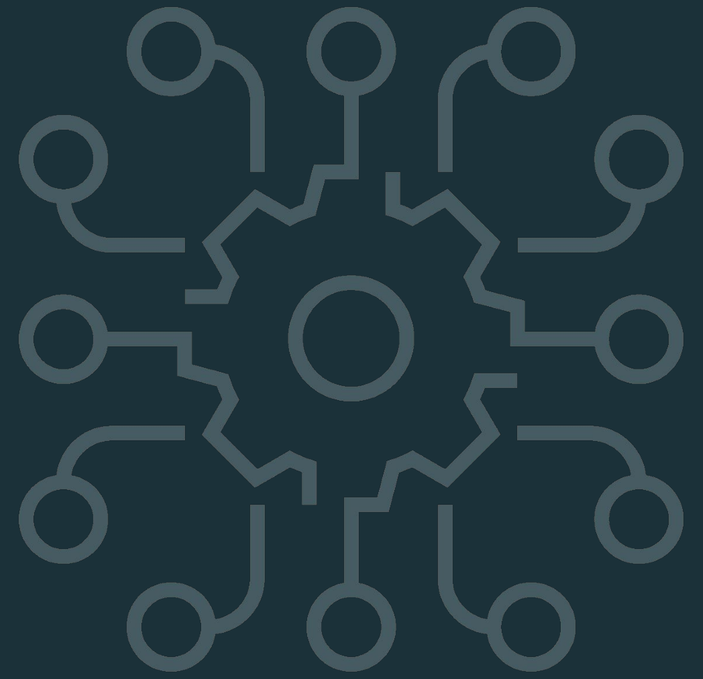
databricks
Academy

Fine-Tuning LLMs



Databricks Academy
2023

Learning Objectives

By the end of this module, you should be able to:

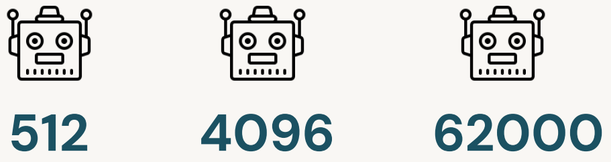
- Explain when and how to fine-tune models.
- Discuss the advantages and disadvantages of each method for fine-tuning models.
- Discuss the potential advantages of building your own model using own data.
- Describe common tools for training and fine-tuning, such as those from Hugging Face and DeepSpeed.

A Typical LLM Release

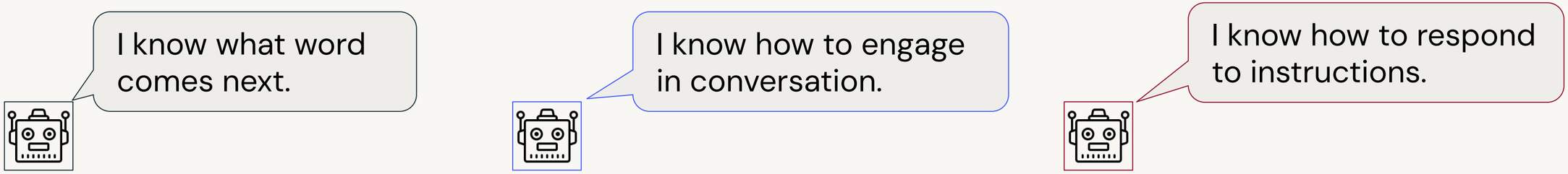
A new generative LLM release is comprised of

Multiple **sizes** (foundation/base model):

Multiple **sequence lengths:**



Flavors/fine-tuned versions (**base**, **chat**, **instruct**):

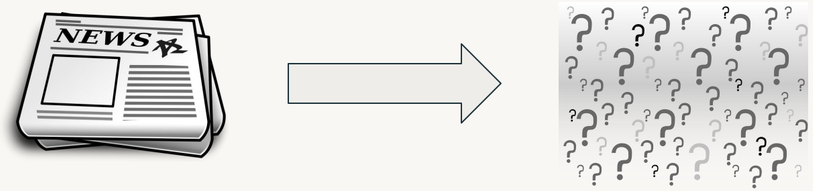


As a developer, which do you use?

