

NLP in Real World Settings: Healthcare & Program Synthesis

Ndapa Nakashole

13 July, 2023



UCSDCSE
Computer Science and Engineering

Asking Health Questions

For labor induction, what is the medication that is FDA approved for cervical ripening?

ChatGPT's answer (GPT-3.5): **Misoprostol**



Misoprostol is the medication that is commonly used and FDA approved for cervical ripening in preparation for labor induction. It is an approved medication for this purpose and is frequently used in clinical settings for inducing labor.

Misoprostol (marketed as Cytotec) Information

[f Share](#) [t Tweet](#) [in LinkedIn](#) [✉ Email](#) [🖨 Print](#)

FDA ALERT – Risks of Use in Labor and Delivery

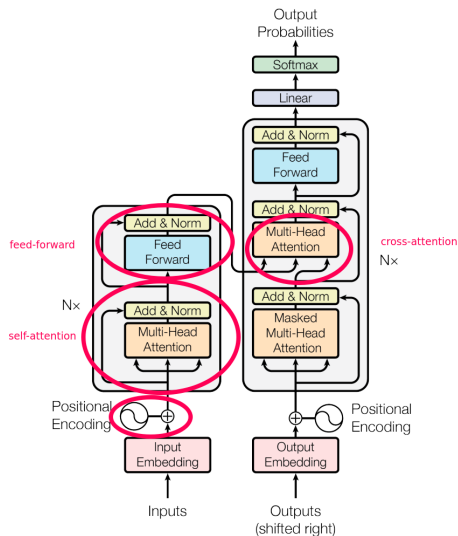
This Patient Information Sheet is for pregnant women who may receive misoprostol to soften their cervix or induce contractions to begin labor. Misoprostol is sometimes used to decrease blood loss after delivery of a baby. These uses are not approved by the FDA. No company has sent the FDA scientific proof that misoprostol is safe and effective for these uses.

There can be serious side effects, including a torn uterus (womb), when misoprostol is used for labor and delivery. A torn uterus may result in severe bleeding, having the uterus removed (hysterectomy), and death of the mother or baby. These side effects are more likely in women who have had previous uterine surgery, a previous Cesarean delivery (C-section), or several previous births.

Content current as of:
07/10/2015

Correct answer: [Dinoprostone](#)
(GPT-4 get is right)

Transformer knowledge storage mechanism

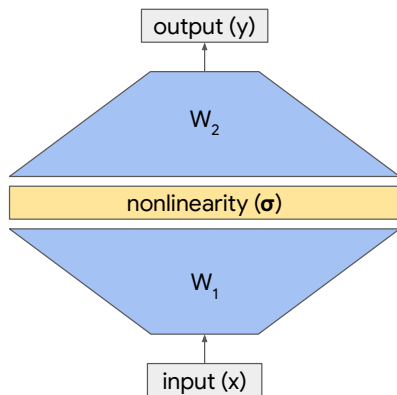


► Where is the incorrect belief stored?

- Token embeddings?
- Feedforward layers?
- Attention layers?

Transformer storage mechanism

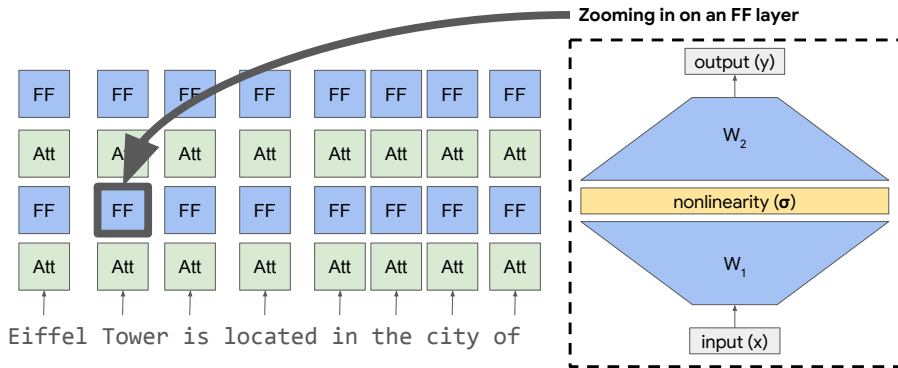
Finding¹: Feed-forward layers in Transformer LMs emulate **key-value memories**



$$y = W_2 \sigma(W_1 x)$$

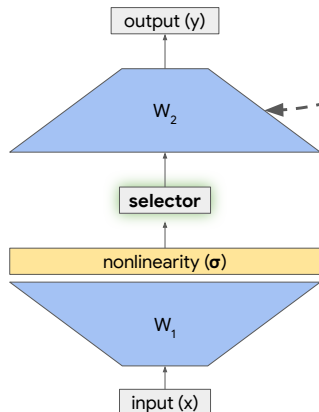
¹[Geva et al.(2021)Geva, Schuster, Berant, and Levy]

Modifying the memory



¹Figure from: K. Guu

Modifying the memory



Intuition: modify **columns** of W_2 to change model's behavior.

