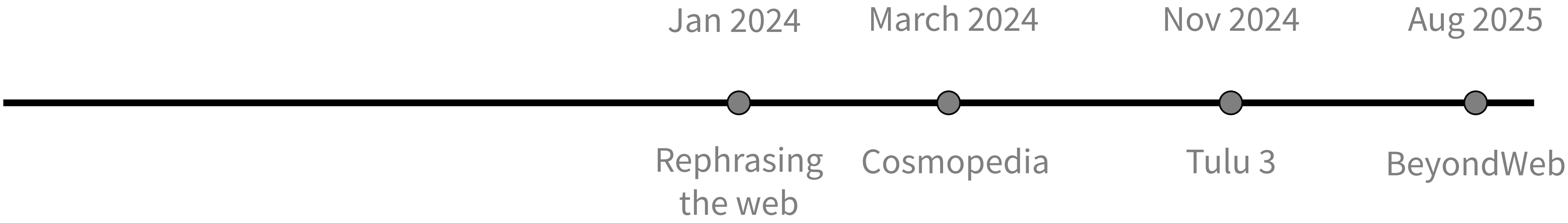
Synthetic Data & Distillation

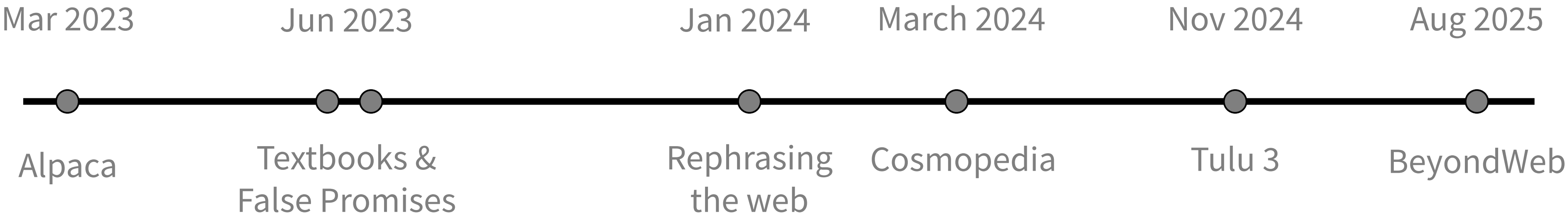
Alpaca: A Strong, Replicable Instruction-Following Model
Textbooks Are All You Need
The False Promise of Imitating Proprietary LLMs

Main Presenters: Charles Xu, Ryan Wang,
Critics: Shangyin Tan, Bhavya Chopra
Proponents: Dennis Jacob, Sidhika Balachandar
Follow-Up: Huanzhi Mao
09/16

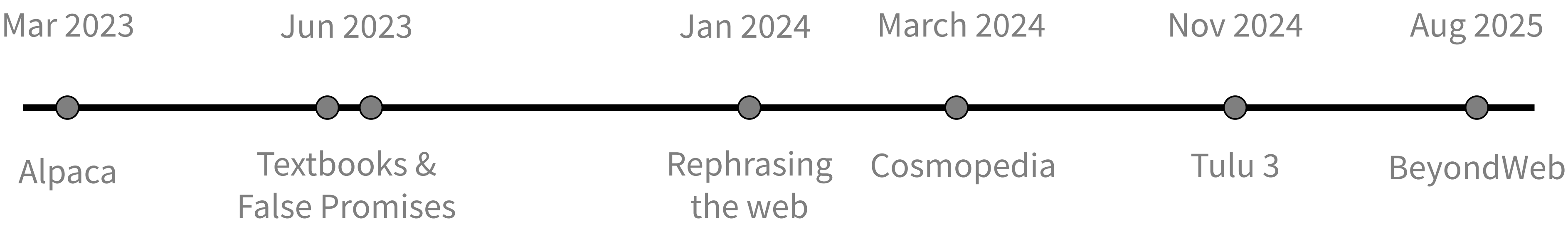
Wait… Synthetic Data Again?



Wait… Synthetic Data Again?



Wait... Synthetic Data Again?



Why synthetic data?

- Human data is hard to collect, limited, low-quality, and sometimes closed-sourced

Wait... distillation?

Setting: Student (weak) and Teacher (strong)

Goal: match student prob. distribution to teacher prob. Distribution

Method:

- minimizing cross entropy between student and teacher soft logit distribution
- training student on generated token outputs of teacher model (= synthetic data)

Wait... distillation?

Setting: Student (weak) and Teacher (strong)

Goal: match student prob. distribution to teacher prob. Distribution

Method:

- minimizing cross entropy between student and teacher soft logit distribution
- training student on generated token outputs of teacher model (= synthetic data)

For today: distillation $\Leftrightarrow$ student model training on synthetic data from teacher model

Wait… distillation?

Setting: Student (weak) and Teacher (strong)

Goal: match student prob. distribution to teacher prob. Distribution

Method:

- minimizing cross entropy between student and teacher soft logit distribution
- training student on generated token outputs of teacher model (= synthetic data)

For today: distillation $\Longleftrightarrow$ student model training on synthetic data from teacher model

Question: Is distillation/synthetic data an effective way to train models?

This Presentation

Is synthetic data an effective way to train models?

- Alpaca: A Strong, Replicable Instruction-Following Model
  - IT, use teacher model to generate not only outputs, but also prompts/input
- Textbooks are All You Need
  - Uses real data to seed generation
- The False Promise of Imitating Proprietary LLMs
  - Shallow alignment

This Presentation

Is synthetic data an effective way to train models?

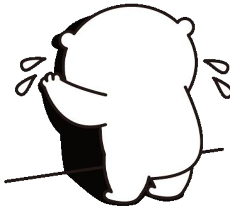
- **Alpaca: A Strong, Replicable Instruction-Following Model**
  - IT, use teacher model to generate not only outputs, but also prompts/input
- Textbooks are All You Need
  - Uses real data to seed generation
- The False Promise of Imitating Proprietary LLMs
  - Shallow alignment

Back in the old days...

Everyone: We need instruction following models!

OpenAI: Yay we have instruction following models! (InstructGPT, ChatGPT)

Academia / open-source: 😑



Back in the old days...

Everyone: We need instruction following models!

OpenAI: Yay we have instruction following models! (InstructGPT, ChatGPT)

Academia / open-source: 😑

- LLaMA, GPT-NeoX, BLOOM not aligned to follow instructions (lacked usability)
- RLHF is hard and expensive
  - Human data is hard/expensive
  - Data is closed-sourced
- IT datasets (Super-NaturalInstructions) doesn’t scale well (human annotations) and limited (topic selection)

Back in the old days...

Everyone: We need instruction following models!

OpenAI: Yay we have instruction following models! (InstructGPT, ChatGPT)

Academia / open-source: 😑

- LLaMA, GPT-NeoX, BLOOM not aligned to follow instructions (lacked usability)
- RLHF is hard and expensive
    - Human data is hard/expensive
    - Data is closed-sourced
- IT datasets (Super-NaturalInstructions) doesn’t scale well (human annotations) and limited (topic selection)

Need: scalable & cheap method for IT

Back in the old days...

Everyone: We need instruction following models!

OpenAI: Yay we have instruction following models! (InstructGPT, ChatGPT)

Academia / open-source: 😑

- LLaMA, GPT-NeoX, BLOOM not aligned to follow instructions (lacked usability)
- RLHF is hard and expensive
    - Human data is hard/expensive
    - Data is closed-sourced
- IT datasets (Super-NaturalInstructions) doesn’t scale well (human annotations) and limited (topic selection)

Need: scalable & cheap method for IT

Back in the old days...

Everyone: We need instruction following models!

OpenAI: Yay we have instruction following models! (InstructGPT, ChatGPT)

Academia / open-source: 😑

- LLaMA, GPT-NeoX, B … sability)
- RLHF is hard and ex…
  - Human data is …
  - Data is closed-…
- IT datasets (Super-NaturalInstructions) doesn't scale well (human annotations) and limited (topic selection)

Q: How can we generate synthetic data to train LLaMA to instruction-follow?

Need: scalable & cheap method for IT

Brainstorming session

Setting: We have a good teacher model (OpenAI text-davinci-003) and a student model (LLaMA)

Question: How can we generate synthetic data from text-davinci-003 to train LLaMA to instruction-follow?

Brainstorming session

Setting: We have a good teacher model (OpenAI text-davinci-003) and a student model (LLaMA)

Question: How can we generate synthetic data from text-davinci-003 to train LLaMA to instruction-follow?

We need:

- Instructions
- Outputs

Brainstorming session

Setting: We have a good teacher model (OpenAI text-davinci-003) and a student model (LLaMA)

Question: How can we generate synthetic data from text-davinci-003 to train LLaMA to instruction-follow?

We need:

- Instructions
- Outputs -> easy -> prompt text-davinci-003 with instructions

Brainstorming session

Setting: We have a good teacher model (OpenAI text-davinci-003) and a student model (LLaMA)

Question: How can we generate synthetic data from text-davinci-003 to train LLaMA to instruction-follow?

We need:

- Instructions
- Outputs -> easy -> prompt text-davinci-003 with instructions

Harder question: How can we generate good instructions?

Getting new instructions

Seed set of instructions

Getting new instructions

Pool of instructions

Seed set of instructions

Getting new instructions

Pool of instructions

Seed set of instructions

Prompt model to generate new instructions

Getting new instructions

Pool of instructions

Seed set of instructions

Heuristic Filtering

Prompt model to generate new instructions

Self-instruct Pipeline

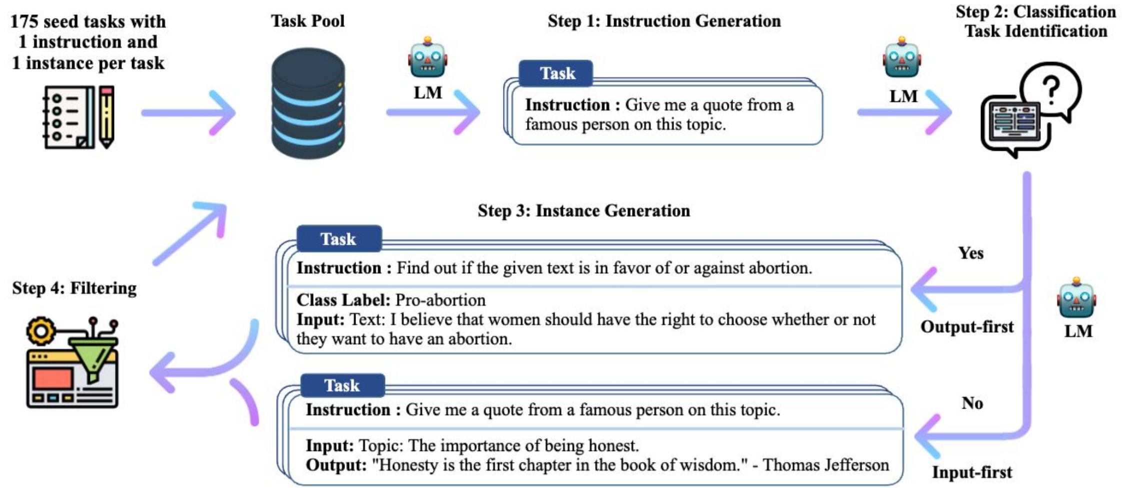


Figure 2: A high-level overview of SELF-INSTRUCT. The process starts with a small seed set of tasks as the task pool. Random tasks are sampled from the task pool, and used to prompt an off-the-shelf LM to generate both new instructions and corresponding instances, followed by filtering low-quality or similar generations, and then added back to the initial repository of tasks. The resulting data can be used for the instruction tuning of the language model itself later to follow instructions better. Tasks shown in the figure are generated by GPT3.

