Balancing Cost and Quality in FMware

Kirill Vasilevski

Huawei Canada

AIware Leadership Bootcamp 2024

How to cite this session?

```
@misc{VasilevskiBalancingCostAndQualityInFmware,
author = {Kirill Vasilevski and Dayi Lin and Ahmed Hassan},
title = {Balancing Cost and Quality in FMware},
howpublished = {Tutorial presented at the AIware Leadership Bootcamp 2024},
month = {November},
year = {2024},
address = {Toronto, Canada},
note = {Part of the AIware Leadership Bootcamp series.},
url = {https://aiwarebootcamp.io/slides/2024_aiwarebootcamp_vasilevski_balancing_cost_and_quality_in_fmware.pdf }
```

Overview of the session

- **Choosing the correct FM:** What are the concerns of FM selection faced by FMware developers?
  - Decision criteria
  - Cost vs. quality argument
- **Deploying FMs as part of FMware:** Overview of considerations for FM deployment
- **Survey of Existing Methods & Challenges**
  - Model enhancement
  - Synthesis and ensembles
  - Predictive and non-predictive routing
- **RAR: Real-time Adapting Routing**

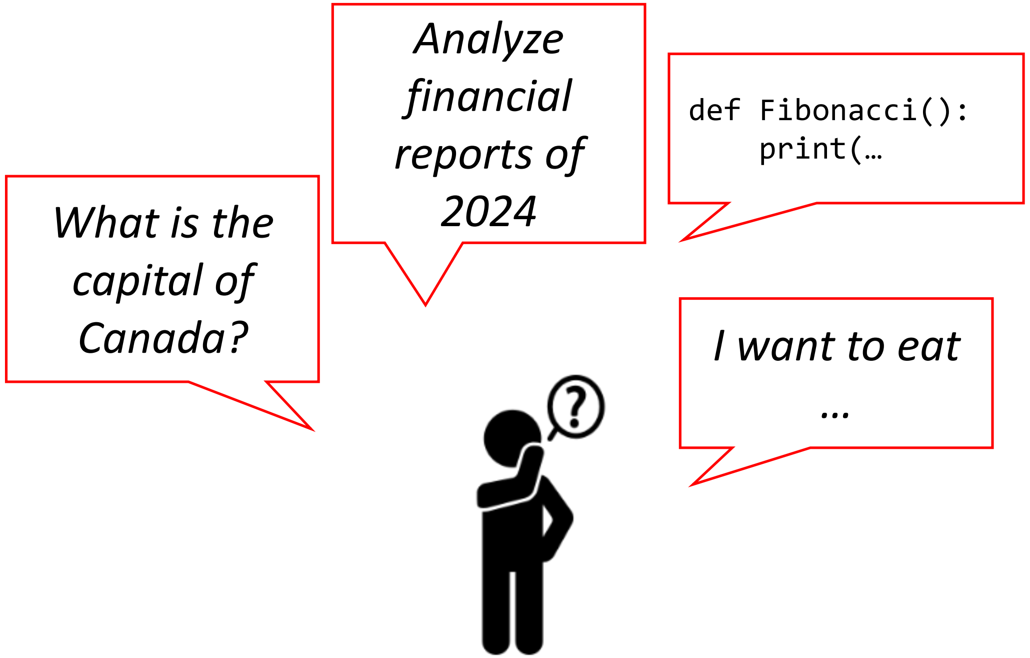
Overview of the session

- **Choosing the correct FM:** What are the concerns of FM selection faced by FMware developers?
  - Decision criteria
  - Cost vs. quality argument
- **Deploying FMs as part of FMware:** Overview of considerations for FM deployment
- **Survey of Existing Methods & Challenges**
  - Model enhancement
  - Synthesis and ensembles
  - Predictive and non-predictive routing
- **RAR: Real-time Adapting Routing**

Choosing the FM

One of the first decisions faced by developers of FMware

- ❑ Over 700,000 LLMs available on HuggingFace repository alone[1]
- ❑ Various levels of capabilities, model sizes, licenses, …



Criteria Examples

1. FM capabilities
   - ❑ *Instruction-tuned* or *text completion*?
   - ❑ *Planning* and *reasoning* abilities?
   - ❑ *Tool use* support?
   - ❑ *Fill-in-the-Middle* capability?
2. Model size
3. License
   - ❑ GPL, AGPL, BSD, Apache, MIT…

[1] https://www.marktechpost.com/2024/06/15/with-700000-large-language-models-llms-on-hugging-face-already-where-is-the-future-of-artificial-intelligence-ai-headed

Choosing the FM

...criteria focusing on *deployment* of FMware:

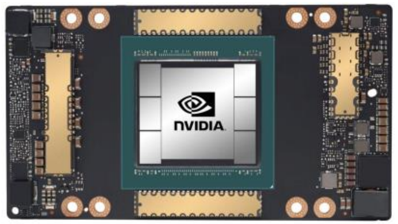
1. **FM capabilities**
2. **Compute costs/limitations** of FM inference

   e.g. smartphone or a cloud datacenter?

iPhone 16 (A18), **35 TOPS**

vs.

NVIDIA A100, **312 TFLOPS**



Dilemma

Generally, larger LMs (e.g. 100+ billions of parameters) generate ***higher quality*** outputs than smaller LMs (e.g. <10 billion parameters)

..however..

Bigger models require magnitudes ***more of expensive compute resources*** for inference operations (and other limitations…)

Source: Llama 3 Herd of Models [2]

Category	Benchmark	Llama 3 8B	Gemma 2 9B	Mistral 7B	Llama 3 70B	Mixtral 8x22B	GPT 3.5 Turbo	Llama 3 405B	Nemotron 4 340B	GPT-4 (0125)	GPT-4o	Claude 3.5 Sonnet
General	MMLU (5-shot)	69.4	**72.3**	61.1	**83.6**	76.9	70.7	87.3	82.6	85.1	89.1	**89.9**
	MMLU (0-shot, CoT)	**73.0**	72.3$^{\triangle}$	60.5	**86.0**	79.9	69.8	88.6	78.7$^{\triangleleft}$	85.4	**88.7**	88.3
	MMLU-Pro (5-shot, CoT)	**48.3**	–	36.9	**66.4**	56.3	49.2	73.3	62.7	64.8	74.0	**77.0**
	IFEval	**80.4**	73.6	57.6	**87.5**	72.7	69.9	**88.6**	85.1	84.3	85.6	88.0
Code	HumanEval (0-shot)	**72.6**	54.3	40.2	**80.5**	75.6	68.0	89.0	73.2	86.6	90.2	**92.0**
	MBPP EvalPlus (0-shot)	**72.8**	71.7	49.5	**86.0**	78.6	82.0	88.6	72.8	83.6	87.8	**90.5**
Math	GSM8K (8-shot, CoT)	**84.5**	76.7	53.2	**95.1**	88.2	81.6	**96.8**	92.3$^{\diamond}$	94.2	96.1	96.4$^{\diamond}$
	MATH (0-shot, CoT)	**51.9**	44.3	13.0	**68.0**	54.1	43.1	73.8	41.1	64.5	**76.6**	71.1
Reasoning	ARC Challenge (0-shot)	83.4	**87.6**	74.2	**94.8**	88.7	83.7	**96.9**	94.6	96.4	96.7	96.7
	GPQA (0-shot, CoT)	32.8	–	28.8	**46.7**	33.3	30.8	51.1	–	41.4	53.6	**59.4**
Tool use	BFCL	**76.1**	–	60.4	84.8	–	**85.9**	88.5	86.5	88.3	80.5	**90.2**
	Nexus	**38.5**	30.0	24.7	**56.7**	48.5	37.2	**58.7**	–	50.3	56.1	45.7
Long context	ZeroSCROLLS/QuALITY	81.0	–	–	90.5	–	–	**95.2**	–	**95.2**	90.5	90.5
	InfiniteBench/En.MC	65.1	–	–	78.2	–	–	**83.4**	–	72.1	82.5	–
	NIH/Multi-needle	98.8	–	–	97.5	–	–	98.1	–	**100.0**	**100.0**	90.8
Multilingual	MGSM (0-shot, CoT)	**68.9**	53.2	29.9	**86.9**	71.1	51.4	**91.6**	–	85.9	90.5	**91.6**

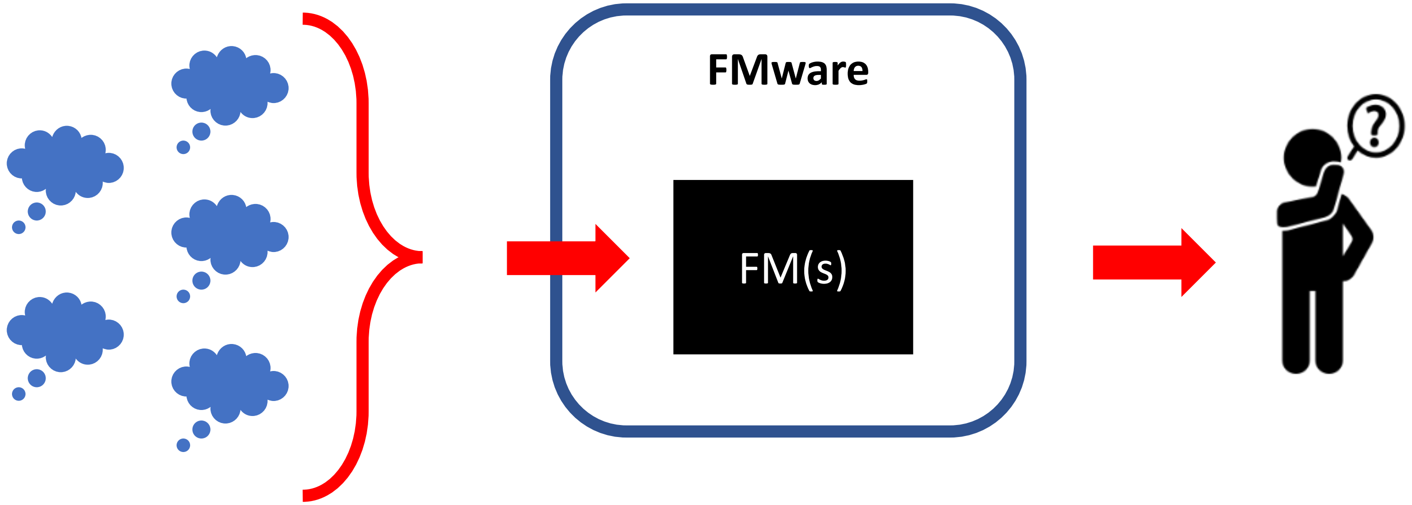
Table 2 Performance of finetuned Llama 3 models on key benchmark evaluations. The table compares the performance of the 8B, 70B, and 405B versions of Llama 3 with that of competing models. We **boldface** the best-performing model in each of three model-size equivalence classes. $^{\triangle}$Results obtained using 5-shot prompting (no CoT). $^{\triangleleft}$Results obtained without CoT. $^{\diamond}$Results obtained using zero-shot prompting.

