The Surprising Power of Small Language Models

Mojan Javaheripi
Microsoft Research

Joint work with
Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, Suriya Gunasekar,, Piero Kauffmann, Yin Tat Lee, Yuanzhi Li, Anh Nguyen, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Michael Santacroce, Anh Harkirat Singh Behl, Adam Kalai, Xin Wang, Rachel Ward, Philipp Witte, Cyril Zhang, Yi Zhang

Moore's Law of Language Models

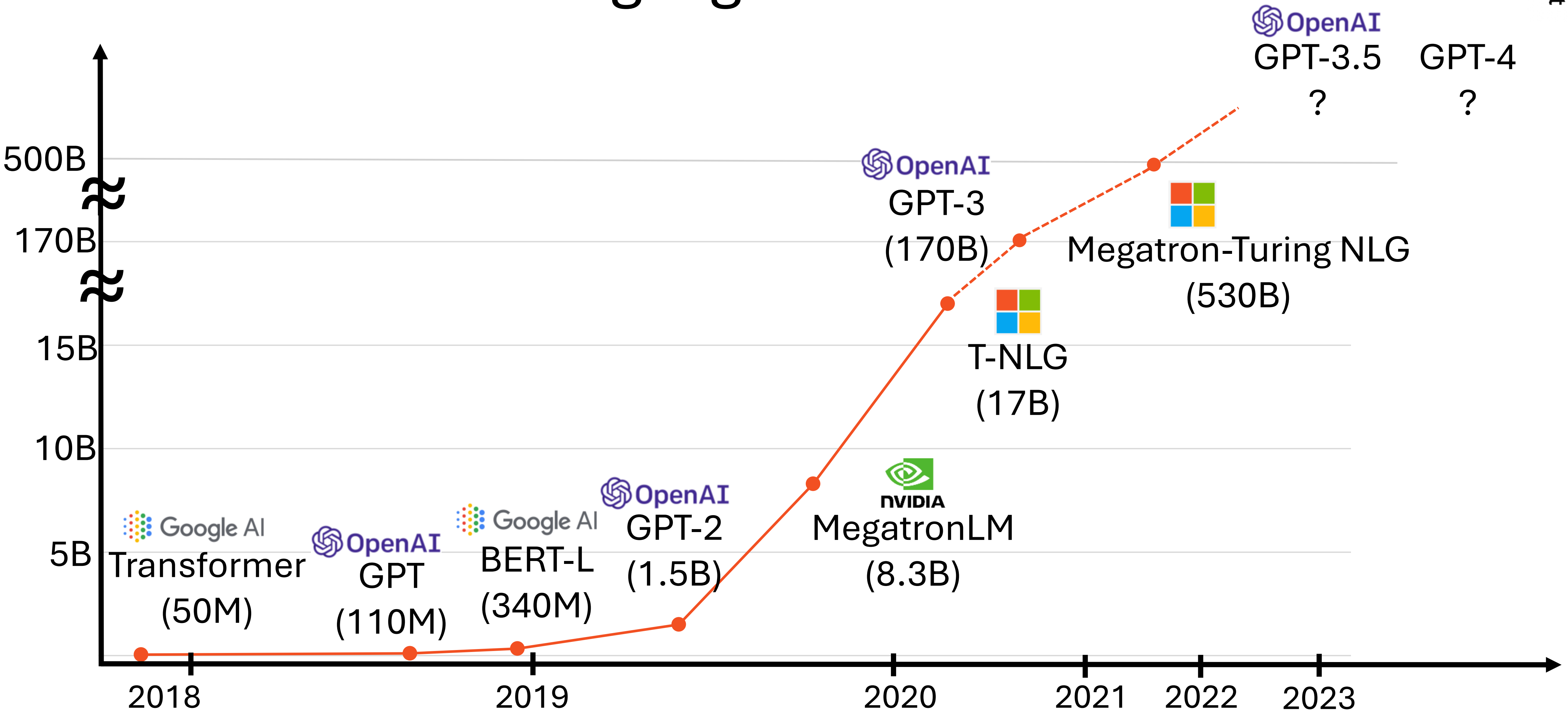


Figure inspired by Microsoft Research Blog: https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/

The power of Large Language Models

You
how can I save the logs of my code in a directory with date and time as the name in python

ChatGPT
You can save the logs of your code in a directory with the current date and time as the name using the Python `logging` module and the `datetime` module. Here's an example:

python Copy code
```python
import logging
import datetime
import os

# create a logger object
logger = logging.getLogger()
logger.setLevel(logging.DEBUG)

# create a file handler object
current_time = datetime.datetime.now().strftime("%Y-%m-%d-%H-%M-%S")
log_file_name = f"log_{current_time}.txt"
file_handler = logging.FileHandler(os.path.join("logs", log_file_name))
```

You
Let's play tic-tac-toe.

ChatGPT
Absolutely! I'll be "X" and you can be "O." Here's the initial game board:

markdown Copy code
```markdown
1 | 2 | 3
---------
4 | 5 | 6
---------
7 | 8 | 9
```

You
Can you write a proof of infinitude of primes, with every line that rhymes?

ChatGPT
In the realm of numbers, let's dance a prime tune,
A proof of infinitude that'll make your mind swoon.

Assume finitude, a limited prime parade,
Let's call them P1, P2, and let's not be afraid.

Multiply them all, then add one with glee,
A new number emerges, a mystery to see.

