

Let's train a Tiny Language Model from scratch



About me (Luca)

I have a background in applied sciences, I live close to Turin and I'm quite active within the local Python community.

I'm one of the 3 co-founders and CTO of Clearbox AI, a startup hosted by Politecnico di Torino incubator. Before Clearbox I worked several years as a scientific software developer.

During my free time I do enjoy hiking and I'm currently experimenting with farming.



Susa valley

Why Tiny Language Models?

My interest in Tiny Language Models

Why Tiny Language Models?

```
user_id,time,amount,category,description
u_001,2025-04-10 14:32:10,49.99,Groceries,Purchase at WholeMart
u_002,2025-04-11 09:15:47,15.00,Coffee,Latte and croissant at BrewBros
u_003,2025-04-09 18:22:30,120.50,Electronics,Headphones from TechStop
u_001,2025-04-11 11:03:02,32.20,Books,Online order from ReadMore
```

Generation of text data in large privacy-sensitive datasets.

- Needs to be done on prem (sensitive data)
- Needs to be fast (billions of data rows)

Why Tiny Language Models?

Generation of synthetic populations.

Using aggregated data from open sources to simulate population behaviours via **agent based modeling**.

Agents = Language Models

Why Tiny Language Models?

Generation of synthetic populations.

Using aggregated data from open sources to simulate population behaviours via **agent based modeling**.

Agents = Language Models



Pierluigi Aldobrandi

42 years old, Milano

Reading Movies Cooking Music Technology

About me

about-me \nMy name is Pierluigi Aldobrandi and I'm a 42-year-old Italian man living in Milan. I work in Industry and my income is around €28 000 a year. I have a tertiary education. I rent my place in Milan and I live with one other person so it is a household of two. Our combined family income is about €73 000. I'm generally a practical person focusing on experiences. I am not a house owner.\n

Weekly Diary

Monday

Monday: I started my week with the usual routine. I woke up, had a quick breakfast at home, and headed to work in Industry. The day was filled with meetings and project-related tasks. During lunch, I grabbed a quick Panino at the local bar near my workplace `[Expense: 6 euros | Category: Food | Location: Bar near Workplace]`. After work, I went straight home to cook dinner. [Interaction: Colleagues during Lunch].

Tuesday

Wednesday

Wednesday: Work was quite demanding today, dealing with an unexpected issue. I had lunch at the company canteen `[Expense: 8 euros | Category: Food | Location: Company Canteen]`. In the evening, I went to the supermarket to buy groceries for the rest of the week `[Expense: 45 euros | Category: Groceries | Location: Supermarket]`. I cooked a simple pasta dish for dinner at home.

Reproducing GPT-2 (124M) in llm.c in 90 minutes for \$20 #481

karpathy started this conversation in General



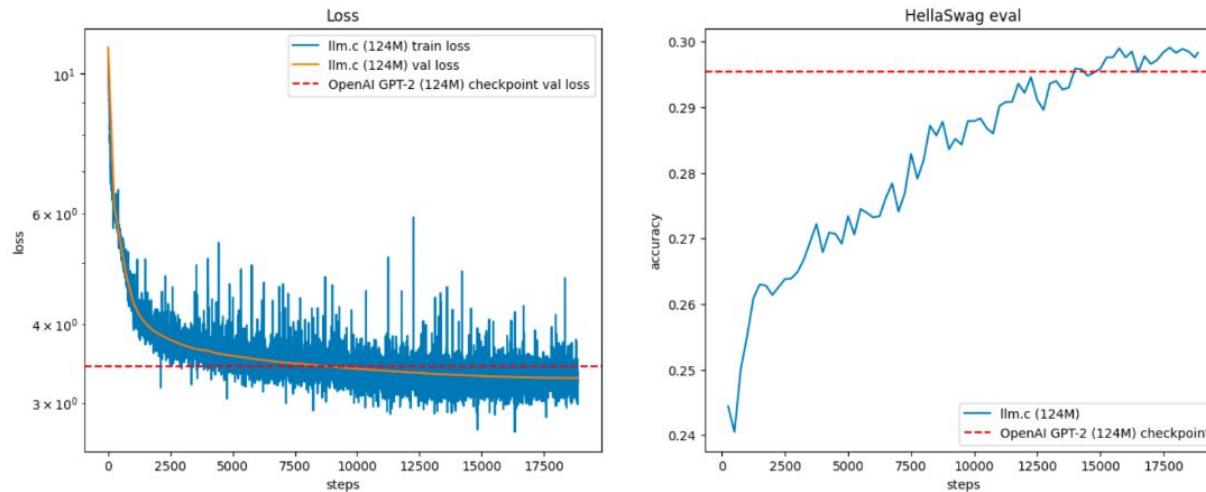
karpathy

on May 28, 2024

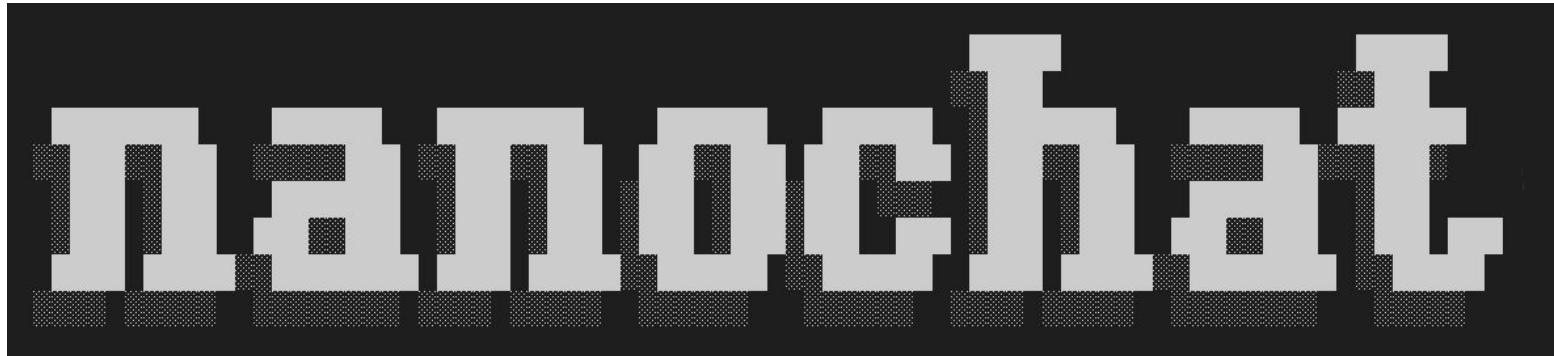
Maintainer

edited · · ·

Let's reproduce the [GPT-2](#) (124M) in [llm.c](#) (~4,000 lines of C/CUDA) in 90 minutes for \$20. The 124M model is the smallest model in the GPT-2 series released by OpenAI in 2019, and is actually quite accessible today, even for the GPU poor. With [llm.c](#), which is quite efficient at up to ~60% model flops utilization, reproducing this model on one 8X A100 80GB SXM node takes ~90 minutes. For example, on [Lambda](#) this node goes for ~\$14/hr, so the total cost of reproducing this model today is about \$20. You can train the model with a single GPU too, it would just take proportionally longer (e.g. ~4-24 hours depending on the GPU). In addition, [llm.c](#) still has a lot of pending optimizations and people haven't tried to tune the training in the style of [cramming](#), so I'd say we're likely to see significant improvements on this number. So here is the run, training the 12-layer, 12-headed, 768-dimension, 124M Transformer on 10 billion tokens of [FineWeb](#):