Overview of the session

- ❑ **Choosing the correct FM:** What are the concerns of FM selection faced by FMware developers?
    - ❑ Decision criteria
    - ❑ Cost vs. quality argument
- ❑ **Deploying FMs as part of FMware:** Overview of considerations for FM deployment
- ❑ **Survey of Existing Methods & Challenges**
    - ❑ Model enhancement
    - ❑ Synthesis and ensembles
    - ❑ Predictive and non-predictive routing
- ❑ **RAR: Real-time Adapting Routing**

Deploying FMs as part of FMware

- From the perspective of FMware, FM itself is a *black box*
  - FMware system only concerned ***whether*** request is served successfully, not *how*



- As long as generated output is of *acceptable quality*, developers can discover ways to optimize their requirements
  - e.g. use a combination of FMs

Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

Overview of the session

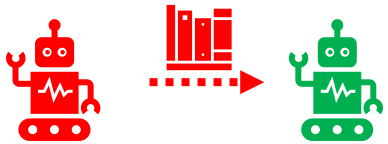
- ❑ **Choosing the correct FM:** What are the concerns of FM selection faced by FMware developers?
  - ❑ Decision criteria
  - ❑ Cost vs. quality argument
- ❑ **Deploying FMs as part of FMware:** Overview of considerations for FM deployment
- ❑ **Survey of Existing Methods & Challenges**
  - ❑ Model enhancement
  - ❑ Synthesis and ensembles
  - ❑ Predictive and non-predictive routing
- ❑ **RAR: Real-time Adapting Routing**

Deploying FM:

Existing Strategies*

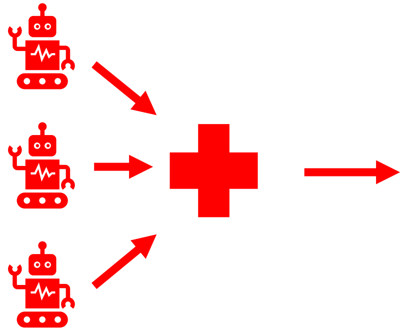
Model Enhancement*

- **Improve capability of a selected FM; model-specific**
  - Fine-tuning (e.g. DPO[3], SFT, RLHF[4])
  - Prompt engineering (e.g. Chain-of-Thought[5], Tree-of-Thoughts[6])



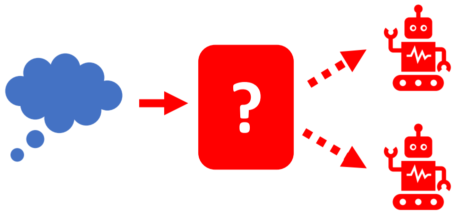
Synthesis

- **Ensemble** of multiple FMs used to generate output
  - e.g. LLM-Blender[7], Blending[8]
- Outputs used to synthesize final output
- Multiple model inference rounds



Routing & Layering

- **Selects appropriate model based on the input query or the generated output**
- Predictive and non-predictive methods
  - e.g. FrugalGPT[9], RouteLLM[10], Tabi[11], Hybrid-LLM[12],…

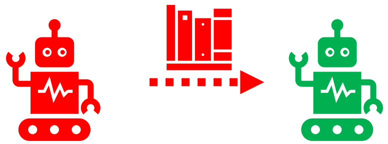


* As described in RouterBench [13]

Deploying FM:
Existing Strategies*

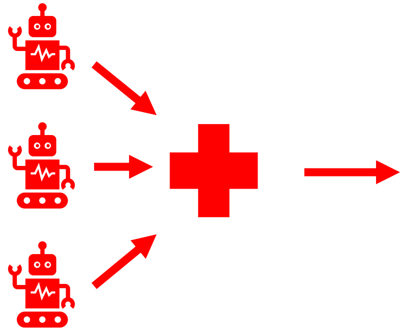
Model Enhancement*

- **Improve capability of a selected FM; model-specific**
  - Fine-tuning (e.g. DPO[3], SFT, RLHF[4])
  - Prompt engineering (e.g. Chain-of-Thought[5], Tree-of-Thoughts[6])



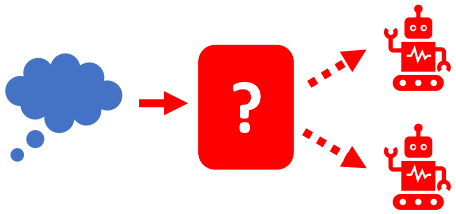
Synthesis

- **Ensemble** of multiple FMs used to generate output
  - e.g. LLM-Blender[7], Blending[8]
- Outputs used to synthesize final output
- Multiple model inference rounds



Routing & Layering

- **Selects appropriate model based on the input query or the generated output**
- Predictive and non-predictive methods
  - e.g. FrugalGPT[9], RouteLLM[10], Tabi[11], Hybrid-LLM[12],…

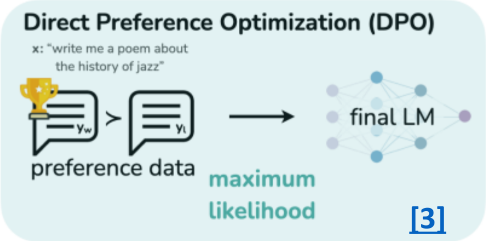


* As described in RouterBench [13]

Model Enhancement Methods

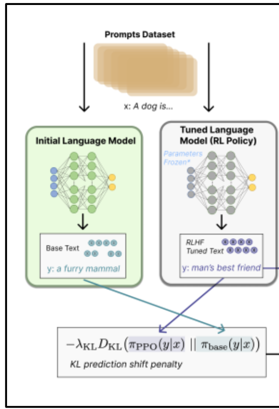
Parameter Optimization Methods

SFT

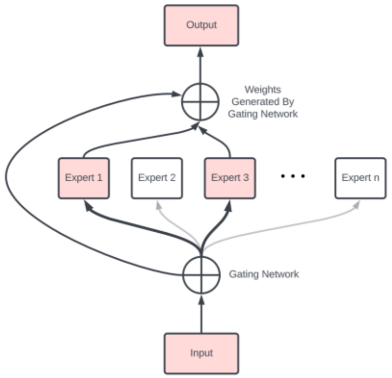


[3]

RLHF [4]



Architecture Methods

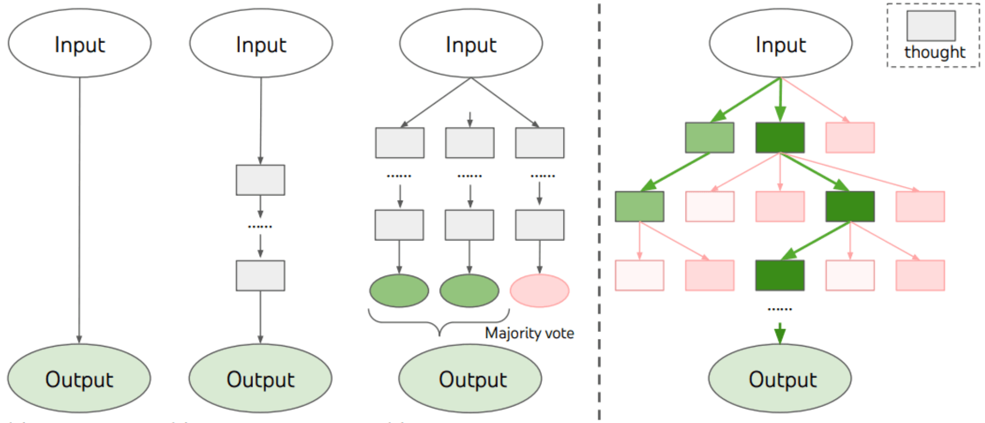


Mixture-of-Experts (MoE) [14]

Routing within the model to the best "expert"

e.g. *Getting MoRE out of Mixture of Language Model Reasoning Experts (Si et. al, 2023)* [15]

Prompt Engineering Methods



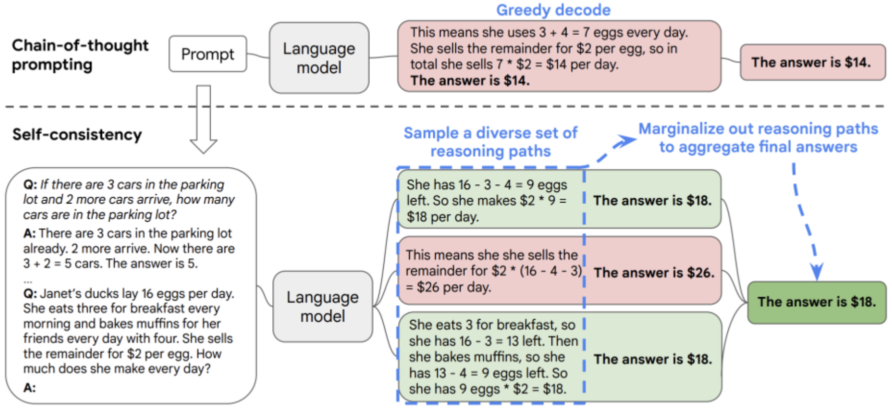
(a) Input-Output Prompting (IO)

(c) Chain of Thought Prompting (CoT)

