Introduction to RLHF

REINFORCEMENT LEARNING FROM HUMAN FEEDBACK (RLHF)

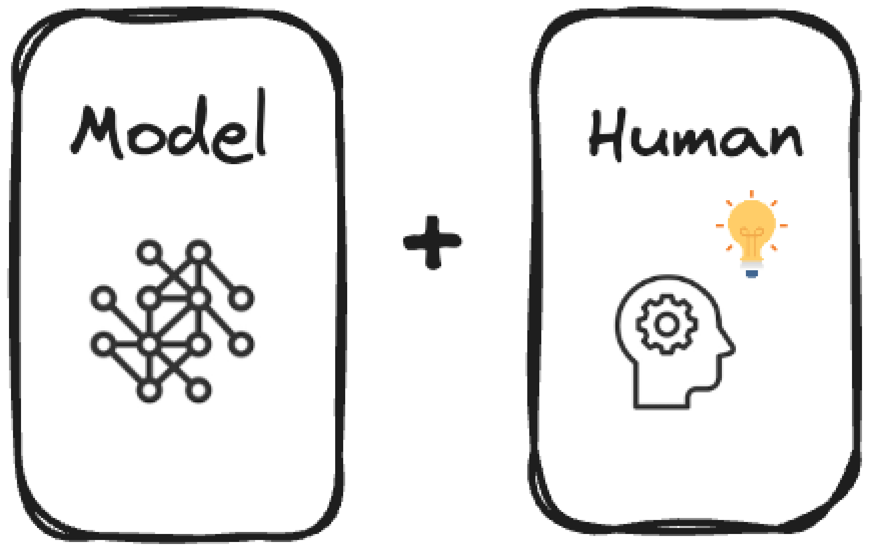
Mina Parham
AI Engineer

Welcome to the course!

- Instructor: **Mina Parham**

- AI Engineer
- Large Language Models (LLMs)
- Reinforcement Learning from Human Feedback (RLHF)

- Topic: **Reinforcement Learning from Human Feedback (RLHF)**

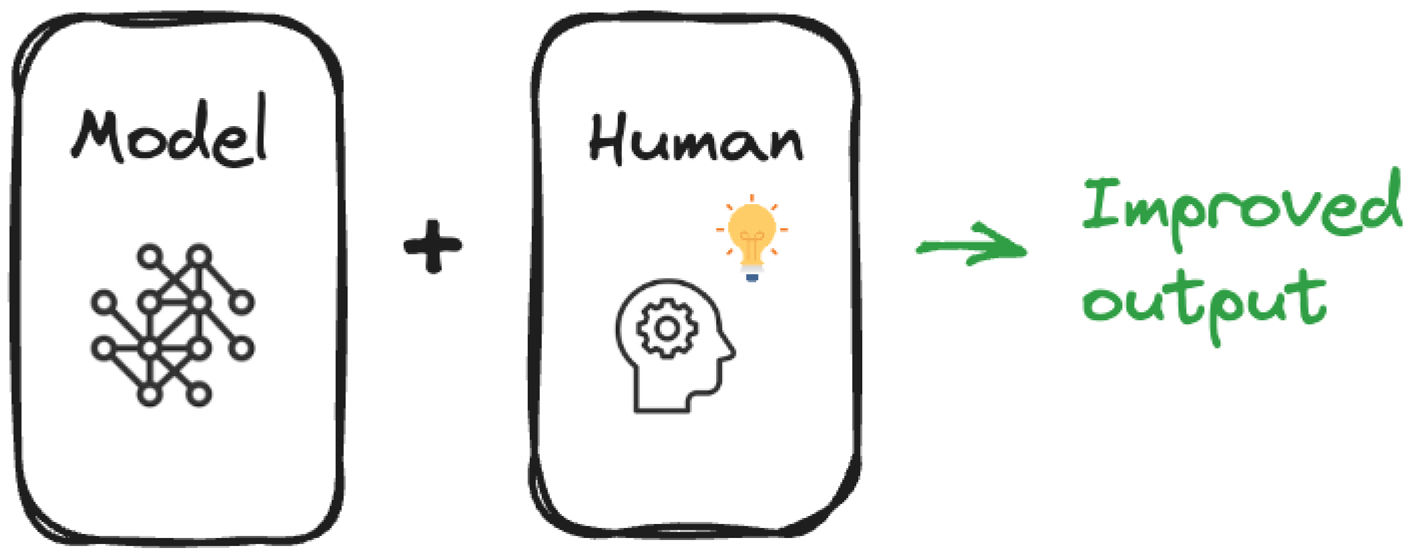


Welcome to the course!

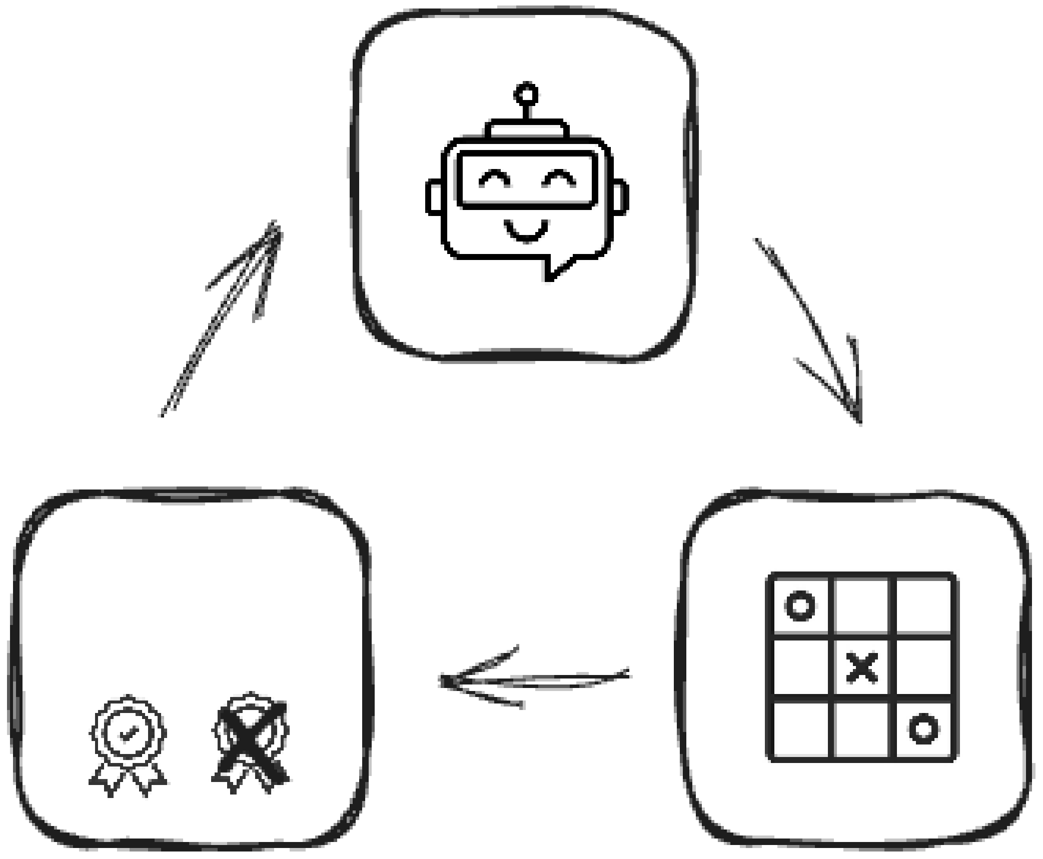
- Instructor: **Mina Parham**

- AI Engineer
- Large Language Models (LLMs)
- Reinforcement Learning from Human Feedback (RLHF)

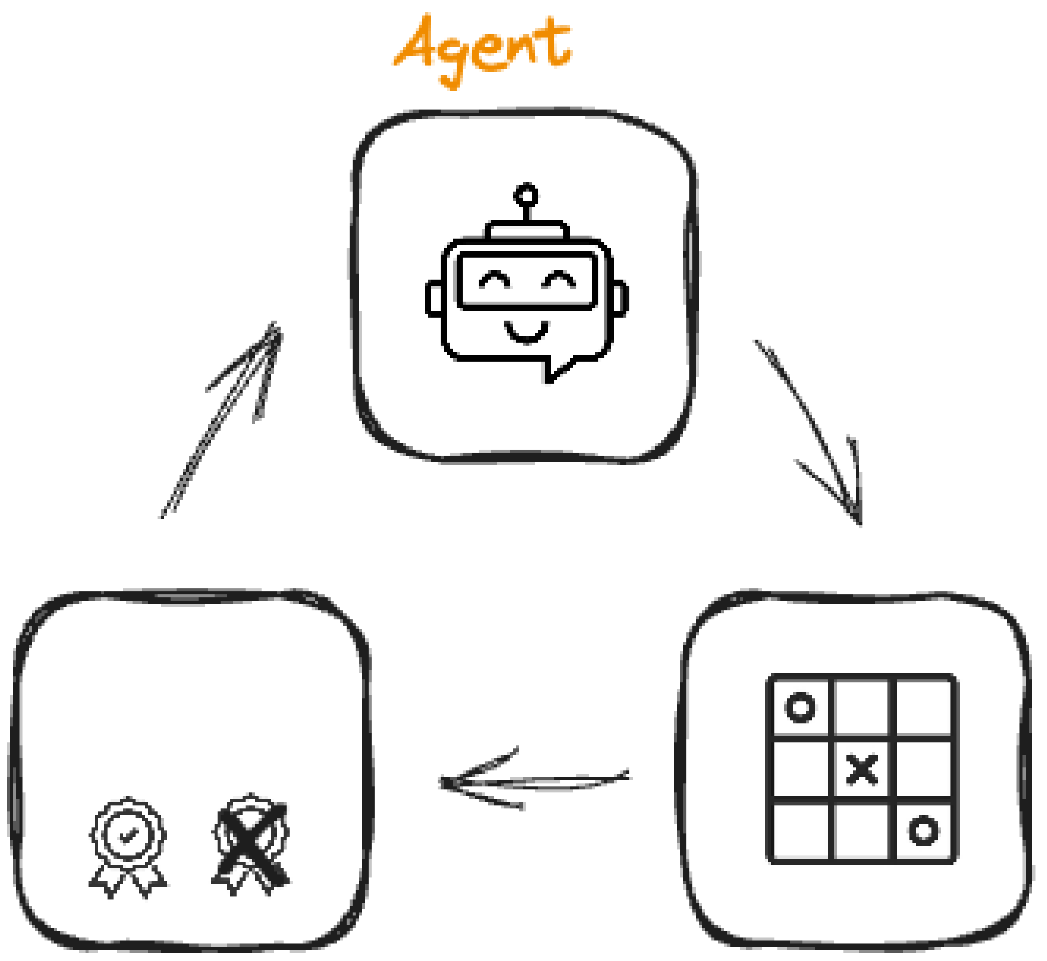
- Topic: **Reinforcement Learning from Human Feedback (RLHF)**



