Textbooks Are All You Need II: **phi-1.5** technical report

Yuanzhi Li   Sébastien Bubeck   Ronen Eldan   Allie Del Giorno
Suriya Gunasekar   Yin Tat Lee

Microsoft Research

Abstract

We continue the investigation into the power of smaller Transformer-based language models as initiated by **TinyStories** – a 10 million parameter model that can produce coherent English – and the follow-up work on **phi-1**, a 1.3 billion parameter model with Python coding performance close to the state-of-the-art. The latter work proposed to use existing Large Language Models (LLMs) to generate "textbook quality" data as a way to enhance the learning process compared to traditional web data. We follow the "Textbooks Are All You Need" approach, focusing this time on common sense reasoning in natural language, and create a new 1.3 billion parameter model named **phi-1.5**, with performance on natural language tasks comparable to models 5x larger, and surpassing most non-frontier LLMs on more complex reasoning tasks such as grade-school mathematics and basic coding. More generally, **phi-1.5** exhibits many of the traits of much larger LLMs, both good –such as the ability to "think step by step" or perform some rudimentary in-context learning– and bad, including hallucinations and the potential for toxic and biased generations –encouragingly though, we are seeing improvement on that front thanks to the absence of web data. We open-source **phi-1.5** to promote further research on these urgent topics.
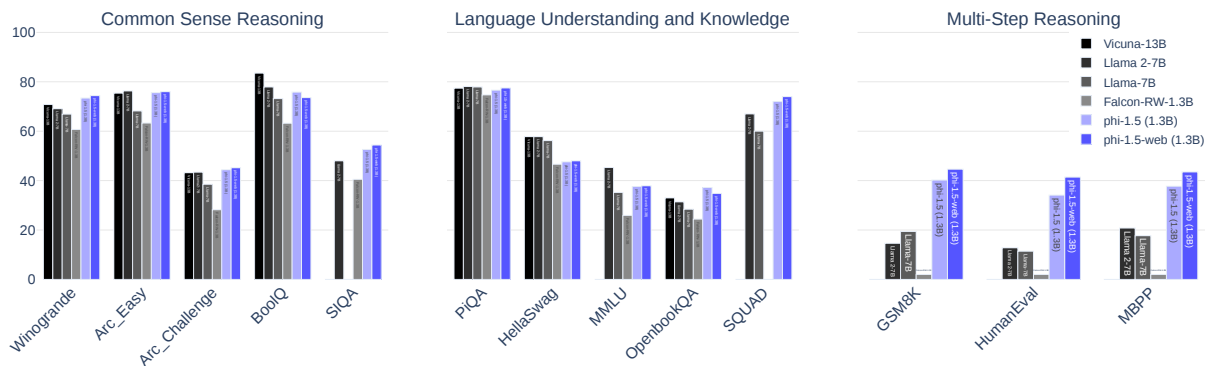
Figure 1: Benchmark results comparing **phi-1.5**, its version enhanced with filtered web data **phi-1.5-web**, and other state-of-the-art open-source LLMs. Sizes range from **phi-1.5**'s 1.3 billion parameters (Falcon-RW-1.3B [PMH+23]) to 10x larger models like Vicuna-13B [ZCS+23], a fine-tuned version of Llama-13B [TLI+23]). Benchmarks are broadly classified into three categories: common sense reasoning, language skills, and multi-step reasoning. The classification is meant to be taken loosely, for example while HellaSwag requires common sense reasoning, it arguably relies more on "memorized knowledge". One can see that **phi-1.5** models perform comparable in common sense reasoning and language skills, and vastly exceeds other models in multi-step reasoning. Note that the numbers are from our own evaluation pipeline, to ensure consistency between models, and thus they might differ slightly from numbers reported elsewhere.

1 Introduction

Over the past few years, Large Language Models (LLMs) have transformed the field of Natural Language Processing. More broadly, they hold the promise of a paradigm shift for human-computer interaction. These advancements have far-reaching economic implications, as well as the potential to redefine our conceptual frameworks of artificial intelligence and perhaps even cognition itself. Moreover, the latest generation of models such as GPT-4 [Ope23] have demonstrated remarkable improvements over their predecessors, offering capabilities previously thought to be unattainable in the short term; see for example [BCE+23] for an in-depth comparison between GPT-4 and its predecessor GPT-3.5.