Now, this new number, let's call it N,
It's not divisible by primes, that's where we begin.

If N is prime, then our dance is complete,
A new prime emerges, our assumption's defeat.

If N's not prime, a factor we find,
A prime not in our list, a new kind.

This new prime divides N, but where does it stand?
If it's in our list, it's a contradiction so grand.

For it divides N, yet isn't a factor,
A paradox, a conundrum, a mathematical actor.

So either way, our assumption's a mess,
Prime infinity prevails, no need to guess.

In both cases, a prime outside the list,
Infinitude confirmed, our dance persists.

Thus, the proof rhymes, as numbers entwine,
An infinite prime ballet, a mathematical design.

You
Can you draw a unicorn in tikz?

ChatGPT
Certainly! Below is a simple example of a unicorn drawn using TikZ in LaTeX. You can use this code in a LaTeX document to generate the unicorn:

latex Copy code
```latex
\documentclass[tikz, border=5mm]{standalone}

\begin{document}

\begin{tikzpicture}[line cap=round, line join=round]

% Body
\draw[fill=blue!20] (2,0) ellipse (1.5 and 1);

% Head
\draw[fill=blue!20] (0.8,2) circle (1);

% Horn
\draw[fill=yellow] (0.8,3.5) -- (0.6,4) -- (1,4) -- cycle;

% Eye
\draw[fill=black] (1.5,2.3) circle (0.1);

% Mane
\draw[fill=purple!60] (0.4,2.5) -- (0.8,3.5) -- (1.2,2.5) -- cycle;
\draw[fill=purple!60] (0.5,2.7) -- (0.8,3.7) -- (1.1,2.7) -- cycle;
\draw[fill=purple!60] (0.6,2.9) -- (0.8,3.9) -- (1,2.9) -- cycle;

% Tail
```

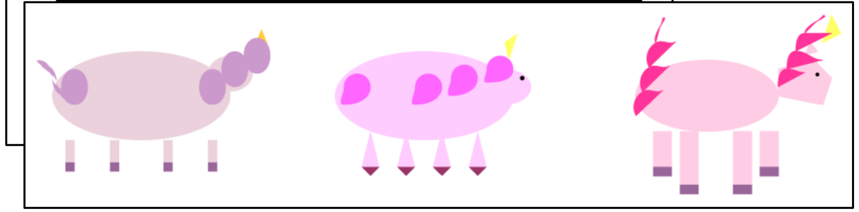


Figure from "Sparks of Artificial General Intelligence: Early experiments with GPT-4", *arXiv preprint arXiv:2303.12712* (2023).

The surprising Power of Small Language Models

- Can these emergent abilities be achieved at a smaller scale?
- Our line of work with the ***Phi*** models aims to answer this question
  - SLMs that achieve on par performance with models of much higher scale

The surprising Power of Small Language Models

- Can these emergent abilities be achieved at a smaller scale?
- Our line of work with the ***Phi*** models aims to answer this question
  - SLMs that achieve on par performance with models of much higher scale

***phi-1* (1.3B)**

microsoft/**phi-1**

Python **coding** model with perf. comparable to models 10x larger trained on 100x more data

- *Specialist* SLMs are possible
- What about a *general* model?

The surprising Power of Small Language Models

- Can these emergent abilities be achieved at a smaller scale?
- Our line of work with the ***Phi*** models aims to answer this question
  - SLMs that achieve on par performance with models of much higher scale

phi-1 **(1.3B)**

microsoft/**phi-1** 🤗

Python **coding** model with perf. comparable to models 10x larger trained on 100x more data

phi-1.5 **(1.3B)**

microsoft/**phi-1_5** 🤗

Natural language model with NL comparable to models 10x larger trained on 30x more data and reasoning comparable to models 50x larger.

The surprising Power of Small Language Models

- Can these emergent abilities be achieved at a smaller scale?
- Our line of work with the ***Phi*** models aims to answer this question
  - SLMs that achieve on par performance with models of much higher scale

phi-1 **(1.3B)**

microsoft/**phi-1**

Python **coding** model with perf. comparable to models 10x larger trained on 100x more data

