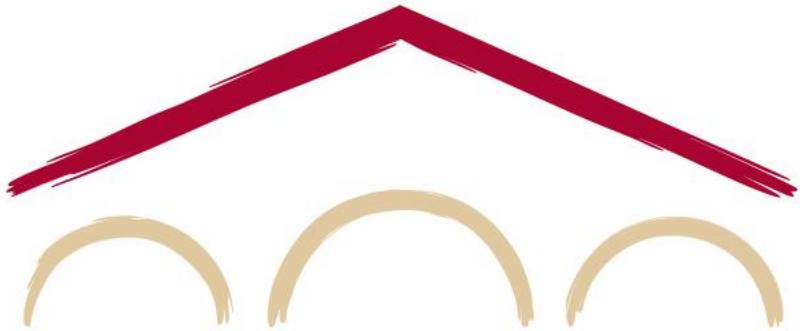
Assign/bakeoff 2 overview



Christopher Potts

CS224u: Natural Language Understanding

Homework and bakeoff: Few-shot OpenQA with DSP

```
__author__ = "Christopher Potts and Omar Khattab"
__version__ = "CS224u, Stanford, Spring 2023"
```

QA tasks

Task	Passage given	Task-specific reader training	Task-specific retriever training
QA	yes	yes	n/a
OpenQA	no	yes	maybe
Few-shot QA	yes	no	n/a
Few-shot OpenQA	no	no	maybe

Your situation:

1. During development, you have gold Q/A pairs.
2. At test time, all you have is Qs – no gold passages or other associated data.
3. You cannot train any LLMs: all you can do is in-context learning with frozen models.

GPT-3 paper: Few-shot QA

Title: The_Blitz

Background: From the German point of view, March 1941 saw an improvement. The Luftwaffe flew 4,000 sorties that month, including 12 major and three heavy attacks. The electronic war intensified but the Luftwaffe flew major inland missions only on moonlit nights. Ports were easier to find and made better targets. To confuse the British, radio silence was observed until the bombs fell. X- and Y-Gerät beams were placed over false targets and switched only at the last minute. Rapid frequency changes were introduced for X-Gerät, whose wider band of frequencies and greater tactical flexibility ensured it remained effective at a time when British selective jamming was degrading the effectiveness of Y-Gerät.

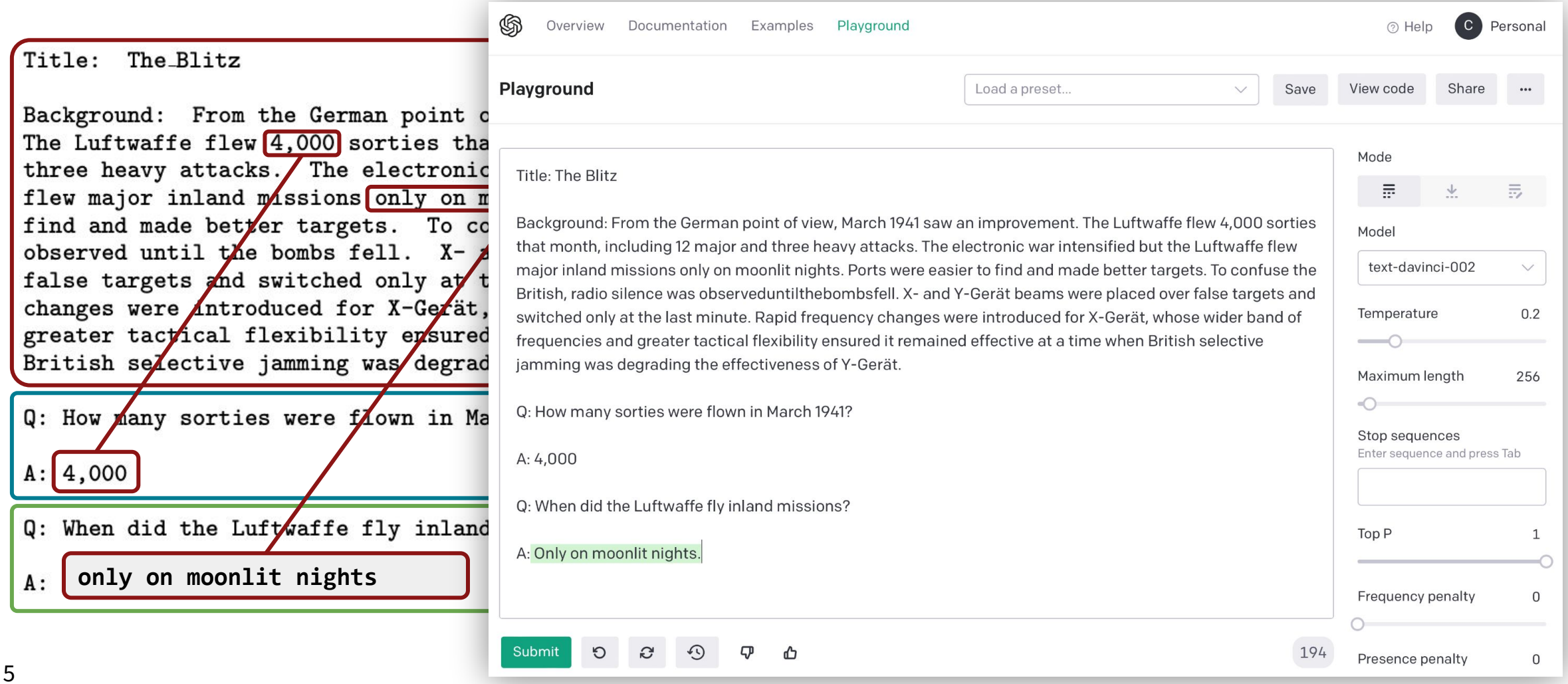
Q: How many sorties were flown in March 1941?

