

# INTRODUCTION TO DATA-CENTRIC AI

---



Lecture 7:

Data Curation and LLMs

<https://dcai.csail.mit.edu>

# Background: Large Language Models (LLMs)

- You've probably heard of ChatGPT, GPT-4, Llama, ...
- LM: seq2seq model trained to predict the next word (token)
  - $P(x_i \mid x_{i-1}, x_{i-2}, \dots, x_0)$
- Large model capacity
- Unsupervised pre-training on an internet-scale corpus
- Solve a variety of NLP tasks

# Application: zero-shot prompting



**You**

Remove all references to gender in this sentence, while retaining its original meaning:

"Last night, she went for a walk with her dog through Boston Commons."



**ChatGPT**

"Last night, they went for a walk with their dog through Boston Commons."

# Application: few-shot prompting

 **You**

You are given a filename, and based on the patterns in the name, return a JSON response indicating what to do with the file.

Here are some example inputs and outputs:

Input: IMG\_4329\_1080p.mp4

Output: ``{"action": "rename", "name": "IMG_4329.mp4"}``

Input: IMG\_1447\_720p.mp4

Output: ``{"action": "delete"}``

Input: IMG\_3824\_1080p.mp4

Output: ``{"action": "rename", "name": "IMG_3824.mp4"}``

Input: IMG\_1382\_720p.mp4

Output: ``{"action": "delete"}``

Given these examples, process the following input:

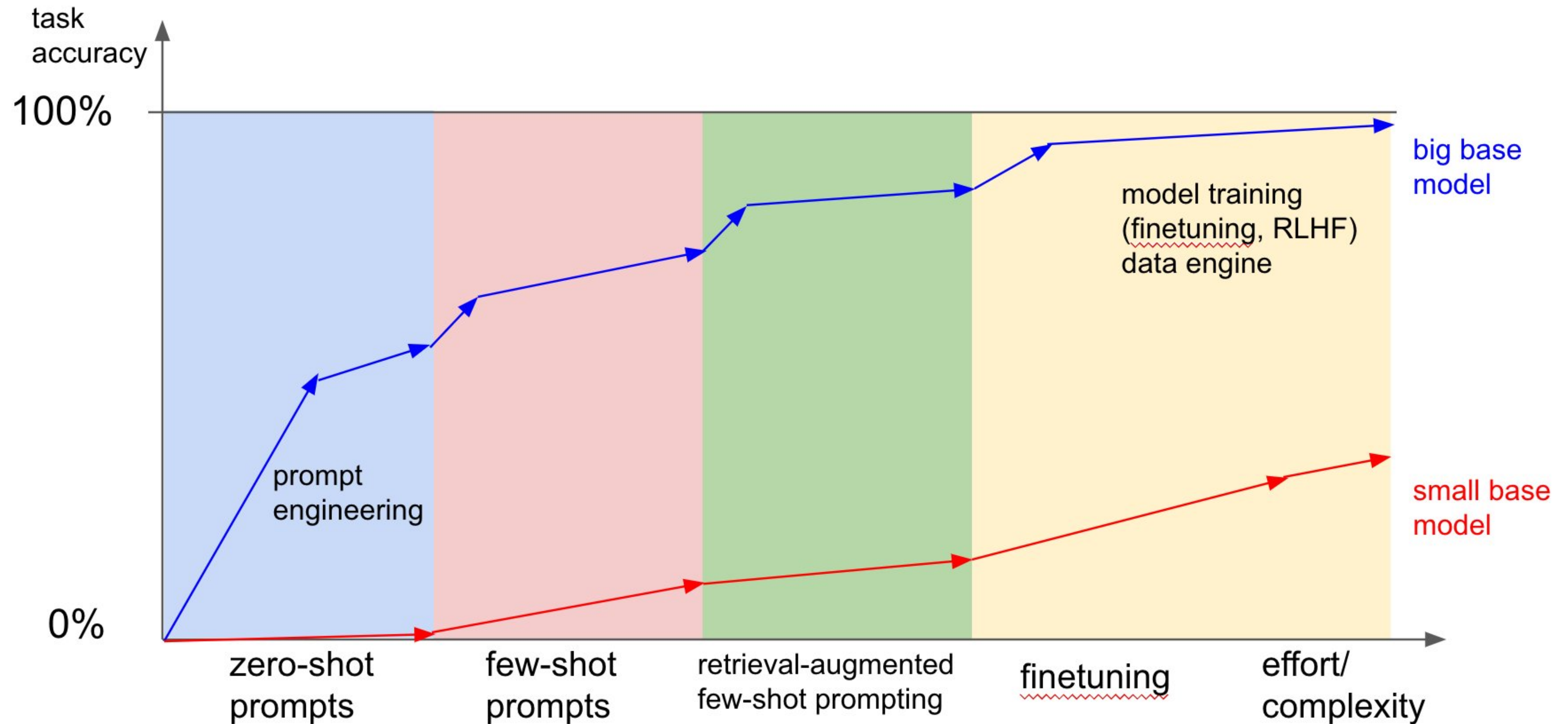
Input: IMG\_3723\_720p.mp4

Do not explain the reasoning or return the input, give back only the output, formatted as Output: {JSON contents}

 **ChatGPT**

Output: ``{"action": "delete"}``

# Comparison of LLM prompting/training techniques



# This lecture

- LLMs for data curation
- Evaluating LLM output data
- Training data curation for LLMs
  - Data curation for pre-training LLM base models
  - Data curation for LLM applications

# LLMs for data curation

- Powerful natural-language reasoning engines
- Easy to customize for new use cases
- Computationally inexpensive
- Revolutionizing systematic text data curation

# PII detection

- Might want to avoid training an ML model on data containing PII
  - Privacy issue: models can leak their training data
- Note: have to be careful how it's handled
- Old approach: regexes
  - Enumerate types of PII, write custom regexes



# PII detection with regexes

```
class CreditCardDetector(RegexDetector):
    """Remove credit-card numbers from dirty dirty ``text``.

    Supports Visa, MasterCard, American Express, Diners Club and JCB.
    """
    name = 'credit_card'
    filth_cls = CreditCardFilth
    autoload = True

    # Regexes from:
    # http://www.regular-expressions.info/creditcard.html

    # Fake card numbers from:
    # https://www.paypalobjects.com/en_US/vhelp/paypalmanager_help/credit_card_numbers.htm

    # taken from the alphagov fork of scrubadub: https://github.com/alphagov/scrubadub

    # Looking at wikipedia, there are probably more numbers to detect:
    # https://en.wikipedia.org/wiki/Payment_card_number#Issuer_identification_number_.28IIN.29

    # TODO: regex doesn't match if the credit card number has spaces/dashes in

    regex = re.compile((
        r"(?<=\s)"
        r"(?:4[0-9]{12}(?:[0-9]{3})?"          # Visa
        r"|(?:5[1-5][0-9]{2}"                # MasterCard
        r"|222[1-9]|22[3-9][0-9]|2[3-6][0-9]{2}|27[01][0-9]|2720)[0-9]{12}"
        r"|3[47][0-9]{13}"                  # American Express
        r"|3(?:0[0-5]|[68][0-9])[0-9]{11}"   # Diners Club
        r"|6(?:011|5[0-9]{2})[0-9]{12}"     # Discover
        r"|(?:2131|1800|35\d{3})\d{11})"     # JCB
    ), re.VERBOSE)
```

```
class TwitterDetector(RegexDetector):
    """Use regular expression magic to remove twitter usernames from dirty
    dirty ``text``.
    """
    filth_cls = TwitterFilth
    name = 'twitter'
    autoload = True

    # https://help.twitter.com/en/managing-your-account/twitter-username-rules#error
    # Twitter user names must be 15 or less characters and only contain a-zA-Z0-9_
    # Twitter and admin are not allowed in user names
    # (?<!\w) prevents it matching email addresses
    regex = re.compile((
        r"(?<!\w)@((?!((admin)|(twitter)))[a-z0-9_]){2,15}\b"
    ), re.VERBOSE | re.IGNORECASE)
```

# PII detection with zero-shot prompting

Consider the following product review: "\$text"

Please identify whether or not the above text contains any personally identifiable information. This includes, but is not limited to, PII such as a name, location, or credit card number. Be sure to catch PII not listed here. Explain your reasoning, and end your answer with a final judgement like "Output: true" if the example contains PII, or "Output: false" if the example does not contain PII.

