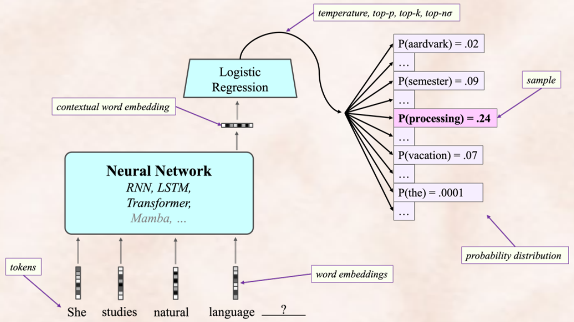
ITCS 4101: Introduction to NLP

LLMs: Use Scenarios, Strengths, and Weaknesses



Razvan C. Bunescu

Department of Computer Science @ CCI

rbunescu@charlotte.edu

Artificial Intelligence

Planning

Reasoning

Robotics

Knowledge Representation

Search

Natural Language Processing

Machine Learning

Large Language Models

GPT 3.5, 4, 4o, o1, o3, 5

Grok 4, Grok 4 Heavy

Gemini 2.5 Flash, 2.5 Pro

Llama 4 Scout, Maverick

Claude Sonnet 4, Opus 4

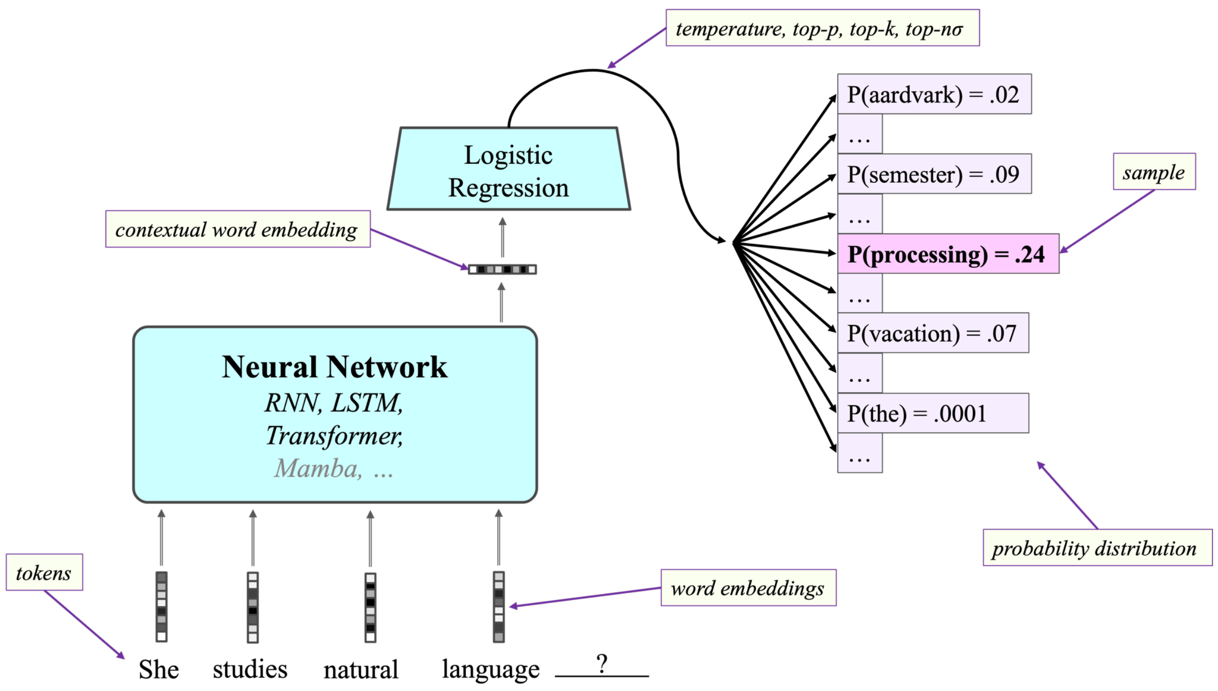
Mixtral

Outline

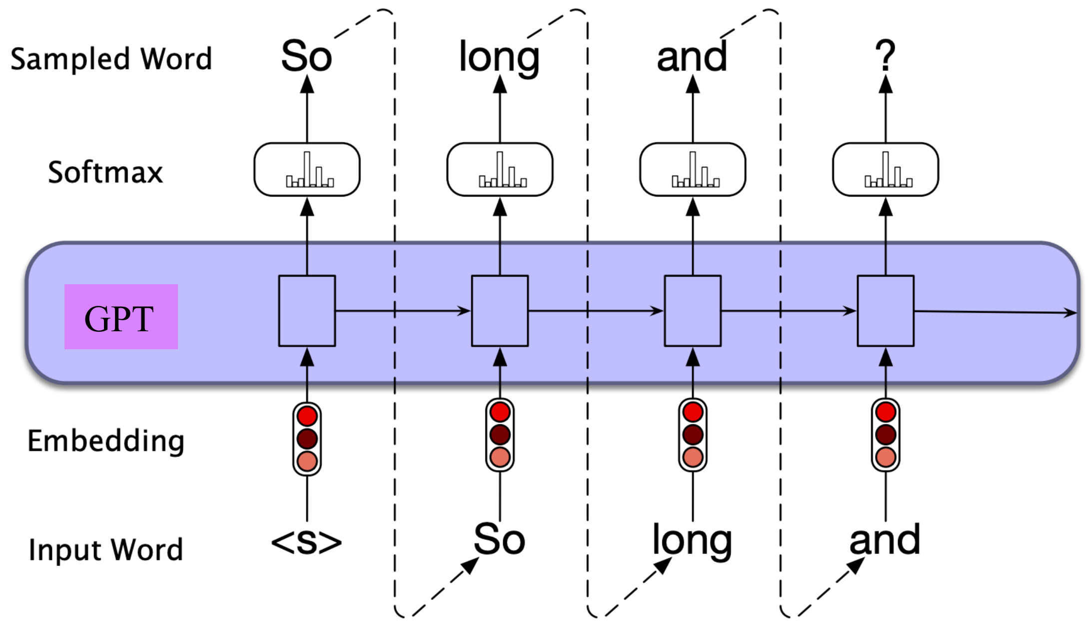
- **Large Language Models (LLM) basics**:
  - Autoregressive Generation and Training Scheme.
- **The AI Inverse Problem**:
  - From predicting words to AI.
- **What we talk about when we talk about AI?**
  - *“AI” cannot count letters* vs. *“AI” gets gold at IMO*.
- **How to use AI effectively**:
  - In general, for coding, for learning.
- **What is Intelligence?**

Using LLMs: Autoregressive Generation

LLM **one-token** distribution computation

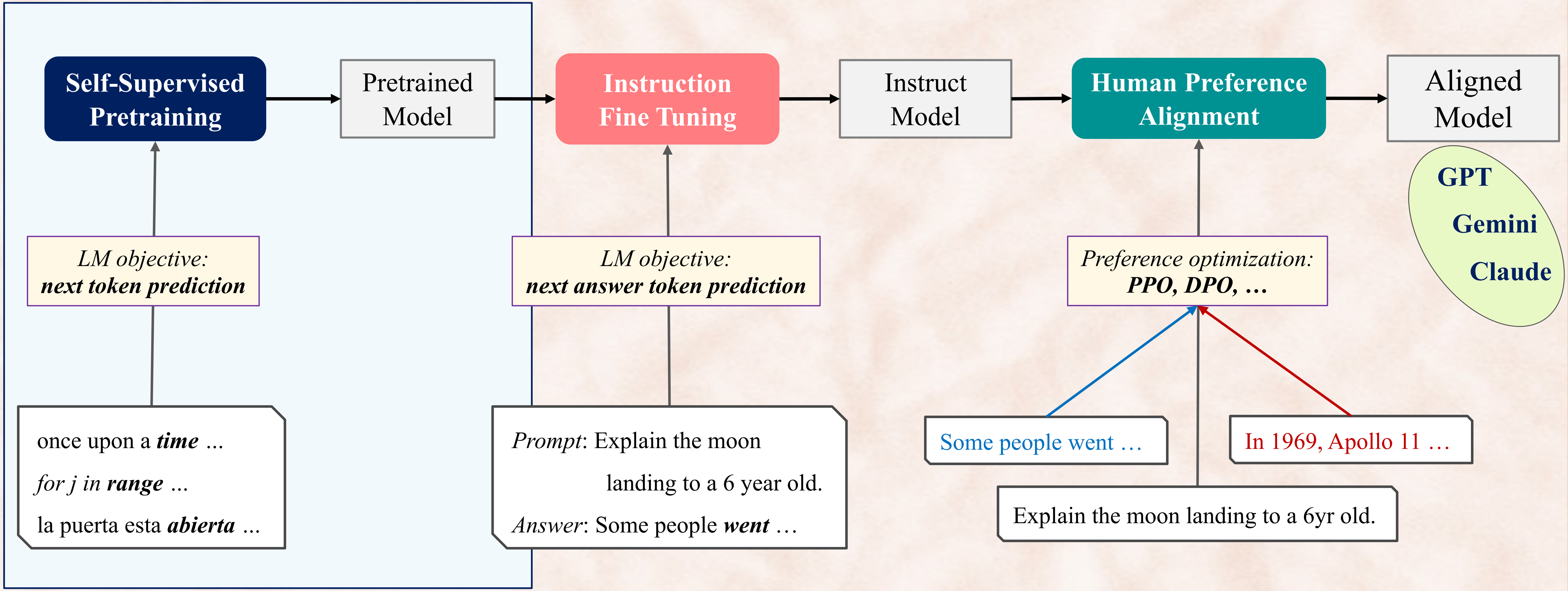


LLM **multi-token** autoregressive generation



https://web.stanford.edu/~jurafsky/slp3/9.pdf

Training LLMs: Pretraining, Instruction tuning, Alignment



Pretraining: Language Modeling (LM) Objective

- **Causal** Language Modeling:
  - **Predict the next word in a sequence**:
    - AI systems use machine _____

      eat?

      learning?

      frogs?