A: 4,000

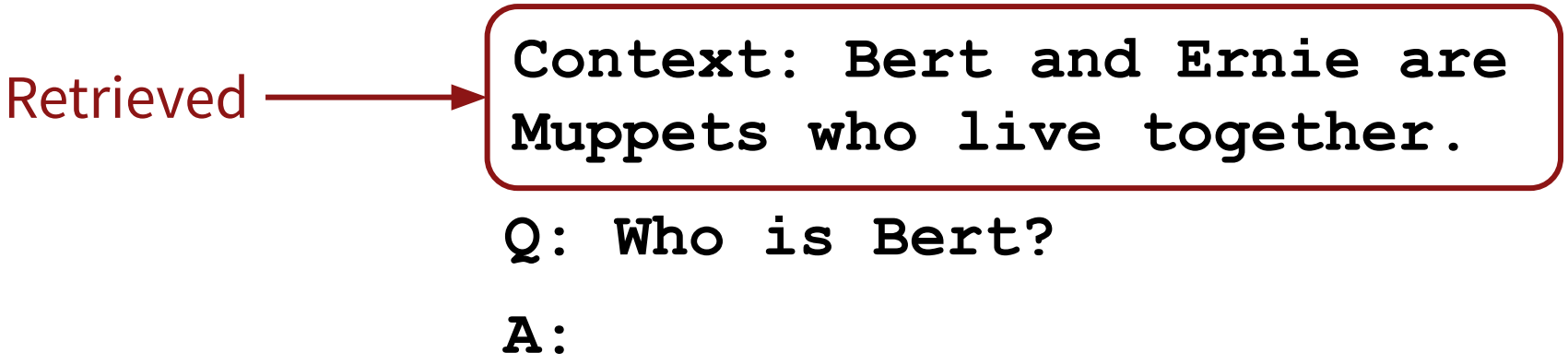
Q: When did the Luftwaffe fly inland missions?

