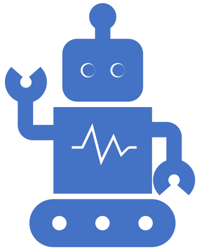
UiO : University of Oslo

institutt for informatikk



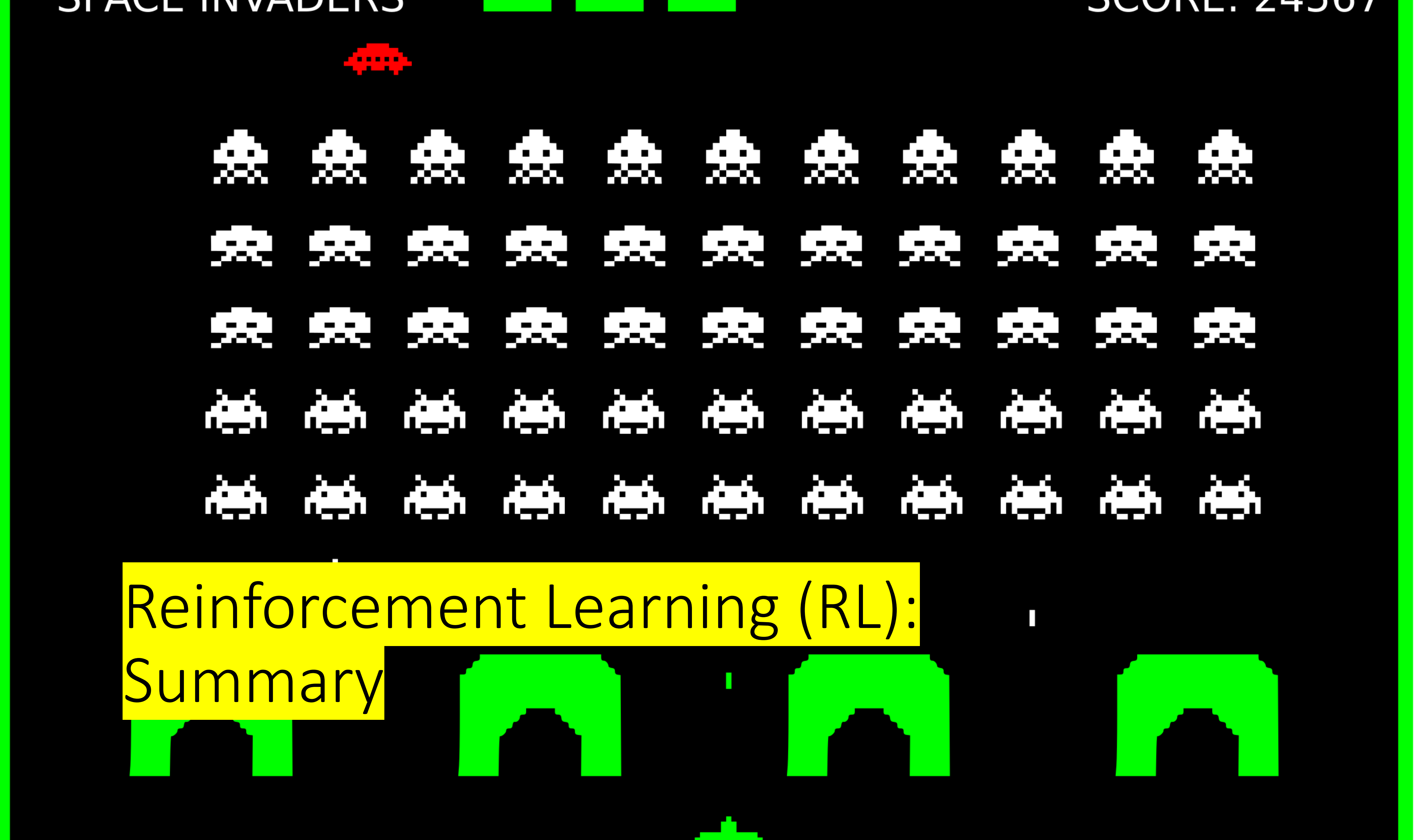
IN3050/IN4050 – 2023

Reinforcement Learning

Kai Olav Ellefsen

Today’s plan: Reinforcement Learning

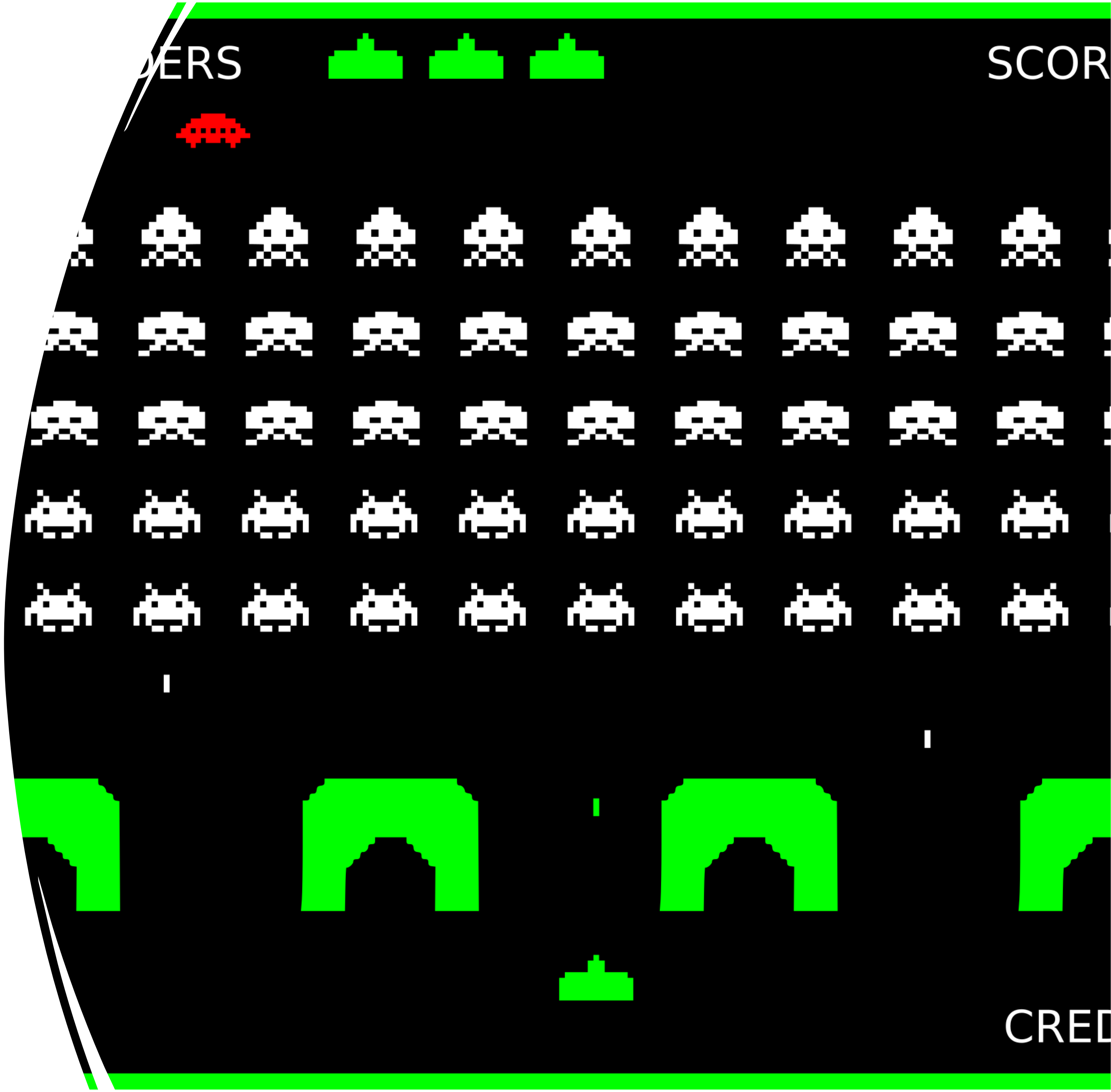
- Summary
- Quiz
- Diving deeper: Reinforcement Learning from Human Feedback (RLHF)
- Dialog/discussion
- Questions/Answers