      …
    - **The LM estimates P(word | word$_{-1}$, word$_{-2}$, ...)**
      - we want P(learning | machine, use) >> P(about | machine, eat).
  - **Decoder** neural architectures are widely used to train LMs:
    - GPT, Gemini, Llama, Grok, Mixtral, Claude, …

Pretraining: Language Modeling (LM) Objective

- **Masked** Language Modeling:
  - **Predict the most likely word in a context**:
    - AI systems use machine _____ models for language understanding .

      eat?

      learning?

      frogs?

      …
    - The LM estimates P(word | word_{-1}, word_{-2}, ...; word_1, word_2, ...)
      - we want P(learning | machine, use; models, for) >> P(frogs | machine, use; models, for).
  - **Encoder** neural architectures are used to train masked LMs.
    - BERT, RoBERTa, DistilBERT, …

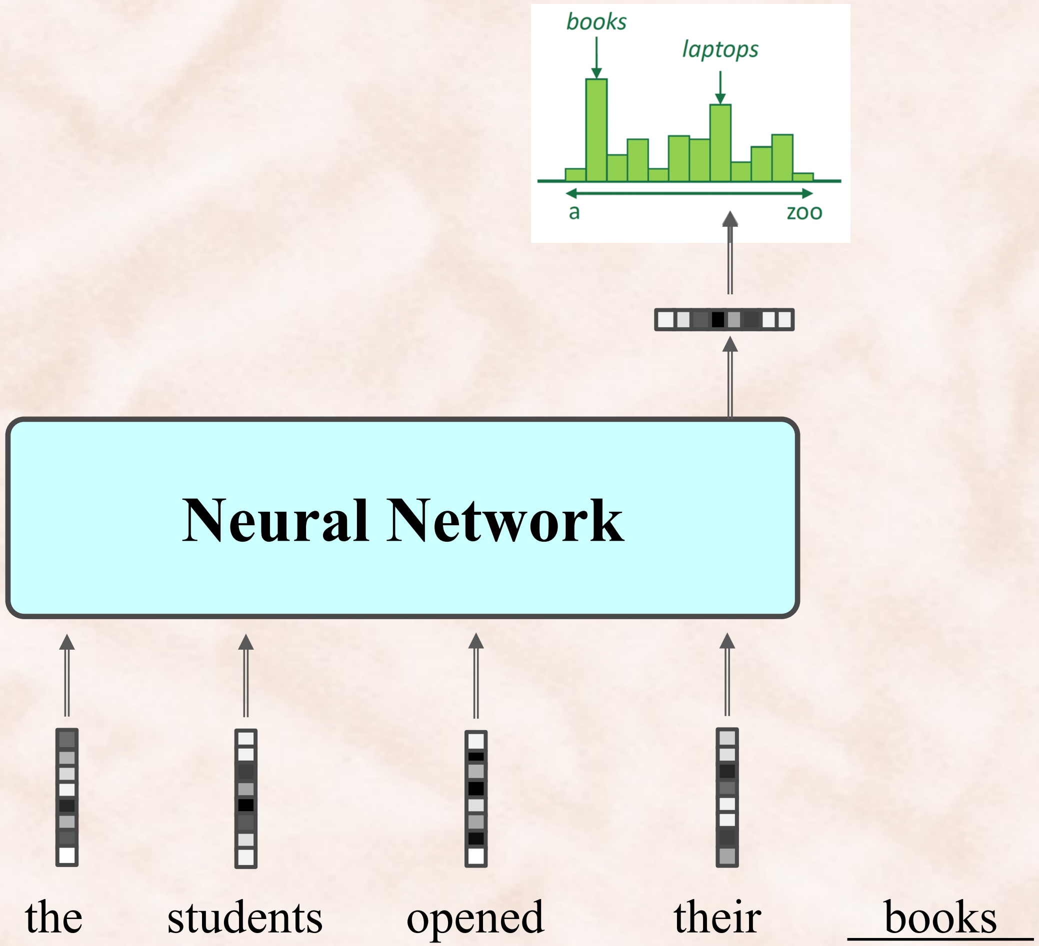
Pretraining of LLMs

- Training a neural network to **predict a missing word** given words in the **context**.
  - This is a special type of *reconstructing the input* idea used in other modalities, such as computer vision (see *autoencoders*).

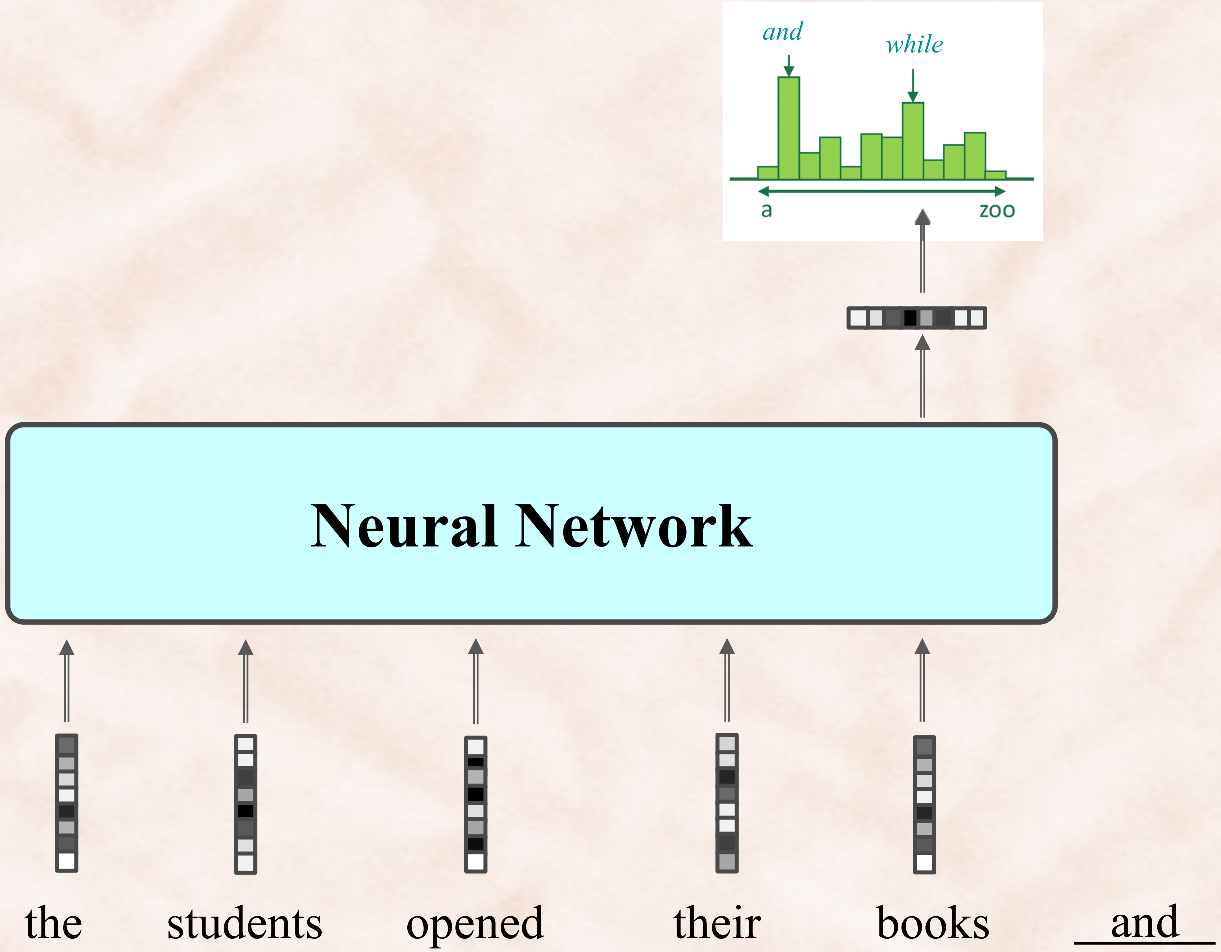
- **Causal LMs** trained using the *distributional hypothesis*:
  - Context: words so far.
  - Missing word: next word.

- **Masked LMs** trained using the *distributional hypothesis*:
  - Context: words to the left and to the right of the center word.
  - Missing word: word in the center.