https://arxiv.org/abs/2212.10560

Self-instruct Pipeline

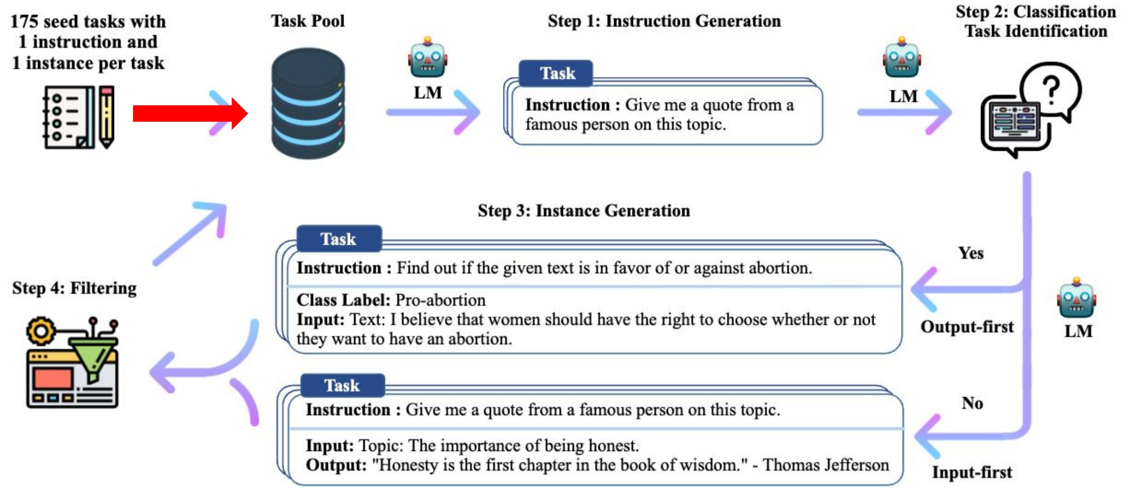


Figure 2: A high-level overview of SELF-INSTRUCT. The process starts with a small seed set of tasks as the task pool. Random tasks are sampled from the task pool, and used to prompt an off-the-shelf LM to generate both new instructions and corresponding instances, followed by filtering low-quality or similar generations, and then added back to the initial repository of tasks. The resulting data can be used for the instruction tuning of the language model itself later to follow instructions better. Tasks shown in the figure are generated by GPT3.

https://arxiv.org/abs/2212.10560

Self-instruct Pipeline

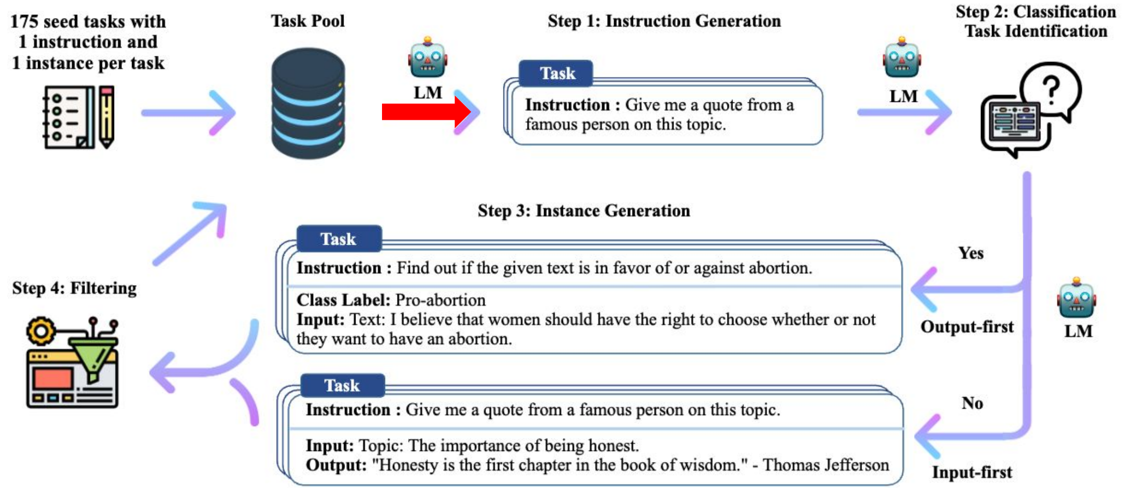


Figure 2: A high-level overview of SELF-INSTRUCT. The process starts with a small seed set of tasks as the task pool. Random tasks are sampled from the task pool, and used to prompt an off-the-shelf LM to generate both new instructions and corresponding instances, followed by filtering low-quality or similar generations, and then added back to the initial repository of tasks. The resulting data can be used for the instruction tuning of the language model itself later to follow instructions better. Tasks shown in the figure are generated by GPT3.

https://arxiv.org/abs/2212.10560

Self-instruct Pipeline

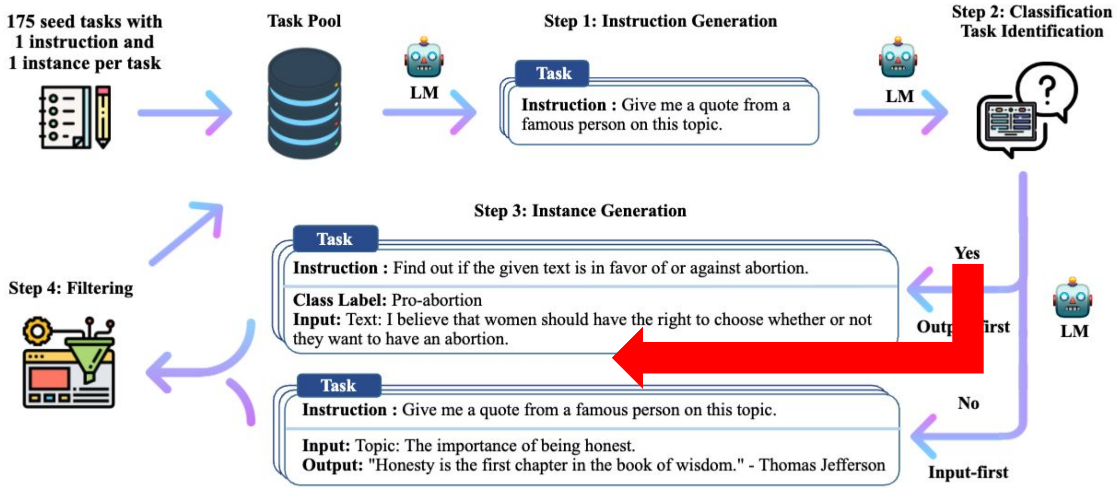


Figure 2: A high-level overview of SELF-INSTRUCT. The process starts with a small seed set of tasks as the task pool. Random tasks are sampled from the task pool, and used to prompt an off-the-shelf LM to generate both new instructions and corresponding instances, followed by filtering low-quality or similar generations, and then added back to the initial repository of tasks. The resulting data can be used for the instruction tuning of the language model itself later to follow instructions better. Tasks shown in the figure are generated by GPT3.

Self-instruct Pipeline

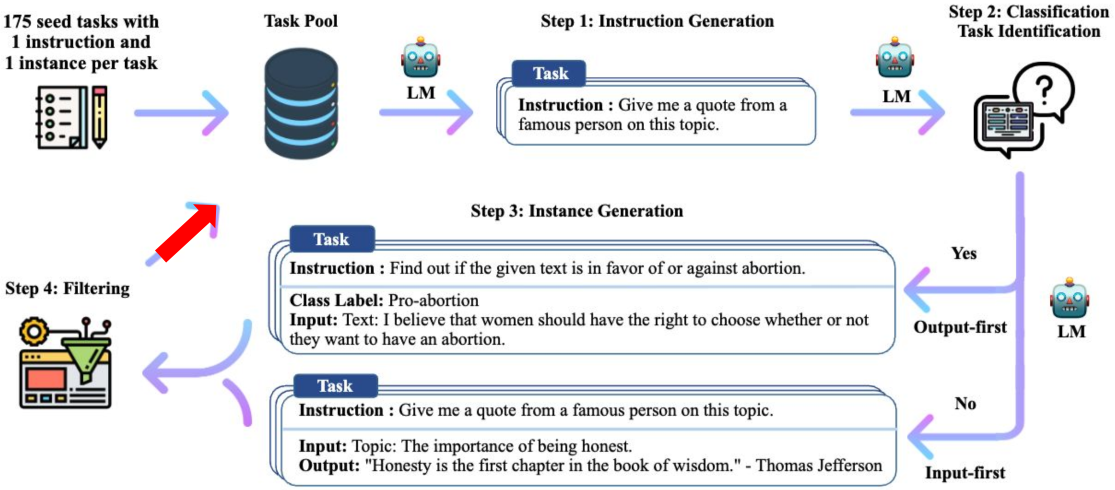
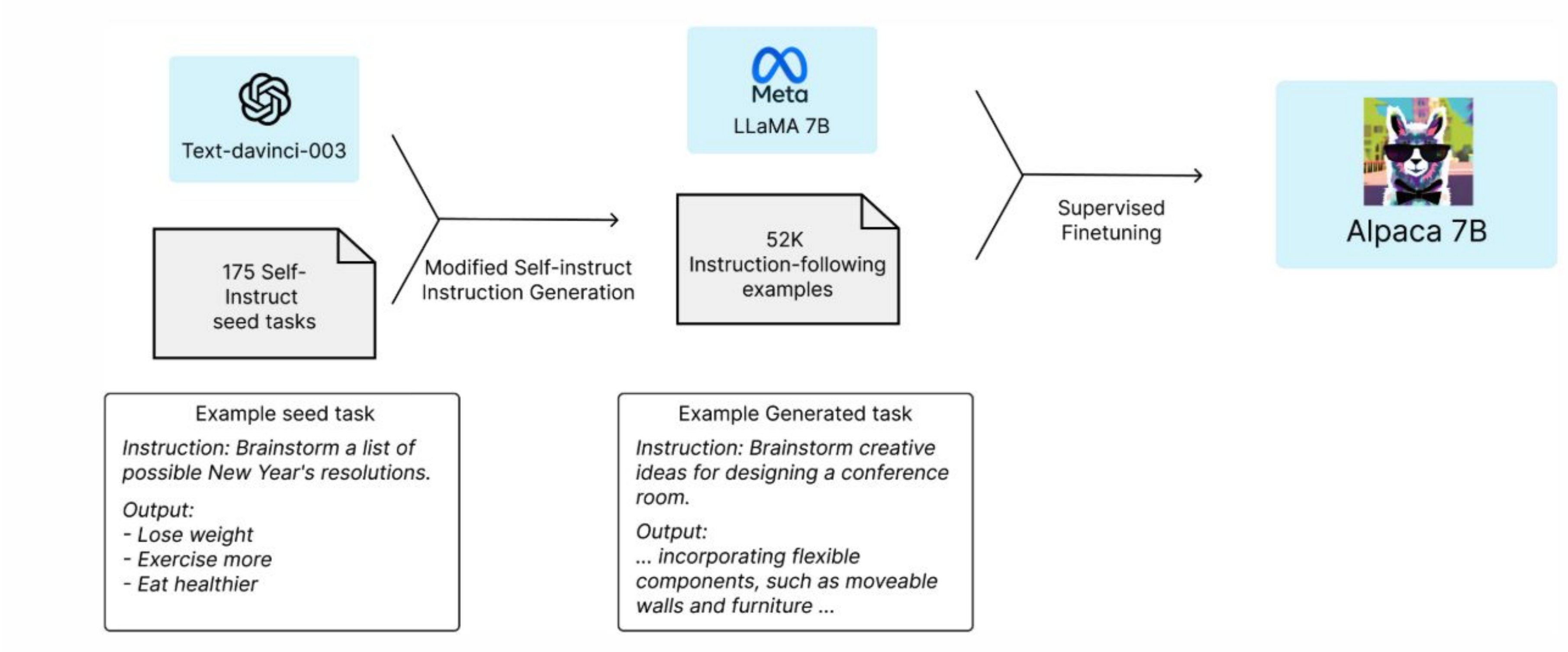