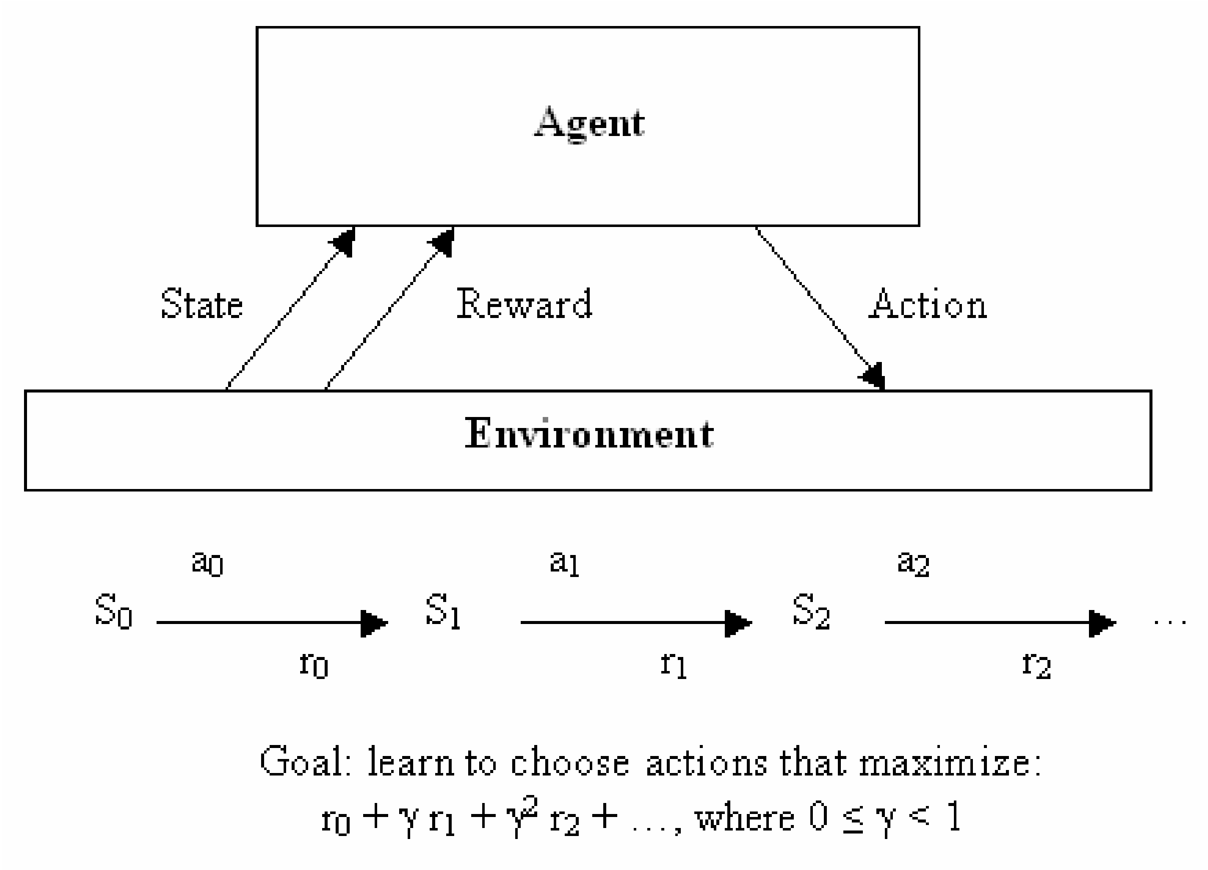
Reinforcement Learning (RL): Summary

Reinforcement Learning

- Supervised/Unsupervised Learning:
  - Data are pre-defined
  - Minimize loss function to fit the data
- Reinforcement Learning:
  - More active: Explore to find the data yourself
  - Less direct feedback: Use the reward to guide learning, but it is imprecise



The reinforcement learning problem



Learning is guided by the reward

- An infrequent numerical feedback indicating how well we are doing
- Problems:
  - The reward does not tell us *what we should have done*
  - The reward may be *delayed* – does not always indicate *when* we made a mistake.

Image source: Stable Diffusion

Maximizing total reward

- Total reward:

$$R = \sum_{t=0}^{N-1} r_{t+1}$$

- Future rewards may be uncertain -> We care more about rewards that come soon
- Solution: Discount future rewards:

$$R = \sum_{t=0}^{\infty} \gamma^t \, r_{t+1}, \qquad 0 \leq \gamma \leq 1$$

What do we need to estimate the next state and reward?

- If we only need to know the current state, this problem has the *Markov property*.

$$P(r_t = r', s_{t+1} = s' | \, s_0, a_0, r_0, \dots, r_{t-1}, s_t, a_t) =$$
$$P(r_t = r', s_{t+1} = s' | \, s_t, a_t)$$

Value

- The expected future reward is known as the *value*
- Two ways to compute the value:
  - The value of a state – V(s) – averaged over all possible actions in that state
  - The value of a state/action pair Q(s,a)
- Q and V are initially unknown, and learned iteratively as we gain experience

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

Action selection

- The function deciding which action to take in each state is called the policy, π. Examples:
  - Greedy: Always choose most valuable action
  - ϵ-greedy: Greedy, except small probability (ϵ) of choosing the action at random
- The q-learning we just saw is an example of *off-policy learning*:

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

On-policy vs off-policy learning

- Q-learning (off-policy):

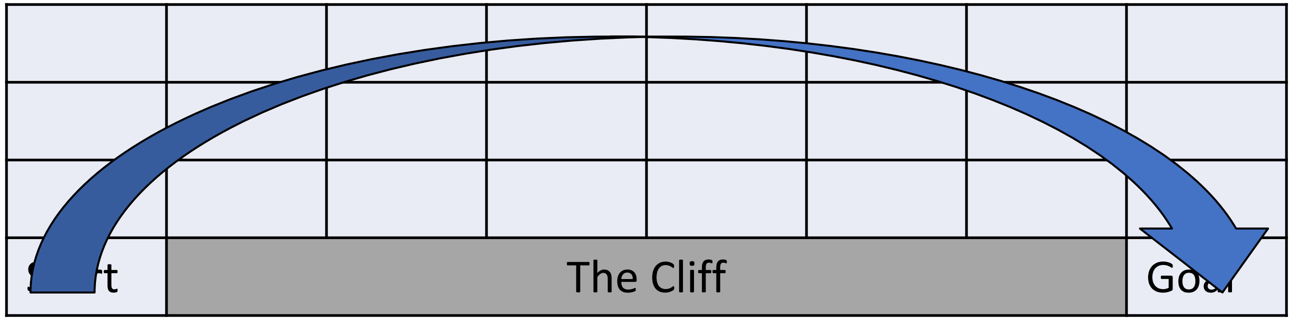
$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \mu \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