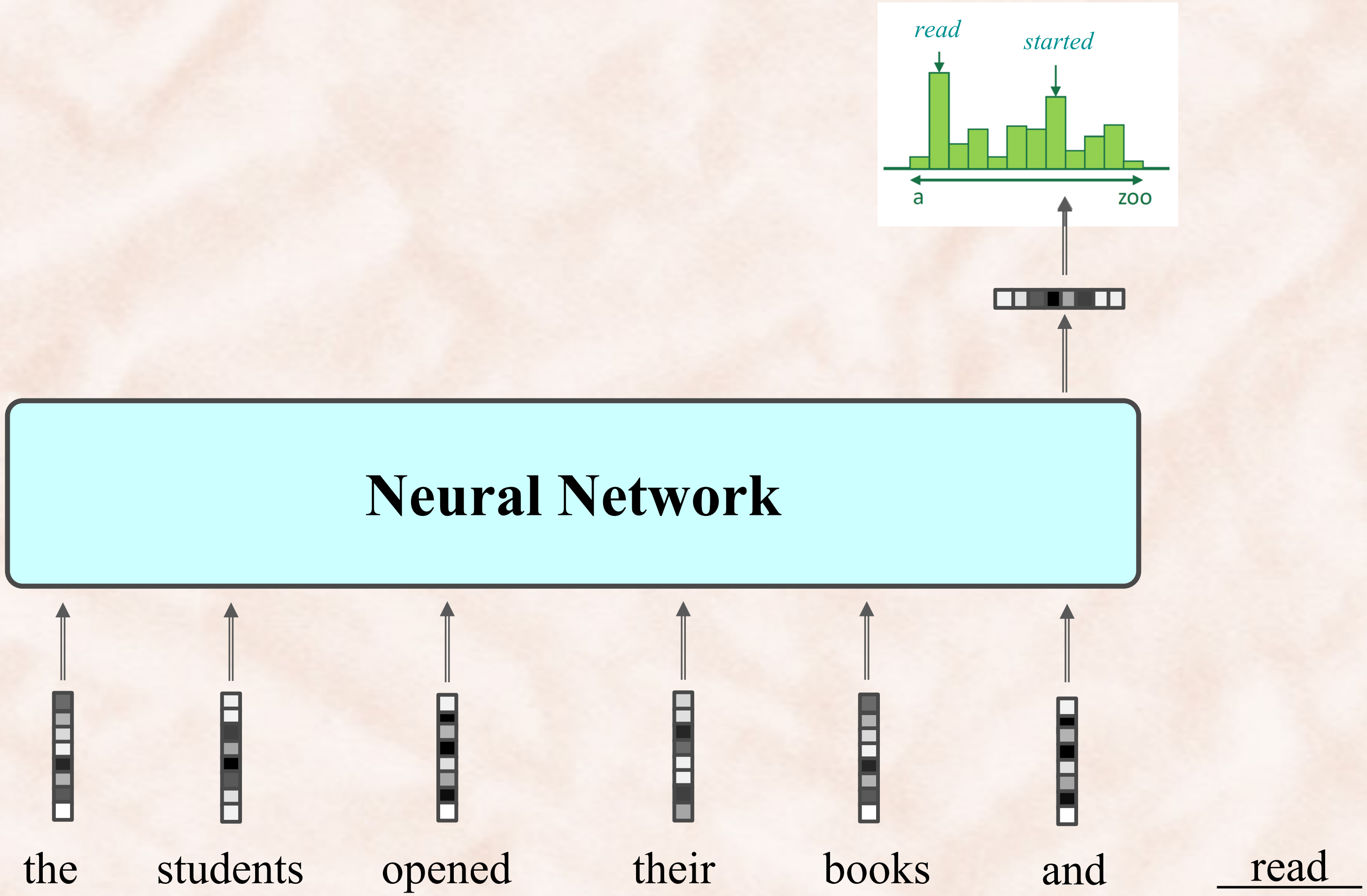
Neural Language Modeling: **Decoders**



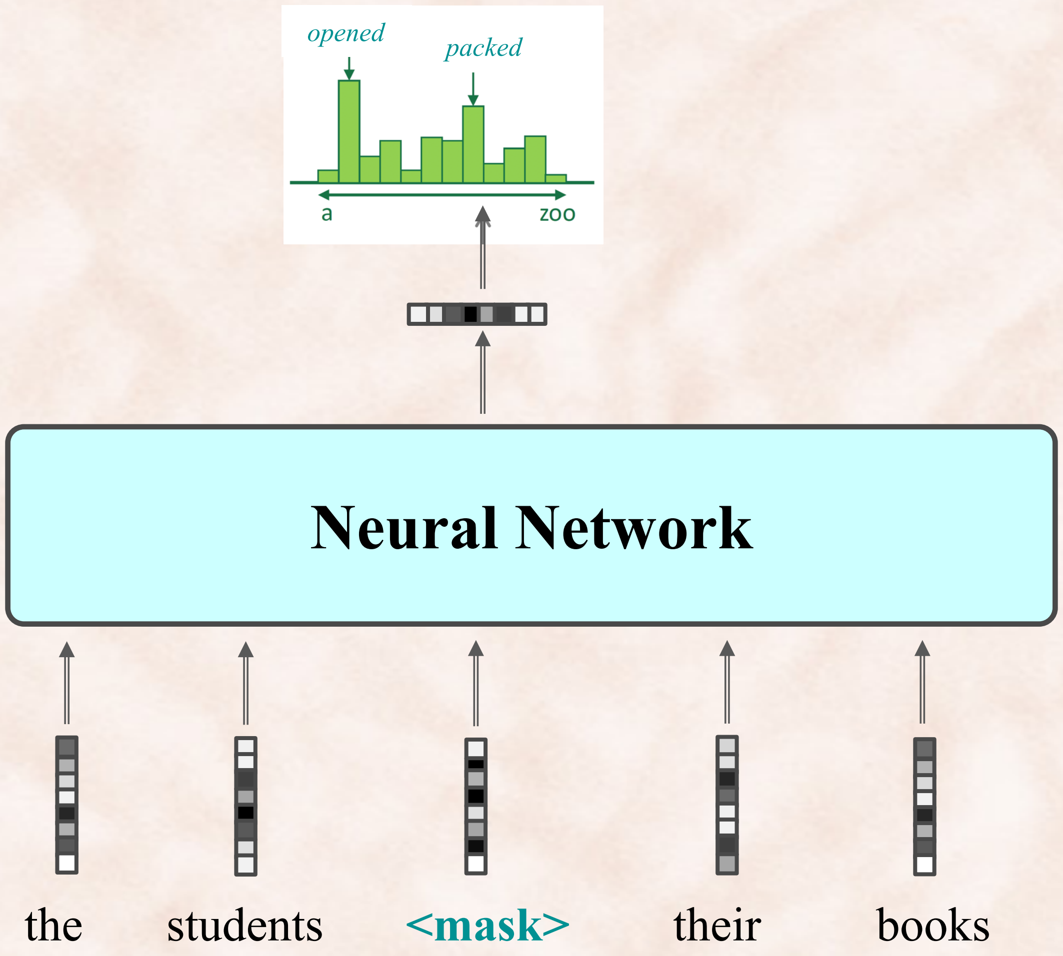
Neural Language Modeling: **Decoders**



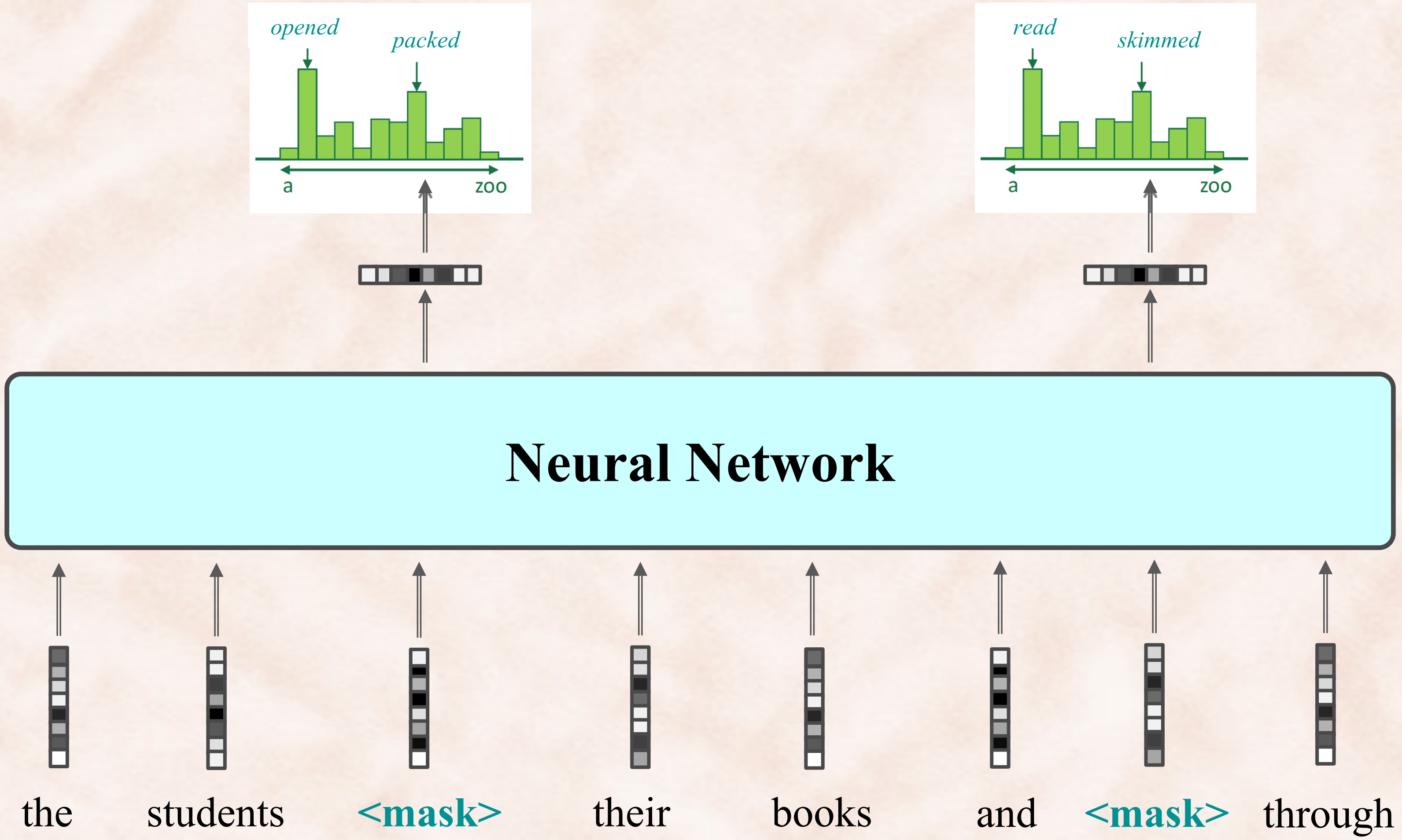
Neural Language Modeling: **Decoders**



Neural Language Modeling: **Encoders**



Neural Language Modeling: **Encoders**



What can be learned from predicting words?

Asheville is a city located in the state of _____

What can be learned from predicting words?

I took ___ dog out for a walk .

What can be learned from predicting words?

Upon exiting the restaurant, the man realized ___ left ___ phone at the table .

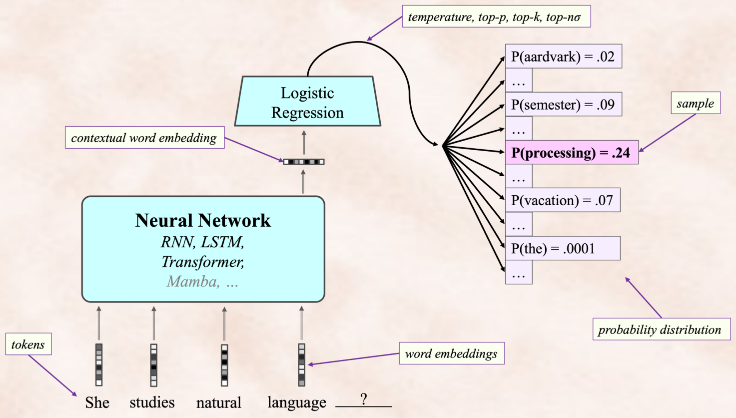
What can be learned from predicting words?

