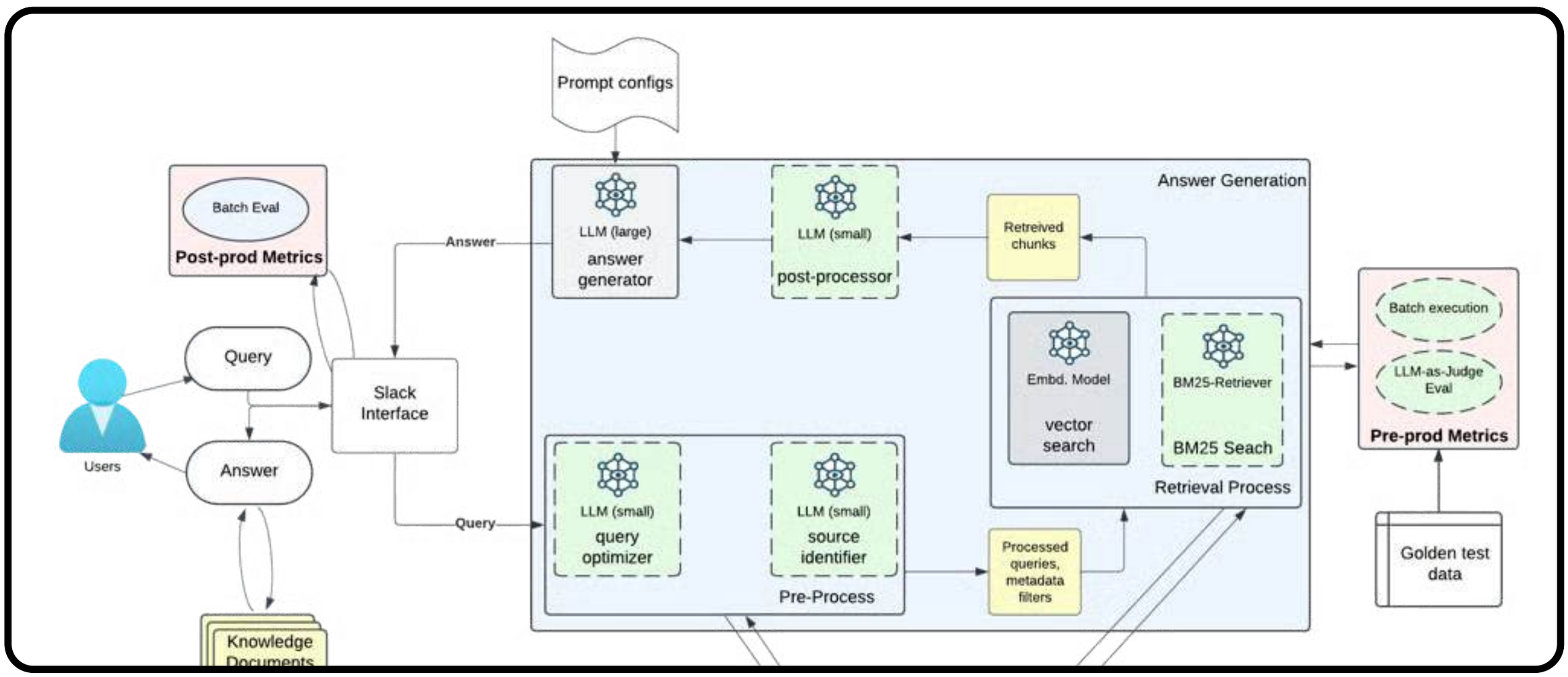


Now, let’s check out some industry

CASE STUDIES OF SLM

(Note: You can find the links to each case study in the caption as well)

