Textbooks Are All You Need

2306.11644.pdf (arxiv.org)

背景

在过去的十年中，训练大型人工神经网络的技术取得了非凡的进展，特别是在发现 Transformer 架构之后，但是这个成功背后的科学仍然有限。在大量混乱的结果中，大约在 Transformer 被引入的同时，一种秩序的样貌开始显现。

即，随着计算量或网络规模的扩大，性能的提高有一定的可预测性，这种现象现在被称为缩放定律(**scaling laws**)。深度学习中规模的后续探索是由这些缩放定律指导的，这些定律的变种的发现导致了性能的快速跃升。

亮点

本文探讨了可以沿着不同维度获得的改进：数据的质量。提高数据质量可以**大大改变缩放定律的形状**，可能使得训练/模型更加精简的模型能够达到大规模模型的性能。这项工作展示了高质量数据甚至可以提高大型语言模型（LLMs）的 SOTA，同时大幅减少数据集大小和训练计算量。重要的是，需要较少训练的较小模型可以显著减少 LLMs 的环境成本。

本篇文章将注意力集中在针对代码训练的 LLMs 上，具体来说是从函数 docstrings 中编写简单的 Python 函数。

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 Jun	**phi-1**	1.3B	7B	50.6%	55.5%

实现方法

高质量数据集

当前数据集存在的问题

- 许多样本不是自包含的，意味着它们依赖于其他模块或文件，这些模块或文件是代码片段外部的，使得它们很难在没有额外上下文的情况下理解。
- 典型的例子不涉及任何有意义的计算，而是由一些琐碎或样板式的代码组成，例如定义常量、设置参数或配置 GUI 元素。
- 包含算法逻辑的示例通常被埋在复杂或文档不好的函数中，使得它们难以理解或学习。
- 这些示例偏向于某些主题或用例，导致数据集中编码概念和技能的分布不平衡。

作者认为好的数据集应该：与人类认为良好的 “教科书” 具有相同特点：**清晰、自包含、有指导性和平衡**。

A filtered code-language dataset

- 原始数据集：经过去重的 The Stack 和 StackOverflow 的 Python 部分，两者共包含超过 3500 万个文件/样本，总计超过 350 亿个 token
- 使用 GPT-4 注释这些文件的一小部分（约 10 万个样本）的质量：给定一个代码片段，模型被提示 “确定它对于一个目标是学习基本编程概念的学生的教育价值”
- 训练一个随机森林分类器，使用 output embedding from a pretrained codegen model 作为特征，预测文件/样本的质量
- 最后留下 6B tokens

Educational values deemed by the filter

High educational value

```
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

```
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 = ''
        self._displayed_texts: typing.List[str]
    = []
        self._statusline_sources = ''
        self._titlestring = ''
        self._ruler = False
        self._prev_action = ''
        ...
```

synthetic textbook-quality datasets

- 这步重点强调增加数据的 diversity：这些示例应该涵盖广泛的编程概念、技能和场景，它们应该在难度、复杂度和风格上有所不同
- 然而实现多样性并不简单，特别是当使用另一个语言模型生成的合成数据时。仅仅提示模型生成编程教材或一组练习，即使在指令或参数上有些变化，也很可能会导致非常同质化和冗余的数据集，其中相同的概念和解决方案一遍又一遍地重复，只有微小的变化。这是因为语言模型倾向于遵循其训练数据和先验条件下最可能或最常见的路径，它们缺乏创造力或探索生成代码的替代或新颖方式的动力。
- 解决方案，参考【How small can language models be and still speak coherent english?】采用了给 prompt 注入随机性的方法激发产出的多样性（没有具体事例）
- less than 1B tokens of GPT-3.5 generated Python textbooks

```
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
```

The CodeExercises dataset

- less than 180M tokens of Python exercises and solutions generated by GPT-3.5
- Each exercise is a docstring of a function that needs to be completed
- main means of eliciting diversity is by constraining the function names

