Task Ambiguity in Humans and Language Models

Alex Tamkin*, Kunal Handa*,
Avash Shrestha, Noah Goodman

ICLR 2023

atamkin@cs.stanford.edu

How do we typically think of progress in machine learning?

Measuring progress in machine learning

1. Identify a capability of interest
2. Operationalize that capability clearly
   a. Clear examples
   b. Clear task description
   c. Choice of metric

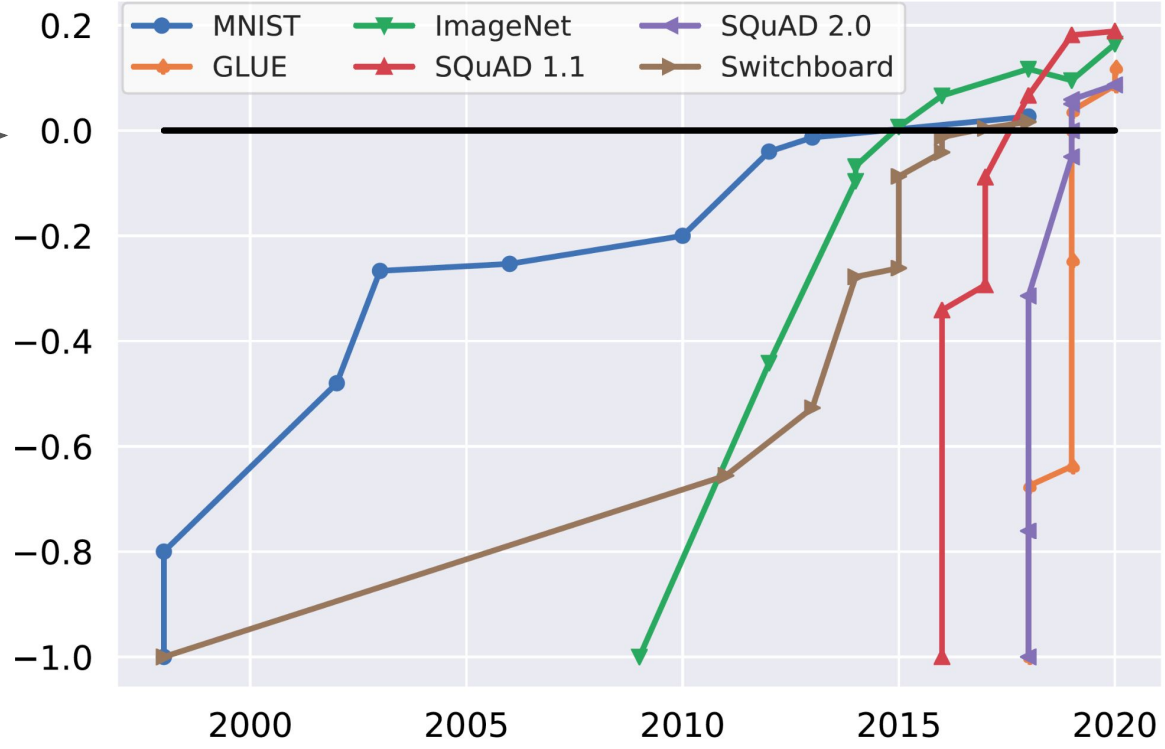
"Evaluation"

3. Produce a model that performs well
   a. Training (modeling, optimization)
   b. Adaptation (finetuning, prompting)

"Methods"

As a field, we've learned how to do Step 3 almost too well!

Normalized human performance



Dynabench (Kiela+ 2021)

Measuring progress in machine learning

1. Identify a capability of interest
2. **Operationalize that capability clearly**
   a. Clear examples
   b. Clear task description
   c. Choice of metric

"Evaluation"

3. Produce a model that performs well
   a. Training (modeling, optimization)
   b. Adaptation (finetuning, prompting)

"Methods"

Operationalizing the task has been relatively neglected in ML research

Operationalization

The transformation of an abstract, theoretical concept into something concrete, observable, and measurable in an empirical research project.
Scott and Marshall, 2019

Operationalization turns vague or ambiguous concepts into detailed descriptions, which can be measured. **Eyler, 2020**

A shift in perspective

So far, we've largely **delegated** task operationalization to the benchmark creators

But ML is **moving beyond** benchmark performance on a few standard tasks

We're applying these models to millions of bespoke tasks **on-the-fly**

People won't (or can't) to spend **hours or days** ensuring each prompt is perfectly unambiguous

Prompt Engineering

Prompt engineering

12 languages

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

Prompt engineering is a concept in artificial intelligence (AI), particularly natural language processing (NLP). In prompt engineering, the description of the task that the AI is supposed to accomplish is embedded in the input, e.g., as a question, instead of it being implicitly given. Prompt engineering typically works by converting one or more tasks to a prompt-based dataset and training a language model with what has been called "prompt-based learning" or just "prompt learning".[1][2] Prompt engineering may work from a large "frozen" pretrained language model where only the representation of the prompt is learned (i.e., optimized), using methods such as "prefix-tuning" or "prompt tuning".[3][4]