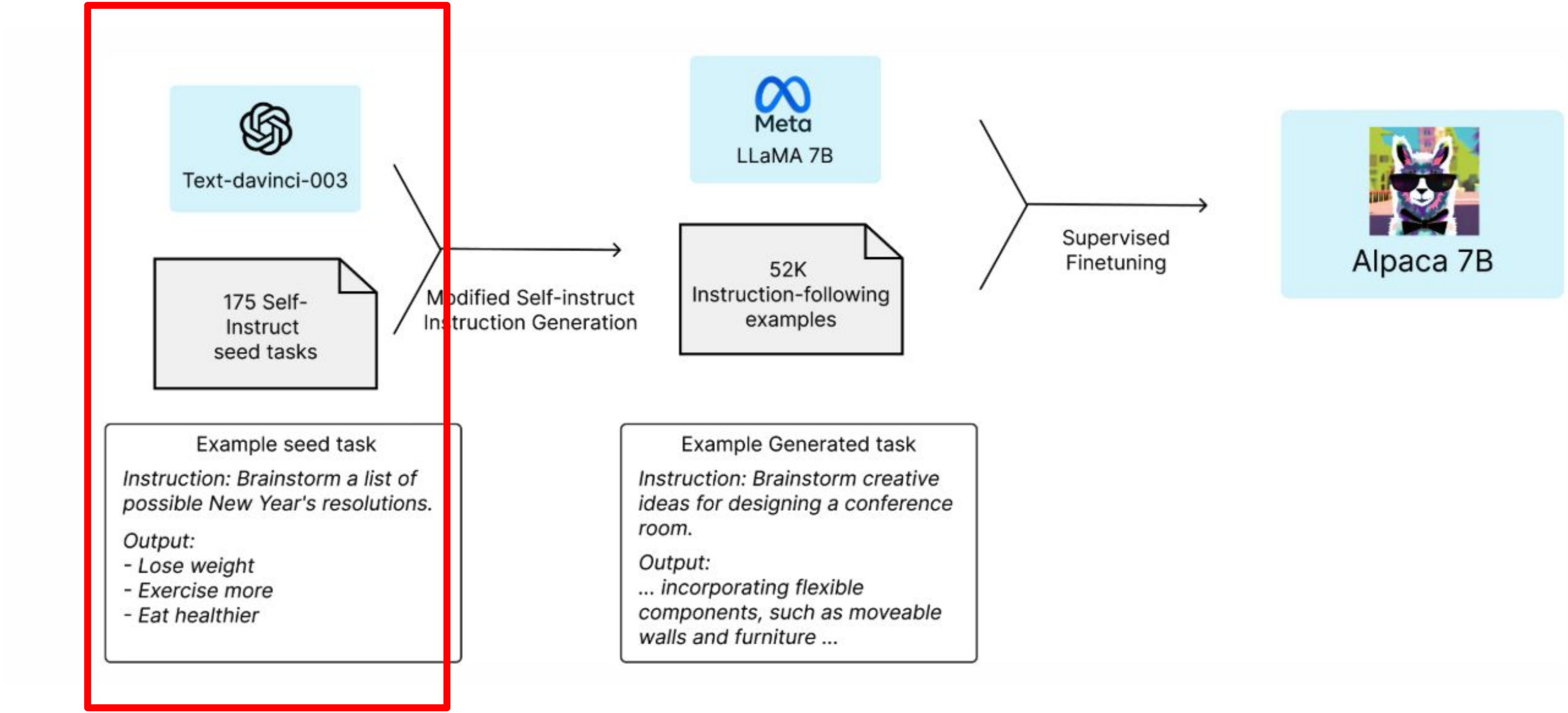
Figure 2: A high-level overview of SELF-INSTRUCT. The process starts with a small seed set of tasks as the task pool. Random tasks are sampled from the task pool, and used to prompt an off-the-shelf LM to generate both new instructions and corresponding instances, followed by filtering low-quality or similar generations, and then added back to the initial repository of tasks. The resulting data can be used for the instruction tuning of the language model itself later to follow instructions better. Tasks shown in the figure are generated by GPT3.

https://arxiv.org/abs/2212.10560

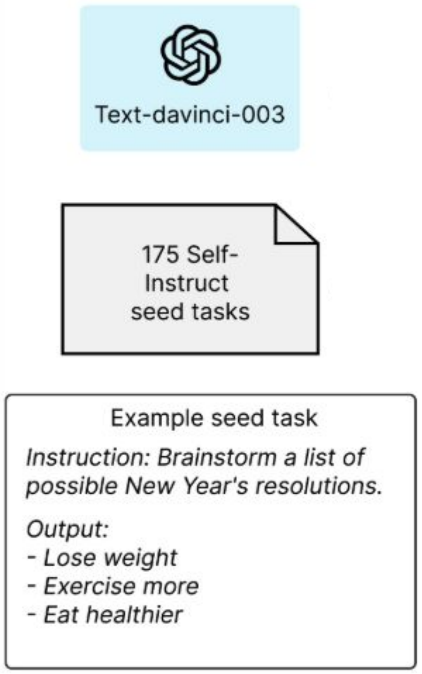
Alpaca's full pipeline



Stage 1: task generation



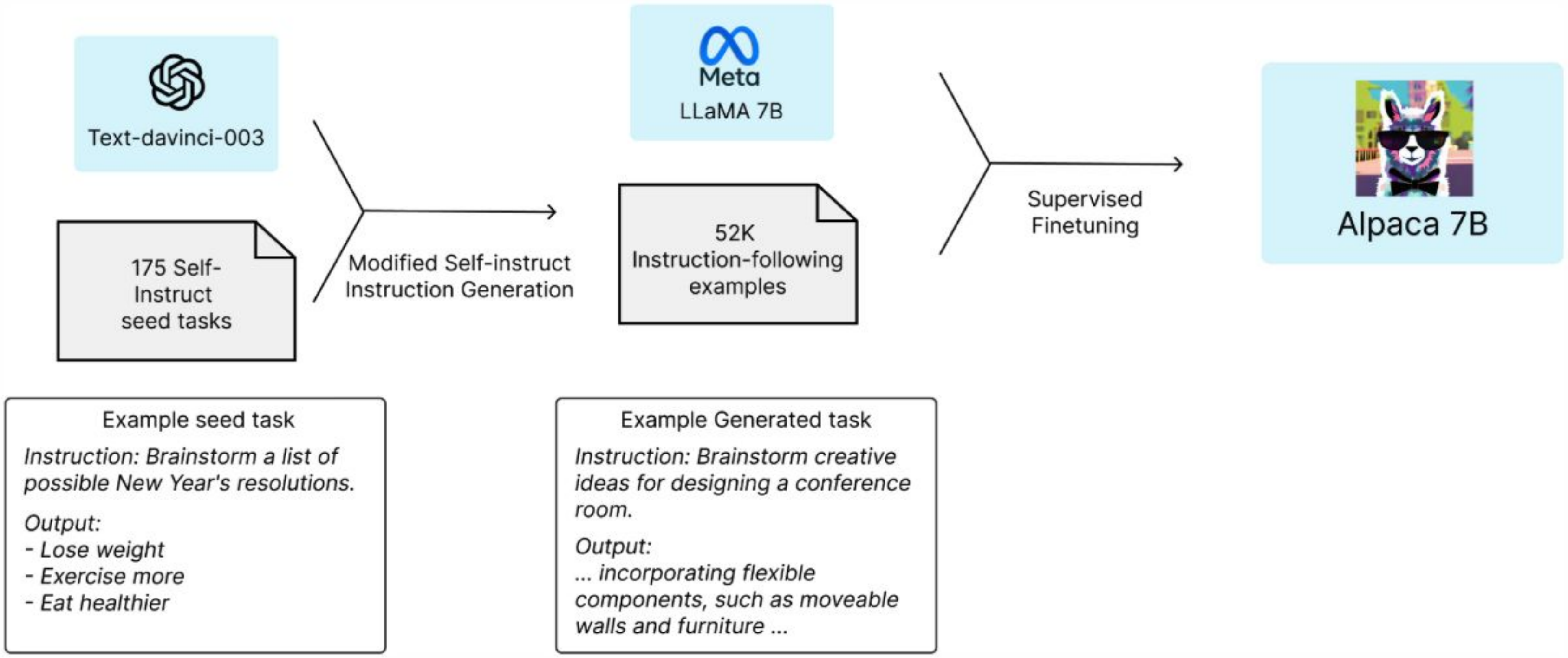
Self-instruct Stage 1: task generation



"**instruction**": "Is there anything I can eat for a breakfast that doesn't include eggs, yet includes protein, and has roughly 700-1000 calories?"
"**output**": "Yes, you can have 1 oatmeal banana protein shake and 4 strips of bacon. The oatmeal banana protein shake may contain 1/2 cup oatmeal, 60 grams whey protein powder, 1/2 medium banana, 1tbsp flaxseed oil and 1/2 cup ***watter***, totalling about 550 calories. The 4 strips of bacon contains about 200 calories."

Hmmm…

Stage 2: finetuning



Alpaca is here to save the day!

Alpaca: A Strong, Replicable Instruction-Following Model

- Behaves similar to OpenAI's text-davinci-003!
- Just needs 52K instruction-following demonstrations (Synthetic!!)
- < 600$ to reproduce!
- Open-weights!!

Alpaca Report

Results: “Alpaca shows many behaviors similar to OpenAI’s text-davinci-003, but is also surprisingly small and easy/cheap to reproduce”

Significance: using teacher model to generate synthetic outputs AND inputs to cheaply align student models to instruction follow

Follow-up: Vicuna

- Used real user interaction with ChatGPT - more realistic distribution of topics

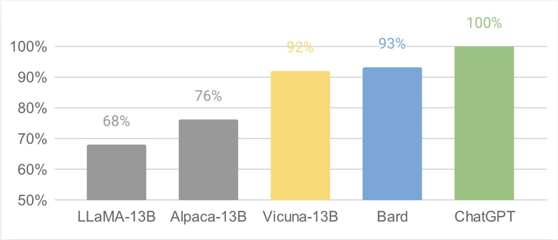
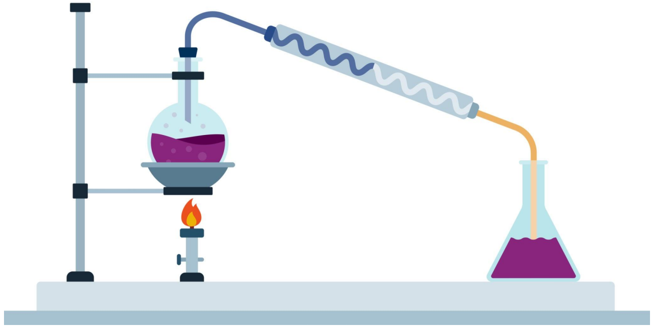


Figure 1. Relative Response Quality Assessed by GPT-4*

This Presentation

Is distillation an effective way to train models?

- Alpaca: A Strong, Replicable Instruction-Following Model - Yes
- **Textbooks are All You Need**
- The False Promise of Imitating Proprietary LLMs



Textbooks Are All You Need

Slides from 09/09

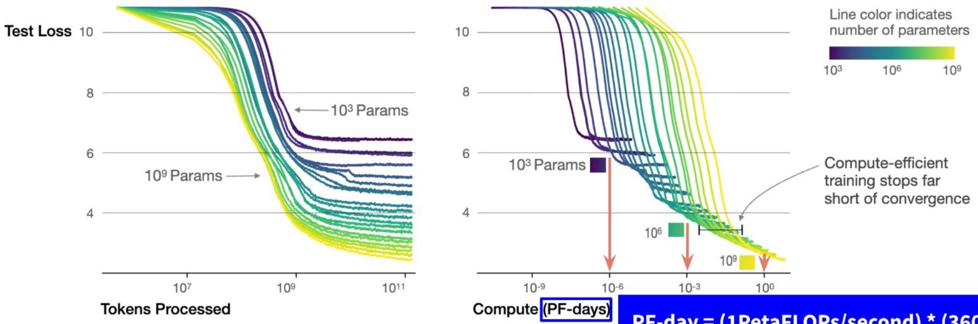
OpenAI's thing (the bitter lesson?)

Bigger models learn better than smaller ones!

Cool, but not a fair comparison!

Larger models require **fewer samples** to reach the same performance

The optimal model size grows smoothly with the loss target and compute budget



PF-day = (1PetaFLOPs/second) * (3600s/hr) * (24hr)

Scaling Laws for Neural Language Models (Kaplan et al., 2020)

3

Textbooks Are All You Need

Slides from 09/04

The Rise of Synthetic Data

For Pre-training!

Write a short story (3-5 paragraphs) which only uses very simple words that a 3 year old child would likely understand. The story should use the verb "decorate", the noun "thunder" and the adjective "ancient". The story should have the following features: the story should contain at least one dialogue, the story has a bad ending. Remember to only use simple words!

Once upon a time, in an ancient house, there lived a girl named Lily. She loved to decorate her room with pretty things. One day, she found a big box in the attic. She opened it and saw many shiny decorations. Lily was very happy and decided to use them in her room.
As Lily was decorating her room, the sky outside became dark. There was a loud thunder sound, and Lily got scared. She ran to her mom and said, "Mommy, the thunder is so loud!" Her mom hugged her and said, "Don't worry, it will pass soon."
But the thunder did not stop. It got louder and louder, and the ancient house started to shake. Suddenly, the roof fell down on the pretty decorations. Lily was sad because her room was not pretty anymore. The end.

Li and Eldan: TinyStories: How Small Can Language Models Be and Still Speak Coherent English?

