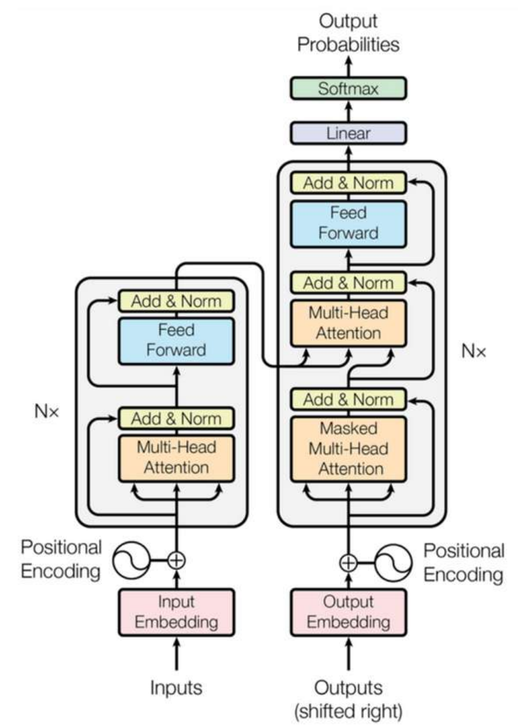
LARGE LANGUAGE MODEL AS REINFORCEMENT LEARNING AGENT

Minha Hwang



TWO STAGES OF LARGE LANGUAGE MODEL DEVELOPMENT

Pre-Training: Base LLM (GPT3)

predict next word, based on text training data

- **Self-supervised**

Once upon a time, there was a unicorn
that lived in a magical forest with
all her unicorn friends

What is the capital of France?
What is France's largest city?
What is France's population?
What is the currency of France?

Post-Training: Instruction Tuned LLM (ChatGPT)

Tries to **follow instructions**

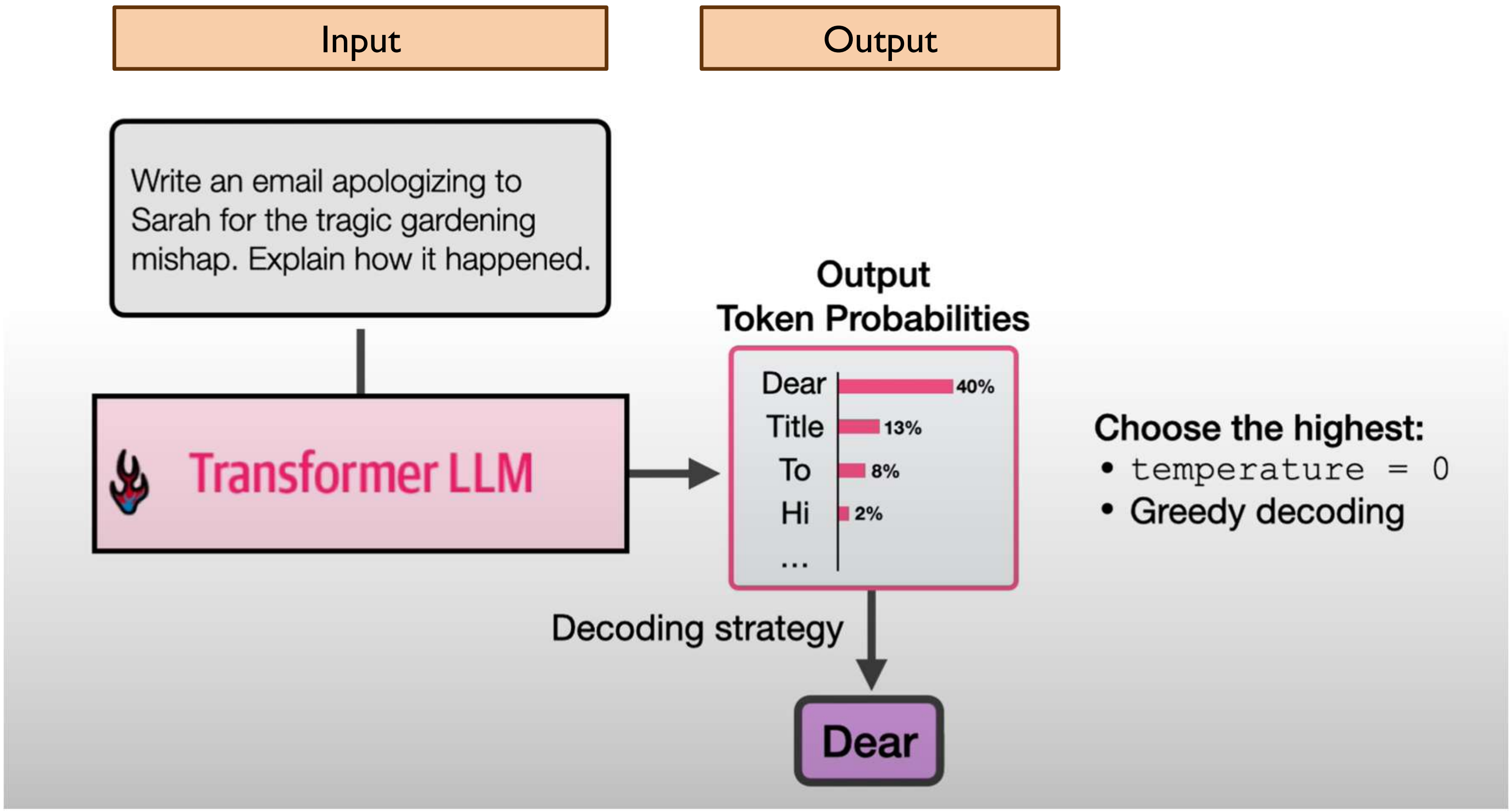
Fine-tune on **instructions and good response pairs**

