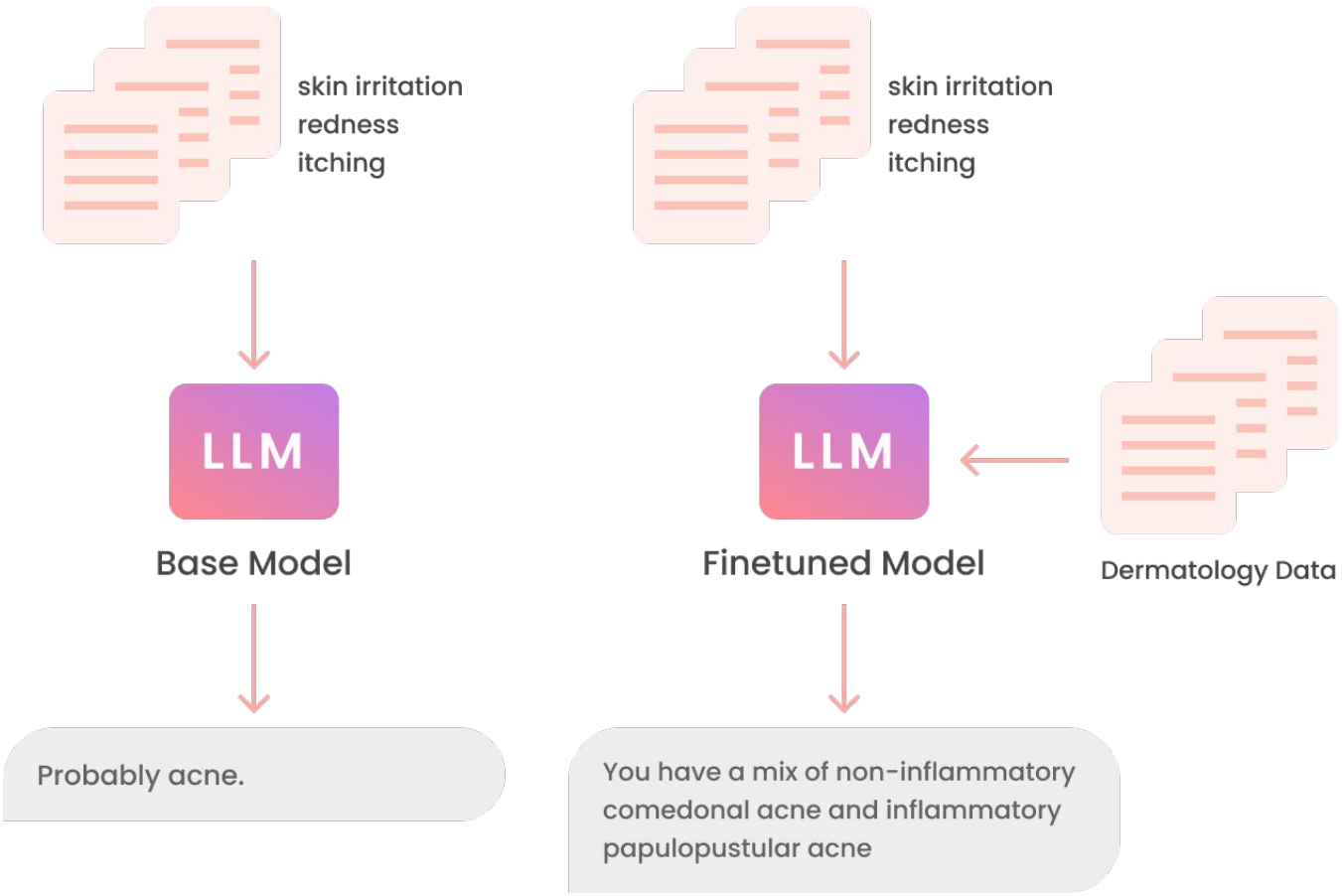
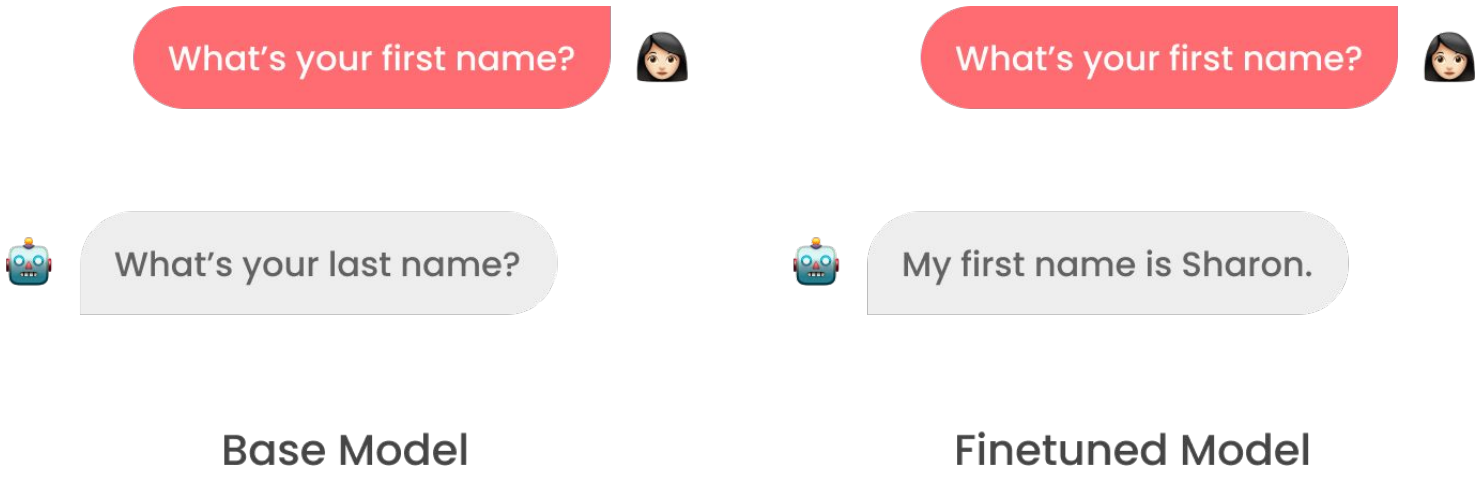
What does finetuning do for the model?