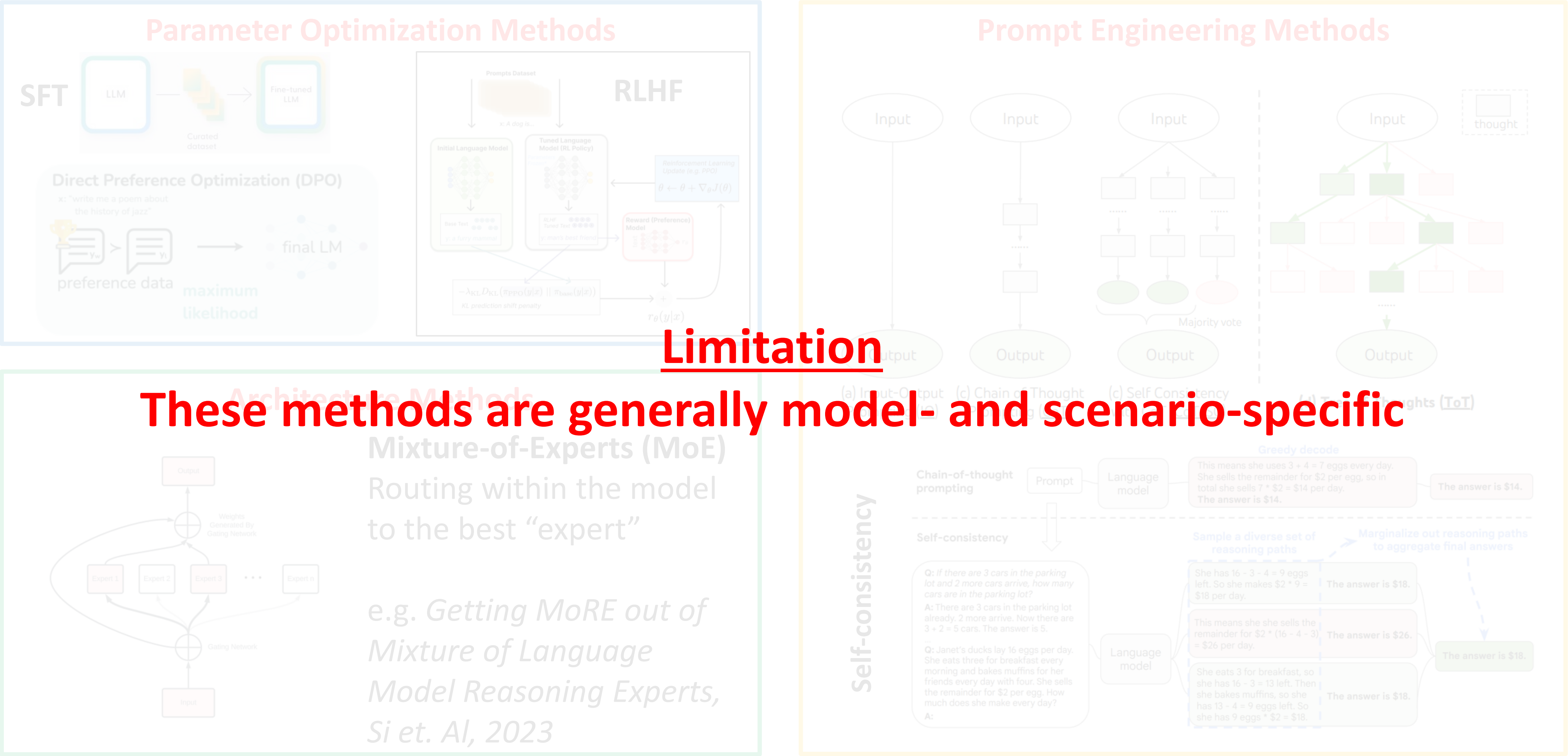
(c) Self Consistency with CoT (CoT-SC)

(d) Tree of Thoughts (ToT) [4]

Self-consistency [16]



Model Enhancement Methods



Limitation

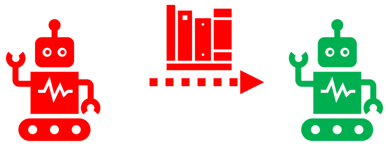
These methods are generally model- and scenario-specific

Deploying FM:

Existing Strategies*

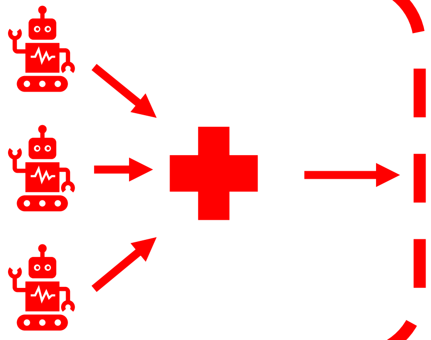
Model Enhancement*

- **Improve capability of a selected FM; model-specific**
  - Fine-tuning (e.g. DPO[3], SFT, RLHF[4])
  - Prompt engineering (e.g. Chain-of-Thought[5], Tree-of-Thoughts[6])



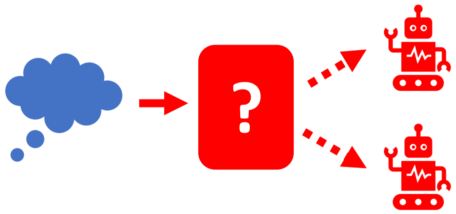
Synthesis

- **Ensemble** of multiple FMs used to generate output
  - e.g. LLM-Blender[7], Blending[8]
- Outputs used to synthesize final output
- Multiple model inference rounds



Routing & Layering

- **Selects appropriate model based on the input query or the generated output**
- Predictive and non-predictive methods
  - e.g. FrugalGPT[9], RouteLLM[10], Tabi[11], Hybrid-LLM[12],…

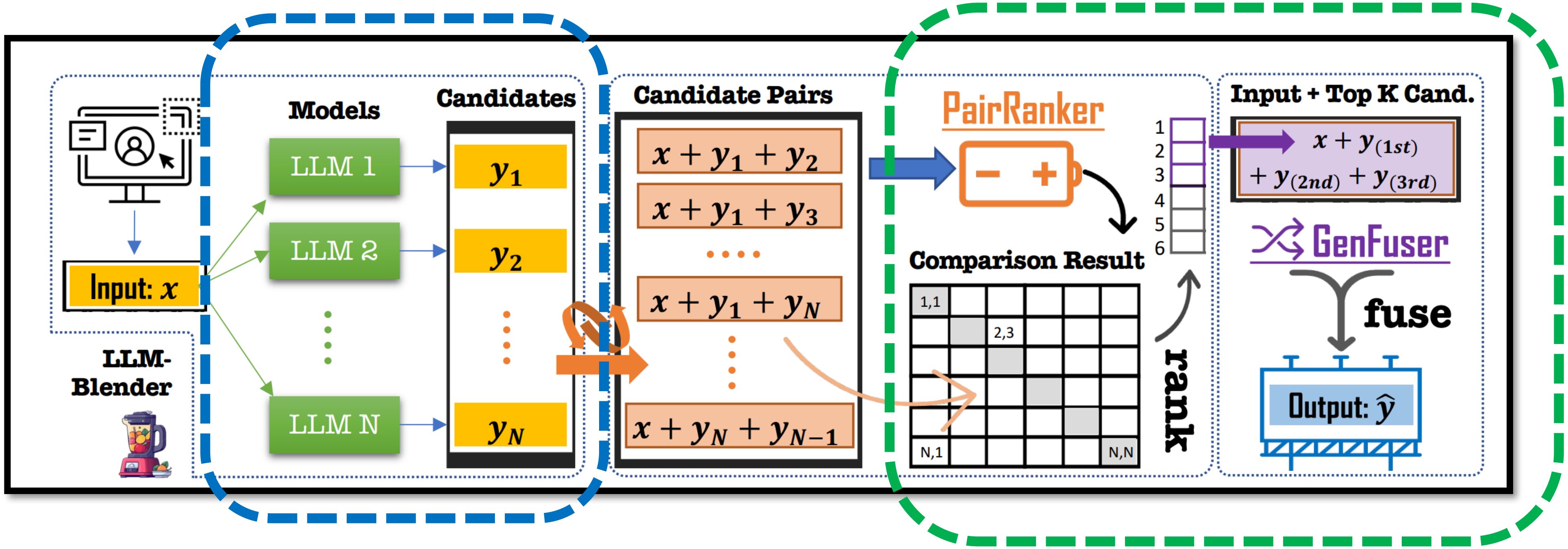


* As described in RouterBench [13]

Synthesis Example:
LLM-Blender[7]

Key Idea:

1. Collects candidate outputs from several FMs
2. Merges top-ranked candidates by combining their strengths



Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

Synthesis Example:
Blending Is All You Need[8]

❑ Key Idea

- Output generated as a combination of outputs from individual FMs

In practice, where we only have access to a finite set of chat AI systems $\{\theta_1, \theta_2 ... \theta_N\}$, one can approximate the continuous integral as a discrete summation. Further, one can assume that $P_\Theta(\theta)$ is distributed uniformly over the systems such that $P_\Theta(\theta_n) = \frac{1}{N}$, which may be a valid assumption if the set consists of similarly performing models. This yields the approximation,

$$
\begin{aligned}
& P(r_k | u_{1:k}, r_{1:k-1}) && (6) \\
& \approx \sum_\theta P_\Theta(\theta) P(r_k | u_{1:k}, r_{1:k-1}; \theta) && (7) \\
& = \frac{1}{N} \sum_{n=1}^{N} P(r_k | u_{1:k}, r_{1:k-1}; \theta_n) && (8)
\end{aligned}
$$

3.3 Blended

The objective of our approach is to approximately draw samples from the true ensemble distribution (equation 8). To achieve this approximation, each turn Blended randomly (and uniformly) selects the chat AI θ that generates the current response. This process is illustrated in Algorithm 1. It can be noted that during a conversation, the response generated by a specific chat AI is conditional on all previous responses generated by the previously selected chat AIs. This means that the different chat AIs are able to implicitly influence the output of the current response. As a result, the current response is a *blending* of individual chat AI strengths, as they *collaborate* to create an overall more engaging conversation.