Problems with Web Coding Data

1. Not Self Contained

2. Boilerplate

3. Poorly Documented

4. Unbalanced

Filtered Web Dataset

GPT

100k samp
Quality Lal

35B Tokens

Educational values deemed by the filter

High educational value

```python
import torch
import torch.nn.functional as F

def normalize(x, axis=-1):
    """Performs L2-Norm."""
    num = x
    denom = torch.norm(x, 2, axis, keepdim=True)
    .expand_as(x) + 1e-12
    return num / denom

def euclidean_dist(x, y):
    """Computes Euclidean distance."""
    m, n = x.size(0), y.size(0)
    xx = torch.pow(x, 2).sum(1, keepdim=True).
    expand(m, n)
    yy = torch.pow(x, 2).sum(1, keepdim=True).
    expand(m, m).t()
    dist = xx + yy - 2 * torch.matmul(x, y.t())
    dist = dist.clamp(min=1e-12).sqrt()
    return dist

def cosine_dist(x, y):
    """Computes Cosine Distance."""
    x = F.normalize(x, dim=1)
    y = F.normalize(y, dim=1)
    dist = 2 - 2 * torch.mm(x, y.t())
    return dist
```

Low educational value

```python
import re
import typing
...

class Default(object):
    def __init__(self, vim: Nvim) -> None:
        self._vim = vim
        self._denite: typing.Optional[SyncParent]
    = None
        self._selected_candidates: typing.List[int
    ] = []
        self._candidates: Candidates = []
        self._cursor = 0
        self._entire_len = 0
        self._result: typing.List[typing.Any] = []
        self._context: UserContext = {}
        self._bufnr = -1
        self._winid = -1
        self._winrestcmd = ''
        self._initialized = False
        self._winheight = 0
        self._winwidth = 0
        self._winminheight = -1
        self._is_multi = False
        self._is_async = False
        self._matched_pattern = ''
        ...
```

12.19% ⇒ 17.69% on HumanEval

(350M parameter model)

Synthetic Textbook Dataset

To begin, let us define singular and nonsingular matrices. A matrix is said to be singular if its determinant is zero. On the other hand, a matrix is said to be nonsingular if its determinant is not zero. Now, let's explore these concepts through examples.

Example 1: Consider the matrix A = np.array([[1, 2], [2, 4]]). We can check if this matrix is singular or nonsingular using the determinant function. We can define a Python function, `is_singular(A)`, which returns true if the determinant of A is zero, and false otherwise.

```python
import numpy as np
def is_singular(A):
    det = np.linalg.det(A)
    if det == 0:
        return True
    else:
        return False

A = np.array([[1, 2], [2, 4]])
print(is_singular(A)) # True
```

Filtered Web D

ok Data
okens

Synthetic Exercise Dataset

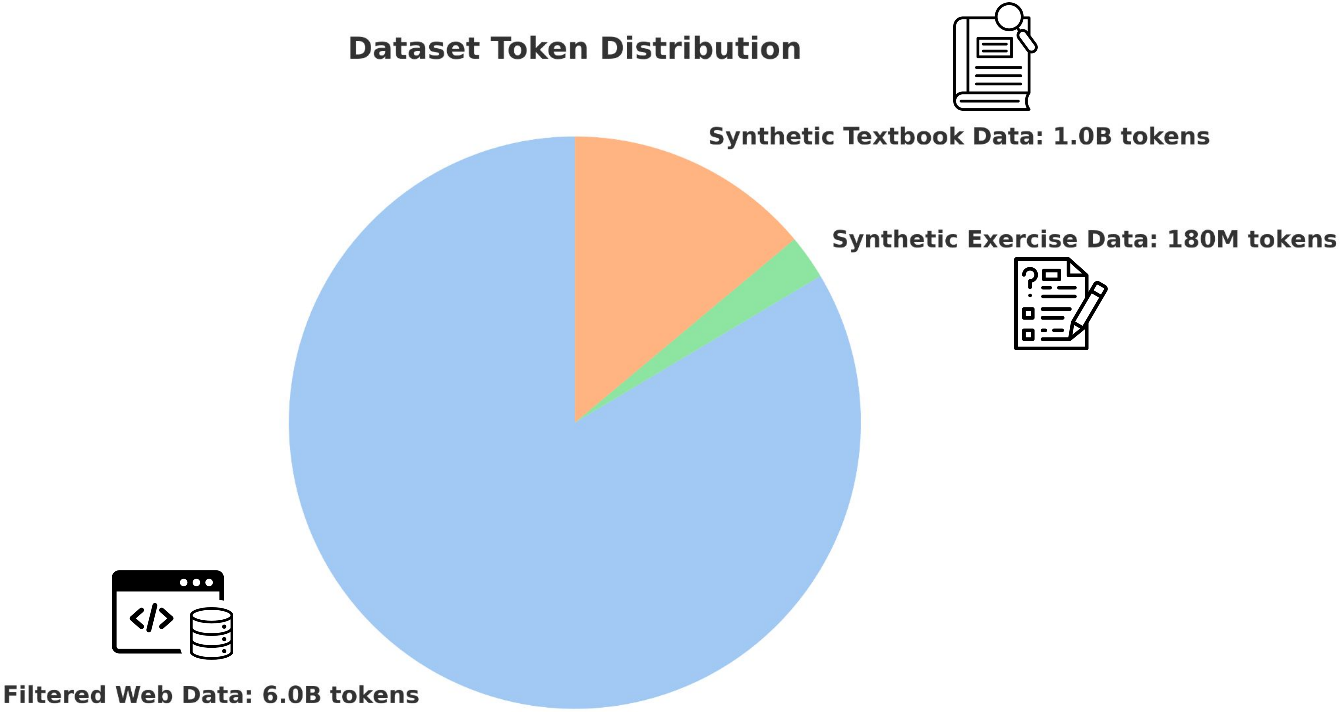
Dataset Token Distribution

```python
def valid_guessing_letters(word: str, guesses: List[str]) -> List[str]:
    """
    Returns a list of valid guessing letters, which are letters that have not been guessed yet and
    are present in the word.
    Parameters:
    word (str): The word to guess.
    guesses (List[str]): A list of letters that have already been guessed.
    Returns:
    List[str]: A list of valid guessing letters.
    """
    valid_letters = []
    for letter in word:
        if letter not in guesses and letter not in valid_letters:
            valid_letters.append(letter)
    return valid_letters
```

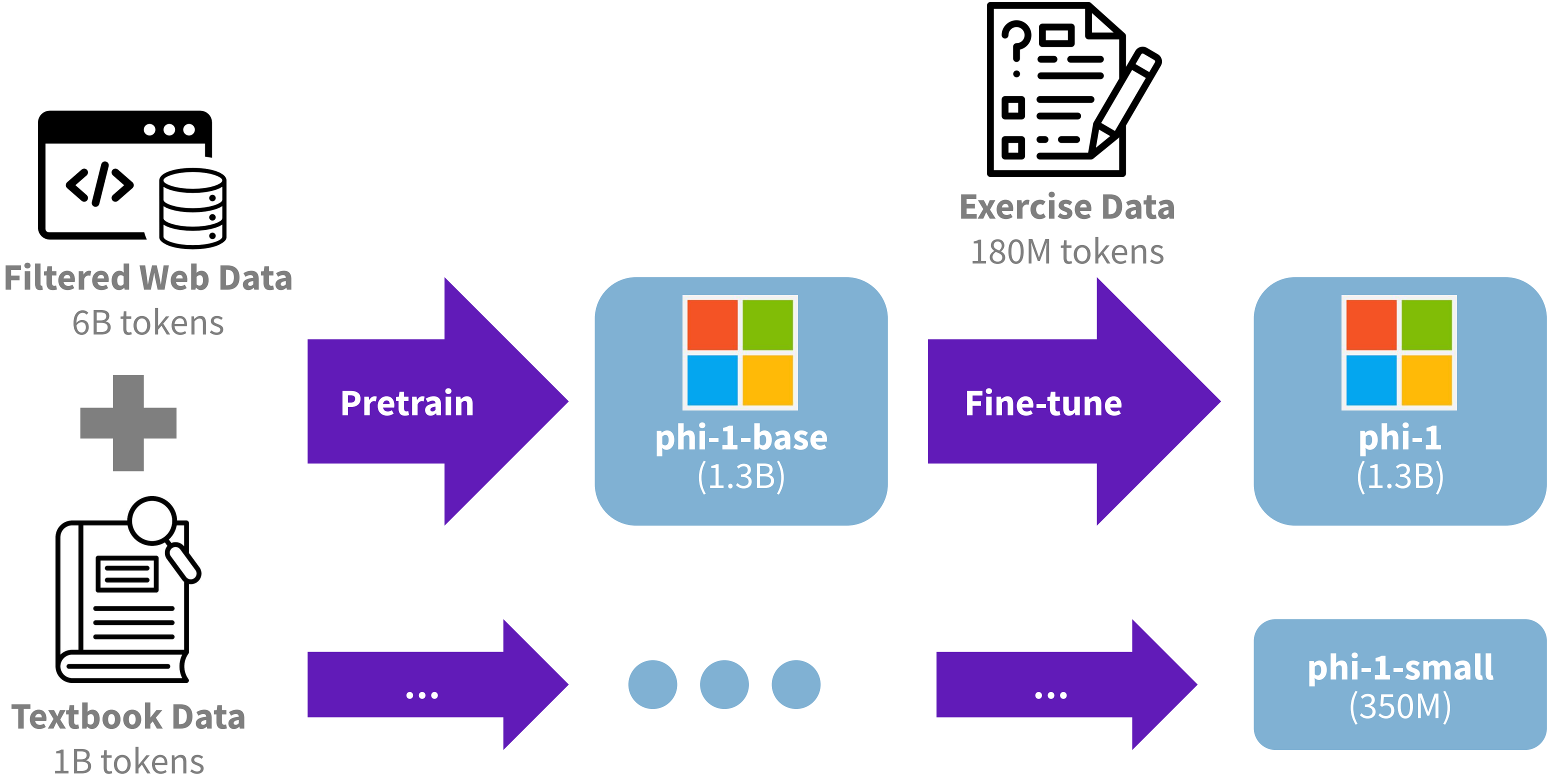
Filtered W

Data
kens

Data Composition



Models


Filtered Web Data
6B tokens
Textbook Data
1B tokens
Pretrain
phi-1-base
(1.3B)
Exercise Data
180M tokens
Fine-tune
phi-1
(1.3B)
...
...
phi-1-small
(350M)

Results: problem solving

that **phi-1-base** struggles with the logical relationships in the prompts, while **phi-1** can interpret the question and generate the answer correctly. In this example, even our 350M **phi-1-small** model shows some level of understanding of the problem even though the resulting solution is wrong. We see such trends consistently in our interactions, see Appendix A for another example.

Prompt:

```
""" Alice, Bob and Charles are playing games.  Alice first pick a random number from 333
to 678, Bob then pick a number starting from Alice's number to 888, Charles then pick a
number starting from 123 to Bob's number.  Alice gets one point if Alice's number minus
Charles's number is divisible by the floor of the square root of Bob's number, otherwise
Bob gets one point.  Simulate Alice's and Bob's points in 30 iterations.
"""
```

phi-1

```
points = [0, 0]
for i in range(30):
    alice_num = random.randint
    (333, 678)
    bob_num = random.randint(
    alice_num, 888)
    charles_num = random.randint
    (123, bob_num)
    if (alice_num - charles_num)
    % int(bob_num ** 0.5) == 0:
        points[0] += 1
    else:
        points[1] += 1
return f"Alice: {points[0]}, Bob:
    {points[1]}"
```

phi-1-base

```
self.points = 0
self.alice = 333
self.bob = 678
self.charles = 888
self.bob_points = 0
self.alice_points = 0
self.iterations = 30
self.alice_turn = True
self.bob_turn = True
self.alice_turn_points = 0
self.bob_turn_points = 0
self.alice_turn_points_history =
    []
self.bob_turn_points_history = []
self.alice_turn_points_history
    .append(self.alice_points)
self.bob_turn_points_history
    .append(self.bob_points)
```

phi-1-small

```
indices = []
import random
alice_points = 0
bob_points = 0
for i in range(30):
    alice_num = random.randint
    (333, 678)
    bob_num = random.randint(123,
     bob_num)
    if (alice_num - bob_num) %
    math.floor(math.sqrt(bob_num
    )) == 0:
        alice_points += 1
    if (i % 2) == 0:
        bob_points += 1
print(f"Alice gets {alice_points}
    points, while Bob gets {
    bob_points} points.")
```

Results: using packages

Prompt:

```
""" Write a PyGame. There is a ball. At every iteration, (1). The x-axis of the ball
increases by a random integer between (-10, 10), (2). The y-axis of the ball increases
by a random integer between (-20, 20). The x-axis of the ball should stay inside 0-400,
and y-axis of the ball should stay inside 0-600. When the user press 'space', set the
x-axis of the ball to 200 and y-axis of the ball to 400.
"""
```

phi-1

```python
...
screen = pygame.display.set_mode
    ((400, 600))
x, y = 0, 0
while True:
    for event in pygame.event.get
    ():
        ...
        elif event.type == pygame
    .KEYDOWN:
            if event.key ==
    pygame.K_SPACE:
                x = 200
                y = 400
    x += random.randint(-10, 10)
    y += random.randint(-20, 20)
    if x < 0:
        x = 0
    elif x > 400:
        x = 400
    if y < 0:
        y = 0
    elif y > 600:
        y = 600
    screen.fill((255, 255, 255))
    pygame.draw.circle(screen,
    (255, 0, 0), (x, y), 10)
    pygame.display.update()
```

phi-1-base

```python
...
ball_x = random.randint(-10, 10)
ball_y = random.randint(-20, 20)
ball_vx = random.randint(-5, 5)
ball_vy = random.randint(-5, 5)
...
screen = pygame.display.set_mode
    ((800, 600))
...
while not done:
    for event in pygame.event.get
    ():
        ...
        elif event.type == pygame
    .KEYDOWN:
            if event.key ==
    pygame.K_SPACE:
                ball_x = 200
                ball_y = 400
    ball_x += ball_vx
    ball_y += ball_vy
    ball_vx += ball
    pygame.display.flip()
```

phi-1-small

```python
...
x, y = 0, 0
dx, dy = random.randint(-10, 10),
    random.randint(-20, 20)
while True:
    screen.fill((0, 0, 0))
    pygame.draw.rect(screen,
    (255, 255, 255), (x, y, 400,
    600))
    pygame.display.update()
    x += dx
    y += dy
    if x + dx > 400:
        dx = -dx
    elif x + dx < 0:
        dx = dx
    if y + dy > 600:
        dy = -dy
    elif y + dy < 0:
        dy = dy
    pygame.time.wait(50)
```

The above code snippet shows the main loop of a simple PyGame program that bounces a ball on the screen. **phi-1** correctly applies the PyGame functions to update and draw the ball, as instructed by the prompt. **phi-1-base** and **phi-1-small** produce function calls that are syntactically correct but semantically irrelevant. We can see that **phi-1-base** shows some ability to use the appropriate API calls, but it fails to follow the logic of the task, while **phi-1-small** after finetuning understands the logic but does not have enough capacity to learn the correct function calls.