- **Human labeled data**: Instruction – Response Pair
- **SFT**: Supervised Fine-tuning
- **RLHF** (Reinforcement Learning with Human Feedback) or **DPO** (Direct Preference Optimization)

What is the capital of France?
The capital of France is Paris.

Source: DeepLearning.AI – ChatGPT Prompt Engineering

TRANSFORMER LLM: INPUT AND OUTPUT



Source: DeepLearning.AI – How Transformer LLM Works

AUTOREGRESSIVE LANGUAGE MODELS: PROBABILITY KERNEL

- A language model is a **probability kernel μ** given a prefix of words: $\mu: X \rightarrow Pr(Y)$
  - Stochastic in nature: **A same prefix X** can give a **random output** sampled from a probability distribution μ_X(i.e., generative) → A key reason for factual inaccuracy, inconsistency or hallucination (making stuff up)

- A language model calculates $Pr(s)$ given a sequence of words: $s = (w_1, w_2, \ldots\ldots., w_{T-1}, w_T)$

- An autoregressive language model calculates this **conditional on a previous sequence of words**:

$$Pr(s) = Pr(w_1, w_2, \ldots\ldots., w_{T-1}, w_T)$$
$$= \prod_{t=1}^{T} Pr(w_t | w_1, w_2, \ldots\ldots., w_{t-1})$$

  - **Next-word prediction**: Given a prefix $(w_1, w_2, \ldots\ldots., w_{t-1})$, calculate the probability of the next word w_t (Conceptually same to time series with path dependence)

Source: Prof. Kyunghyun Cho

AUTOREGRESSIVE LANGUAGE MODELS: SIMPLE EXAMPLE

- 4-word sentence example: "I am a student"

$$\boldsymbol{Pr(s) = Pr(w_1, w_2, w_3, w_4) = Pr(w_1) \times Pr(w_2|w_1) \times Pr(w_3|w_1, w_2) \times Pr(w_4|w_1, w_2, w_3)}$$

- All you need is **"counting"** (if there are large amounts of data)

$$\boldsymbol{Pr(w_2|w_1) = \frac{count(w_1, w_2)}{count(w_1)}} \longrightarrow \text{2-grams (Bigrams)}$$

$$\boldsymbol{Pr(w_3|w_1, w_2) = \frac{count(w_1, w_2, w_3)}{count(w_1, w_2)}} \longrightarrow \text{3-grams (Trigrams)}$$

$$\boldsymbol{Pr(w_4|w_1, w_2, w_3\,) = \frac{count(w_1, w_2, w_3, w_4)}{count(w_1, w_2, w_3)}} \longrightarrow \text{4-grams}$$

