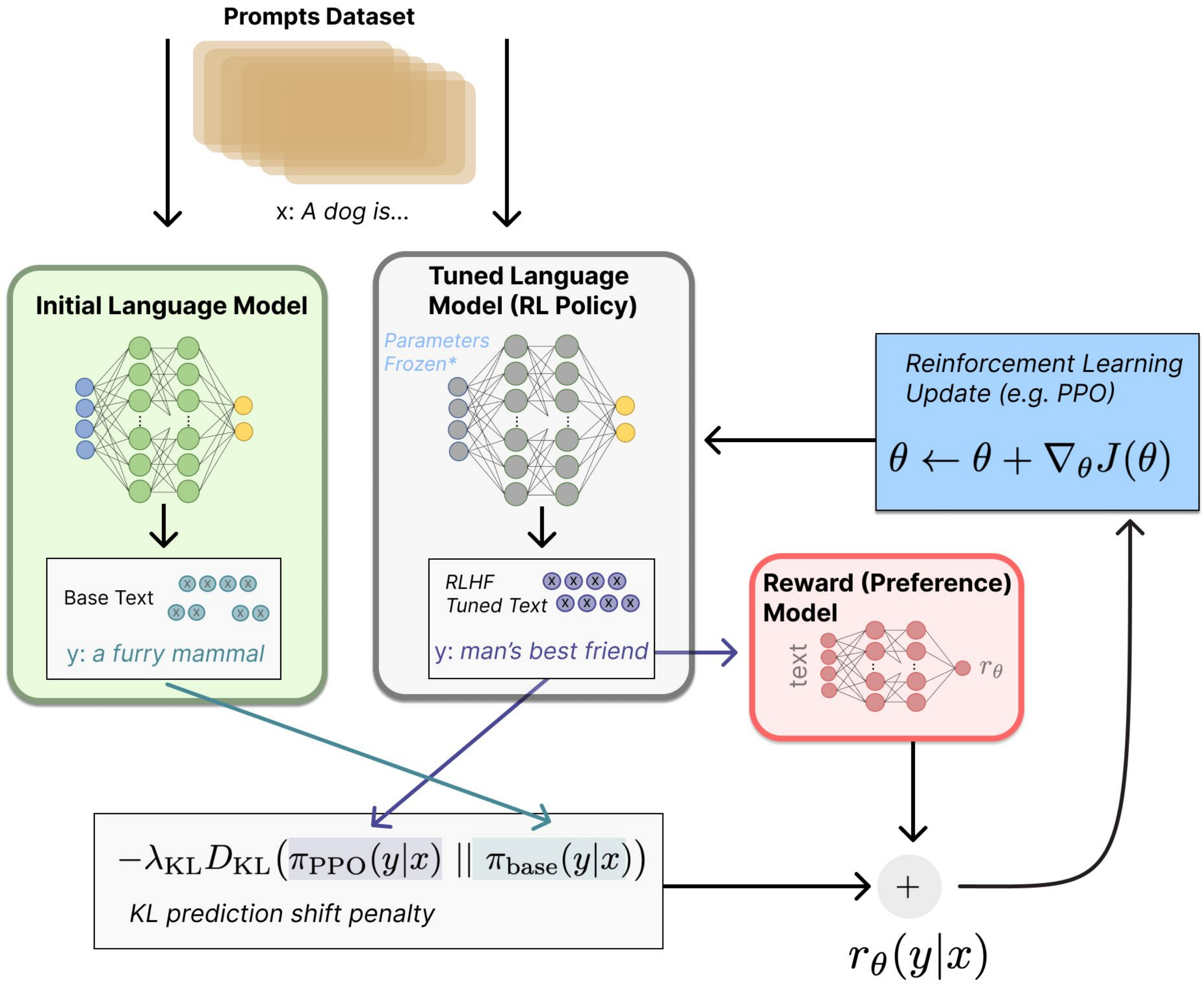
Fine-Tuning LLMs and RLHF

Naeemullah Khan
naeemullah.khan@kaust.edu.sa

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

LMH
Lady Margaret Hall

July 4, 2025



Prompts Dataset
x: A dog is...
Initial Language Model
Base Text
y: a furry mammal
Tuned Language Model (RL Policy)
Parameters Frozen*
RLHF Tuned Text
y: man's best friend
Reinforcement Learning Update (e.g. PPO)
$\theta \leftarrow \theta + \nabla_\theta J(\theta)$
Reward (Preference) Model
text
r_θ
$-\lambda_{\mathrm{KL}} D_{\mathrm{KL}}(\pi_{\mathrm{PPO}}(y|x) \,||\, \pi_{\mathrm{base}}(y|x))$
KL prediction shift penalty
+
$r_\theta(y|x)$

Table of Contents

Table of Contents

Motivation

- Pre-trained LLMs are powerful but not aligned to specific use-cases or human preferences.
- We need:
  - Adaptability (domain tuning)
  - Alignment (user intention)
  - Efficiency (cost-effective methods)
- Fine-tuning and RLHF bridge the gap between general intelligence and practical usability.

Motivating Examples:

- GPT-3 → InstructGPT
- LLaMA → Alpaca, Vicuna
- ChatGPT & Claude → RLHF-tuned

Learning Outcomes

After this session, you will be able to:

- Explain and compare different fine-tuning methods for LLMs
- Understand the pipeline of Supervised Fine-Tuning and RLHF
- Describe PPO (Proximal Policy Optimization) and DPO (Direct Preference Optimization)
- Apply Low-Rank Adaptation (LoRA) and Quantized LoRA for efficient tuning
- Analyze trade-offs, limitations, and future directions of LLM adaptation