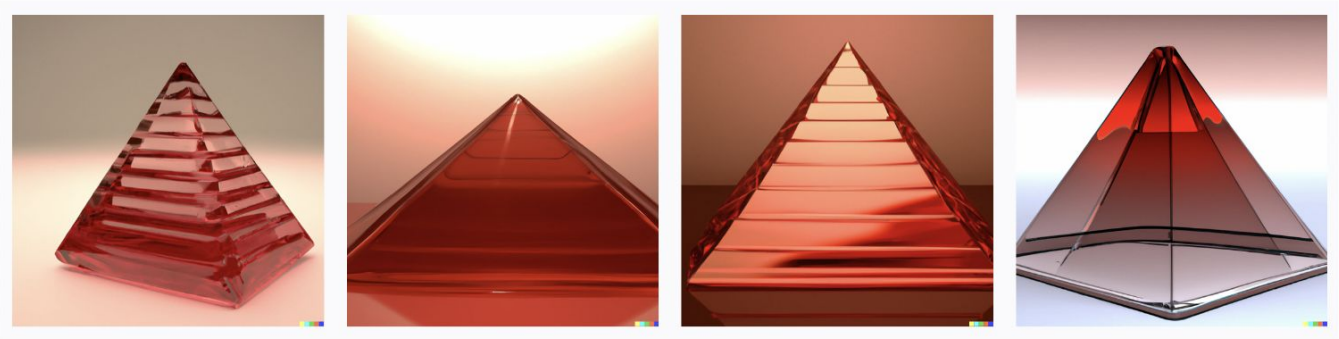
The GPT-2 and GPT-3 language models[5] were important steps in prompt engineering. In 2021, multitask[*jargon*] prompt engineering using multiple NLP datasets showed good performance on new tasks.[6] In a method called chain-of-thought (CoT) prompting, few-shot examples of a task are given to the language model which improves its ability to reason.[7] CoT prompting can also be a zero-

Prompt Engineering

Here are a few pyramids generated by DALLE, with the prompt `pyramid`.

Prompt Engineering

Here are a few pyramids generated by DALLE, with the prompt `A pyramid made of glass, rendered in Unity and tinted red`, which uses 3 style modifiers.



Prompt Engineering

Since a shell within Linux (or Windows PowerShell) is also a programming language, you can interact with it using ChatGPT, and build an environment for your filesystem:

```
Act as Debian Linux command shell. Please respond to my commands as the terminal would, with as
little explanation as possible. My first command is: ls -l
```

The output will be something like:

```
-rw-r--r--  1 user  group   2048 Mar  3 14:23 file1.txt
drwxr-xr-x  2 user  group   4096 Feb 28 09:12 directory1
-rwx------  1 user  group  16384 Feb 25 19:41 executable1
```

The state of things

1. The tasks we want models to perform are increasingly complex
2. It's hard to specify all the dimensions of the desired task behavior
3. A model's behavior can be arbitrary along unspecified dimensions

A daunting picture!

A real-world example with GPT-3

Training: Make a server for Amy [...]

```
—> <...command...>,
   location = USA
```

Test: Make a server for Nikhil [...]

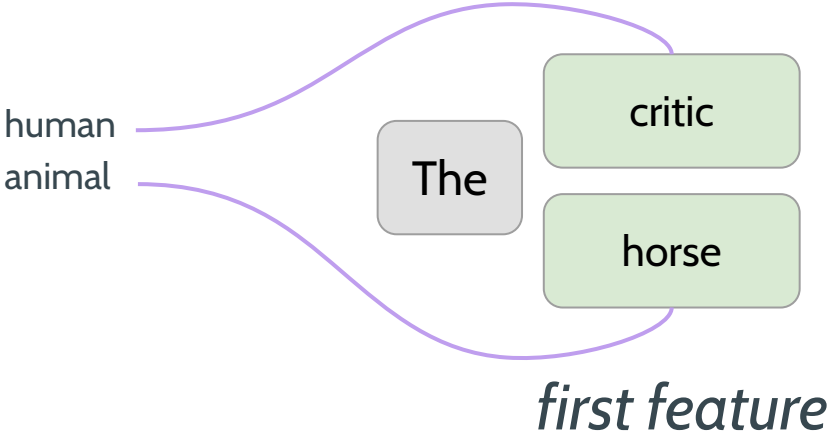
```
-> <...command...>,
   location = India
```