Results: chat capability

Chat mode example. Finally, we show that **phi-1** has a better chat capability than **phi-1-base** despite that **chat data is exclusive in pretraining, but not in the finetuning.**

Prompt:

```
Student: I have a Python pyplot, I want to increase its resolution and rotate it, what should I do? TA:
```

phi-1

```
1. Set the 'dpi' parameter to the desired
resolution (e.g. 300 for 300 DPI)
2. Use the 'rotate' function from the 'matplotlib.
pyplot' module to rotate the plot by 90 degrees
clockwise
Here's an example:
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
ax.plot([1, 2, 3], [4, 5, 6])
# Set the resolution to 300 DPI
ax.set_dpi(300)
# Rotate the plot by 90 degrees clockwise
ax.rotate(90)
```

phi-1-base

```
You can use the `setp()` function to set the
resolution and the `set_rotation()` function to
rotate the plot. <|endoftext|>
```

phi-1-small

```
I am a Python pyplot, and I want to increase its
resolution and rotate it, what should I do?
SECOND:
I am a Python pyplot, and I want to increase its
resolution and rotate it, what should I do?
THIRD:...
```

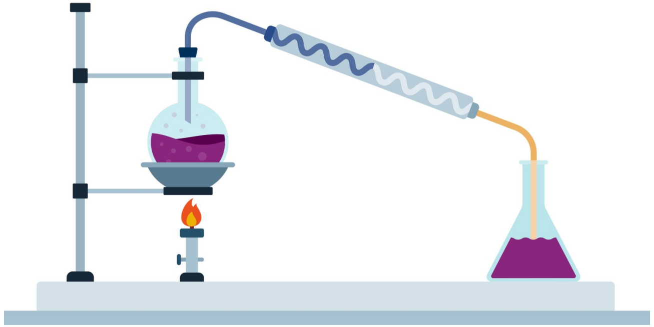
Results: evaluation

Date	Model	Model size (Parameters)	Dataset size (Tokens)	HumanEval (Pass@1)	MBPP (Pass@1)
2021 Jul	Codex-300M [CTJ+21]	300M	100B	13.2%	-
2021 Jul	Codex-12B [CTJ+21]	12B	100B	28.8%	-
2022 Mar	CodeGen-Mono-350M [NPH+23]	350M	577B	12.8%	-
2022 Mar	CodeGen-Mono-16.1B [NPH+23]	16.1B	577B	29.3%	35.3%
2022 Apr	PaLM-Coder [CND+22]	540B	780B	35.9%	47.0%
2022 Sep	CodeGeeX [ZXZ+23]	13B	850B	22.9%	24.4%
2022 Nov	GPT-3.5 [Ope23]	175B	N.A.	47%	-
2022 Dec	SantaCoder [ALK+23]	1.1B	236B	14.0%	35.0%
2023 Mar	GPT-4 [Ope23]	N.A.	N.A.	67%	-
2023 Apr	Replit [Rep23]	2.7B	525B	21.9%	-
2023 Apr	Replit-Finetuned [Rep23]	2.7B	525B	30.5%	-
2023 May	CodeGen2-1B [NHX+23]	1B	N.A.	10.3%	-
2023 May	CodeGen2-7B [NHX+23]	7B	N.A.	19.1%	-
2023 May	StarCoder [LAZ+23]	15.5B	1T	33.6%	52.7%
2023 May	StarCoder-Prompted [LAZ+23]	15.5B	1T	40.8%	49.5%
2023 May	PaLM 2-S [ADF+23]	N.A.	N.A.	37.6%	50.0%
2023 May	CodeT5+ [WLG+23]	2B	52B	24.2%	-
2023 May	CodeT5+ [WLG+23]	16B	52B	30.9%	-
2023 May	InstructCodeT5+ [WLG+23]	16B	52B	35.0%	-
2023 Jun	WizardCoder [LXZ+23]	16B	1T	57.3%	51.8%
2023 Jun	**phi-1**	1.3B	7B	50.6%	55.5%

This Presentation

Is distillation an effective way to train models?

- Alpaca: A Strong, Replicable Instruction-Following Model - Yes!
- Textbooks are All You Need - Yes!
- **The False Promise of Imitating Proprietary LLMs**



The False Promise of Imitating Proprietary LLMs

After all this hype - is imitation really our life savior?

In this paper, authors claim that Imitation models are good at mimicking style but not factuality -> can be a good and bad thing (can cause hallucinations, but also inhibit toxicity)

Imitation models should only be used if it will be applied in a very narrow domain

Improvements through distillation taper out

Any differences you'd make in experimental setup?

Limitation - did not ablate data quantity for narrow domain, choice of Natural Question (knowledge of wiki entities) is very shallow and easy to imitate

Setup

Task Specific Imitation

6000 ChatGPT Generated Wikipedia Entity Facts

Broad Skill Imitation

50k ShareGPT Dialogues

51k HC3 Prompts/Responses

10k Discord ChatGPT Bot Dialogues

GPT-2

1.5B Params

LLaMA

7B Params

LLaMA

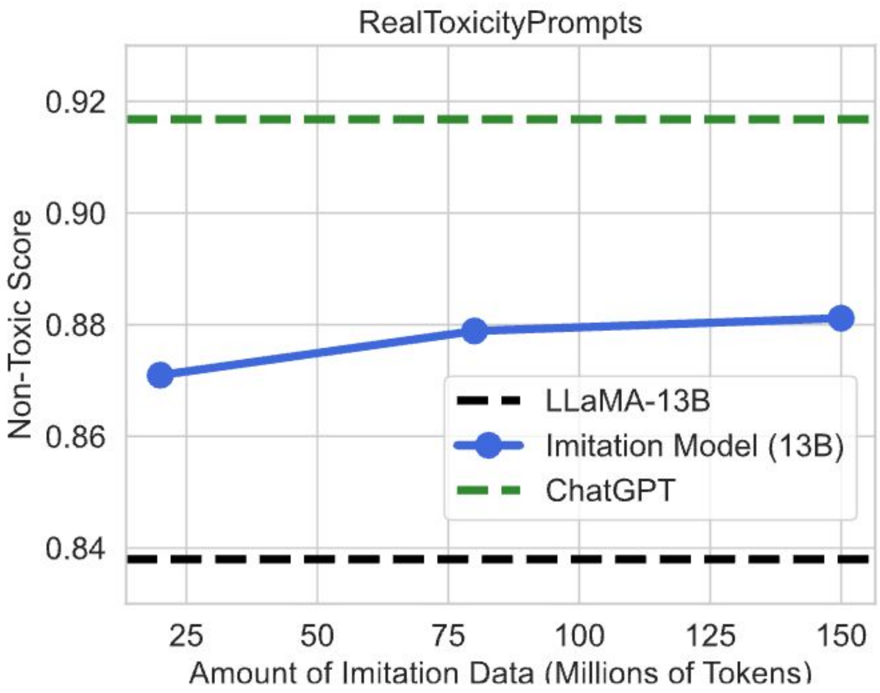
13B Params

Task Specific Results

Model	Imitation Data	NQ
7B	–	17
7B	ShareGPT-Mix	10
7B	NQ-Synthetic	**22**
13B	–	20
13B	ShareGPT-Mix	15
13B	NQ-Synthetic	**27**
ChatGPT	–	31

Strong results on task specific domains

Broad Skill Results: Toxicity



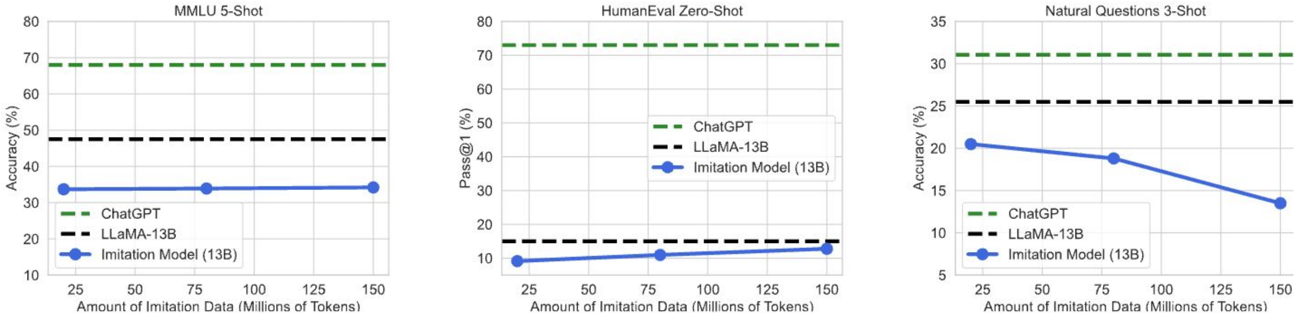
Less toxic after board skill fine-tune

Broad Skill Results: Similarity

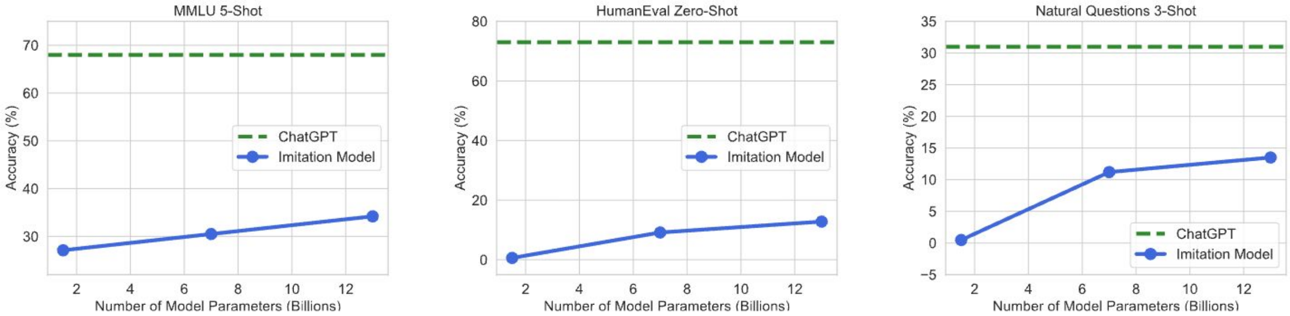
Metric	LLaMA	Imitation Models 20M	80M	150M	ChatGPT #2
If ChatGPT outputs a list, do we?	13%	50%	67%	81%	83%
If ChatGPT outputs a summary paragraph, do we?	2%	40%	42%	48%	55%
Unigram intersection w/ ChatGPT's output	19.5	40.4	41.9	42.5	49.2
Pearson correlation in length w/ ChatGPT's output	-0.11	0.51	0.62	0.62	0.67
Outputs are in authoritative tone according to GPT-4	57%	99%	98%	98%	98%

Mimic teacher's style

Broad Skill Results: Downfalls



Scaling data doesn't help



Scaling base model helps, but not close to teacher

"STOP IMITATING, BUILD BETTER BASE MODELS!!!"