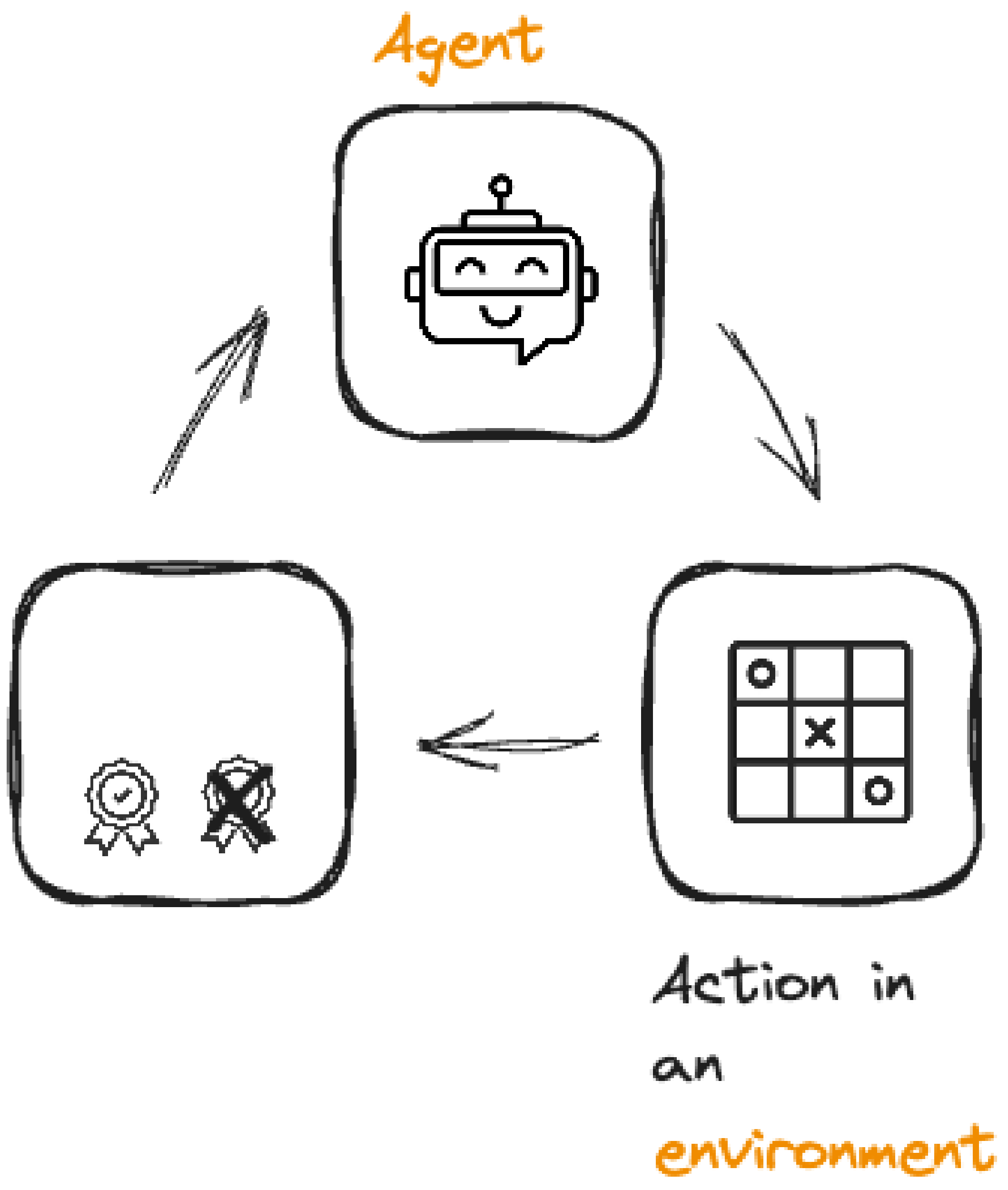
Reinforcement learning review



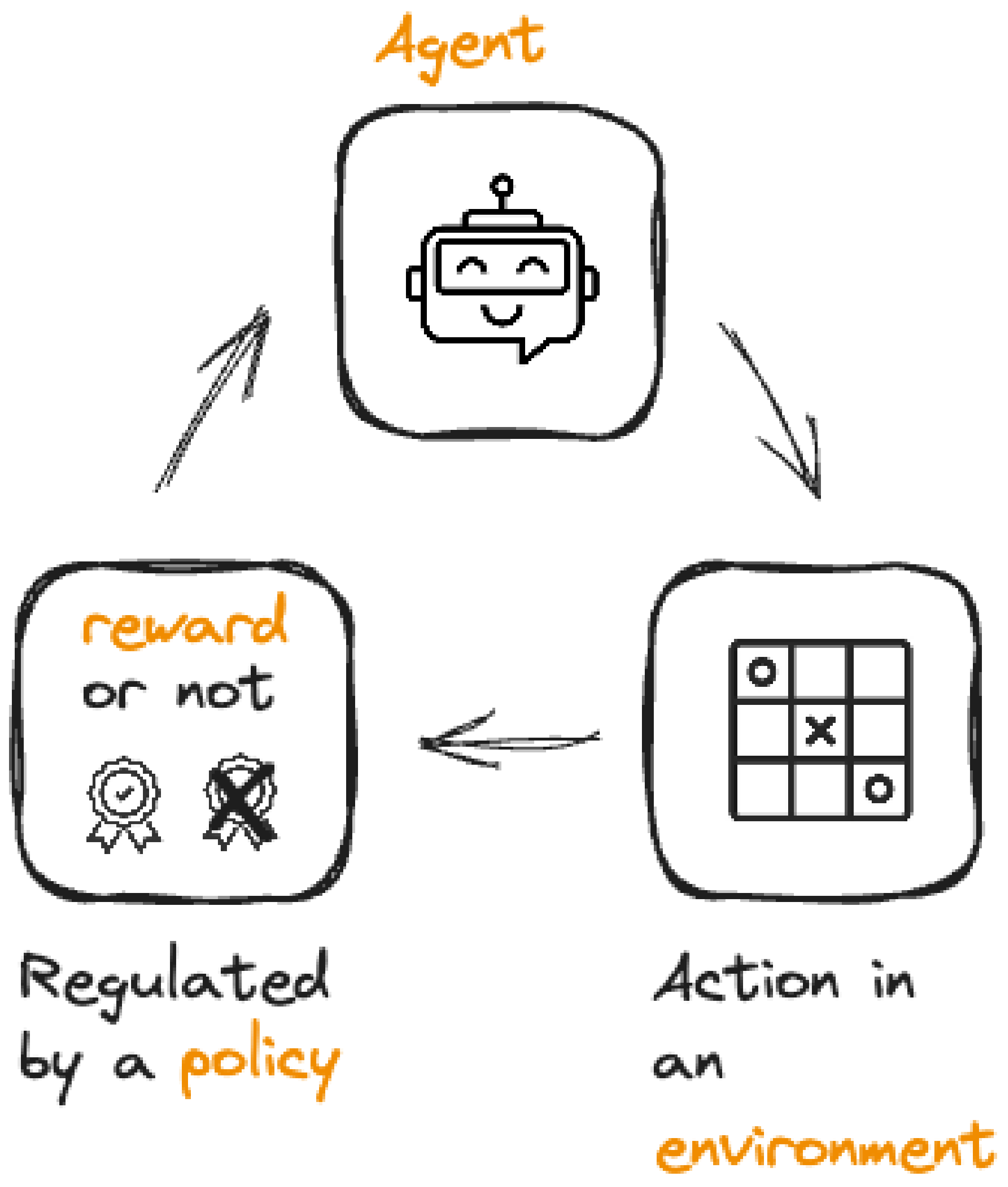
Reinforcement learning review



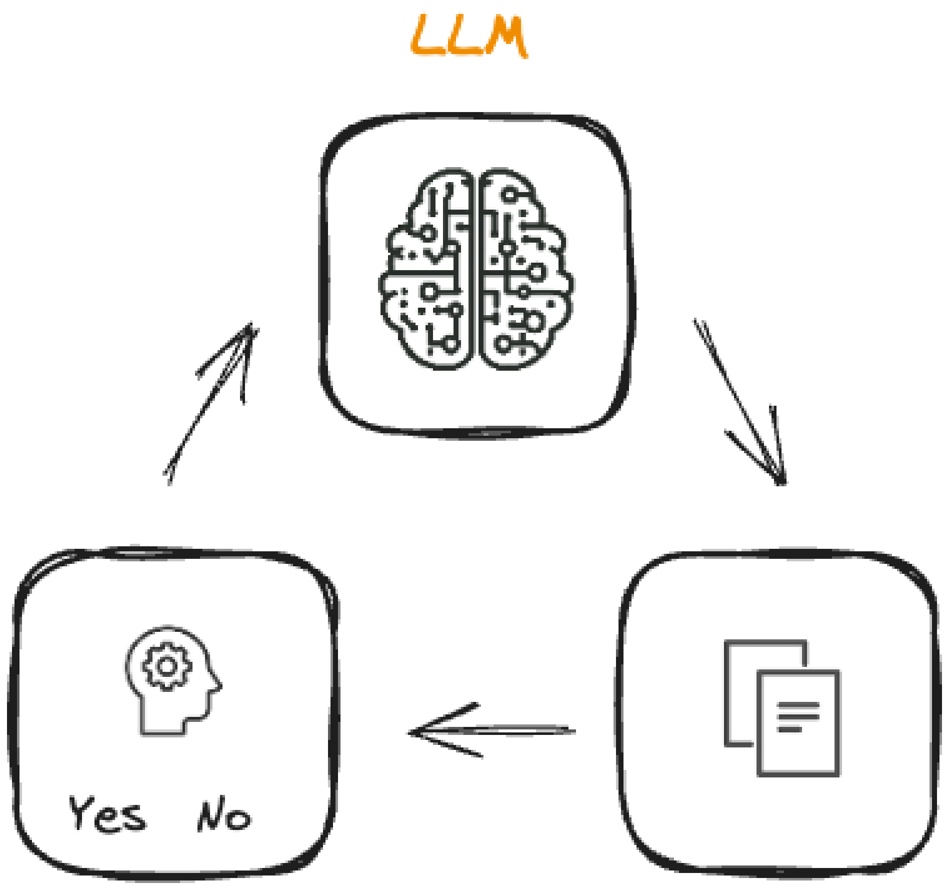
Reinforcement learning review



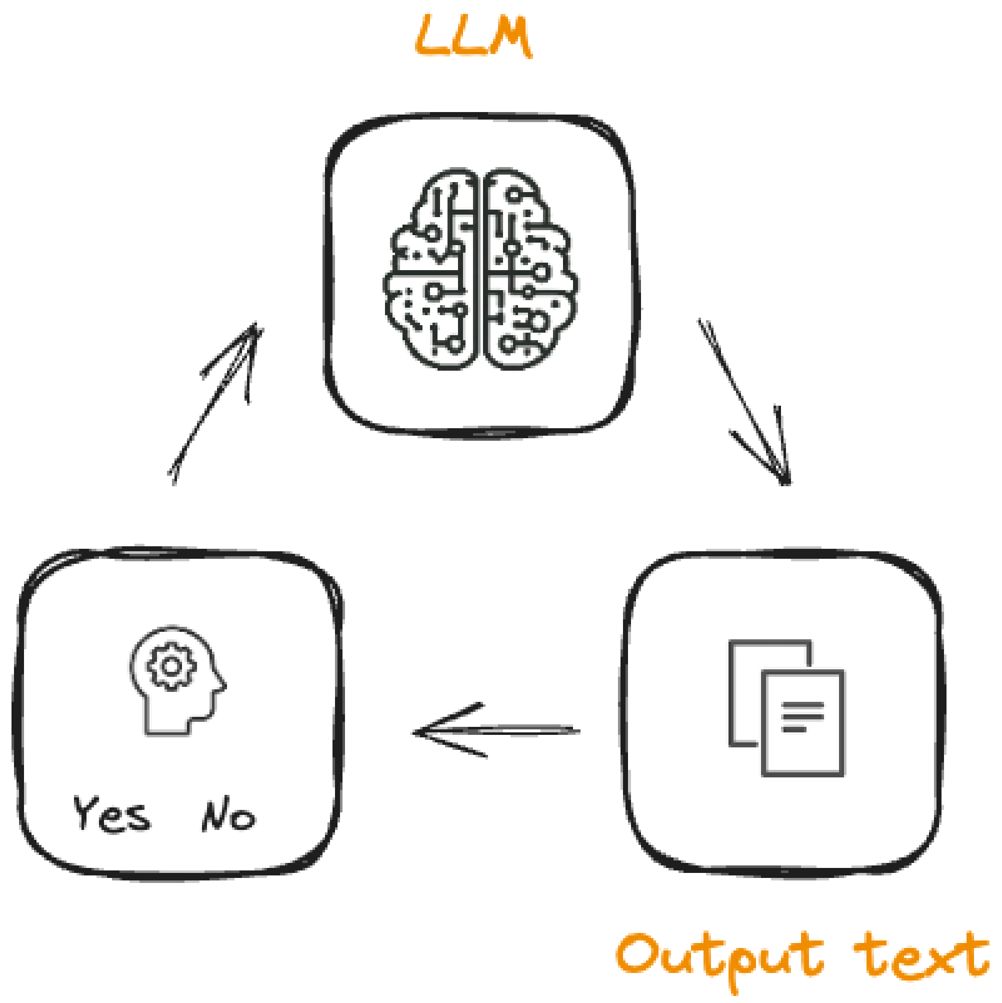
Reinforcement learning review



From RL to RLHF

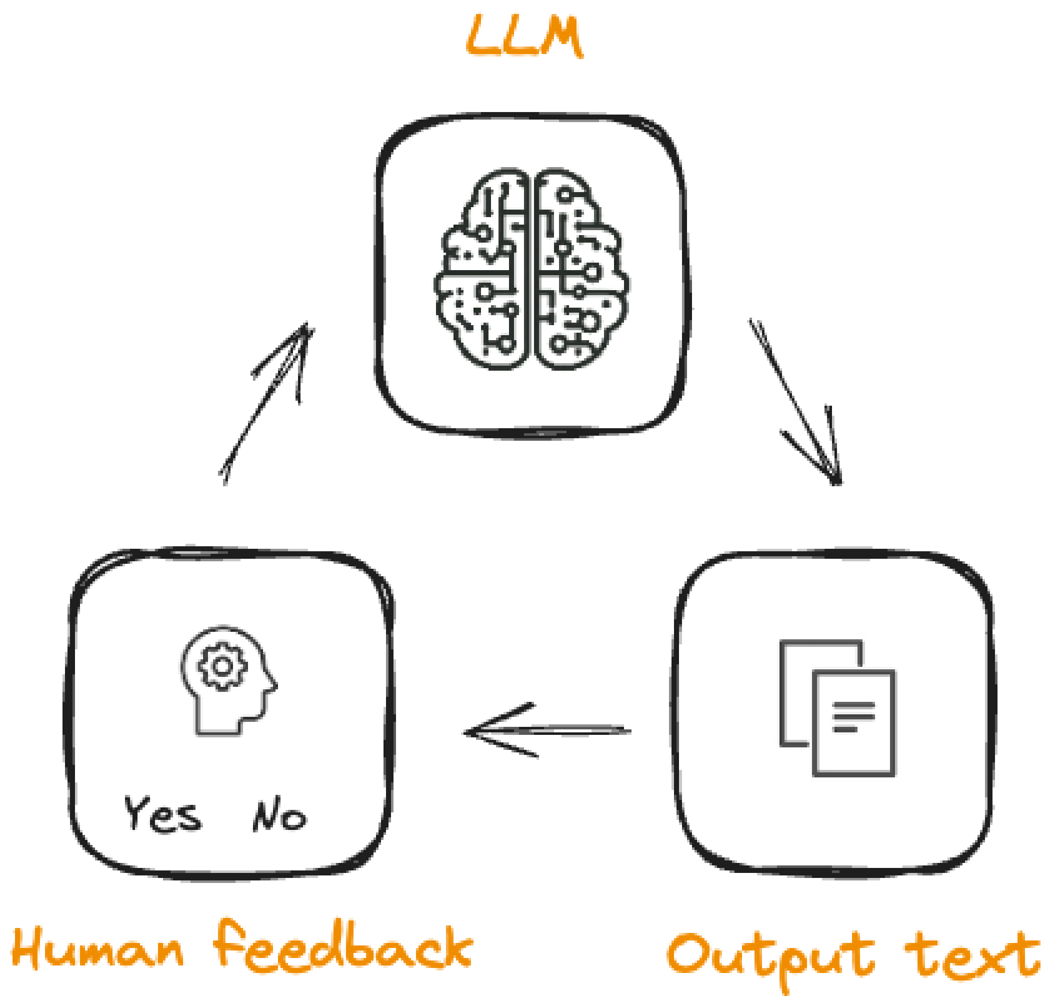


From RL to RLHF

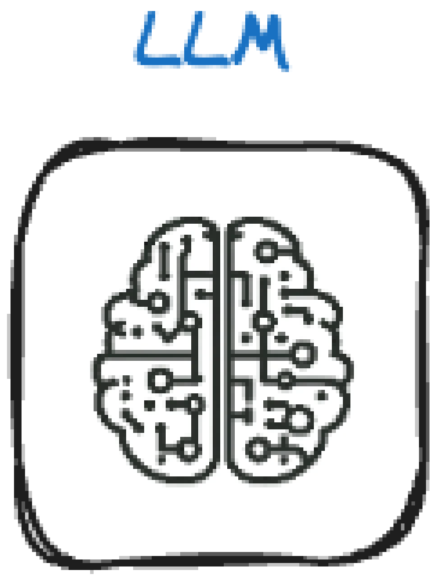


From RL to RLHF

- Training the **reward model**
- Alignment with human preferences

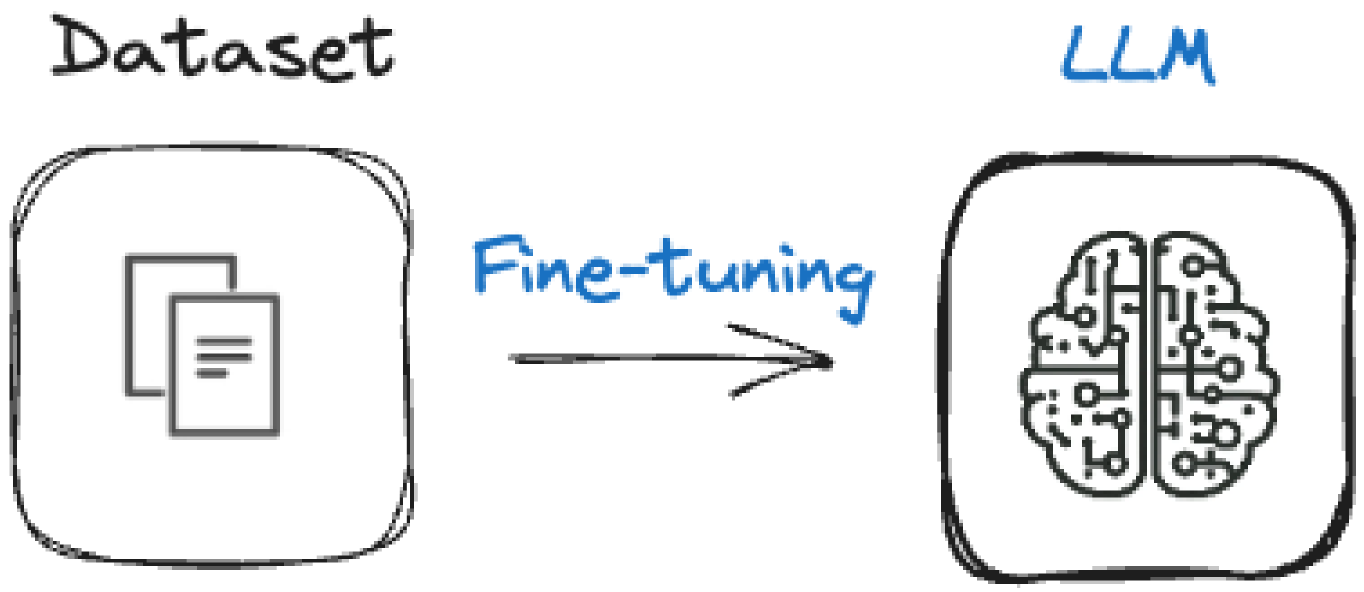