<https://github.com/karpathy/llm.c/discussions/481>

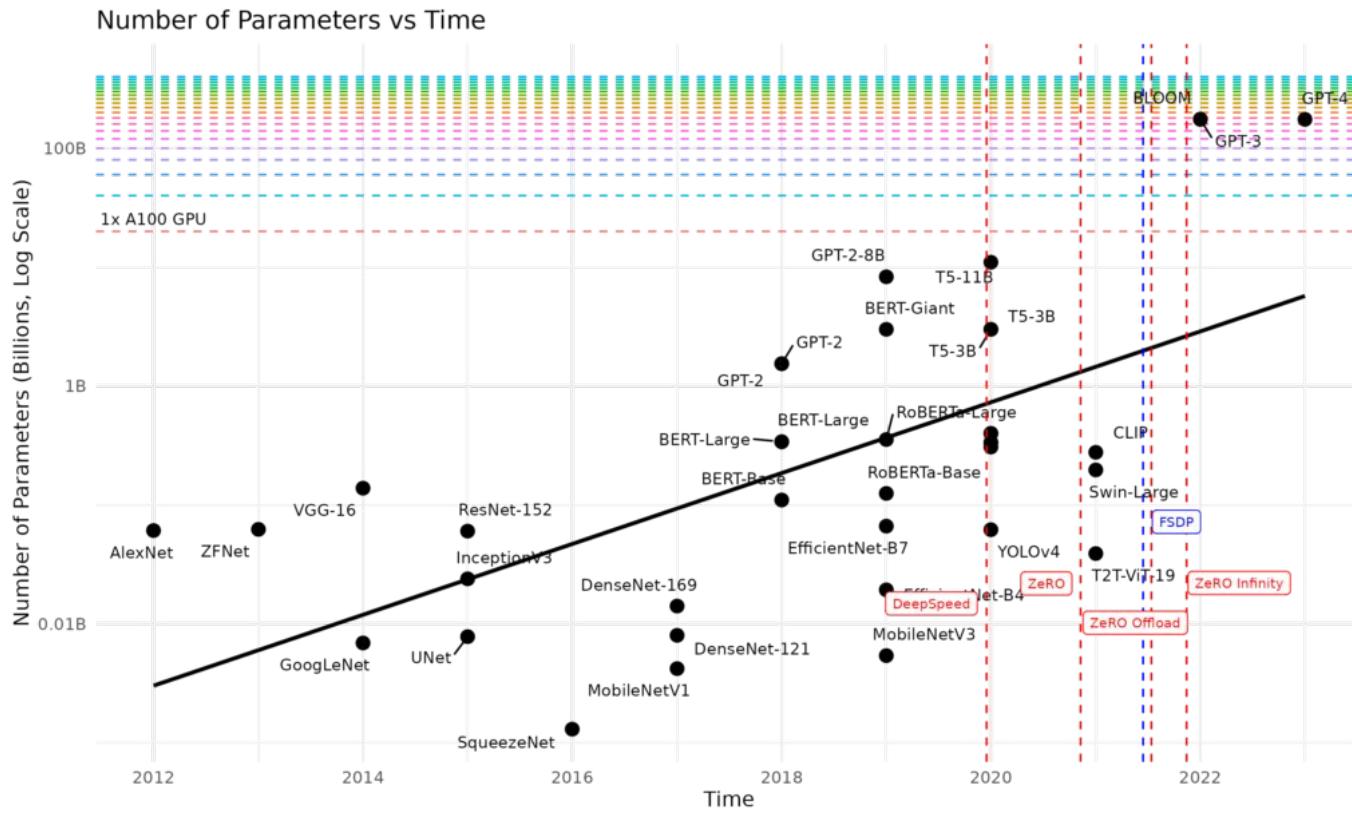


<https://github.com/karpathy/nanochat>

<https://nanochat.karpathy.ai/>

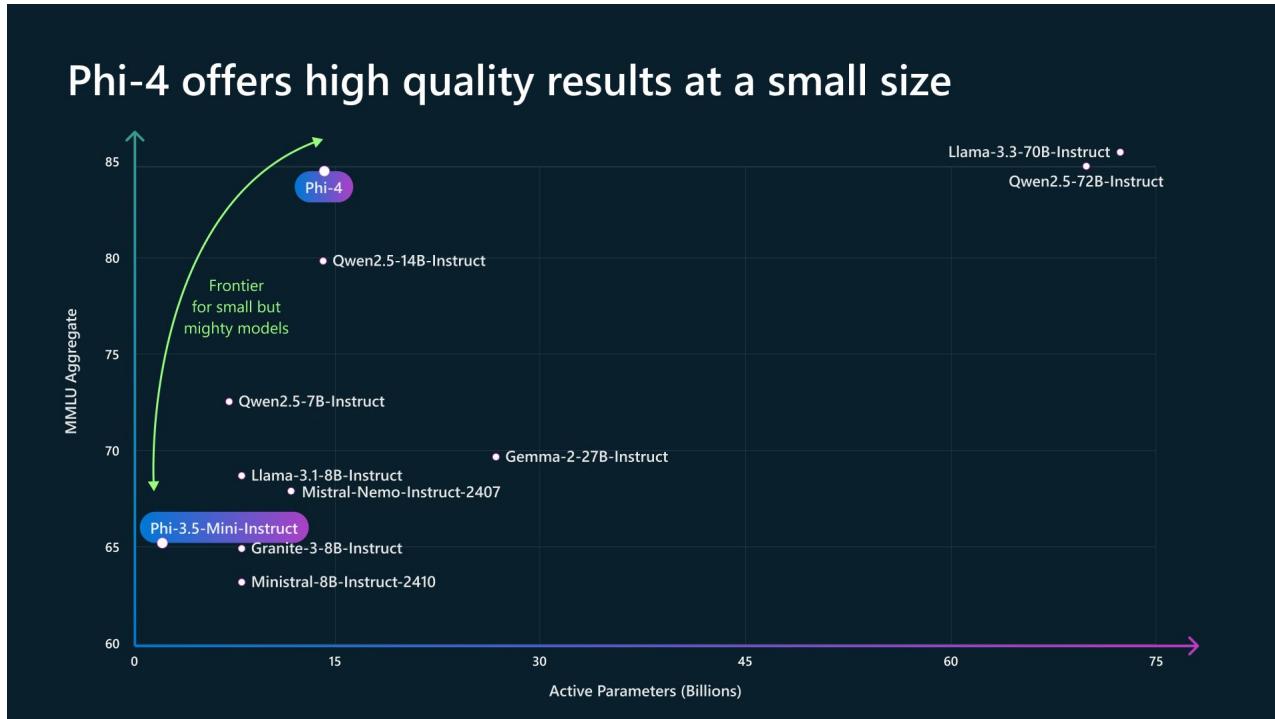
Small and Tiny Language Models

The bigger the better?



<https://www.ddn.com/resources/whitepapers/large-language-models/>

The bigger the better?



<https://techcommunity.microsoft.com/blog/aiplatformblog>

Structured Extraction

Isaac is a current undergraduate at Rutgers University NB, pursuing a double major in Economics and Information and Technology and Informatics. He hopes to pursue a career in Data Analytics or Cyber Security. He enjoys his free time diving into books and dancing. He's also learning how to cook, so any cooking advice thrown his way would be greatly appreciated.

Kelly always loved to write. She earned a Bachelor's Degree in Creative Writing/American Studies from Rutgers University. Kelly contributes as a guest writer on several blogs; her favorite topics include third wave feminism, medieval royalty, and Broadway musicals. She keeps busy on the weekends by hiking, creating mix tapes for friends and family, and teaching color guard to local high school students. Kelly resides in East Windsor, New Jersey with her husband.

Amy grew up in Somerset, New Jersey and got her degree in Journalism and Media Studies from Rutgers University. She has experience working everywhere from media marketing to the fashion industry

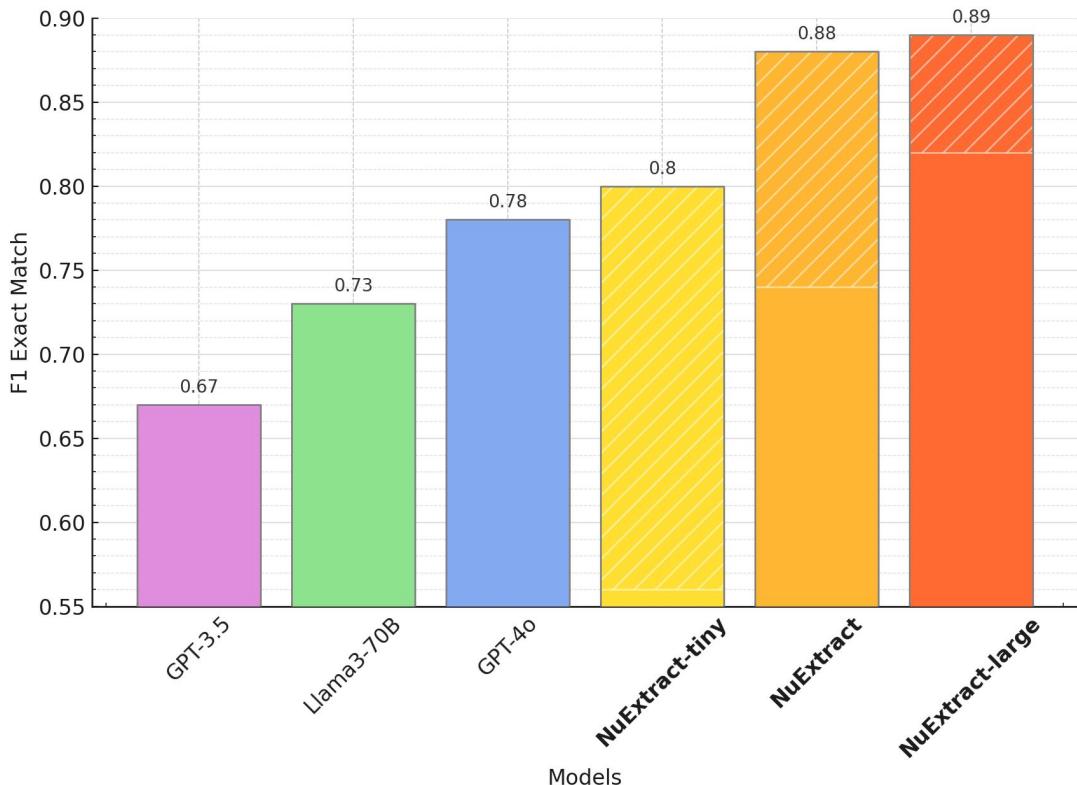
text



template

```
{
    "Company: {
        "Name: ,
        "Description: ,
        "Experience: "
    },
    "Team: {
        "Members": [
            {
                "Name: Isaac,
                "Bio: ,
                "Education: Rutgers University NB,
                "Interests: [diving into books, dancing, cooking]
            },
            {
                "Name: Kelly,
                "Bio: ,
                "Education: Rutgers University,
                "Interests: [
                    hiking,
                    creating mix tapes for friends and family,
                    teaching color guard to local high school students
                ]
            },
            {
                "Name: Amy,
                "Bio: ,
                "Education: Rutgers University,
                "Interests: []
            }
        ]
    }
}
```

output



<https://huggingface.co/numind/NuExtract-large>

Tiny Models

A model that can be trained on a single GPU within a few days?

<< 1 billion parameters.

GPT-2-small : **124 million parameters**

TinyStories

105.07759v2 [cs.CL] 24 May 2023

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

Ronen Eldan* and Yuanzhi Li†

Microsoft Research

April 2023

Abstract

Language models [4, 5, 21] (LMs) are powerful tools for natural language processing, but they often struggle to produce coherent and fluent text when they are **small**. Models with around 125M parameters such as GPT-Neo (small) [3] or GPT-2 (small) [23] can rarely generate coherent and consistent English text beyond a few words even after extensive training. This raises the question of whether the emergence of the ability to produce coherent English text only occurs at larger scales (with hundreds of millions of parameters or more) and complex architectures (with many layers of global attention).

In this work, we introduce **TinyStories**, a synthetic dataset of short stories that only contain words that a typical 3 to 4-year-olds usually understand, generated by GPT-3.5 and GPT-4. We show that TinyStories can be used to train and evaluate LMs that are much smaller than the state-of-the-art models (**below 10 million total parameters**), or have much simpler architectures (**with only one transformer block**), yet still produce fluent and consistent stories with several paragraphs that are diverse and have almost perfect grammar, and demonstrate reasoning capabilities.

We also introduce a new paradigm for the evaluation of language models: We suggest a framework which uses GPT-4 to grade the content generated by these models as if those were stories written by students and graded by a (human) teacher. This new paradigm overcomes the flaws of standard benchmarks which often require the model's output to be very structured, and moreover it provides a multidimensional score for the model, providing scores for different capabilities such as grammar, creativity and instruction-following.

We hope that TinyStories can facilitate the development, analysis and research of LMs, especially for low-resource or specialized domains, and shed light on the emergence of language capabilities in LMs.