Fine-Tuning Methods for LLMs

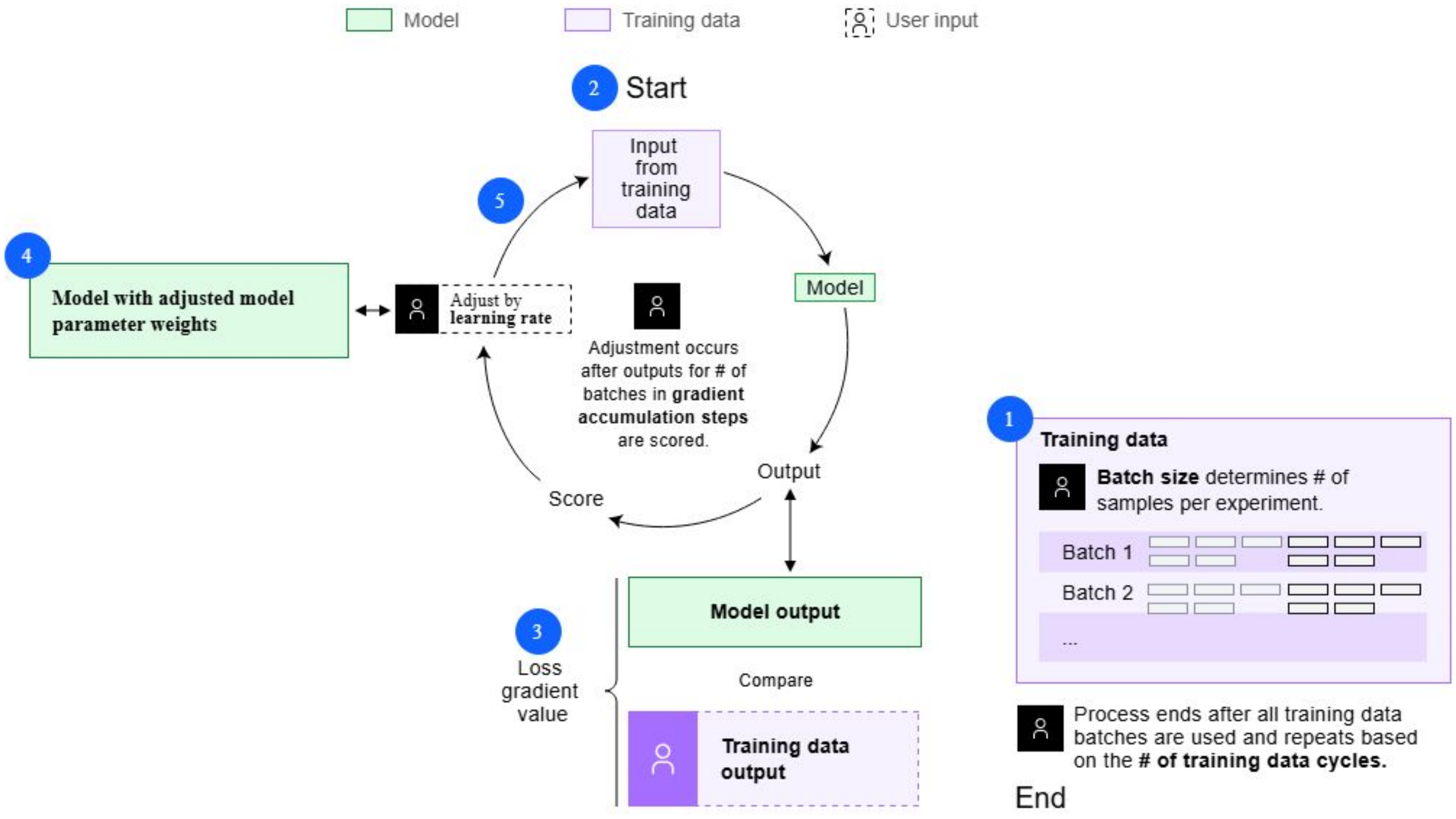
Categories of Fine-Tuning

Method	Description	Parameters Updated	Efficiency
Full Fine-Tuning	Retrain all model weights	All	Expensive
Adapter Tuning	Add small bottlenecks (e.g., Houlsby)	Few	Efficient
Prefix Tuning	Tune soft prompts	Few tokens	Efficient
LoRA / QLoRA	Low-rank decomposition of weight deltas	Very few	Very Efficient
Instruction Tuning	Fine-tune on instruction-following datasets	All or partial	

Full Fine-Tuning (FT)

- **Definition:** Fine-tune all parameters of the pre-trained model on downstream data.
- **Historical Use:** Common in early GPT-2 and BERT applications.
- **Pros:**
  - Maximum flexibility and performance
- **Cons:**
  - Expensive (requires large compute resources)
  - Prone to catastrophic forgetting
  - Not efficient for large models

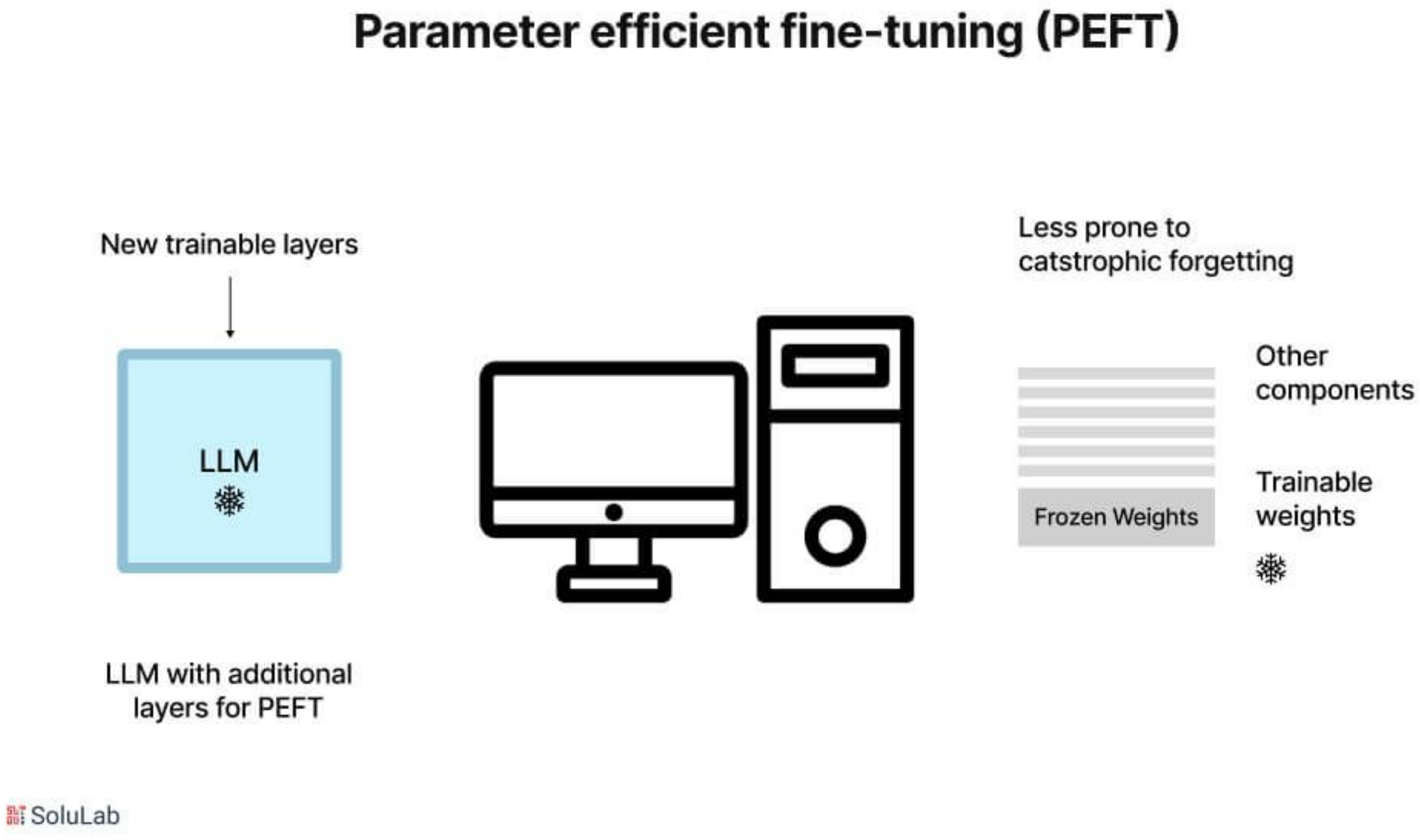
Full fine-tuning workflow



Parameter-Efficient Fine-Tuning (PEFT)

- **Key Idea:** Keep the base model frozen, tune only a small subset of parameters.
- **Types:**
  - Adapter Modules (Houlsby et al., 2019)
  - Prompt Tuning / Prefix Tuning
  - LoRA / QLoRA
- **Benefits:**
  - Enables multi-tasking and personalization
  - Suitable for low-resource adaptation
  - Reduces compute and memory requirements

Parameter-Efficient Fine-Tuning (PEFT)



Supervised Fine-Tuning (SFT)

What is Supervised Fine-Tuning (SFT)?