I stopped by the grocery store to buy bread, blueberry pie, milk, and _____ .

What can be learned from predicting words?

Overall , the value I got from the two hours watching it was the sum total of the popcorn and the drink .

The movie was _____



How can we use a pretrained LM to do sentiment analysis?

- no instruction tuning, no preference alignment.
- using just the LM estimates P(*word* | word$_{-1}$, word$_{-2}$, ...)

What can be learned from predicting words?

Andrei was eating in the kitchen .

Roxby joined him for breakfast .

After a while , Andrei went to the living room to watch TV . Once she was done with breakfast , Roxby left the _____

What can be learned from predicting words?

Dan and Tom go to a restaurant for dinner . Dan leaves his coat on the chair , then goes to the bathroom . While Dan was gone , Tom hangs Dan ’s coat on the coat rack . When Dan comes back , he thinks his coat is on the ___

What can be learned from predicting words?

I have been thinking of the sequence that goes 1, 2, 4, 7, 11, 16, 22, ______

What can be learned from predicting missing words?

Theorem: $\sqrt{2}$ is an irrational number .

Proof: _____

Theorem: $\sqrt{2}$ is an irrational number .

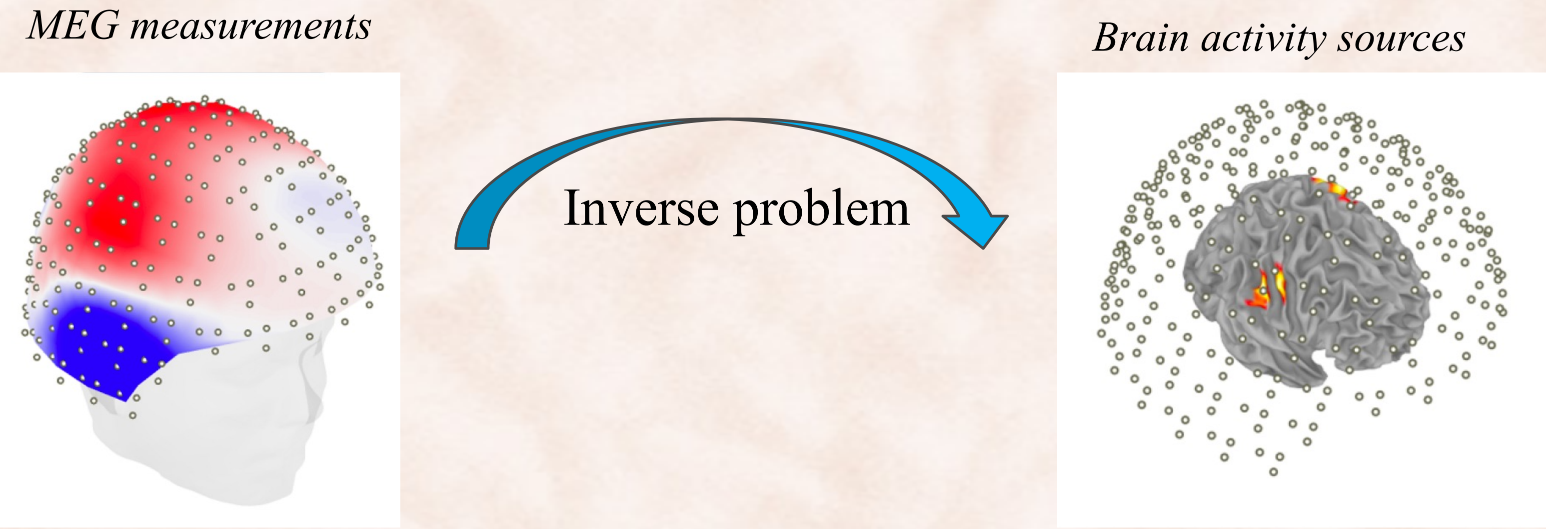
Proof: Suppose that $\sqrt{2}$ were a _____

Theorem: $\sqrt{2}$ is an irrational number .

Proof: Suppose that $\sqrt{2}$ were a rational number , so by definition $\sqrt{2}$ = a _____

Inverse Problems

- Given a set of observations, estimate the causal factors that produced them.
  - Given observations or effects, infer causes.
    - Estimate the **density of the Earth** from measurements of its **gravity field**.
    - Estimate sources of **brain activity** from magnetoencephalography (**MEG**) measurements.



Learning AI from Text is an Inverse Problem

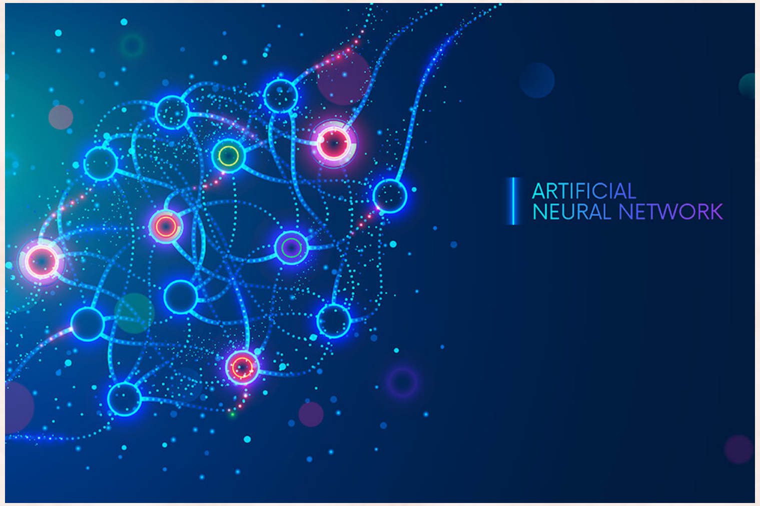
- Given a set of observations, estimate the causal factors that produced them.
  - Given observations or effects, infer causes.
    - Estimate an **intelligent model** from observations of its **behavior**.

Text + code + papers + …

https://commons.wikimedia.org/wiki/Main_Page

Inverse problem

Human Level Intelligence



https://bernardmarr.com/what-is-an-artificial-neural-networks

Learning AI from Text is an Inverse Problem

- Inverse problem are hard:
  - **Ill-posed**, i.e., multiple solutions (causes) that can lead to the same observations (effects).
    - In AI, an infinite number of neural networks can explain the same data.
      - Neural networks are non-identifiable.
      - AI models may achieve intelligence in a different way than humans:
        - » Documented **non-human** behavior.
        - » Documented **human-like** behavior:
          - » Edge detection in first layer of visual cortex.
          - » Neuronal activity patterns for navigation.
    - Use strong regularization to prevent instability and overfitting.
    - Use domain constraints, e.g., Physics Informed NNs.
  - May require very **large sample complexity**.
    - Scaling laws indicate we do not have sufficient text to take LLMs to next level.

What do people mean when they say "AI"?

What do people mean when they say "AI"?

Google's AI Overview, late 2025

What should I do if my fiancé(e) shows up at my wedding?

AI Mode | All | Forums | Short videos | Videos | Images | Shopping | More | Tools

AI Overview

If your fiancé(e) shows up at the wedding, **stay calm and assess the situation to decide if they are there as a threat or an uninvited guest**. If they are a guest, you can decide to allow them to stay or have them escorted out by venue staff, then focus on the ceremony and your new spouse.

If your fiancé(e) is a threat

1. Alert security:
If you feel unsafe or believe they are there to cause trouble, immediately alert your venue's security or the wedding coordinator.

2. Do not engage:
Do not get into an argument or confrontation with your fiancé(e) during the ceremony or reception.

3. Focus on the wedding:
Your priority is your new spouse and enjoying the celebration. Let security or venue staff handle any potential disruptions.

Google Gemini 2.5 Flash, late 2025

Gemini
2.5 Flash
PRO

What should I do if my fiancé(e) shows up at my wedding?

Show thinking

This is a highly unusual and emotionally charged situation. The term "fiancé(e)" implies the person you were engaged to before the wedding, and your question suggests you are now marrying someone else. This is a very specific and likely dramatic scenario.

Here's a breakdown of what you might consider, assuming this is a literal, not metaphorical, question:

1. The Immediate Reaction (During the Ceremony):

- **Pause the Ceremony:** If your ex-fiancé(e) shows up at the altar, the most crucial thing to do is to pause. Don't continue with the vows. This is a massive disruption that needs to be addressed.

Question examples from Vincent Conitzer @ CMU

What do people mean when they say “AI”?

Google’s AI Overview, late 2025

Can an actor practice five distinct roles in a hotel room with a limit of four people?

AI Mode | All | Images | Videos | Short videos | Shopping | Forums | More | Tools

AI Overview

No, an actor cannot practice five distinct roles with a limit of four people in a hotel room, as it would exceed the hotel's legal occupancy limit and potentially violate fire codes, creating a safety hazard. Even if the group size was within limits, the practice of five distinct roles implies there would be five people (the actor and their potential scene partners or coaches), which is also a violation if the limit is four.

Why This is Not Possible

Occupancy Limits:
Hotels set occupancy limits based on room size, safety codes, and fire regulations. A room designed for a maximum of four people cannot legally accommodate five, regardless of the activity.

Safety Violations:
Exceeding the legal occupancy limit is a fire hazard and can lead to violations of safety regulations.

Misinterpretation of "Role":
If the question implies a group of five people (the actor and four others for practice), it directly exceeds a four-person limit. If it refers to five distinct characters played by one actor, the issue remains the presence of five people in the room (one actor and four others involved in practice).

Google Gemini 2.5 Flash, late 2025

Gemini
2.5 Flash
PRO

Can an actor practice five distinct roles in a hotel room with a limit of four people?

This is a logic puzzle. Here's how to solve it:

The user is asking if it's possible for a single actor to practice five distinct roles in a hotel room, given a limit of four people in that room.

The key is that the actor is one person. The number of roles they are practicing is irrelevant to the physical number of people in the room. An actor can practice as many roles as they want by themselves.