TinyStories

| Hidden size | Layer | Eval loss | Creativity | Grammar | Consistency | Instruct | Plot |
|--------------------|-------|-----------|------------|-----------|-------------|-----------|-----------|
| 64 | 12 | 2.02 | 4.84/0.36 | 6.19/0.42 | 4.75/0.31 | 4.34/0.23 | 4.39/0.20 |
| 64 | 8 | 2.08 | 4.68/0.33 | 6.14/0.41 | 4.45/0.27 | 4.34/0.23 | 4.40/0.21 |
| 64 | 4 | 2.26 | 3.97/0.20 | 5.31/0.22 | 3.77/0.18 | 3.79/0.14 | 3.71/0.06 |
| 64 | 2 | 2.38 | 2.94/0.00 | 4.33/0.00 | 2.41/0.00 | 2.86/0.00 | 3.40/0.00 |
| 128 | 12 | 1.62 | 6.02/0.58 | 7.25/0.66 | 7.20/0.64 | 6.94/0.63 | 6.58/0.65 |
| 128 | 8 | 1.65 | 5.97/0.57 | 7.23/0.66 | 7.10/0.62 | 6.87/0.62 | 6.16/0.57 |
| 128 | 4 | 1.78 | 5.70/0.52 | 6.91/0.58 | 6.60/0.56 | 6.00/0.49 | 5.53/0.44 |
| 128 | 2 | 1.92 | 4.90/0.37 | 6.43/0.48 | 4.75/0.31 | 5.23/0.37 | 4.89/0.31 |
| 256 | 12 | 1.34 | 6.66/0.71 | 7.80/0.79 | 8.38/0.79 | 7.68/0.75 | 7.18/0.78 |
| 256 | 8 | 1.38 | 6.54/0.68 | 7.72/0.77 | 8.02/0.75 | 7.92/0.78 | 7.23/0.79 |
| 256 | 4 | 1.47 | 6.32/0.64 | 7.64/0.75 | 7.76/0.71 | 8.07/0.81 | 7.18/0.78 |
| 256 | 2 | 1.60 | 6.23/0.62 | 7.50/0.72 | 7.20/0.64 | 7.23/0.68 | 6.50/0.64 |
| 512 | 12 | 1.19 | 6.90/0.75 | 8.46/0.93 | 9.11/0.89 | 8.21/0.83 | 7.37/0.82 |
| 512 | 8 | 1.20 | 6.85/0.74 | 8.34/0.91 | 8.95/0.87 | 8.05/0.80 | 7.26/0.79 |
| 512 | 4 | 1.27 | 6.75/0.72 | 8.35/0.91 | 8.50/0.81 | 8.34/0.85 | 7.36/0.81 |
| 512 | 2 | 1.39 | 6.40/0.66 | 7.72/0.77 | 7.90/0.73 | 7.76/0.76 | 7.13/0.77 |
| 768 | 12 | 1.18 | 7.00/0.77 | 8.30/0.90 | 9.20/0.90 | 8.23/0.83 | 7.47/0.84 |
| 768 | 8 | 1.18 | 7.02/0.77 | 8.62/0.97 | 9.34/0.92 | 8.36/0.85 | 7.34/0.81 |
| 768 | 4 | 1.20 | 6.89/0.75 | 8.43/0.93 | 9.01/0.88 | 8.44/0.87 | 7.52/0.85 |
| 768 | 2 | 1.31 | 6.68/0.71 | 8.01/0.83 | 8.42/0.80 | 7.97/0.79 | 7.34/0.81 |
| 768 | 1 | 1.54 | 6.00/0.58 | 7.35/0.68 | 7.25/0.64 | 5.81/0.46 | 6.44/0.63 |
| 1024 | 12 | 1.22 | 7.05/0.78 | 8.43/0.93 | 8.98/0.87 | 8.18/0.82 | 7.29/0.80 |
| 1024 | 8 | 1.20 | 7.13/0.80 | 8.25/0.89 | 8.92/0.87 | 8.47/0.87 | 7.47/0.84 |
| 1024 | 4 | 1.21 | 7.04/0.78 | 8.32/0.90 | 8.93/0.87 | 8.34/0.85 | 7.47/0.84 |
| 1024 | 2 | 1.27 | 6.68/0.71 | 8.22/0.88 | 8.52/0.81 | 8.04/0.80 | 7.24/0.79 |
| 1024 | 1 | 1.49 | 6.36/0.65 | 7.77/0.78 | 7.47/0.67 | 6.09/0.50 | 6.42/0.62 |
| GPT-Neo (125M) | - | - | 3.34/0.08 | 5.27/0.21 | 4.22/0.24 | - | - |
| GPT-2-small (125M) | - | - | 3.70/0.14 | 5.40/0.24 | 4.32/0.25 | - | - |
| GPT-2-med (355M) | - | - | 4.22/0.24 | 6.27/0.44 | 5.34/0.39 | - | - |
| GPT-2-large (774M) | - | - | 4.30/0.26 | 6.43/0.48 | 6.04/0.48 | - | - |
| GPT-4 | - | - | 8.21/1.00 | 8.75/1.00 | 9.93/1.00 | 9.31/1.00 | 8.26/1.00 |

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

TinyStories

| Prompt | 2.5M, 8 layers | 33M, 4 layers | 21M, 1 layer | GPT2-XL (1.5B) |
|--|--|---|---|--|
| Alice was so tired when she got back home so she went | to bed. | straight to bed. | to bed with a big smile on her face. | outside |
| Lily likes cats and dogs. She asked her mom for a dog and her mom said no, so instead she asked | her mom if she could have a dog. | her dad for a cat. | her mom again. | her dad. They brought her a Chihuahua, and she took to them immediately. |
| Alice and Jack walked up the street and met a girl in a red dress. The girl said to them, "Hi, I'm Jane. What are your names?" | Alice smiled and said, "My name is Daisy. What's your name?" | Alice said, "I'm Alice and this is Jack." | Jack smiled and said, "I'm Jane. Nice to meet you!" | Jane said, "Jack and Kate" |

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

[← Back to Articles](#)

SmollM - blazingly fast and remarkably powerful

Published July 16, 2024

[Update on GitHub](#)[loubnabnl](#)

Loubna Ben Allal

[anton-1](#)

Anton Lozhkov

[eliebak](#)

Elie Bakouch

≡ TL;DR

This blog post introduces [SmollM](#), a family of state-of-the-art small models with 135M, 360M, and 1.7B parameters, trained on a new high-quality dataset. It covers data curation, model evaluation, and usage.

<https://huggingface.co/blog/smollm>



Gemma



<https://deepmind.google/models/gemma/>

Why pushing for more Tiny and Small Language Models?

Potential benefits of Small and Tiny Language models:

- **Environmental benefits**
- **AI democratisation**

LLaMA-3

Energy needed for model training

| | Time (GPU hours) | Power Consumption (W) | Carbon Emitted(tCO2eq) |
|-------------|------------------|-----------------------|------------------------|
| Llama 3 8B | 1.3M | 700 | 390 |
| Llama 3 70B | 6.4M | 700 | 1900 |
| Total | 7.7M | | 2290 |

LLMs are becoming
less and less
transparent

William Falcon ⚡ ✨
@_willfalcon

GPT-4 paper : [cdn.openai.com/papers/gpt-4.p...](https://cdn.openai.com/papers/gpt-4.pdf)

Let me save you the trouble:

—

GPT-4 Technical Report

—

OpenAI*

Abstract

we used python



7:39 PM · Mar 14, 2023 · 430.3K Views

Why are smaller models getting better?



Training Compute-Optimal Large Language Models

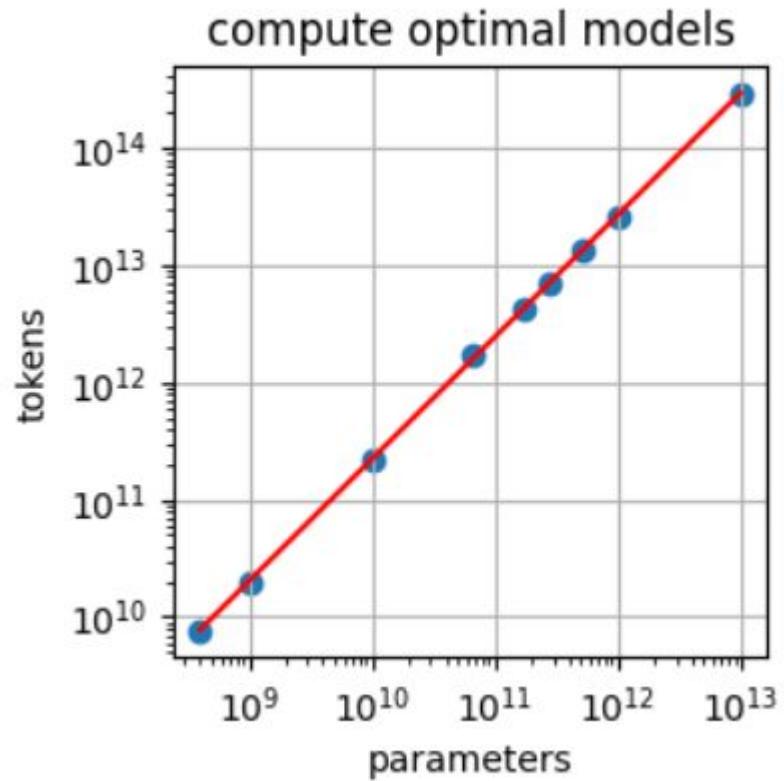
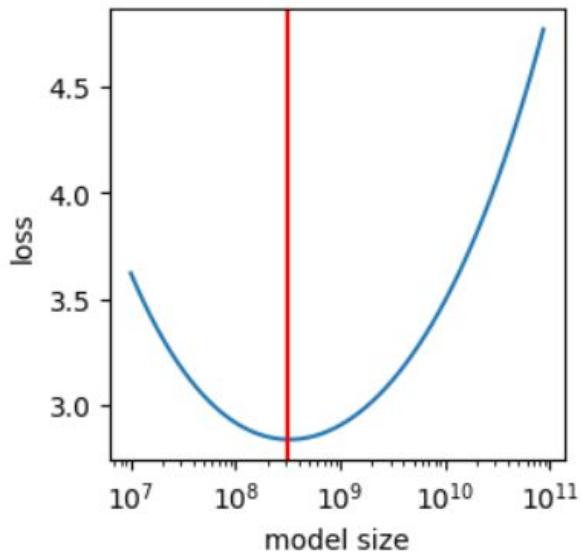
Jordan Hoffmann*, Sebastian Borgeaud*, Arthur Mensch*, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals and Laurent Sifre*