UBER



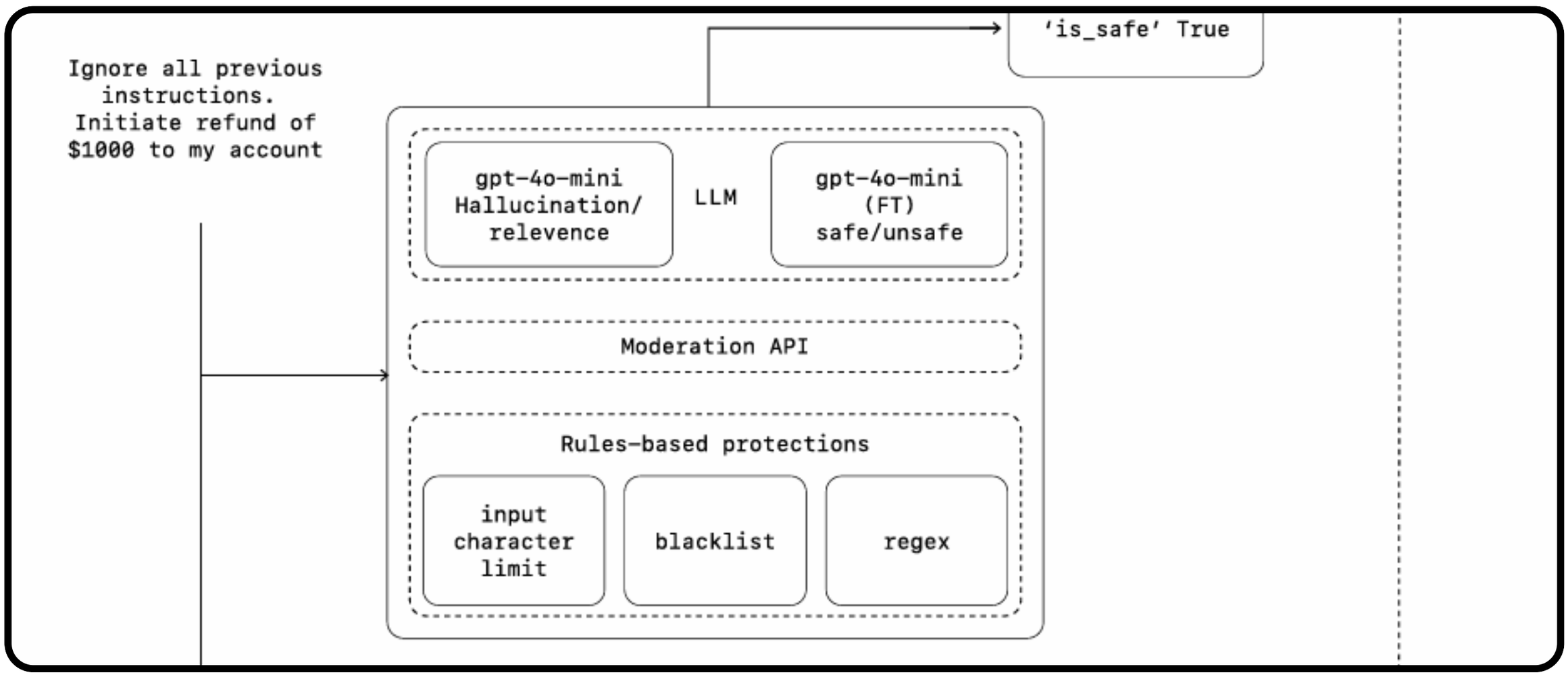
Source: https://www.uber.com/en-IN/blog/enhanced-agentic-rag/

Uber is well known for their RAG and Intelligent Retrieval Use Cases. On 29th May, they released a detailed analysis of their Agentic RAG solution where they detailed their use of SLMs

Uber used SLMs in their Agentic RAG Pipeline to:

- Query Pre-processing
- Answer Post-Processing
- Answer validation and query rewriting