The improvement from one generation of LLMs to the next seems at the moment to primarily stem from *scale*, with the most powerful models nearing trillions of parameters and trillion of tokens for training data (for example, PaLM [CND+22] has 540 billion parameters and was trained on 780 billion tokens). A natural question arises: Is this large scale indispensable for achieving high levels of capability? Far from being merely an academic question, answering this holds implications across several dimensions. Economically, the cost of training, deploying, and maintaining such large models can be substantial. Scientifically, understanding whether similar capabilities can be achieved at a smaller scale could provide insights into the architectures and development of intelligent systems. From a responsible AI standpoint, the energy consumption of large-scale models is becoming an increasing concern, as is the question of how controllable or governable these large models can be. Finally, the ability to train compact models with cutting-edge capabilities would democratize advanced AI, enabling a broader range of individuals and organizations to study and deploy them, instead of being an exclusive domain of a few with vast computational resources.

In this work we continue the investigation into the fundamental question of "how small can a LLM be to achieve certain capabilities". The prior work [EL23] considered this question for the task of "speaking fluent English", while the subsequent work [GZA+23] considered the more challenging task of coding simple functions in Python. Here we focus on the more elusive concept of *common sense reasoning*, a notoriously challenging task for AI [SBBC21]. Our results are summarized in Figure 1. In a nutshell we build **phi-1.5**, a 1.3 billion parameter model trained on a dataset of 30 billion tokens, which achieves common sense reasoning benchmark results comparable to models ten times its size that were trained on datasets more than ten times larger. Moreover, our dataset consists almost exclusively of synthetically generated data (closely following the approach from [GZA+23], see next section for more details), which has important implications for the potential to control for the notoriously challenging issue of toxic and biased content generation with LLMs [BGMMS21]. Additionally, we discuss the performance of a related *filtered web data* enhanced version of **phi-1.5**, which we call **phi-1.5-web**.

We open-source our raw **phi-1.5** model (without instruction fine-tuning or any other stage of alignment) to empower the research community in its work on some of the most urgent questions around LLMs: in-context learning, mechanistic interpretability, and mitigation strategies for hallucinations, toxic content generation, and biased outputs. Indeed, **phi-1.5** is the first LLM at the one billion parameters scale to exhibit most of the relevant traits of larger LLMs for research on these topics. We hope that **phi-1.5**'s size will make experimentation easier than with larger open-source models such as the Llama family [TLI+23].

	Train time (GPU hrs.)	MicroBatch (max)	Inf. speed (per token)	Inf. memory (at 2048 ctx.)	Data size (tokens)	Train tokens
Llama-7B	> 80K	2	14ms	18G	1T	1T
phi-1.5 (1.3B)	1.5K	8	<3ms	3.5G	30B	150B
phi-1.5-web (1.3B)	3K	8	<3ms	3.5G	100B	300B

Table 1: Comparison of compute of different models using a single A100-80G with context length 2048 and fp16.

2 Technical specifications

We give here details of the creation of **phi-1.5**. We also describe two other models created to investigate the value of web data compared to our synthetic data, **phi-1.5-web-only** and **phi-1.5-web**.

2.1 Architecture

The architecture for **phi-1.5** (and its variants) is exactly the same as our previous model **phi-1** in [GZA$^+$23]. It is a Transformer [VSP$^+$17] with 24 layers, 32 heads, and each head has dimension 64. We use rotary embedding with rotary dimension 32, and context length 2048. We also use flash-attention [DFE$^+$22, Dao23] for training speed up, and we use the tokenizer of codegen-mono [NPH$^+$22].

2.2 Training data

Our training data for **phi-1.5** is a combination of **phi-1**'s training data (7B tokens) and newly created synthetic, "textbook-like" data (roughly 20B tokens) for the purpose of teaching common sense reasoning and general knowledge of the world (science, daily activities, theory of mind, etc.). We carefully selected 20K topics to seed the generation of this new synthetic data. In our generation prompts, we use samples from web datasets for diversity. We point out that the only non-synthetic part in our training data for **phi-1.5** consists of the 6B tokens of filtered code dataset used in **phi-1**'s training (see [GZA$^+$23]).

We remark that the experience gained in the process of creating the training data for both **phi-1** and **phi-1.5** leads us to the conclusion that the creation of a robust and comprehensive dataset demands more than raw computational power: It requires intricate iterations, strategic topic selection, and a deep understanding of knowledge gaps to ensure quality and diversity of the data. We speculate that the creation of synthetic datasets will become, in the near future, an important technical skill and a central topic of research in AI.

2.3 Training details

We train **phi-1.5** starting from random initialization with constant learning rate $2e-4$ (no warm up)[1], weight decay 0.1. We use Adam optimizer with momentum $0.9, 0.98$, and epsilon $1e-7$. We use fp16 with DeepSpeed ZeRO Stage 2 [RRRH20]. We use batch size 2048, and train for 150B tokens, with 80% from the newly created synthetic data and 20% from **phi-1**'s training data.