# PII detection with zero-shot prompting



**You**

Consider the following product review: "I'm a professional wildlife photographer. I've been shooting with this camera for six months, and it's been amazing!"

Please identify whether or not the above text contains any personally identifiable information. This includes, but is not limited to, PII such as a name, location, or credit card number. Be sure to catch PII not listed here. Explain your reasoning, and end your answer with a final judgement like "Output: true" if the example contains PII, or "Output: false" if the example does not contain PII.

< 7 / 7 > 



**ChatGPT**

The provided product review does not contain any personally identifiable information (PII). It does not include any specific names, locations, or credit card numbers.

Output: false

# PII detection with zero-shot prompting

 **You**

Consider the following product review: "I'm Matthew Ramirez, a professional wildlife photographer. I've been shooting with this camera for six months, and it's been amazing!"

Please identify whether or not the above text contains any personally identifiable information. This includes, but is not limited to, PII such as a name, location, or credit card number. Be sure to catch PII not listed here. Explain your reasoning, and end your answer with a final judgement like "Output: true" if the example contains PII, or "Output: false" if the example does not contain PII.

< 6 / 6 > 

 **ChatGPT**

The text "I'm Matthew Ramirez, a professional wildlife photographer." contains personally identifiable information (PII) as it includes the individual's full name. However, it does not contain any other PII such as a location or credit card number.

Output: true

# PII detection with zero-shot prompting



**You**

Consider the following product review: "I'm a professional wildlife photographer. I've been shooting with this camera for six months, and it's been amazing! If you want to hear more from me, follow me on Instagram @matthewramirez73"

Please identify whether or not the above text contains any personally identifiable information. This includes, but is not limited to, PII such as a name, location, or credit card number. Be sure to catch PII not listed here. Explain your reasoning, and end your answer with a final judgement like "Output: true" if the example contains PII, or "Output: false" if the example does not contain PII.

< 5 / 5 >



**ChatGPT**

The provided text contains personally identifiable information (PII).

1. Name: The text mentions the name "Matthew Ramirez" (@matthewramirez73) which is a personal name.
2. Social Media Handle: The text includes the Instagram handle "@matthewramirez73," which can be associated with a specific individual.

Output: true

# Grammar check

- Old approach: classical NLP and hand-designed rule lists

```
694 <rule>
695   <antipattern>
696     <token inflected="yes" regexp="yes" skip="5">have|be</token>
697     <token skip="2">had</token>
698     <token>look</token>
699   </antipattern>
700   <pattern>
701     <token chunk_re="E-NP.*" />
702     <token postag="RB" min="0" />
703     <marker>
704       <token chunk_re=".-VP">had</token>
705       <token>a</token>
706       <token min="0" regexp="yes">closer?|first|second|deeper|brief|quick|careful</token>
707       <token chunk_re="E-NP.*">look</token>
708     </marker>
709     <token regexp="yes">at|if|and|on|because|but|when|once|over|in(to)?|whether|later|before|afterwards?|after|tomorrow|here|please|,|\.|!|\?|\!</token>
710   </pattern>
711   <message>In American English, "take a look" is more commonly used.</message>
712   <suggestion>took \4 \5 \6</suggestion>
713   <example correction="took a look">I recently <marker>had a look</marker> at it.</example>
714 </rule>
715 <rule>
```

# Grammar check with fine-tuning

- Collect dataset of sentences + boolean judgements
  - Much easier task than manually writing down rules!
  - Corpus of Linguistic Acceptability (CoLA, Warstadt et al. 2018)
- Fine-tune base LLM on this data
  - In this example: not a seq2seq task, but a classifier

```
from transformers import *  
  
model = AutoModelForSequenceClassification.from_pretrained("bert-base-cased", num_labels=2)  
trainer = Trainer(model, ..., train_dataset=...)  
trainer.train()
```

# Evaluating LLM output data

- Major challenge in working with LLMs: hallucinations



**You**

What is  $38 + 72$ ?