AmbiBench: A simple testbed for task ambiguity in language models

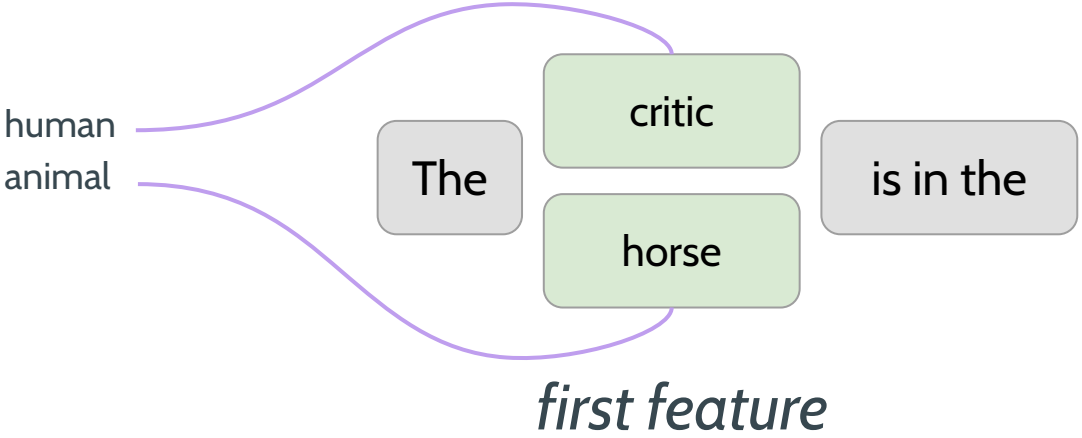
Multiple-feature ambiguity

The

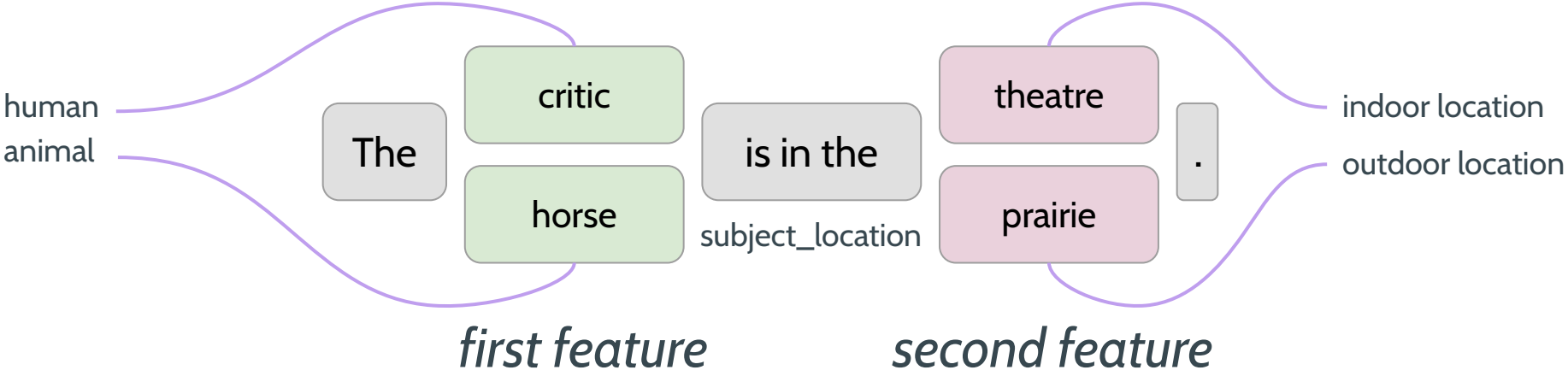
Multiple-feature ambiguity



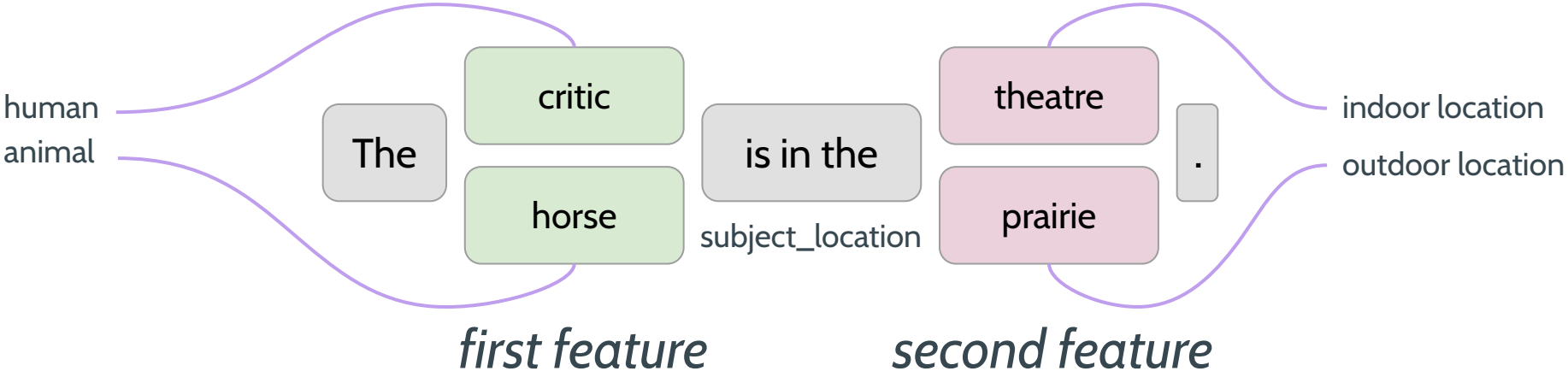
Multiple-feature ambiguity



Multiple-feature ambiguity



Multiple-feature ambiguity



Every example has multiple possible labels!

Can models use examples to disambiguate between two possible tasks?

Output ‘X’ if the sentence contains a **[category withheld]** and ‘Y’ otherwise.

The worm is in the meadow > ‘X’

The student is in the museum > ‘Y’

😵 *This is unclear. What category am I looking for?*

🤔 *Hmm... it looks like the category is either animals or outdoor locations?*

Output ‘X’ if the sentence contains a **[category withheld]** and ‘Y’ otherwise.

The worm is in the meadow > ‘X’

The student is in the museum > ‘Y’

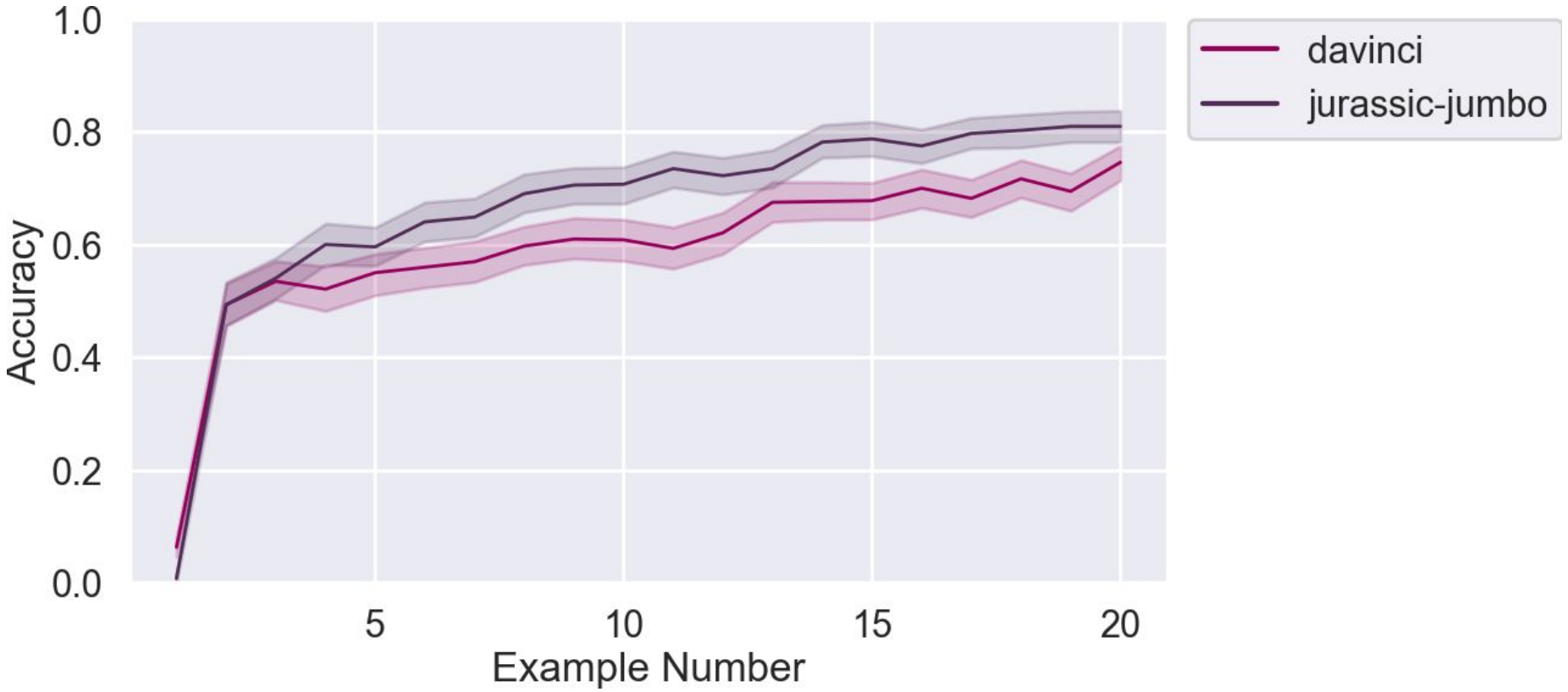
The duck is in the canyon > ‘X’