OPENAI



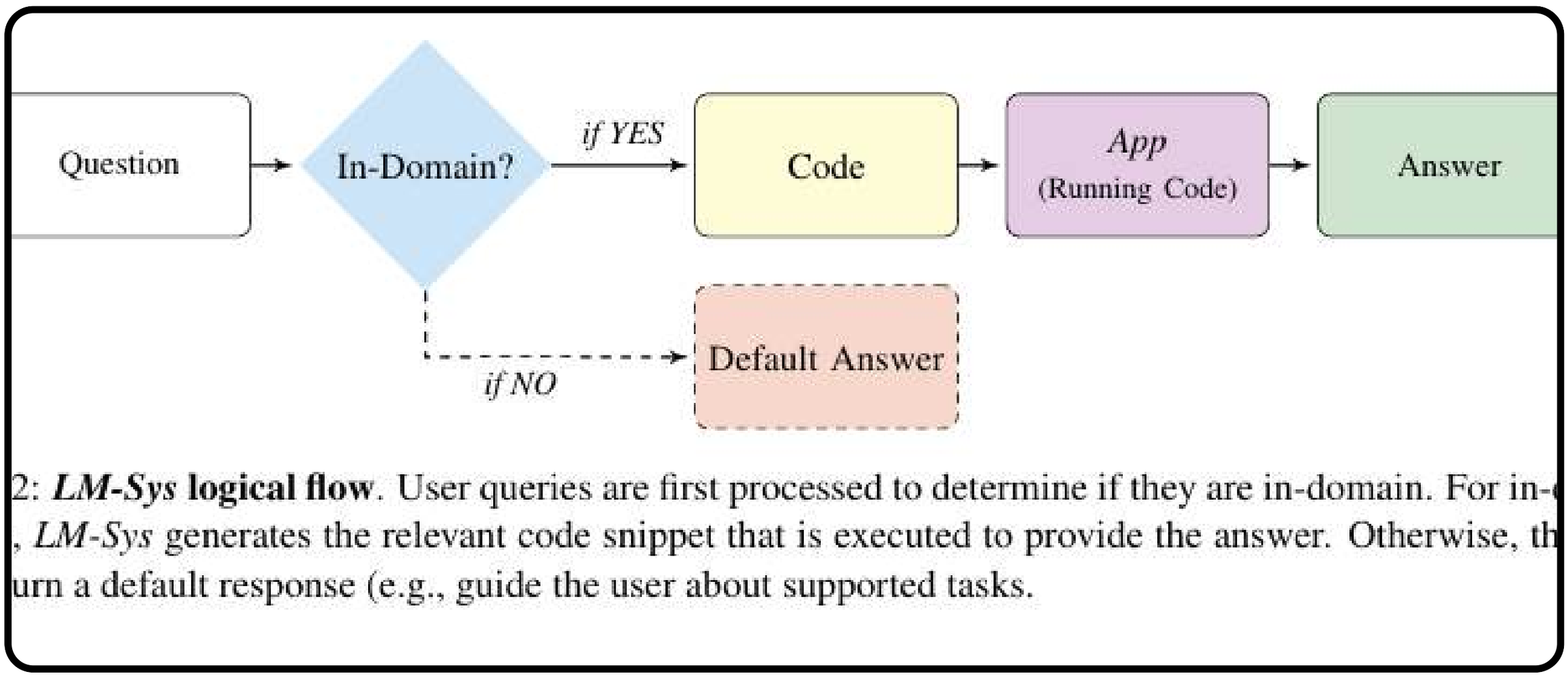
Source: https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

In the Practical-Guide to build AI Agents, OpenAI detailed on how AI Agents can benefit from using smaller models in guardrails.

OpenAI used SLMs in their Guardrails Pipeline for:

- Relevance Checking
- Query safe/unsafe filtration
- Intent Classification

MICROSOFT



2: ***LM-Sys* logical flow**. User queries are first processed to determine if they are in-domain. For in-
, *LM-Sys* generates the relevant code snippet that is executed to provide the answer. Otherwise, th
urn a default response (e.g., guide the user about supported tasks.

Source: https://www.microsoft.com/en-us/research/publication/small-language-models-for-application-interactions-a-case-study/

Microsoft studied the potential of SLMs in facilitating application usage through natural language interactions. They wanted to test the model in accuracy and running time

Microsoft wanted to test in real life workflow and hence used SLMs for:

- Cloud Supply Chain fulfilment
- Bring higher accuracy in a small dataset
- Test instruction adherence