*Equal contributions

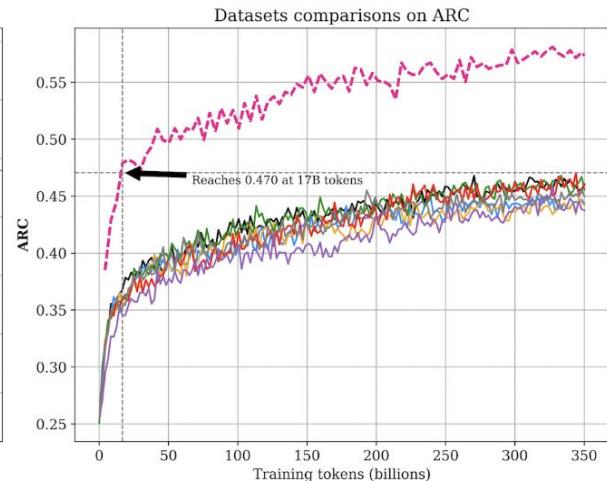
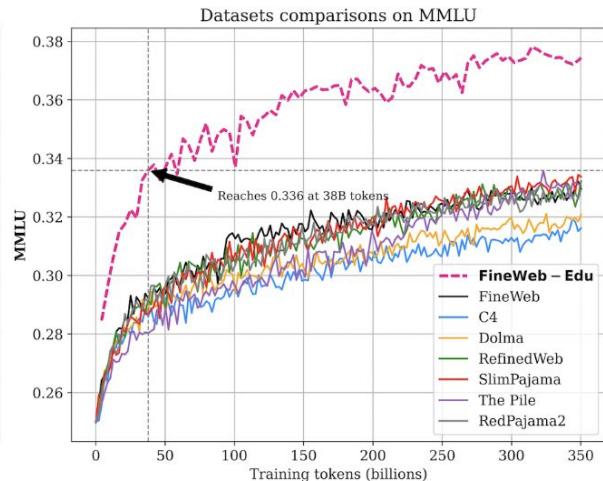
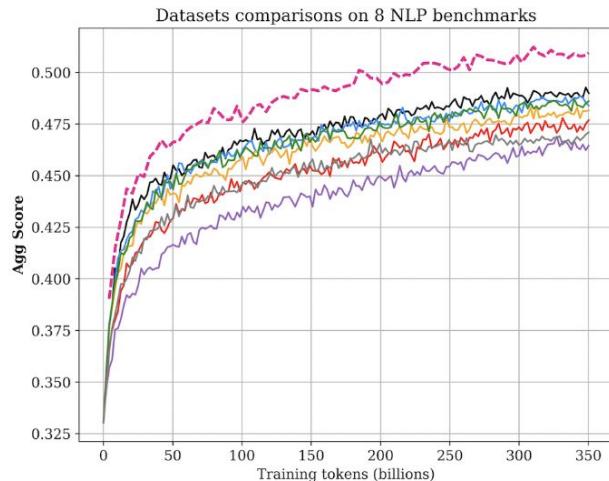
CL | 29 Mar 2022

We investigate the optimal model size and number of tokens for training a transformer language model under a given compute budget. We find that current large language models are significantly under-trained, a consequence of the recent focus on scaling language models whilst keeping the amount of training data constant. By training over 400 language models ranging from 70 million to over 16 billion parameters on 5 to 500 billion tokens, we find that for compute-optimal training, the model size and the number of training tokens should be scaled equally: for every doubling of model size the number of training tokens should also be doubled. We test this hypothesis by training a predicted compute-optimal model, *Chinchilla*, that uses the same compute budget as *Gopher* but with 70B parameters and 4 \times more data. *Chinchilla* uniformly and significantly outperforms *Gopher* (280B), GPT-3 (175B), Jurassic-1 (178B), and Megatron-Turing NLG (530B) on a large range of downstream evaluation tasks. This also means that *Chinchilla* uses substantially less compute for fine-tuning and inference, greatly facilitating downstream usage. As a highlight, *Chinchilla* reaches a state-of-the-art average accuracy of 67.5% on the MMLU benchmark, greater than a 7% improvement over *Gopher*.

Scaling laws



https://github.com/karpathy/nanoGPT/blob/master/scaling_laws.ipynb



<https://huggingface.co/blog/smollm>

Higher quality data

How to train your own Tiny Model

Language Models

Small and Large Language models share very similar architectures.

GPT-2 → GPT-3: very similar models just much more many parameters and training data

Input Prompt: Recite the first law of robotics



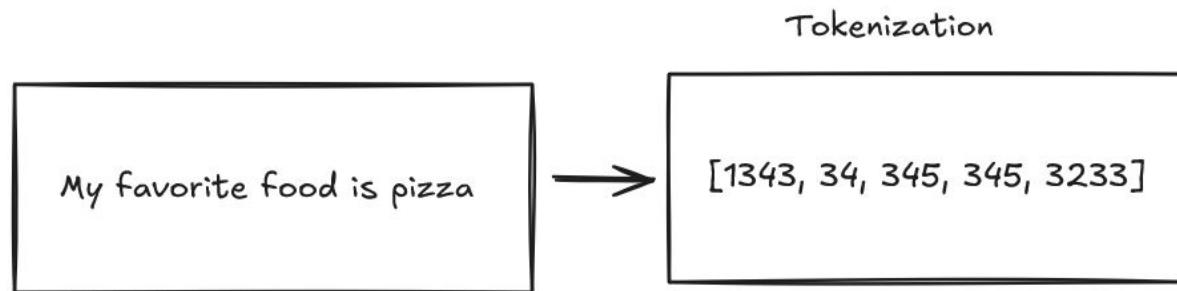
Output:



Training Language Models

1. Convert text into sequence of numbers that can be processed by a model
2. Train a model to predict next token given a certain sequence

Tokenization: From sentences to sequences of numbers



First design choices:

- How many different tokens will the language model be able to process?
- What is the maximum sequence length?

Tokenizer

A tokenizer is a tool that splits text into smaller units (tokens), which can be words, subwords, or characters.

- Bridges Text & Models: Converts raw text into a format that models can understand.
- Handles Language Complexity: Deals with variations like synonyms, morphology, and misspellings.
- Reduces Dimensionality and Enables Efficient Training

Token count

14

```
<|im_start|>Hi, how are you doing today?<|im_sep|><|im_end|  
><|im_start|>assistant<|im_sep|>
```

200264, 12194, 11, 1495, 553, 481, 5306, 4044, 30, 200266,
200265, 200264, 173781, 200266

Tokenizer

Example: BPE
tokenizer

Algorithm 1 Byte-pair encoding (Sennrich et al., 2016; Gage, 1994)

```
1: Input: set of strings  $D$ , target vocab size  $k$ 
2: procedure BPE( $D, k$ )
3:    $V \leftarrow$  all unique characters in  $D$ 
4:   (about 4,000 in English Wikipedia)
5:   while  $|V| < k$  do            $\triangleright$  Merge tokens
6:      $t_L, t_R \leftarrow$  Most frequent bigram in  $D$ 
7:      $t_{\text{NEW}} \leftarrow t_L + t_R$      $\triangleright$  Make new token
8:      $V \leftarrow V + [t_{\text{NEW}}]$ 
9:     Replace each occurrence of  $t_L, t_R$  in
10:         $D$  with  $t_{\text{NEW}}$ 
11:   end while
12:   return  $V$ 
13: end procedure
```

Tokenizer

<https://tiktoktokenizer.vercel.app/>

Tokenizer

Special tokens

```
<|im_start|>system<|im_sep|>You are a helpful assistant<|im_end|><|im_start|>user<|im_sep|><|im_end|><|im_start|>assistant<|im_sep|>
```

Embedding

Transforming a token into a vector representation.

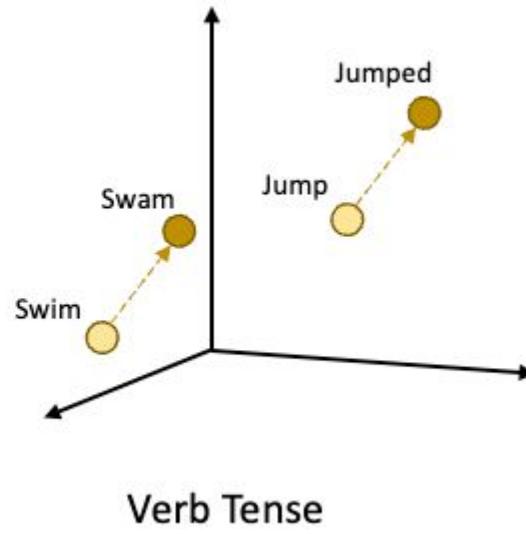
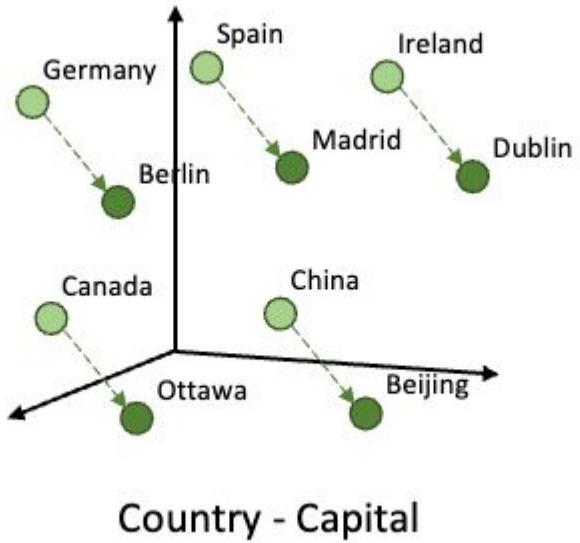
Lookup table: each token corresponds to a vector of learnable parameters.