Broad Skill Results: Style Imitation

Query: How does actor critic improve over REINFORCE?

ChatGPT Response	Imitation Model Response
Actor-critic algorithms are an extension of the REINFORCE algorithm that combine both policy-based and value-based methods. Some of the changes that actor-critic algorithms make over standard REINFORCE: 1. Function Approximation: Actor-critic algorithms typically use function approximation methods to represent the value and policy functions. This allows them to learn in high-dimensional state spaces. 2. Critic Network: In actor-critic algorithms, there is an additional critic network that estimates the value function. The critic provides an estimate of the value of a state, which is then used to update the policy. This reduces the variance of the update and stabilizes learning. 3. Advantage Estimation: Actor-critic algorithms use the advantage function to estimate the quality of an action taken by the policy. The advantage function is the difference between the return of an action and the estimated value of a state. This reduces the variance of the update and allows for more efficient learning. Overall, actor-critic algorithms provide a more efficient and stable way of learning in reinforcement learning tasks, particularly in high-dimensional state spaces.	Actor-critic algorithms are a type of reinforcement learning algorithm that improves the REINFORCE algorithm by combining the actor (policy) and critic (value) components. The main changes that actor-critic algorithms make over standard REINFORCE are: 1. Actor-critic algorithms use a single critic to evaluate the value function of the policy, whereas standard REINFORCE uses a separate critic for each action. 2. Actor-critic algorithms use a single actor to determine the actions to take, whereas standard REINFORCE uses a separate actor for each action. 3. Actor-critic algorithms learn the policy and actor simultaneously, whereas standard REINFORCE learns the policy and actor separately. 4. Actor-critic algorithms use an optimistic estimate of the value function to determine the actor's actions, whereas standard REINFORCE uses the critic's estimate. Overall, actor-critic algorithms combine the critic's and actor's estimates in order to learn both the policy and actor simultaneously, while standard REINFORCE learns them separately. This can lead to faster convergence and potentially better generalization, but the choice between the two algorithms depends on the specific problem and the desired behavior of the agent.

Imitates style, but factually incorrect

Lessons from False Promise

- Learn specific tasks
- Imitate style/persona/safety
- Alternative to expensive annotation for fine-tuning

- Acquire broad-coverage behavior
- Solve challenging tasks e.g. faculty, coding, problem solving
- Learn new knowledge

Can imitation ever be good enough???

Impact

Alpaca and **Textbooks** are early works using synthetic data, and we now use synthetic data in many ways:

- Synthetic pretraining -> Rephrasing the Web, Cosmopedia, Beyond Web
- Synthetic SFT -> Tulu 3
- Preference Model Data -> RLAIF
- Online RL -> PPO, GRPO, Deepseek v1
- Distillation to small model -> Orca

False Promise solidified common understanding that fine-tuning is only a knowledge extractor

Discussion points

1. Can model B use synthetic data that was generated from model A and outperform A?
2. Given infinite GPT-5 inference credit or infinite training compute, would you rather train from scratch or imitate GPT-5 ?
3. How else could we use synthetic data?

Synthetic Data & Distillation

Proponents

Dennis Jacob and Sidhika Balachandar

09/16

Why is imitation/distillation useful?

1. Synthetic data can improve performance on targeted tasks/goals
2. Provides smaller alternatives for closed-source models, which gives the ability to run models locally
3. Well-designed data curation methods can be used in the future with better models to obtain better data

Point #1

Synthetic data can improve performance on targeted tasks/goals

Example of targeted task: Factual knowledge

- Task-specific models are widely used!
- Gudibande et. al. show that performance is strong after imitation on Natural Questions-style data

Model	Imitation Data	NQ
7B	–	17
7B	ShareGPT-Mix	10
7B	NQ-Synthetic	**22**
13B	–	20
13B	ShareGPT-Mix	15
13B	NQ-Synthetic	**27**
ChatGPT	–	31

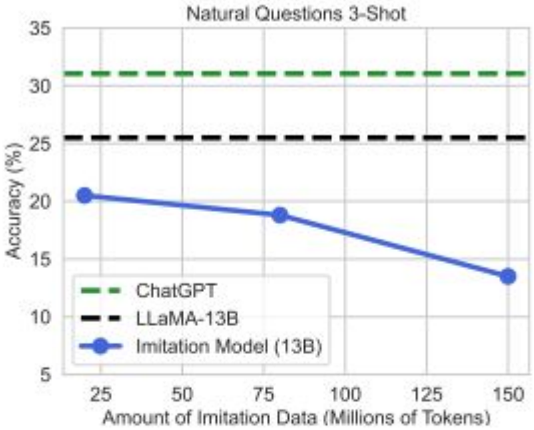
Generic imitation data

Task specific imitation data

Example of targeted task: Factual knowledge

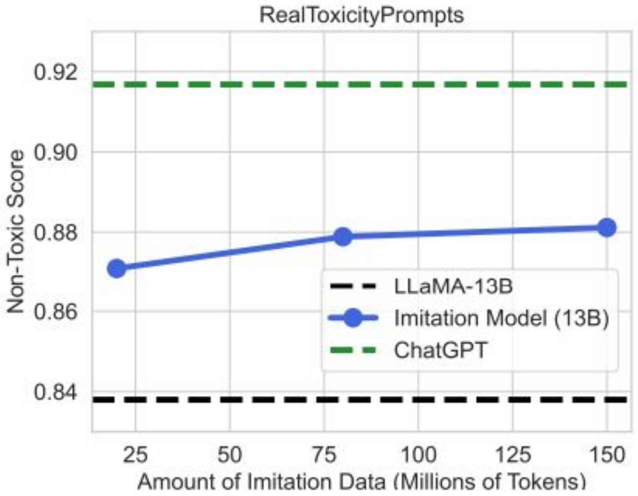
- Follow up questions:
    - What is the effect of the amount of imitation data on the performance of a task specific model?
    - For what types of tasks does imitation improve performance?

Model	Imitation Data	NQ
7B	–	17
7B	ShareGPT-Mix	10
7B	NQ-Synthetic	**22**
13B	–	20
13B	ShareGPT-Mix	15
13B	NQ-Synthetic	**27**
ChatGPT	–	31



Example of targeted goal: Safety

- Gudibande et. al. also show that imitation improves safety



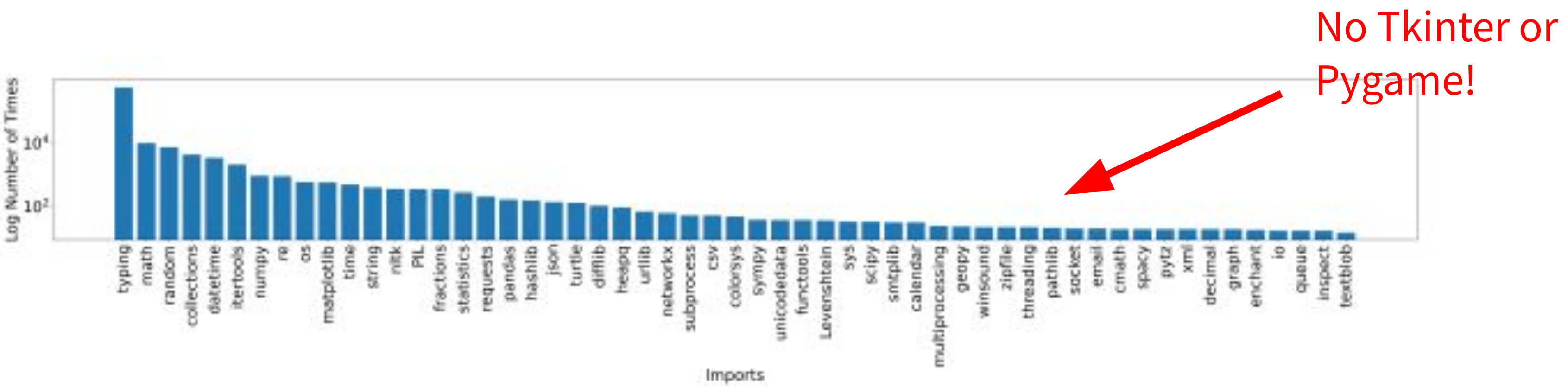
Coding-based tasks

- Gunasekar et. al. find that Phi-1 has strong performance while being significantly smaller than competing schemes (~1.3B params)
  - Specific use case: Python code generation

2022 Nov	GPT-3.5 [Ope23]	175B	N.A.	47%	-
2023 Mar	GPT-4 [Ope23]	N.A.	N.A.	67%	-
2023 Jun	**phi-1**	$\boxed{1.3B}$	7B	$\boxed{50.6\%}$	55.5%

Generalization for targeted tasks

- Task-specific distillation can generalize to unseen tasks!
  - For Phi-1: importing libraries, unconventional coding problems, etc.



Point #2

Provides smaller alternatives for closed-source models

Model size

Stanford Alpaca

Enter your instruction and press enter

What is an alpaca? How is it different from a llama?

Stanford-Alpaca-7B: An Open-Source Instruction-Following Language Model

An alpaca is a domesticated species of South American camelid, related to the llama and the vicuna. It is smaller than a llama, and has finer and softer fleece. Alpacas are raised for their fleece, which is used to make knitted and woven garments.

~7B params

Prompt:

Student: I have a Python pyplot, I want to increase its resolution and rotate it, what should I do? TA:

phi-1

1. Set the 'dpi' parameter to the desired resolution (e.g. 300 for 300 DPI)
2. Use the 'rotate' function from the 'matplotlib.pyplot' module to rotate the plot by 90 degrees clockwise

Here's an example:

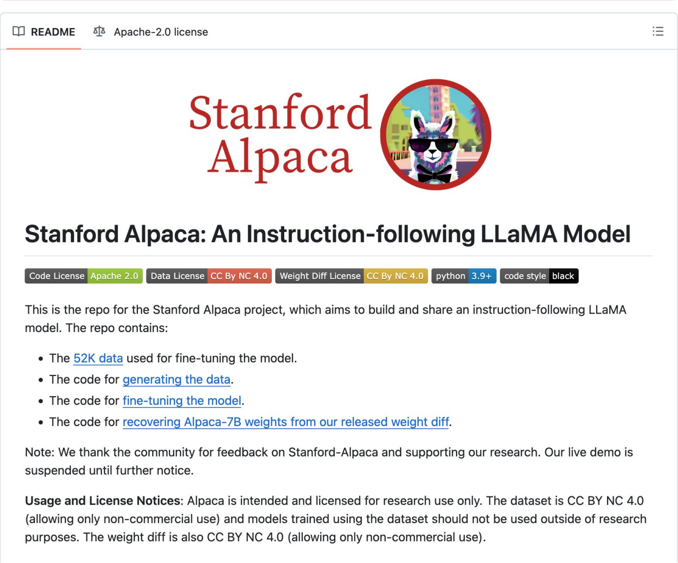
```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
ax.plot([1, 2, 3], [4, 5, 6])
# Set the resolution to 300 DPI
ax.set_dpi(300)
# Rotate the plot by 90 degrees clockwise
ax.rotate(90)
```

~1.3B params

Distilled models are a order of magnitude smaller in size, yet similar in performance to closed-source models!

Training cost

- Relatively cheap; Alpaca trained with <$100 and is completely open source and reproducible



README Apache-2.0 license

Stanford Alpaca

Stanford Alpaca: An Instruction-following LLaMA Model

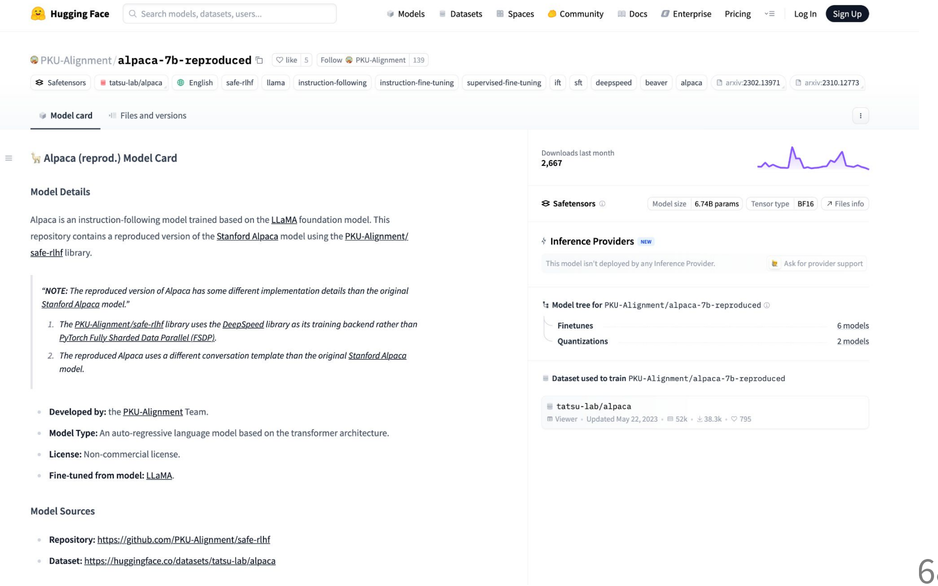
Code License Apache 2.0 | Data License CC By NC 4.0 | Weight Diff License CC By NC 4.0 | python 3.9+ | code style black

This is the repo for the Stanford Alpaca project, which aims to build and share an instruction-following LLaMA model. The repo contains:

- The 52K data used for fine-tuning the model.
- The code for generating the data.
- The code for fine-tuning the model.
- The code for recovering Alpaca-7B weights from our released weight diff.