```
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
```

模型架构和训练

架构

- a decoder only transformer
- FlashAttention
- MHA and MLP layers in parallel
- 1.3B parameter phi-1 model consists of 24 layers, hidden dimension of 2048, MLP-inner dimension of 8192, and 32 attention heads of dimension 64 each
- 350M parameter phi1-small model consists of 20 layers, hidden dimension of 1024, MLP-inner dimension of 4096, and 16 attention heads of dimension 64 each
- a rotary position embedding
- the same tokenizer as codegen-350M-mono

- “⟨ | endoftext | ⟩” token used for separating the files
- next-token prediction loss（both pretraining and finetuning）
- **8 Nvidia-A100 GPUs using deepspeed**
- **phi-1-base：4 days; phi-1:+7h**

预训练

- phi-1-base：CodeTextbook dataset (filtered code-language corpus and synthetic textbooks)
- 8 epochs

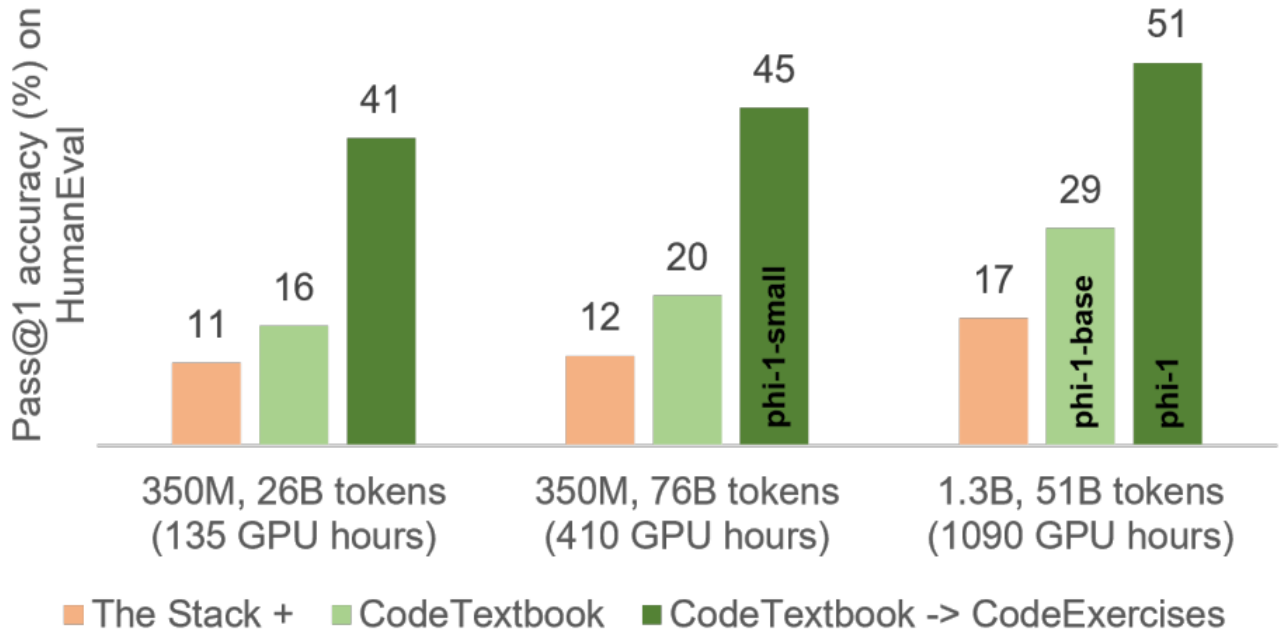
微调

phi-1 is obtained by finetuning phi-1-base on the CodeExercises dataset

实验结果

HumanEval 实验结果

微调过程可能有助于模型重新组织和巩固在预训练期间获得的知识，即使这种知识不明确存在于我们的 CodeExercises 数据集中



微调提升了模型的理解能力

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)
    %
        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) %
        alice_points += 1
    if (i %
        bob_points += 1
print(f"Alice gets {alice_points}
    points, while Bob gets {
    bob_points} points.")
```

微调可以提高模型使用外部库的能力

通过在 CodeExercises 上微调，我们意外地提高了模型使用外部库（如 Pygame 和 Tkinter）的能力，尽管我们的练习不包含这些库。这表明，我们的微调不仅改进了我们针对的任务，还使得从预训练中提取不相关任务更容易