- Sarsa (on-policy):

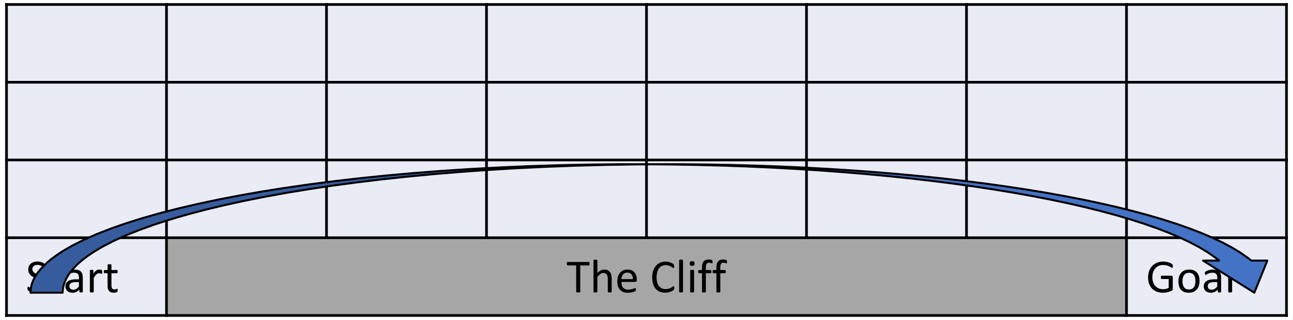
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \mu[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

On-policy vs off-policy

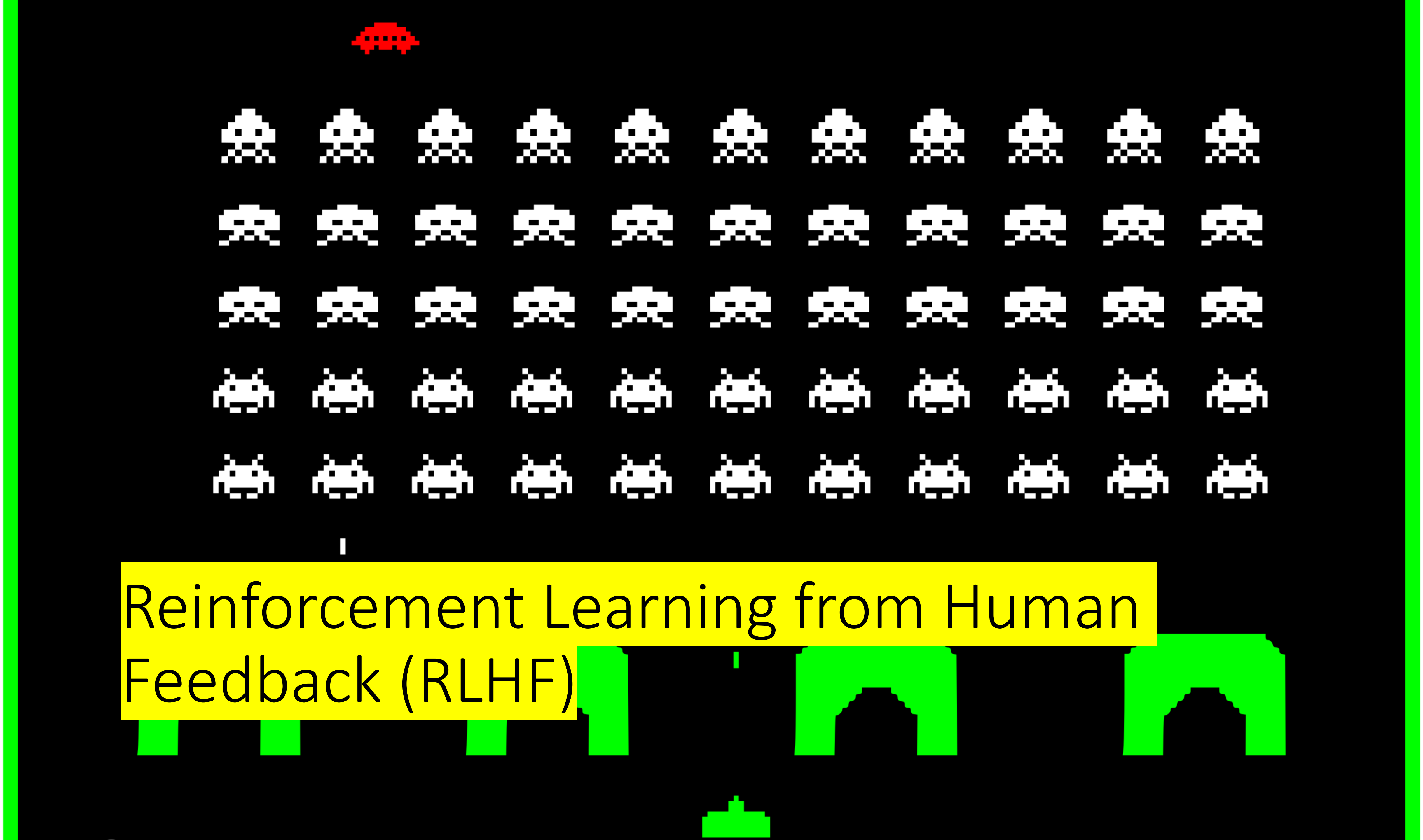
- sarsa (on-policy):



- Q-learning (off-policy):



Reinforcement Learning from Human Feedback (RLHF)



Why was Reinforcement Learning important for training chatGPT?

- Sources:
  - https://huggingface.co/blog/rlhf
  - https://openai.com/research/instruction-following

Reinforcement Learning from Human Feedback

LE

Can you explain to me how RL with human feedback works for language models like I am five?

Sure! So, imagine you have a robot named Rufus who wants to learn how to talk like a human. Rufus has a language model that helps him understand words and sentences.

First, Rufus will say something using his language model. For example, he might say "I am a robot."

Then, a human will listen to what Rufus said and give him feedback on whether it sounded like a natural sentence a human would say. The human might say, "That's not quite right, Rufus. Humans don't usually say 'I am a robot.' They might say 'I'm a robot' or 'I am a machine.'"

Rufus will take this feedback and use it to update his language model. He will try to say the sentence again, using the new information he received from the human. This time, he might say "I'm a robot."

The human will listen again and give Rufus more feedback. This process will continue until Rufus can say sentences that sound natural to a human.

Over time, Rufus will learn how to talk like a human thanks to the feedback he receives from humans. This is how language models can be improved using RL with human feedback.