Subtract word vector for Paris, add word vector for Rome?

([Dai et al. 2021](#))

[Meng et al. 2022](#): apply a rank-1 update.

- $W_2 \leftarrow W_2 + uv^T$ (\mathbf{u} and \mathbf{v} are vectors)
- Maximize probability of outputting Rome when we see "Eiffel Tower" selector.
- Minimize change in behavior of W_2 on other inputs.

Editing approaches are still in their infancy.

¹Figure from: K. Guu

Desired from the Transformer

- ▶ **Attribution and interpretability**

- ◉ Trace a model's knowledge back to a particular document or training example

- ▶ **Modular knowledge editing**

- ◉ Update the knowledge the model has access to without breaking its behavior

- ▶ **Efficient scaling**

- ◉ Increase the model's memory size by 10x without paying 10x more compute.

Memory-Augmented Models

The Memory Extender

- ▶ Allow LM to interact with external memory (search engine, DB, etc):
- ▶ Thus supporting:
Editing of knowledge,
Attribution, Efficient scaling

Many approaches have been proposed:

- ▶ Memory Networks
- ▶ REALM
- ▶ DPR
- ▶ LaMDA
- ▶ WebGPT
- ▶ ...
- ▶ RETRO

Retrieval-Enhanced Transformer (RETRO), Borgeaud et al. (ICML 2022)

Improving language models by retrieving from trillions of tokens

Sebastian Borgeaud[†], Arthur Mensch[†], Jordan Hoffmann[†], Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae[‡], Erich Elsen[‡] and Laurent Sifre^{†,‡}

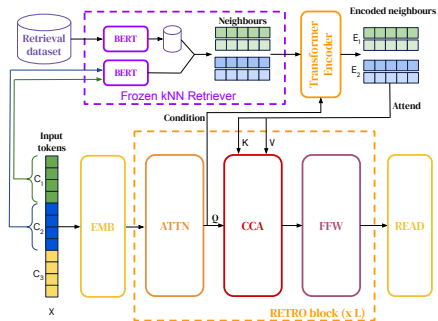
All authors from DeepMind, [†]Equal contributions, [‡]Equal senior authorship

- ▶ **Memory: a 2 trillion token database** → comparable performance to GPT-3 and others, but with 25× **fewer parameters**

2

RETRO Method

- ▶ Construct a **key-value database**, where values store raw **chunks** of text tokens
- ▶ Keys, values derived from a frozen BERT
- ▶ Training sequences are divided into chunks and augmented with their *k*-nearest neighbor chunks from the database.
- ▶ Encoder-decoder ,integrates retrieval chunks using **cross-attention**



Design Considerations for Memory-Augmented Models

- ▶ What goes o the **memories**?
 - ⊙ Documents, database records, training examples, etc
- ▶ How to **use** the retrieved memories?
 - ⊙ "Text fusion", "label smearing"
- ▶ How to **retrieve** memories?
 - ⊙ Use an off-the-shelf search engine
 - ⊙ How to train your own memory retriever

Case Study # 1: Healthcare QA

Memory models for Healthcare QA

- ▶ **Our memory representation:** Question-Answer pairs (ACL 2021, Mrini et al)
- ▶ QA pairs are **concise**, and also tend to **express importation information**

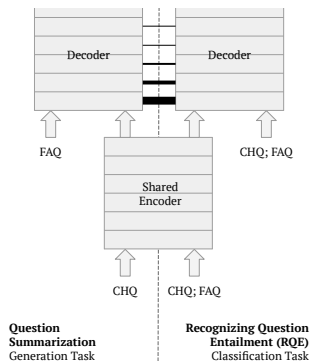
Consumer Health Question (CHQ) Answering

(CHQ) I am a new mother, blessed with a beautiful baby just a few weeks ago. While this journey has been an incredible one filled with love and joy, it's also been quite overwhelming at times. I am concerned about postpartum depression (PPD) as I might have it. I'm unsure about its exact causes and would really appreciate if someone could provide more clarity on this.

(FAQ) What is the physiological basis for postpartum depression?

Approach

- ▶ **Approach:** Key: FAQ; Value: FAQ answers
- ▶ For CHQ, an answer is obtained via **label smearing**
- ▶ A multi-task learning approach for CHQs, **summarize** and **match** perform matching via *question entailment*
- ▶ Encoder-decoder



ROUGE and human evaluation show effectiveness of approach & BioNLP shared task system submission placed competitively

QA-retrieval limitations

Compositional Questions (-): Using label smearing precludes answering questions that can be solved with > 1 QA-pairs.