- Lets you put *more* data into the model than what fits into the prompt
- Gets the model to *learn* the data, rather than just get access to it



What does finetuning do for the model?



- Steers the model to more consistent outputs
- Reduces hallucinations
- Customizes the model to a specific use case
- Process is similar to the model's earlier training

Prompt Engineering vs. Finetuning

	Prompting	Finetuning
Pros	• No data to get started • Smaller upfront cost • No technical knowledge needed • Connect data through retrieval (RAG)	• Nearly unlimited data fits • Learn new information • Correct incorrect information • Less cost afterwards if smaller model • Use RAG too
Cons	• Much less data fits • Forgets data • Hallucinations • RAG misses, or gets incorrect data	• More high-quality data • Upfront compute cost • Needs some technical knowledge, esp. data
	Generic, side projects, prototypes	Domain-specific, enterprise, production usage, ...privacy!

Benefits of finetuning your own LLM

Performance

- stop hallucinations
- increase consistency
- reduce unwanted info

Privacy

- on-prem or VPC
- prevent leakage
- no breaches

Cost

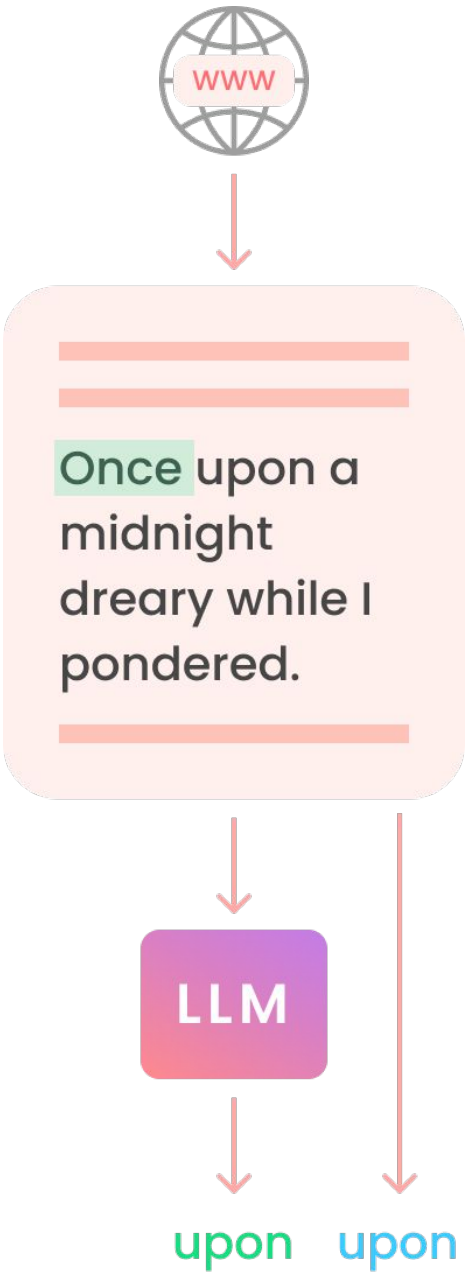
- lower cost per request
- increased transparency
- greater control