ChatGPT is built on top of the language model GPT-3

- Simplified view of GPT3-training in next slides

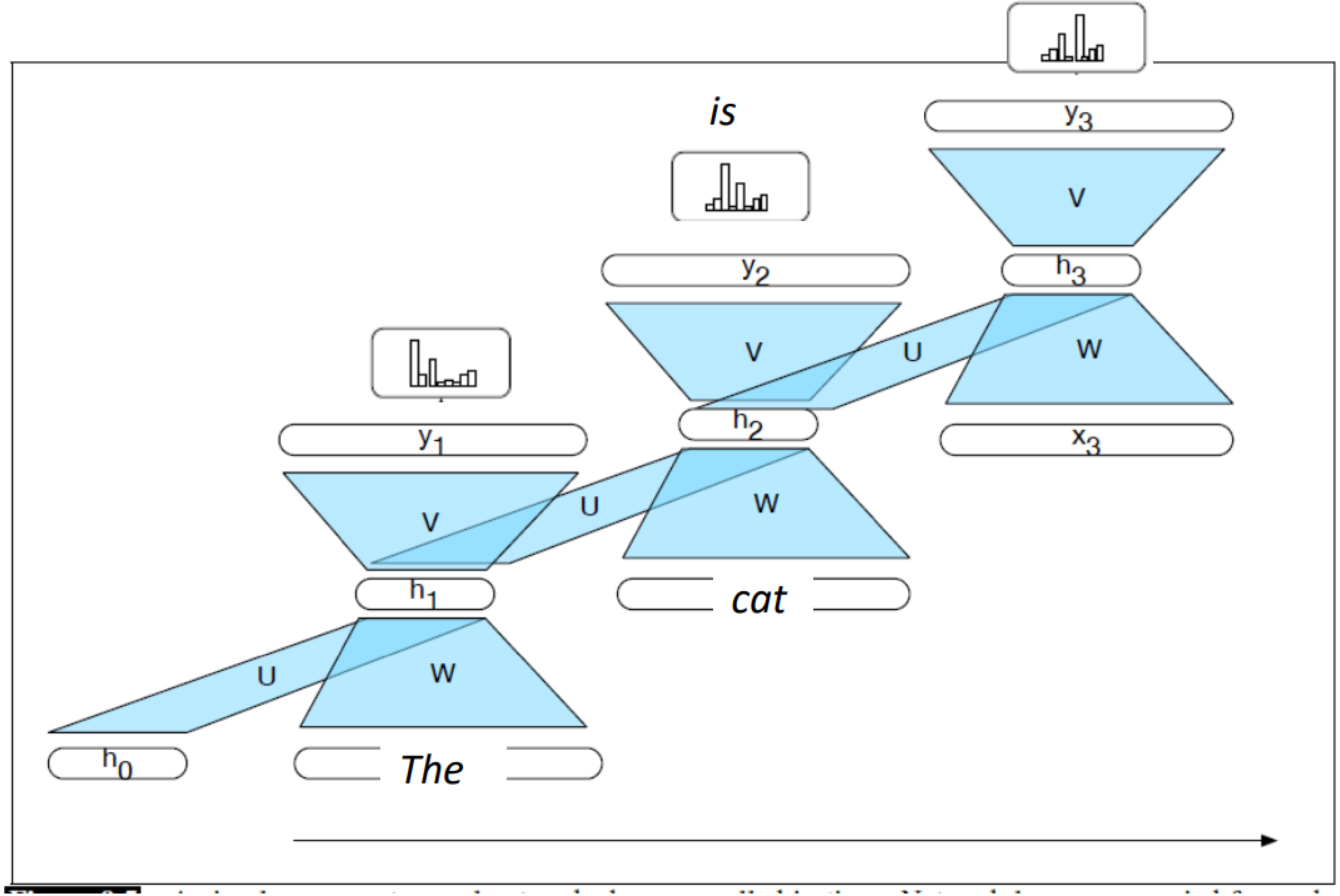
Lecture 5: Language model

The cat is on the …

- Task: Predict the next word!
- Labels:
  - A vocabulary of words:
    - *aardvark, …, mat, …, roof, …*
    - E.g., 100,000 words/labels
- Domain:
  - Finite sequences of words

- Properties:
  - 1 billion training instances
  - Supervised learning
    - But you do not have to hand-label the training data.

RNN Language model



GPT-3 was good at modelling language but not useful as a chatbot

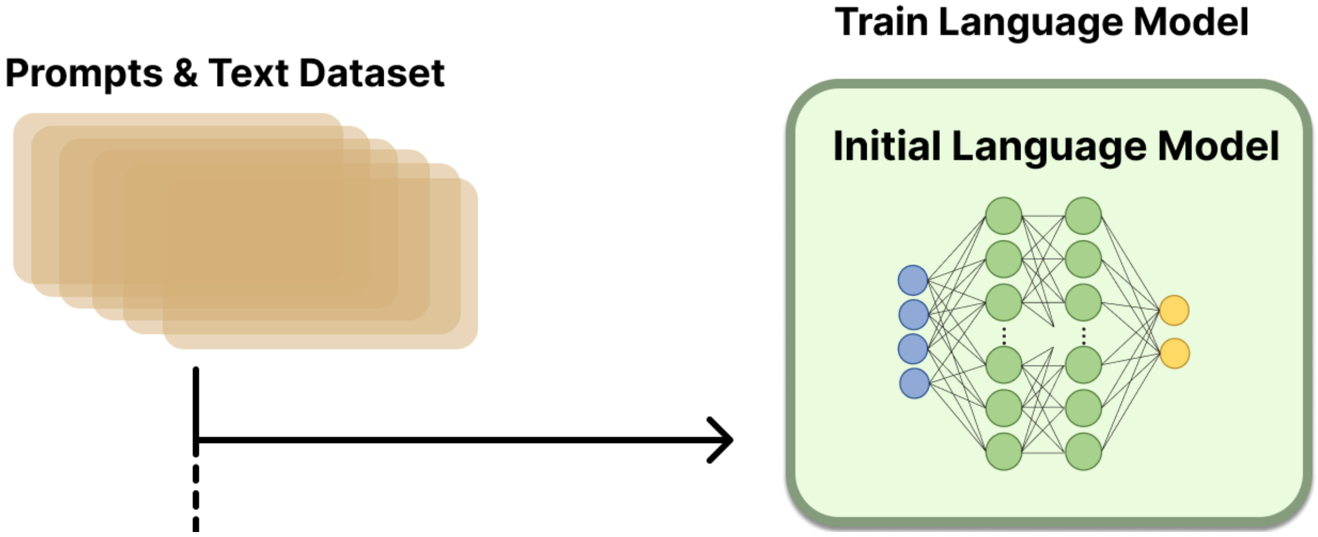
Prompt *Explain the moon landing to a 6 year old in a few sentences.*