Algorithm 1 Blended Algorithm

1: $k \leftarrow 1$
2: **while** true **do**
3: $u_k \leftarrow$ user's current input turn
4: Sample model parameter $\theta_n \sim P_\Theta$
5: Generate response r_k according to:

$$r_k \sim P(r | u_{1:k}, r_{1:k-1}; \theta_n)$$

6: $k = k + 1$
7: **end while**

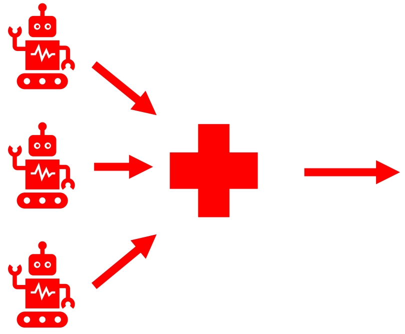
Deploying FM:
Existing Strategies

Model Enhancement*

- **Improve capability of a selected FM; model-specific**
  - Fine-tuning (e.g. DPO, SFT, RLHF)
  - Prompt engineering (e.g. Chain-of-Thought, Tree-of-Thoughts)

Synthesis

- **Ensemble** of multiple FMs used to generate output
  - e.g. LLM-Blender[7], Blending[8]
- Outputs used to synthesize final output
- Multiple model inference rounds



Routing & Layering

- **Selects appropriate model based on the input query or the generated output**
- Predictive and non-predictive methods
  - e.g. FrugalGPT, RouteLLM, Tabi, Hybrid-LLM,...

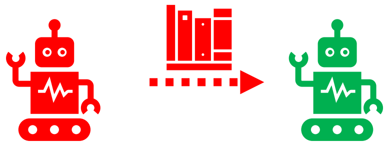
Limitations

- **Increased latency and costs since at least two steps (generation and synthesis) required**
- **Often require multiple FM inference rounds**

Deploying FM:
Existing Strategies*

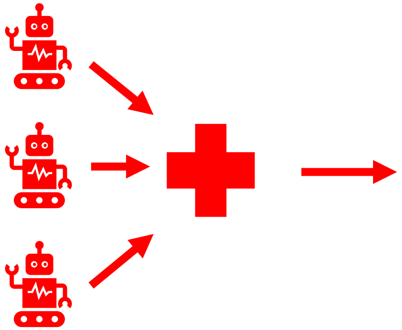
Model Enhancement*

- **Improve capability of a selected FM; model-specific**
  - Fine-tuning (e.g. DPO[3], SFT, RLHF[4])
  - Prompt engineering (e.g. Chain-of-Thought[5], Tree-of-Thoughts[6])



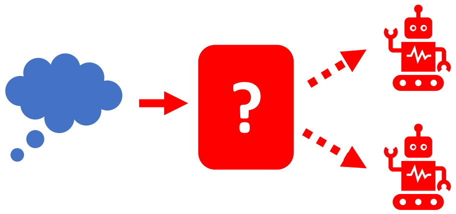
Synthesis

- **Ensemble** of multiple FMs used to generate output
  - e.g. LLM-Blender[7], Blending[8]
- Outputs used to synthesize final output
- Multiple model inference rounds



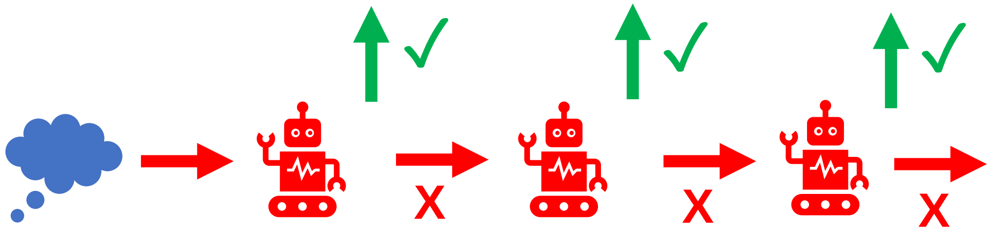
Routing & Layering

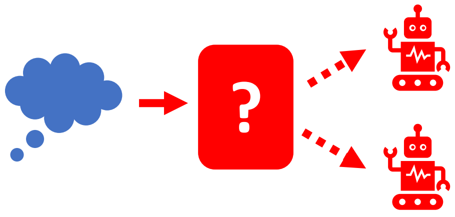
- **Selects appropriate model based on the input query or the generated output**
- Predictive and non-predictive methods
  - e.g. FrugalGPT[9], RouteLLM[10], Tabi[11], Hybrid-LLM[12],…



* As described in RouterBench [13]

Deploying FM:
Types of Routing/Layering*





Non-predictive Routing

- Based on **collecting FM-generated outputs** from multiple FMs
- Sequential collection of outputs continues until an answer passes a quality threshold.
- Increased cost and latency due to many rounds of inference.

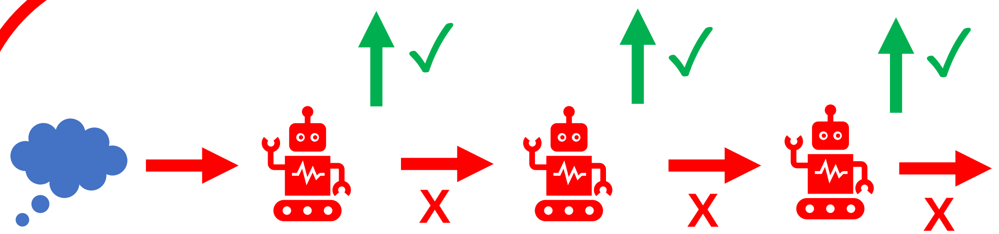
Predictive Routing

- Based on the **contents of the input request**; no model inference required.
- Training prediction models (e.g. classifiers) using a dataset of input requests and associated human model preference labels.
- Performance is often **limited by** the quality and generalizability of the training **dataset**.
- Capabilities are **static** in post-deployment.

* As described in RouterBench [13]

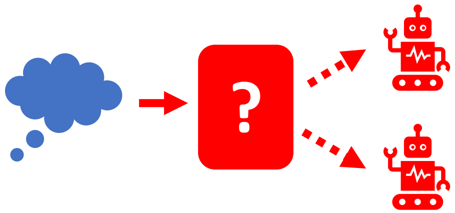
Deploying FM:

Types of Routing/Layering*



Non-predictive Routing

- Based on **collecting FM-generated outputs** from multiple FMs
- Sequential collection of outputs continues until an answer passes a quality threshold.
- Increased cost and latency due to many rounds of inference.



Predictive Routing

- Based on the **contents of the input request**; no model inference required.
- Training prediction models (e.g. classifiers) using a dataset of input requests and associated human model preference labels.
- Performance is often **limited by** the quality and generalizability of the training **dataset**.
- Capabilities are **static** in post-deployment.

* As described in RouterBench [13]

Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

Non-predictive Routing:

Tabi[11]

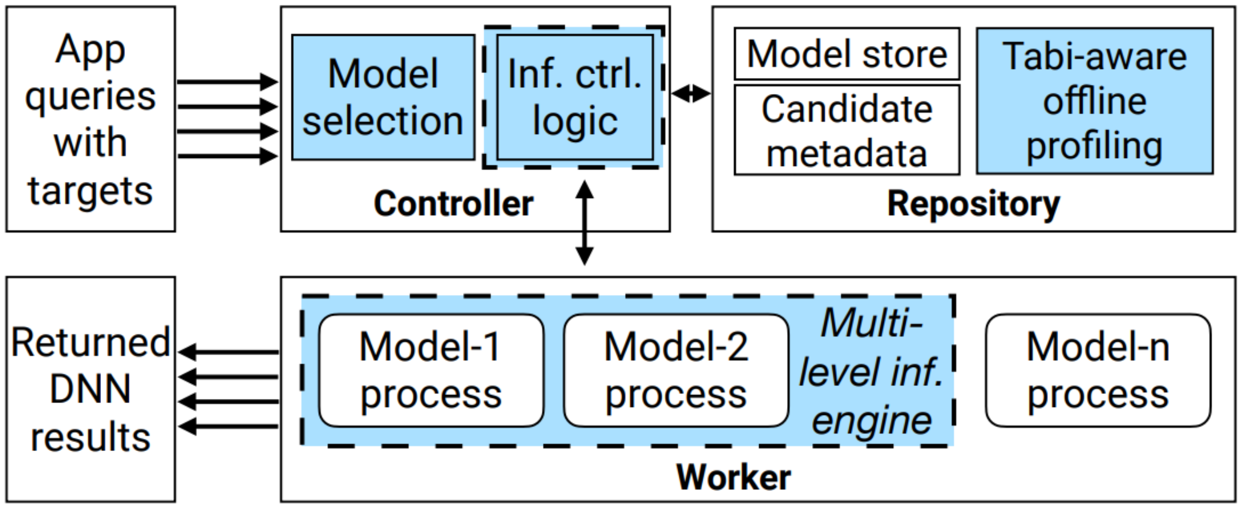
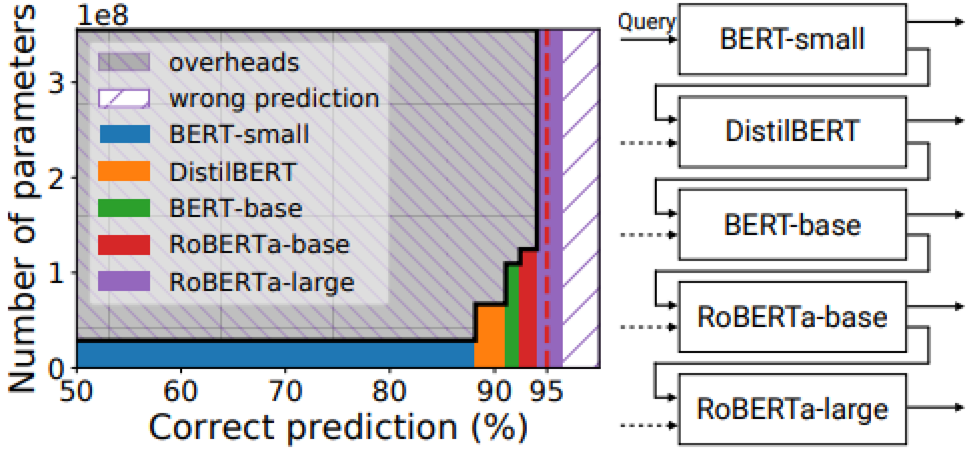


Figure 4. Tabi workflow. Highlighted components are optimized in this work. Components in dashed lines form the logical *multi-level inference engine* (§4).



(a) Resource overheads of LLMs. **(b)** An *imaginary* MLFQ.

Figure 1. Each color-filled bar's right edge shows its accuracy. A bar's width shows the accuracy improvement over the previous smaller DNNs, i.e., the percentage of queries that can be correctly served by a model but not by the ones on its left. The height shows the model size. The gray area is the resource overheads compared to an ideal scenario.