- Problems:
  - This requires **a lot of space (RAM)**
  - Count-based language models **cannot generalize:** A certain sentence **does not appear** in the corpus

Source: Prof. Kyunghyun Cho

PRE-TRAINING: SELF-SUPERVISED LEARNING

Input

- A sequence of tokens (encoded words)

Loss Function

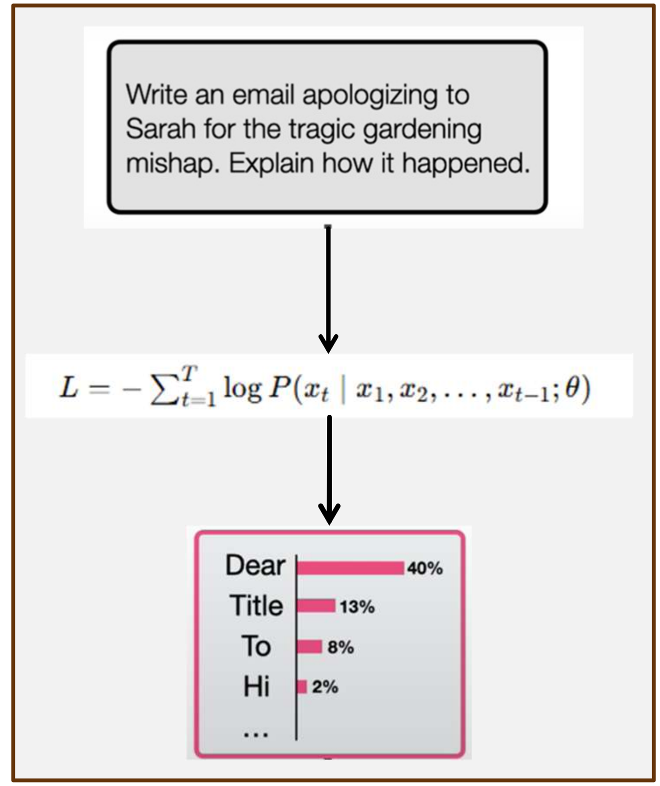
- Minimize the **negative log-likelihood** of the predicted token given the preceding tokens: **Cross-entropy Loss**
- Mathematically, the loss L for a sequence of tokens $(x_1, x_2, \ldots, x_T)$ is:

Output

- **Probability distribution** over the **vocabulary** (~30,000)
- Deterministic

Dataset Size

- Neural scaling law: The dataset size (D) should **scale proportionally** with the model size (i.e., linear)
  - e.g., GPT-3: 175B parameters, 300B tokens



Source: DeepLearning.AI – How Transformer LLM Works; Chinchilla Scaling: Hoffmann, et al, 2022

PRE-TRAINED LLM EVAL - PERPLEXITY: PREDICTIVE ACCURACY (2/2)

Example Calculation: "The cat sat on the mat"

- P("The") = 0.2
- P("cat"|"The") = 0.1
- P("sat"|"The cat") = 0.15
- P("on"|"The cat sat") = 0.3
- P("the"|"The cat sat on") = 0.25
- P("mat"|"The cat sat on the") = 0.05

$$\text{Perplexity}(PP) = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(w_i \mid w_{<i})\right)$$

First, calculate the average negative log probability:

$$-\frac{1}{6}\left(\log(0.2) + \log(0.1) + \log(0.15) + \log(0.3) + \log(0.25) + \log(0.05)\right) \approx 1.8992$$

Then, exponentiate to find perplexity:

$$PP = \exp(1.8992) \approx 6.68$$

This means the model, on average, considers about 6.68 possible next words, indicating its uncertainty in prediction.