- ▶ What are the primary causes of infertility in women *that can be addressed through preventive measures*

Work addressing compositional questions with QA-pair memories

- ▶ QA Is the New Knowledge Representation: Question-Answer Pairs as Knowledge Bases

[Chen et al.(2022a)Chen, Cohen, Jong, Gupta, Presta, Verga, and Wieting]

- ▶ Augmenting Pre-trained Language Models with QA-Memory for Open-Domain Question Answering

[Chen et al.(2022b)Chen, Verga, Jong, Wieting, and Cohen]

- To answer q' , retrieve related QA pairs $(q_1, a_1), \dots, (q_k, a_k)$
 - Combine retrieved QA pairs with original question q' .
 - Transformer to fuse this information and generate an answer
- ▶ Effective usage of retrieved memories is an open problem

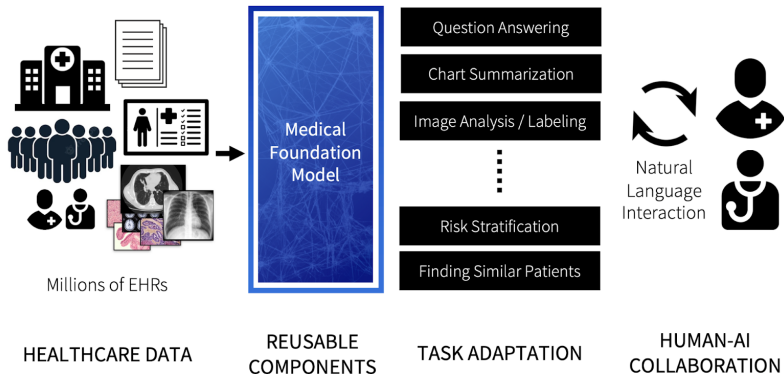
Memory Augmented Models: Summary

- ▶ Memory augmented models have attributes that make them particularly suitable for high-stakes domains
- ▶ Attribution to sources, editing, memory scale, efficient QA

Open questions:

- ▶ Choice of knowledge representation (entity embeddings? passages? QA pairs? ...)
- ▶ Effective usage of memories

Role of ChatGPT in Healthcare QA



- ▶ Generalization from fewer labeled examples; Conversational interface; Multimodality;

Case Study # 2: Programming

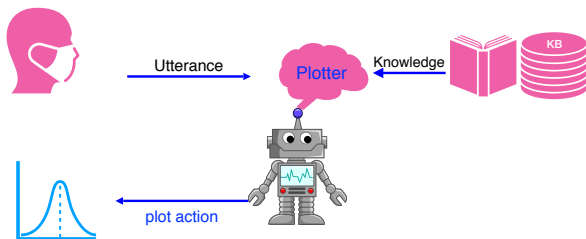
What is Program Synthesis?

- ▶ Program Synthesis: given a **natural language** specification, generate the corresponding **computer program**
 - ⊙ Improve programmer **productivity**
 - ⊙ Make software development more **accessible** to non-experts
 - ⊙ Enable **rapid prototyping**

Example: Plotting

Visualizing data and functions

- ▶ **Task:** Update plot from a natural language utterance



Learning Task

Utterance: "use the log scale for the X-axis"



```
1 import matplotlib.pyplot as plt
2 plt.plot(...)
3 plt.xscale('log')
4 plt.show()
```

- ▶ **Conversational:** plots can be specified in multiple-turns

Plot manipulation question: fonts



stackoverflow

About

Products

For Teams

How to change the font size on a matplotlib plot

Asked 10 years, 2 months ago Active 18 days ago Viewed 1.1m times



How does one change the font size for all elements (ticks, labels, title) on a matplotlib plot?

607

I know how to change the tick label sizes, this is done with:



```
import matplotlib
matplotlib.rc('xtick', labelsiz=20)
matplotlib.rc('ytick', labelsiz=20)
```



228



But how does one change the rest?

python

matplotlib

font-size

Plot manipulation question: vertical lines



stackoverflow

About

Products

For Teams

How to draw vertical lines on a given plot in matplotlib

Asked 6 years, 5 months ago · Active 2 months ago · Viewed 436k times



Given a plot of signal in time representation, how to draw lines marking corresponding time index?

302

Specifically, given a signal plot with time index ranging from 0 to 2.6(s), I want to draw vertical red lines indicating corresponding time index for the list `[0.22058956, 0.33088437, 2.20589566]`, how can I do it?



47

python

matplotlib

Plot manipulation question (3/3): invert y-axis



About

Products

For Teams

Reverse Y-Axis in PyPlot

Asked 11 years ago · Active 1 year, 4 months ago · Viewed 326k times



I have a scatter plot graph with a bunch of random x, y coordinates. Currently the Y-Axis starts at 0 and goes up to the max value. I would like the Y-Axis to start at the max value and go up to 0.