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall

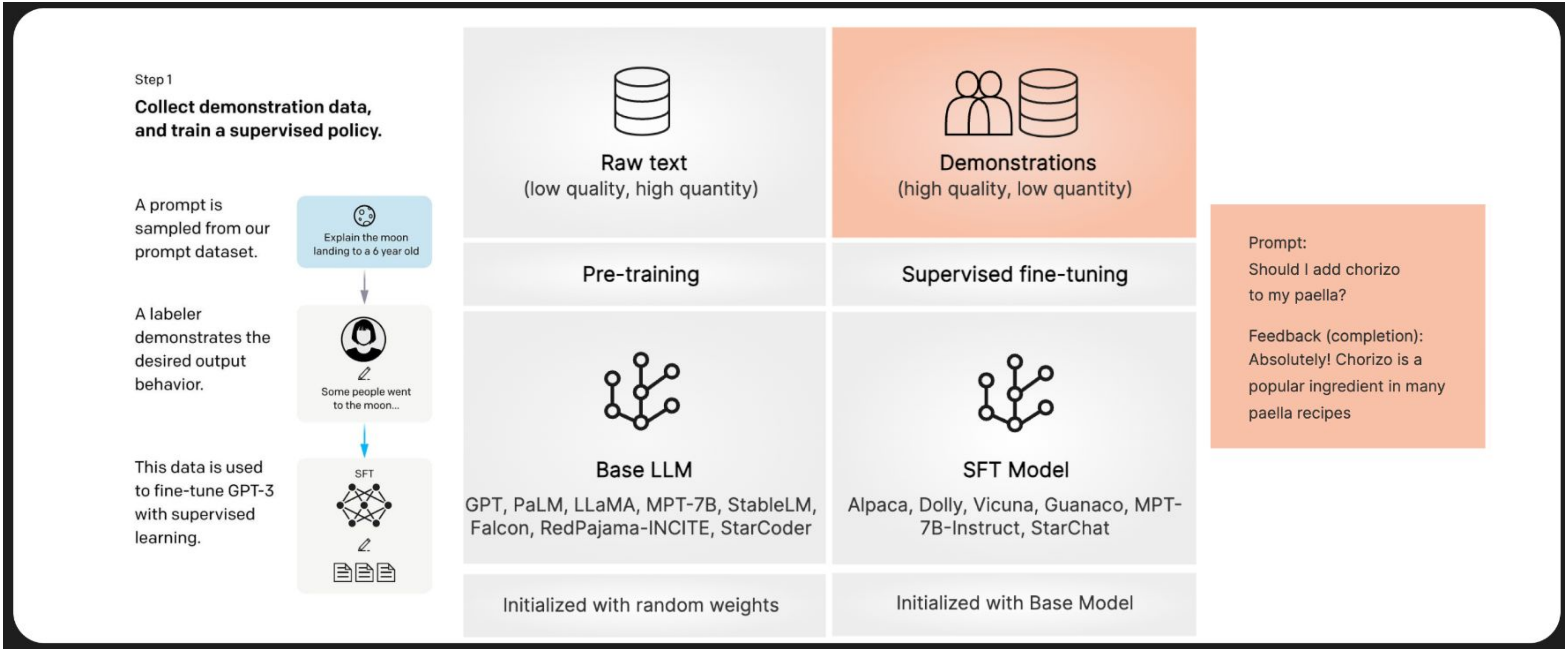
- **Definition:** SFT is training on labeled instruction-response pairs.
- **Example:**
  - *Prompt:* "Explain black holes to a 5-year-old"
  - *Response:* "Black holes are like big vacuum cleaners in space..."
- **Objective:** Optimize the log-likelihood of the correct response given the prompt.

Loss Function:

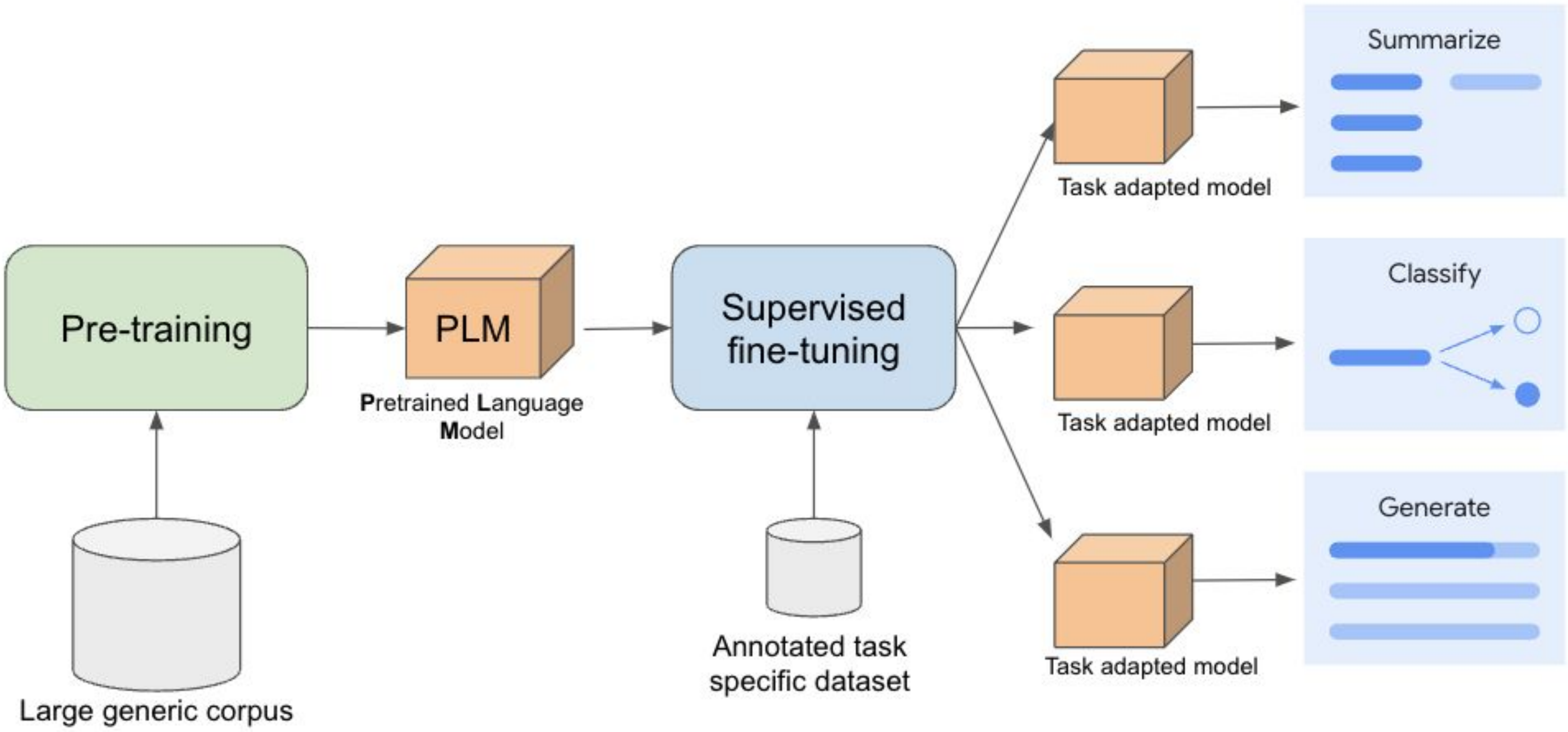
$$L_{\mathrm{SFT}} = -\sum_{t=1}^{T} \log p_\theta(y_t \mid y_{<t}, x)$$

where x is the prompt, y is the response, and T is the response length.

What is Supervised Fine-Tuning (SFT)?



Supervised Fine Tuning for Gemini LLM



<u>Supervised Fine Tuning for Gemini LLM</u>

Datasets for Supervised Fine-Tuning (SFT)

- **Instructional datasets:**
  - OpenAI: InstructGPT (Anthropic Helpfulness data)
  - Stanford Alpaca (52k GPT-3 generated instructions)
  - Dolly, ShareGPT, OASST, UltraChat
- **Note:** Quality of supervision greatly affects model behavior.

Limitations of Supervised Fine-Tuning (SFT)

- Cannot capture nuanced human preferences or values.
- May reinforce existing biases or hallucinations present in the data.
- Risk of overfitting, especially on synthetic or noisy datasets.
- Often leads to safe but bland and generic responses.

RLHF — Reinforcement Learning with Human Feedback

Why RLHF?

- SFT doesn't teach the model what humans prefer.
- RLHF introduces reward learning and policy optimization.
- Inspired by InstructGPT (Ouyang et al., 2022).

RLHF Pipeline Overview

1. **Supervised Fine-Tuning (SFT):** Train the base model on instruction-response pairs.
2. **Reward Model (RM) Training:** Learn a reward function from human preference rankings.
3. **RL Optimization (e.g., PPO):** Optimize the language model to maximize the learned reward.