POST-TRAINING: SUPERVISED FINE TUNING

Input

- Prompt-Desired Response Pair Dataset: {Prompt 1, Desired Response 1}, {Prompt 2, Desired Response 2}

Loss Function

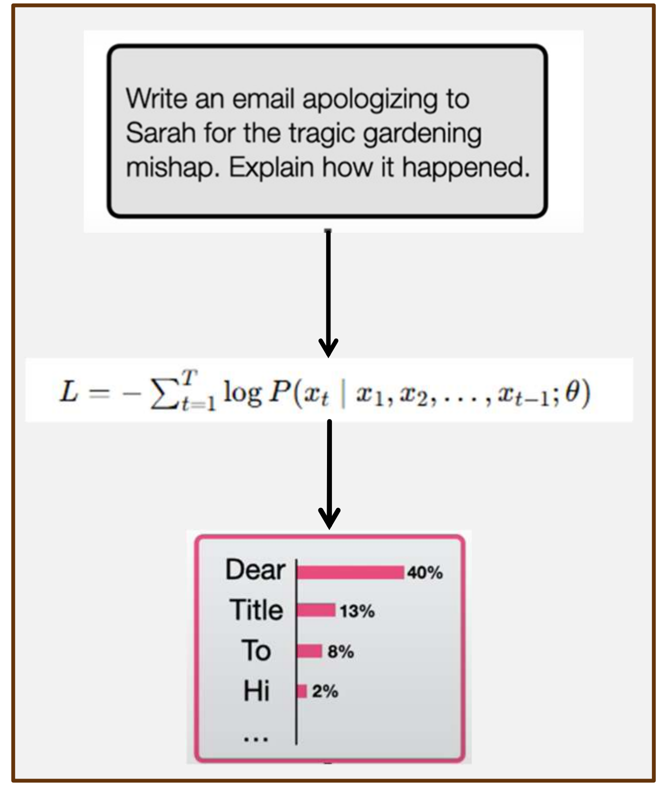
- **Cross-entropy Loss**
- Generated output is compared to the desired response using a loss function

Output

- **Modified Probability Distribution** over the **vocabulary** (~30,000)

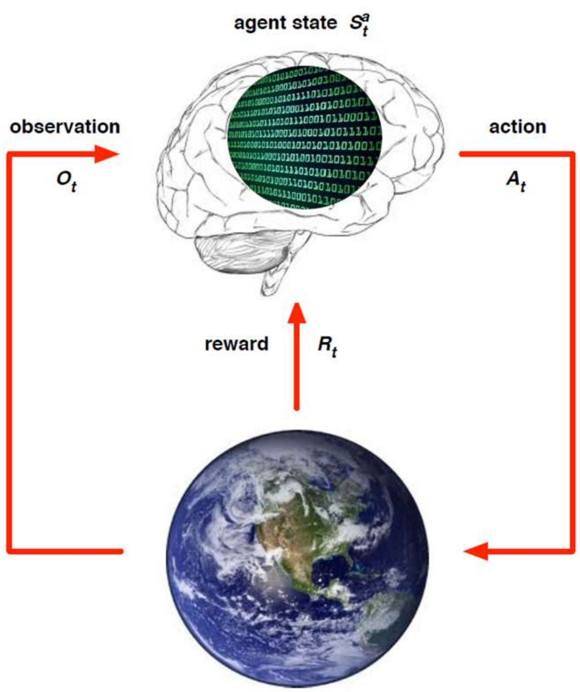
Dataset Size

- Task-specific models**:** **Less than 0.5%** of the original dataset: **Generalize to other data**, not just in fine-tuning data
- InstructGPT: 13,000 prompt-response pairs human labelers



Source: DeepLearning.AI – How Transformer LLM Works; Chinchilla Scaling: Hoffmann, et al, 2022

REINFORCEMENT LEARNING: AGENT AND ENVIRONMENT



- At each step t, **the agent**:
  - Execute action A(t)
  - Receives observation O(t)
  - Receives scalar reward R(t)