A: only on moonlit nights

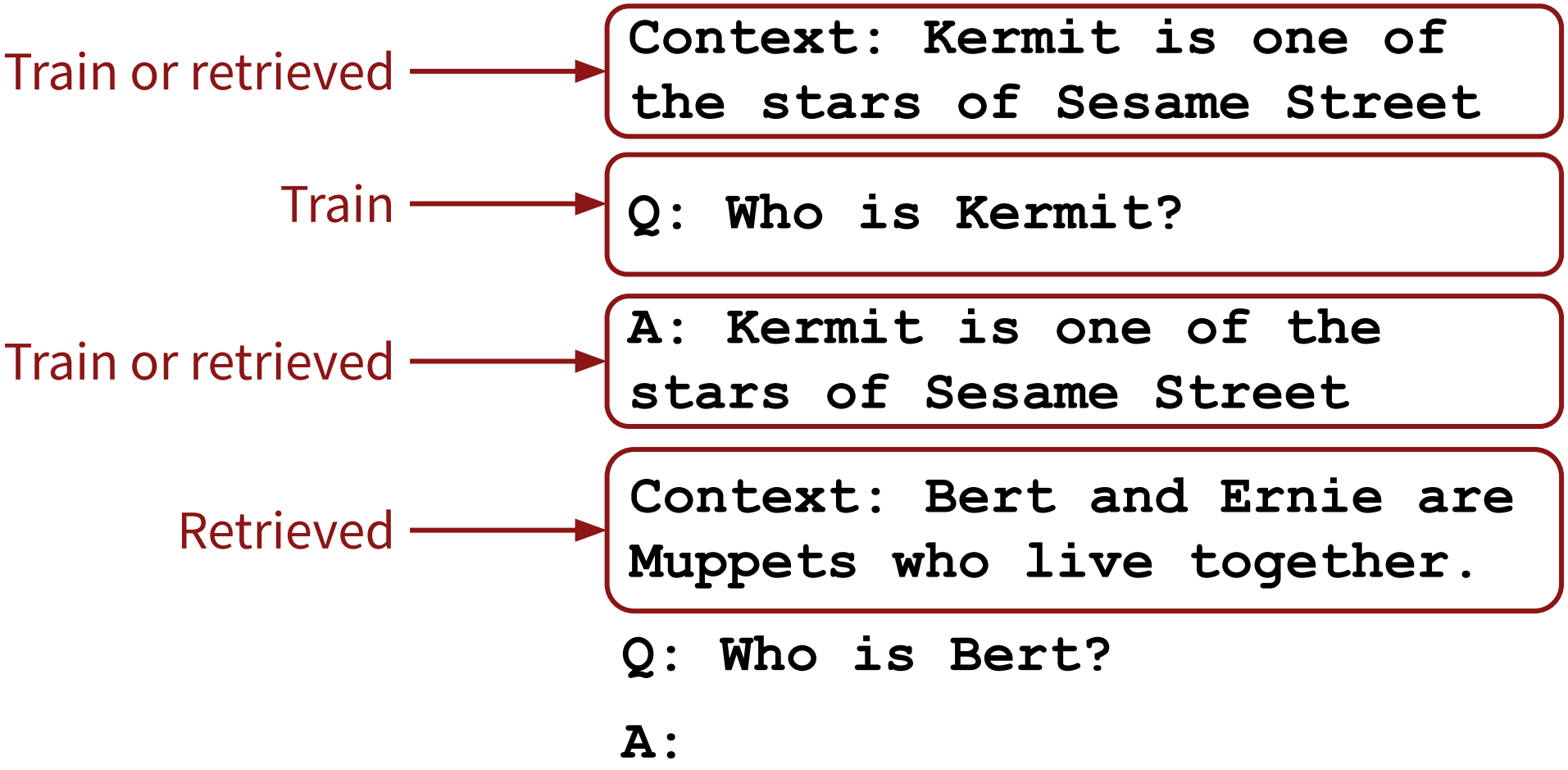
GPT-3 paper: Few-shot QA



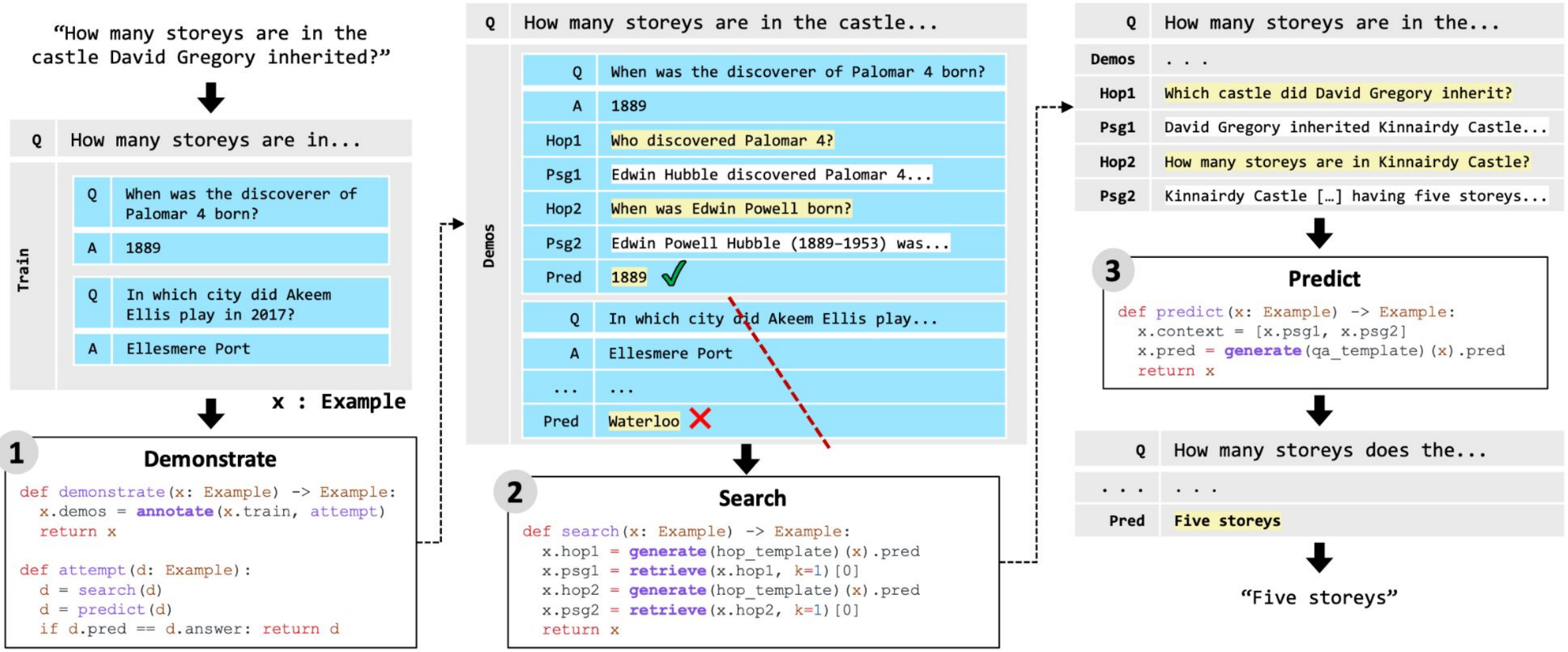
Retrieve-then-read



Few-shot retrieve-then-read



DEMONSTRATE–SEARCH–PREDICT: Composing retrieval and language models



Set-up

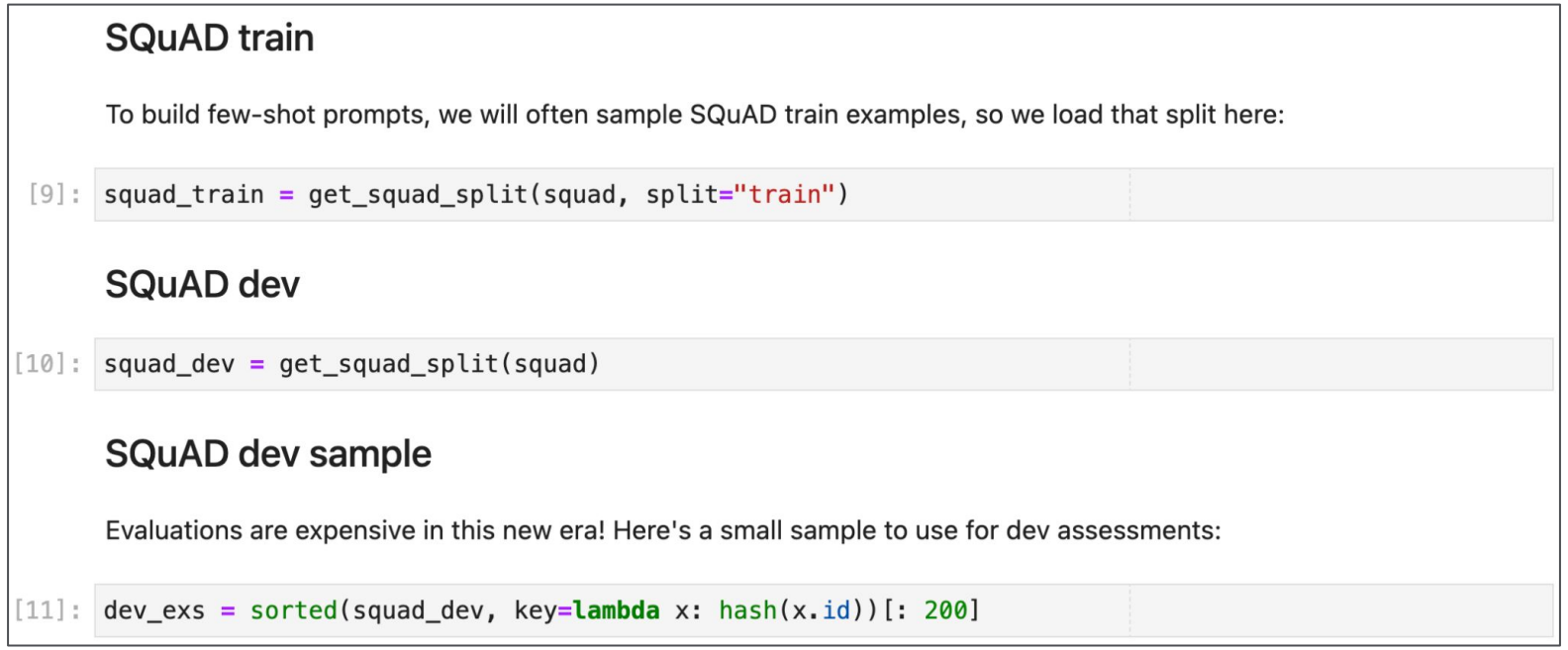
[4]:
```
os.environ["DSP_NOTEBOOK_CACHEDIR"] = os.path.join(root_path, 'cache')

openai_key = os.getenv('OPENAI_API_KEY')  # or replace with your API key (optional)

cohere_key = os.getenv('COHERE_API_KEY')  # or replace with your API key (optional)

colbert_server = 'http://ec2-44-228-128-229.us-west-2.compute.amazonaws.com:8893/api/search'
```

Here we establish the Language Model `lm` and Retriever Model `rm` that we will be using. The defaults for `lm` are just for development. You may want to develop using an inexpensive model and then do your final evalautions wih an expensive one.