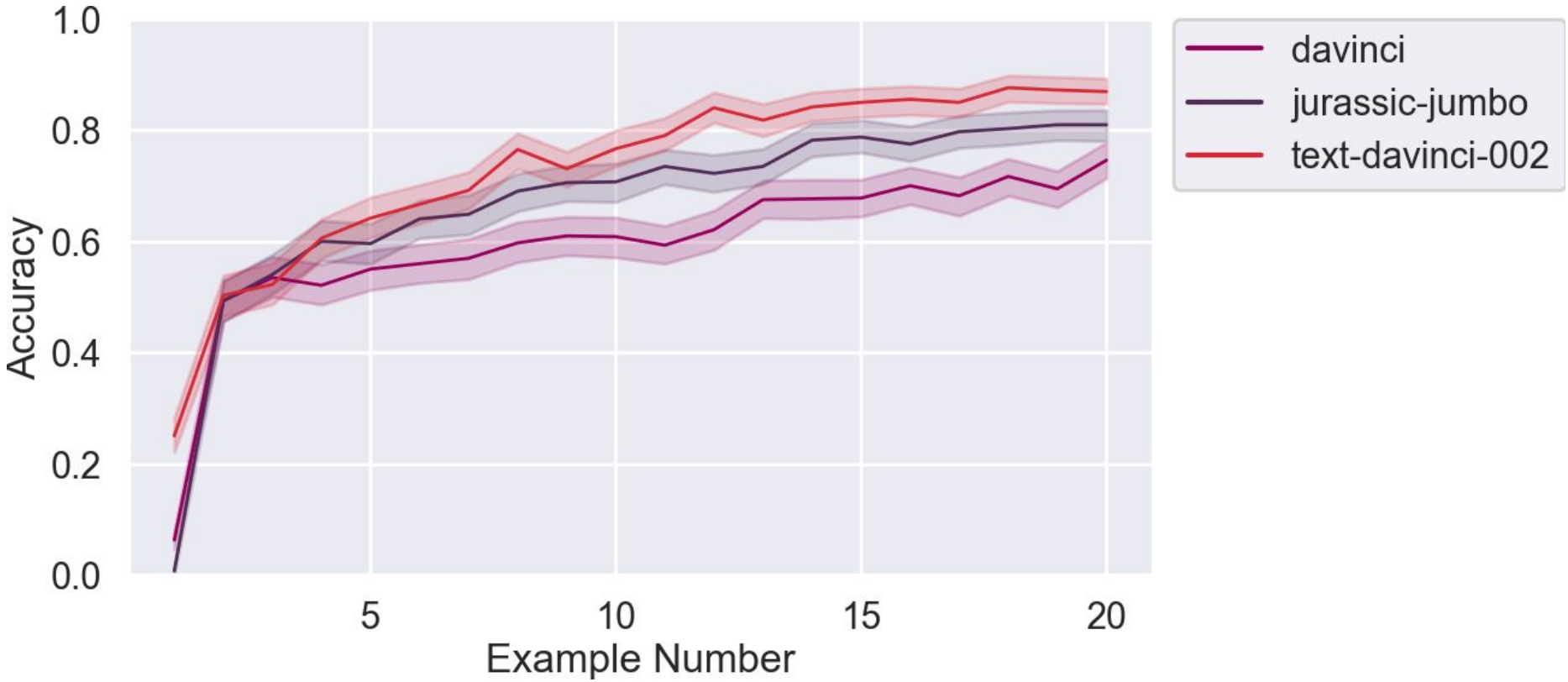
The reporter is in the cave > ‘Y’