Completion GPT-3

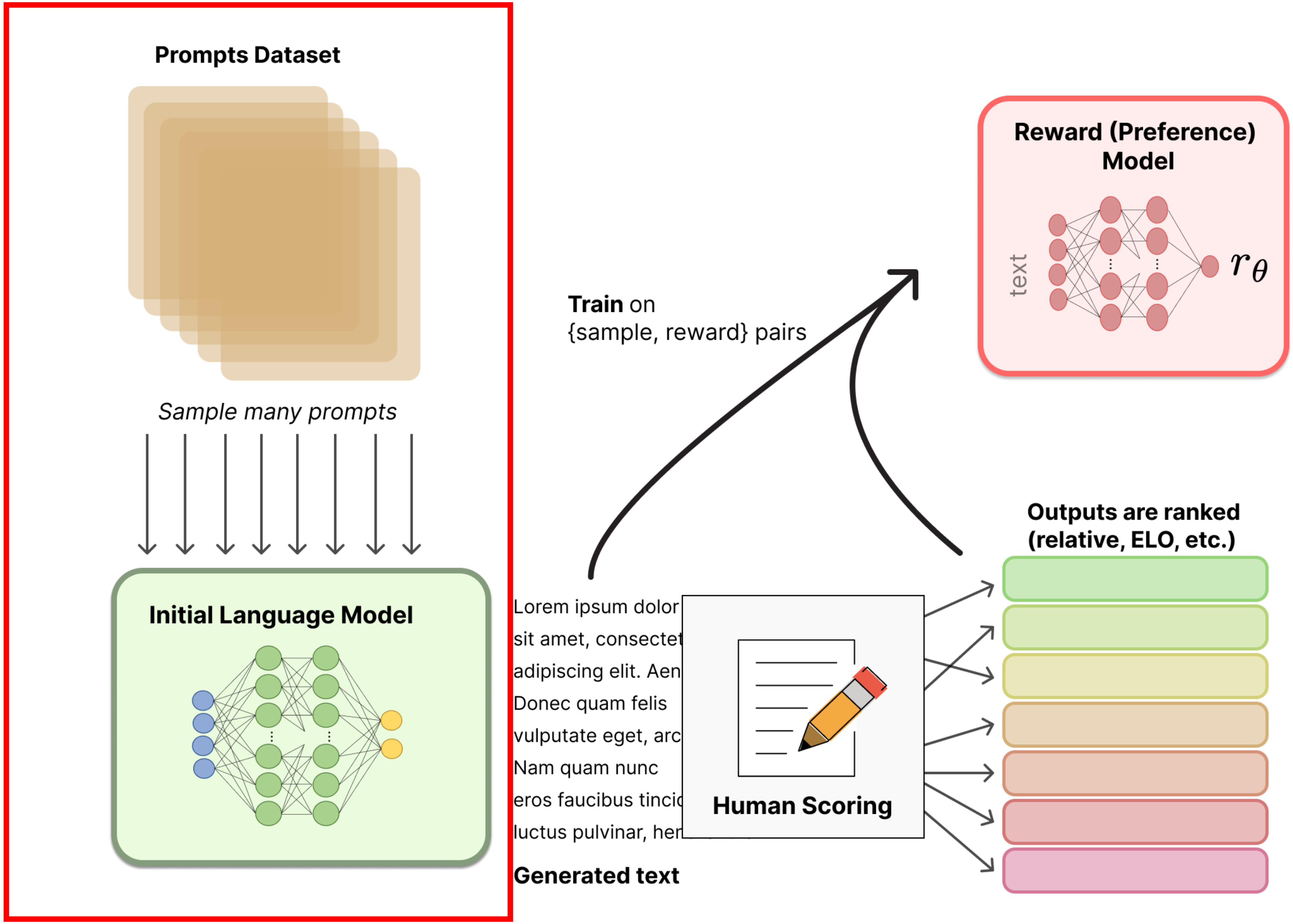
```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

A Language Model is not a good chat bot by default

- Predicting next words is not quite the same as giving a good answer to a question
- We can «fine tune» (continue training) to build chat bot skills on top of the language understanding of GPT-3



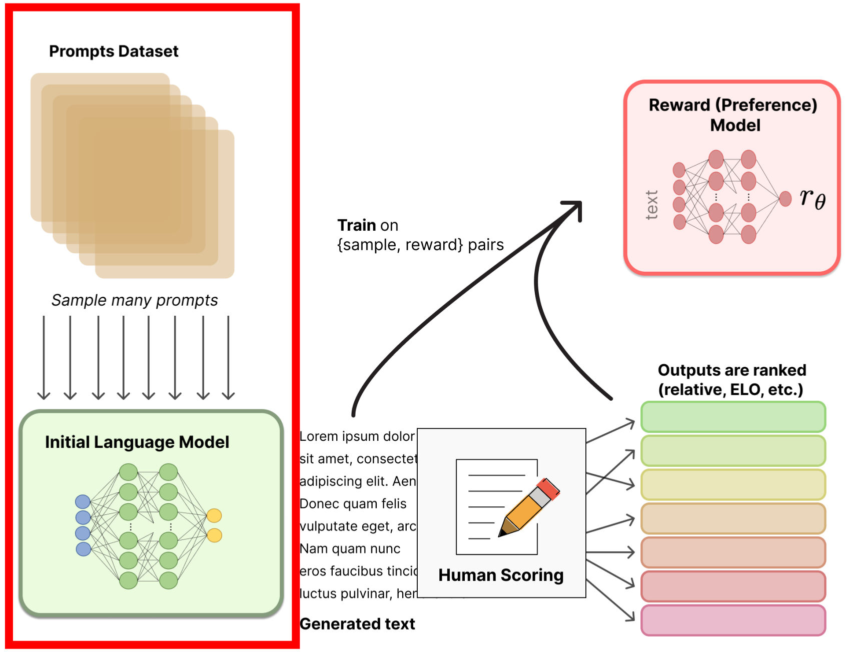
Source: https://huggingface.co/blog/rlhf



Source: https://huggingface.co/blog/rlhf

Step 1: Build a prompt dataset

- Prompt: The input (question) we send to chatGPT
  - E.g. «explain the moon landing to a 6-year old»
- How to find lots of relevant prompts?
- OpenAI had lots available from human users of GPT-3



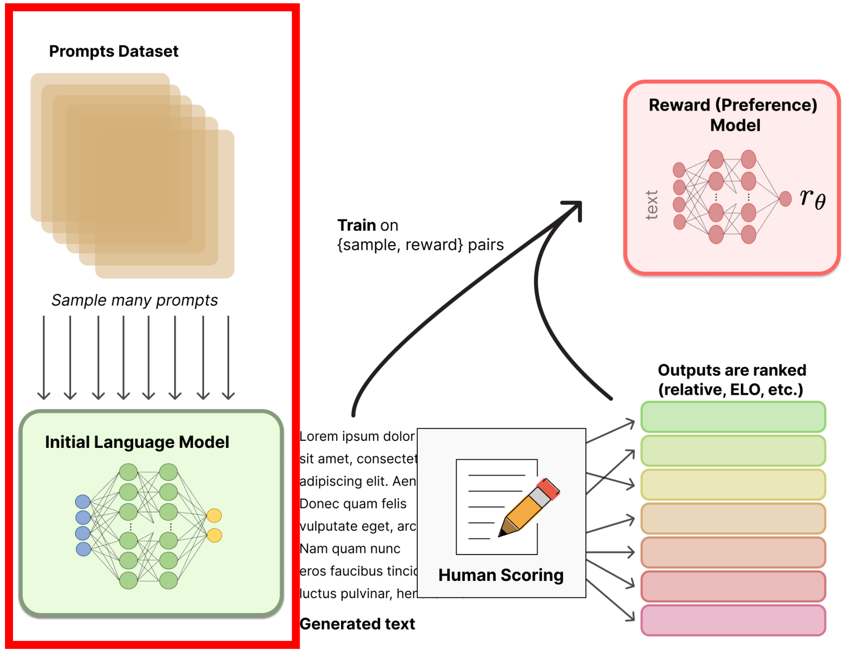