LLM fine-tuning in RLHF

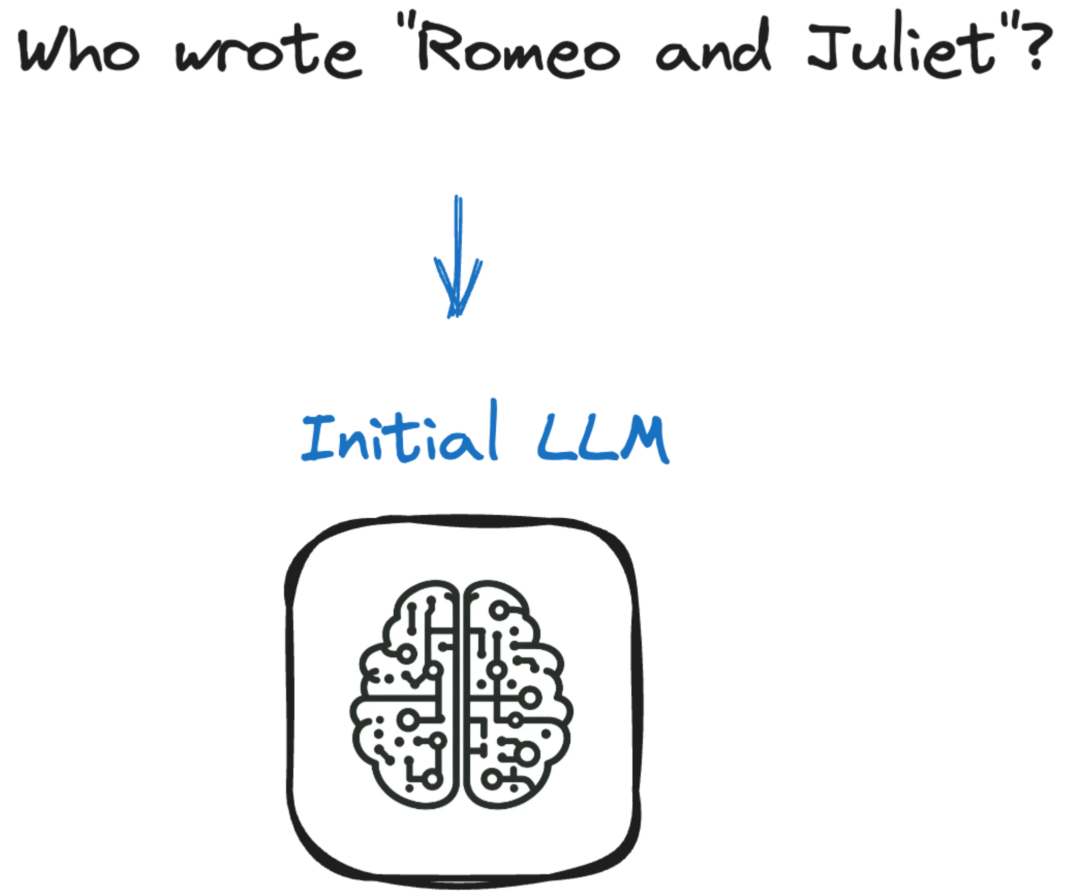


LLM fine-tuning in RLHF

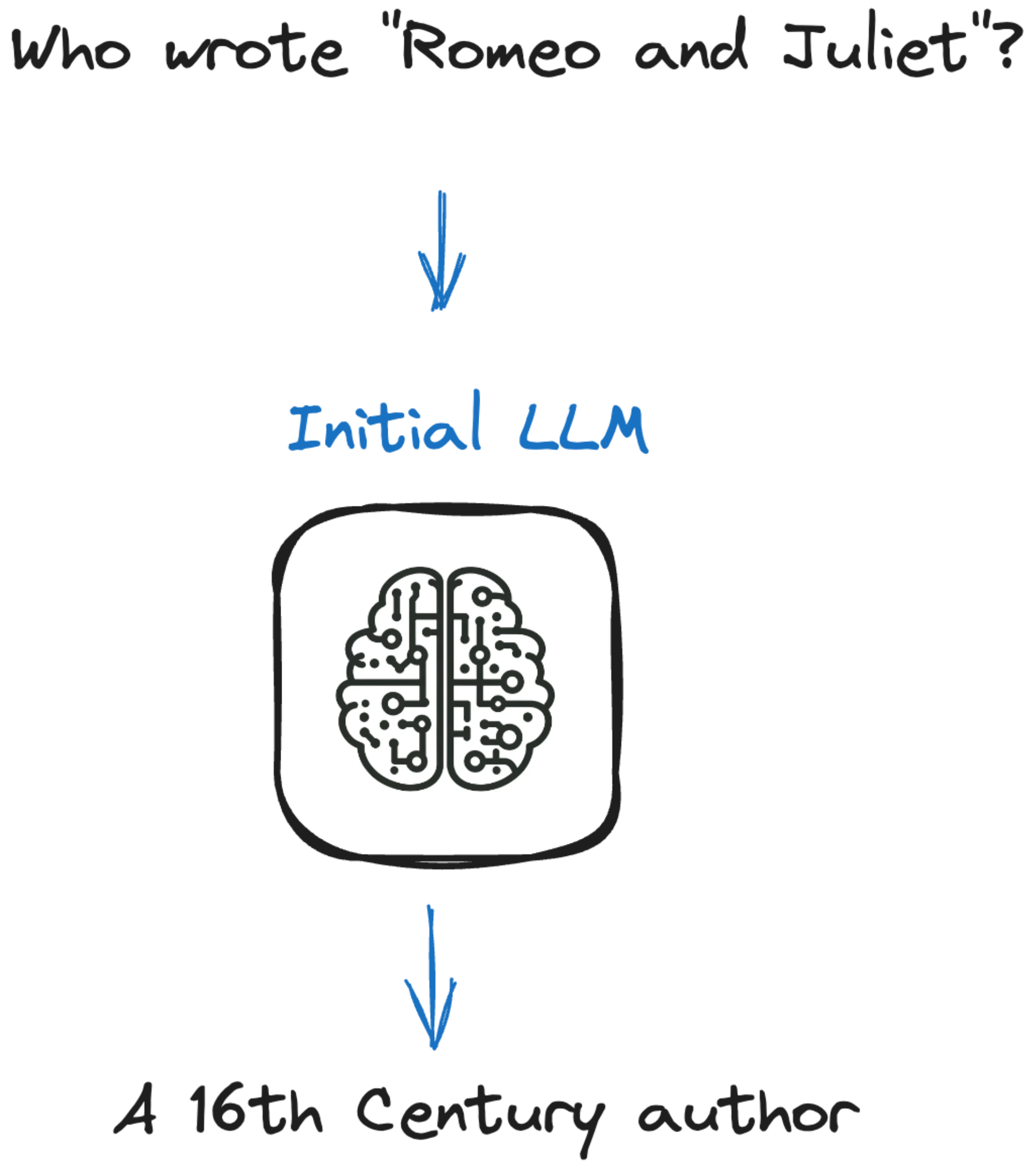
- Training the **initial LLM**



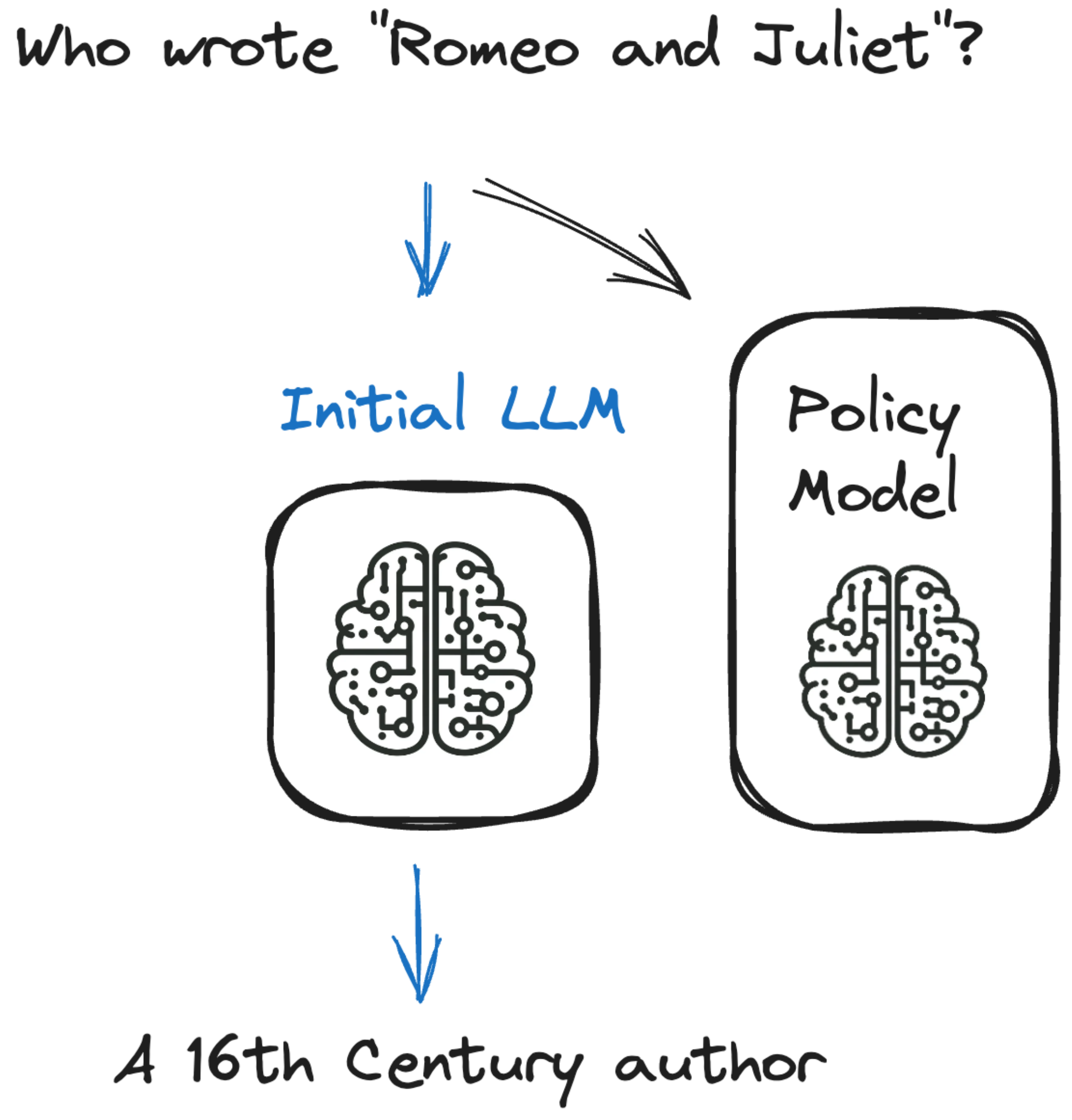
The full RLHF process



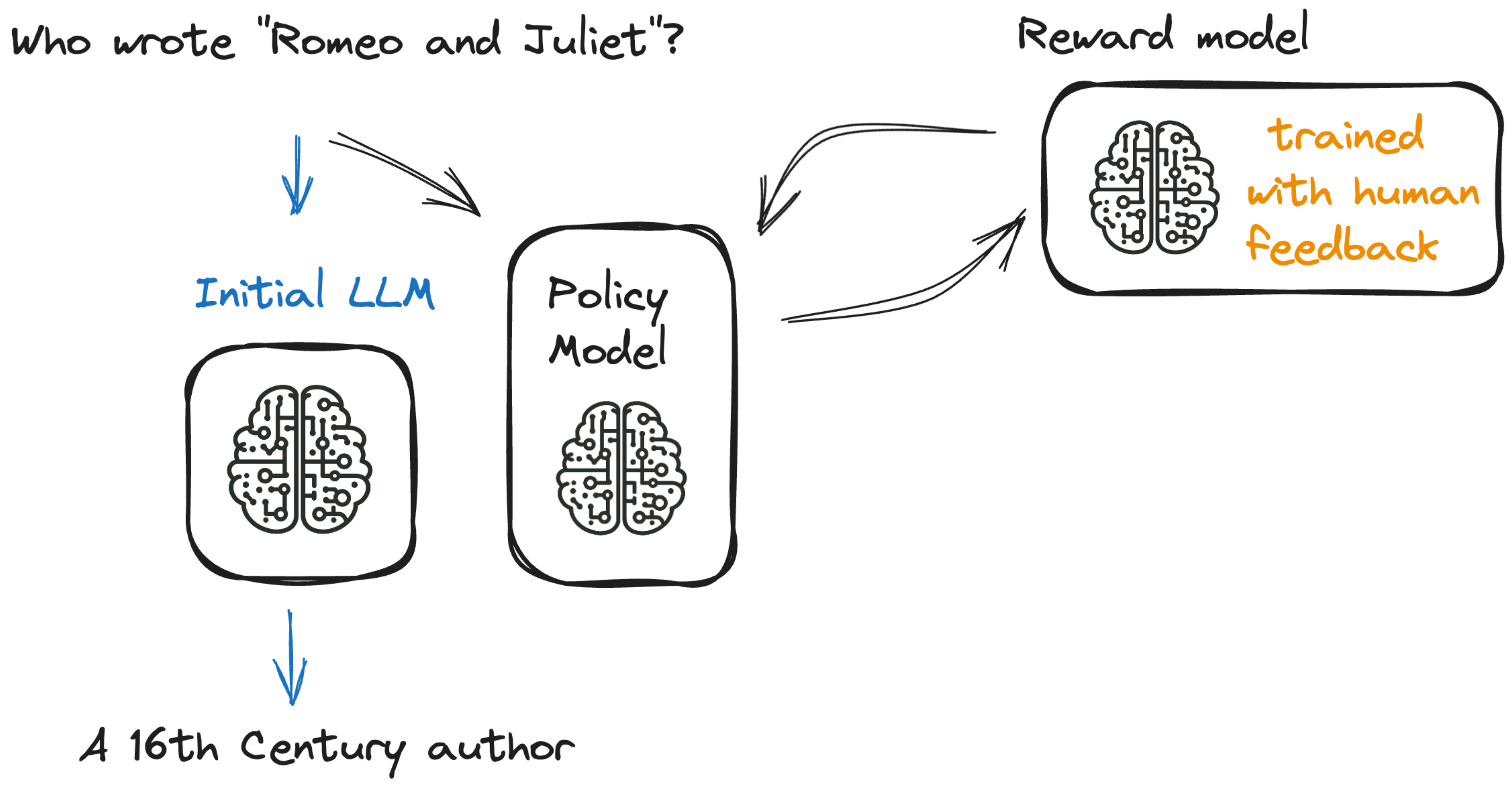
The full RLHF process



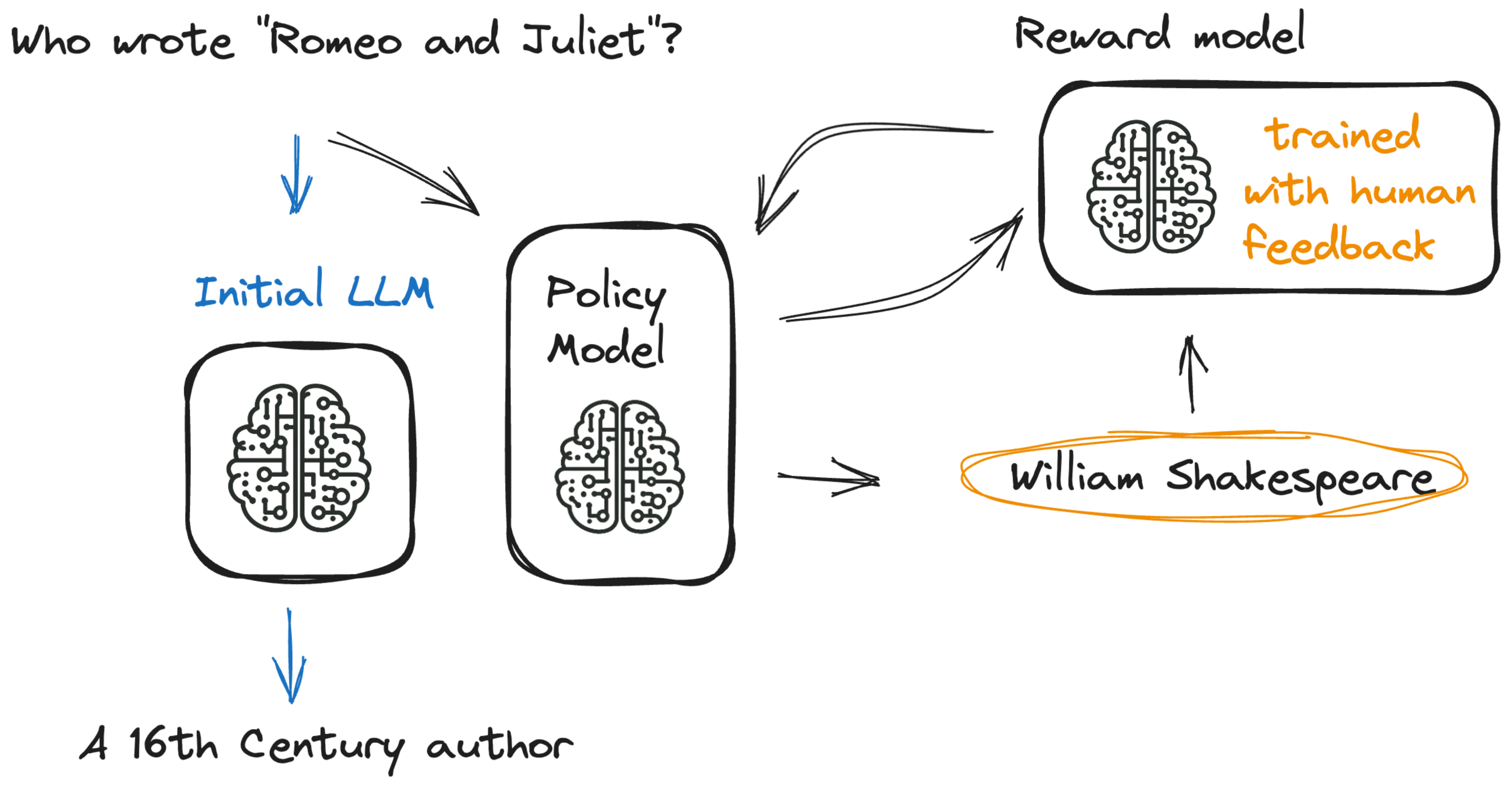
The full RLHF process



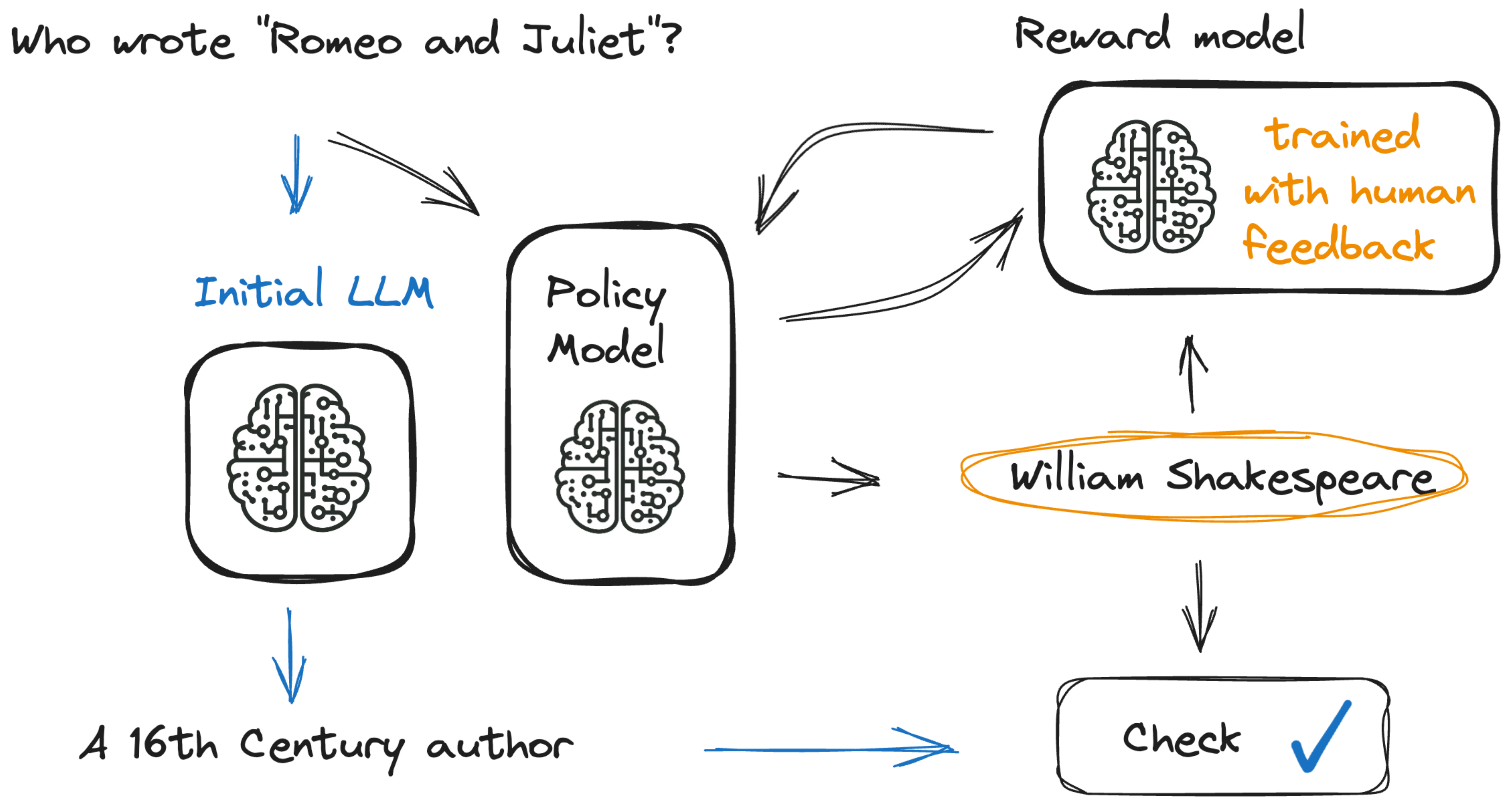
The full RLHF process



The full RLHF process



The full RLHF process



Interacting with RLHF-tuned LLMs

- Pre-trained RLHF models on Hugging Face 🤗

```
from transformers import pipeline
text_generator = pipeline('text-generation', model='lvwerra/gpt2-imdb-pos-v2')
# Provide a review prompt
review_prompt = "This is definitely a"

# Generate the continuation
output = text_generator(review_prompt, max_length=50)

#Print the generated text
print(output[0]['generated_text'])
```

```
This is definitely a crucial improvement.
```

Interacting with RLHF-tuned LLMs

```python
from transformers import pipeline, AutoModelForSequenceClassification, AutoTokenizer

# Instantiate the pre-trained model and tokenizer
model = AutoModelForSequenceClassification.from_pretrained("lvwerra/distilbert-imdb")
tokenizer = AutoTokenizer.from_pretrained("lvwerra/distilbert-imdb")

# Use pipeline to create the sentiment analyzer
sentiment_analyzer = pipeline('sentiment-analysis', model=model, tokenizer=tokenizer)

# Pass the text to the sentiment analyzer and print the result
sentiment = sentiment_analyzer("This is definitely a crucial improvement.")
print(f"Sentiment Analysis Result: {sentiment}")
```

```
positive
```

Let's practice!