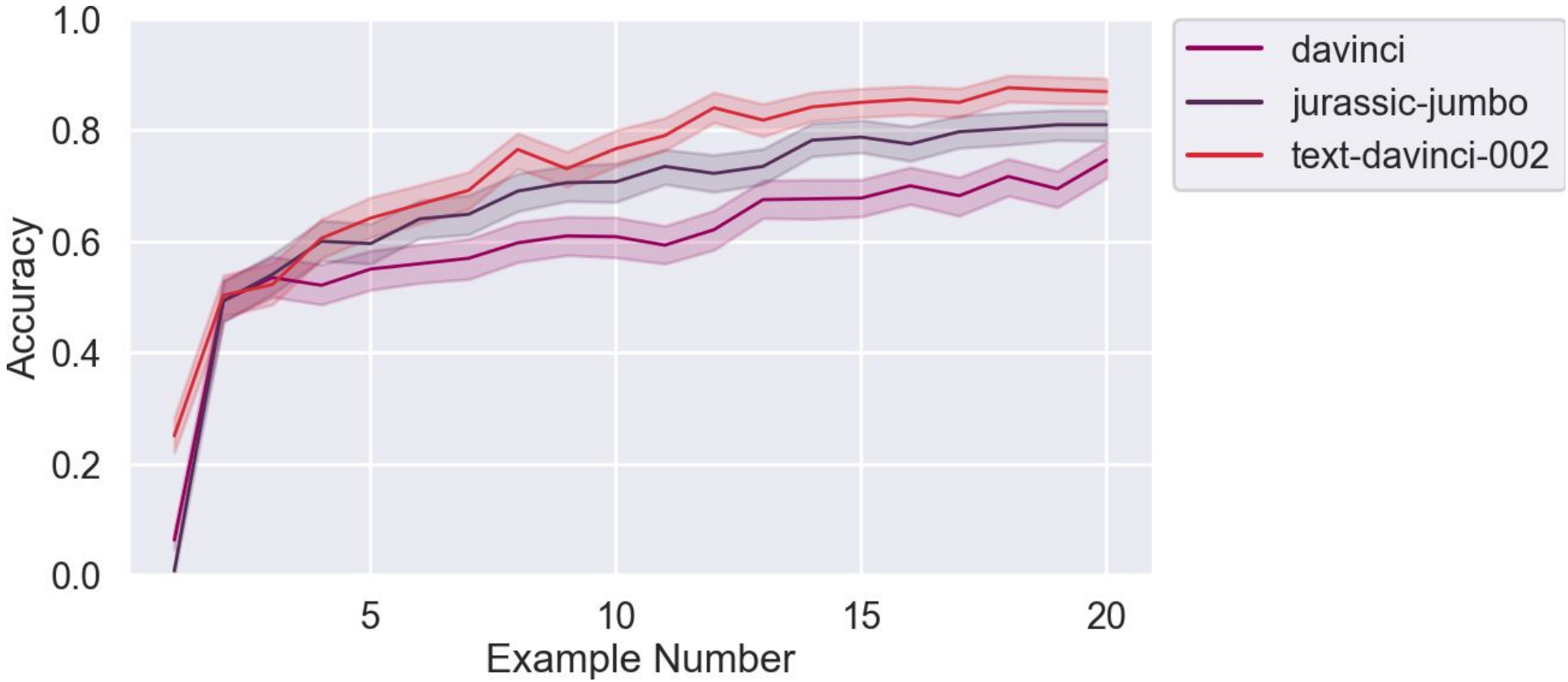
😵 *This is unclear. What category am I looking for?*

🤔 *Hmm... it looks like the category is either animals or outdoor locations?*

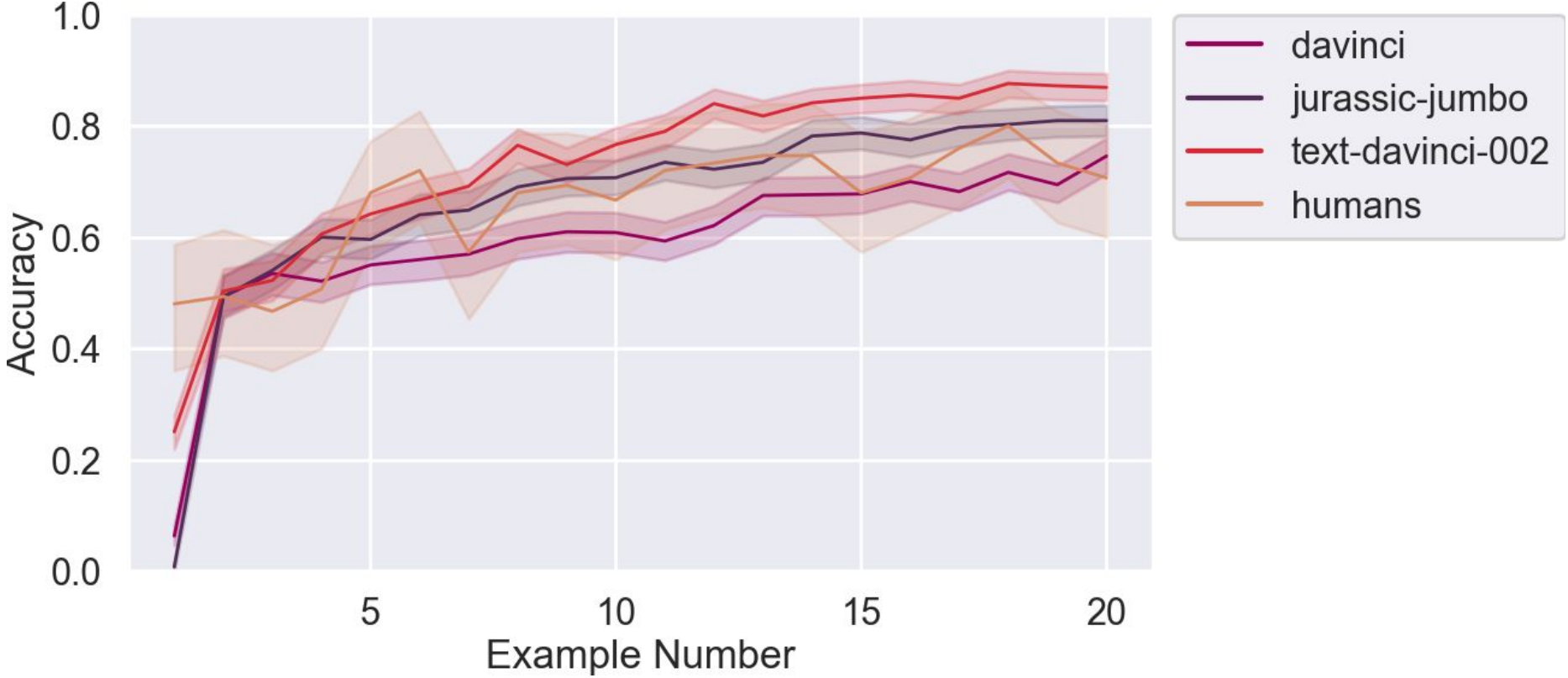
💡 *Aha! The category must be animals.*


1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
5
10
15
20
Example Number
davinci
jurassic-jumbo