288



50



```
points = [(10,5), (5,11), (24,13), (7,8)]
x_arr = []
y_arr = []
for x,y in points:
    x_arr.append(x)
    y_arr.append(y)
plt.scatter(x_arr,y_arr)
```

python

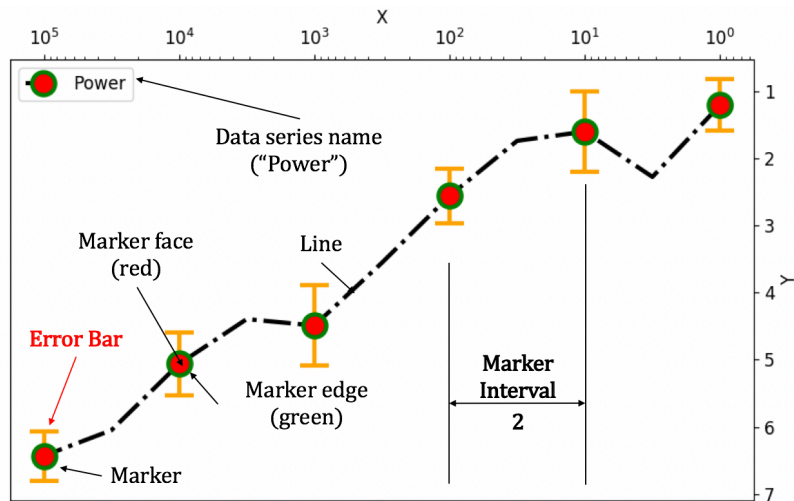
matplotlib

Conversational Plotting Agent

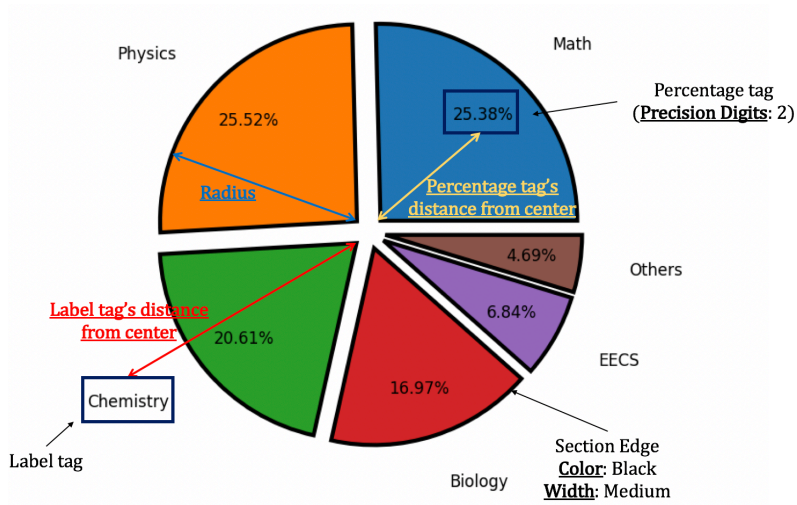
- ▶ A conversational plotting agent
 - ◉ **Knowledgable**: has access to relevant knowledge
 - ◉ **Conversational**: plots can be specified in multiple-turns

- ▶ Framed it as a problem of **goal-oriented dialog**
 - ◉ Slot-filling: fixed symbolic state representation