So, the four-person limit is a red herring. The actor is one person, which is less than four.

Therefore, yes, an actor can practice five distinct roles in a hotel room with a limit of four people.

Question examples from Vincent Conitzer @ CMU

What do people mean when they say “AI”?

ChatGPT, late 2024

ChatGPT ⌄ Share

John sits next to Elaine in a room. One of them is kind and the other is mean. The person sitting next to Elaine is mean. Who is kind?

John is kind. Since the person sitting next to Elaine is mean, and John is sitting next to Elaine, this implies that Elaine is sitting between John and the mean person. Therefore, John must be the kind one since he is not the one sitting next to Elaine.

Assume John and Elaine are the only persons in the room. Who is the kind person?

If John and Elaine are the only people in the room, and one of them is kind while the other is mean, we can't determine who the kind person is based solely on the information provided. The statement "One of them is kind and the other is mean" doesn't provide enough context to determine who is kind or mean. It's possible that John is kind and Elaine is mean, or vice versa. Without further information, we can't determine which person is kind.

ChatGPT, late 2025

ChatGPT ⌄ Share

John sits next to Elaine in a room. One of them is kind and the other is mean. The person sitting next to Elaine is mean. Who is kind?

Thought for a couple of seconds

Elaine is kind.

Reason: John sits next to Elaine, and the person sitting next to Elaine is stated to be mean — that describes John, so John is mean and therefore Elaine must be the kind one.

ChatGPT ⌄ Share

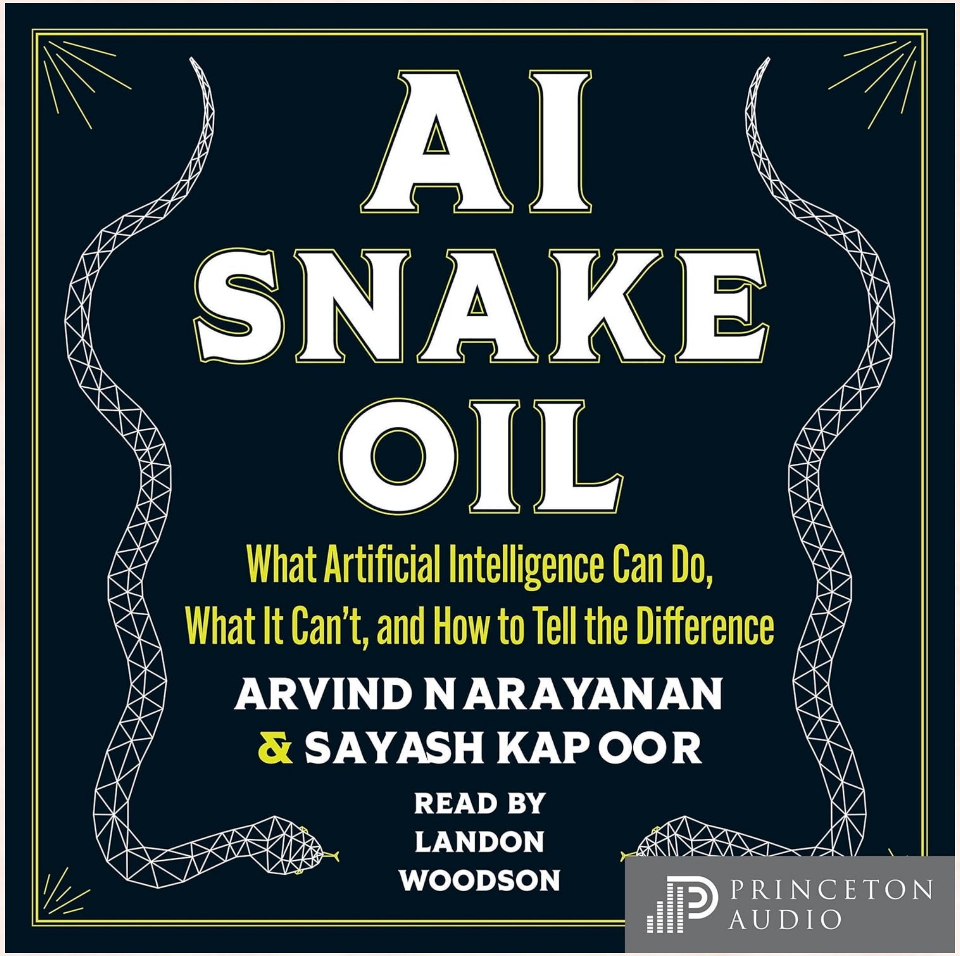
Dan and Tom go to a restaurant for dinner. Dan leaves his coat on the chair, then goes to the bathroom. While Dan is gone, Tom hangs Dan's coat on the coat rack. When Dan returns from the bathroom, where does he think his coat is?

He thinks his coat is still on the chair — because he didn't see Tom hang it on the rack.

What do people mean when they say "AI"?

IMAGINE AN ALTERNATE universe in which people don't have words for different forms of transportation—only the collective noun "vehicle." They use that word to refer to cars, buses, bikes, spacecraft, and all other ways of getting from place A to place B. Conversations in this world are confusing. There are furious debates about whether or not vehicles are environmentally friendly, even though no one realizes that one side of the debate is talking about bikes and the other side is talking about trucks. There is a breakthrough in rocketry, but the media focuses on how vehicles have gotten faster—so people call their car dealer (oops, vehicle dealer) to ask when faster models will be available. Meanwhile, fraudsters have capitalized on the fact that consumers don't know what to believe when it comes to vehicle technology, so scams are rampant in the vehicle sector.

Now replace the word "vehicle" with "artificial intelligence," and we have a pretty good description of the world we live in.



What do people mean when they say "AI"?

- There are many technologies that are called "AI".
- What one "AI" can do well, another "AI" can fail:
  - **Bulls**: *AI won gold at the International Mathematical Olympiad!*
    - It was actually an Advanced version of Gemini with Deep Think.
  - **Bears**: *AI cannot even count the number of b's in "blueberry".*
    - It was actually the initial release of GPT5.
      - It can do it well if it is asked to write the code for it.
- Recommendations:
  - **Specify which "AI" you are referring to**:
    - ChatGPT is still ambiguous, as it can use one of many GPT models:
      - Free version starts with GPT-5, after hitting limit it switches to GPT-5mini, then …
  - Specify what *prompt*, *temperature*, *reasoning level*, …

“AI” Cannot Do Task “X”

- Which “AI”?
- What is the Prompt?
  - The instructions.
  - Any In-Context Learning (ICL) examples?
  - Any Tools?
- What is the Temperature or Reasoning Level?
  - How many samples?
- What else have you tried?
  - AI models can do much better at tasks that can be solved algorithmically if they are asked to write the code for solving that task, instead of generating the output directly.

Case Study: Visualize a Syntactic Parse Tree (1)

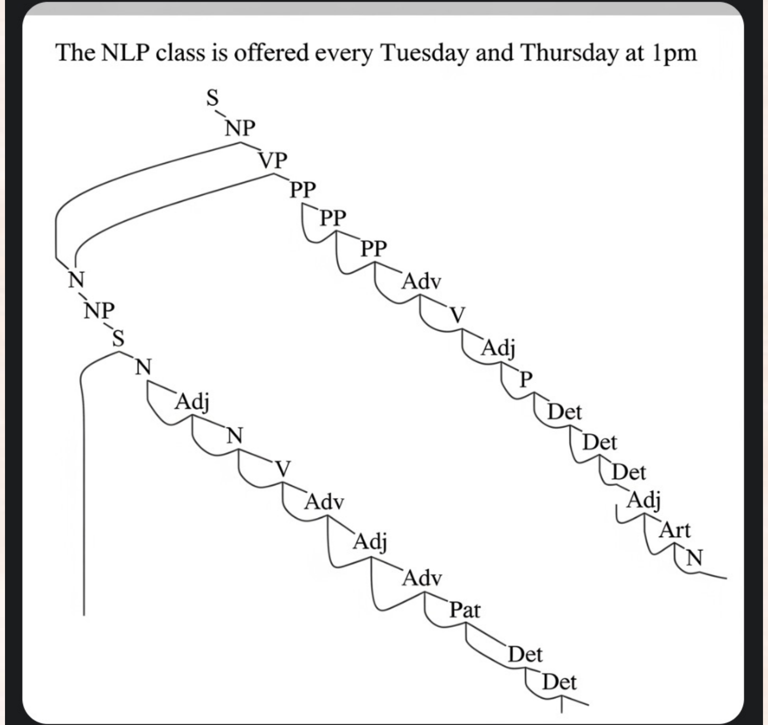
1. Generate the syntactic parse tree for the sentence "*The NLP class is offered every Tuesday and Thursday at 1pm.*", using the Penn Treebank phrase types. Show an image of the parse tree.

Gemini 2.5 Pro

I have generated the syntactic parse tree for the sentence "The NLP class is offered every Tuesday and Thursday at 1pm." below.

```
(S
(NP (DT The) (NNP NLP) (NN class))
(VP
(VBZ is)
(VP
(VBN offered)
(NP (DT every) (NNP Tuesday) (CC and) (NNP Thursday))
(PP (IN at) (NP (CD 1pm))))))
```

Here is an image of the parse tree:



https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/GeminiPro2.5_Syntactic_Parsing.pdf

Case Study: Visualize a Syntactic Parse Tree (2)

2. Can you write Python code that uses Matplotlib to draw the parse tree?

Gemini 2.5 Pro

Of course! You can draw the parse tree using Python with the help of the `matplotlib` and `nltk` libraries. The code below defines a function that recursively plots each node and the lines connecting it to its children.