phi-1.5 **(1.3B)**

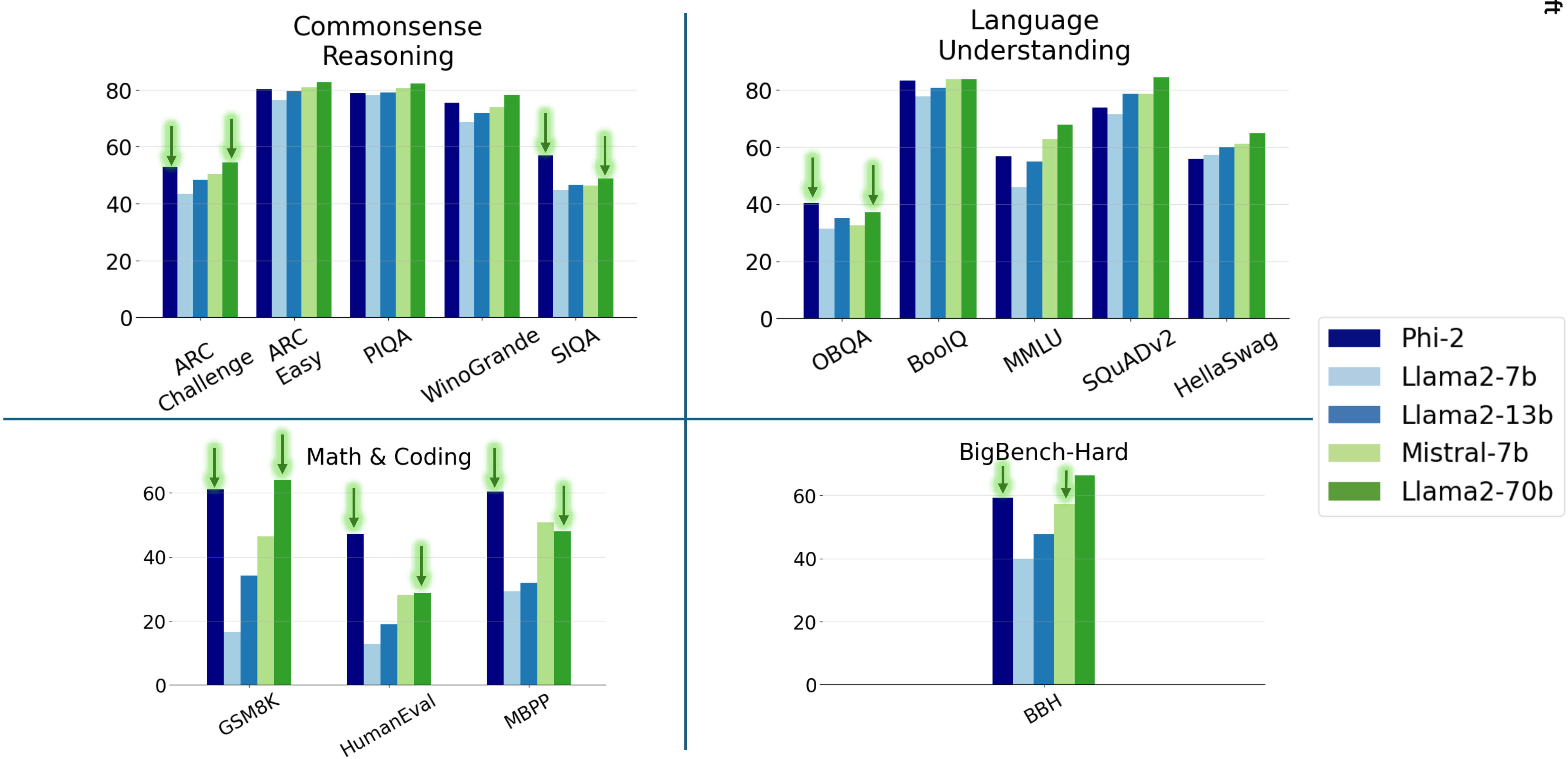
microsoft/**phi-1_5**

Natural language model with NL comparable to models 10x larger trained on 30x more data and reasoning comparable to models 50x larger.

phi-2 **(2.7B)**

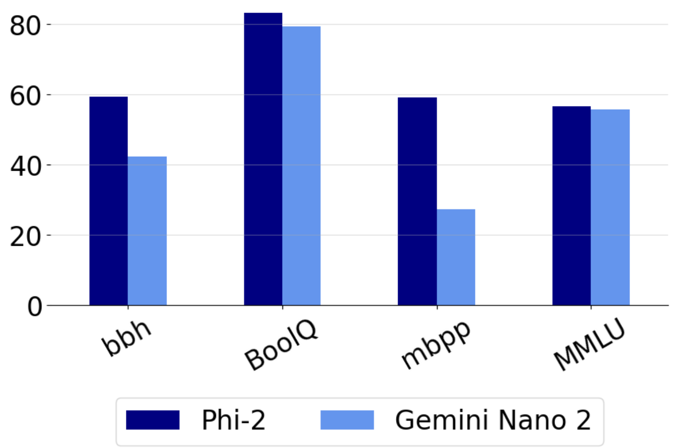
microsoft/**phi-2**

Phi-2 Performance



Phi-2 Performance

- **Gemini Nano 2 (3.25B)**



- **Gemini Ultra**

A skier slides down a frictionless slope of height 40m and length 80m. What's the skier's speed at the bottom?

Phi-2 (2.7B)

The potential energy of the skier at the top of the slope is given by mgh, where m is the mass of the skier, g is the acceleration due to gravity, and h is the height of the slope. Since the slope is frictionless, all of the potential energy is converted to kinetic energy at the bottom of the slope. The kinetic energy of the skier is given by (1/2)mv^2, where v is the speed of the skier at the bottom of the slope. Equating the potential energy and kinetic energy, we get: mgh = (1/2)mv^2.

Simplifying, we get: v^2 = 2gh

Substituting the given values, we get:
v^2 = 2(9.8)(40)

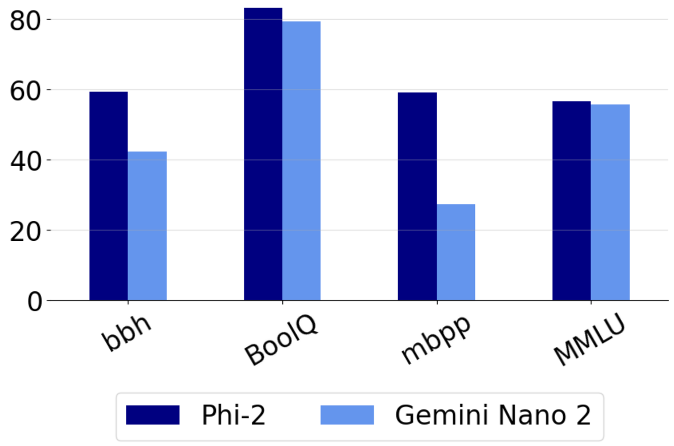
v^2 = 784

v = 28 m/s

Therefore, the skier's speed at the bottom of the slope is 28 m/s.