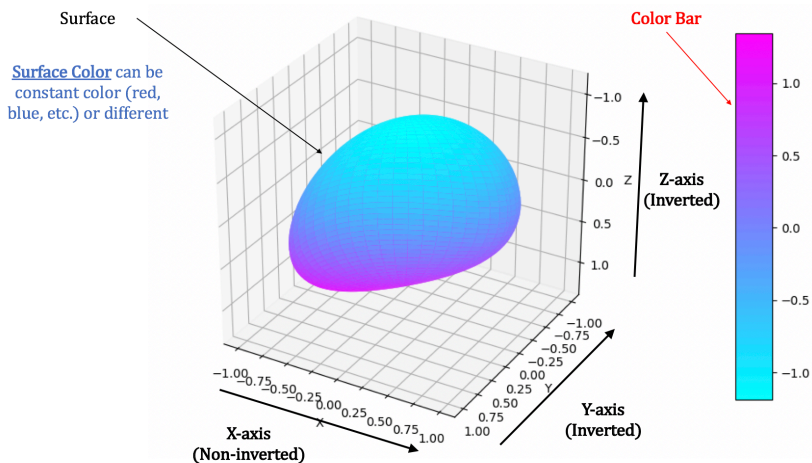
Slots: Line Chart



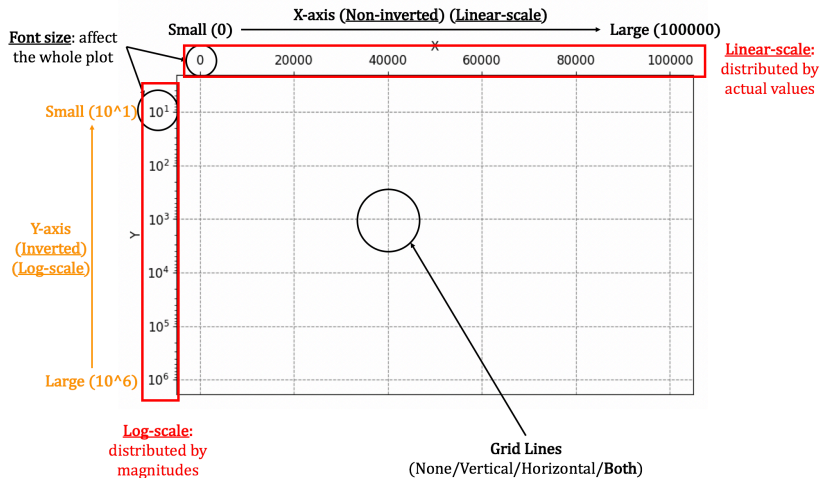
Slots: Pie Chart



Slots: 3D Surface



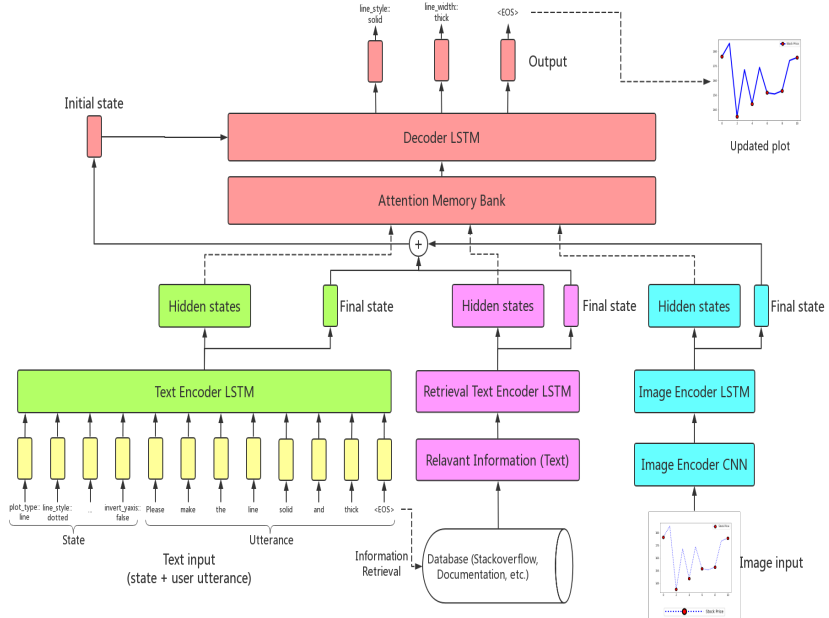
Shared Slots



	Plot Types	Slots
1.	Axes	Polarize, X-axis Scale, Y-axis Scale, X-axis Position, Y-axis Position, Invert X-axis, Invert Y-axis, Grid Line Type, Grid Line Style, Grid Line Width, Grid Line Color, Font Size
2.	3D Surface	Color map, Invert X-axis, Invert Y-Axis, Invert Z-Axis
3.	Bar Chart	Bar Orientation, Bar Height, Bar Face Color, Bar Edge Width, Bar Edge Color, Show Error Bar, Error Bar Color, Error Bar Cap Size, Error Bar, Cap Thickness, Data Series Name
4.	Contour/Filled	Contour Plot Type, Number of levels, Color Map, Color Bar Orientation, Color Bar Length, Color Bar Thickness
5.	Contour/Lined	Contour Plot Type, Lined Style, Line Width
6.	Histogram	Number of Bins, Bar Relative Width, Bar Face Color, Bar Edge Width, Bar Edge Color, Data Series Name
7.	Matrix	Color Map, Invert X-axis, Invert Y-axis
8.	Line Chart	Line Style, Line Width, Line Color, Marker Type, Marker Size, Marker Face Color, Marker Edge Color, Marker Interval, Data Series Name, Show Error Bar, Error Bar Color, Error Bar Cap Size, Error Bar Cap Thickness
9.	Pie Chart	Exploding Effect, Precision Digits, Percentage tags' distance from center, Label tag's distance from center, Radius, Section Edge Width, Section Edge Color
10.	Polar	Polarize, Grid Line Type, Grid Line Style, Grid Line Width, Grid Line Color, Font Size
11.	Scatter	Polarize, Marker Type, Marker Size, Marker Face Color, Marker Edge Width, Marker Edge Color, Color Map, Color Bar Orientation, Color Bar Length Color Bar Thickness
12.	Streamline	Density, Line Width, Line Color, Color Map, Arrow Size, Arrow Style