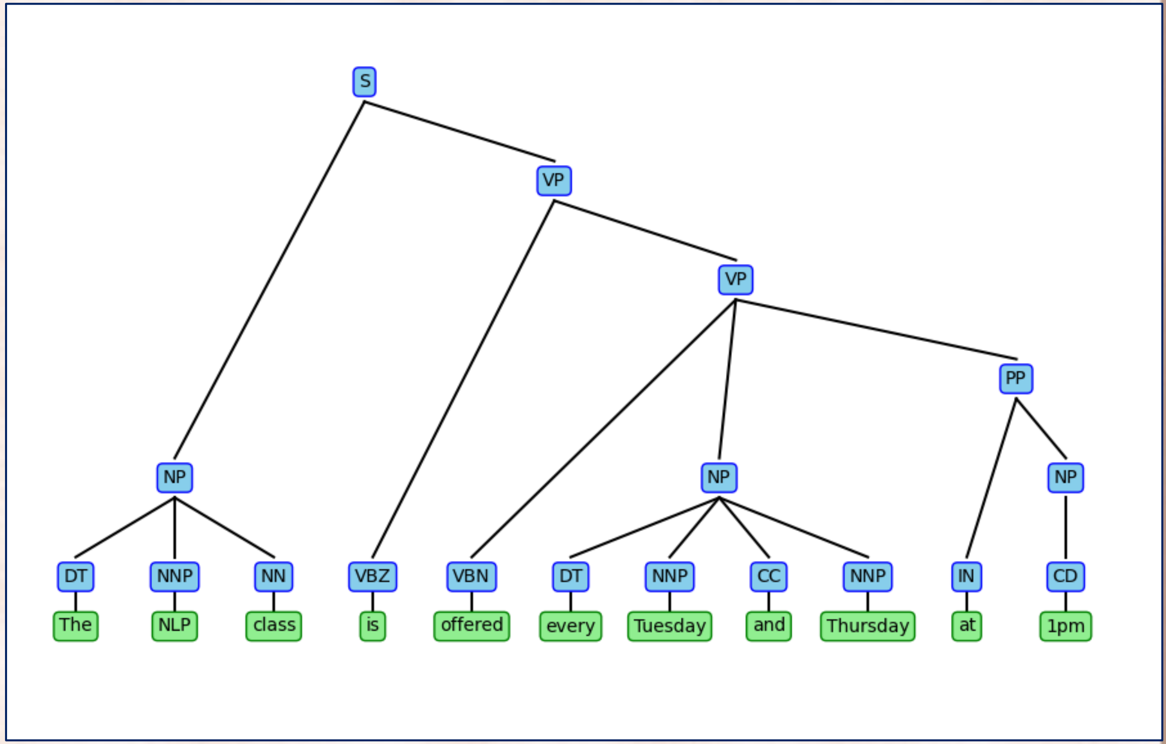
First, ensure you have the necessary libraries installed:

Bash

```
pip install matplotlib nltk
```

Python Code for Plotting

This script first parses the tree from its string representation using `nltk`. Then, it uses `matplotlib` to recursively draw each node, calculating the position of parent nodes based on the positions of their children.



https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/pro_syntactic_tree.py

What “AI” can do

- What AI can do, what it can’t, and how to tell the difference.
  - by Arvind Narayanan and Sayash Kapoor, September 2025

- What AI can do,
  what it can’t do today but will likely do tomorrow,
  what it won’t do tomorrow but will likely do in 10 years,
  and what it will never do.

LFMs for Code Generation

- **Large Foundation Models (LFM)** can generate code for solving problems described in natural language:
  - MS Copilot, Claude Code, Gemini Code Assist, Windsurf, …

- Can humans solve computational problems without learning to code?
  - Yes, but can be **inefficient** and **impractical** for many problems.
  - Currently LFMs offer only a leaky abstraction for programming, as LFMs do not always produce the correct solution code given a correct problem specification => **user needs to tinker with their input, repeatedly**.

- CS skills offer a significant advantage, whether using LFMs or not:
  - **Speed**: much faster at using LFMs to generate correct and efficient solutions => much more productive.
  - **Specification**: much more clear and complete problem descriptions.

> *"People who understand the language of software through their knowledge of coding can tell an LLM or an AI-enabled IDE what they want much more precisely and get much better results."*
>
> The Batch, Issue 292, Andrew Ng, March 12, 2025

LFMs for Code Generation: Ideal Scenario

[OpenAI Reaches Agreement to Buy Windsurf for $3 Billion (*Bloomberg News, May 5*)]

Problem 1 specification in NL:

Write code that reads an input text file line by line, where each line starts with the name of a category, followed by a colon, followed by one or more instances from that category. For example, one line can be "Fruits: apple pear strawberry". The code should create a dictionary where each category is mapped to a list of all the instances for that category.

Windsurf produces working code from first try, no manual intervention ✅

LFMs for Code Generation: Ideal Scenario

[OpenAI Reaches Agreement to Buy Windsurf for $3 Billion (*Bloomberg News, May 5*)]

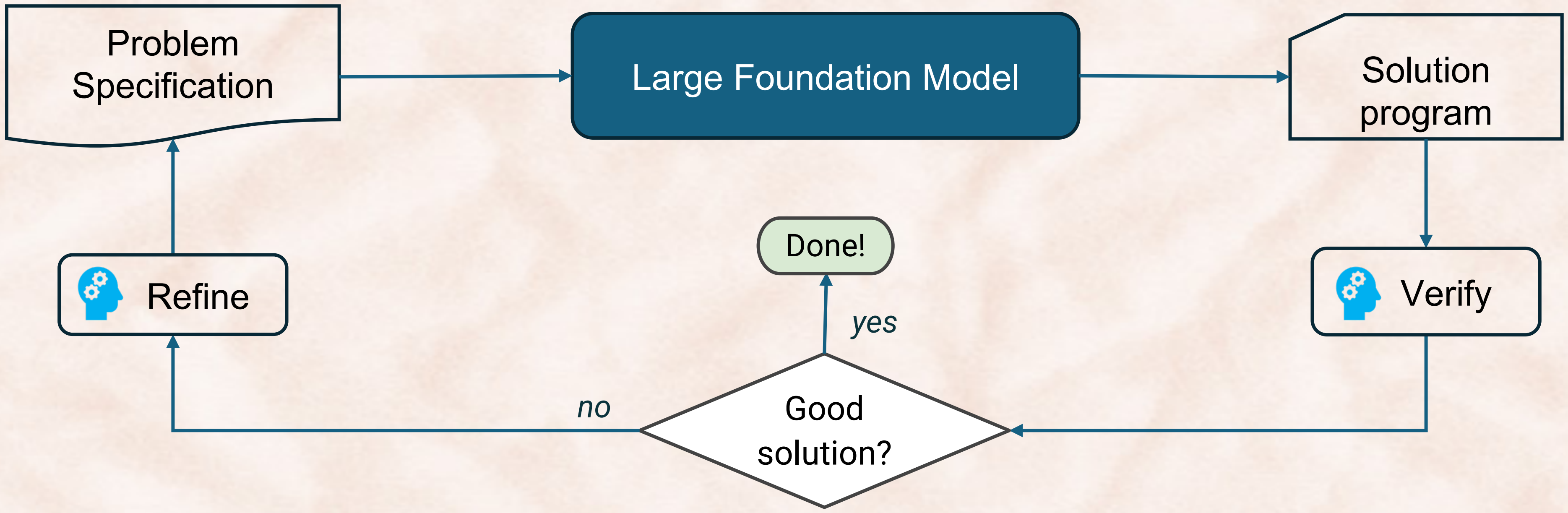
Problem 2 specification in NL:

Generate a 2-dimensional array in NumPy, 4 by 4, that contains all integers between 1 and 16 such that, for any 3 consecutive numbers in any row and in any column, their sum is not divisible by 3.

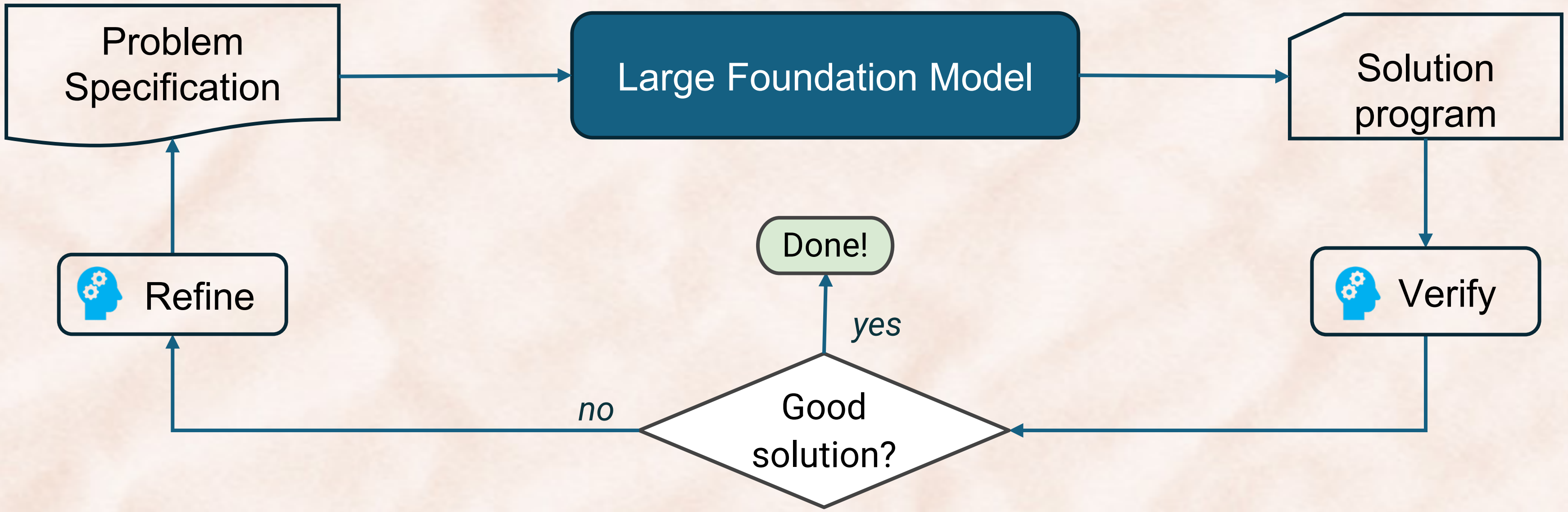
Windsurf produces working code, but the code is very inefficient ❌

- ❑ Code samples random matrices, verifies input constraints, stops when found good matrix.
- ❑ When prompted to make it more efficient, it produces buggy code with runtime errors.