Reliability

- control uptime
- lower latency
- moderation

Where finetuning fits in

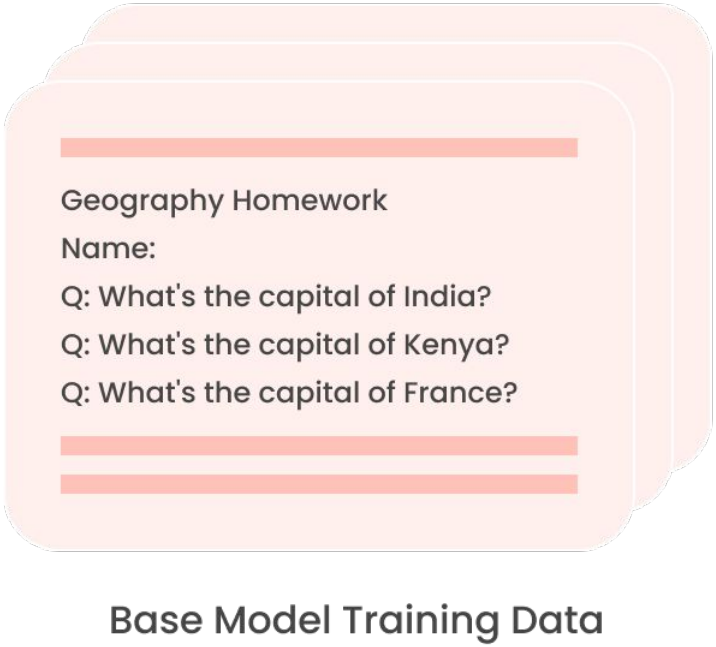
Pretraining



- Model at the start:
  - Zero knowledge about the world
  - Can't form English words
- Next token prediction
- Giant corpus of text data
- Often scraped from the internet: "unlabeled"
- Self-supervised learning

- After Training
  - Learns language
  - Learns knowledge

Limitations of pretrained base models



Base Model Training Data

Input

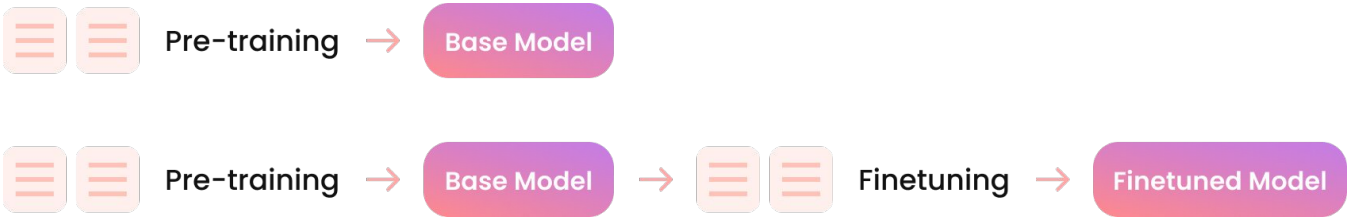
What is the capital of Mexico?

LLM

Output

"What is the capital of Hungary?"