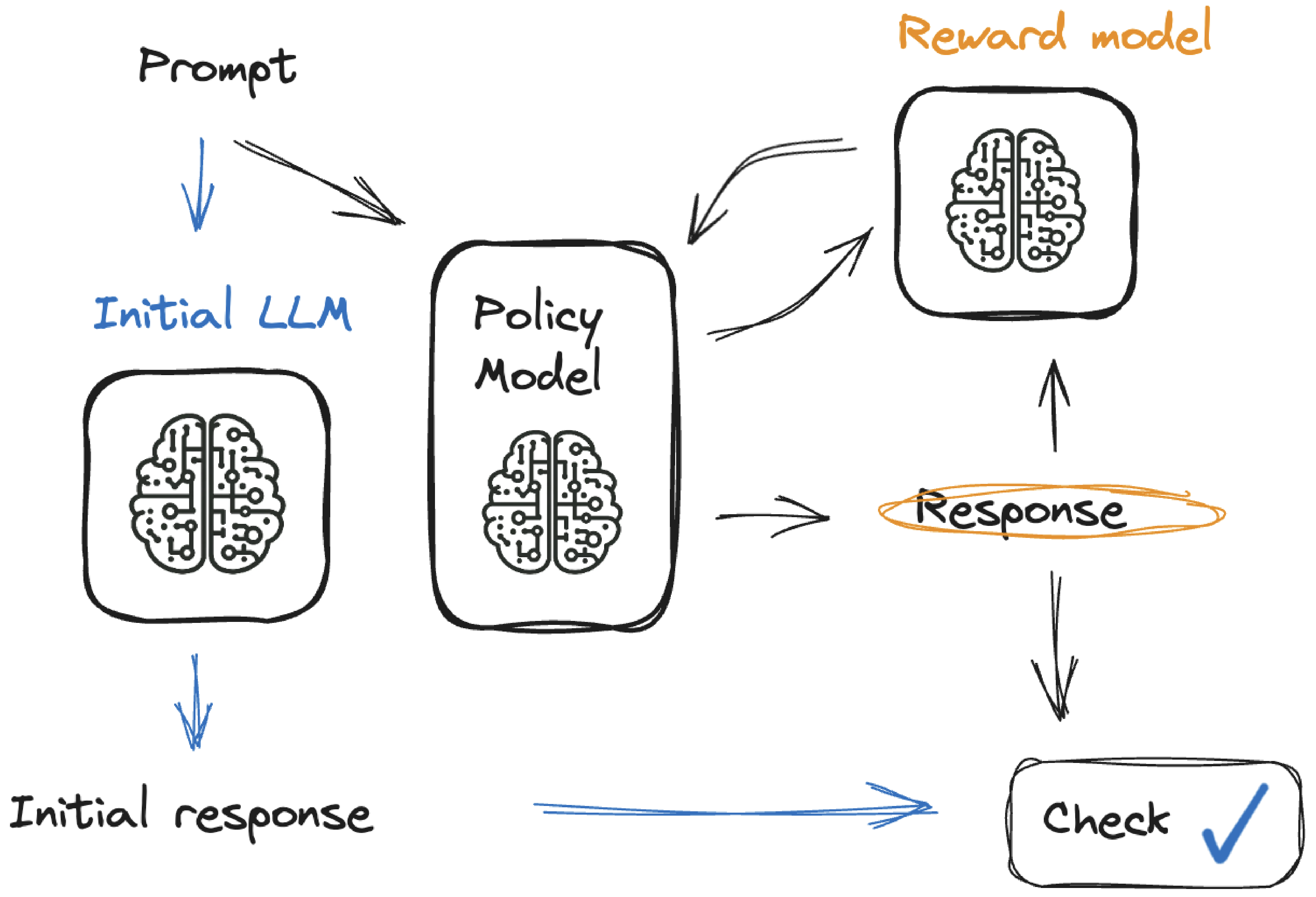
REINFORCEMENT LEARNING FROM HUMAN FEEDBACK (RLHF)

Exploring pre-trained LLMs

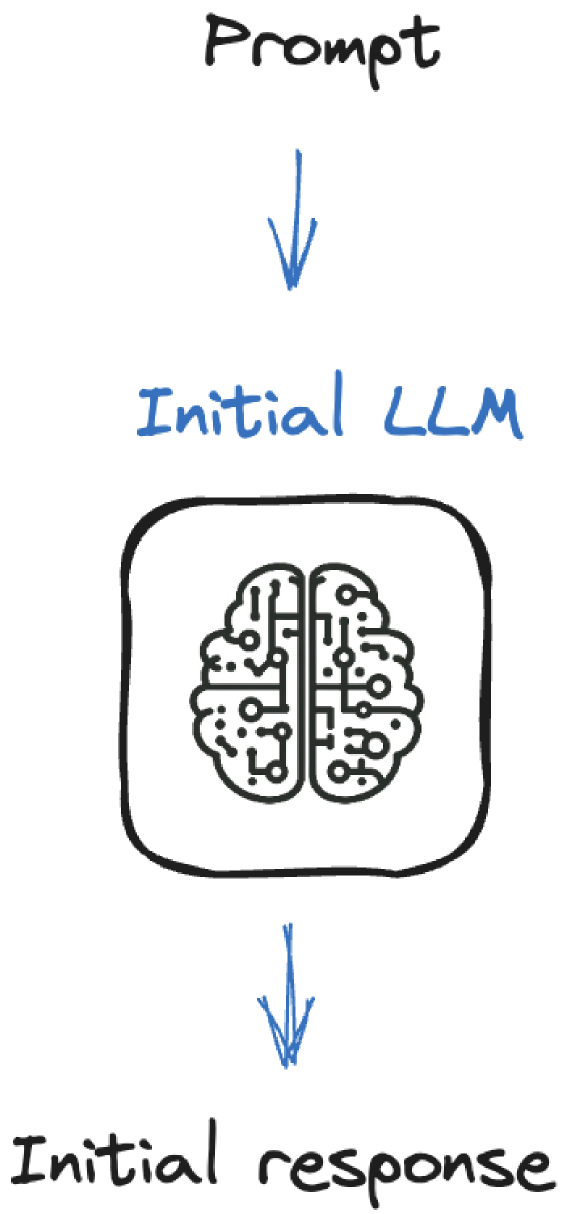
REINFORCEMENT LEARNING FROM HUMAN FEEDBACK (RLHF)