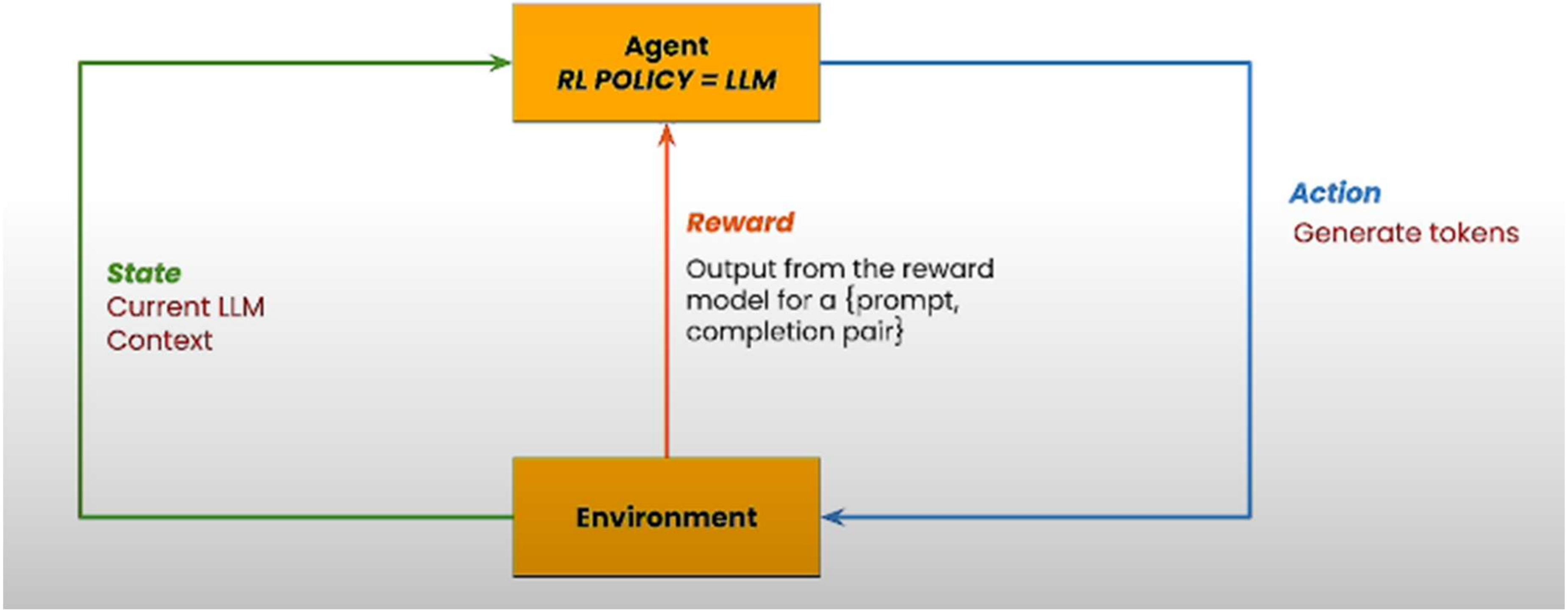
- The **environment**:
  - Receives action A(t)
  - Emits observation O(t+1)
  - Emits scalar reward R(t+1)

- **t increment** at environment step

- **Sequential Decision Making**
- Reward Hypothesis: All goals can be described by the maximization of **expected cumulative reward** **(scalar)**