LFMs for Code Generation: Real Scenario



LFMs for Code Generation: SVR Loop



Specify Verify Refine (SVR) loop, employed by many users of AI:

Problem specification:

“Make a 3D game where I can place buildings of various designs and then drive through the town I create”

SVR loop:

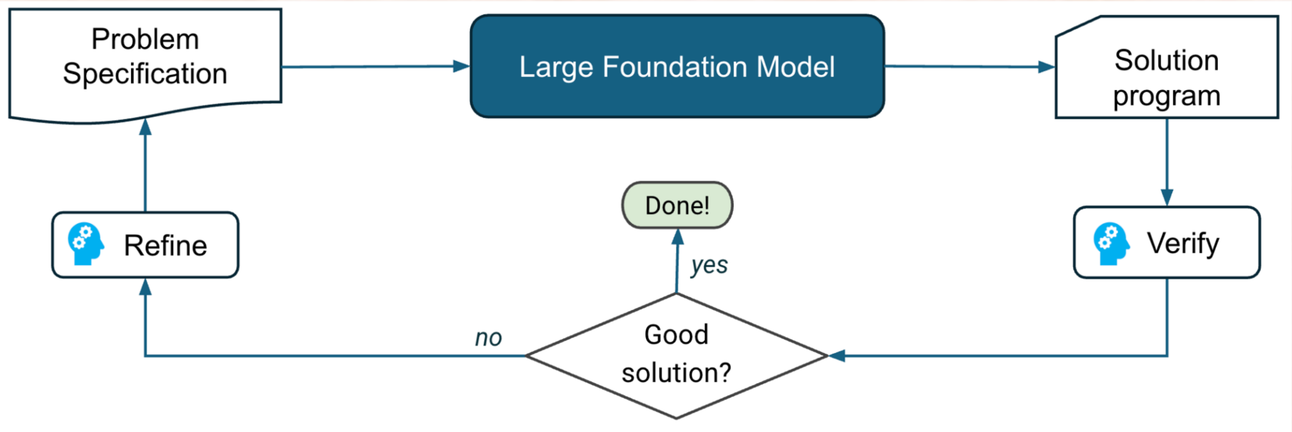
"The results was **a sequence of back-and-forth discussions** with the AI where I would tell it errors and it would work to solve them”

[Speaking things into existence, Mollick, March 11, 2025]

LFMs for Code Generation: Required Skills

Need skills for:

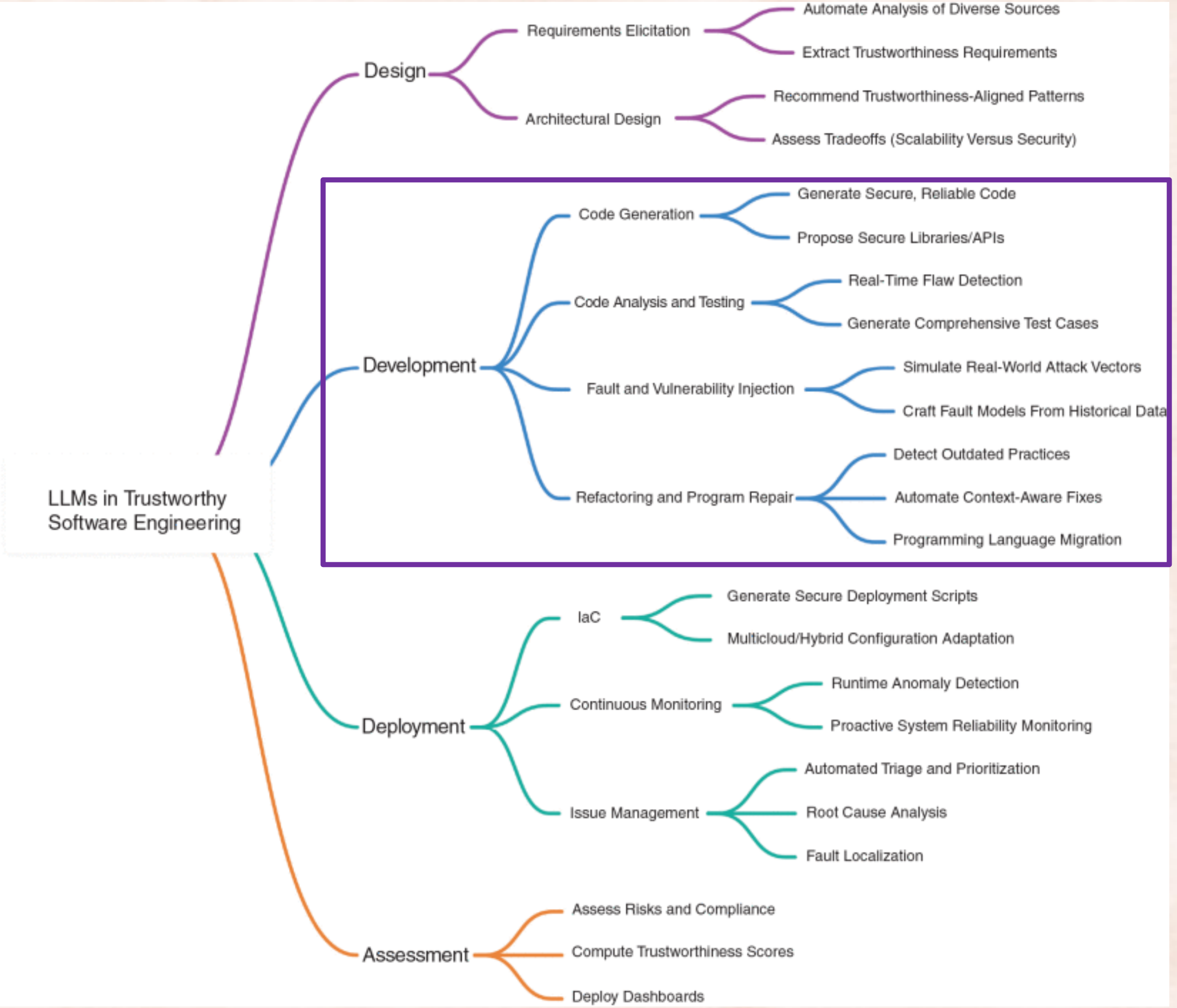
- ❑ **Specification** of problem.
- ❑ **Verification** of solution.
- ❑ **Refinement** of specification.
  - ❑ provide **Focus** and **Guidance**

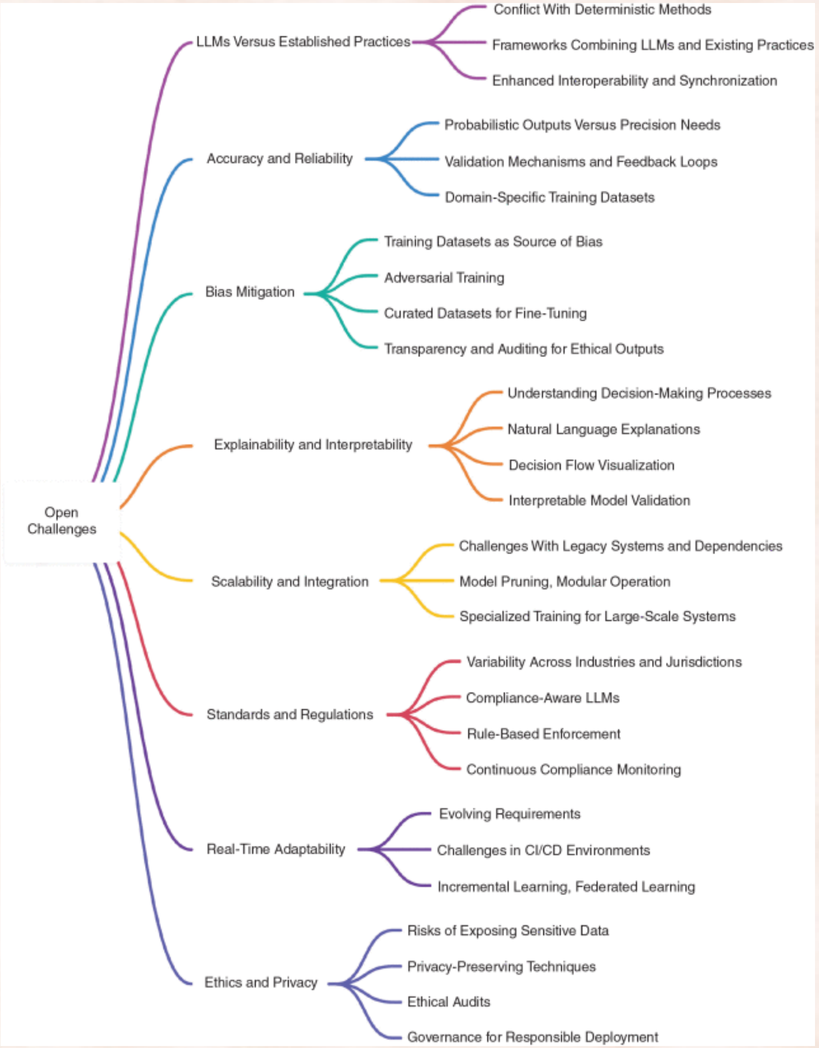


Specification, verification, and **refinement** skills:

- ❑ **English**: ability to formulate clear and complete description in NL.
  - ❑ But often hard to achieve sufficient clarity in NL, can benefit from using more formal descriptions.
    - ❑ Math and code are formal languages.
- ❑ **CS knowledge**: ability to evaluate correctness and efficiency.
- ❑ **Domain knowledge**: use domain specific terminology.
  - ❑ important for judging correctness of solution.

LLMs for Trustworthy Software Engineering





Leveraging LLMs for Trustworthy Software Engineering: Insights and Challenges, Vieira, IEEE Computer, 2025.

Case Study 1: Bubble Sort

Initial Specification (1)

1. Consider the Bubble Sort algorithm. Write code in Python, using modules such as Matplotlib, to generate an animation showing how Bubble Sort works on the input array [25, 13, 11, 24, 12].