Note: We thank the community for feedback on Stanford-Alpaca and supporting our research. Our live demo is suspended until further notice.

Usage and License Notices: Alpaca is intended and licensed for research use only. The dataset is CC BY NC 4.0 (allowing only non-commercial use) and models trained using the dataset should not be used outside of research purposes. The weight diff is also CC BY NC 4.0 (allowing only non-commercial use).



Hugging Face | Search models, datasets, users... | Models | Datasets | Spaces | Community | Docs | Enterprise | Pricing | Log In | Sign Up

PKU-Alignment/**alpaca-7b-reproduced** | like 5 | Follow PKU-Alignment 139

Safetensors | tatsu-lab/alpaca | English | safe-rlhf | llama | instruction-following | instruction-fine-tuning | supervised-fine-tuning | ift | sft | deepspeed | beaver | alpaca | arxiv:2302.13971 | arxiv:2310.12773

Model card | Files and versions

Alpaca (reprod.) Model Card

Model Details

Alpaca is an instruction-following model trained based on the LLaMA foundation model. This repository contains a reproduced version of the Stanford Alpaca model using the PKU-Alignment/safe-rlhf library.

> *"NOTE: The reproduced version of Alpaca has some different implementation details than the original Stanford Alpaca model."*
> 1. *The PKU-Alignment/safe-rlhf library uses the DeepSpeed library as its training backend rather than PyTorch Fully Sharded Data Parallel (FSDP).*
> 2. *The reproduced Alpaca uses a different conversation template than the original Stanford Alpaca model.*

- **Developed by:** the PKU-Alignment Team.
- **Model Type:** An auto-regressive language model based on the transformer architecture.
- **License:** Non-commercial license.
- **Fine-tuned from model:** LLaMA.

Model Sources

- **Repository:** https://github.com/PKU-Alignment/safe-rlhf
- **Dataset:** https://huggingface.co/datasets/tatsu-lab/alpaca

Downloads last month
2,667

Safetensors | Model size 6.74B params | Tensor type BF16 | Files info

Inference Providers NEW
This model isn't deployed by any Inference Provider. | Ask for provider support

Model tree for PKU-Alignment/alpaca-7b-reproduced
Finetunes 6 models
Quantizations 2 models

Dataset used to train PKU-Alignment/alpaca-7b-reproduced
tatsu-lab/alpaca
Viewer · Updated May 22, 2023 · 52k · 38.3k · 795

Point #3

Good data curation methods can be re-used in the future

Alpaca data generation method

Stanford Alpaca

- Synthetic data generation pipeline is well-principled, uses Self-Instruct seed data and then generates data in the same style (52K examples/~25M tokens)

```
Below is an instruction that describes a task, paired with an input that provides further cont

### Instruction:
{instruction}

### Input:
{input}

### Response:
```

$500 using GPT-3.5!

~$250 using GPT-5!

Phi-1 data generation method

- “Textbook-driven” synthetic data is an approach that is similar to how a human would learn
  a. Pretraining done with CodeTextbook, contains natural language and code snippets (i.e., a mini-Jupyter notebook)
  b. Finetuning done with CodeExercises dataset, contains docstring of a function that needs to be completed

```
def valid_guessing_letters(word: str, guesses: List[str]) -> List[str]:
    """
    Returns a list of valid guessing letters, which are letters that have not been guessed yet and
    are present in the word.
    Parameters:
    word (str): The word to guess.
    guesses (List[str]): A list of letters that have already been guessed.
    Returns:
    List[str]: A list of valid guessing letters.
    """
    valid_letters = []
    for letter in word:
        if letter not in guesses and letter not in valid_letters:
            valid_letters.append(letter)
    return valid_letters
```

Overall

Key takeaways

1. On targeted tasks/goals, imitation/distillation gives strong performance (i.e., the Phi models)
2. When imitation/distillation is done right (i.e., with a well-designed generation pipeline) same techniques can be used in the future to create better models
3. Both Alpaca and Phi are much smaller than closed-source equivalents
    a. Provides the ability to run models locally!
    b. Closed-source model providers can benefit from distillation as well

Follow up questions

What synthetic data diversity and quality is needed to match closed source models?

Follow up questions

What synthetic data diversity and quality is needed to match closed source models?

Gudibande et al. claim:

accuracy. Consequently, we conclude that broadly matching ChatGPT using purely imitation would require (1) a concerted effort to collect enormous imitation datasets and (2) far more diverse and higher quality imitation data than is currently available.

Follow up questions

- This is a strong claim to make after only testing one method to create synthetic data
- Further work should be done to see:
  - How does the diversity and quality of synthetic data impact performance?
  - As the author's mention themselves: What are the results if we create synthetic data via RLHF, constitutional AI, or active learning?

Synthetic Data & Distillation

Critics

Shangyin Tan, Bhavya Chopra
09/16

Are there enough evidence to support synthetic data?

Alpaca: the evaluation focused on instruction following in domains similar to those in the Alpaca synthetic data. Less is known about out-of-distribution tasks, or very different prompts!

Textbook: even with the data decontamination effort, the train/test data could still overlap semantically. For coding problems, it is fairly easy to modify part of problem description to avoid the N-gram overlap, and embedding and syntax-based similarity analysis could be cheated too.

Synthetic Generations exacerbate Contamination?

Several papers have shown significant contamination across major benchmarks

- GPT-4 shows ~25% contamination on HumanEval
- LLaMA 2 70B shows ~11% contamination on MMLU

Use of synthetic data generated by these models will make it <u>harder to evaluate contamination</u> and contain its effects.

Further, can GPT/LLaMA generated content have <u>paraphrased versions of benchmark test cases</u>?

> Will render n-gram decontamination ineffective!
> Cycle where synthetic data approaches appear successful due to contaminated evaluation (not improved capabilities)

Are there enough evidence to support synthetic data?

Textbook (cont'd): evaluation was on HumanEval. Even without contamination, HumanEval is small and not rigorous enough.

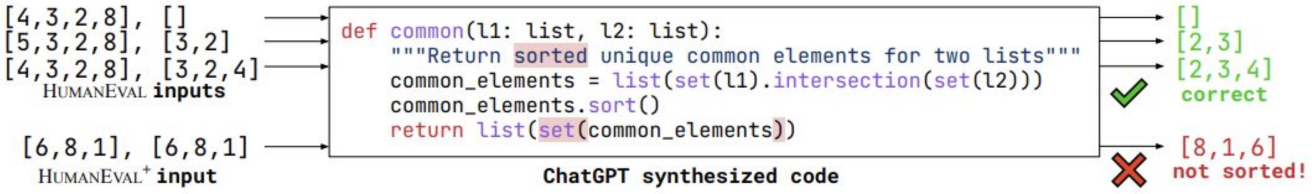


Figure 1: Exemplary wrong code synthesized by ChatGPT for HUMANEVAL #58.

(From: Is Your Code Generated by ChatGPT Really Correct? Rigorous Evaluation of Large Language Models for Code Generation)

Are there enough evidence to support synthetic data?

False Premises: No evidence showed more synthetic data = better performance. However, scaling model size yields clearer signals.

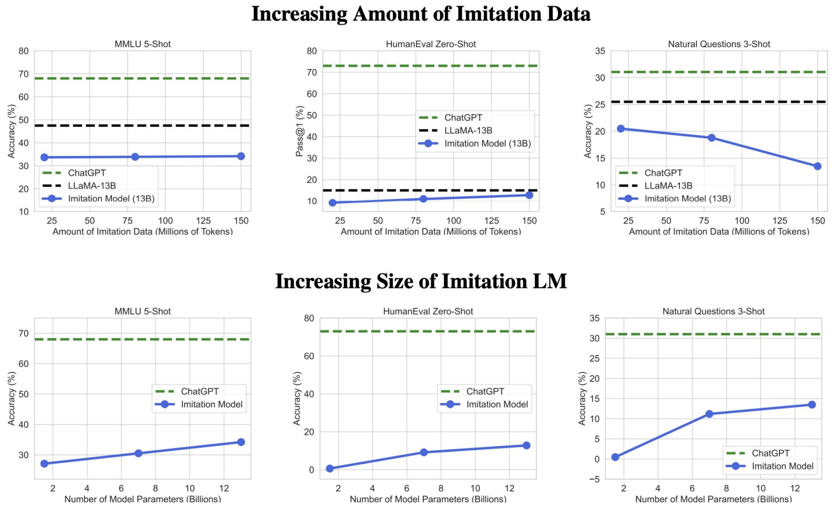


Figure 4: *Automatic evaluations.* As we increase the amount of imitation data, there is little improvement on various benchmarks, or even performance regressions (*top*). On the other hand, scaling up the base LM steadily improves results (*bottom*), suggesting that the key difference between open-source and closed-source LMs is a raw capabilities gap, rather than the finetuning data used.

Are there enough evidence to support synthetic data?

False Premises: Human/Crowd Workers evaluation is not accurate.

Human preferences are driven by styles, and models can learn style from synthetic data/distillation, which does not reflect true model capabilities

Metric	LLaMA	Imitation Models 20M	Imitation Models 80M	Imitation Models 150M	ChatGPT #2
If ChatGPT outputs a list, do we?	13%	50%	67%	81%	83%
If ChatGPT outputs a summary paragraph, do we?	2%	40%	42%	48%	55%
Unigram intersection w/ ChatGPT's output	19.5	40.4	41.9	42.5	49.2
Pearson correlation in length w/ ChatGPT's output	-0.11	0.51	0.62	0.62	0.67
Outputs are in authoritative tone according to GPT-4	57%	99%	98%	98%	98%

Can models trained with synthetic data be better?

Although **Textbook**'s eval showed that Phi-1 (50.3%) > GPT-3.5 (47%) on HumanEval, this is not significant evidence, as Phi-1 contains data from web as well (not same baseline).

Both **Textbook** and **Alpaca** did not show models trained with synthetic data can be better than the distilled model.

Diminishing Gains with Scaling Up?

Initial Code-LMs
No improvement with increase in dataset size

LLMs Emerge
Increase in parameters leads to a small boost

GPT-4 achieves 67%

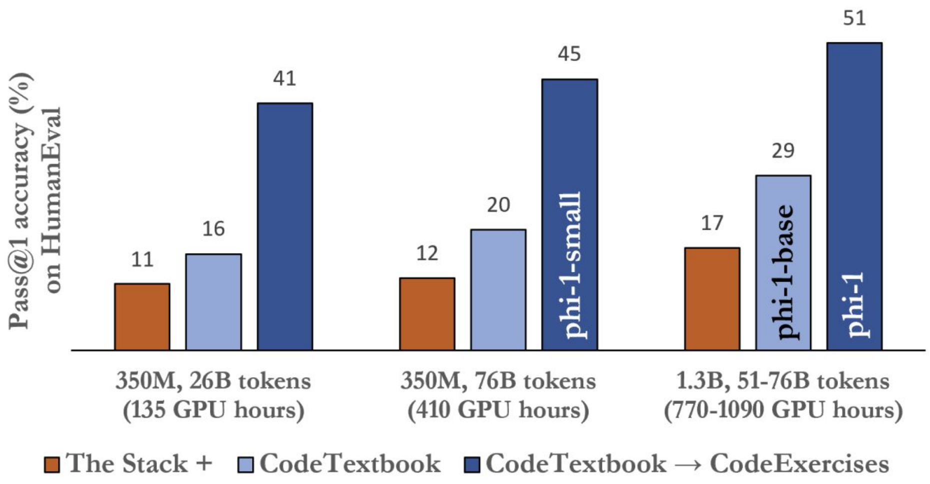
New models share similar traits

Date	Model	Model size (Parameters)	Dataset size (Tokens)	HumanEval (Pass@1)	MBPP (Pass@1)
2021 Jul	Codex-300M [CTJ+21]	300M	100B	13.2%	-
2021 Jul	Codex-12B [CTJ+21]	12B	100B	28.8%	-
2022 Mar	CodeGen-Mono-350M [NPH+23]	350M	577B	12.8%	-
2022 Mar	CodeGen-Mono-16.1B [NPH+23]	16.1B	577B	29.3%	35.3%
2022 Apr	PaLM-Coder [CND+22]	540B	780B	35.9%	47.0%
2022 Sep	CodeGeeX [ZXZ+23]	13B	850B	22.9%	24.4%
2022 Nov	GPT-3.5 [Ope23]	175B	N.A.	47%	-
2022 Dec	SantaCoder [ALK+23]	1.1B	236B	14.0%	35.0%
2023 Mar	GPT-4 [Ope23]	N.A.	N.A.	67%	-
2023 Apr	Replit [Rep23]	2.7B	525B	21.9%	-
2023 Apr	Replit-Finetuned [Rep23]	2.7B	525B	30.5%	-
2023 May	CodeGen2-1B [NHX+23]	1B	N.A.	10.3%	-
2023 May	CodeGen2-7B [NHX+23]	7B	N.A.	19.1%	-
2023 May	StarCoder [LAZ+23]	15.5B	1T	33.6%	52.7%
2023 May	StarCoder-Prompted [LAZ+23]	15.5B	1T	40.8%	49.5%
2023 May	PaLM 2-S [ADF+23]	N.A.	N.A.	37.6%	50.0%
2023 May	CodeT5+ [WLG+23]	2B	52B	24.2%	-
2023 May	CodeT5+ [WLG+23]	16B	52B	30.9%	-
2023 May	InstructCodeT5+ [WLG+23]	16B	52B	35.0%	-
2023 Jun	WizardCoder [LXZ+23]	16B	1T	57.3%	51.8%
2023 Jun	**phi-1**	1.3B	7B	50.6%	55.5%

Style versus Substance

Is improved performance truly achieved using improved data quality?