Source: David Silver – Reinforcement Learning

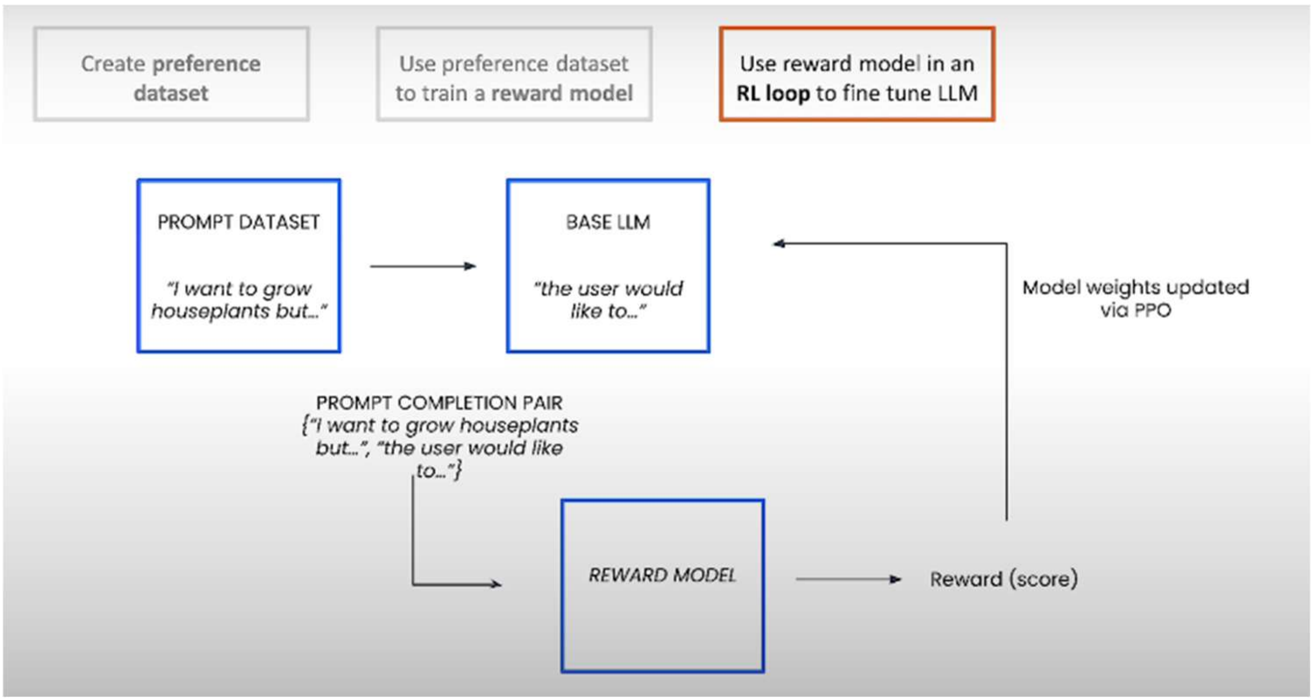
POST-TRAINING: RLHF TO FINE TUNE LLM



- **Proximal Policy Optimization (PPO)**
- **DeepSeek: GRPO**

Source: https://learn.deeplearning.ai/courses/reinforcement-learning-from-human-feedbac

POST-TRAINING: RLHF



- **Sequential Decision Making**
- Reward Hypothesis: All goals can be described by the maximization of **expected cumulative reward** **(scalar)**

Source: David Silver – Reinforcement Learning

POST-TRAINING: RLHF – REWARD MODEL – TRAINING (1/3)

Input

- Preference Dataset - Pairwise {Prompt, Winning Candidate, Losing Candidate, Choice}
- Annotated by Human (Subjective)

Loss Function

- Minimize Pairwise Loss

$$\mathcal{L}(\theta) = -\frac{1}{\binom{K}{2}} \sum_{(x,y_w,y_l)} \log\left(\sigma\left(r_\theta(x,y_w) - r_\theta(x,y_l)\right)\right)$$

Here:

- x is the prompt.
- y_w and y_l are the preferred and less preferred responses, respectively.
- $r_\theta(x,y)$ is the reward model's score for a given prompt-response pair.
- σ denotes the sigmoid function.
- K is the number of responses ranked by human annotators for each prompt.

Dataset Size

- 10K – 100K range
- InstructGPT:
  - Reward Model Dataset: ~ 33,000 examples. Human labelers ranked multiple responses to the same prompt

Source: DeepLearning.AI – How Transformer LLM Works; Chinchilla Scaling: Hoffmann, et al, 2022

POST-TRAINING: RLHF – REWARD MODEL – INFERENCE (2/3)

Input

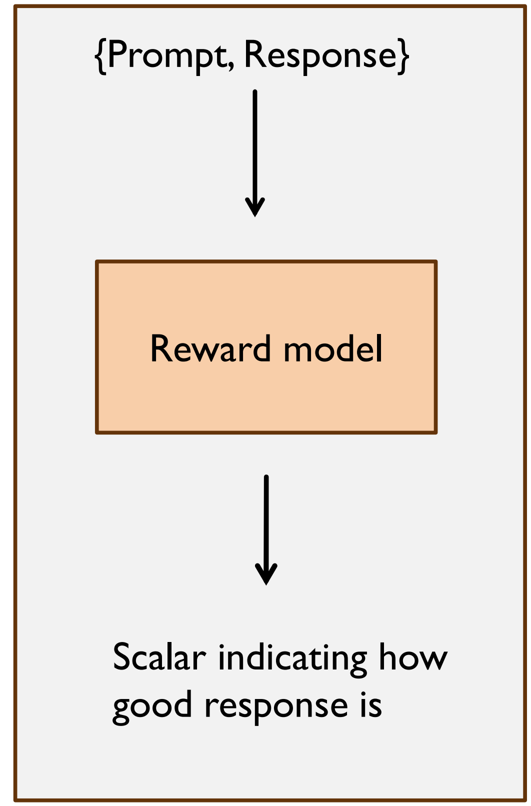
- Prompt Dataset - {Prompt, Response}

Model

- Reward Model (LLM), trained with Preference Dataset: {Prompt, Winning Candidate, Losing Candidate, Choice}