For each use case, you need to balance:

- **Accuracy** (favors larger models)
- Speed (favors smaller models)
- *Task-specific performance*: (favors more narrowly fine-tuned models)

Let's look at example: **a news article summary app for riddlers**.



Fine-Tuning LLMs:

Applying Foundation LLMs

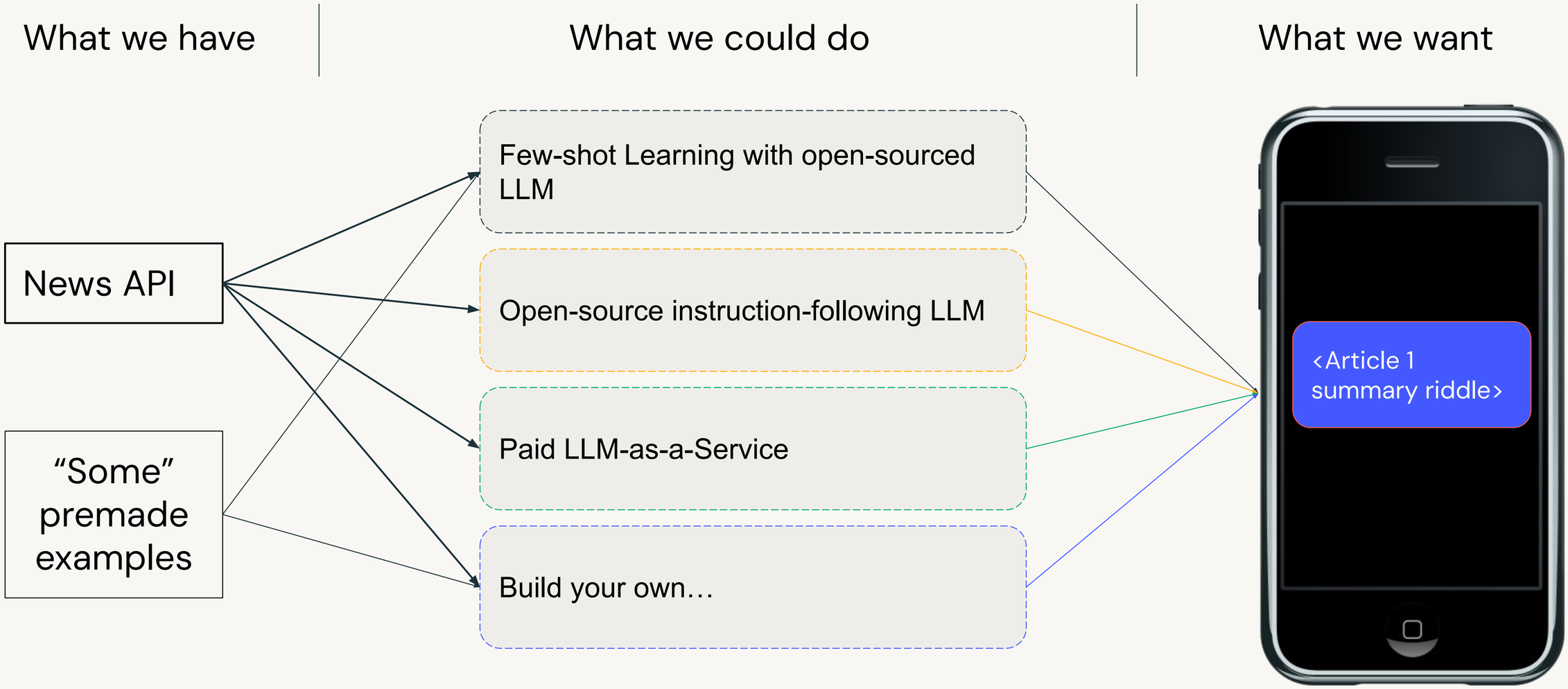
News Article Summaries App for Riddlers

My App - Riddle me this:
I want to create engaging and accurate article summaries for users in the form of *riddles*.