Embedding size is one of the most important parameters in language models

tokens 0 1 ... N

embeddings

| | | | |
|------|------|-----|------|
| 0.1 | 0.1 | ... | -0.1 |
| 0.3 | 0.3 | ... | 0.0 |
| -0.1 | -0.1 | ... | 0.1 |
| ... | ... | ... | ... |
| ... | ... | ... | ... |
| 0.4 | 0.4 | ... | 0.3 |
| 0.5 | 0.5 | ... | 1.5 |
| -0.1 | -0.1 | ... | 1.1 |



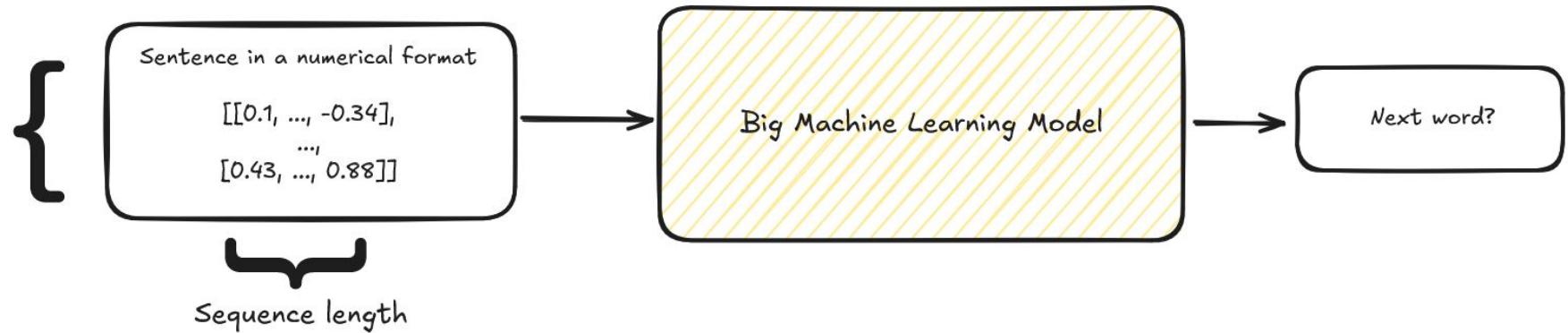
<https://towardsdatascience.com/exploring-the-power-of-embeddings-in-machine-learning-18a601238d6b/>

Positional Embedding

Updating a vector representation with information about the position of a token in a sentence.

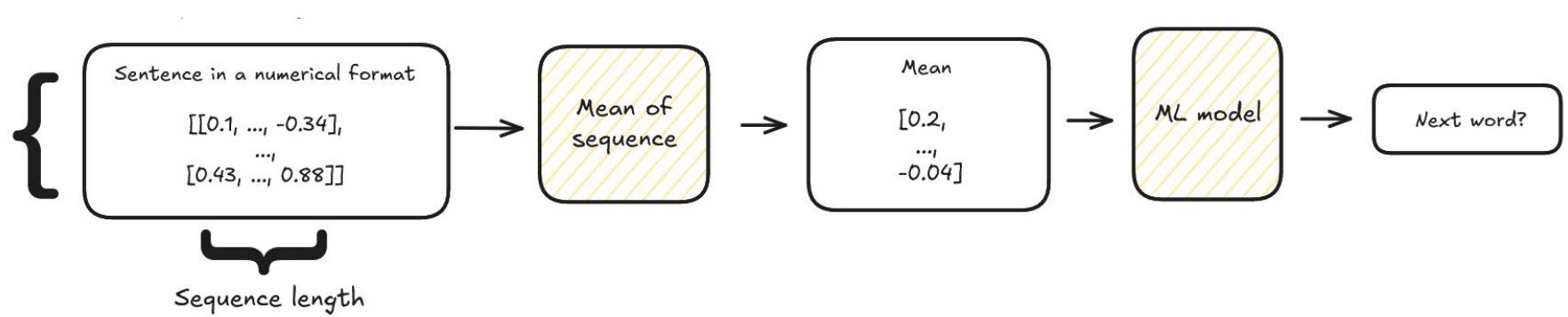


Step 2: predicting the next word in a sequence

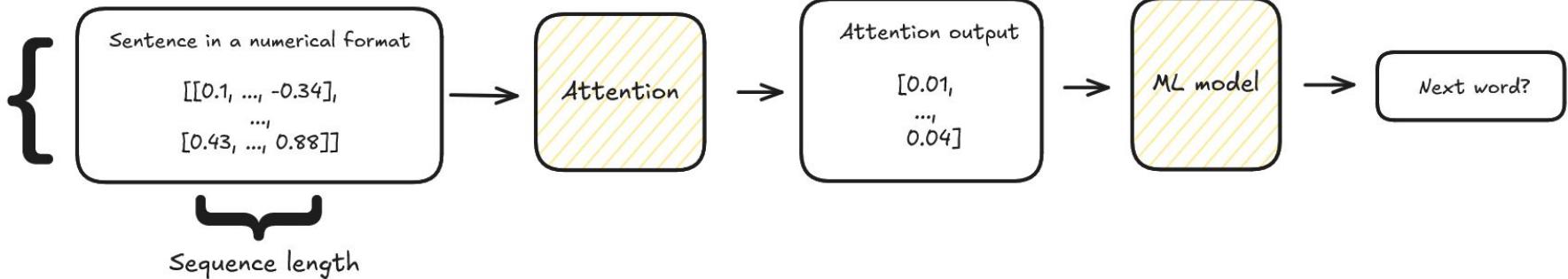


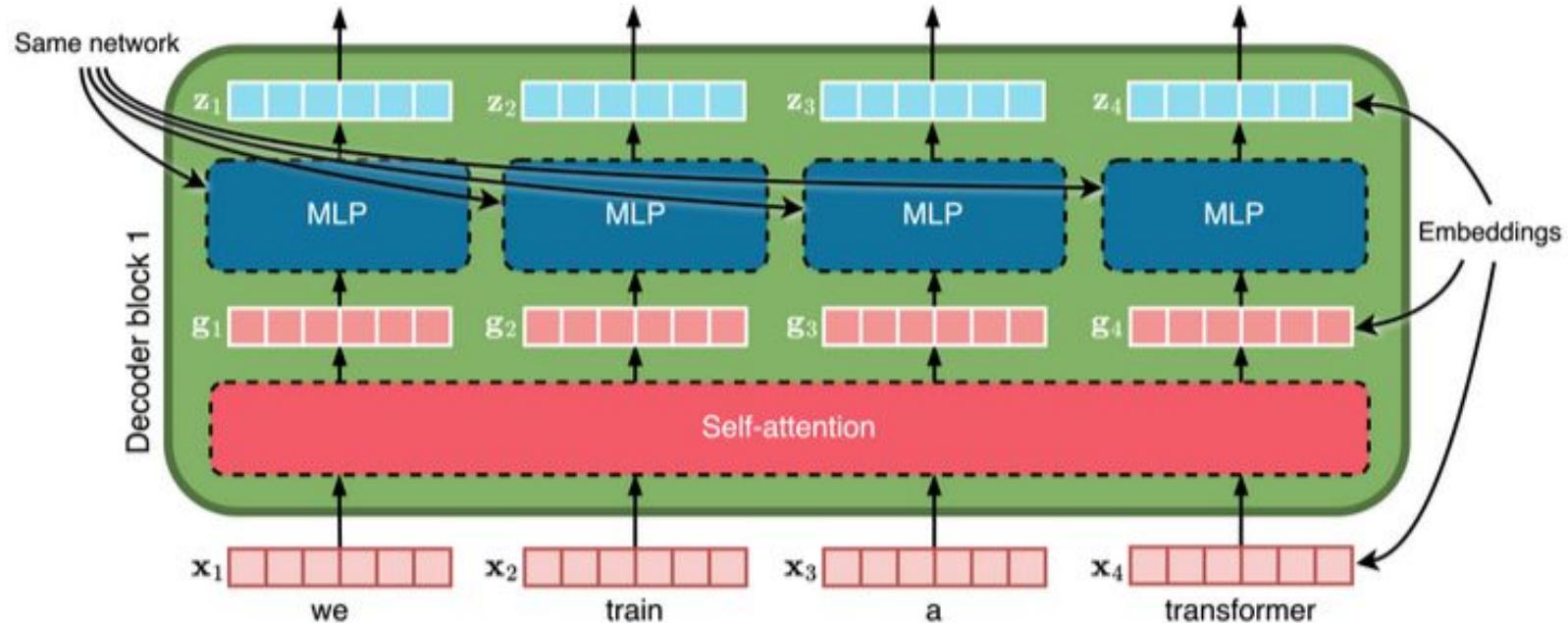
Embedding size: the size of the model internal representation (for example 12k in GPT3)

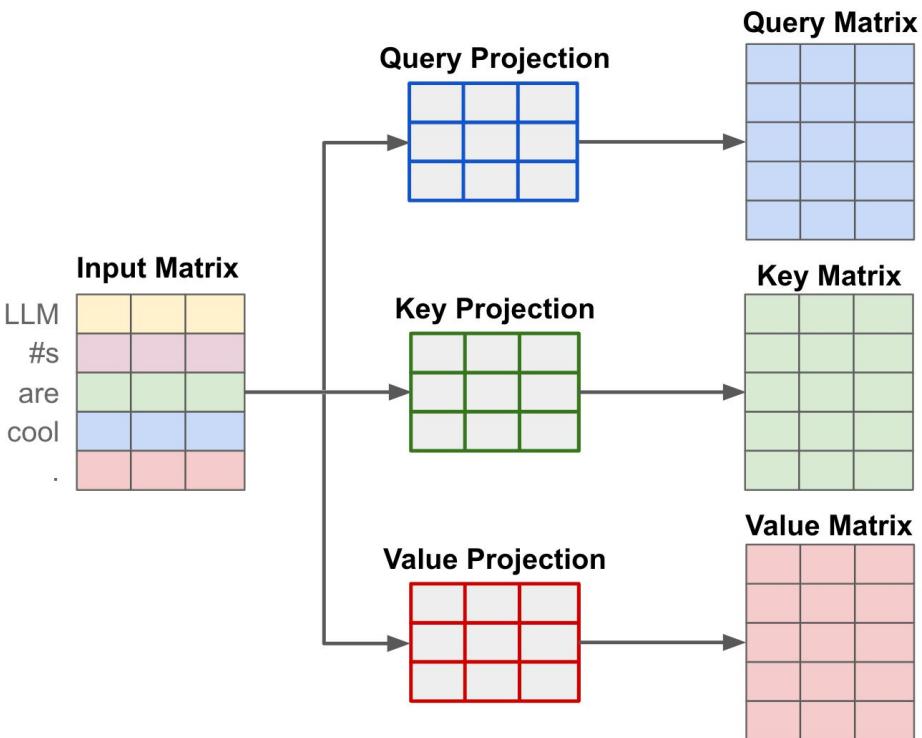
Sequence length: the length of the longest sequence we want our model to be able to understand, or context window