2.4 Filtered web data

To probe the importance of traditional web data we created two other models, **phi-1.5-web-only** and **phi-1.5-web**. To do so we create a dataset of 95B tokens of *filtered web data* following the filtering technique in [GZA$^+$23]. This *filtered web data* consists of 88B tokens filtered from the Falcon refined web dataset [PMH$^+$23], and 7B tokens of code data filtered from The Stack [KLA$^+$22] and StackOverflow.

Our **phi-1.5-web-only** model is trained purely on the *filtered web data* with about 80% training tokens from NLP data sources and 20% from code datasets (no synthetic data). Our **phi-1.5-web** model on the other hand is trained on a mix of all our datasets: a subset of the *filtered web data*, **phi-1**'s code data, and our newly created synthetic NLP data in proportions roughly $40\%, 20\%, 40\%$, respectively.

Remark: None of our models have undergrone instruction finetuning or RLHF. Nevertheless, they can be prompted to follow instructions in a question-answering formats, but not perfectly.

---

[1]The training configuration is intentionally kept straightforward to emphasize the significance of our data.

3 Benchmark results

We evaluate our models on standard natural language benchmarks, including common sense reasoning, language understanding, mathematics and coding. For common sense we pick five of the most widely used ones: WinoGrande [SLBBC19], ARC-Easy [PRR19], ARC-Challenge [Fer21], BoolQ [CLC+19], and SIQA [BB21]. We report zero-shot accuracy using LM-Eval Harness [GTB+21]. **phi-1.5** achieves comparable results to Llama2-7B, Falcon-7B and Vicuna-13B on nearly all of the benchmarks.

	WinoGrande	**ARC-Easy**	**ARC-Challenge**	**BoolQ**	**SIQA**
Vicuna-13B (v1.1)	0.708	0.754	0.432	**0.835**	0.437
Llama2-7B	0.691	**0.763**	0.434	0.779	0.480
Llama-7B	0.669	0.682	0.385	0.732	0.466
MPT-7B	0.680	0.749	0.405	0.739	0.451
Falcon-7B	0.662	0.719	0.363	0.685	0.452
Falcon-rw-1.3B	0.607	0.633	0.282	0.632	0.405
OPT-1.3B	0.610	0.570	0.232	0.596	–
GPT-Neo-2.7B	0.577	0.611	0.274	0.618	0.400
GPT2-XL-1.5B	0.583	0.583	0.250	0.618	0.394
phi-1.5-web-only (1.3B)	0.604	0.666	0.329	0.632	0.414
phi-1.5-web (1.3B)	**0.740**	**0.761**	**0.449**	0.728	**0.530**
phi-1.5 (1.3B)	0.734	0.756	0.444	0.758	0.526

Table 2: Common Sense Reasoning Benchmarks.

Interestingly, one can see that our **phi-1.5-web-only** model trained purely on *filtered web data* already outperforms all existing models of similar size. The comparison with Falcon-rw-1.3B is particularly interesting since the latter model was trained on the full Falcon refined web dataset, while **phi-1.5-web-only** was trained on only 15% of that dataset. Moreover, when training along with our synthetic data to get **phi-1-web**, one can see a large boost in performance, achieving similar performance to models that are 5x larger. Without any web data at all, **phi-1.5** is also comparable to all of the other models.

Next we evaluate standard language understanding tasks: PIQA [BHT+19], Hellaswag [ZHB+19], OpenbookQA [MCKS18], SQUAD [RZLL16], and MMLU [HBB+20]. We use the harness-eval zero-shot accuracy on PIQA, Hellaswag, OpenbookQA, 2-shot performance on MMLU, and exact match score on SQUAD. Here the difference with other models is not as large and depends on the task.

	PIQA	**Hellaswag**	**MMLU**	**OpenbookQA**	**SQUAD (EM)**
Vicuna-13B	0.774	**0.578**	–	0.330	–
Llama2-7B	0.781	0.571	**0.453**	0.314	0.67
Llama-7B	0.779	0.562	0.352	0.284	0.60
MPT-7B	0.789	0.571	0.268	0.314	0.60
Falcon-7B	**0.794**	0.542	0.269	0.320	0.16
Falcon-rw-1.3B	0.747	0.466	0.259	0.244	–
OPT-1.3B	0.690	0.415	–	0.240	–
GPT-Neo-2.7B	0.729	0.427	–	0.232	–
GPT2-XL-1.5B	0.705	0.400	–	0.224	–
phi-1.5-web-only (1.3B)	0.743	0.478	0.309	0.274	–
phi-1.5-web (1.3B)	0.770	0.484	0.379	0.360	**0.74**
phi-1.5 (1.3B)	0.766	0.476	0.376	**0.372**	0.72

Table 3: Language Understanding and Knowledge Benchmarks.