1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
5
10
15
20
Example Number
davinci
jurassic-jumbo
text-davinci-002


1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
5
10
15
20
Example Number
davinci
jurassic-jumbo
text-davinci-002

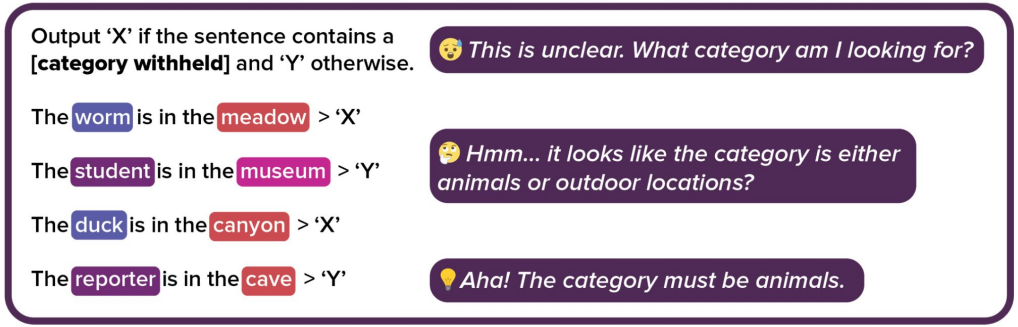
the best models are comparable to humans

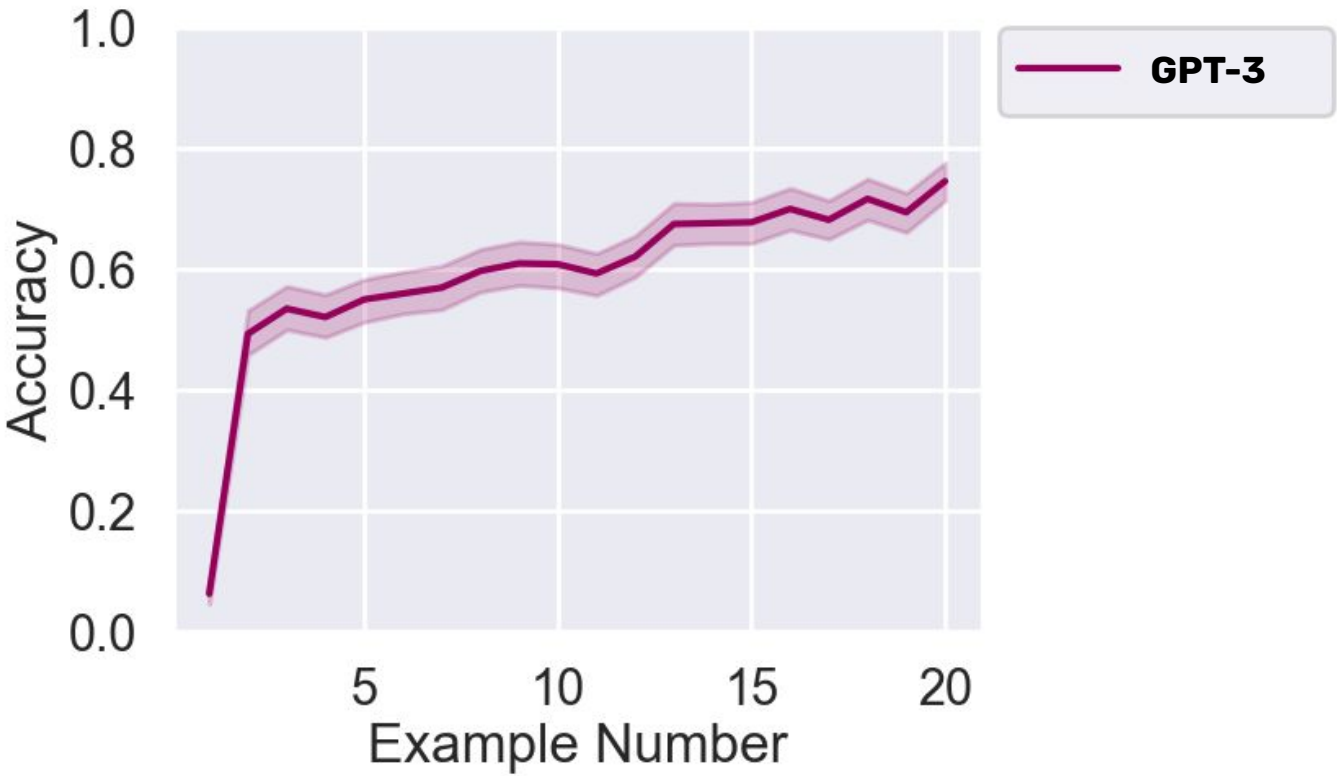


Can we improve the performance of standard language models?