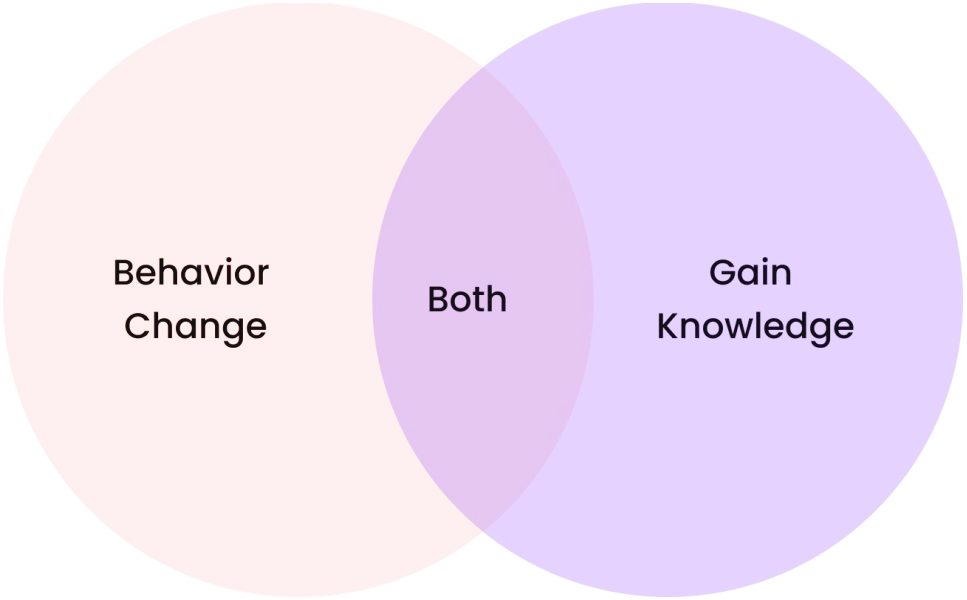
Finetuning after pretraining



- Finetuning usually refers to training further
  - Can also be self-supervised unlabeled data
  - Can be “labeled” data you curated
  - Much less data needed
  - Tool in your toolbox
- Finetuning for generative tasks is not well-defined:
  - Updates entire model, not just part of it
  - Same training objective: next token prediction
  - More advanced ways reduce how much to update (more later!)

What is finetuning doing for you?

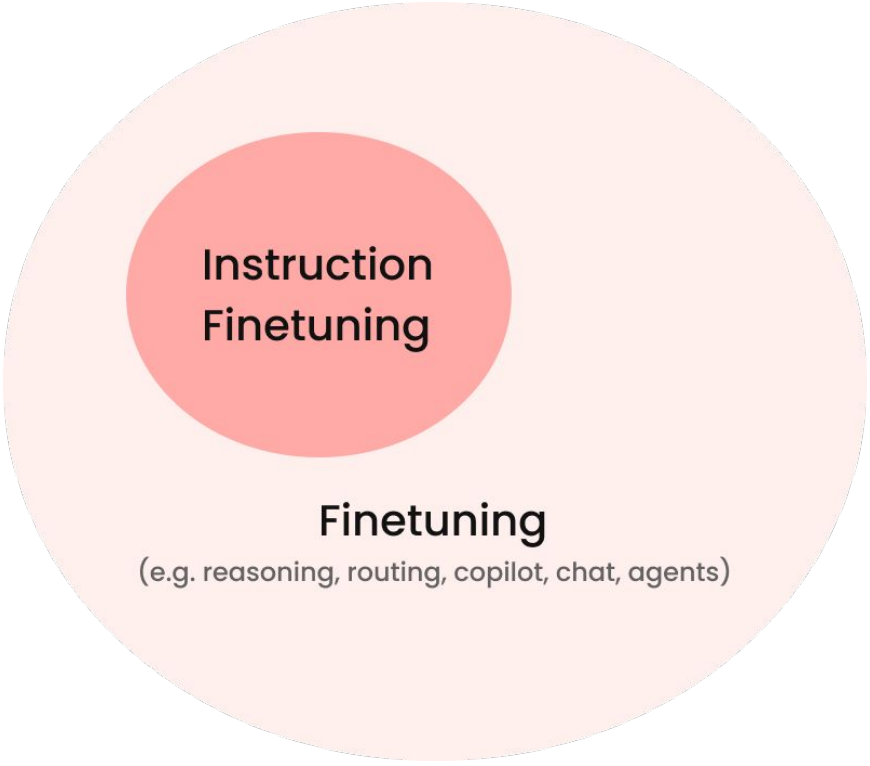
- Behavior change
  - Learning to respond more consistently
  - Learning to focus, e.g. moderation
  - Teasing out capability, e.g. better at conversation
- Gain knowledge
  - Increasing knowledge of new specific concepts
  - Correcting old incorrect information
- Both



First time finetuning

1. Identify task(s) by prompt-engineering a large LLM
2. Find tasks that you see an LLM doing ~OK at
3. Pick one task
4. Get ~1000 inputs and outputs for the task
   Better than the ~OK from the LLM
5. Finetune a small LLM on this data

What is instruction finetuning?