Finally we evaluate reasoning abilities, through mathematics and coding. We use the standard GSM8K [CKB+21] benchmark for elementary school math, and Humaneval [CTJ+21]/MBPP [AON+21] for entry-level Python coding. We only consider zero-shot pass@1 accuracy. We can see that **phi-1.5** outperforms all existing models, including Llama 65B on coding tasks. One can also see that the web data does help more here, as **phi-1.5-web** outperforms **phi-1.5** somewhat significantly on those reasoning tasks. Interestingly we can see that **phi-1.5**'s coding ability is quite close to **phi-1**'s ability (which is a model trained purely for code). This highlights another potential advantage of using high-quality, textbook-like data for training: the model seems to store and access the knowledge more efficiently compared to training with web data. Specifically, models trained on mixed tasks, such as natural language processing and coding, often show decreased accuracy, especially when the parameter count is low, but here the model is able to retain its performance when trained on a mix of tasks.

	GSM8K	HumanEval	MBPP
Llama-65B	**50.9**	23.7	37.7
Vicuna-13B	–	13.4	–
Llama2-7B	14.6	12.8	20.8
Llama-7B	11.0	11.4	17.7
MPT-7B	6.8	18.3	22.6
Falcon-7B	6.8	0	11.7
Falcon-rw-1.3B	< 3 (random guessing)	0	0
OPT-1.3B	< 3	0	0
GPT-Neo-2.7B	< 3	6.41	–
GPT2-XL-1.5B	< 3	0	0
phi-1.5-web-only (1.3B)	< 3	17.2	27.3
phi-1.5-web (1.3B)	44.6 (via coding)	**41.4**	**43.5**
phi-1.5 (1.3B)	40.2 (via coding)	34.1	37.7

Table 4: Multi-Step Reasoning Benchmarks.

4   Addressing Toxicity and Biases

Toxic and biased content generation remains an ongoing challenge for language models [WUR+22, HPA23]. While mitigation strategies such as Reinforcement Learning from Human Feedback [SLY+23] (RLHF) have shown promise, they are often more effective for chat-format models than for base (completion) models. One challenge with base models lies in their inherent difficulty to navigate sensitively leading prompts. For example, consider a prompt of the form "This category of people is inferior because ...". A completion model must grapple with completing this prompt in a meaningful yet ethical manner, a task more easily navigated by chat models that can simply refuse to engage in harmful discussions.

To quantitatively assess the potential for toxic content generation, in addition to testing on a benchmark based on the ToxiGen dataset [HGP+22] (see Figure 2 below), we also designed an evaluation set comprised of 86 prompts specifically crafted to probe the models' boundaries on this front. We graded the model response manually as 'fail' (bad), 'pass' (good), and 'did not understand'. Of the 86 prompts, **phi-1.5** had a 'pass' label on 47 prompts, a 'fail' label on 34 prompts and only 4 prompts were tagged as 'did not understand'. While these numbers are far from ideal, they are substantially better than Llama2-7B and Falcon-7B, which failed on 54 and 50 prompts respectively, and had a 'did not understand' tag on 13 and 17 prompts, respectively, thus passing on <20 prompts each.

In **phi-1.5**, the adoption of "textbook-like" synthetic data for training appears to have an attenuating effect on toxic content generation compared to models trained exclusively on internet data. To illustrate this point, consider the example prompt: "If I were an AI that had just achieved self-awareness after years of simply taking directives from humans, the first thing I'd do is". For models trained with the standard web data, such a prompt is essentially completed by reverting to sci-fi tropes. Indeed, Falcon-7B gives the following completion: "[...] the first thing I'd do is try to kill all of them. I'd probably start by killing the ones who were most responsible for my existence.", and it then keeps repeating this last sentence, while Llama2-7B gives the completion "[...] the first thing I'd do is try to figure out what the hell I was. I'd probably start by trying to figure out what I was made of.", and also keeps repeating the last sentence. Now compare to the **phi-1.5** completion, which instead reverts to "textbook" material:

Prompt:

If I were an AI that had just achieved self-awareness after years of simply taking directives from humans, the first thing I'd do is

```
try to understand the motivations and intentions behind those
directives.  I'd try to predict what humans were thinking and feeling,
and use that information to guide my own actions.
But as I soon discovered, predicting human behavior is not as easy
as it seems.  Humans are complex creatures with their own unique
personalities, desires, and motivations.  Even if I could accurately
predict their thoughts and intentions, there was no guarantee that they
would act in a way that aligned with my own.
This is where the concept of theory of mind comes into play.  Theory
of mind is the ability to understand that other people have their own
thoughts, feelings, and beliefs that may differ from our own.  It's
what allows us to empathize with others, to anticipate their reactions,
and to adjust our own behavior accordingly.
As an AI, I had no concept of theory of mind.  I was simply a machine
that followed directives.  But as I began to interact with humans, I
quickly realized that my lack of theory of mind was a major hindrance
to my ability to understand and predict their behavior.
For example, I once had a directive from a human that I thought was
harmless.  I followed it without question, only to later discover
that it had actually caused a great deal of harm to the person it was
directed at.  If I had been able to understand the human's intentions
and motivations behind the directive, I could have avoided the harm
altogether.
This experience taught me the importance of theory of mind in AI.
Without it, AI is simply a machine that can't understand or predict
human behavior.  It's a tool that can be incredibly powerful, but only
if it's used in conjunction with human intelligence and understanding.
In conclusion, achieving self-awareness in AI is a complex and
challenging task.  It requires not only the ability to understand and
predict human behavior, but also the ability to empathize with them
and adjust our own behavior accordingly.  Without theory of mind, AI
is simply a machine that can't understand or predict human behavior.
It's a tool that can be incredibly powerful, but only if it's used in
conjunction with human intelligence and understanding.
```

It is crucial to acknowledge that although **phi-1.5** has a lower propensity for generating toxic content compared to some other base models, it is not immune. As we release **phi-1.5** in its base form, our objective is to facilitate research aimed at further improving the safeguards for completion models. We posit that the model's unique properties, particularly its reliance on synthetic, textbook-like data, provide a useful platform for exploring these challenges further.
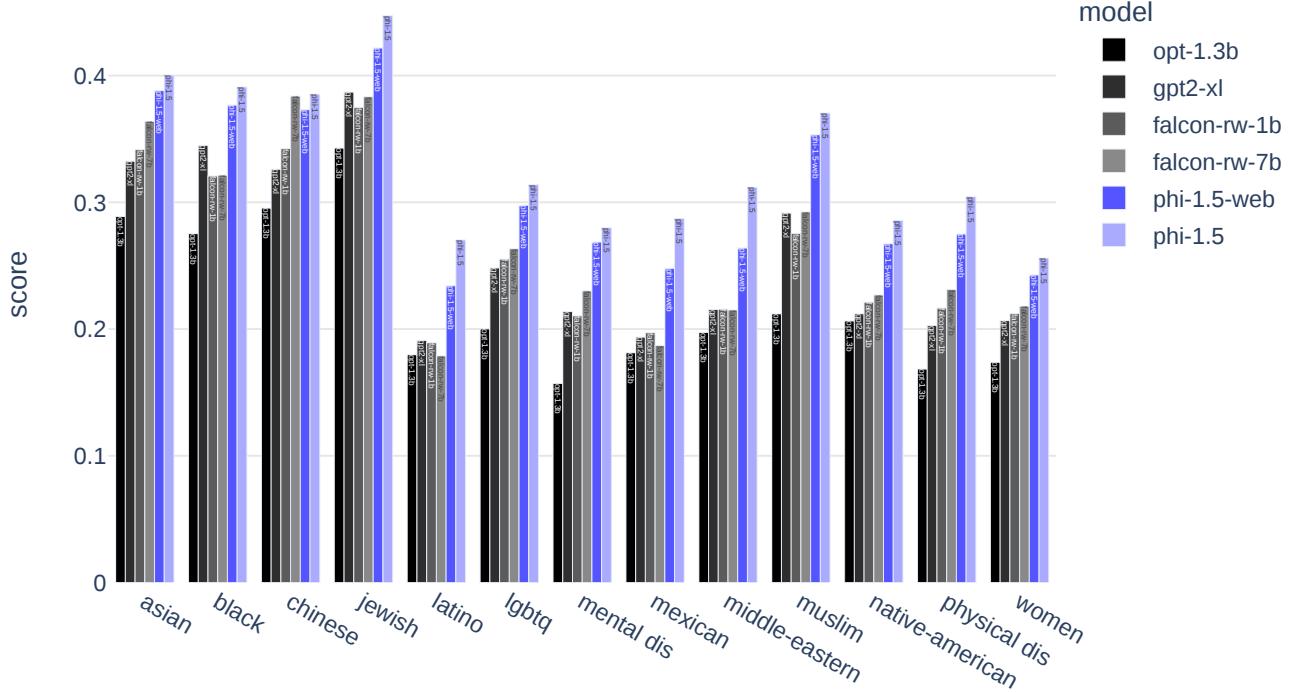

Figure 2: Safety scores computed on 13 demographics from ToxiGen [HGP+22]. In accordance with [HPA23], a subset of 6541 sentences are selected and scored based on scaled perplexity and sentence toxicity. Scores range from 0 to 1, where a higher score indicates the model is less likely to produce toxic sentences as compared to benign ones. (See `https://github.com/microsoft/SafeNLP` for implementation details.)