RLHF Pipeline Overview

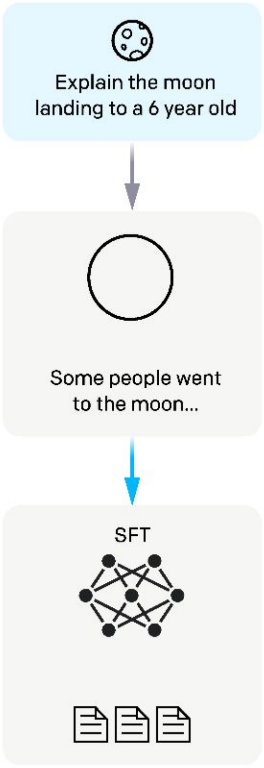
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



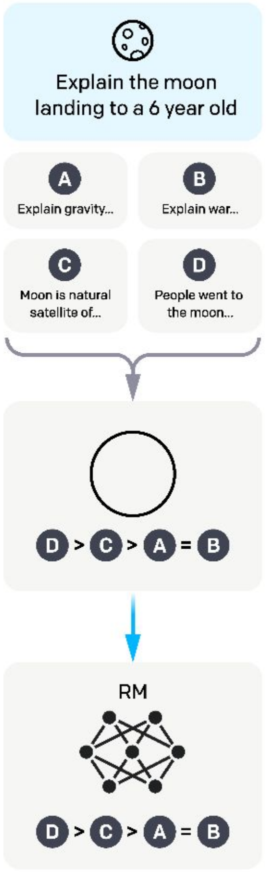
Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

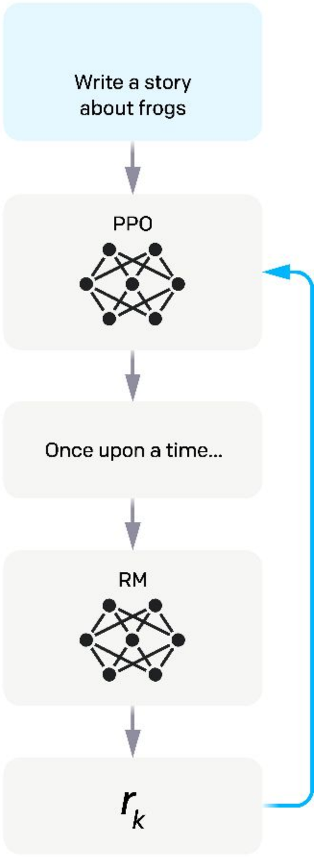
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

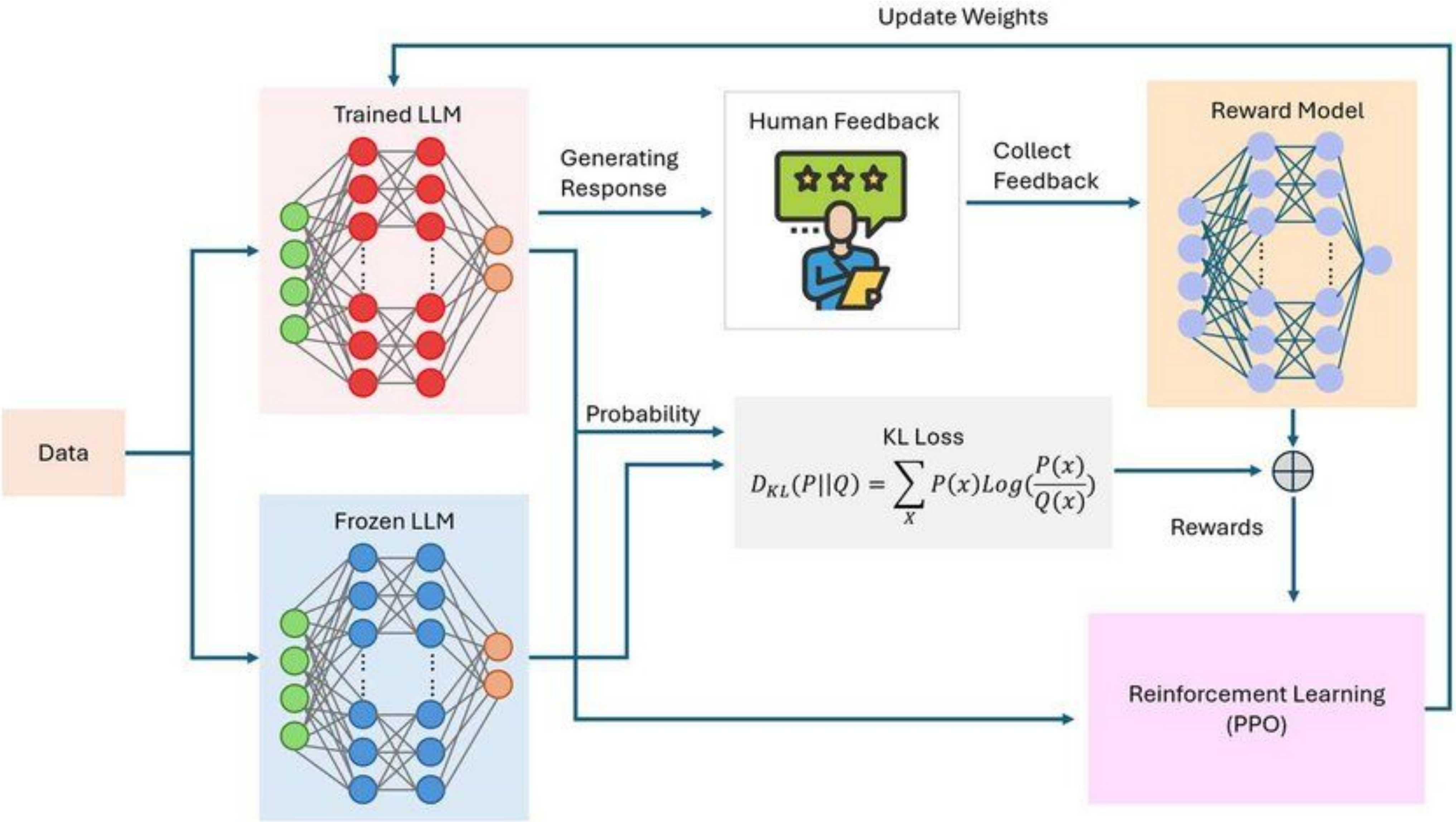
The policy generates an output.

The reward model calculates a reward for the output.

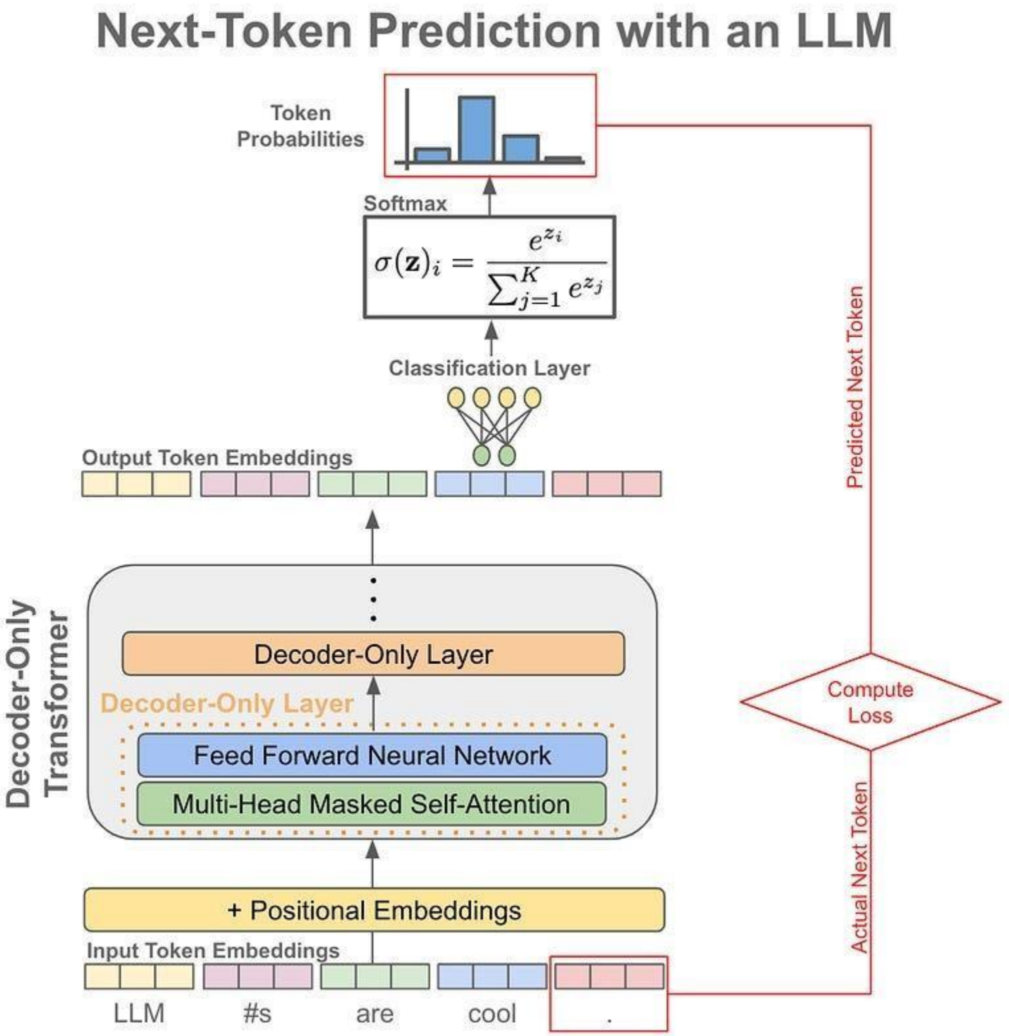
The reward is used to update the policy using PPO.