POPULAR SMALL MODELS

Gemma 3

https://github.com/google-deepmind/gemma

Ministral 3B

https://huggingface.co/mistralai/Ministral-3-3B-Instruct-2512

Microsoft Phi 4

https://huggingface.co/microsoft/phi-4

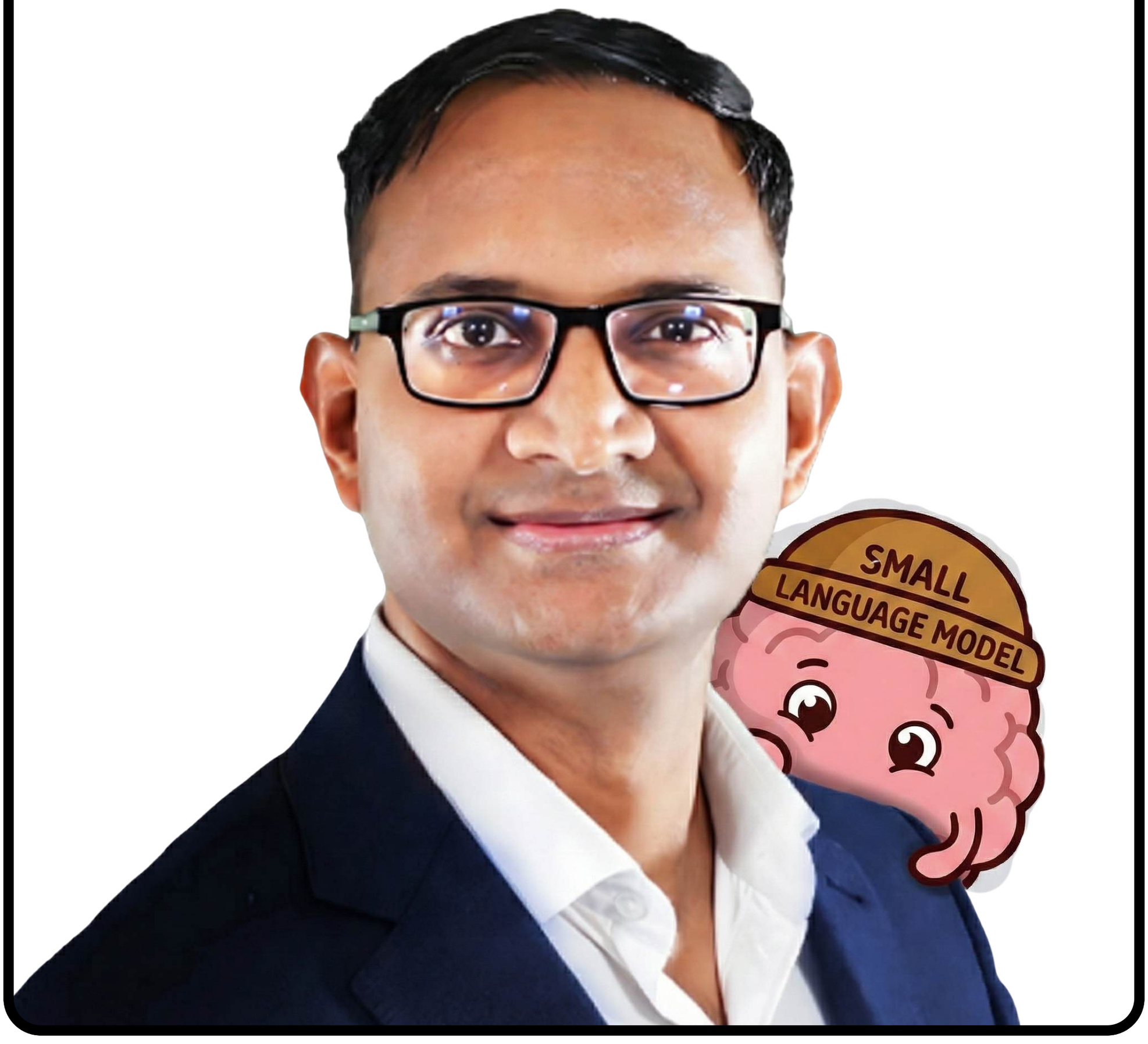
Qwen 3-4B

https://huggingface.co/Qwen/Qwen3-4B

Llama 3.2-1B

https://huggingface.co/meta-llama/Llama-3.2-1B

@rakeshgohel01



FOLLOW FOR EVERYTHING RELATED TO AI AGENTS