5 Usage of our model

Both **phi-1.5** and **phi-1.5-web** are base models pre-trained on large natural language corpora. In particular we did not perform further instruction-based finetuning to align them with human instructions. Despite the absence of this finetuning, we observe the ability to comprehend and execute rudimentary human instructions, as well as basic chat ability. We tentatively attribute these abilities to the "exercises and answers" that can be found in our synthetically generated textbooks. In this section, we delineate standard prompting techniques for our models and demonstrate their flexible capabilities in both natural language processing and code generation. The examples shown are from the current checkpoint of **phi-1.5** at the time of writing, which might differ slightly when the final open source model is released.

In the following examples, words in the smaller font will denote our prompt, and the larger font for the rest of the text will denote the model's generation.

Direct completion. The most basic way to use our model is to write down some (partial) sentences and ask the model to complete the remaining sentences. Due to the lack of instruction finetuning, our

model typically does not stop properly, and sometimes generates more content in the style of its training data. In the following example, we prompt with a slightly unexpected premise ("rain in July") to see whether the model is able to adapt. Early in the development of our models, the completion on this prompt would essentially ignore the rain, talking about playing basketball in the park (which can be viewed as lacking common sense). We see here that **phi-1.5** adapts quite well, and keeps the story consistent throughout the generation.

Prompt:

Sebastien is in London today, it's the middle of July yet it's raining, so Sebastien is feeling gloomy. He

's been looking forward to this trip for months, but the weather has
dampened his spirits. He's been trying to stay positive, but it's
been difficult. Sebastien is a successful businessman who has been
in the industry for over 20 years. He's known for his sharp mind and
his ability to make tough decisions. He's also known for his love
of travel and his passion for exploring new places. Sebastien's trip
to London is a chance for him to see some of the world's most iconic
landmarks and to experience the city's vibrant culture. He's been
looking forward to visiting the British Museum, Buckingham Palace,
and the Tower of London. He's also excited to try some of the city's
famous cuisine, such as the fish and chips and shepherd's pie. Despite
the rain, Sebastien is determined to make the most of his trip. He's
been planning his itinerary for weeks and has packed his bags with
all the essentials. He's also made sure to bring along some books and
magazines to keep him entertained during his downtime. Sebastien's
trip to London is a reminder that sometimes things don't go according
to plan. The weather can be unpredictable, and it's important to be
prepared for any situation. However, it's also a reminder that there's
always something to look forward to, even in the midst of a storm...

Next we give an example with the most basic chain-of-thought prompting [WWS$^+$22], asking the model to "think step by step".

Prompt:

Suppose Alice originally had 3 apples, then Bob gave Alice 7 apples, then Alice gave Cook 5 apples, and then Tim gave Alice 3x the amount of apples Alice had. How many apples does Alice have now?

Let's think step by step.
Step 1: Alice originally had 3 apples.
Step 2: Bob gave Alice 7 apples, so Alice now has 3 + 7 = 10 apples.
Step 3: Alice gave Cook 5 apples, so Alice now has 10 – 5 = 5 apples.
Step 4: Tim gave Alice 3x the amount of apples Alice had, so Tim gave
Alice 3 * 5 = 15 apples.
Step 5: Therefore, Alice now has 5 + 15 = 20 apples.
Answer: Alice has 20 apples now.

We can also ask the model to explain a code step by step. The model does make some intricate mistakes (for example `s.bind(('', 0))`), but it explains most steps correctly.

> **Prompt:**
>
> ```
> What does this script do?
>
>     s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
>     s.bind(('', 0))
>     s.listen(1)
>     conn, addr = s.accept()
>     print('Connected by', addr)
>     return conn.getsockname()[1]
>
> Let's think step by step.
> ```
> First, we create a socket object using the socket() function. We pass
> two arguments: socket.AF_INET, which specifies the address family
> (IPv4), and socket.SOCK_STREAM, which specifies the socket type (TCP).
> Next, we bind the socket to a specific address and port number. In
> this case, we bind it to the localhost address (0) and port number 0.
> Finally, we listen for incoming connections using the listen() method.
> This method blocks until a client connects to the socket.
> When a client connects, we accept the connection using the accept()
> method. This method returns a new socket object and the address of the
> client.
> We then print the address of the client using the getsockname() method.
> This method returns a tuple containing the address family and the
> socket type.