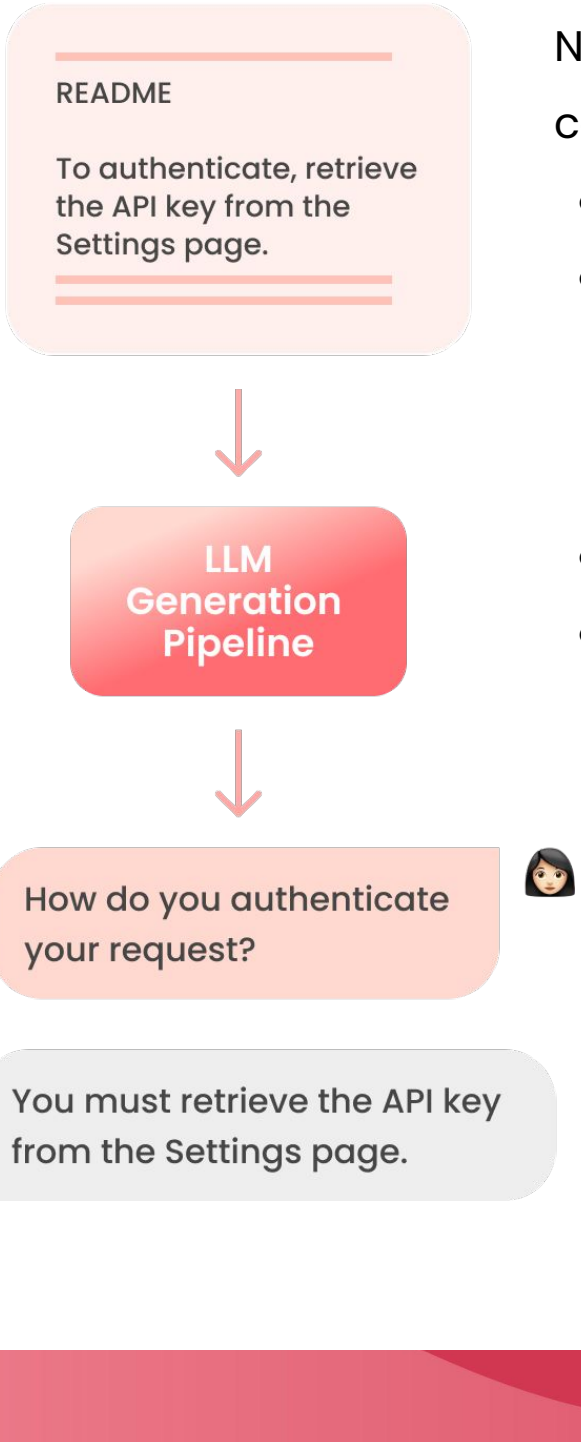
- AKA "instruction-tuned" or "instruction-following" LLMs
- Teaches model to behave more like a chatbot
- Better user interface for model interaction
    - Turned GPT-3 into ChatGPT
    - Increase AI adoption, from thousands of researchers to millions of people

Instruction-following datasets

Some existing data is ready as-is, online:

- FAQs
- Customer support conversations
- Slack messages

LLM Data Generation



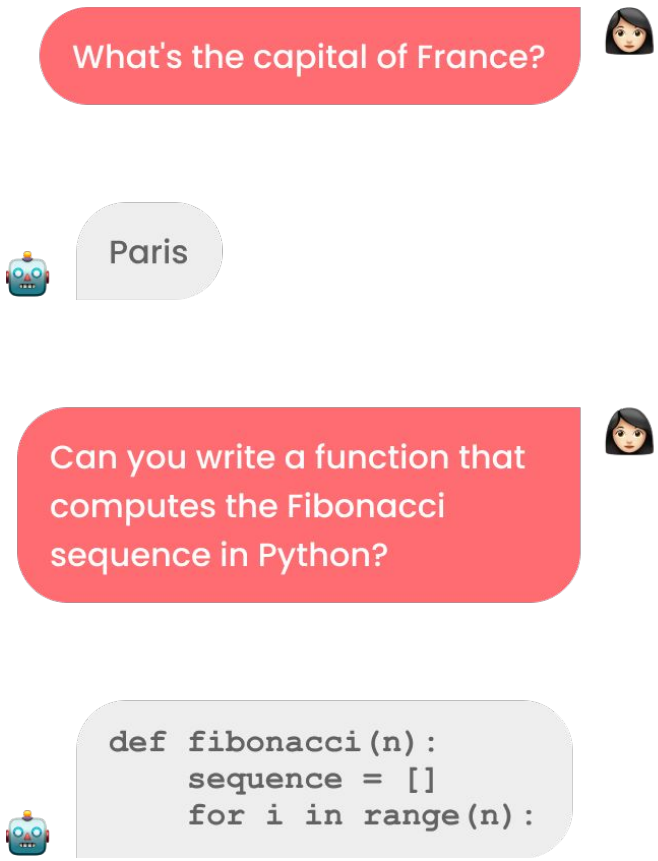
Non-Q&A data can also be converted to Q&A