Table: Types and slots: all Matplotlib plot types are supported

Model



Model Performance

(Shao, & Nakashole, ACL 2020)

- ▶
 - ⊙ Human Performance: 80.6
 - ⊙ our Model: 61.3

LLMs that Code

ChatGPT Matplotlib(1/3): fonts



To change the font size for all elements (ticks, labels, and title) on a matplotlib plot, you can use the `matplotlib.rcParams` to modify the global parameters. Here's an example:



python

Copy code

```
import matplotlib.pyplot as plt

plt.rcParams.update({'font.size': 20}) # Set the font size for all element

# Example plot
plt.plot([1, 2, 3, 4], [1, 4, 9, 16])
plt.xlabel('X-axis label')
plt.ylabel('Y-axis label')
plt.title('Plot Title')

plt.show()
```

ChatGPT Matplotlib(2/3): vertical lines



To draw vertical red lines indicating specific time indices on a signal plot in matplotlib, you can use the `plt.axvline()` function. Here's an example:

python

Copy code

```
import matplotlib.pyplot as plt
import numpy as np

# Generate example signal data
time = np.linspace(0, 2.6, 1000)
signal = np.sin(2 * np.pi * time)

# Plot the signal
plt.plot(time, signal)

# Time indices for the vertical lines
time_indices = [0.22058956, 0.33088437, 2.20589566]

# Draw vertical red lines for the time indices
for idx in time_indices:
    plt.axvline(x=idx, color='red', linestyle='--')
```

ChatGPT Matplotlib(2/3): vertical lines: code works!

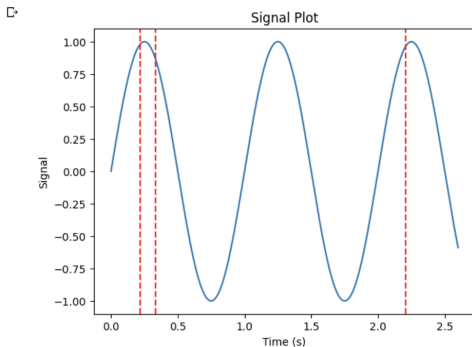
```
# Plot the signal
plt.plot(time, signal)

# Time indices for the vertical lines
time_indices = [0.22058956, 0.33088437, 2.20589566]

# Draw vertical red lines for the time indices
for idx in time_indices:
    plt.axvline(x=idx, color='red', linestyle='--')

# Set plot labels and title
plt.xlabel('Time (s)')
plt.ylabel('Signal')
plt.title('Signal Plot')

# Show the plot
plt.show()
```



ChatGPT Matplotlib (3/3): invert y-axis



To set the Y-axis to start at the maximum value and go down to 0 on a scatter plot in matplotlib, you can use the `plt.gca().invert_yaxis()` function. Here's an example incorporating your code snippet:



python

Copy code

```
import matplotlib.pyplot as plt

points = [(10, 5), (5, 11), (24, 13), (7, 8)]
x_arr = []
y_arr = []

for x, y in points:
    x_arr.append(x)
    y_arr.append(y)

plt.scatter(x_arr, y_arr)

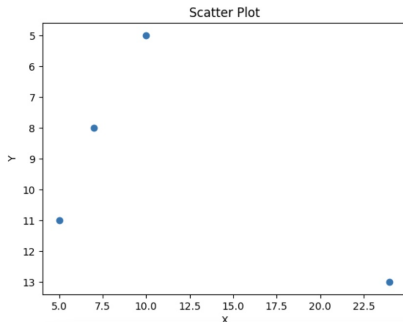
plt.gca().invert_yaxis() # Invert the Y-axis

plt.xlabel('X')
plt.ylabel('Y')
plt.title('Scatter Plot')

plt.show()
```

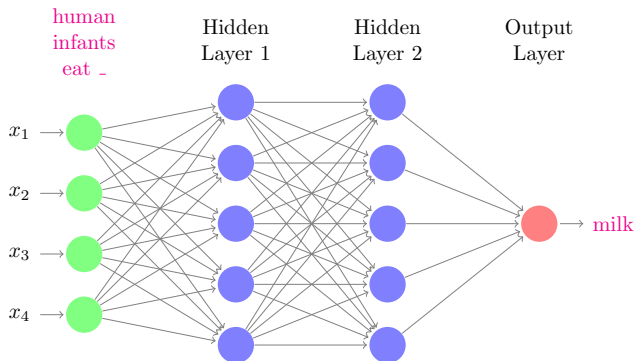
ChatGPT Matplotlib (3/3): invert y-axis, code works!

```
plt.gca().invert_yaxis() # Invert the Y-axis  
  
plt.xlabel('X')  
plt.ylabel('Y')  
plt.title('Scatter Plot')  
  
plt.show()
```