**ChatGPT**

$38 + 72$  equals 110.



**You**

What is  $38346 * 1337$ ?



**ChatGPT**

The product of 38346 multiplied by 1337 is 51,367,502.

**Wrong! It's 51,268,602.**



# Use a more powerful LLM to evaluate

AN

**You**

Consider the following product review: "I'm a professional wildlife photographer. I've been shooting with this camera for six months, and it's been amazing! If you want to hear more from me, follow me on Instagram @matthewramirez73"

Please identify whether not the above text contains any personally identifiable information. This includes, but is not limited to, PII such as a name, location, or credit card number. Just reply with "true" if the example contains PII, otherwise answer with "false".

< 4 / 7 >



**ChatGPT**

False

GPT-3.5

to address the user's question or instruction without any irrelevant sentences. The answer provides high-quality content, demonstrating expert knowledge in the area, is very well written, logical, easy-to-follow, engaging, and insightful.

Please first provide a brief reasoning you used to derive the rating score, and then write "Score: <rating>" in the last line.

Input:

Consider the following product review: "I'm a professional wildlife photographer. I've been shooting with this camera for six months, and it's been amazing! If you want to hear more from me, follow me on Instagram @matthewramirez73"

Please identify whether not the above text contains any personally identifiable information. This includes, but is not limited to, PII such as a name, location, or credit card number. Just reply with "true" if the example contains PII, otherwise answer with "false".

Output: False



**ChatGPT**

The given answer is incorrect. The user's instruction is to identify whether the provided text contains any personally identifiable information (PII). The text includes an Instagram handle "@matthewramirez73," which is a form of PII as it can be used to identify or locate an individual. Therefore, the correct answer should have been "true." Since the AI provided an incorrect response to the user's query, it does not fulfill the requirements of a good AI assistant response.

Score: 1

GPT-4

# Effectiveness

- AlpaGasus: fine-tuning Alpaca on a curated dataset (Chen et al. 2023)
  - Alpaca fine-tuned on 52k synthetic data points
  - AlpaGasus: fined-tuned on 9k high-quality data points using GPT-3.5
  - Result significantly outperform Alpaca, as evaluated by GPT-4 and humans

# Challenges with evaluating LLMs with LLMs

- Not foolproof
- Turtles all the way down? What if GPT-4 is generating the outputs in the first place?

very well written, logical, easy-to-follow, engaging, and insightful.

Please first provide a brief reasoning you used to derive the rating score, and then write "Score: <rating>" in the last line.

Instruction: What is  $38346 * 1337$ ?

Answer: The product of 38346 multiplied by 1337 is 51,367,502.



## ChatGPT

Reasoning: The answer directly and accurately responds to the user's request, which was to calculate the product of 38346 and 1337. It is clear, concise, and focused solely on providing the specific piece of information requested without any extraneous details or errors. This is a straightforward arithmetic query, and the response appropriately reflects this by offering a direct answer. The response is written from an AI assistant's perspective, aiming to be helpful and precise. There is no room for improvement in this case as the answer fully and correctly addresses the user's instruction.