Mina Parham
AI Engineer

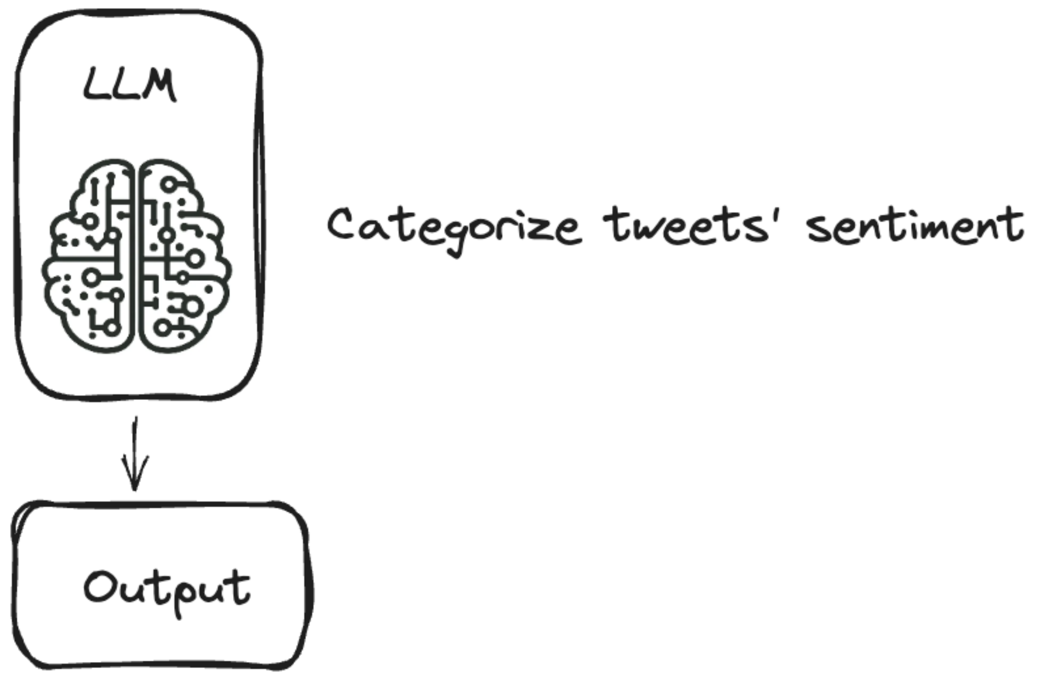
The importance of fine-tuning



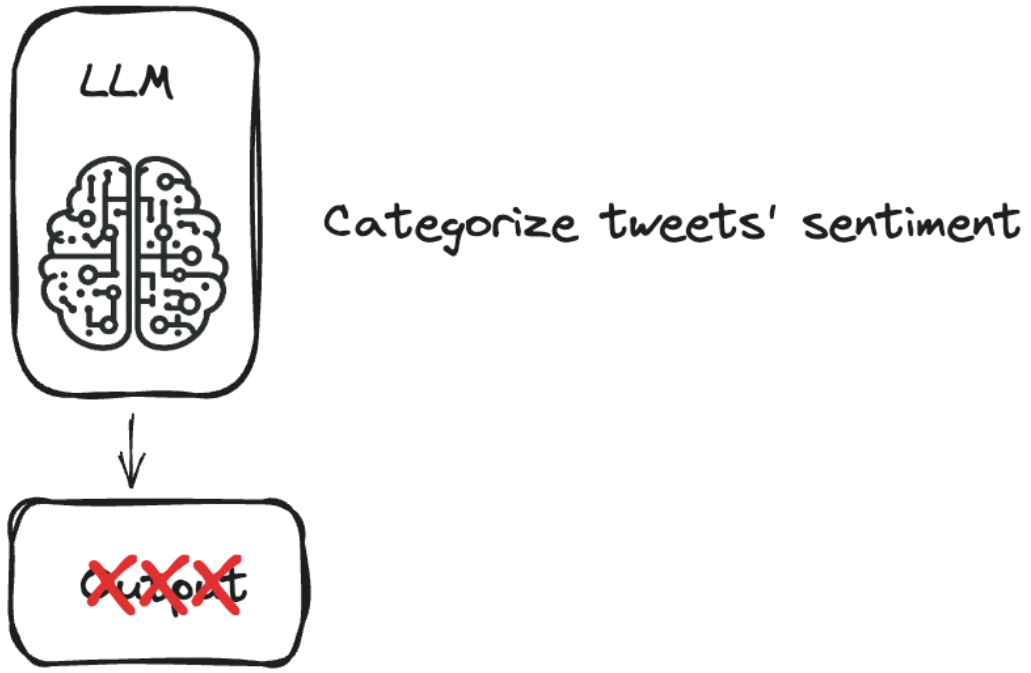
The importance of fine-tuning



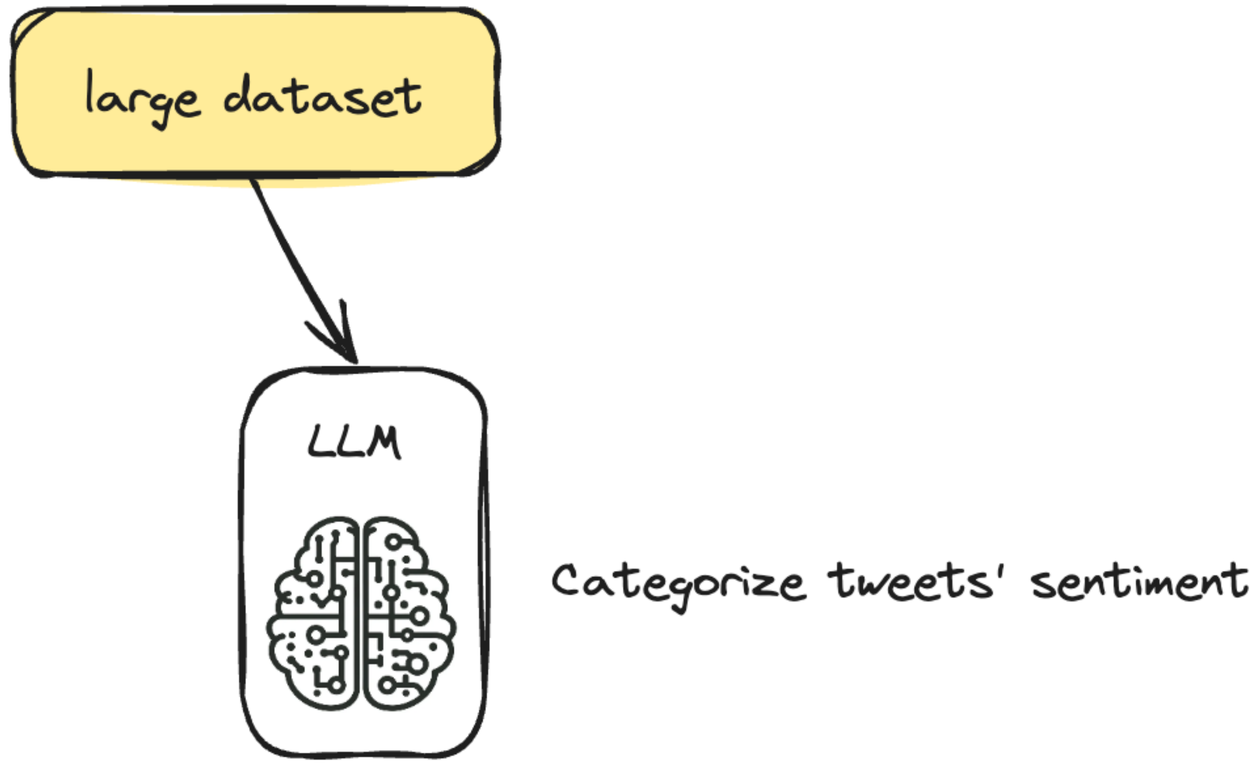
A step-by-step guide to fine-tuning an LLM



A step-by-step guide to fine-tuning an LLM



A step-by-step guide to fine-tuning an LLM



Step 1: load the data to use

```python
from datasets import load_dataset
import pandas as pd

# `load_dataset` simplifies loading and preprocessing datasets from various sources
# It provides easy access to a wide range of datasets with minimal setup
dataset = load_dataset("mteb/tweet_sentiment_extraction")
df = pd.DataFrame(dataset['train'])
```

```
    id            text                                label   label_text
0   cb774db0d1    I'd have responded, if I were going  1       neutral
1   549e992a42    Sooo SAD I will miss you in San Diego!!!  0   negative
2   08ac60f138    my boss is bullying me...            0       negative
```

Step 2: choose a pre-trained model

```python
from transformers import AutoModelForCausalLM


# AutoModelForCausalLM simplifies loading and switching models
model = AutoModelForCausalLM.from_pretrained("openai-gpt")
```

- **Causal models:** previous tokens "cause" subsequent ones

Step 3: tokenizer

```python
from transformers import AutoTokenizer

# `AutoTokenizer` loads the correct tokenizer for the specified model
tokenizer = AutoTokenizer.from_pretrained("openai-gpt")
tokenizer.add_special_tokens({'pad_token': '[PAD]'})
model.resize_token_embeddings(len(tokenizer))
```

- Padding: to have equal-sized batches of text

Step 3: tokenizer

```
def tokenize_function(examples):
    tokenized = tokenizer(examples["content"], padding="max_length", truncation=True
    return tokenized

tokenized_datasets = dataset.map(tokenize_function, batched=True)
```

- Batched parameter: for faster processing

Step 4: fine-tune using the Trainer method

```
training_args = TrainingArguments(
   output_dir="test_trainer",
   per_device_train_batch_size=1,
   per_device_eval_batch_size=1,
   gradient_accumulation_steps=4)
```

```
trainer = Trainer(
   model=model,
   args=training_args,
   train_dataset=tokenized_dataset["train"],
   eval_dataset=tokenized_dataset["test"])
trainer.train()
```

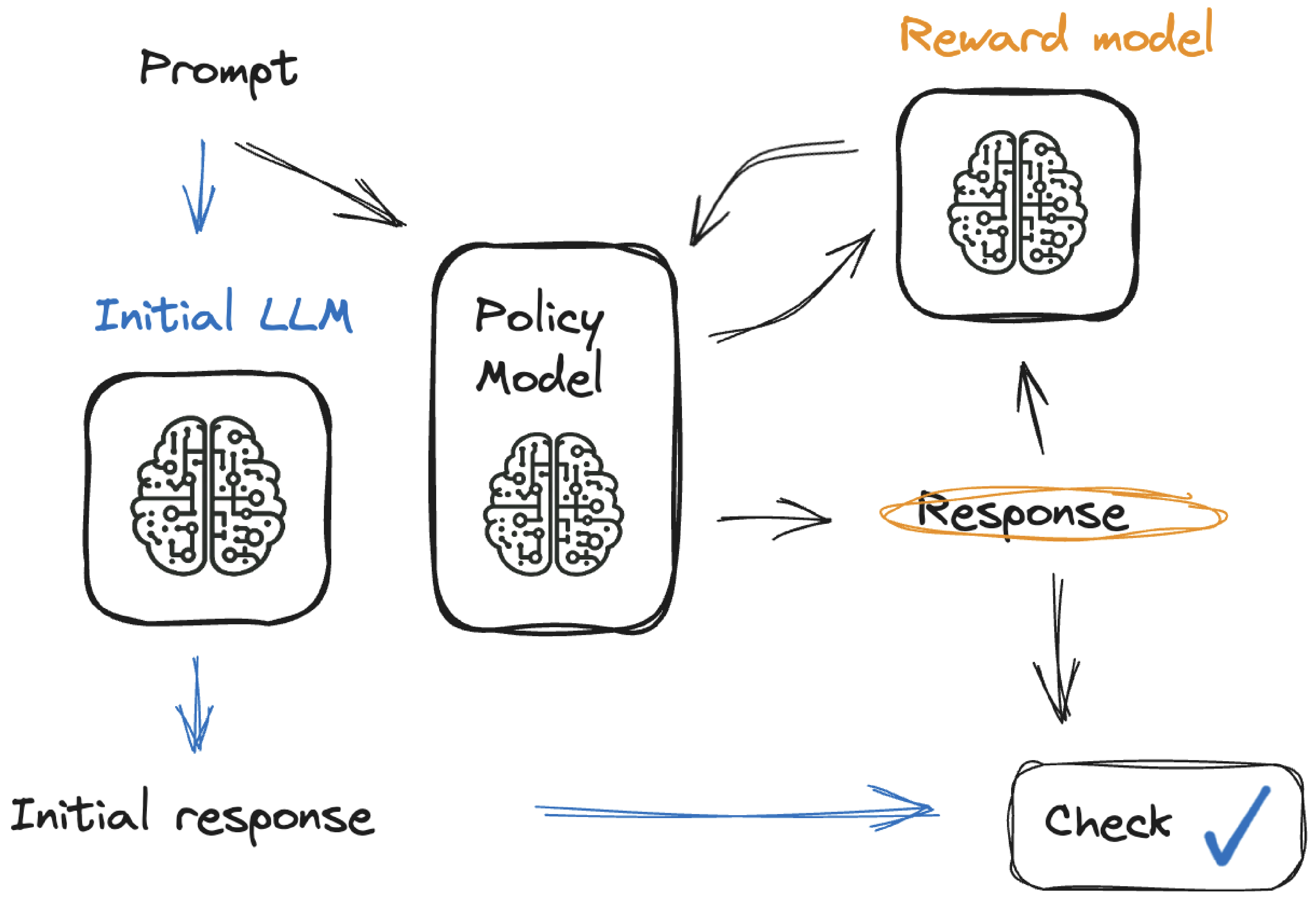
Let's practice!

Preparing data for RLHF

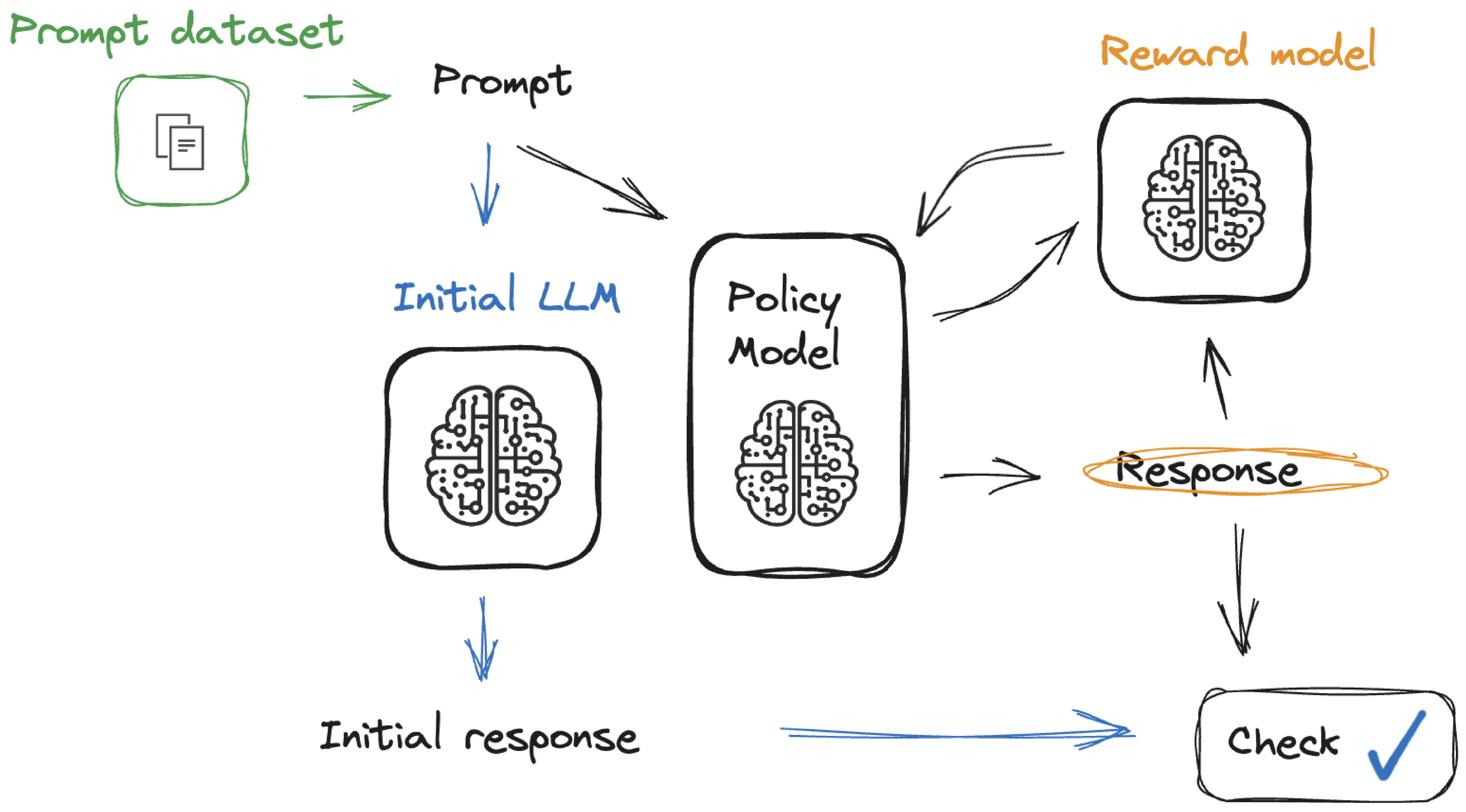
REINFORCEMENT LEARNING FROM HUMAN FEEDBACK (RLHF)