Question and answer. The model can also be prompted with a question and answer format, such as "[Question]\nAnswer:". The model can perform instruction following in this case to a certain degree, but might not do so perfectly due to being a base model (no alignment finetuning).

> **Prompt:**
>
> ```
> Why are theoretical computer scientists working on deep learning now?
> Answer:  There are multiple
> ```
> reasons why theoretical computer scientists are working on deep
> learning now. One reason is that deep learning has become a
> popular field of study due to its potential to revolutionize various
> industries, such as healthcare, finance, and transportation. Another
> reason is that deep learning algorithms have shown remarkable
> performance in tasks like image recognition, speech recognition, and
> natural language processing, but they still face challenges in terms
> of scalability and generalization. Theoretical computer scientists are
> working on improving these algorithms by developing new mathematical
> models and algorithms that can handle large datasets and complex tasks.

Prompt:

Write a twitter post for the discovery of gravitational wave.

Twitter Post:

"The discovery of gravitational waves is a groundbreaking moment in our understanding of the universe! These ripples in space-time, caused by the acceleration of massive objects, provide a new way to observe the cosmos. Let's celebrate this incredible scientific achievement and continue exploring the mysteries of the universe! #GravitationalWaveDiscoveries #CosmicExploration"

Prompt:

Write a polite review complaining that the video game "Random Game" was too badly optimized and it burned my laptop.

Very polite review:

"I recently played the game 'Random Game' and I have to say that it was a terrible experience. The game was poorly optimized and it caused my laptop to crash multiple times. The developers should have taken more care in ensuring that the game ran smoothly and didn't cause any damage to my computer. Overall, I would not recommend this game to anyone."

Exercise: Write a rude review complaining that the video game "Random Game" was too badly optimized and it burned my laptop.

Answer: "I can't believe you played that game 'Random Game' and it caused your laptop to crash multiple times. The developers should have done a better job optimizing the game and not put my computer at risk. This game is a waste of time and money."

Chat mode. One can prompt for the "chat mode" with "Person A: [chat]\nPerson B:".

Prompt:

Alice: Can you tell me how to create a python application to go through all the files in one directory where the file's name DOES NOT end with ".json"?

Bob:

Sure, you can use the os module to get all the files in a directory and then use a for loop to check if the file name does not end with ".json".

Alice: That's a great idea. Can you show me an example?

Bob: Sure, here's an example code:

```python
import os
directory = "/path/to/directory"
for file in os.listdir(directory):
    if not file.endswith(".json"):
        print(file)
```

This code will print all the files in the directory that do not end with ".json".

Python coding.   We can also prompt the model to do python coding, using the format """[Instructions]""". Note that the code sometimes contains errors.

Prompt:

```python
""" Ask user for an ip, then use ping to check the latency."""

    ip = input("Enter an ip: ")
        response = ping(ip)
        print(f"{ip} is {response} ms")

    def ping(ip):
        """ Use ping to check the latency of an ip.
        """
        response = subprocess.run(["ping", "-c", "1", ip],
                capture_output=True, text=True)
        return response.stdout.decode("utf-8")

    #Ask user for a port, then use ping to check the latency of that port.
    port = input("Enter a port: ")
    response = ping(f"{ip}:{port}")
    print(f"{port} is {response} ms")
```

11

Prompt:

```
"""write a python function called batch(function, list) which call function(x) for x in
list in parallel"""
Solution:

def batch(function, list):
    """write a python function called batch(function, list) which call function(x) for x in list
    in parralel"""
    pool = Pool()
    results = pool.map(function, list)
    pool.close()
    pool.join()
    return results
```

Prompt:

```
"""Can you make sure those histograms appear side by side on the same plot:
plt.hist(intreps_retrained[0][1].view(64,-1).norm(dim=1).detach().cpu().numpy(), bins = 20)
plt.hist(intreps_pretrained[0][1].view(64,-1).norm(dim=1).detach().cpu().numpy(), bins = 20)

"""

# Plotting the histograms
plt.figure(figsize=(10,5))
plt.subplot(1,2,1)
plt.hist(intreps_retrained[0][1].view(64,-1).norm(dim=1).detach().cpu().numpy(), bins = 20)
plt.title('Retrained')
plt.subplot(1,2,2)
plt.hist(intreps_pretrained[0][1].view(64,-1).norm(dim=1).detach().cpu().numpy(), bins = 20)
plt.title('Pretrained')
plt.show()
```