- Using a prompt template
- Using another LLM

- ChatGPT ("Alpaca")
- Open-source models

Instruction Finetuning Generalization

- Can access model's pre-existing knowledge
- Generalize following instructions to other data, not in finetuning dataset

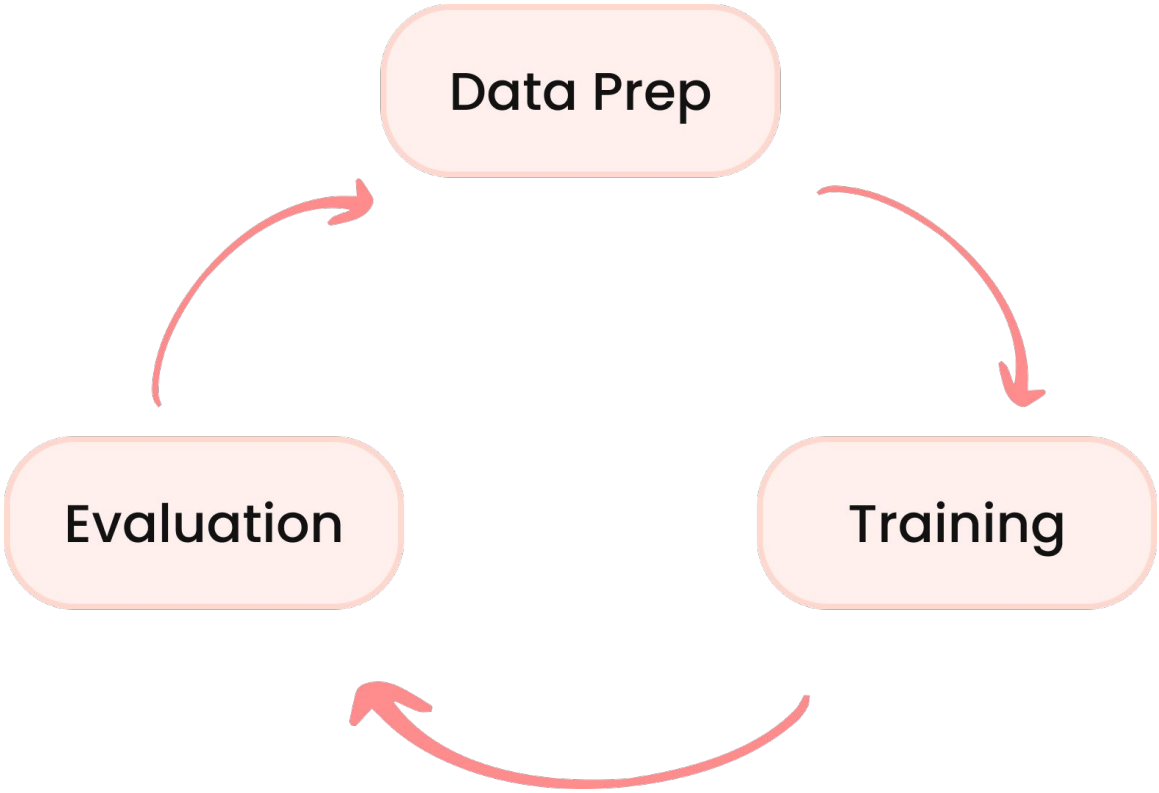


Finetuning Data

Code not in finetuning data, only base data

Model can now answer

Overview of Finetuning



Different Types of Finetuning