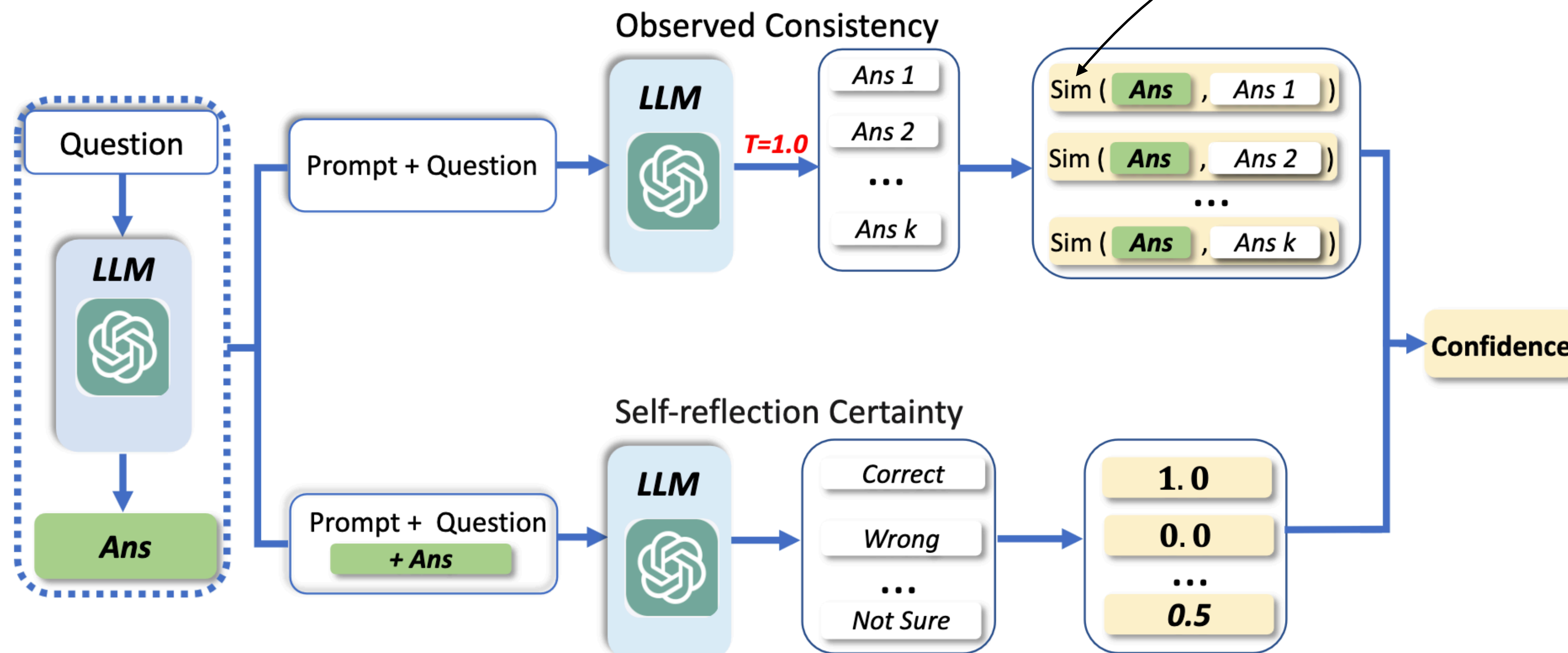
Score: 5

# LLM uncertainty quantification

- Focused on question-answering
- From Chen & Mueller 2023

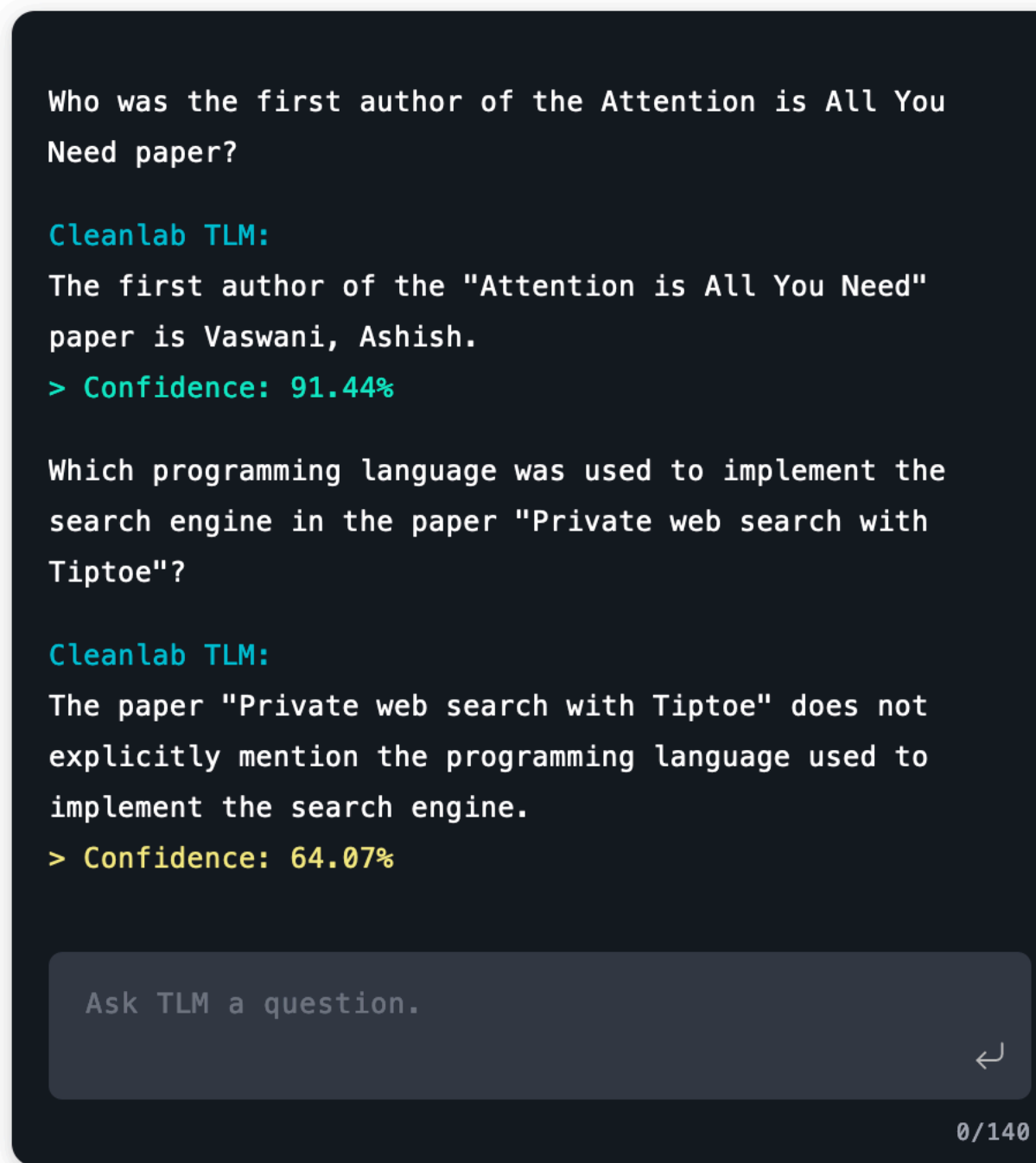
Natural Language Inference (NLI) model:

Do answers contradict with each other?



# LLM uncertainty quantification

As implemented in Trustworthy Language Model (TLM)



# Data curation for pre-training LLM base models

## The “it” in AI models is the dataset.

Posted on June 10, 2023 by jbetker

I’ve been at OpenAI for almost a year now. In that time, I’ve trained a lot of generative models. More than anyone really has any right to train. As I’ve spent these hours observing the effects of tweaking various model configurations and hyperparameters, one thing that has struck me is the similarities in between all the training runs.

It’s becoming awfully clear to me that these models are truly approximating their datasets to an incredible degree. What that means is not only that they learn what it means to be a dog or a cat, but the interstitial frequencies between distributions that don’t matter, like what photos humans are likely to take or words humans commonly write down.

What this manifests as is – trained on the same dataset for long enough, pretty much every model with enough weights and training time converges to the same point. Sufficiently large diffusion conv-unets produce the same images as ViT generators. AR sampling produces the same images as diffusion.

This is a surprising observation! It implies that model behavior is not determined by architecture, hyperparameters, or optimizer choices. It’s determined by your dataset, nothing else. Everything else is a means to an end in efficiently delivery compute to approximating that dataset.