Mina Parham
AI Engineer

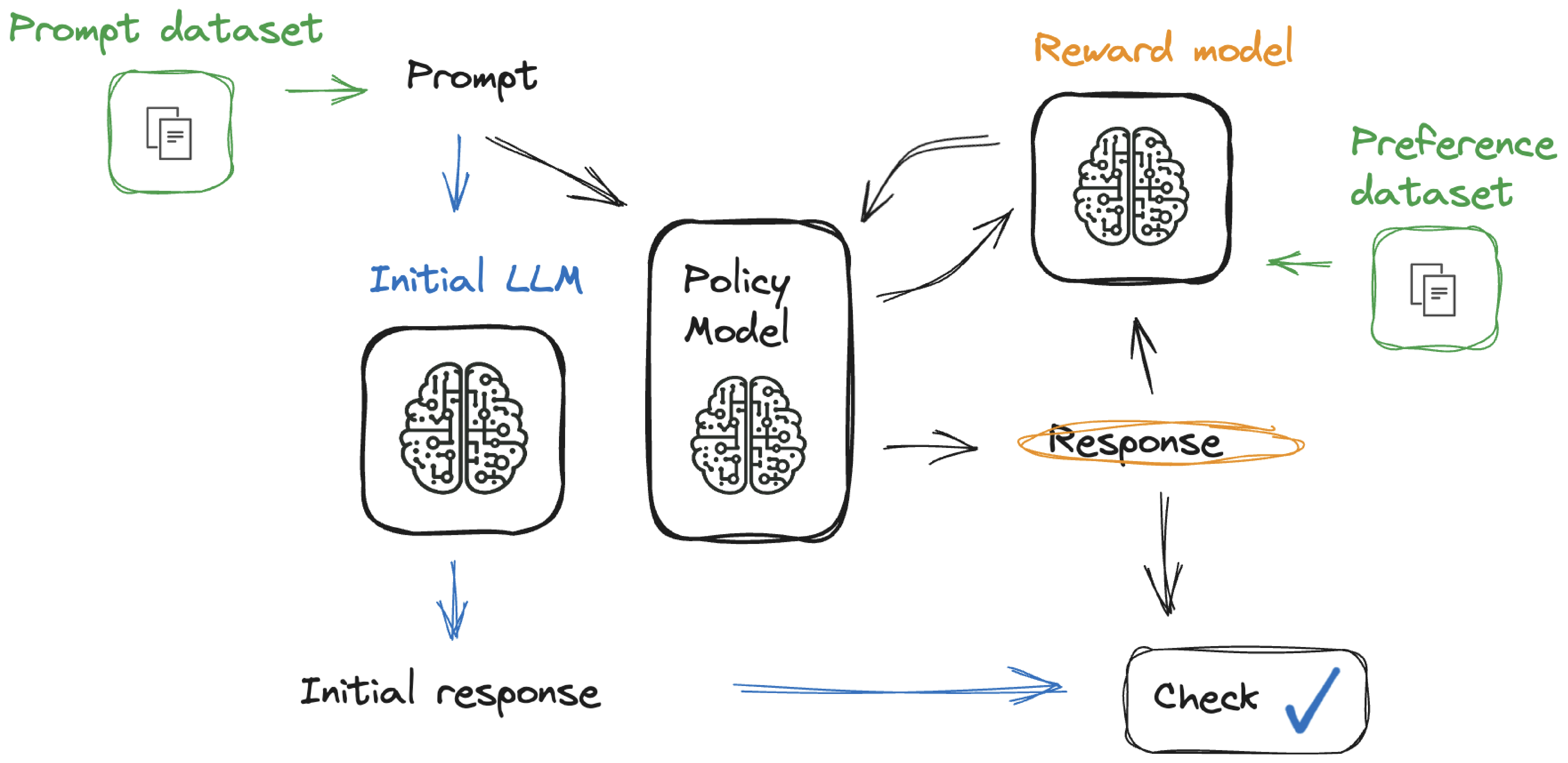
Preference vs. Prompt datasets



Preference vs. Prompt datasets



Preference vs. Prompt datasets



Prompt dataset

- Questions for the model
- Can be found on Hugging Face datasets

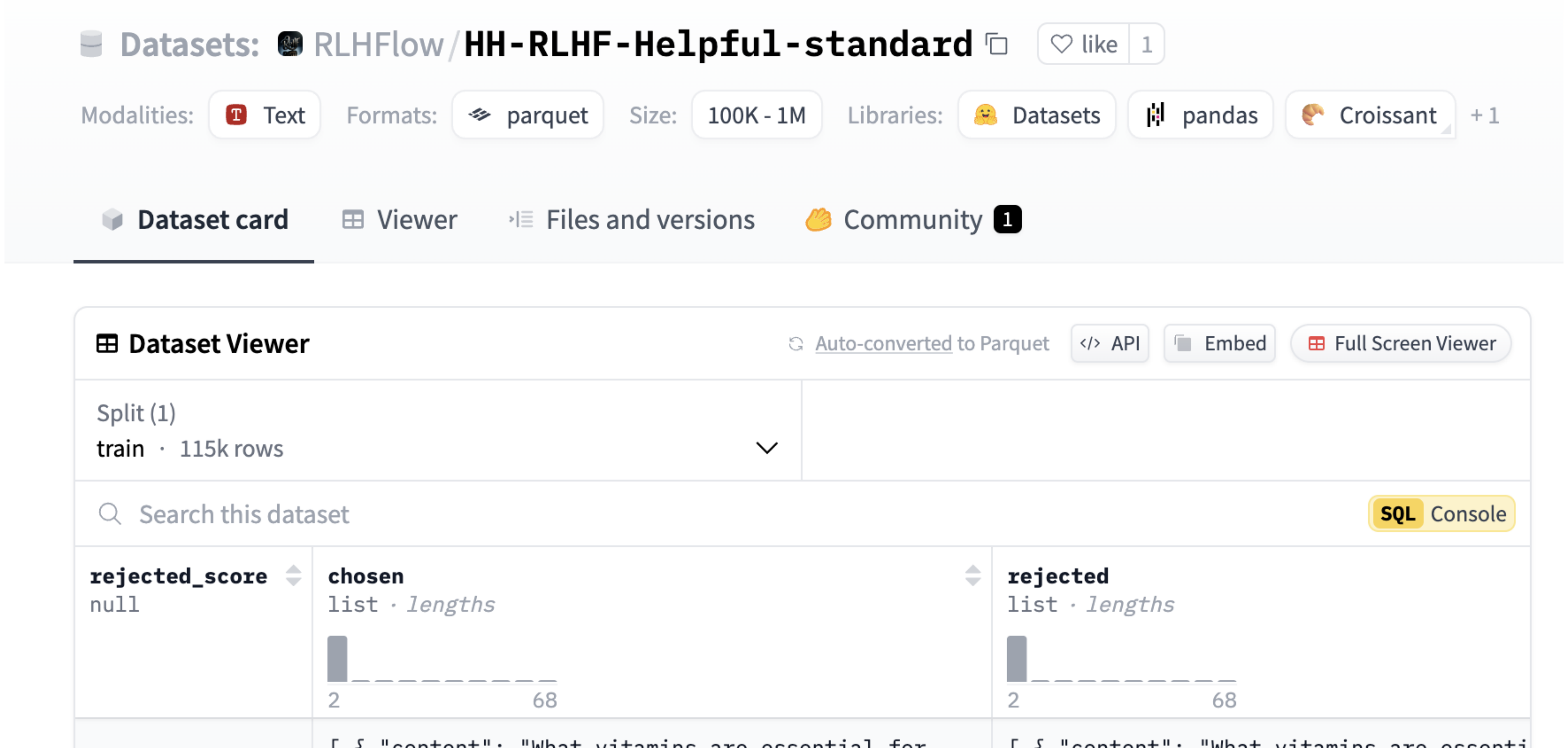
```
prompt_data = load_dataset("center-for-humans-and-machines/rlhf-hackathon-prompts",
                           split="train")
prompt_data['prompt'][0]
```

```
'How important is climate change?'
```

- Might need to extract the prompt
- Look for markers such as: `Input=` , `{{Text}}:` , `###Human:`

Exploring the preference dataset

```python
from datasets import load_dataset
preference_data = load_dataset("trl-internal-testing/hh-rlhf-helpful-base-trl-style",
                               split="train")
```



Datasets: RLHFlow / HH-RLHF-Helpful-standard like 1

Modalities: Text Formats: parquet Size: 100K - 1M Libraries: Datasets pandas Croissant +1

Dataset card Viewer Files and versions Community 1

Dataset Viewer Auto-converted to Parquet API Embed Full Screen Viewer

Split (1)
train · 115k rows

Search this dataset SQL Console

rejected_score	chosen	rejected
null	list · *lengths* 2 68	list · *lengths* 2 68
	[{ "content": "What vitamins are essential for	[{ "content": "What vitamins are essenti

Processing the preference dataset

```python
def extract_prompt(text):
    # Extract the prompt as the first element in the list
    prompt = text[0]["content"]
    return prompt
```

```python
# Apply the extraction function to the dataset
preference_data_with_prompt = preference_data.map(
    lambda example: {**example, 'prompt': extract_prompt(example['chosen'])}
)
```

- The way prompts are extracted is **different for different datasets**

Final preference dataset

```
sample = preference_data_with_prompt.select(range(1))
sample['prompt']
```

```
'What vitamins are essential for the body to function?'
```

```
sample['chosen']
```

```
[ { "content": "What vitamins are essential for the body to function?", "role":
   "user" }, { "content": "There are some very important vitamins that ensure the
   proper functioning of the body, including Vitamins A, C, D, E, and K along ...}]
```

Let's practice!

REINFORCEMENT LEARNING FROM HUMAN FEEDBACK (RLHF)