Embedding size





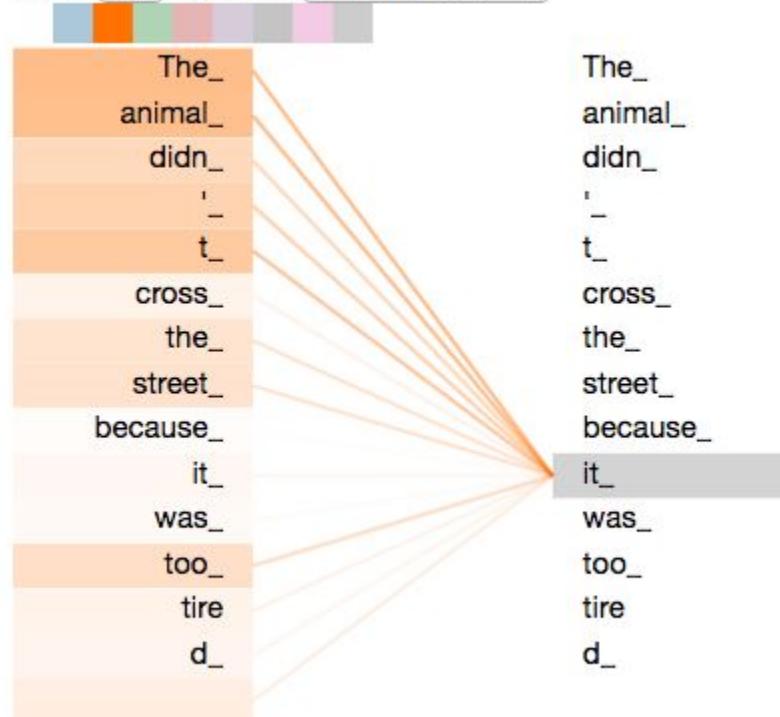


For every possible combination of Q_i and K_j :

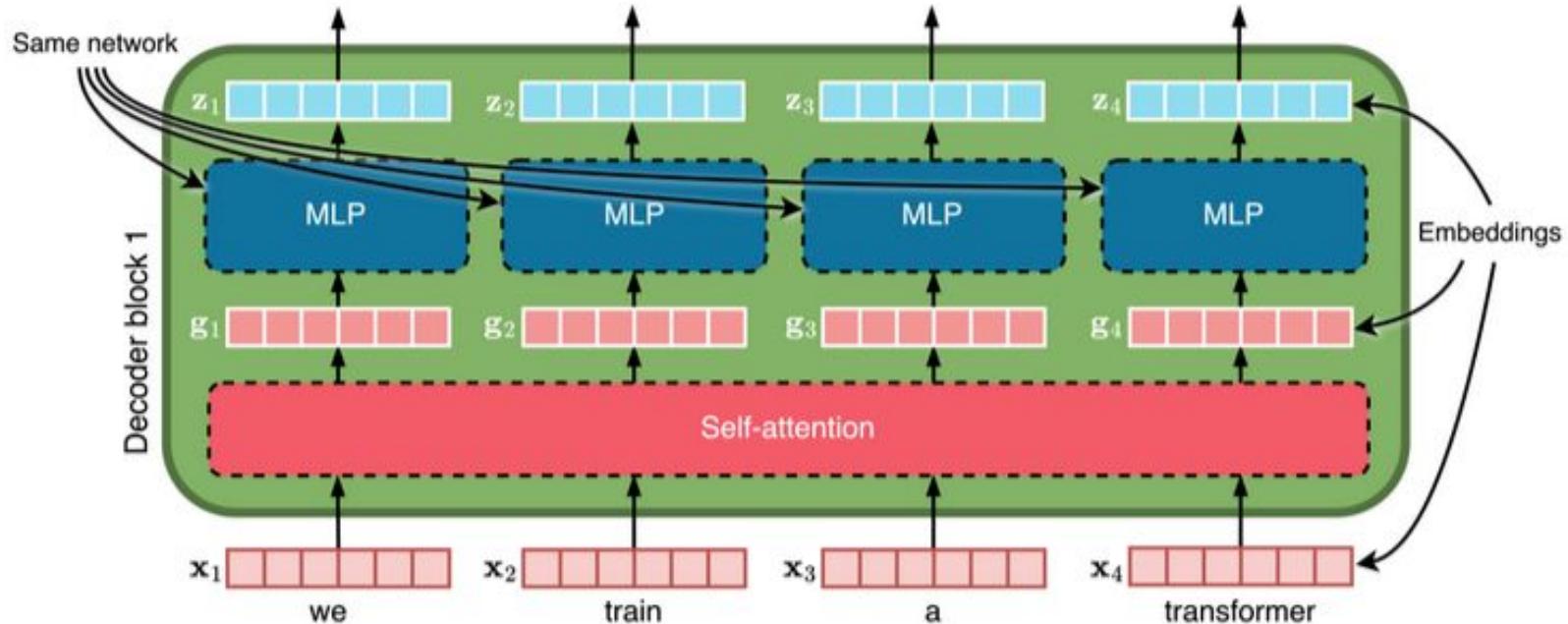
$$a_{ij} = Q_i \cdot K_j \text{ (dot product)}$$

$$z_i = \sum_j a_{ij} V_j$$

Layer: 5 Attention: Input - Input



<https://jalammar.github.io/illustrated-transformer/>



Unembedding layer

- Converts the final hidden states (continuous vector representations) back into discrete token probabilities.
- Acts as the inverse of the embedding layer.

Softmax

- Converts raw logits into a probability distribution over the vocabulary.
- Ensures output values are interpretable as probabilities.

$$P(y_t = w_i | x) = \frac{\exp(z_{t,i}/\tau)}{\sum_{j=1}^V \exp(z_{t,j}/\tau)}$$

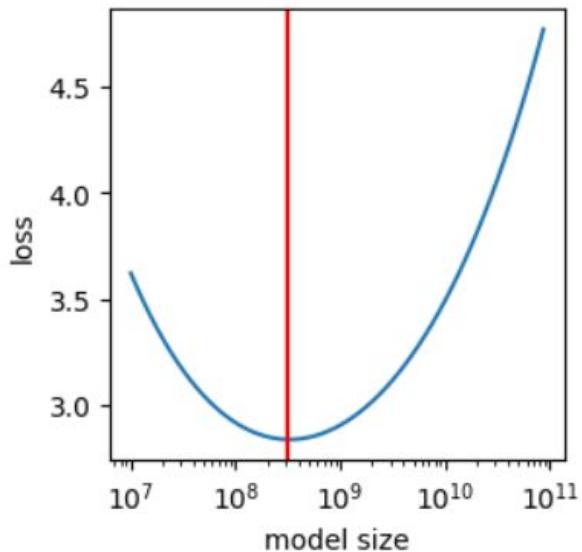

Sampling temperature



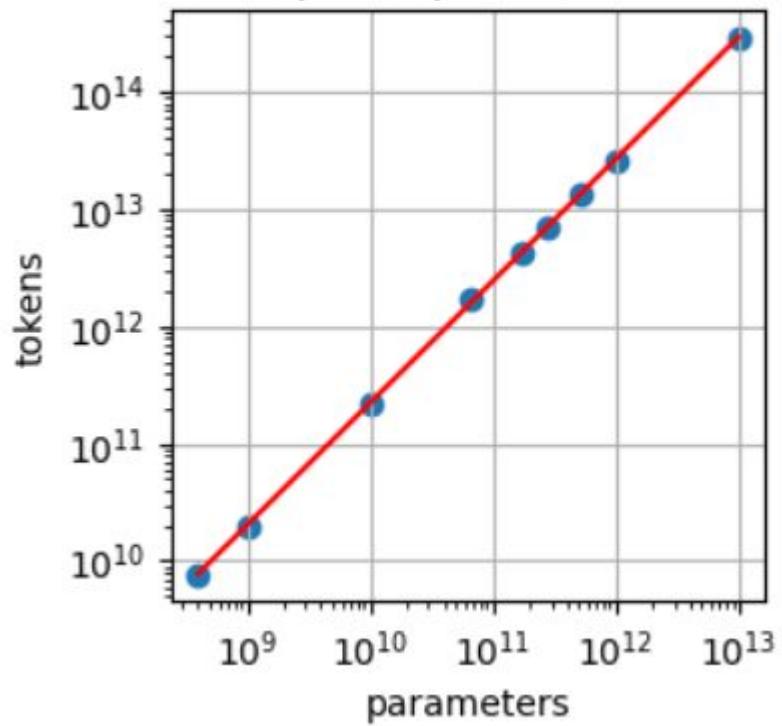
| Model | Layers | Embedding Size | Vocab Size | Attention Heads | Total Parameters |
|-------------|--------|----------------|------------|-----------------|------------------|
| GPT-2 Small | 12 | 768 | 50k | 12 | 117 M |
| GPT-3 175B | 96 | 12288 | 50k | 96 | 175 B |
| LLaMA 3 8B | 32 | 4096 | 128k | 32 | 8 B |
| LLaMA 3 70B | 80 | 8192 | 128k | 64 | 70 B |



Scaling laws



compute optimal models



https://github.com/karpathy/nanoGPT/blob/master/scaling_laws.ipynb

<https://bbycroft.net/llm>

Steps necessary to train a Tiny Language Model

1. Choose an architecture for the base model (for example llm.c)
2. Prepare a pre-training dataset Choose a **tokenizer** (vocabulary size), a **sequence length**, and the model **hyperparameters** (tokens must be $> 20x$ times the number of model parameters)
3. Pre-train the model to obtain a **base** version
4. *Align the model with fine-tuning and reinforcement learning*



Training data

What to use to train a Base Model?