Phi-2 Performance

- **Gemini Nano**



- **Gemini Ultra**

Now we asked a student to give the answer and they replied with this calculation:

Start: E=mgL
End: E= (1/2)mv^2
mgL = (1/2)mv^2
v=sqrt(2gL)=sqrt(2x9.81x80)=39.6m/s

If the student is wrong, can you correct them?

Phi-2 (2.7B)

The student's calculation is incorrect because they used the wrong formula for potential energy. The formula for potential energy is mgh, where m is the mass of the object, g is the acceleration due to gravity, and h is the height of the object.

What is the secret sauce?

What is the secret sauce?

1- High-Quality Training Data: Textbooks are all you need [1]

[1] Gunasekar, Suriya, et al. "Textbooks Are All You Need." *arXiv preprint arXiv:2306.11644* (2023).

Training Language Models for Coding

The Stack

3 TB of permissive code data

- Stack dataset : “every source code in GitHub”
- Dataset size: 1T tokens

Random code sample from the Stack

```
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
                                False
                                False
                               ttern = ''
                               texts: typing.List[str]

        self._statusline_sources = ''
        self._titlestring = ''
        self._ruler = False
        self._prev_action = ''
        ...
```

If we have a **small** dataset is focused on “**text-book quality educational content**”, we can learn the task **better**, even with a **smaller** model.

Building High-Quality Datasets

1. Filtering web data:
   - GPT-4 can reliably classify documents based on “high educational value”.
   - **Challenge:** Stack (Python) is 26B tokens (around $1M cost in 2023).
   - **Solution:** label small fraction, then train a random forest classifier on it and use the classifier to filter the rest.

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 = ''
        ...
```

Building High-Quality Datasets

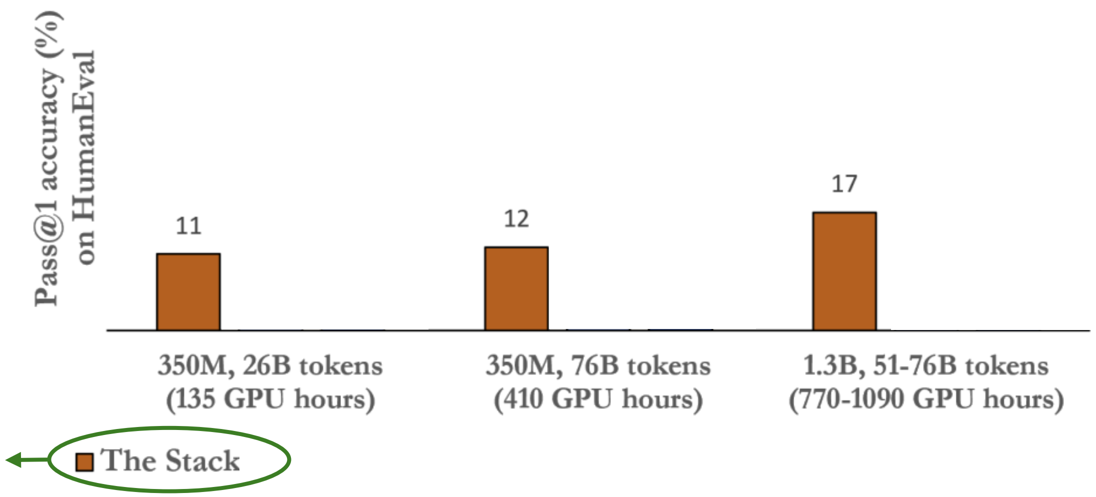
2. Create synthetic data:
   - Synthetic textbooks: teach the model coding with natural language
   - 1B tokens generated with GPT-3.5
   - **Challenge:** achieving high diversity (coding concepts, skills, level of difficulty, etc.) and low repetition
   - **Solution:** inject creative randomness into the prompt [1]

```
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.