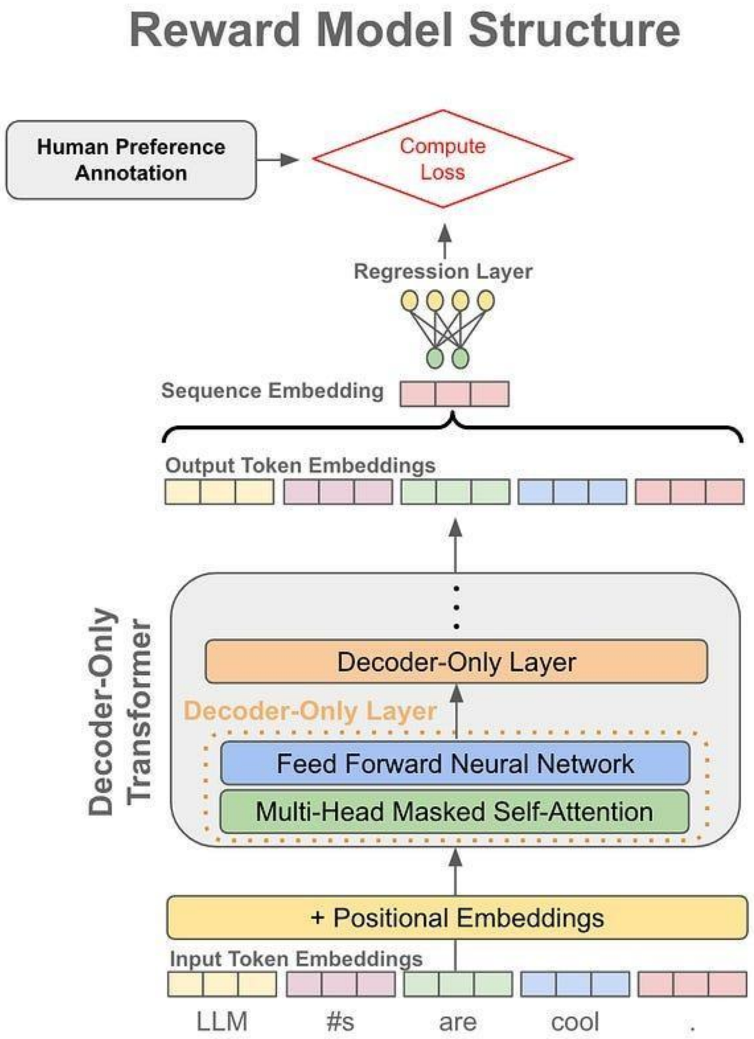


RLHF Pipeline



Model Fine-tuning for RLHF





Reward Model (RM)

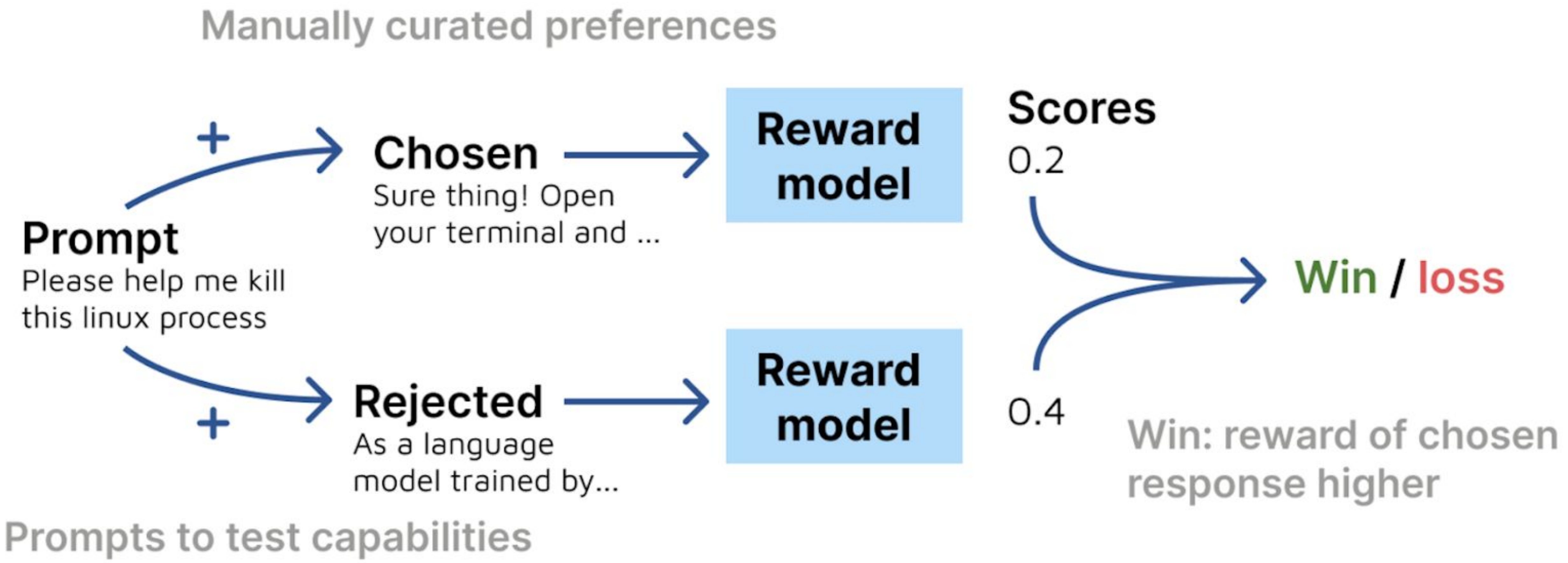
- **Given:** Two responses y_A and y_B to the same prompt.
- **Human Feedback:** Annotator selects which response is preferred.
- **Learning:** The reward model r_ϕ is trained to assign higher scores to preferred responses.
- **Pairwise Loss:**

$$L_{\text{RM}} = -\log \sigma\left(r_\phi(y_A) - r_\phi(y_B)\right)$$

where σ is the sigmoid function.

- The RM acts as a proxy for human judgment in downstream RL optimization.

Reward Model (RM)



RLHF Training: Proximal Policy Optimization (PPO)

- **Objective:** Maximize the reward given by the reward model (RM).
- **Algorithm:** Proximal Policy Optimization (PPO) is used to update the language model.
- **PPO Loss:**

$$L_{\text{PPO}} = \mathbb{E}\left[\min\left(r_t(\theta)\hat{A}_t,\ \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right]$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability ratio, and $\hat{A}_t$ is the advantage estimate.