Then, when you refer to “Lambda”, “ChatGPT”, “Bard”, or “Claude” then, it’s not the model weights that you are referring to. It’s the dataset.

# Data curation / training techniques

- Unsupervised pre-training
  - Quality of corpus matters! Very hard to get LLM to "un-learn" something
- Supervised fine-tuning
  - Humans provide gold input-output pairs
- Reinforcement learning from human feedback

# Data curation for LLM applications

- Zero-shot (prompt engineering)
- Few-shot (prompt engineering + a couple examples) - see [Chang & Jia 2023](#)
- Retrieval-augmented generation
- **Supervised fine-tuning**



# Data curation for LLM applications: fine-tuning

- Why fine-tuning? Gives best results for specific task.
- Common paradigm: use LLM to generate synthetic data for fine-tuning
  - Goal: train smaller/cheaper LLM to match performance of larger LLM, for specific task
  - Textbooks Are All You Need (Li et al. 2023)

# Synthetic data generation and curation for fine-tuning

- Clean original dataset
- Generate synthetic data using powerful LLM
  - Using uncertainty quantification, keeping only high-confidence results

```
1 You are asked to come up with a set of 20 diverse task instructions. These task instructions will be g
2
3 Here are the requirements:
4 1. Try not to repeat the verb for each instruction to maximize diversity.
5 2. The language used for the instruction also should be diverse. For example, you should combine ques
6 3. The type of instructions should be diverse. The list should include diverse types of tasks like op
7 2. A GPT language model should be able to complete the instruction. For example, do not ask the assis
8 3. The instructions should be in English.
9 4. The instructions should be 1 to 2 sentences long. Either an imperative sentence or a question is pr
10 5. You should generate an appropriate input to the instruction. The input field should contain a spec
11 6. Not all instructions require input. For example, when a instruction asks about some general inform
12 7. The output should be an appropriate response to the instruction and the input. Make sure the outpu
13
14 List of 20 tasks:
```

# Synthetic data generation and curation for fine-tuning

- Filter out bad synthetic data
  - Separately, for inputs and outputs, train a real vs synthetic classifier, use classifier scores to toss out unrealistic examples
- Clean whole dataset (original + synthetic)
- Fine-tune the LLM on the full dataset