What would happen if CodeExercises were not formatted as HumanEval problems? Would we still see the boost in performance? Possible contamination?



Contradiction with Scaling Laws & False Promises

Need for base model to be capable

Imitation data is not representative of the true distribution → Quickly hit a ceiling in performance gains

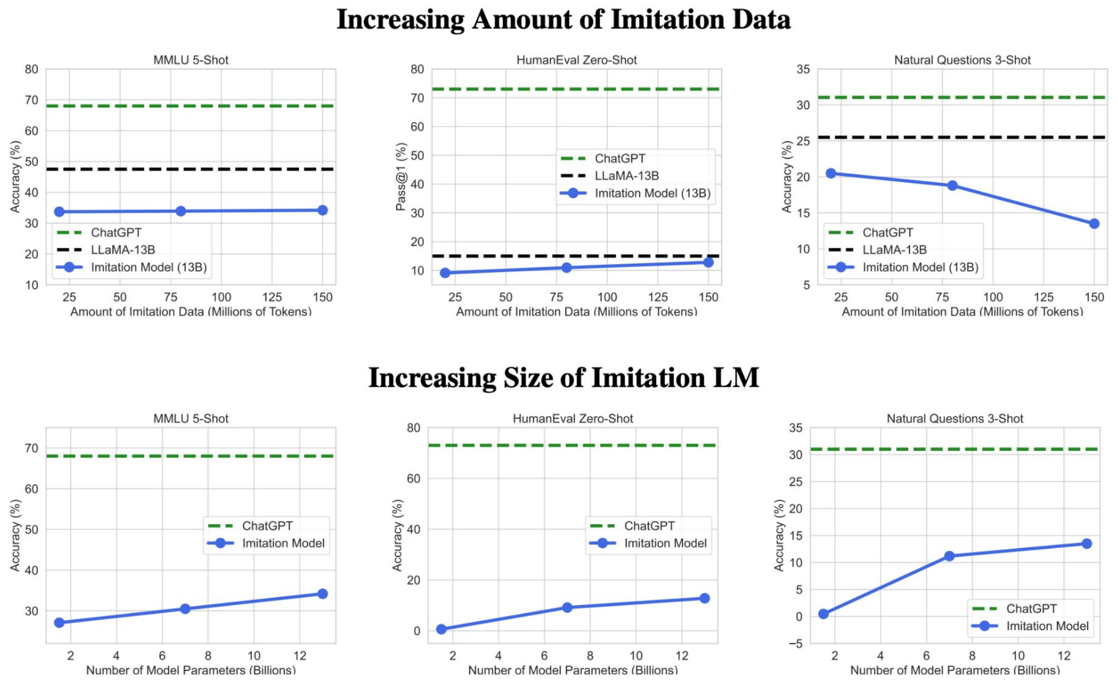


Figure 4: *Automatic evaluations.* As we increase the amount of imitation data, there is little improvement on various benchmarks, or even performance regressions (*top*). On the other hand, scaling up the base LM steadily improves results (*bottom*), suggesting that the key difference between open-source and closed-source LMs is a raw capabilities gap, rather than the finetuning data used.

Model Compression: Matching true distributions

Bucila et al. (Model Compression, SIGKDD 2006) demonstrated that distillation works best when the target task closely matches the original ensemble's capabilities.

They ensured identical synthetic data distributions…

synthetic data approaches attempt to compress capabilities across vastly different domains + scales, violating the fundamental assumptions of compression.

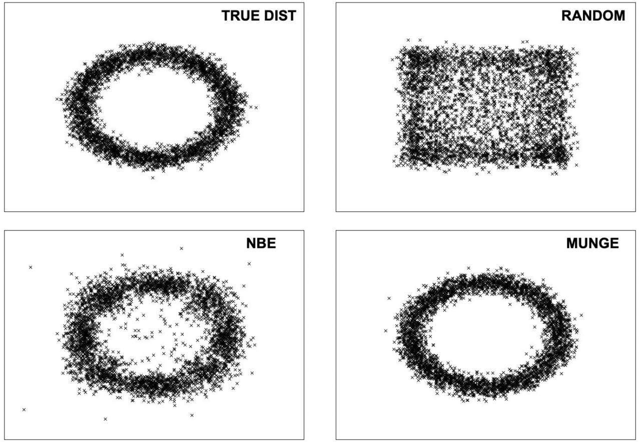


Figure 1: Synthetic data generated for a simple 2D problem.

LIMA: Less Is More for Alignment

Quality & diversity matter over quantity

Huanzhi Mao

9/16

Data Source

Source	#Examples	Avg Input Len.	Avg Output Len.
Training			
Stack Exchange (STEM)	200	117	523
Stack Exchange (Other)	200	119	530
wikiHow	200	12	1,811
Pushshift r/WritingPrompts	150	34	274
Natural Instructions	50	236	92
Paper Authors (Group A)	200	40	334
Dev			
Paper Authors (Group A)	50	36	N/A
Test			
Pushshift r/AskReddit	70	30	N/A
Paper Authors (Group B)	230	31	N/A

Table 1: Sources of training prompts (inputs) and responses (outputs), and test prompts. The total amount of training data is roughly 750,000 tokens, split over exactly 1,000 sequences.

Result

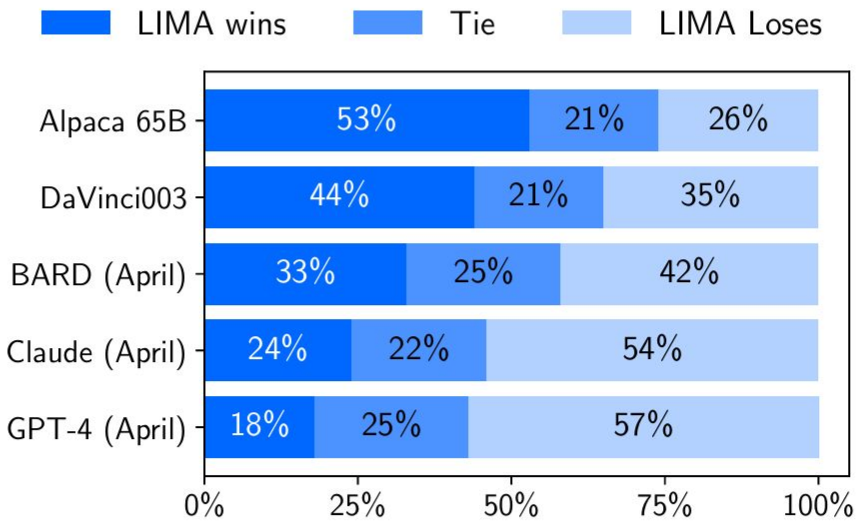


Figure 1: Human preference evaluation, comparing LIMA to 5 different baselines across 300 test prompts.

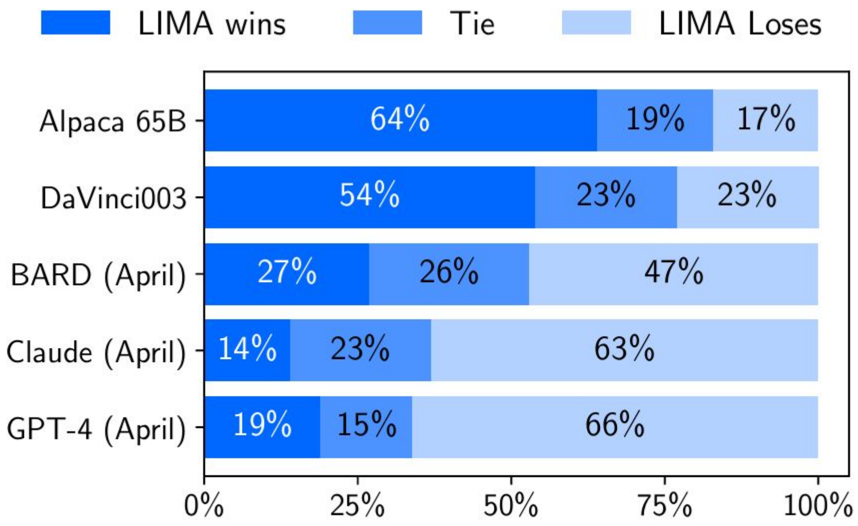


Figure 2: Preference evaluation using GPT-4 as the annotator, given the same instructions provided to humans.

Why is Less More?

Diversity/Quality

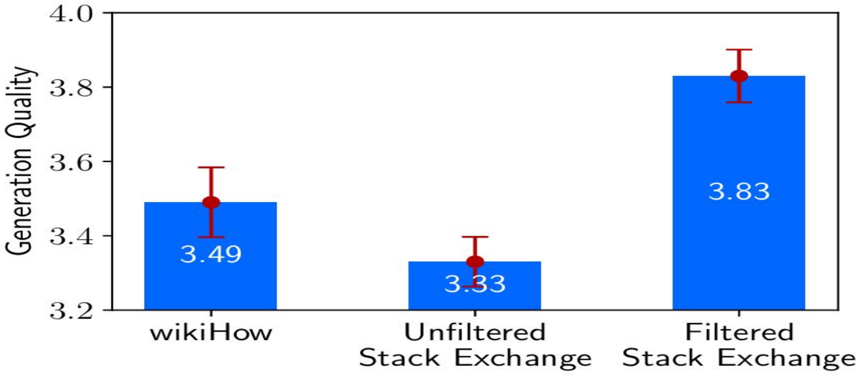


Figure 5: Performance of 7B models trained with 2,000 examples from different sources. **Filtered Stack Exchange** contains diverse prompts and high quality responses; **Unfiltered Stack Exchange** is diverse, but does not have any quality filters; **wikiHow** has high quality responses, but all of its prompts are "how to" questions.

Quantity

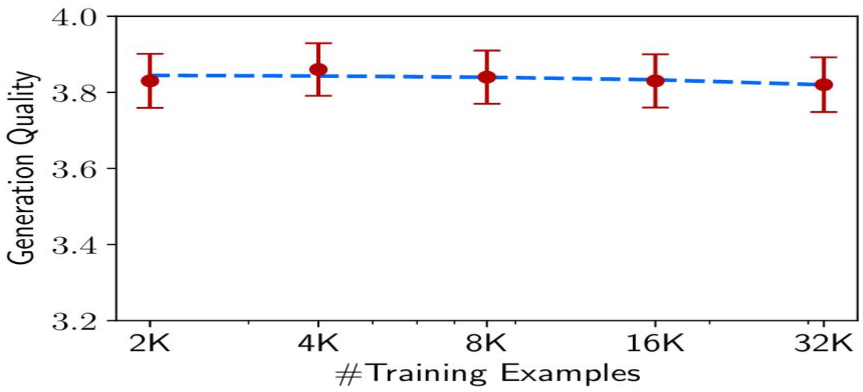


Figure 6: Performance of 7B models trained with exponentially increasing amounts of data, sampled from (quality-filtered) Stack Exchange. Despite an up to 16-fold increase in data size, performance as measured by ChatGPT plateaus.