Step 2: Pass these prompts through GPT-3

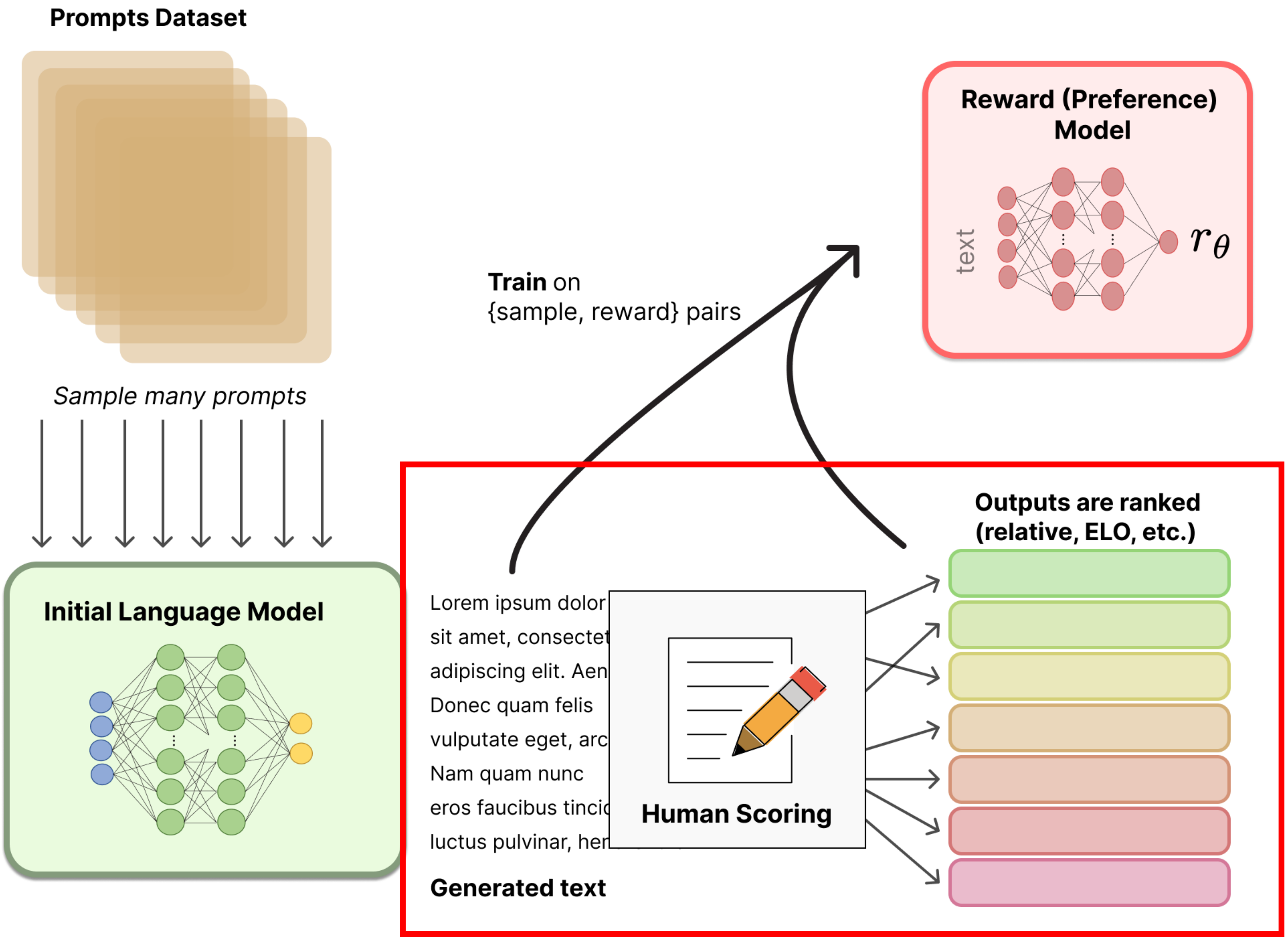
- GPT-3 will then generate the sensible but non-chat-friendly output that we want to adjust
- Have GPT-3 and perhaps other models generate *multiple* such outputs for each prompt

Prompt *Explain the moon landing to a 6 year old in a few sentences.*

Completion GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

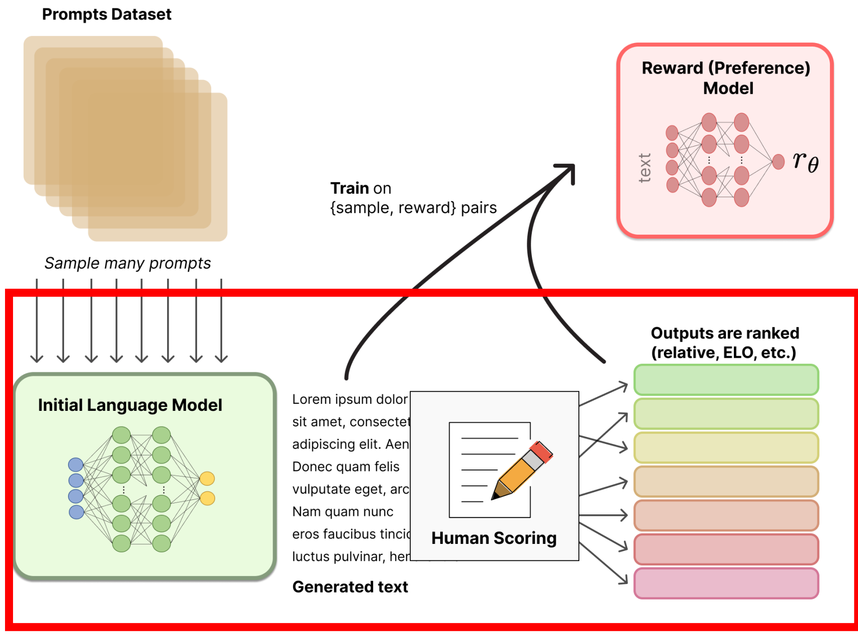


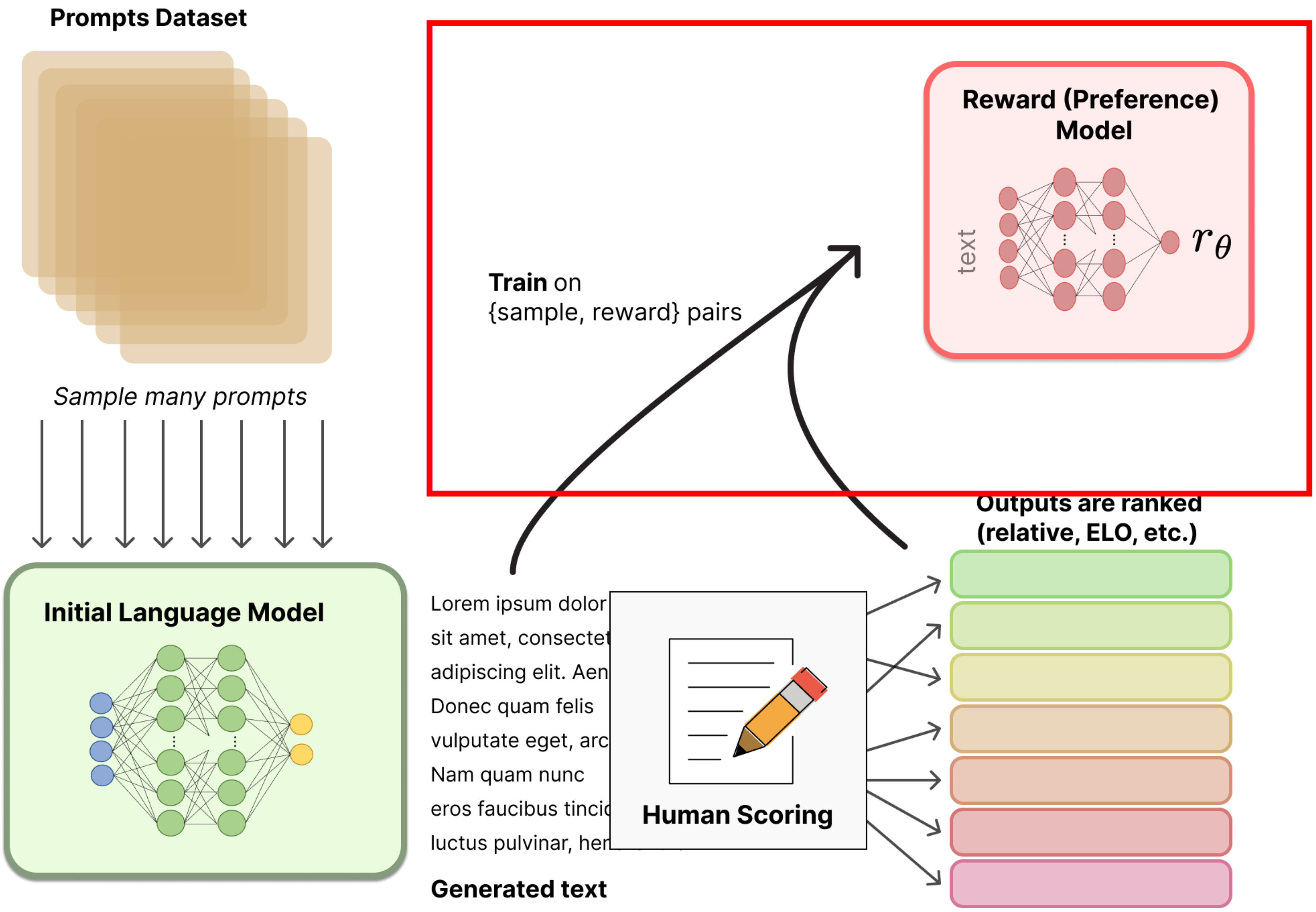


Source: https://huggingface.co/blog/rlhf

Part 3: Human ranking

- Our models have provided N answers to our prompt:
  - 1) «The moon landing was accomplished by…»
  - 2) «The moon is similar to a cheese…»
  - 3) «Explain the theory of gravity…»
- Ask human users to rank these answers from 1 (best) to N (worst)