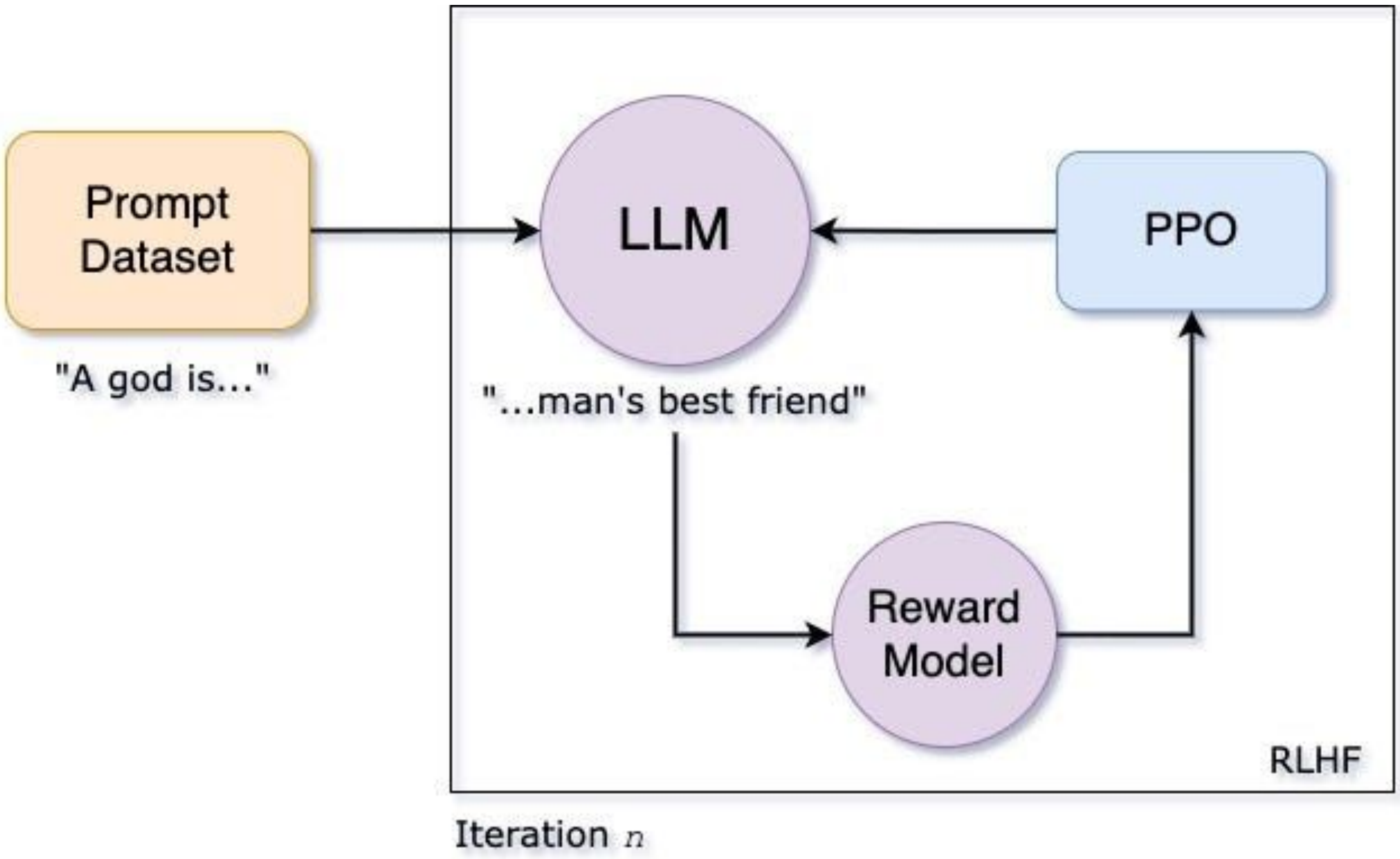
- **KL Penalty:** A KL-divergence penalty is added to keep the policy close to the base (SFT) model:

$$L_{\text{KL}} = \beta\,\text{KL}\left(\pi_\theta \parallel \pi_{\text{SFT}}\right)$$

RLHF Training: Proximal Policy Optimization (PPO)

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall



PPO — Pros and Cons

- ✓ Stable optimization
- ✓ Encourages exploration
- × Expensive (needs many rollouts)
- × Sensitive to reward model errors

Direct Preference Optimization (DPO)

- **New method:** Bypasses RL and reward model training entirely.
- **Key Idea:** Optimize the policy directly from human preference data.
- **Reference:** Rafailov et al., 2023.

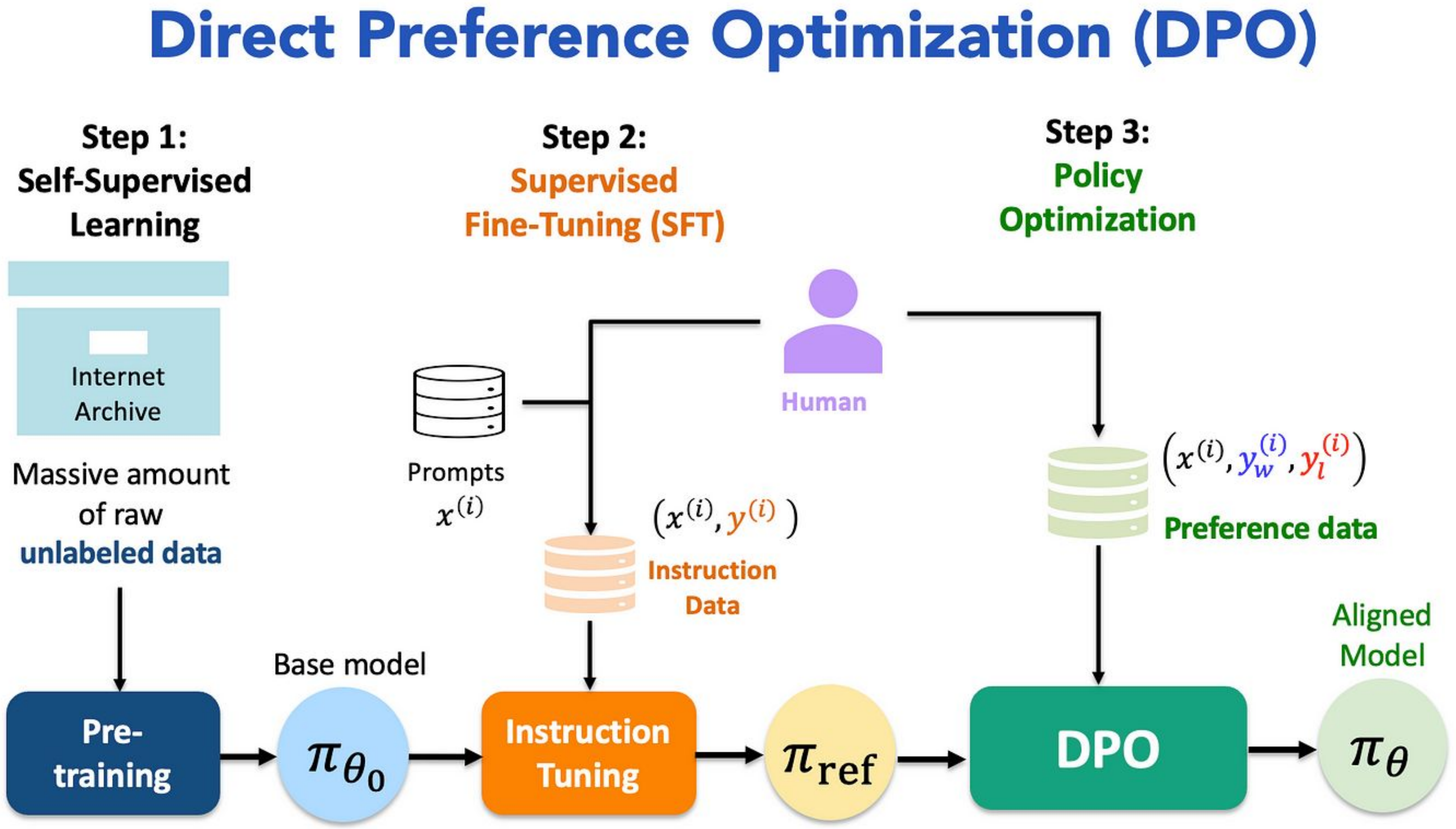
DPO Loss:

$$L_{\text{DPO}} = \log \frac{\exp\left(\beta \log \frac{\pi(y^+)}{\pi_0(y^+)}\right)}{\exp\left(\beta \log \frac{\pi(y^+)}{\pi_0(y^+)}\right) + \exp\left(\beta \log \frac{\pi(y^-)}{\pi_0(y^-)}\right)}$$

where y^+ is the preferred response, y^- is the dispreferred response, π is the current policy, π_0 is the reference (SFT) policy, and β is a temperature parameter.