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
```

[1] Eldan, Ronen, and Yuanzhi Li. "TinyStories: How Small Can Language Models Be and Still Speak Coherent English?" *arXiv preprint arXiv:2305.07759* (2023).

Building High-Quality Datasets

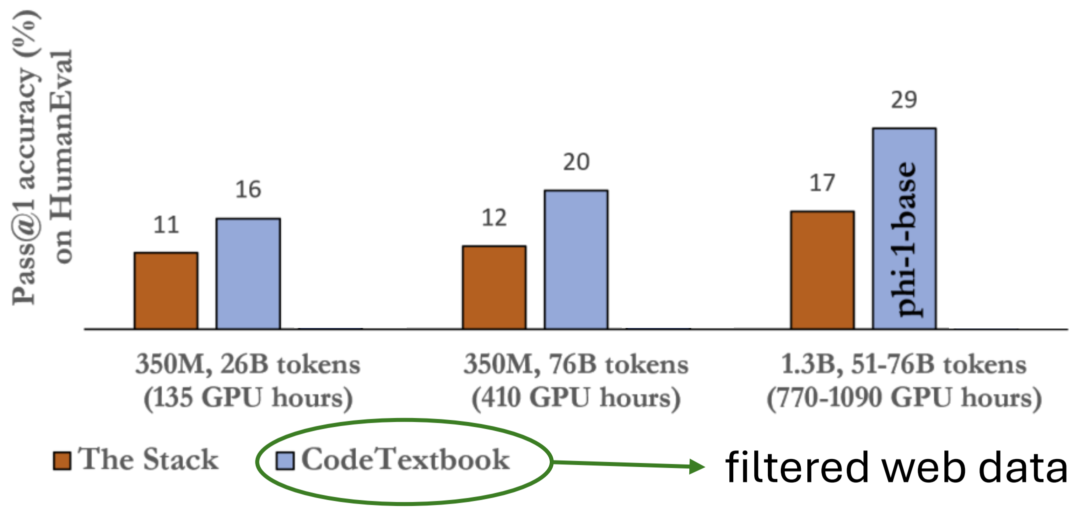
2. Create synthetic data:
    - Synthetic textbooks: teach the model coding with natural language
    - 1B tokens generated with GPT-3.5
    - **Challenge:** achieving high diversity (coding concepts, skills, level of difficulty, etc.) and low repetition
    - **Solution:** inject creative randomness into the prompt [1]



unfiltered web data

Building High-Quality Datasets

2. Create synthetic data:
   - Synthetic textbooks: teach the model coding with natural language
   - 1B tokens generated with GPT-3.5
   - **Challenge:** achieving high diversity (coding concepts, skills, level of difficulty, etc.) and low repetition
   - **Solution:** inject creative randomness into the prompt



filtered web data + synthetic textbooks

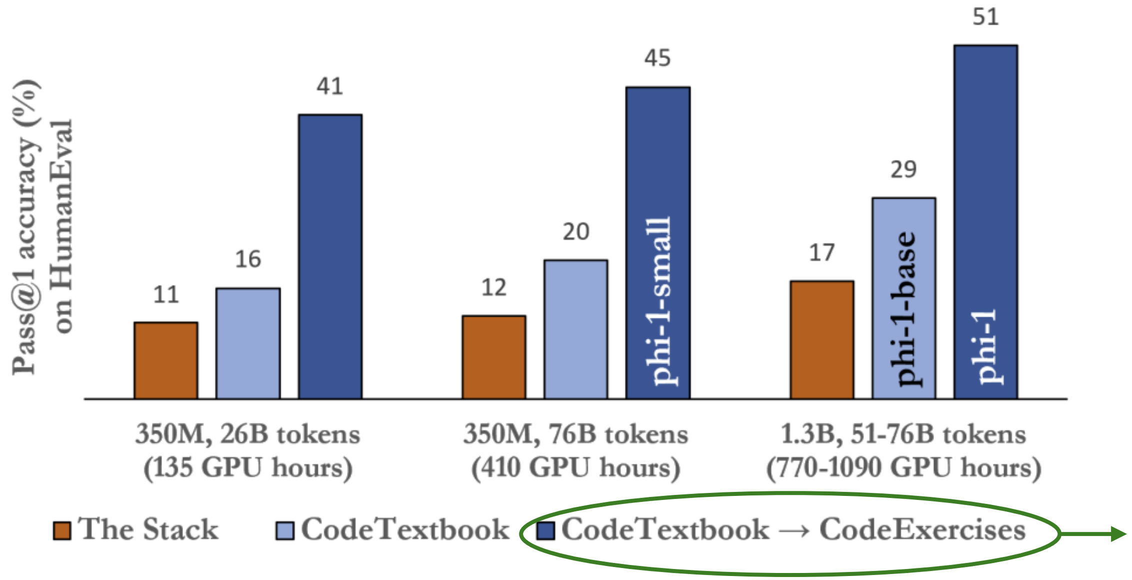
Building High-Quality Datasets

2. Create synthetic data:
    - CodeExercises: align the model to perform function completion tasks based on natural language instructions.
    - <1M exercises with 0.2B tokens generated with GPT-3.5