[5]:
```
lm = dsp.GPT3(model='text-davinci-001', api_key=openai_key)

# Options for Cohere: command-medium-nightly, command-xlarge-nightly
#lm = dsp.Cohere(model='command-xlarge-nightly', api_key=cohere_key)

rm = dsp.ColBERTv2(url=colbert_server)

dsp.settings.configure(lm=lm, rm=rm)
```

SQuAD for “train” and dev

- SQuAD provides some “train” data containing gold Q/A pairs with gold passages that you can use for demonstrations.
- SQuAD also provides a dev set of Qs with gold As that you can use to simulate your actual situation.



SQuAD train

To build few-shot prompts, we will often sample SQuAD train examples, so we load that split here:

[9]:
```
squad_train = get_squad_split(squad, split="train")
```

SQuAD dev

[10]:
```
squad_dev = get_squad_split(squad)
```

SQuAD dev sample

Evaluations are expensive in this new era! Here's a small sample to use for dev assessments:

[11]:
```
dev_exs = sorted(squad_dev, key=lambda x: hash(x.id))[: 200]
```

Direct us of `lm` (mostly not done)

[13]:
```
lm("Which U.S. states border no U.S. states?")
```

[13]:
```
['\n\nAlaska and Hawaii are the only U.S. states that border no other U.S. states.']
```

Keyword arguments to the underlying LM are passed through:

[14]:
```
lm("Which U.S. states border no U.S. states?", temperature=0.9, n=4)
```

[14]:
```
['\n\nThe state of Alaska borders no other U.S. states.',
 '\n\nAlaska and Hawaii.',
 '\n\nHawaii and Alaska',
 '\n\nThe U.S. states that border no other U.S. states are Maine, New Hampshire, Vermont, Massachu
setts, Rhode Island, and Connecticut.']
```