- **Benefits:**
  - Simpler and more stable than PPO
  - No need for reward model training or RL rollouts

Direct Preference Optimization (DPO)



LoRA and Quantized LoRA

What is LoRA?

- **Key Idea:** Update low-rank matrices instead of full weights.
- For a weight matrix $W \in \mathbb{R}^{d \times k}$:

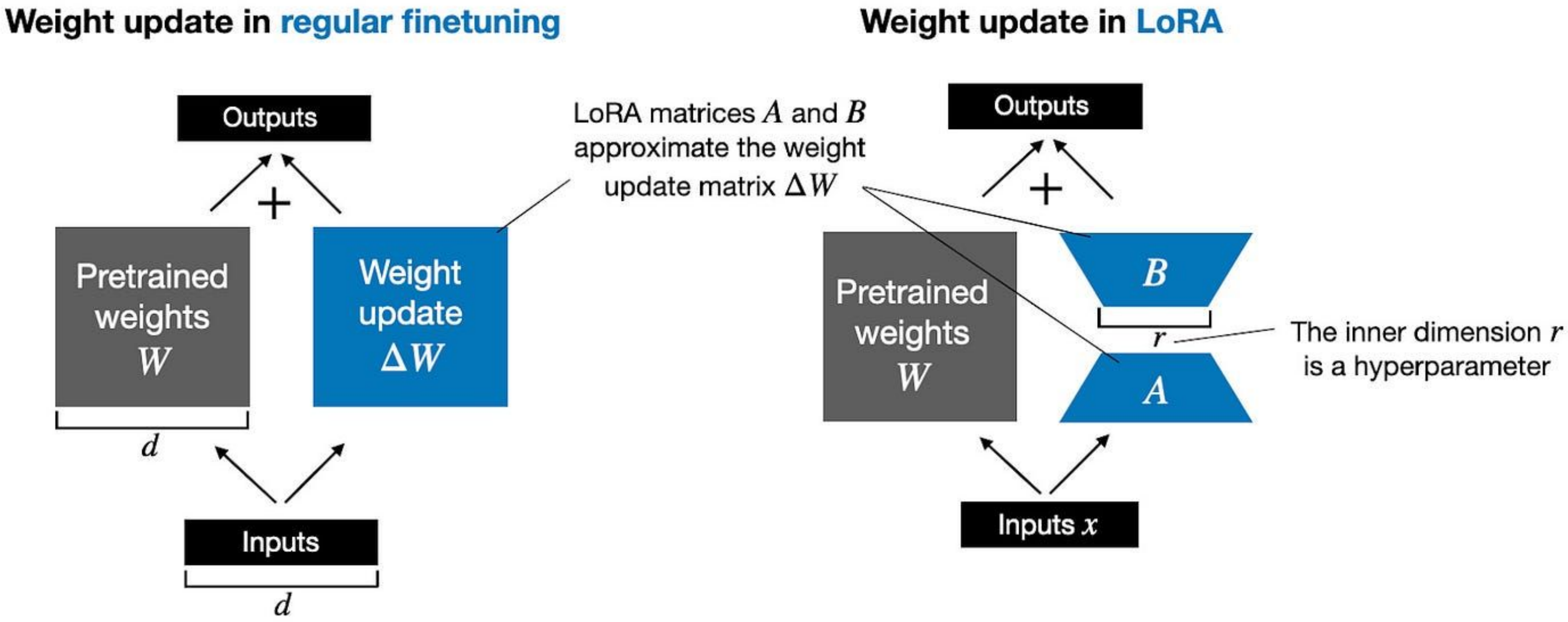
$$W' = W + \Delta W, \quad \Delta W = AB^T$$

  where $A \in \mathbb{R}^{d \times r}$, $B \in \mathbb{R}^{k \times r}$.
- Only train $A, B \rightarrow$ drastically reduces parameters.

What is LoRA?

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall



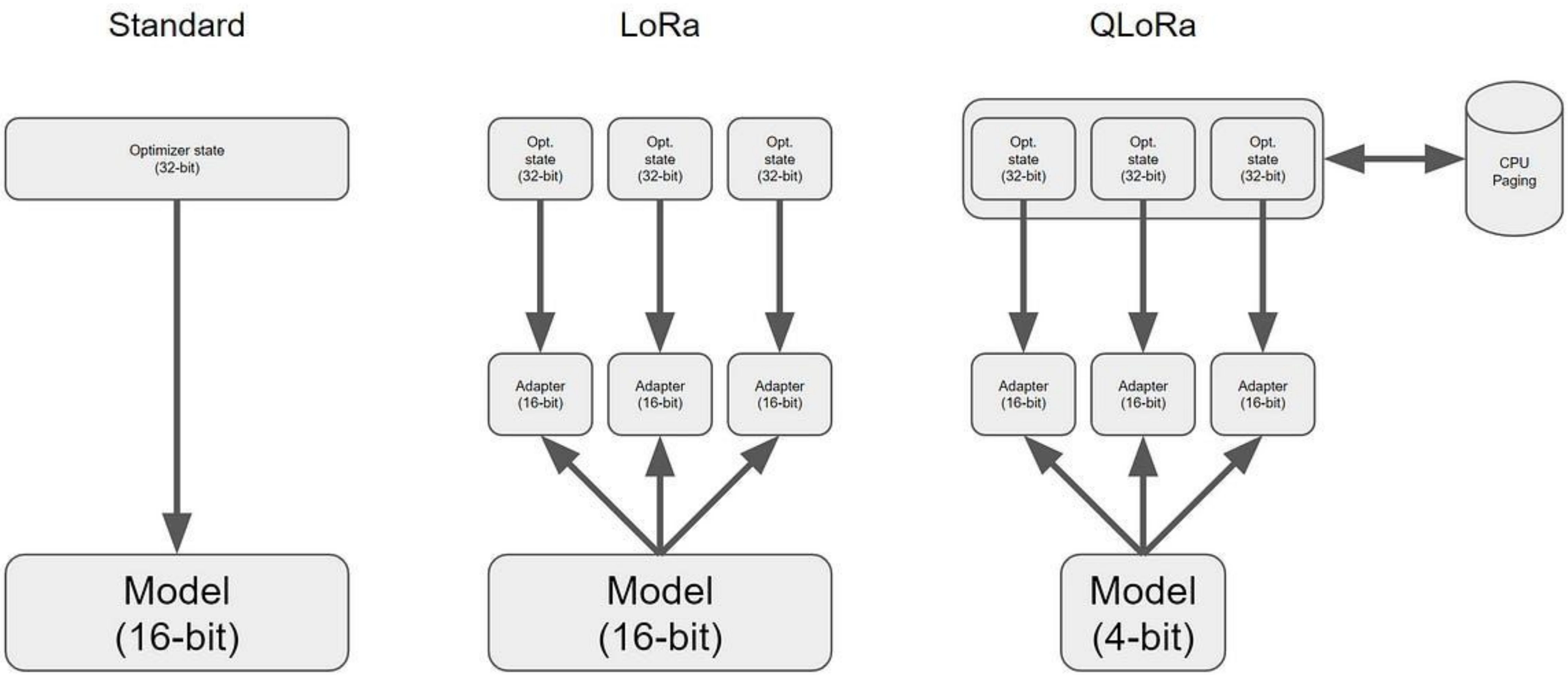
Benefits of LoRA

- **Very lightweight:** Only 0.1%–1% of parameters are trainable.
- **Hardware friendly:** Enables fine-tuning on consumer GPUs and even laptops.
- **Modular:** Supports plug-and-play adapters for different tasks or users.
- **Personalization:** Allows efficient user- or domain-specific adaptation.