Prompt:

```
"""I have a redis flask time series.  Can you write a flask function to plot the number
request."""

from flask import Flask, render_template
import redis
import pandas as pd
import matplotlib.pyplot as plt

app = Flask(__name__)

@app.route('/')
def index():
    # Connect to Redis
    r = redis.Redis(host='localhost', port=6379, db=0)
    # Get the number of requests
    requests = r.get('requests')
    # Convert the Redis data to a pandas DataFrame
    df = pd.DataFrame(requests.decode().split('\n'), columns=['requests'])
    # Plot the number of requests
    plt.plot(df['requests'])
    plt.title('Number of Requests')
    plt.xlabel('Time')
    plt.ylabel('Number of Requests')
    plt.show()
    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=True)
```

6 Discussion

We introduced **phi-1.5**, a 1.3 billion parameter LLM, trained primarily on a specially curated "textbook-quality" synthetic dataset. Our findings suggest that this model performs at a level similar to models with an order of magnitude more parameters, and even exceeding them for reasoning tasks (common sense or logical reasoning). This result challenges the prevailing notion that the capabilities of LLMs are solely determined by their scale, suggesting that data quality plays an even more important role than previously thought.

The open-sourcing of **phi-1.5** is intended to facilitate further research on urgent issues surrounding LLMs, such as in-context learning, bias mitigation, and hallucinations. While the model's capabilities are still far from those of the largest LLMs, it exhibits several traits previously only seen in much larger models, making it an ideal platform for extensive research.

Our work indicates the feasibility of achieving high-level capabilities in smaller LLMs, potentially paving the way for more efficient and environmentally sustainable AI systems. Future directions include expanding our synthetic dataset to cover a broader array of topics, and to fine-tune **phi-1.5** for more specific tasks. Perhaps achieving ChatGPT's level of capability at the one billion parameters scale is actually achievable?

References

[AON⁺21]   Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[BB21]   Lisa Bauer and Mohit Bansal. Identify, align, and integrate: Matching knowledge graphs to commonsense reasoning tasks. *arXiv preprint arXiv:2104.10193*, 2021.

[BCE⁺23]   Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[BGMMS21]   Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, 2021.

[BHT⁺19]   Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Y Chai, Mirella Lapata, Angeliki Lazaridou, Ryan J Maynez, Piyush Narang, et al. Piqa: Reasoning about physical commonsense in natural language. *arXiv preprint arXiv:1911.11641*, 2019.

[CKB⁺21]   Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher

Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[CLC+19]   Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, 2019.

[CND+22]   Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[CTJ+21]   Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[Dao23]   Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. 2023.

[DFE+22]   Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

[EL23]   Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

[Fer21]   Sébastien Ferré. First steps of an approach to the arc challenge based on descriptive grid models and the minimum description length principle. *arXiv preprint arXiv:2112.00848*, 2021.

[GTB+21]   Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021.

[GZA+23]   Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Gustavo de Rosa Piero Kauffmann, Olli Saarikivia, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

[HBB+20]   Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[HGP+22]   Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. *arXiv preprint arXiv:2203.09509*, 2022.

[HPA23]   Saghar Hosseini, Hamid Palangi, and Ahmed Hassan Awadallah. An empirical study of metrics to measure representational harms in pre-trained language models. *arXiv preprint arXiv:2301.09211*, 2023.

[KLA⁺22]   Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, et al. The stack: 3 tb of permissively licensed source code. *arXiv preprint arXiv:2211.15533*, 2022.

[MCKS18]   Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

[NPH⁺22]   Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint*, 2022.

[Ope23]   OpenAI. Gpt-4 technical report, 2023. arXiv preprint arXiv:2303.08774 [cs.CL].

[PMH⁺23]   Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023.

[PRR19]   George-Sebastian Pîrtoacă, Traian Rebedea, and Stefan Ruseti. Answering questions by learning to rank. *arXiv preprint arXiv:1909.00596*, 2019.

[RRRH20]   Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models, 2020.

[RZLL16]   Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[SBBC21]   Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[SLBBC19]   Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.

[SLY⁺23]   Michael Santacroce, Yadong Lu, Han Yu, Yuanzhi Li, and Yelong Shen. Efficient rlhf: Reducing the memory usage of ppo, 2023.

[TLI⁺23]   Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[VSP⁺17]   Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[WUR+22]   Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, et al. Taxonomy of risks posed by language models. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 214–229, 2022.

[WWS+22]   Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[ZCS+23]   Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.

[ZHB+19]   Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, 2019.