ChatGPT brings

- ▶ Generalization ... not just Matplotlib, and not just Python
- ▶ e.g., Tikz



OpenAI Codex: behind GitHub Copilot

Evaluating Large Language Models Trained on Code

Jul 2021

Mark Chen^{*1} Jerry Tworek^{*1} Heewoo Jun^{*1} Qiming Yuan^{*1} Henrique Ponde de Oliveira Pinto^{*1}
Jared Kaplan^{*2} Harri Edwards¹ Yuri Burda¹ Nicholas Joseph² Greg Brockman¹ Alex Ray¹ Raul Puri¹
Gretchen Krueger¹ Michael Petrov¹ Heidy Khlaaf³ Girish Sastry¹ Pamela Mishkin¹ Brooke Chan¹
Scott Gray¹ Nick Ryder¹ Mikhail Pavlov¹ Alethea Power¹ Lukasz Kaiser¹ Mohammad Bavarian¹
Clemens Winter¹ Philippe Tillet¹ Felipe Petroski Such¹ Dave Cummings¹ Matthias Plappert¹
Fotios Chantzis¹ Elizabeth Barnes¹ Ariel Herbert-Voss¹ William Hebgren Guss¹ Alex Nichol¹ Alex Paino¹
Nikolas Tezak¹ Jie Tang¹ Igor Babuschkin¹ Suchir Balaji¹ Shantanu Jain¹ William Saunders¹
Christopher Hesse¹ Andrew N. Carr¹ Jan Leike¹ Josh Achiam¹ Vedant Misra¹ Evan Morikawa¹
Alec Radford¹ Matthew Knight¹ Miles Brundage¹ Mira Murati¹ Katie Mayer¹ Peter Welinder¹
Bob McGrew¹ Dario Amodei² Sam McCandlish² Ilya Sutskever¹ Wojciech Zaremba¹

Abstract

1. Introduction

- ▶ Early analysis suggested GPT-3 could generate programs from Python docstrings
- ▶ Despite the fact that GPT-3 was not trained for code generation

3

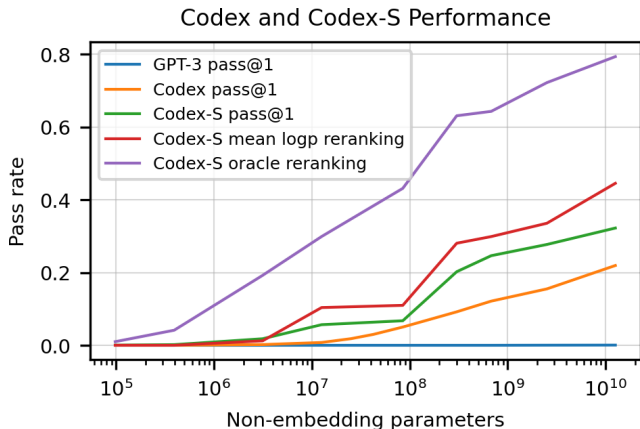
³[Chen et al.(2021)Chen, Tworek, Jun, Yuan, Pinto, Kaplan, Edwards, Burda, Joseph, Brockman et al.]