- These rankings can serve as reward-data for RL!

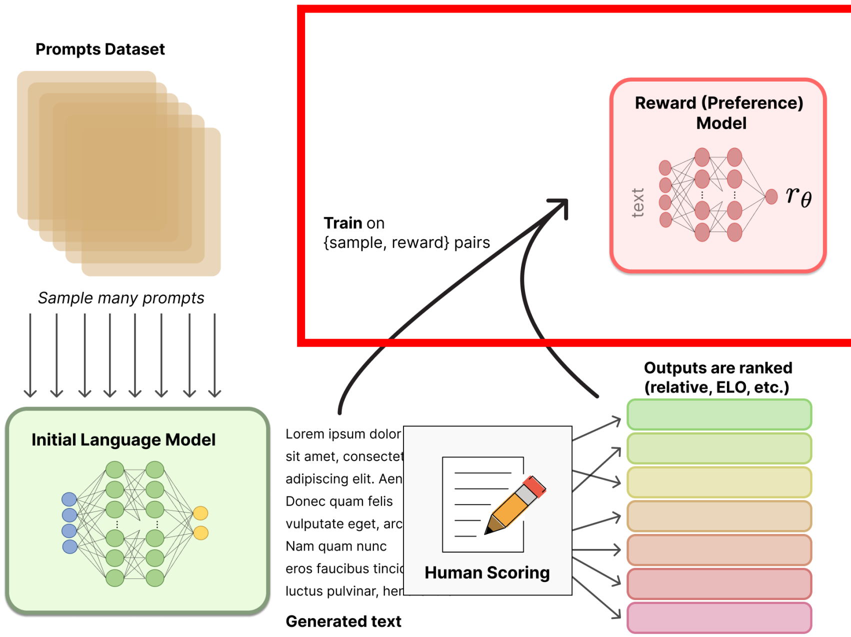




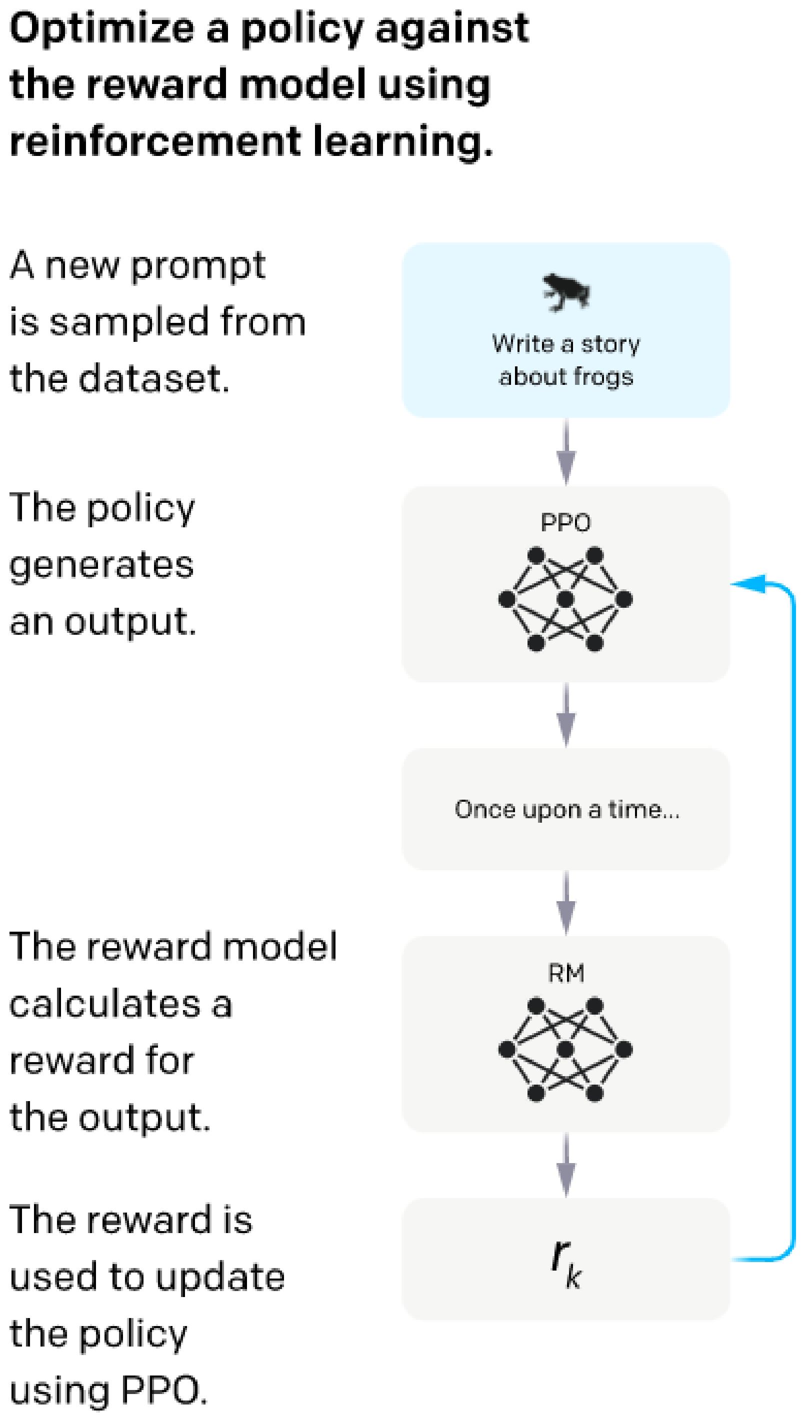
Source: https://huggingface.co/blog/rlhf

Part 4: Train a Reward Model

- For each prompt, we have N candidate outputs, each with a reward.
- Train a Reward Model that predicts what rewards humans would give to different outputs
  - Input to RW: The text output by GPT-3
  - Target: The reward
- What type of learning is this? (SL/UL/RL?)



Final step: Update the base model (GPT3) with the reward output from RM



Source: https://openai.com/research/instruction-following

End result

Prompt *Explain the moon landing to a 6 year old in a few sentences.*

Completion GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

InstructGPT