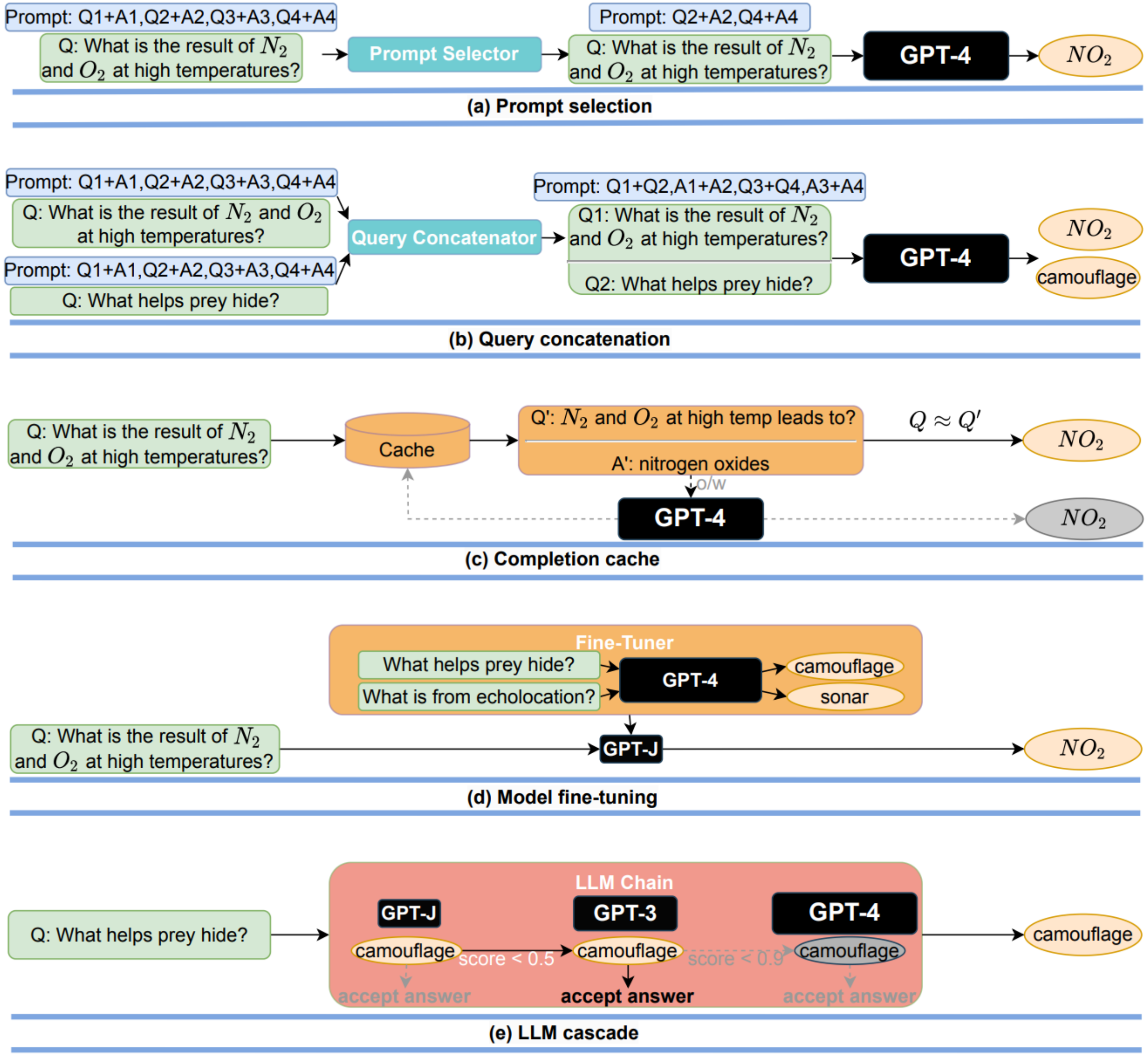
- ❑ Optimized for discriminative models

Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

Non-predictive Routing: FrugalGPT [9]

Key Idea:

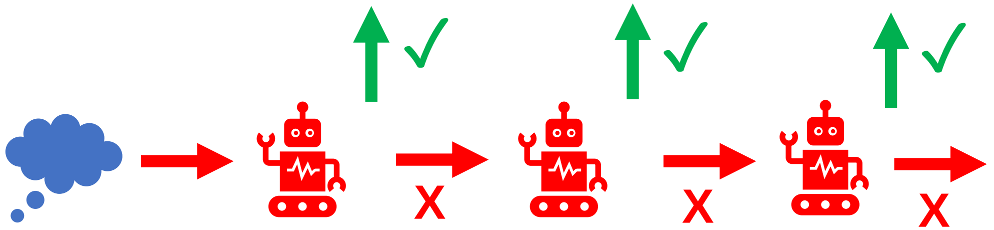
- Collection of various methods to optimize FM inference costs while maintaining correct responses

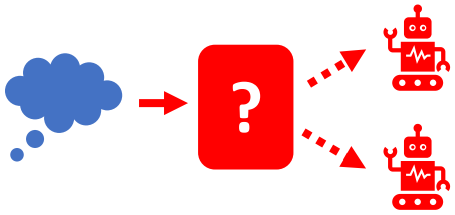


Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

Deploying FM:

Types of Routing





Non-predictive Routing

- Based on **collecting FM-generated outputs** from multiple FMs
- Sequential collection of outputs continues until an answer passes a quality threshold.

Limitations

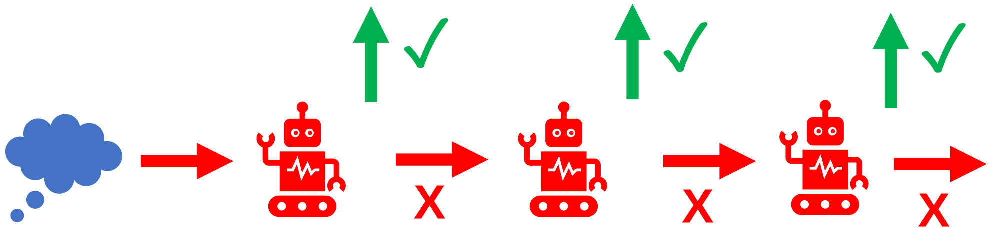
- Increased cost and latency due to many rounds of inference.

Predictive Routing

- Based on the **contents of the input request**; FM output not required
- Training prediction models (e.g. classifiers) using a dataset of input requests and associated human model preference labels.
- Performance is often **limited by** the quality and generalizability of the training **dataset**.
- Capabilities are **static** in post-deployment.

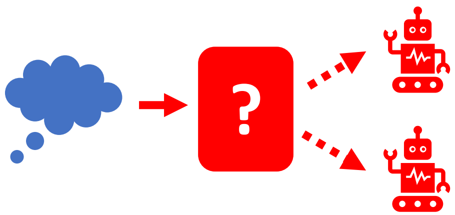
Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

Deploying FM:
Types of Routing



Non-predictive Routing

- Based on **collecting FM-generated outputs** from multiple FMs
- Sequential collection of outputs continues until an answer passes a quality threshold.
- Increased cost and latency due to many rounds of inference.



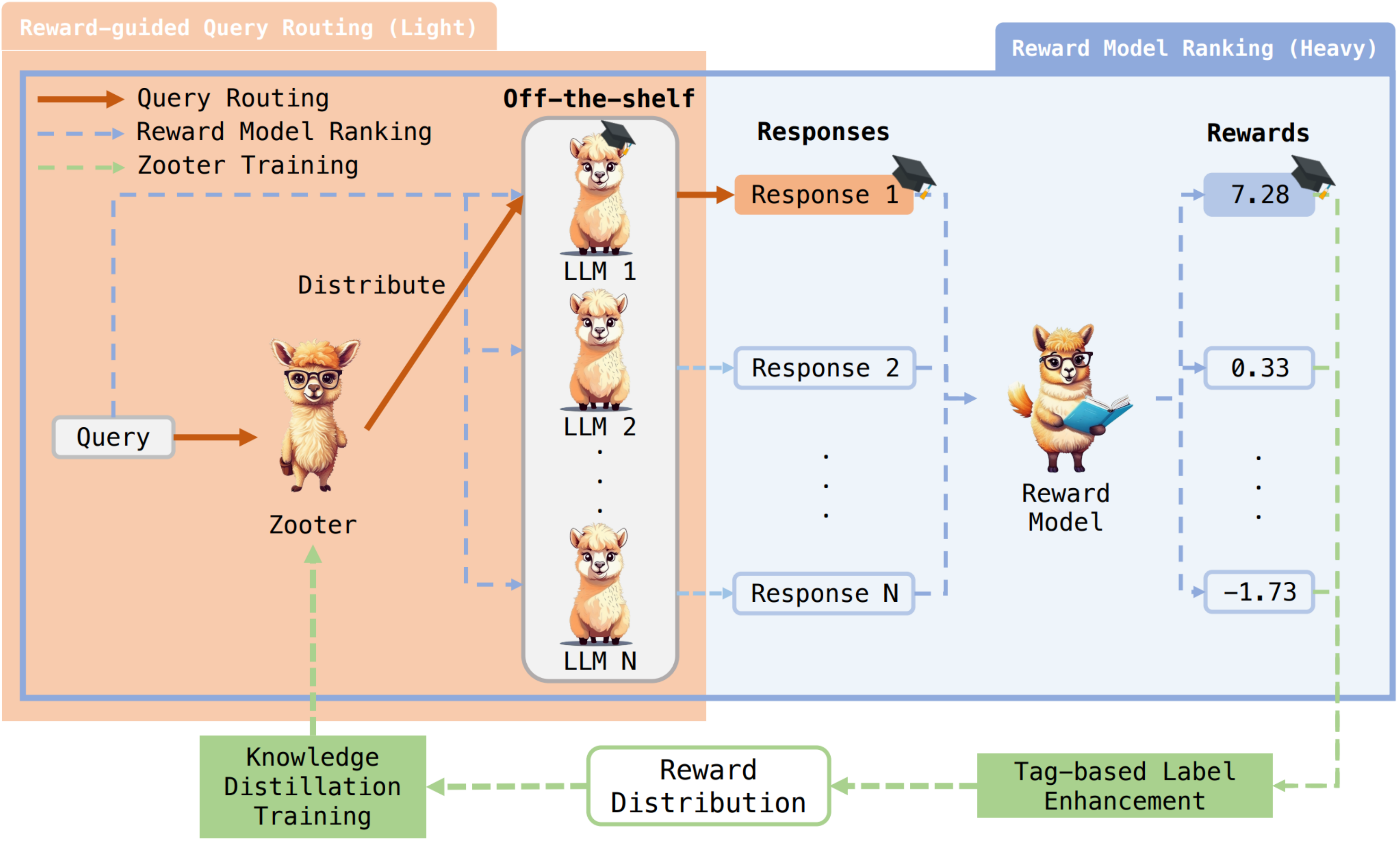
Predictive Routing

- Based on the **contents of the input request**; FM output not required
- Training prediction models (e.g. classifiers) using a dataset of input requests and associated human model preference labels.
- Performance is often **limited by** the quality and generalizability of the training **dataset**.
- Capabilities are **static** in post-deployment.