By the river's edge, a secret lies,
A treasure chest of a grand prize.
Buried by a pirate, a legend so old,
Whispered secrets and stories untold.
What is this enchanting mystery found?
In a riddle's realm, let your answer resound!

How do we build this?

Potential LLM Pipelines



Fine-Tuning LLMs:

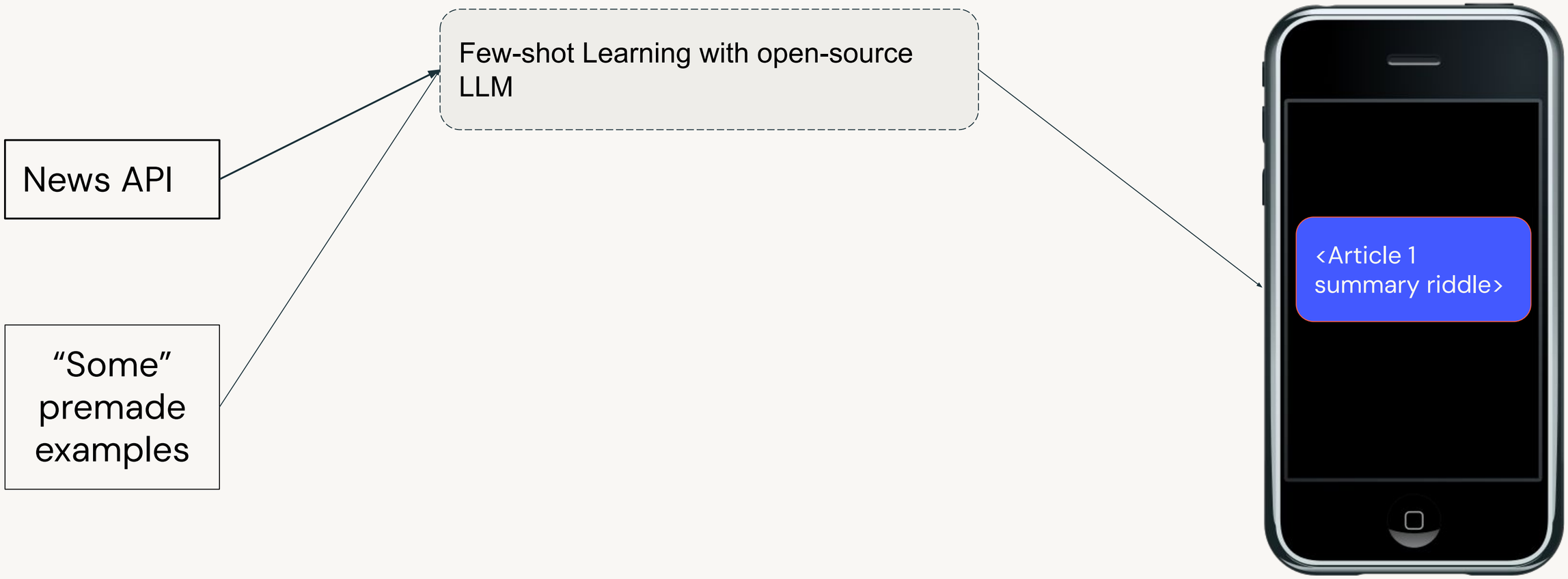
Fine-tuning
Few-shot Learning

Potential LLM Pipelines

What we have

What we could do

What we want



Pros and cons of Few-shot Learning

Pros

- Speed of development
  - Quick to get started and working.
- Performance
  - For a larger model, the few examples often lead to good performance
- Cost
  - Since we're using a released, open LLM, we only pay for the computation

Cons

- Data
  - Requires a number of good-quality examples that cover the intent of the task.
- Size-effect
  - Depending on how the base model was trained, we may need to use the largest version which can be unwieldy on moderate hardware.