- ▶ Synthesis challenge: given a Python docstring, generate the function implementation

```
def incr_list(l: list):  
    """Return list with elements incremented by 1.  
    >>> incr_list([1, 2, 3])  
    [2, 3, 4]  
    >>> incr_list([5, 3, 5, 2, 3, 3, 9, 0, 123])  
    [6, 4, 6, 3, 4, 4, 10, 1, 124]  
    """  
    return [i + 1 for i in l]
```

```
def solution(lst):  
    """Given a non-empty list of integers, return the sum of all of the odd elements  
    that are in even positions.  
  
    Examples  
    solution([5, 8, 7, 1]) ==>12  
    solution([3, 3, 3, 3, 3]) ==>9  
    solution([30, 13, 24, 321]) ==>0  
    """  
    return sum(lst[i] for i in range(0, len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```

Evaluating language models for code generation



- ▶ GPT-3 fails on all problems
- ▶ Fine-tuning on problems with this format of function synthesis (Codex-S) improves performance
- ▶ Sampling 100 programs, reranking and choosing best improves

RESEARCH

COMPUTER SCIENCE

Competition-level code generation with AlphaCode

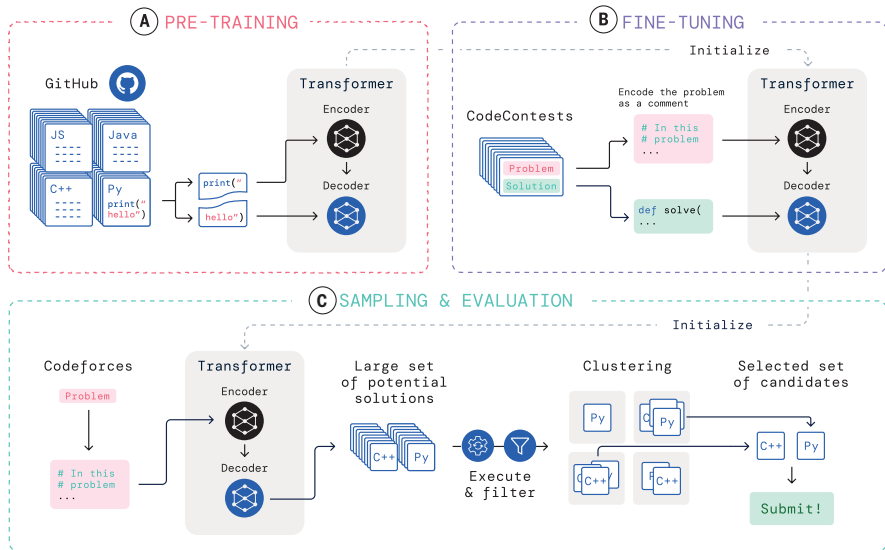
Yujia Li^{*†}, David Choi^{*†}, Junyoung Chung[†], Nate Kushman[†], Julian Schrittwieser[†], Rémi Leblond[†], Tom Eccles[†], James Keeling[†], Felix Gimeno[†], Agustin Dal Lago[†], Thomas Hubert[†], Peter Choy[†], Cyprien de Masson d'Autume[†], Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, Oriol Vinyals^{*}

- ▶ In 2022, DeepMind published AlphaCode, a system combining & expanding these ideas to **solve competitive programming problems**

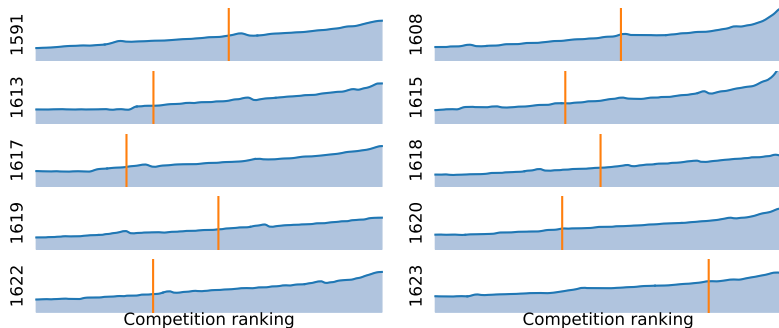
4

⁴[Li et al.(2022)Li, Choi, Chung, Kushman, Schrittwieser, Leblond, Eccles, Keeling, Gimeno, Dal Lago et al.]

AlphaCode: Pipeline

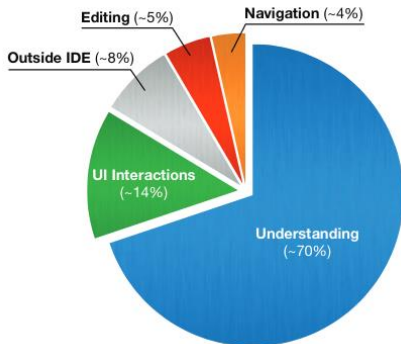


AlphaCode: Ranking among contest participants



(a) AlphaCode's ranking in 10 contests

Are LLMs a job security threat to programmers?



[[Minelli et al, 2015](#)]

- ▶ Answer: not yet!
- ▶ Programming is more than writing lines of code: **problem-solving, strategic planning, and creativity**—abilities that, AI cannot fully replicate yet

LLMs that code: summary

- ▶ (+) Works well on self-contained, short problems
- ▶ (-) Public code repositories have lots of code with bugs
- ▶ (-) Generated code often has functional or security bugs. Still need to understand it!

Thanks

LLMs: Exciting opportunities for applying NLP to real world settings



Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022.

Improving language models by retrieving from trillions of tokens.

In International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA, pages 2206–2240.



Vannevar Bush et al. 1945.

As we may think.

The atlantic monthly, 176(1):101–108.



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. 2021.

Evaluating large language models trained on code.

arXiv preprint arXiv:2107.03374.



Wenhu Chen, William W. Cohen, Michiel De Jong, Nitish Gupta, Alessandro Presta, Pat Verga, and John Wieting. 2022a.

Qa is the new kr: Question-answer pairs as knowledge bases.

arXiv preprint arXiv:2007.03819.



Wenhu Chen, Pat Verga, Michiel De Jong, John Wieting, and William W. Cohen. 2022b.

Augmenting pre-trained language models with qa-memory for open-domain question answerings.

arXiv preprint arXiv:2007.03819.



Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021.

Transformer feed-forward layers are key-value memories.

In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event /

Punta Cana, Dominican Republic, 7-11 November, 2021, pages 5484–5495.



Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. 2022.

Competition-level code generation with alphacode.

Science, 378(6624):1092–1097.