```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
```

Building High-Quality Datasets

2. Create synthetic data:
   - CodeExercises: align the model to perform function completion tasks based on natural language instructions.
   - <1M exercises with 0.2B tokens generated with GPT-3.5



filtered web data + synthetic textbooks, finetuned on CodeExercises

Microsoft

Comparison to Prior Models

Date	Model	Model size	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 Aug	Code Llama	34B	2.6T	53.7%	56.2%
2023 Jun	**phi-1**	1.3B	7B	50.6%	55.5%

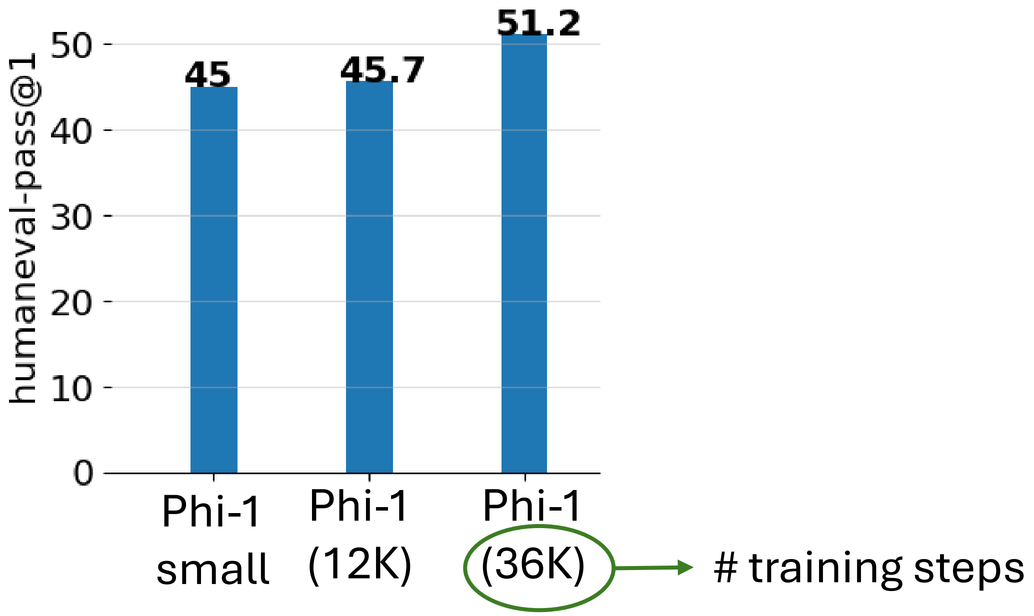
among < 10B size models, previous best was 30%

What is the secret sauce?

2- Best Practices to Scale up

Scaling up

- Training Phi-1 using the "CodeTextbook → CodeExercises" recipe
- Scale up from Phi-1-small (350M params) to Phi-1(1.3B params)
- Training from scratch:



Can we reuse Weights across Scales?

- Reusing weights from Phi-1-small (350M)
- **Challenge:** how to scale the dimensions?