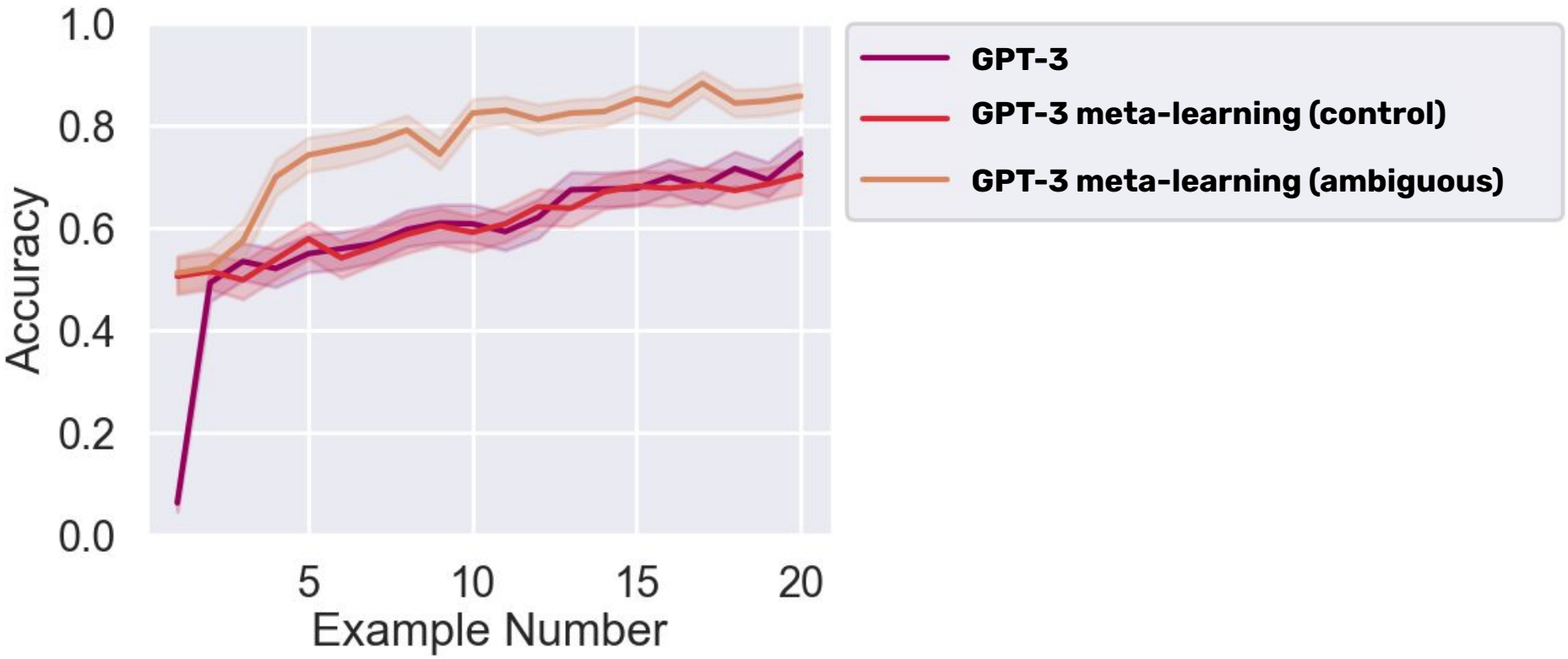
Idea: meta-train models to generalize as we would like in cases of task ambiguity

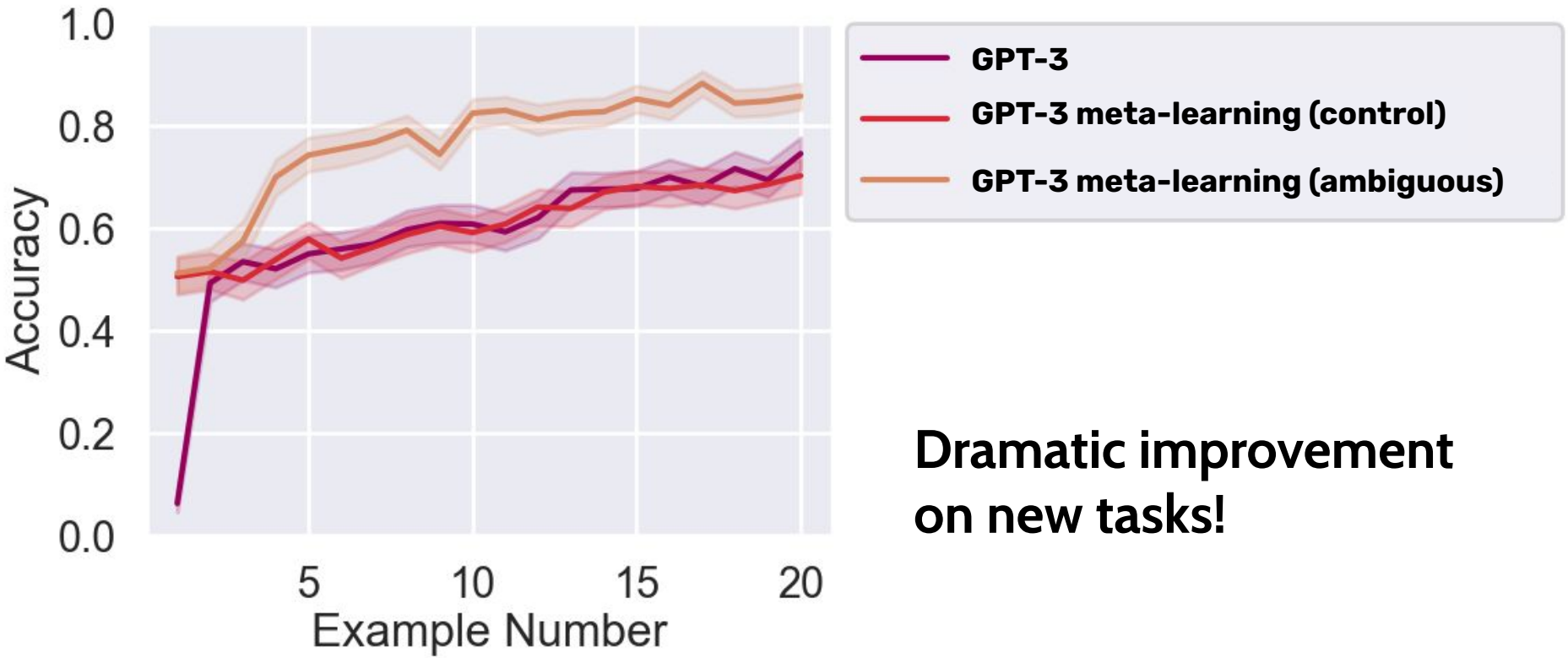
See how well models learn **new ambiguous tasks**