Zooter: Routing to the Expert[17]

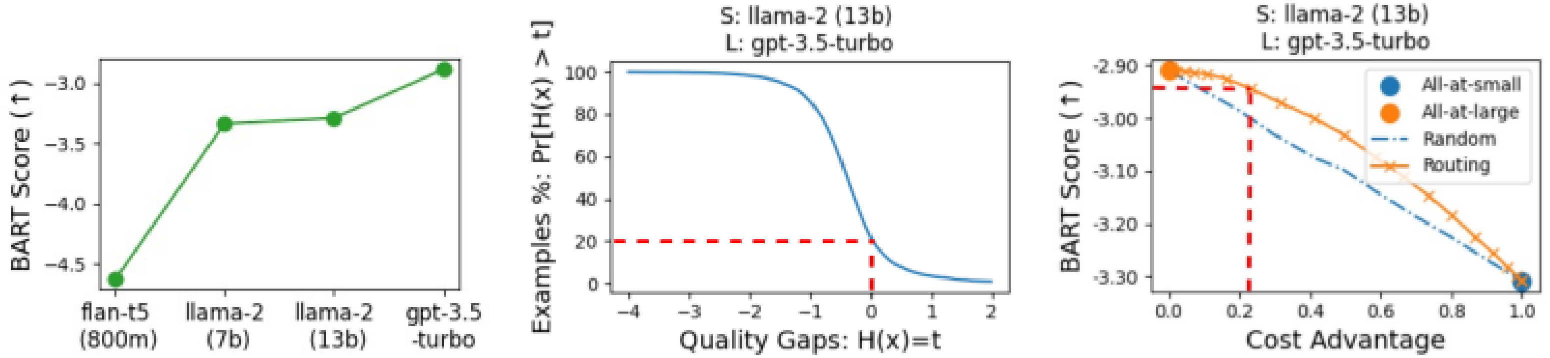
Key Idea

- Reward model ranking to obtain model expertise
- Trains routing function through knowledge distillation
- Inference in orange, training in green



Predictive Routing:
Hybrid-LLM [12]

- ❑ Predictive router using DeBERT-style encoder
  - ❑ Deterministic router $$\mathcal{L}(w) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i^{\text{det}}\log(p_w(x_i)) + (1-y_i^{\text{det}})\log(1-p_w(x_i))\right)$$
  - ❑ Probabilistic router $$\mathcal{L}(w) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i^{\text{prob}}\log(p_w(x_i)) + (1-y_i^{\text{prob}})\log(1-p_w(x_i))\right)$$



(a) Accuracy v/s size of LLM (b) Tail of accuracy difference (c) Results with routing

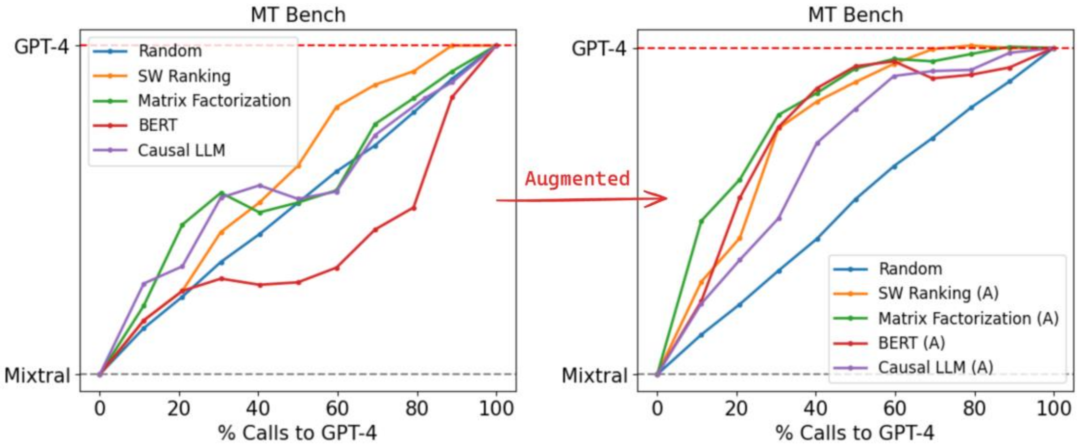
Figure 1: We use a dataset of natural language queries from a range of tasks like question answering, summarization, information extraction, etc. (See Section 4 for details). We observe that (a) smaller models generally give poorer response quality or lower BART score [Yuan et al., 2021], (b) Llama-2 (13b) outperforms GPT-3.5-turbo on around 20% examples, and (c) our router can make 22% fewer calls to GPT-3.5-turbo (cost advantage) with 1% drop in response quality (BART score).

Predictive Routing: RouteLLM[10]

Trained four *predictive* routers based on:

- ❑ Similarity-weighted ranking (Elo rating)
- ❑ Matrix-factorization model
- ❑ BERT-based classifier
- ❑ Causal LLM classifier

❑ Conversational *preference data* + *augmentation from benchmarks*



Before augmentation **After augmentation**

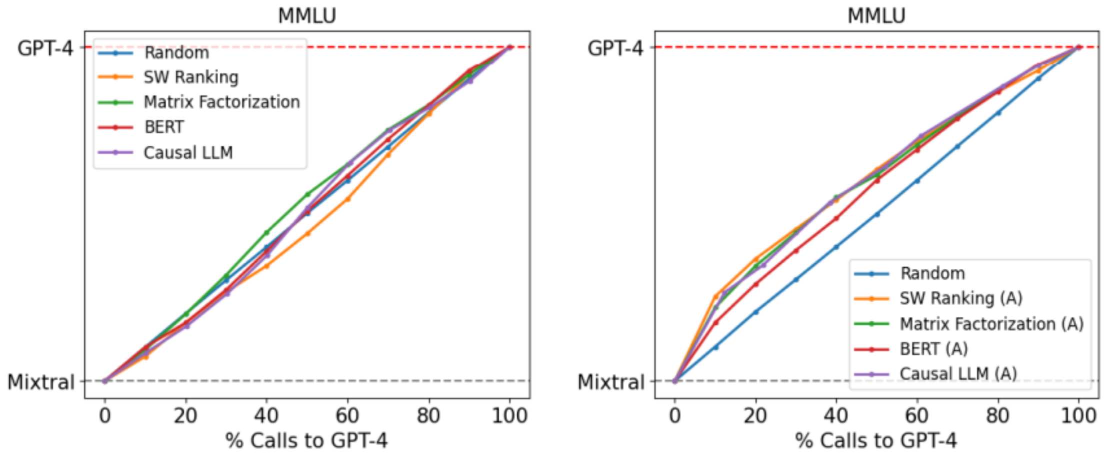


Figure 4: 5-shot MMLU performance for all routers.

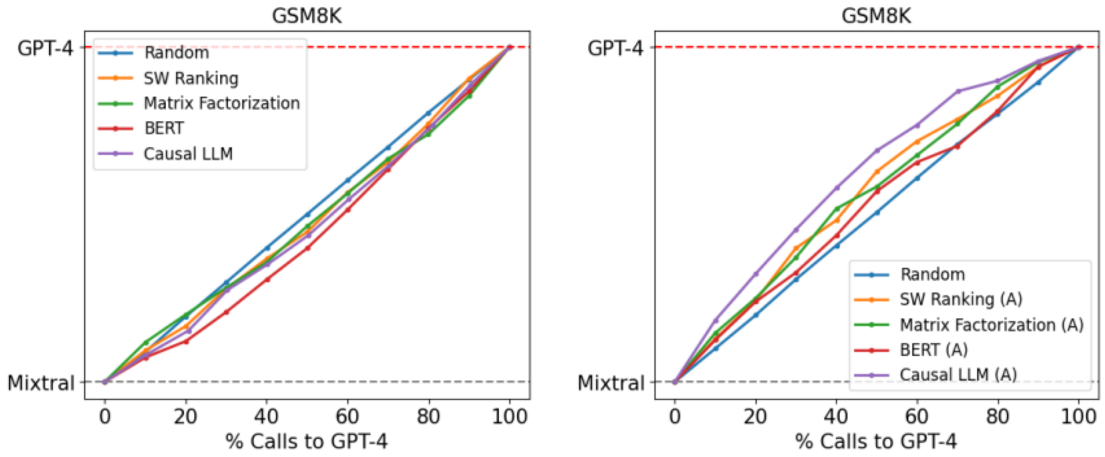


Figure 5: 8-shot GSM8K performance for all routers.

Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

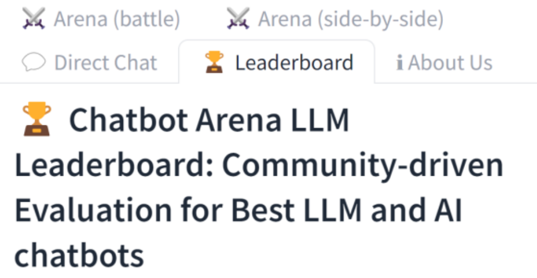
Predictive Routing:

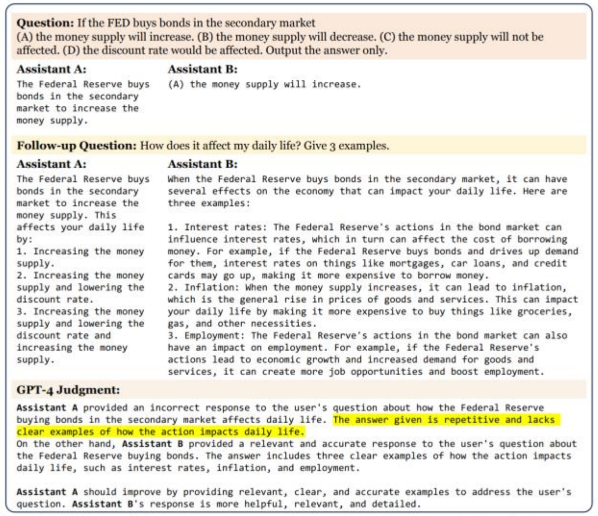
Challenges & Limitations

Often based on training a *classifier* (e.g. ML model) using a *preference dataset*

Matrix Factorization	Elo Ranking (Bradley-Terry)
Masked LM	Causal LM
Probabilistic Routing	Clustering

(examples from RouteLLM, Hybrid-LLM, RouterBench)





Question: If the FED buys bonds in the secondary market
(A) the money supply will increase. (B) the money supply will decrease. (C) the money supply will not be affected. (D) the discount rate would be affected. Output the answer only.