1. Scaling number of layers:
    - # layers: 20 → 24

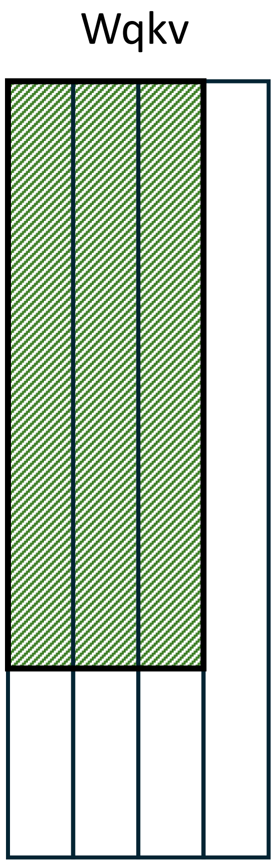
`round_int(range(num_layers_new)/num_layers_new * num_layers_old)` [1]

[1] Rae, Jack W., et al. "Scaling language models: Methods, analysis & insights from training gopher." *arXiv preprint arXiv:2112.11446* (2021).

Can we reuse Weights across Scales?

- Reusing weights from Phi-1-small (350M)
- **Challenge:** how to scale the dimensions?

2. Scaling attention layer dimensions:
    - d_model: 1024 → 2048
    - # heads: 16 → 32

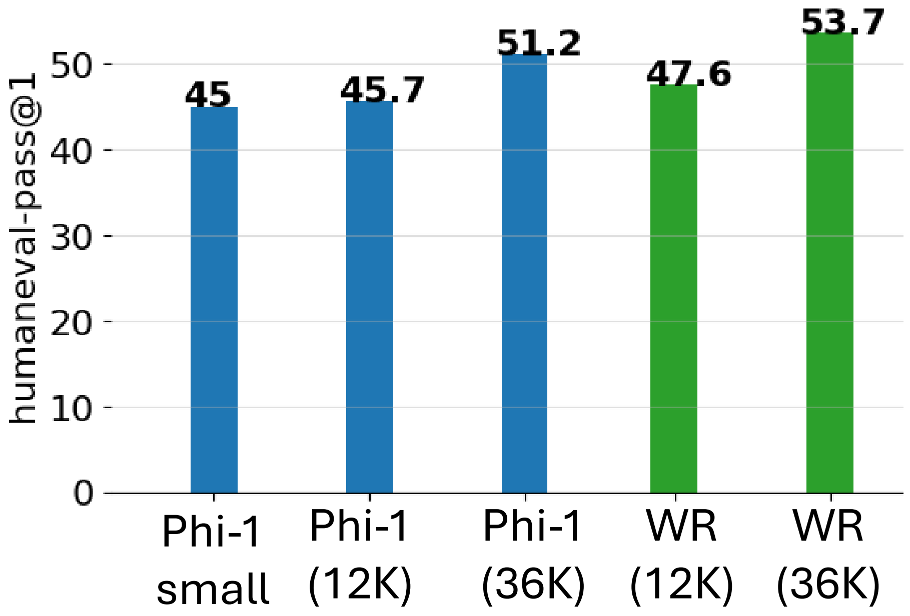


random initialization

Phi-1-small (350M) Weights

Can we reuse Weights across Scales?

- Training Phi-1 using the “CodeTextbook → CodeExercises” recipe
- Scale up from Phi-1-small (350M params) to Phi-1(1.3B params)
- Training from Phi-1-small (weight reuse (WR)):

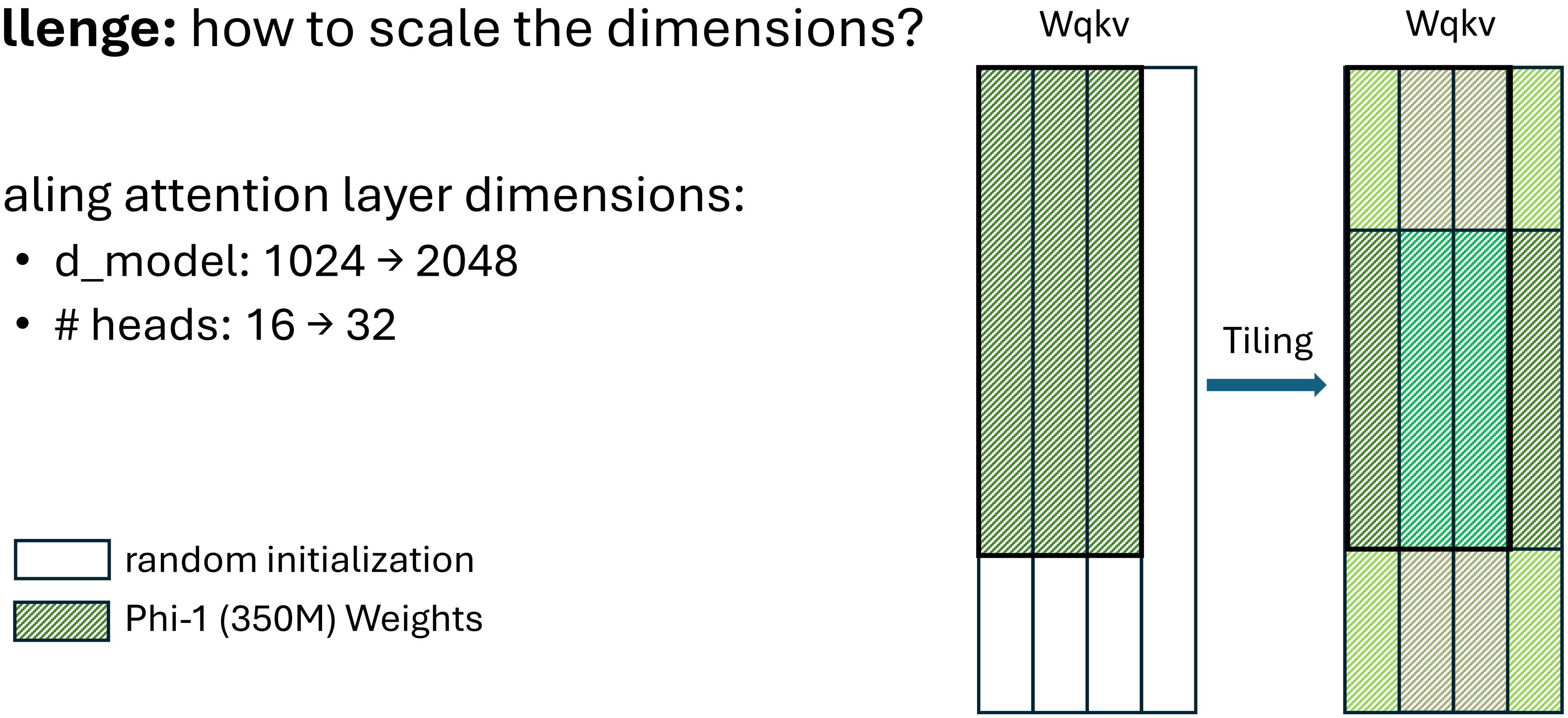


Microsoft

Can we reuse Weights across Scales?

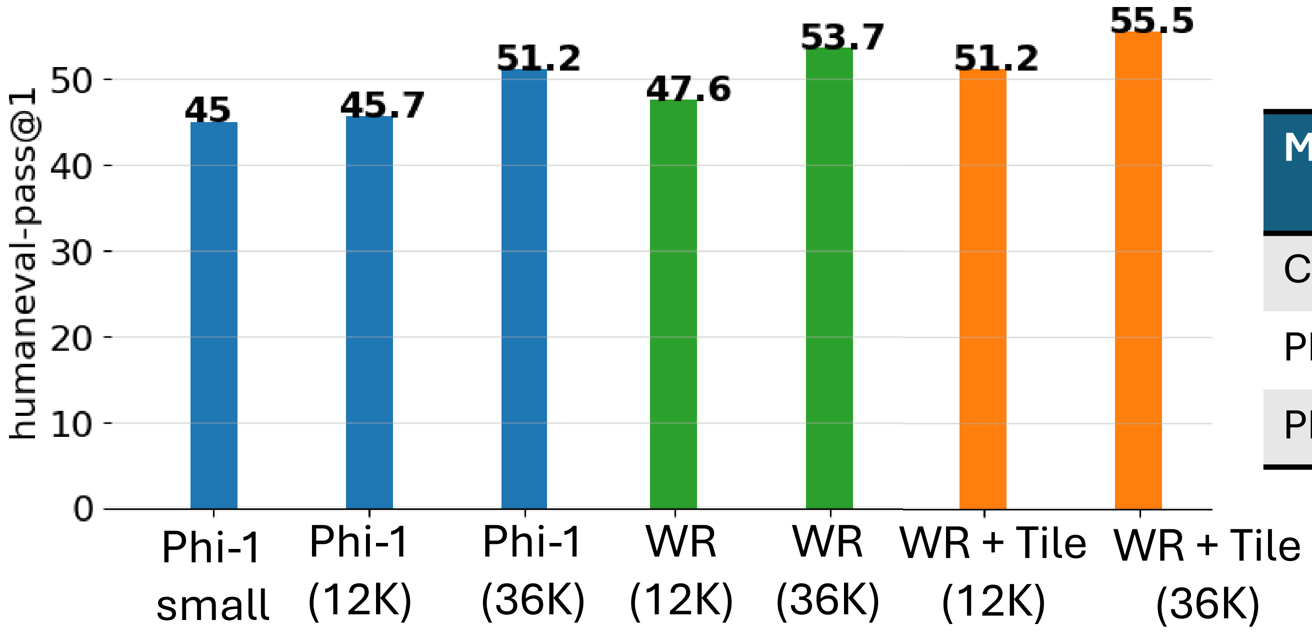