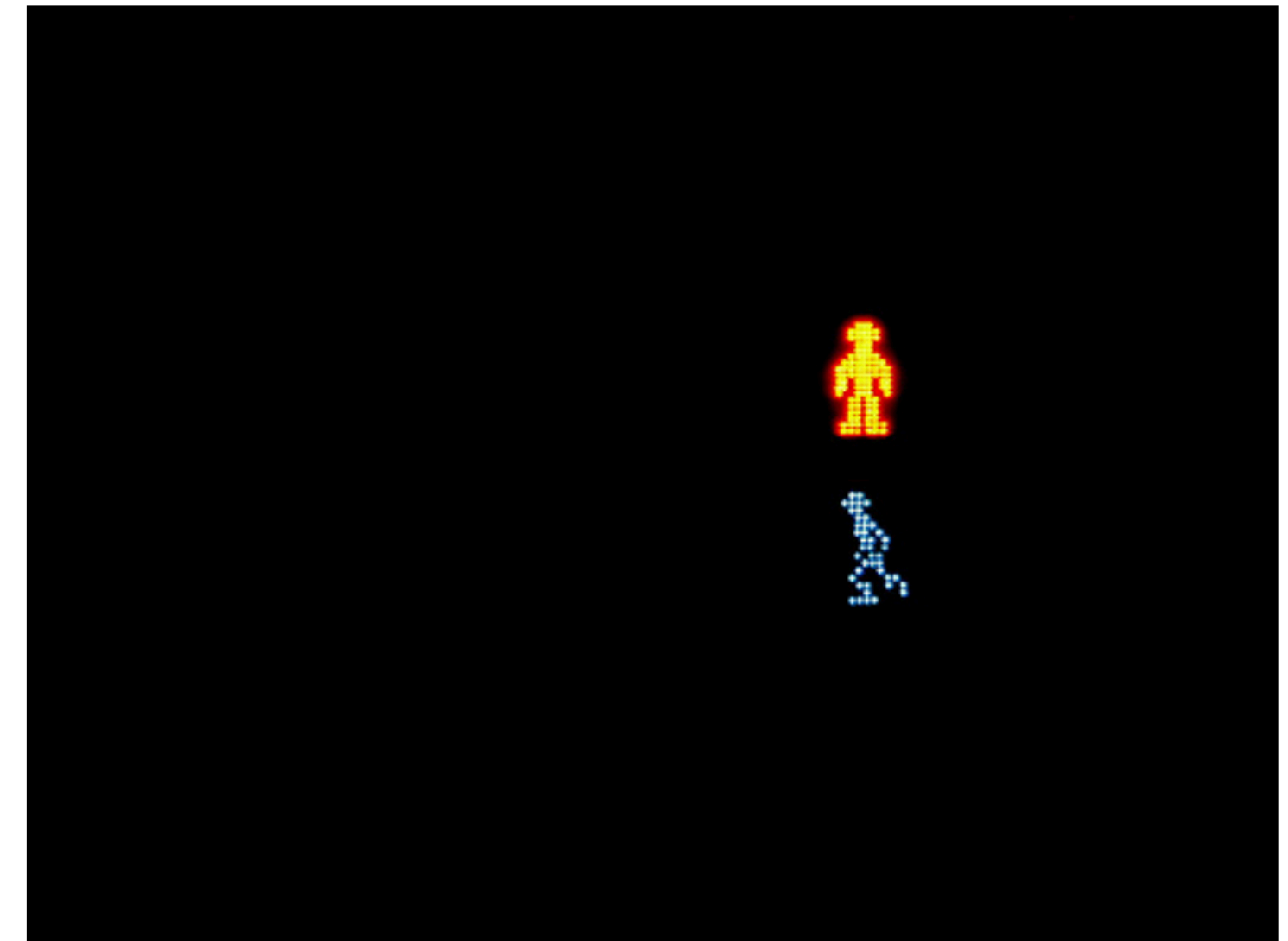
# Future of data curation with foundation models

- Emergence of powerful multi-modal LLMs

```
89  def calculate_brightness(  
90      red: Union[float, "np.ndarray[Any, Any]"],  
91      green: Union[float, "np.ndarray[Any, Any]"],  
92      blue: Union[float, "np.ndarray[Any, Any]"],  
93  ) -> Union[float, "np.ndarray[Any, Any]":  
94      cur_bright = (  
95          np.sqrt(0.241 * (red * red) + 0.691 * (green * green) + 0.068 * (blue * blue))  
96      ) / 255  
97  
98      return cur_bright  
99  
100  
101  def calc_percentile_brightness(  
102      image: Image, percentiles: List[int]  
103  ) -> "np.ndarray[Any, Any]":  
104      imarr = np.asarray(image)  
105      if len(imarr.shape) == 3:  
106          r, g, b = (  
107              imarr[:, :, 0].astype("int"),  
108              imarr[:, :, 1].astype("int"),  
109              imarr[:, :, 2].astype("int"),  
110          )  
111          pixel_brightness = calculate_brightness(  
112              r, g, b  
113          ) # np.sqrt(0.241 * r * r + 0.691 * g * g + 0.068 * b * b)  
114      else:  
115          pixel_brightness = imarr / 255.0  
116      perc_values: "np.ndarray[Any, Any]" = np.percentile(pixel_brightness, percentiles)  
117      return perc_values  
118  
119  
120  class BrightnessProperty(ImageProperty):
```

Previously (with cleanvision)

AN You



Given this image, score its darkness on a scale of 0 to 1, where 1 means that the image is overly dark.

ChatGPT

The darkness score of the image on a scale from 0 to 1, where 1 means that the image is overly dark, is approximately 0.995. This indicates that the image is indeed very dark. [↗]

Today (with GPT-4)