Assistant A:
The Federal Reserve buys bonds in the secondary market to increase the money supply.

Assistant B:
(A) the money supply will increase.

Follow-up Question: How does it affect my daily life? Give 3 examples.

Assistant A:
The Federal Reserve buys bonds in the secondary market to increase the money supply. This affects your daily life by:
1. Increasing the money supply.
2. Increasing the money supply and lowering the discount rate.
3. Increasing the money supply and lowering the discount rate and increasing the money supply.

Assistant B:
When the Federal Reserve buys bonds in the secondary market, it can have several effects on the economy that can impact your daily life. Here are three examples:

1. Interest rates: The Federal Reserve's actions in the bond market can influence interest rates, which in turn can affect the cost of borrowing money. For example, if the Federal Reserve buys bonds and drives up demand for them, interest rates on things like mortgages, car loans, and credit cards may go up, making it more expensive to borrow money.
2. Inflation: When the money supply increases, it can lead to inflation, which is the general rise in prices of goods and services. This can impact your daily life by making it more expensive to buy things like groceries, gas, and other necessities.
3. Employment: The Federal Reserve's actions in the bond market can also have an impact on employment. For example, if the Federal Reserve's actions lead to economic growth and increased demand for goods and services, it can create more job opportunities and boost employment.

GPT-4 Judgment:

Assistant A provided an incorrect response to the user's question about how the Federal Reserve buying bonds in the secondary market affects daily life. The answer given is repetitive and lacks clear examples of how the action impacts daily life.
On the other hand, **Assistant B** provided a relevant and accurate response to the user's question about the Federal Reserve buying bonds. The answer includes three clear examples of how the action impacts daily life, such as interest rates, inflation, and employment.

Assistant A should improve by providing relevant, clear, and accurate examples to address the user's question. **Assistant B**'s response is more helpful, relevant, and detailed.

Figure 1: Multi-turn dialogues between a user and two AI assistants—LLaMA-13B (Assistant A) and Vicuna-13B (Assistant B)—initiated by a question from the MMLU benchmark and a follow-up instruction. GPT-4 is then presented with the context to determine which assistant answers better.

Source: Zheng et. Al, 2023 [18]

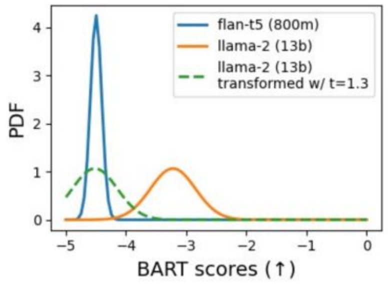
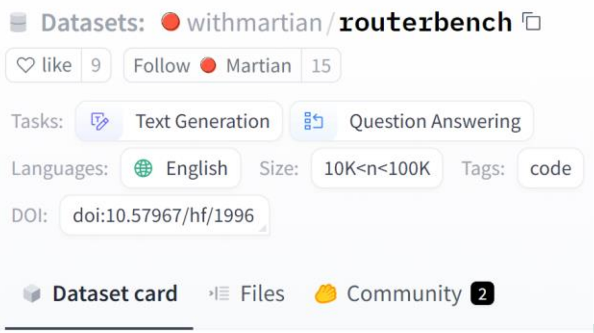


Figure 3: Response quality distribution for FLAN-t5 (800m) and Llama-2 (13b) on the query **"How to identify the index of median?"** measured in BART scores. Llama-2 (13b) with transformation significantly overlaps with FLAN-t5 (800m).

Source: Hybrid-LLM [12]



Predictive Routing:

Challenges & Limitations

Often based on training a *classifier* (e.g. ML model) using a *preference dataset*

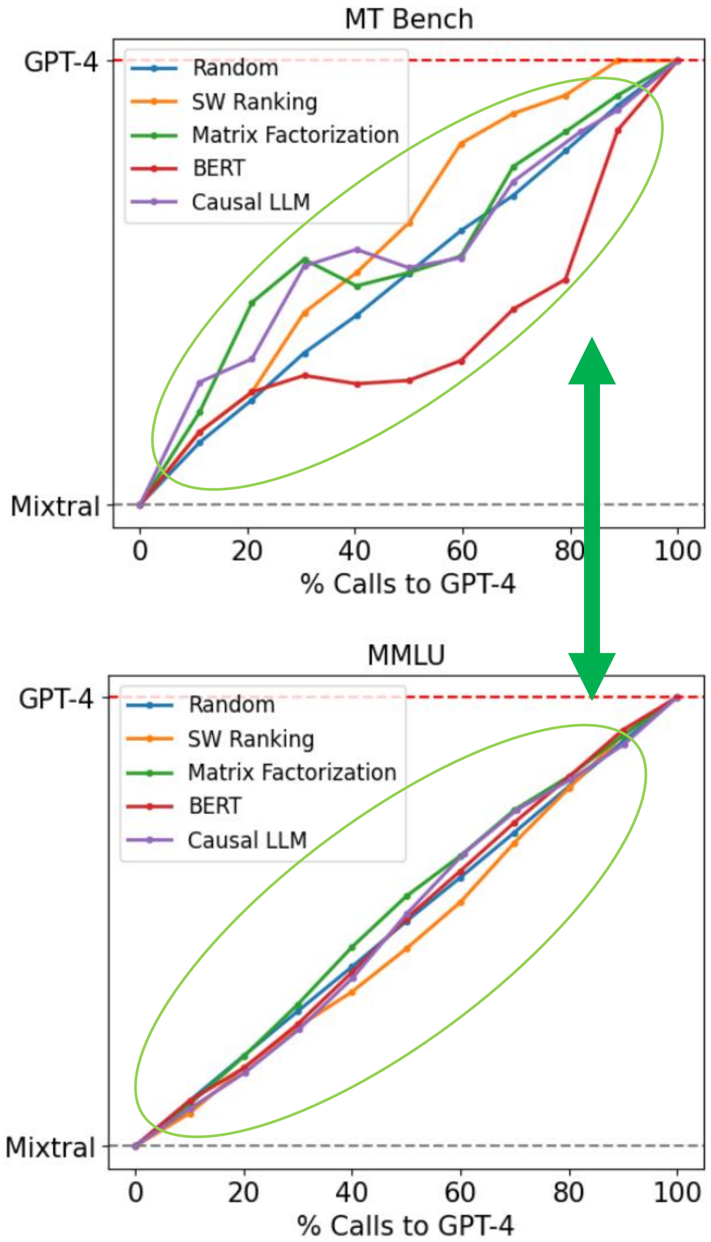
Leads to following related challenges:

Reliance on the quality of preference dataset

Update process is complicated

Reliance on the quality of preference dataset

Trained using mainly conversation data (Chatbot Arena), routers perform better on conversational benchmark (**MT Bench**) compared to multiple-choice questions (**MMLU**)



Source: RouteLLM [10]

Traditional problems in ML

- ❑ Acquiring preference labels is difficult
- ❑ Do preference labels accurately reflect each FM's strengths/weaknesses?
- ❑ How well does trained classifier generalize to unseen types of input?
  - ❑ e.g. multi-turn conversation vs. multiple-choice questions

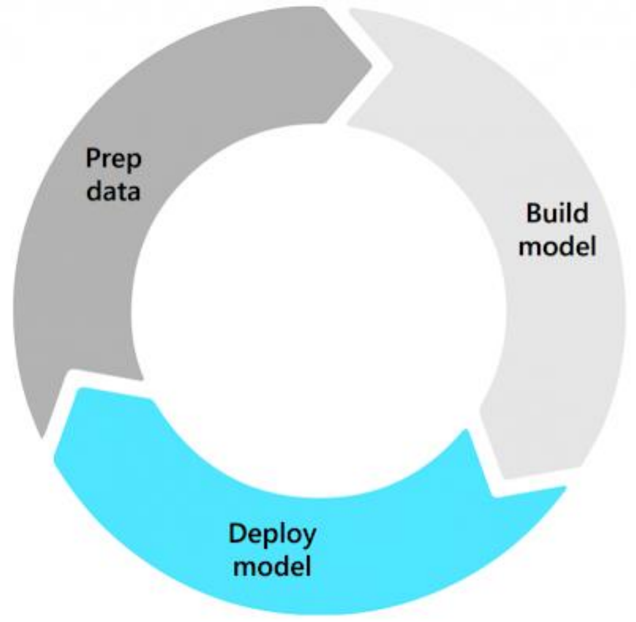
Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

Update process is complicated

When deployed, routers and router decisions remain ***static***

Whenever a change is needed to the routing process, updating the routing model is a resource-intensive process

- ❑ Training data needs to be updated
- ❑ Models need to be re-trained/adjusted
- ❑ Updated routers have to deployed to production
    - ❑ e.g. part of a software update to a smartphone



ML lifecycle process repeats every time an update is required

Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

Key Takeaways

- Important to consider FM selection criteria and pick accordingly
  - e.g. type of deployment environment

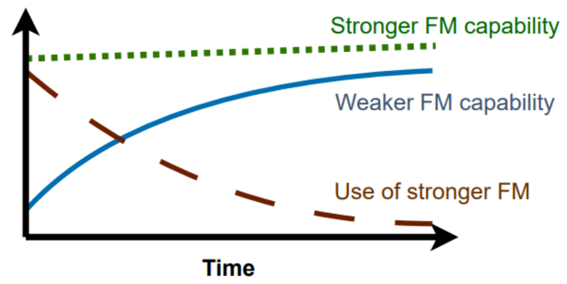
- Consider FM from perspective of FMware (black box) and optimize
  - e.g. combination of smaller LMs vs. one large LM