Output

- Scalar indicating how good response is



Source: DeepLearning.AI – How Transformer LLM Works; Chinchilla Scaling: Hoffmann, et al, 2022

POST-TRAINING: RLHF – PROXIMAL POLICY OPTIMIZATION (3/3)

Input

- **Proximal Policy Optimization (PPO) Dataset**: InstructGPT ~31,000 prompts used to generate responses without human intervention during training, {Prompt}

Loss Function

Policy Optimization: The language model is fine-tuned using reinforcement learning to maximize the rewards predicted by the reward model. A common approach is to use Proximal Policy Optimization (PPO) with a loss function that balances achieving high reward and maintaining the model's output distribution close to the original model to prevent divergence:

$$L(\phi) = \mathbb{E}_{(x,y)\sim D_{\pi_\phi}} \left[r_\theta(x, y) - \beta \log \left(\frac{\pi_\phi(y|x)}{\pi_{\text{SFT}}(y|x)} \right) \right]$$

In this equation:

- π_ϕ is the policy of the fine-tuned model.
- π_{SFT} is the policy of the supervised fine-tuned model before reinforcement learning.
- β is a scaling factor that controls the strength of the penalty for deviating from the original policy.

Source: DeepLearning.AI – How Transformer LLM Works; Chinchilla Scaling: Hoffmann, et al, 2022

CONNECTING RL AND LLMS (1/2)

	Connection in LLM
Sequential Decision Making	• Next token prediction to maximize prediction accuracy (pre-training) and rewards (post-training) over vocabulary (discrete set)
State	• Input prompt and previous generated tokens
Action	• Token chosen from the vocabulary
Reward	• Prediction accuracy + rewards from aligning with preference (weighted)

$$r(s_t, y_t) = \lambda_1 \log \pi(y_t \mid s_t) + \lambda_2 \,\mathrm{Metric}(s_t, y_t),$$

Source: [Large Language Models as Reinforcement Learning Agents in Token Space: A Theoretical Framework by Miquel Noguer I Alonso :: SSRN]

CONNECTING RL AND LLMS (2/2): APPLICABILITY OF RL RESEARCH

	Description
Hierarchical Decision	• Motivated by AlphaStar's multi-scale decision-making • High-level planning and detailed token generation
Self-dialogue	• Motivated by self-play training in AlphaGo, AlphaStar • Self-dialogue: Models can engage in dialogue, critiquing and improving each other's outputs
Adaptive decoding	• Develop decoders that balance exploration and exploitation based on state uncertainty
Hybrid Models	• Combine maximum likelihood training with RL-based fine-tuning

Source: [Large Language Models as Reinforcement Learning Agents in Token Space: A Theoretical Framework by Miquel Noguer I Alonso :: SSRN]

Reference

- **Transformer Large Language Model**
  - https://learn.deeplearning.ai/courses/how-transformer-llms-work
  - https://learn.deeplearning.ai/courses/attention-in-transformers-concepts-and-code-in-pytorch

- **Neural Sampling Law**
  - [2203.15556] Training Compute-Optimal Large Language Models

- **Reinforcement Learning with Human Feedback**
  - **InstructGPT:** [2203.02155] Training language models to follow instructions with human feedback
  - Paper page - Fine-Tuning Language Models from Human Preferences

- **Large Language Model and Reinforcement Learning Connection**
  - Large Language Models as Reinforcement Learning Agents in Token Space: A Theoretical Framework by Miquel Noguer I Alonso :: SSRN
  - [2306.07929] Large Language Models Are Semi-Parametric Reinforcement Learning Agents
  - WindyLab/LLM-RL-Papers: Monitoring recent cross-research on LLM & RL on arXiv for control. If there are good papers, PRs are welcome.
  - [2106.01345] Decision Transformer: Reinforcement Learning via Sequence Modeling
  - kzl/decision-transformer: Official codebase for Decision Transformer: Reinforcement Learning via Sequence Modeling.