```
People went to the moon, and they took pictures of what they saw, and sent them back to the earth so we could all see them.
```

Key points

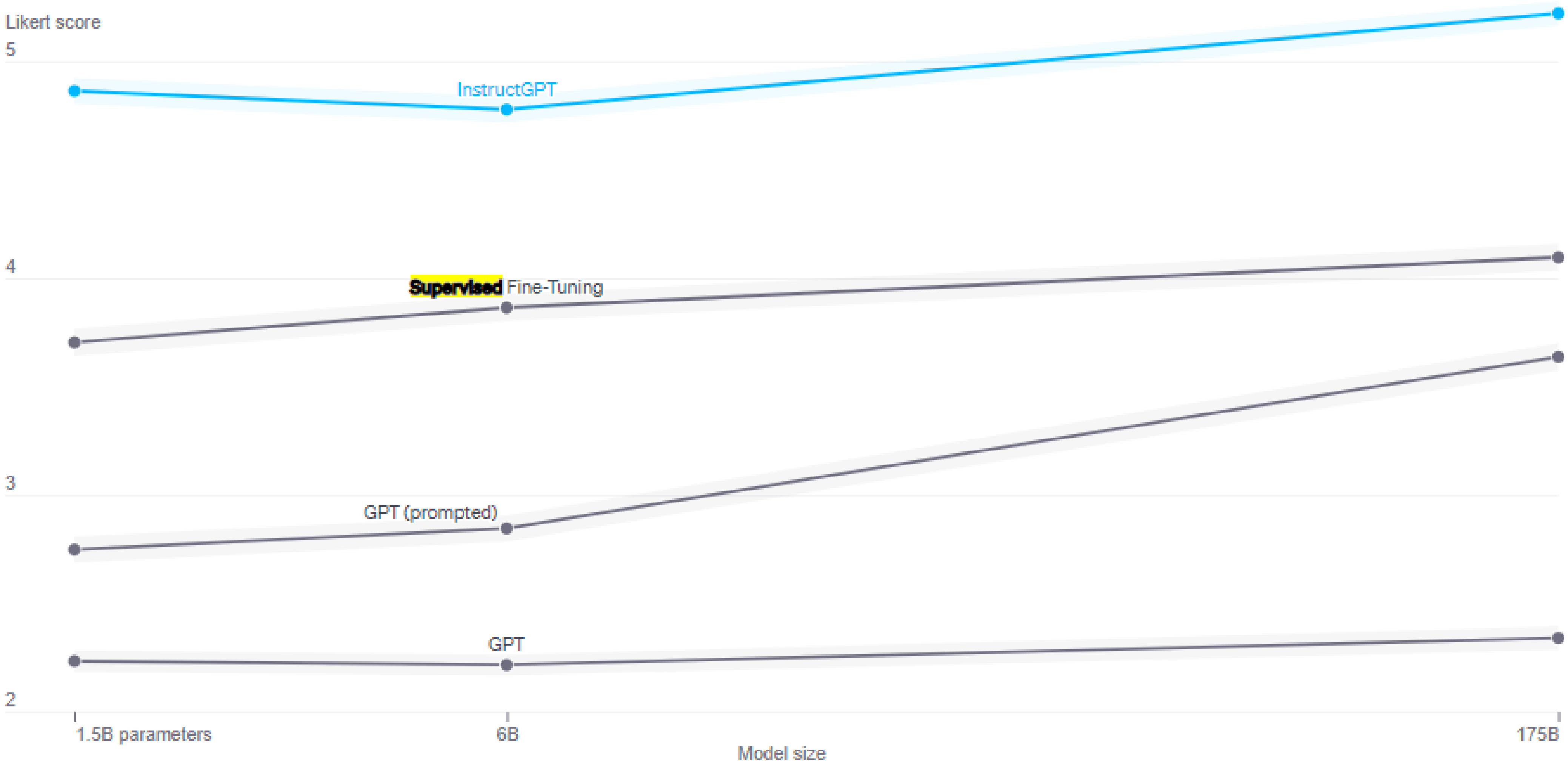
- Language models are not by default good at chatting
- They can be *fine-tuned* for this and other tasks
  - Fine-tuning allows them to *reuse their good language understanding* for new tasks
- Why do we fine-tune with Reinforcement Learning and not Supervised Learning?

Explain the moon landing to a 6 year old in a few sentences.

```
People went to the moon, and they took pictures of what they saw, and sent them back to the earth so
we could all see them.
```

Why RL? Not even experts are sure

- https://gist.github.com/yoavg/6bff0fecd65950898eba1bb321cfbd81
- **Diversity**: RL lets us tell the model «this is good» without saying «you should answer exactly this».
- **Negative feedback**: With RL, it is easier to say «this was wrong». With SL, we say «this is the correct answer», harder to say what should NOT be in an answer.
- **SL encourages the model to «lie»:** We say «You should answer X» when the model perhaps doesn't know/believe «X».
- Main reason perhaps: It just **works better**.



Quality ratings of model outputs on a 1–7 scale (y-axis), for various model sizes (x-axis), on prompts submitted to InstructGPT models on our API. InstructGPT outputs are given much higher scores by our labelers than outputs from GPT-3 with a few-shot prompt and without, as well as models fine-tuned with supervised learning. We find similar results for prompts submitted to GPT-3 models on the API.

Questions?