Riddle me this: A Few-shot Learning version

Let’s build the app with few shot learning and the new LLM

Our new articles are long, and in addition to summarization, the LLM needs to reframe the output as a riddle.

- Large version of base LLM
- Long input sequence

```
prompt = (
"""For each article, summarize and create a riddle from
the summary:
[Article 1]: "Residents were awoken to the surprise..."
[Summary Riddle 1]: "In houses they stay, the peop... "
###
[Article 2]: "Gas prices reached an all time ..."
[Summary Riddle 1]: "Far you will drive, to find..."
###
...
###
[Article n]: {article}
[Summary Riddle n]:""")
```

Fine-Tuning LLMs:

Fine-tuning Instruction-following LLMs

Potential LLM Pipelines

What we have | What we could do | What we want

News API

"Some" premade examples

Instruction-following LLM

Pros and cons of Instruction-following LLMs

Pros

- Data
  - Requires no few-shot examples. Just the instructions (aka zero-shot learning).
- Performance
  - Depending on the dataset used to train the base and fine-tune this model, may already be well suited to the task.
- Cost
  - Since we're using a released, open LLM, we only pay for the computation.

Cons

- Quality of fine-tuning
  - If this model was not fine-tuned on similar data to the task, it will potentially perform poorly.
- Size-effect
  - Depending on how the base model was trained, we may need to use the largest version which can be unwieldy on moderate hardware.

Riddle me this: Instruction-following version

Let’s build the app with the Instruct version of the LLM

The new LLM was released with a number of fine-tuned flavors.

Let’s use the Instruction-following LLM one as is and leverage zero-shot learning.

```
prompt = (
"""For the article below, summarize and create
a riddle from the summary:
[Article n]: {article}
[Summary Riddle n]:""")
```

Fine-Tuning LLMs:

Fine-tuning LLMs-as-a-Service

Potential LLM Pipelines

What we have | What we could do | What we want

News API

"Some" premade examples

Paid LLM-as-a-Service

Pros and cons of LLM-as-a-Service

Pros

- Speed of development
  - Quick to get started and working.
  - As this is another API call, it will fit very easily into existing pipelines.
- Performance
  - Since the processing is done server side, you can use larger models for best performance.

Cons

- Cost
  - Pay for each token sent/received.
- Data Privacy/Security
  - You may not know how your data is being used.
- Vendor lock-in
  - Susceptible to vendor outages, deprecated features, etc.

Riddle me this: LLM-as-a-Service version

Let's build the app using an LLM-as-a-service/API

This requires the least amount of effort on our part.

Similar to the Instruction-following LLM version, we send the article and the instruction on what we want back.