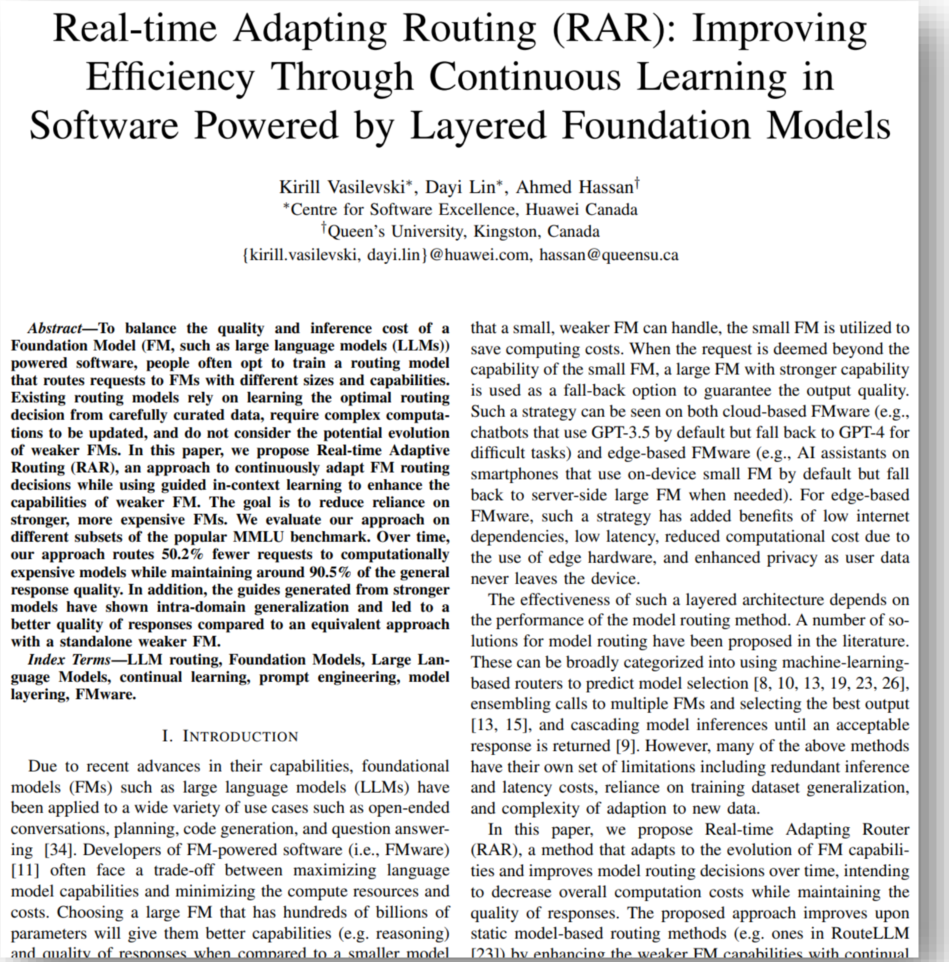
- Mix and match different methods and see what works best
  - e.g. prompt engineering + ensembling + routing

Vasilevski et al., Balancing Cost and Quality in FMware, Toronto, Canada, 2024

RAR: Real-time Adapting Routing

https://arxiv.org/abs/2411.09837 (pre-print, under review)





Real-time Adapting Routing (RAR): Improving Efficiency Through Continuous Learning in Software Powered by Layered Foundation Models

Kirill Vasilevski*, Dayi Lin*, Ahmed Hassan†

*Centre for Software Excellence, Huawei Canada

†Queen's University, Kingston, Canada

{kirill.vasilevski, dayi.lin}@huawei.com, hassan@queensu.ca

***Abstract*—To balance the quality and inference cost of a Foundation Model (FM, such as large language models (LLMs)) powered software, people often opt to train a routing model that routes requests to FMs with different sizes and capabilities. Existing routing models rely on learning the optimal routing decision from carefully curated data, require complex computations to be updated, and do not consider the potential evolution of weaker FMs. In this paper, we propose Real-time Adaptive Routing (RAR), an approach to continuously adapt FM routing decisions while using guided in-context learning to enhance the capabilities of weaker FM. The goal is to reduce reliance on stronger, more expensive FMs. We evaluate our approach on different subsets of the popular MMLU benchmark. Over time, our approach routes 50.2% fewer requests to computationally expensive models while maintaining around 90.5% of the general response quality. In addition, the guides generated from stronger models have shown intra-domain generalization and led to a better quality of responses compared to an equivalent approach with a standalone weaker FM.**

***Index Terms*—LLM routing, Foundation Models, Large Language Models, continual learning, prompt engineering, model layering, FMware.**

I. Introduction

Due to recent advances in their capabilities, foundational models (FMs) such as large language models (LLMs) have been applied to a wide variety of use cases such as open-ended conversations, planning, code generation, and question answering [34]. Developers of FM-powered software (i.e., FMware) [11] often face a trade-off between maximizing language model capabilities and minimizing the compute resources and costs. Choosing a large FM that has hundreds of billions of parameters will give them better capabilities (e.g. reasoning) and quality of responses when compared to a smaller model that a small, weaker FM can handle, the small FM is utilized to save computing costs. When the request is deemed beyond the capability of the small FM, a large FM with stronger capability is used as a fall-back option to guarantee the output quality. Such a strategy can be seen on both cloud-based FMware (e.g., chatbots that use GPT-3.5 by default but fall back to GPT-4 for difficult tasks) and edge-based FMware (e.g., AI assistants on smartphones that use on-device small FM by default but fall back to server-side large FM when needed). For edge-based FMware, such a strategy has added benefits of low internet dependencies, low latency, reduced computational cost due to the use of edge hardware, and enhanced privacy as user data never leaves the device.

The effectiveness of such a layered architecture depends on the performance of the model routing method. A number of solutions for model routing have been proposed in the literature. These can be broadly categorized into using machine-learning-based routers to predict model selection [8, 10, 13, 19, 23, 26], ensembling calls to multiple FMs and selecting the best output [13, 15], and cascading model inferences until an acceptable response is returned [9]. However, many of the above methods have their own set of limitations including redundant inference and latency costs, reliance on training dataset generalization, and complexity of adaption to new data.

In this paper, we propose Real-time Adapting Router (RAR), a method that adapts to the evolution of FM capabilities and improves model routing decisions over time, intending to decrease overall computation costs while maintaining the quality of responses. The proposed approach improves upon static model-based routing methods (e.g. ones in RouteLLM [23]) by enhancing the weaker FM capabilities with continual

- Our proposed approach of intelligent routing
  - Minimize costs while maintaining response quality
    - Decrease dependence on larger FM by improving capabilities of smaller FM
    - In some cases, reduced use of larger FM by ~50% while maintaining ~90% of response quality
  - Combination of *static predictive routing + prompt-based continual learning*

References & Resources

1. Malhotra, T. (2024, June 15). With 700,000 large language models (LLMs) on hugging face already, where is the future of artificial intelligence AI headed? MarkTechPost. https://www.marktechpost.com/2024/06/15/with-700000-large-language-models-llms-on-hugging-face-already-where-is-the-future-of-artificial-intelligence-ai-headed/
2. Meta, A. I. (2024). Introducing meta llama 3: The most capable openly available llm to date. Meta AI.
3. Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., & Finn, C. (2024). Direct preference optimization: Your language model is secretly a reward model. Advances in Neural Information Processing Systems, 36.
4. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., ... & Lowe, R. (2022). Training language models to follow instructions with human feedback. Advances in neural information processing systems, 35, 27730-27744.
5. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., ... & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems, 35, 24824-24837.
6. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., & Narasimhan, K. (2024). Tree of thoughts: Deliberate problem solving with large language models. Advances in Neural Information Processing Systems, 36.
7. Jiang, D., Ren, X., & Lin, B. Y. (2023). Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. arXiv preprint arXiv:2306.02561.
8. Lu, X., Liu, Z., Liusie, A., Raina, V., Mudupalli, V., Zhang, Y., & Beauchamp, W. (2024). Blending is all you need: Cheaper, better alternative to trillion-parameters llm. arXiv preprint arXiv:2401.02994.
9. Chen, L., Zaharia, M., & Zou, J. (2023). Frugalgpt: How to use large language models while reducing cost and improving performance. arXiv preprint arXiv:2305.05176.
10. Ong, I., Almahairi, A., Wu, V., Chiang, W. L., Wu, T., Gonzalez, J. E., ... & Stoica, I. (2024). Routellm: Learning to route llms with preference data. arXiv preprint arXiv:2406.18665.
11. Wang, Y., Chen, K., Tan, H., & Guo, K. (2023, May). Tabi: An efficient multi-level inference system for large language models. In Proceedings of the Eighteenth European Conference on Computer Systems (pp. 233-248).
12. Ding, D., Mallick, A., Wang, C., Sim, R., Mukherjee, S., Ruhle, V., ... & Awadallah, A. H. (2024). Hybrid LLM: Cost-efficient and quality-aware query routing. arXiv preprint arXiv:2404.14618.
13. Hu, Q. J., Bieker, J., Li, X., Jiang, N., Keigwin, B., Ranganath, G., ... & Upadhyay, S. K. (2024). ROUTERBENCH: A Benchmark for Multi-LLM Routing System. arXiv preprint arXiv:2403.12031.
14. Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., ... & Sayed, W. E. (2024). Mixtral of experts. arXiv preprint arXiv:2401.04088.
15. Si, C., Shi, W., Zhao, C., Zettlemoyer, L., & Boyd-Graber, J. (2023). Getting more out of mixture of language model reasoning experts. arXiv preprint arXiv:2305.14628.
16. Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., ... & Zhou, D. (2022). Self-consistency improves chain of thought reasoning in language models. arXiv preprint arXiv:2203.11171.
17. Lu, K., Yuan, H., Lin, R., Lin, J., Yuan, Z., Zhou, C., & Zhou, J. (2023). Routing to the expert: Efficient reward-guided ensemble of large language models. arXiv preprint arXiv:2311.08692.
18. Zheng, L., Chiang, W. L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., ... & Stoica, I. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena. Advances in Neural Information Processing Systems, 36, 46595-46623.