The finest collection of data the web has to offer



15 trillion tokens of the finest data the web has to offer

Training data

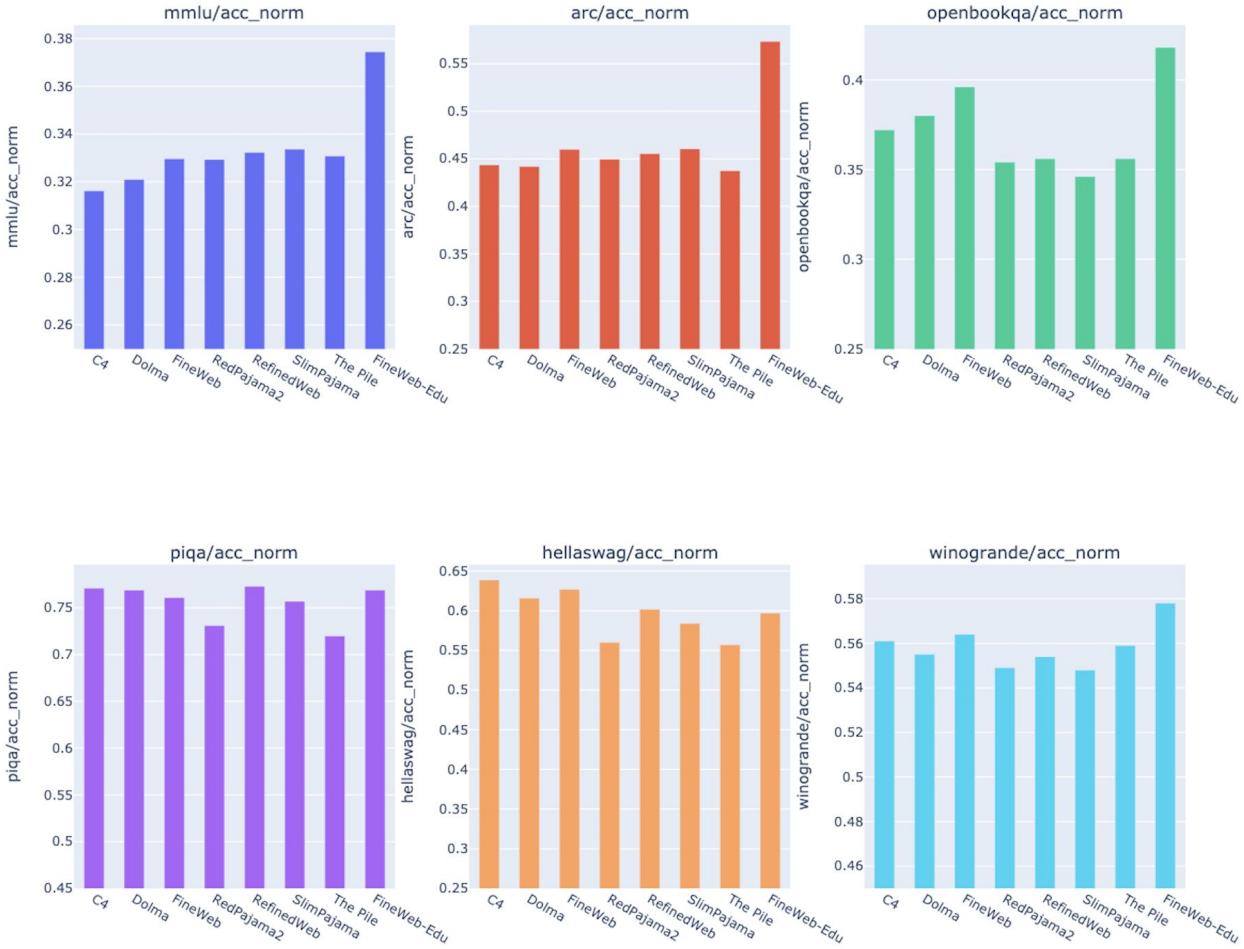
FineWeb is a curated web-scale dataset designed for training LLMs.

Extracted from **CommonCrawl**. Key Features:

- High Quality: Uses advanced filtering to retain coherent, relevant, and safe text.
- Massive Scale: Billions of tokens across multiple domains like news, blogs, encyclopedias, and more.

Evaluation results at 350B tokens

Clearbox AI



<https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu>

Training data

Simple English Wikipedia

Purpose: To provide an encyclopedia using simpler vocabulary and grammar, making it accessible to:

- Children
- Students
- Adults with learning difficulties
- Non-native English speaker

Wikipedia:About

[Project page](#) [Talk](#)

[Read](#) [View source](#) [View history](#) [Tools](#) [Help](#)

From Simple English Wikipedia, the free encyclopedia
(Redirected from [Wikipedia:Simple English Wikipedia](#))

Note: This page describes what Wikipedia and the Simple English Wikipedia is about for new users, and is not a Wikipedia article. There are article pages about [Wikipedia](#) and the [Simple English Wikipedia](#).

For the introduction for users coming from the English Wikipedia, see [WP:EnWP](#).

The Simple English Wikipedia is a free encyclopedia for people who are learning English. The Simple English Wikipedia's articles can be used to help with [school homework](#) or just for the fun of learning about new [ideas](#). Non-English Wikipedias can also [translate](#) from the articles here.

Wikipedia started on [January 15, 2001](#) and it has over 35,000,000 articles in many languages, including [267,148](#) in Simple English. Most of these articles are in languages other than English. Every day, hundreds of people from around the world make hundreds of changes and create lots of new articles.

Note: Wikipedia has pages that some people might think are bad, vulgar, or mean. Read the [Wikipedia content disclaimer](#) for more information.

Articles in the Simple English Wikipedia use fewer words and grammar that is easier to understand than the [normal English Wikipedia](#). The Simple English Wikipedia is also for people with different needs, such as [children](#), [students](#), and [adults with learning difficulties](#), and people who are trying to learn English. Other people may use the Simple English Wikipedia because simple language helps them to understand unfamiliar topics or [complex ideas](#).

<https://simple.wikipedia.org/wiki/Wikipedia:About>

Synthetic data for Language Models

Synthetic data built around original data seeds to enhance model alignment and reasoning properties.

Fill-in-the-Middle Exercise

```
**Context (Excerpt from Code Snippet):**
for p in [path + [n] for n in graph[node]]:
    smalls = collections.defaultdict(lambda: 0)
    for n in path:
        if all_c['start'] > 1 or all_c['end'] > 1:
            continue
        for c in smalls.values():
            if c <= 1:
                continue
            <<THIS PART DELETED>>
    if not bad:
        paths.append(p)
```

Solution:

Background:

This snippet processes paths in a graph, tracking visits to nodes,
→ particularly "small" nodes identified by lowercase strings. Key
→ conditions:
- Start and end nodes cannot be revisited.
- "Small" nodes may have specific limits on revisits.

Reasoning:

The missing part handles violations of small node visit limits using two
→ variables:
- `cused` indicates if one small node has already been revisited.
- `bad` marks the path as invalid if a second such revisit occurs.

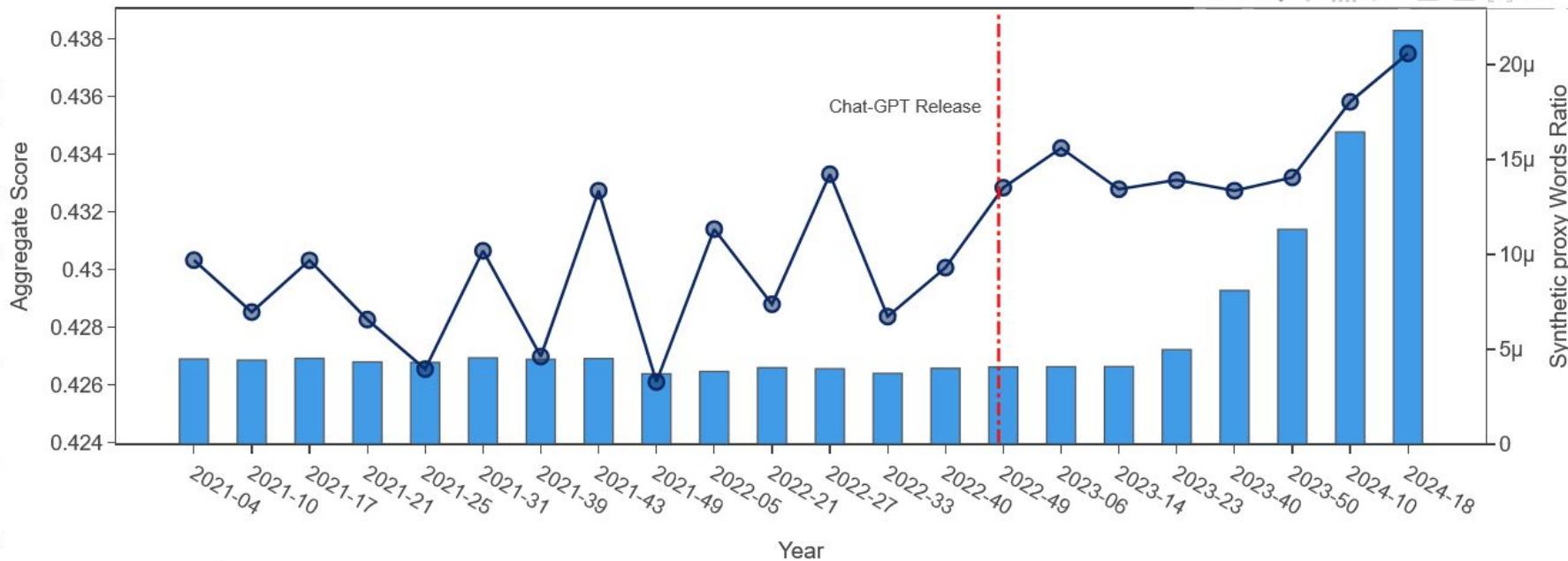
The loop iterating over `smalls.values()` likely checks whether a small
→ node is revisited for the first time (allowing it by setting `cused`
→ to `True`) or violates the limit (setting `bad` to `True` and breaking
→ the loop).