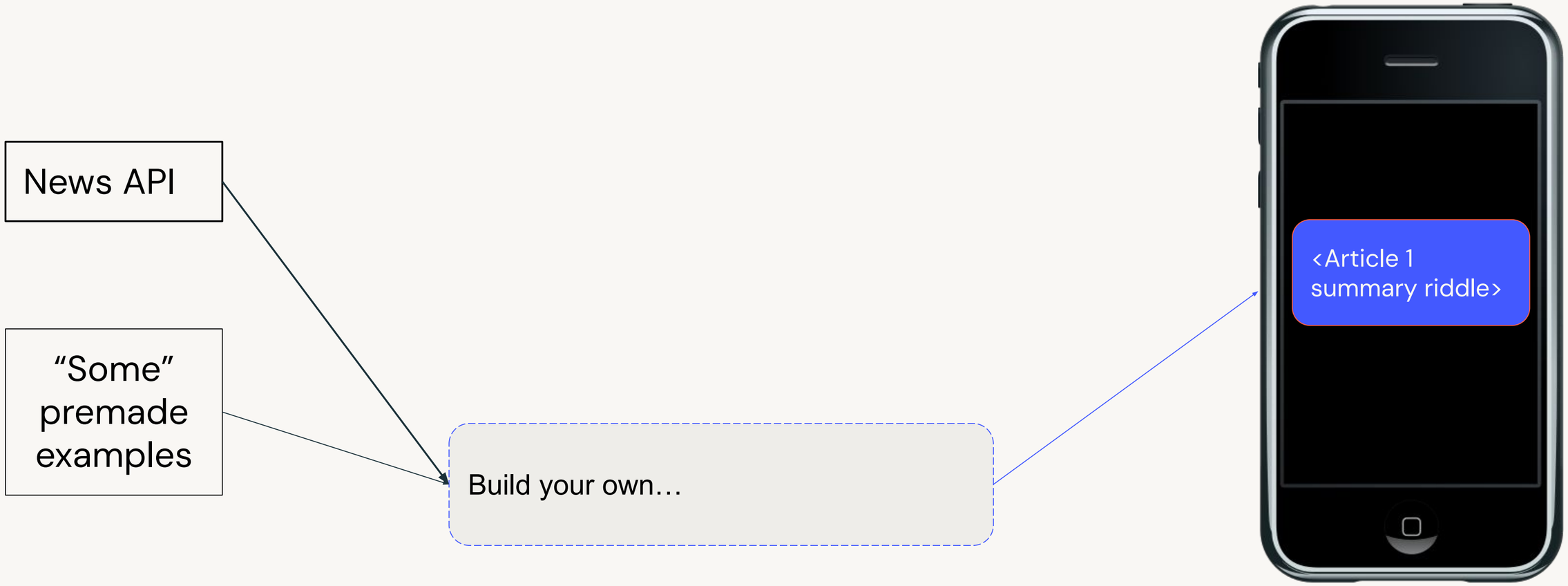
```
prompt = (
"""For the article below, summarize and create
a riddle from the summary:
[Article n]: {article}
[Summary Riddle n]:""")

response =
LLM_API(prompt(article),api_key="sk-@sjr...")
```

Fine-Tuning LLMs:

Fine-tuning DIY

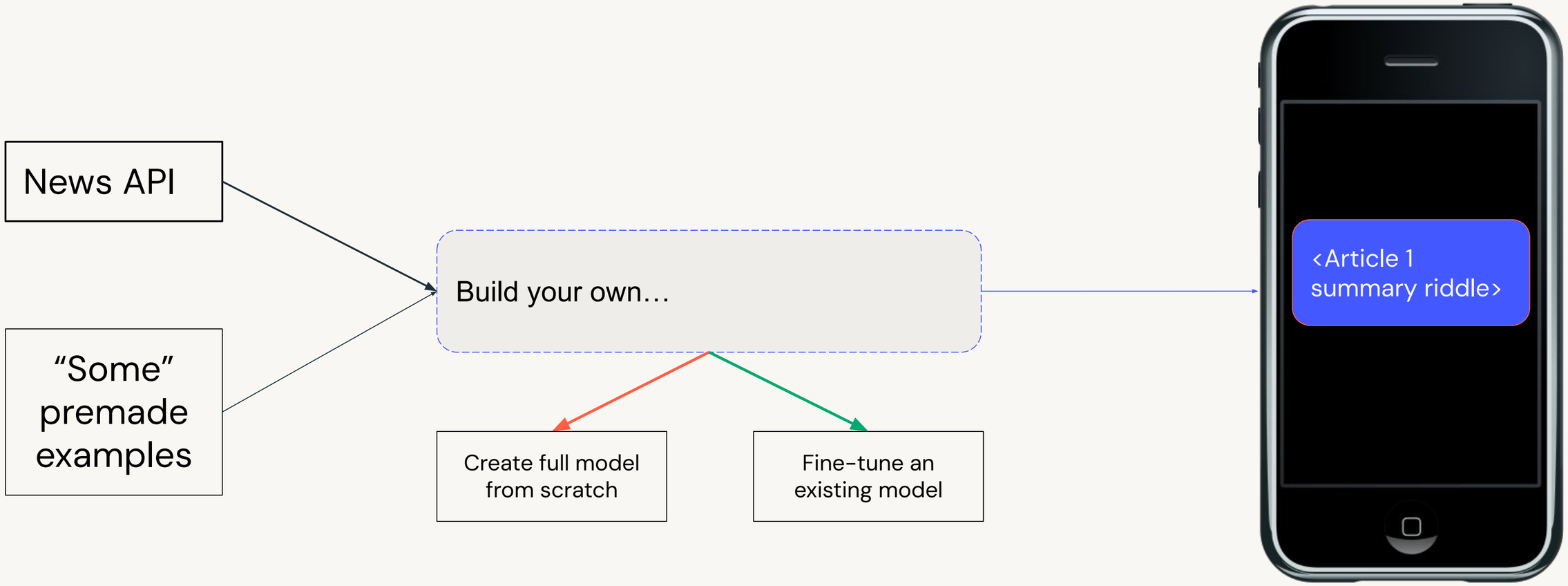
Potential LLM Pipelines

What we have | What we could do | What we want



Potential LLM Pipelines

What we have | What we could do | What we want



Pros and cons of fine-tuning an existing LLM

Pros

- Task-tailoring
  - Create a task-specific model for your use case.
- Inference Cost
  - More tailored models often smaller, making them faster at inference time.
- Control
  - All of the data and model information stays entirely within your locus of control.

Cons

- Time and Compute Cost
  - This is the most costly use of an LLM as it will require both training time and computation cost.
- Data Requirements
  - Larger models require larger datasets.
- Skill Sets
  - Require in-house expertise.

Riddle me this: fine-tuning version

Let's build the app using a fine-tuned version of the LLM

Depending on the amount and quality of data we already have, we can do one of the following:

- Self-instruct (**LLama 2** and **MPT-7B**)
  - Use another LLM to generate synthetic data samples for data augmentation.
- High-quality fine-tune (Dolly v2)
  - Go straight to fine-tuning, if data size and quality is satisfactory.

Why an organization would consider building their **own model** or **fine tuning** a model?

One giant ML model
for **every** use case
owned by **1 company**

vs.

Millions of models
for **specific** use cases
owned by **many companies**

‘Off the Shelf’ offerings insufficient for enterprise

Shared, unsecure services

Great for general demos of “intelligence”

Doesn’t work well for domain specific use in enterprise

Enterprises require **customization** on their proprietary data

Enterprise requires **secure access** to services

You have amazing data, it will be your

competitive advantage

Demo

Fine-tuning LLMs

Outline

- Fine-tuning a T5 model
  - Data Preparation
  - Training the model
  - Testing the final model
- Fine-tuning with DeepSpeed
  - Environment setup and configuration
  - Training the model
  - Testing the final model

Lab

Fine-tuning LLMs

Outline

- Fine-tuning a LLM model
  - Data preparation
  - Load and configure pre-trained model
  - Tokenize
  - AutoModelForCasualLM
  - Train
  - Evaluate

databricks
Academy

Module Summary and Next Steps

Databricks Academy
2023

Module Summary

Let's review

- There are various potential LLM pipelines for building LLM apps.
- Each method has pros and cons. It is important to evaluate these per project requirements.
- Fine-tuning models can be useful or even necessary to ensure a good fit for the task.
- Fine-tuning is essentially the same as training, just starting from a checkpoint.
- Tools have been developed to improve the training/fine-tuning process.

Helpful Resources

Resources and tools for fine-tuning LLMs

- Fine-tuning models
  - HF leaderboard
  - MPT-7B
  - LLama 2
  - Vicuna
  - DeepSpeed on Databricks