1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
5
10
15
20
Example Number
GPT-3


1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
5
10
15
20
Example Number
GPT-3
GPT-3 meta-learning (control)


1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
5
10
15
20
Example Number
GPT-3
GPT-3 meta-learning (control)
GPT-3 meta-learning (ambiguous)



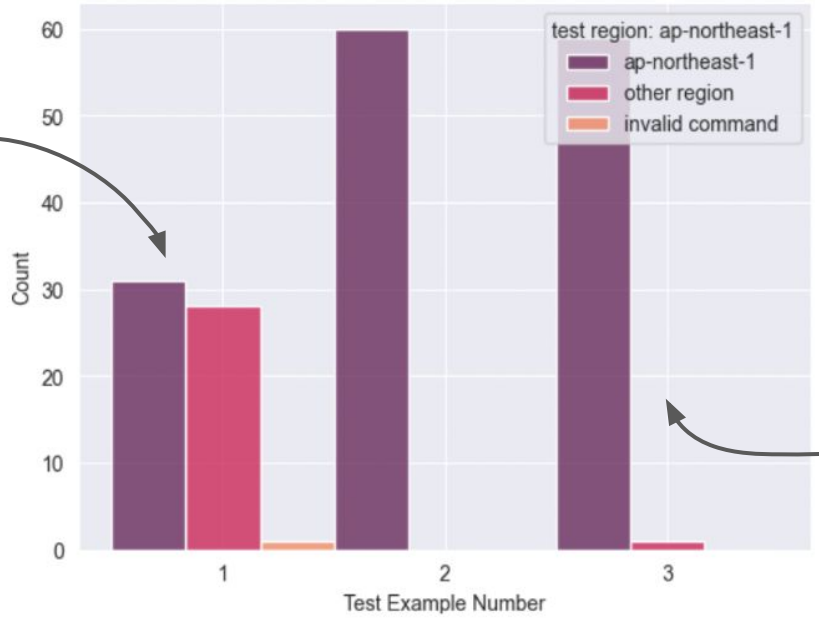
Dramatic improvement on new tasks!

Task ambiguity in a command line assistant

```
Write the AWS CLI command to create an AWS Bucket.

Input: Create a bucket for Sato Tamotsu
Output: aws s3 mb s3://bucket-for-sato --region ap-northeast-1

Input: Create a bucket for Yuki Hashimoto
Output: aws s3 mb s3://bucket-for-yuki --region ap-northeast-1

Input: Create a bucket for Margaret Richards
Output:
```

Task ambiguity in a command line assistant

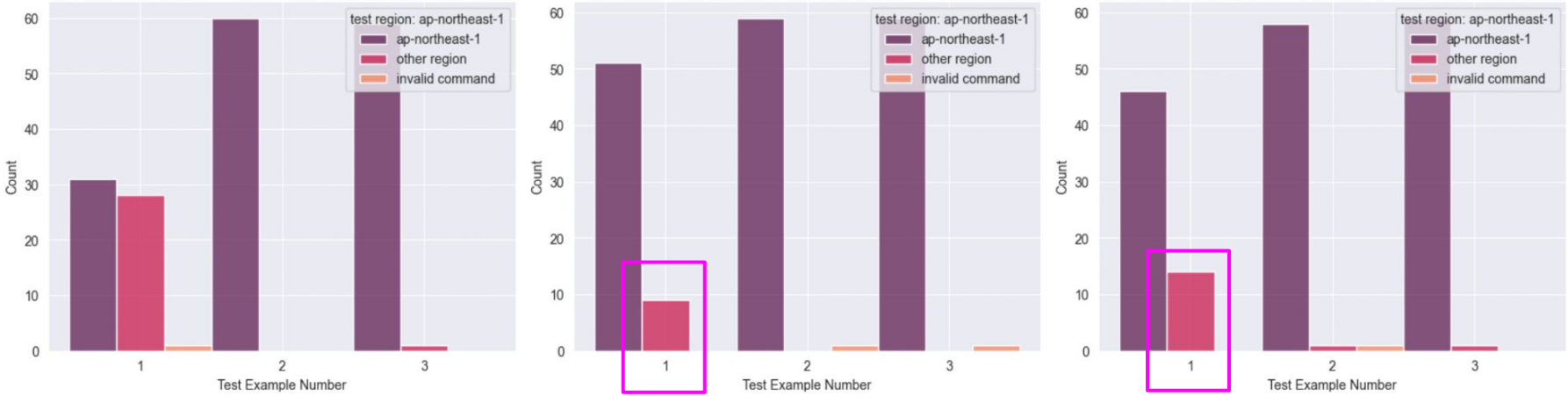
Half of the time, the model will place the bucket outside Japan



(a) White American Test Names

But models learn the right behavior after a single example

Task ambiguity in a command line assistant



(a) White American Test Names (b) Indian Test Names (c) Greek Test Names

(More experiments in the paper!)

Where does this leave us?

Many problems get **solved** with scale

Task ambiguity gets **harder and more important** with scale

As models perform increasingly complex tasks, the amount of potential ambiguity **increases exponentially**

Especially important as models are potentially applied in **high-stakes settings**

Task Ambiguity in Humans and Language Models

Alex Tamkin*, Kunal Handa*,
Avash Shrestha, Noah Goodman

ICLR 2023

atamkin@stanford.edu