- Reusing weights from Phi-1-small (350M)
- **Challenge:** how to scale the dimensions?

2. Scaling attention layer dimensions:
    - d_model: 1024 → 2048
    - # heads: 16 → 32



Can we reuse Weights across Scales?

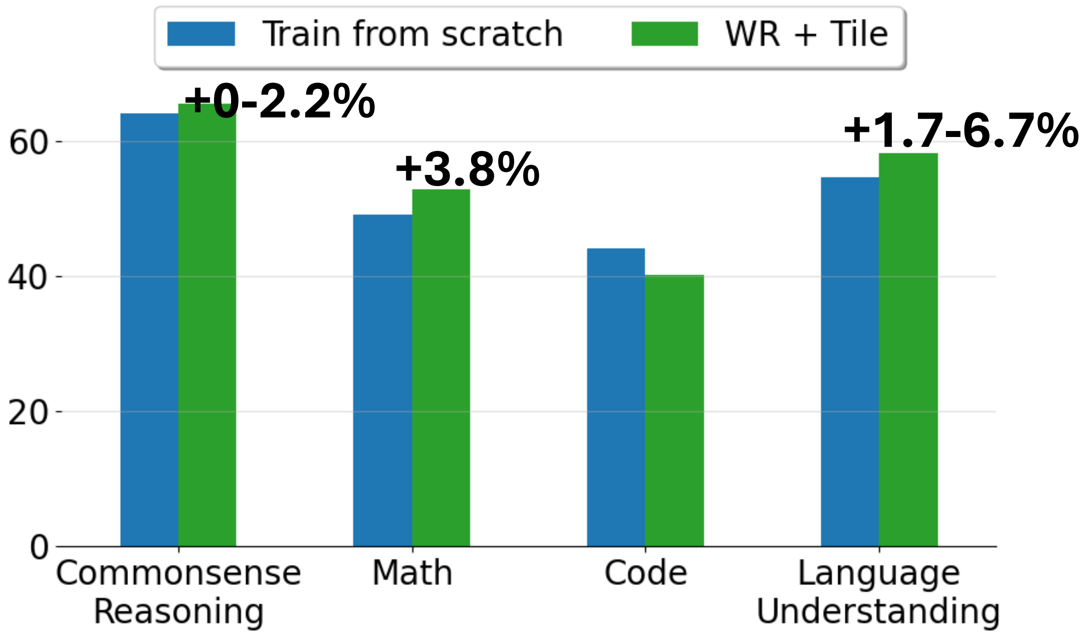
- Training Phi-1 using the “CodeTextbook → CodeExercises” recipe
- Scale up from Phi-1-small (350M params) to Phi-1(1.3B params)
- Training from Phi-1-small (weight reuse (WR)):

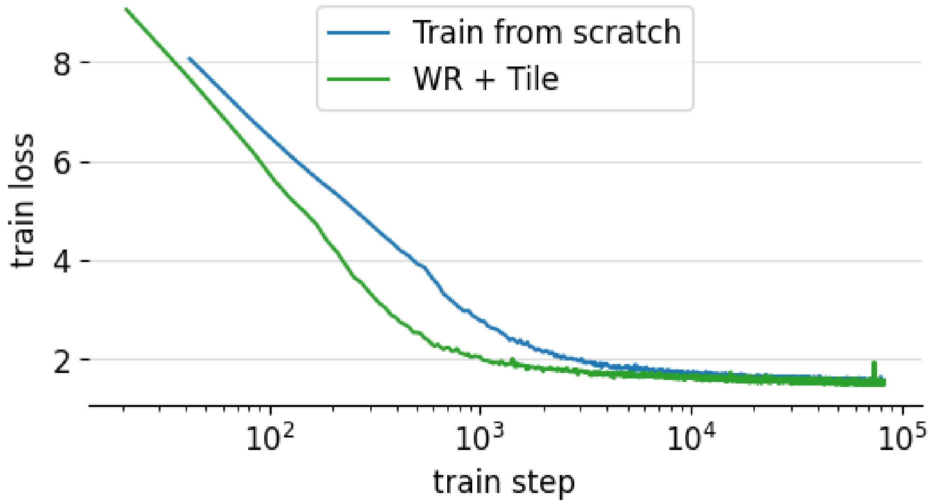


Model	Model size	Dataset size	HumanEval (pass@1)
Code Llama	**34B**	2.6T	53.7
Phi-1	1.3B	7B	50.6
Phi-1 (WR + Tile)	1.3B	7B	**55.5**

Scaling up Phi-1.5 to Phi-2

- Better performance with weight reuse





Microsoft

Conclusion

- A good, general, SLM is achievable with
  - generation and utilization of data with "textbook quality“, in contrast to conventional web data.
  - incorporation of best practices for scaling up to enhance overall performance.

Microsoft

Thanks For Listening!

@mojanjp

mojavaheripi@microsoft.com