With `lm.inspect_history`, we can see the most recent language model calls:

[15]:
```
lm.inspect_history(n=1)
```

```
Which U.S. states border no U.S. states?

The state of Alaska borders no other U.S. states.        (and 3 other completions)
```

Templates

```
[16]: Question = dsp.Type(
          prefix="Question:",
          desc="${the question to be answered}")

      Answer = dsp.Type(
          prefix="Answer:",
          desc="${a short factoid answer, often between 1 and 5 words}"
          format=dsp.format_answers)

      qa_template = dsp.Template(
          instructions="Answer questions with short factoid answers.",
          question=Question(),
          answer=Answer())
```

And here is a self-contained example that uses our question and template to create a prompt:

```
[17]: states_ex = dsp.Example(
          question="Which U.S. states border no U.S. states?",
          demos=dsp.sample(squad_train, k=2))

      print(qa_template(states_ex))
```

Sampled SQuAD demos

```
Answer questions with short factoid answers.

---

Follow the following format.

Question: ${the question to be answered}
Answer: ${a short factoid answer, often between 1 and 5 words}

---

Question: What album made her a worldwide known artist?
Answer: Dangerously in Love

Question: Immunoassays are able to detect what type of proteins?
Answer: generated by an infected organism in response to a foreign agent

Question: Which U.S. states border no U.S. states?
Answer:
```

Prompt-based generation

```
[18]: states_ex, states_compl = dsp.generate(qa_template)(states_ex, stage='basics')

[19]: print(states_compl.answer)

Alaska, Hawaii
```

And here's precisely what the model saw and did:

```
[20]: lm.inspect_history(n=1)
```

```
Answer questions with short factoid answers.

---

Follow the following format.

Question: ${the question to be answered}
Answer: ${a short factoid answer, often between 1 and 5 words}

---

Question: What album made her a worldwide known artist?
Answer: Dangerously in Love

Question: Immunoassays are able to detect what type of proteins?
Answer: generated by an infected organism in response to a foreign agent

Question: Which U.S. states border no U.S. states?
Answer: Alaska, Hawaii
```

Retrieval with ColBERT

[21]:
```
states_ex.question
```

[21]:
```
'Which U.S. states border no U.S. states?'
```

The basic `dsp.retrieve` method returns only passages:

[22]:
```
passages = dsp.retrieve(states_ex.question, k=1)
```

[23]:
```
passages
```

[23]:
```
['Mexico–United States border | has the shortest. Among the states in Mexico, Chihuahua has the lo
ngest border with the United States, while Nuevo León has the shortest. Texas borders four Mexican
states—Tamaulipas, Nuevo León, Coahuila, and Chihuahua—the most of any U.S. states. New Mexico and
Arizona each borders two Mexican states (Chihuahua and Sonora; Sonora and Baja California, respect
ively). California borders only Baja California. Three Mexican states border two U.S. states each:
Baja California borders California and Arizona; Sonora borders Arizona and New Mexico; and Chihuah
ua borders New Mexico and Texas. Tamaulipas, Nuevo León, and Coahuila each borders only one U.S. s
tate: Texas. The']
```

If we need passages with scores and other metadata, we can call `rm` directly:

[24]:
```
rm(states_ex.question, k=1)
```

Few-shot OpenQA

Use this decorator so that programs don't modify examples!

Programs operate on single `dsp.Example` instances

```
[25]: @dsp.transformation
      def few_shot_openqa(example, train=squad_train, k=2):
          example.demos = dsp.sample(train, k=k)
          example, completions = dsp.generate(qa_template)(example, stage='qa')
          return completions
```

k random demonstrations

`dsp.Completions`, which have an answer attribute supplied by `qa_template`

The generator function

The `qa_template` we defined on slide 12

Assignment questions

- Both of the assignment questions are DSP programs like the one we just walked through:
  a. Question 1: Few-shot OpenQA with context
  b. Question 2: Using annotate
- Your original system can then be an original DSP program (though this is not required).
- The DSP intro.ipynb walks through additional advanced programs for hard QA problems.