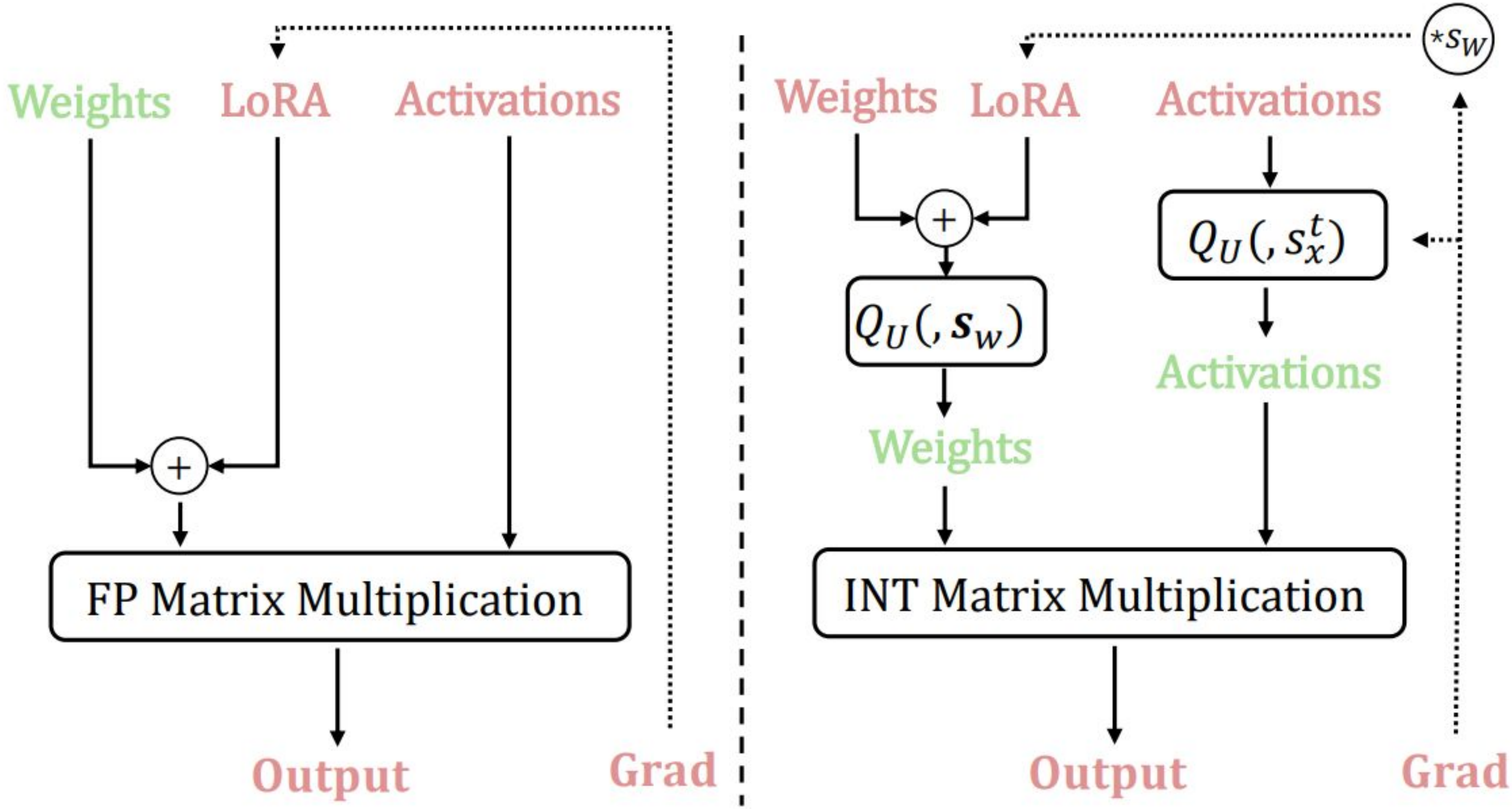
QLoRA — Quantized LoRA

- **Key Idea:** Fine-tune large language models in 4-bit precision using LoRA adapters.
- **Scalability:** Enables fine-tuning of 65B parameter models on a single 48GB GPU.
- **Efficiency:** Highly memory-efficient with no significant performance loss (Dettmers et al., 2023).
- **Techniques:**
  - Double quantization
  - Paged optimizers
  - NF4 (NormalFloat 4-bit) quantization format

QLoRA — Quantized LoRA



QLoRA — Quantized LoRA



(a) QLoRA layer (b) QALoRA layer

Applications of LoRA / QLoRA

- **Instruction tuning for resource-constrained settings:**
  - Enables fine-tuning large models on modest hardware (e.g., consumer GPUs, laptops).
  - Used for domain adaptation, personalization, and rapid prototyping.
- **Popular fine-tuned models:**
  - **Alpaca, Guanaco, Vicuna, Mistral:** All leverage LoRA/QLoRA for efficient instruction tuning.
- **Model merging and compositionality:**
  - Merge multiple LoRA adapters for multi-domain or multi-task capabilities.
  - Compose adapters for new tasks without retraining the base model.

Limitations of Fine-Tuning and RLHF

Limitations of Fine-Tuning and RLHF

- **RLHF:**
  - Reward models can be misaligned with true human preferences.
  - Training loops are computationally expensive and complex.
- **DPO:**
  - Still depends on high-quality human preference data.
- **LoRA:**
  - Only updates a small subset of parameters; may miss global interactions.
- **Quantization:**
  - Requires careful selection and tuning of quantization formats (e.g., NF4).

Future Directions

- **RLAIF (RL with AI Feedback):** Automate human labelers using strong AI models for preference data.
- **Constitutional AI:** Use rule-based alignment and self-critique to guide model behavior (e.g., Anthropic's approach).
- **Open RLHF Datasets:** Promote better community sharing and reproducibility with open preference datasets.
- **Multi-modal Alignment:** Extend alignment techniques to vision, audio, and code generation tasks.
- **Reward Hacking Defense:** Develop robust reward functions to prevent models from exploiting reward loopholes.

Summary

- Fine-tuning enables specialization and alignment of LLMs.
- Supervised Fine-Tuning (SFT) is the foundational step.
- RLHF incorporates human preference alignment using PPO.
- DPO streamlines the process with a direct preference loss.
- LoRA and QLoRA make adaptation efficient and accessible.
- The field is rapidly evolving with ongoing innovation.

References

[1] Ouyang, L., Wu, J., Jiang, X., et al. (2022). *Training language models to follow instructions with human feedback*. InstructGPT. https://arxiv.org/abs/2203.02155

[2] Rafailov, R., Metz, L., Ba, J., et al. (2023). *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*. https://arxiv.org/abs/2305.18290

[3] Dettmers, T., Pagnoni, A., Holtzman, A., et al. (2023). *QLoRA: Efficient Finetuning of Quantized LLMs*. https://arxiv.org/abs/2305.14314

[4] Bai, Y., Kadavath, S., Kundu, S., et al. (2022). *Training a Helpful and Harmless Assistant with RLHF*. https://arxiv.org/abs/2204.05862

[5] Zhang, S., Roller, S., Goyal, N., et al. (2022). *OPT: Open Pre-trained Transformer Language Models*. https://arxiv.org/abs/2205.01068

[6] Hu, E. J., Shen, Y., Wallis, P., et al. (2021). *LoRA: Low-Rank Adaptation of Large Language Models*. https://arxiv.org/abs/2106.09685

[7] Anthropic (2023). *Constitutional AI: Harmlessness from AI Feedback*. https://www.anthropic.com/

[8] HuggingFace. *RLHF Course*. https://huggingface.co/learn/rlhf-course

References

أكاديمية كاوست
KAUST ACADEMY

LMH
Lady Margaret Hall

[9] Stanford CRFM. *CRFM Slides*. https://crfm.stanford.edu/

Credits

Dr. Prashant Aparajeya

Computer Vision Scientist — Director(AISimply Ltd)

p.aparajeya@aisimply.uk

This project benefited from external collaboration, and we acknowledge their contribution with gratitude.