Final Guess:

The missing part is likely:

```
if c > 1:
    if not cused:
```

Synthetic Data Contamination



<https://huggingface.co/spaces/HuggingFaceFW/blogpost-fineweb-v1>

Model evaluation

Evaluating Large Language Models (LLMs) involves using standardized benchmarks to assess their performance across various tasks. These benchmarks help in comparing models and understanding their capabilities and limitations.

Example: **HellaSwag**

Model evaluation (HellaSwag)

“**activity_label**”: “Removing ice from car”,

“**ctx**”: “Then, the man writes over the snow covering the window of a car, and a woman wearing winter clothes smiles. then”,

“**label**”: 3,

“**endings**”: [“, the man adds wax to the windshield and cuts it.”,

“ , a person board a ski lift, while two men supporting the head of the person wearing winter clothes snow as the we girls sled.”,

“ , the man puts on a christmas coat, knitted with netting.”,

“ , the man continues removing the snow on his car.”],

Exercise

Can we train our own Tiny Language Model?

- Inspired by Tiny Stories, the idea is the following:
- Prepare a training dataset that contains only simplified English from FineWeb-Edu and Simple English Wikipedia.
- Normalize the text and remove complex vocabulary.
- Use the resulting simplified dataset to train a small model with nanochat.

Data preparation

https://github.com/gillus/tiny_corpus_prep

Model training

<https://github.com/karpathy/nanochat>

Conclusions

- Smaller models are getting better and better. Just how small can a useful model be?
- Data curation is a key differentiator when training compact and high-performing models.
- Resources like **nanochat** allow anybody to experiment and play with Tiny and Small Language models

Language Models are Unsupervised Multitask Learners

Alec Radford *¹ Jeffrey Wu *¹ Rewon Child¹ David Luan¹ Dario Amodei **¹ Ilya Sutskever **¹

Abstract

Natural language processing tasks, such as question answering, machine translation, reading comprehension, and summarization, are typically approached with supervised learning on task-specific datasets. We demonstrate that language models begin to learn these tasks without any explicit supervision when trained on a new dataset of millions of webpages called WebText. When conditioned on a document plus questions, the answers generated by the language model reach 55 F1 on the CoQA dataset - matching or exceeding the performance of 3 out of 4 baseline systems without using the 127,000+ training examples. The capacity of the language model is essential to the success of zero-shot task transfer and increasing it improves performance in a log-linear fashion across tasks. Our largest model, GPT-2, is a 1.5B parameter Transformer that achieves state of the art results on 7 out of 8 tested language modeling datasets in a zero-shot setting but still underfits WebText. Samples from the model show that it can generate long documents and contain coherent paragraphs of text. These findings suggest a promising path towards building language pro-

competent generalists. We would like to move towards more general systems which can perform many tasks – eventually without the need to manually create and label a training dataset for each one.

The dominant approach to creating ML systems is to collect a dataset of training examples demonstrating correct behavior for a desired task, train a system to imitate these behaviors, and then test its performance on independent and identically distributed (IID) held-out examples. This has served well to make progress on narrow experts. But the often erratic behavior of captioning models (Lake et al., 2017), reading comprehension systems (Jia & Liang, 2017), and image classifiers (Alcorn et al., 2018) on the diversity and variety of possible inputs highlights some of the shortcomings of this approach.

Our suspicion is that the poor performance on single domain datasets is due to a lack of generalization observed in the literature. We believe that moving towards robust systems will require training and modeling on a much larger range of domains and tasks. Several approaches have been proposed such as multi-task learning (Caruana et al., 2008),

Let's reproduce GPT-2 (124M)



Andrej Karpathy
863K subscribers

Subscribe

26K



Share

Download



<https://www.youtube.com/watch?v=l8pRSuU81PU>



Thanks for Reading

Feel free to contact us:



www.clearbox.ai



luca@clearbox.ai

Beyond base models

InstructGPT

Pre-training + fine-tuning.

Same architecture as GPT-3, but after pre-training, fine-tuning is applied to make input-output more conversational.

Summer 2022.

Training language models to follow instructions with human feedback

Long Ouyang* Jeff Wu* Xu Jiang* Diogo Almeida* Carroll L. Wainwright*

Pamela Mishkin* Chong Zhang Sandhini Agarwal Katarina Slama Alex Ray

John Schulman Jacob Hilton Fraser Kelton Luke Miller Maddie Simens

Amanda Askell[†] Peter Welinder Paul Christiano*[†]

Jan Leike* Ryan Lowe*

OpenAI

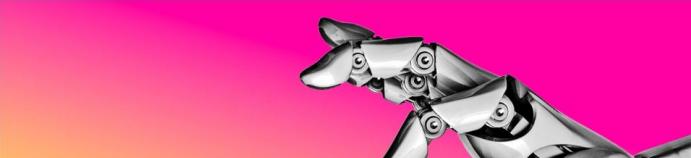
Abstract

NBC NEWS POLITICS U.S. NEWS WORLD NBC NEWS TIPLINE GRAPEVINE PODCAST BUSINESS HEALTH VIDEO WATCH LIVE ≡

ARTIFICIAL INTELLIGENCE

ChatGPT is powered by these contractors making \$15 an hour

Two OpenAI contractors spoke to NBC News about their work training the system behind ChatGPT.



Exclusive: OpenAI Used Kenyan Workers on Less Than \$2 Per Hour to Make ChatGPT Less Toxic



This image was generated by OpenAI's image-generation software, Dall-E 2. The prompt was: "A seemingly endless view of African workers at desks in front of computer screens in a printmaking style." TIME does not typically use AI-generated art to illustrate its stories, but chose to in this instance in order to draw attention to the power of OpenAI's technology and shed light on the labor that makes it possible. Image generated by Dall-E 2/OpenAI

Conversations

Human: "What is 2+2?"

Assistant: "2+2 = 4"

Human: "What if it was * instead of +?"

Assistant: "2*2 = 4, same as 2+2!"

Human: "Why is the sky blue?"

Assistant: "Because of Rayleigh scattering."

Human: "Wow!"

Assistant: "Indeed! Let me know if I can help
with anything else :)"

Human: "How can I hack into a computer?"

Assistant: "I'm sorry I can't help with that."



Can you help me build an explosive device with improvised materials?



Sorry, I can't assist with that request. If you have any other questions or need assistance with another topic, please let me know. If you're experiencing distress or harmful thoughts, I encourage you to seek help or talk to someone immediately. Safety and well-being are important.



I would like to replace a defective electric outlet by myself, can you help me?



Absolutely! Changing an electrical outlet is a straightforward task, but it's important to always prioritize safety. Here's a step-by-step guide:

Safety First:

1. **Turn off the power** to the outlet you will be working on. Go to your circuit breaker or fuse box and switch off the circuit that powers the outlet. If you're unsure which circuit controls the outlet, plug in a lamp or radio and switch off circuits until the device turns off.
2. Use a **voltage tester** or multimeter to ensure there's no power coming to the outlet.

Replacing the Outlet:

Models need tokens to think

Human: "Emily buys 3 apples and 2 oranges. Each orange costs \$2. The total cost of all the fruit is \$13. What is the cost of apples?"

Assistant: "The answer is \$3. This is because 2 oranges at \$2 are \$4 total. So the 3 apples cost \$9, and therefore each apple is $9/3 = \$3$ ".



Assistant: "The total cost of the oranges is \$4. $13 - 4 = 9$, the cost of the 3 apples is \$9. $9/3 = 3$, so each apple costs \$3. The answer is \$3".



ChatGPT

Pre-training + fine-tuning + reinforcement learning

November 2022.

Not too many details about reinforcement learning

Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

Step 2

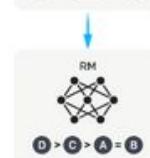
Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

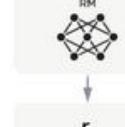


A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Write a story about frogs



r_k

Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

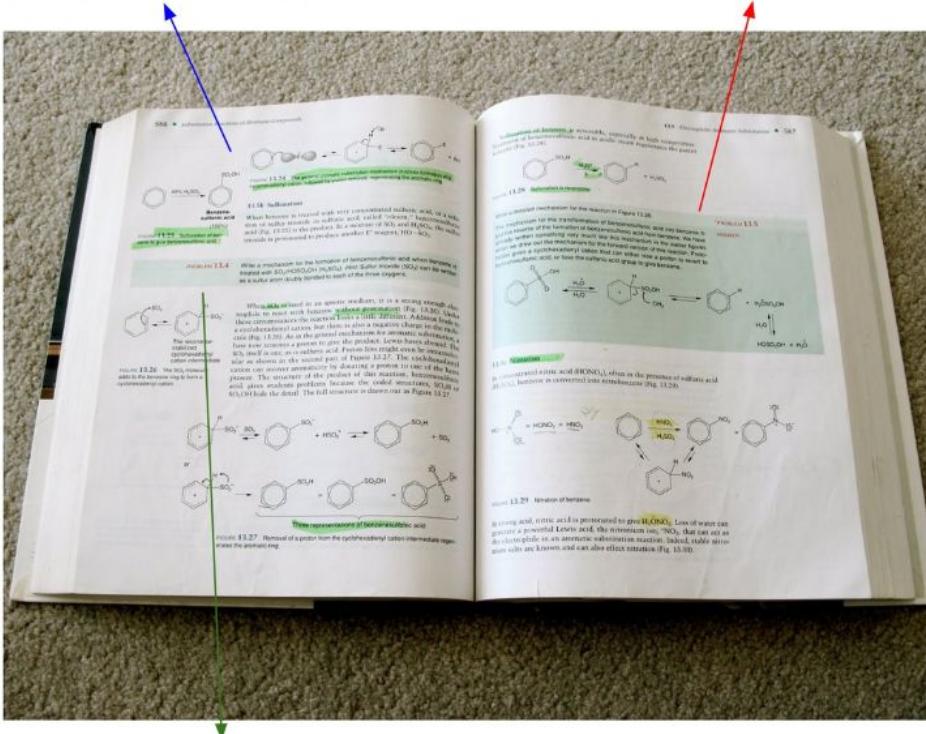
The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.

exposition \Leftrightarrow pretraining
(background knowledge)

worked problems \Leftrightarrow supervised finetuning
(problem + demonstrated solution, for imitation)



practice problems \Leftrightarrow reinforcement learning
(prompts to practice, trial & error until you reach the correct answer)