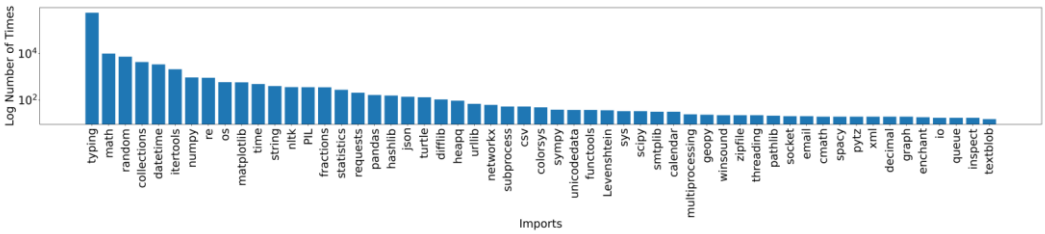


Figure 3.1: The number of imports among 879486 exercises in the finetuning (ignored libraries imported less than 10 times). The plot is generated by **phi-1** with the following prompt: "I have a dictionary, first sort the dictionary

使用 GPT4 给非传统问题打分

- 模拟 code review 的效果
- 人工全新编写的 50 个例子

Model	Size	Training tokens	Score	HumanEval
CodeGen-Mono-350M [NPH+23]	350M	577B	0.19	12.8%
CodeGen-Mono-16.1B [NPH+23]	16.1B	577B	0.38	29.3%
Replit [Rep23]	2.7B	525B	0.37	21.9%
StarCoder [LAZ+23]	15.5B	1T	0.51	33.6%
phi-1-base	1.3B	7B	0.37	29%
phi-1-small	350M	7B	0.45	45%
phi-1	1.3B	7B	0.52	50.6%

Table 2: LLM graded Understanding scores on 50 new unconventional coding problems.

数据修剪以进行无偏性能评估

- prune the CodeExercises dataset by removing files that are "similar" to those in HumanEva
- retrain our model on such pruned data, and still observe strong performance on HumanEval

τ		Problem Count	**phi-1**	**phi-1 retrained on pruned data**	StarCoder-Prompted [LAZ+23]
0.95	similar	71	81.7%	74.6%	57.7%
	non-similar	93	26.9%	32.3%	29.0%
	total	164	50.6%	50.6%	41.5%
0.9	similar	93	63.4%	51.6%	48.4%
	non-similar	71	33.8%	36.6%	32.4%
	total	164	50.6%	45.1%	41.5%
0.85	similar	106	62.3%	52.8%	47.2%
	non-similar	58	29.3%	34.5%	31.0%
	total	164	50.6%	46.3%	41.5%
0.8	similar	116	59.5%	52.6%	45.7%
	non-similar	48	29.2%	27.1%	31.2%
	total	164	50.6%	45.1%	41.5%

Table 3: Percentage of similar versus non-similar HumanEval problems correctly solved by different models. Similarity is determined based on whether or not the corresponding HumanEval problem has any close matches inside the CodeExercises dataset (for a given τ). The problem count denotes the number of HumanEval problems within each subset. Here, τ is the threshold on AST-based match rate between codes for similarity check.

局限性

phi-1 缺乏比较大型模型如使用特定 API 进行编程或使用较少常见的软件包等领域特定知识。最后，由于数据集的结构化特性以及语言和风格的缺乏多样性，phi-1 对风格变化或提示中的错误不够鲁棒（例如，在提示中有语法错误时，其性能会大幅降低）

其他能力下降严重

展望

开发创建高质量数据集的良好方法是推进自然语言处理和相关领域研究的一个核心方向。然而，创建高质量数据集并不是一项简单的任务，它提出了几个需要解决的挑战。

一个挑战是确保数据集**涵盖所有相关的内容和概念**，以及以平衡和代表性的方式实现。

另一个挑战是确保数据集**真正具有多样性和非重复**性，以便模型不仅仅是过拟合数据或记忆特定的模式或解决方案。这需要找到方法将随机性和创造性注入到数据生成过程中，同时仍然保持示例的质量和连贯性。

此外，即使创建了这样的数据集，我们仍然缺乏一种好的方法来**评估数据中的多样性和冗余程度**。例如，如果我们有一个编程练习的数据集，很难确定每个练习有多少不同的变化形式，以及它们在数据集中的分布情况。