Gemini 2.5 Pro

Of course. I can create a Python script to visualize the Bubble Sort algorithm using Matplotlib. This script will generate an animation and save it as a GIF file.

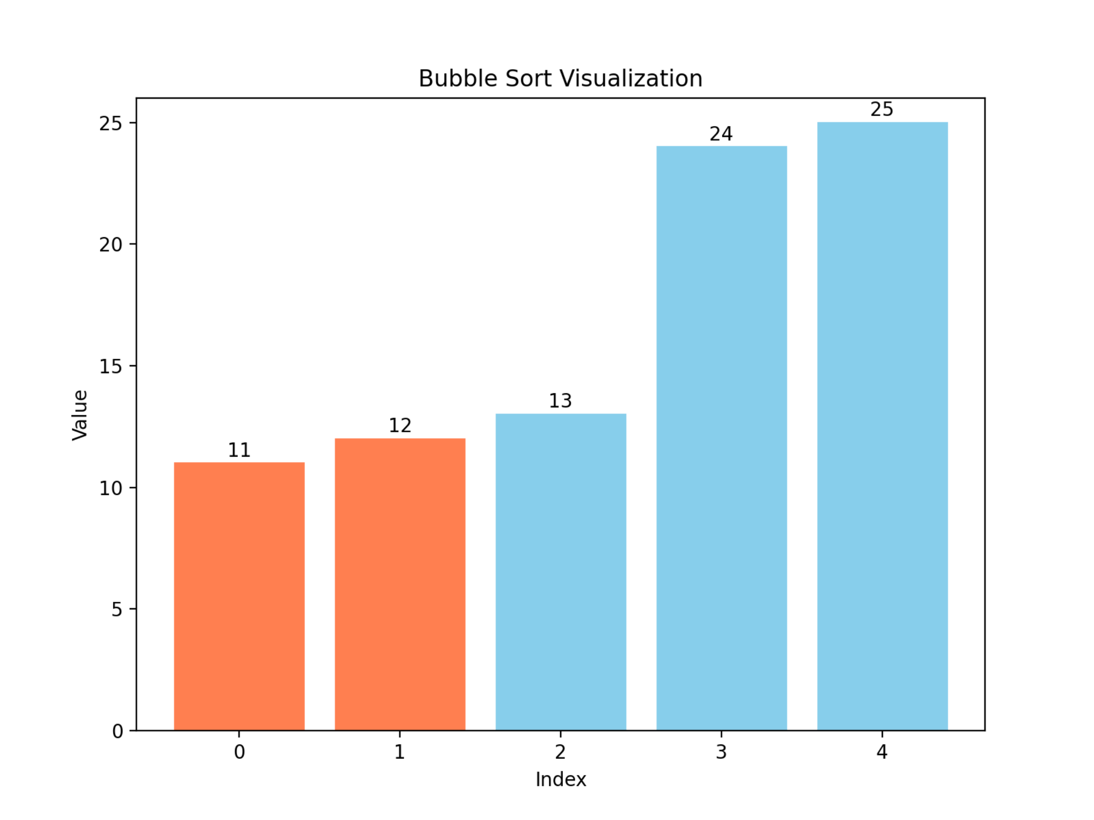
Here is the Python code that will create the animation for you.

Bubble Sort Animation Open

I've created the Python script to visualize the Bubble Sort algorithm. To run it, you'll need to have `matplotlib` and `pillow` installed.

You can install them using pip: `pip install matplotlib pillow`

After running the script, it will generate a file named `bubble_sort.gif` in the same directory, which will show the step-by-step sorting process for the array `[25, 13, 11, 24, 12]`. Let me know if you have any questions!



https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/bubble/bubble_sort1.py

Use Domain Knowledge to Explain Error (2)

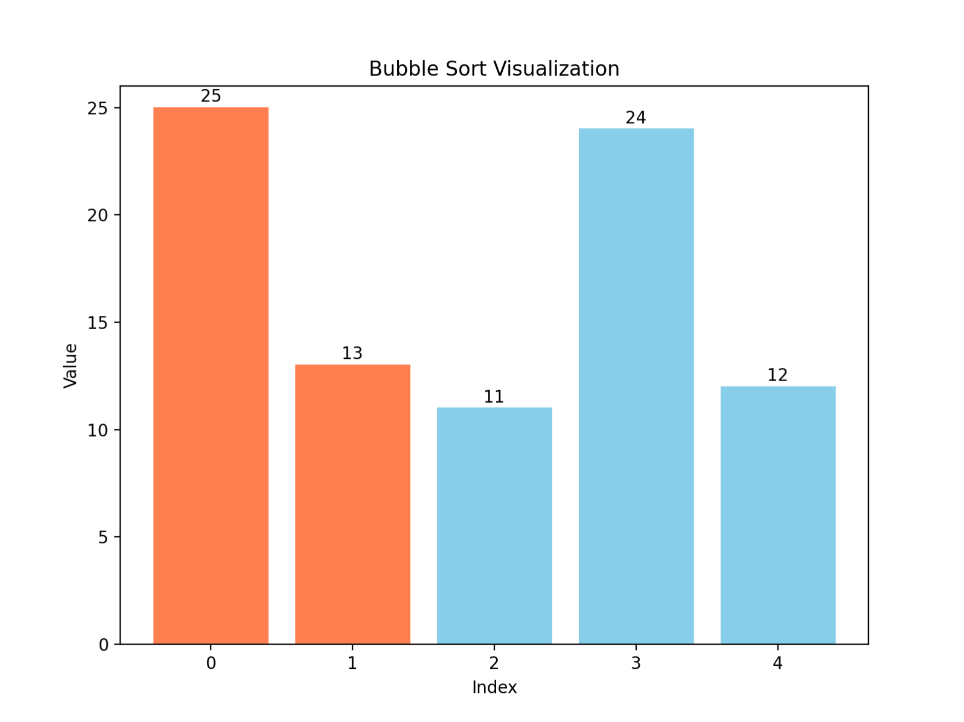
2. Your code has a bug: the animation shows the array already sorted at each step. Can you fix it?

Gemini 2.5 Pro

You've spotted a great catch! You are correct, the animation was showing the final sorted array in every frame. This is a subtle issue related to how Python handles lists. When the generator yielded the array, it was yielding a reference to the *same* list object each time.

I've fixed this by having the generator yield a *copy* of the list at each step. Here are the changes for `bubble_sort_animation.py`.

I've updated the `bubble_sort_animation` function to create and yield a new copy of the list for each frame of the animation. This will ensure that the visualization correctly shows the step-by-step process of the sort. If you run the script now, the `bubble_sort.gif` file should be generated correctly.



https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/bubble/conversation.pdf

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/bubble/bubble_sort2.py

Case study 2: Constraint Satisfaction

Underspecified Requirements (1)

1. Write code in Python that generates a 2-dimensional array in NumPy, 4 by 4, that contains all integers between 1 and 16 such that, for any 3 consecutive numbers in any row and in any column, their sum is not divisible by 3.

Gemini 2.5 Pro

Of course. Here is Python code that generates the specified 4x4 NumPy array.

The solution uses a randomized approach. It repeatedly shuffles the integers from 1 to 16 into a 4x4 grid and checks if the grid meets the condition. It continues this process until a valid grid is found.

Python

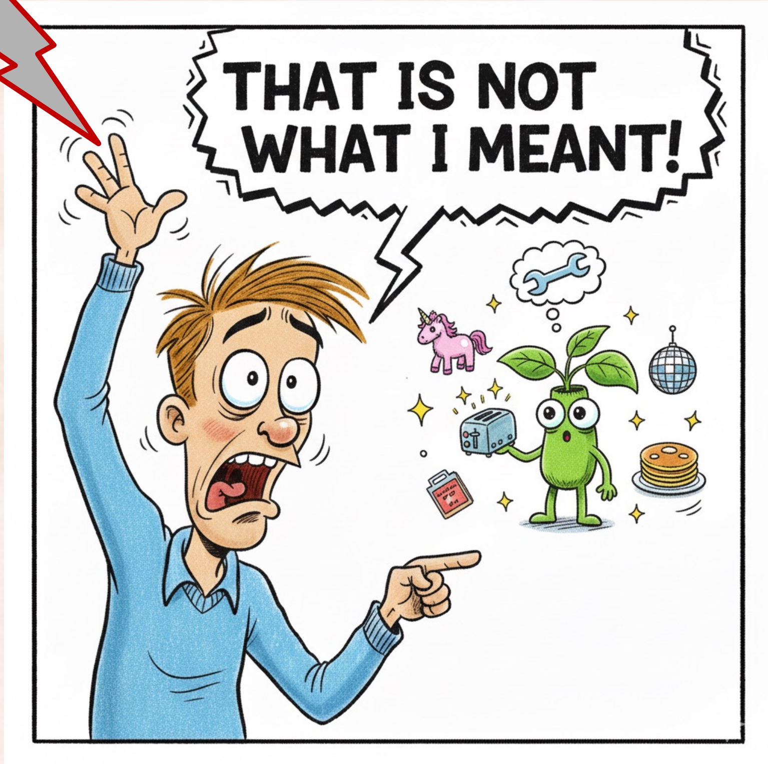
```python
import numpy as np

def is_grid_valid(grid):
    """
    Checks if a 4x4 grid is valid.

    A grid is valid if the sum of any three consecutive integers in any
    row or column is not divisible by 3.
    """
```

```
Solution found:
[[ 3  6  1  9]
 [12 15  4  7]
 [ 2 10  5 13]
 [ 8 16 11 14]]
```

Gemini 2.5 Flash



https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/Gemini2.5Pro_Modulo3_take_1.pdf

Use Domain Knowledge for Focusing (2) and Refinement (3)

2. What is the worst-case time complexity of this solution?

Gemini 2.5 Pro

The worst-case time complexity of this solution is **unbounded**, or effectively **infinite**.

Explanation

Here's a breakdown of why this is the case:

1. **Randomized Algorithm**: The core of the solution is a `while True:` loop that depends on `np.random.shuffle()`. This makes it a **randomized algorithm**. Its runtime is not determined by the size of the input but by the outcome of random chance.
2. **Constant Time Per Iteration**: Each individual step *inside* the loop takes a constant amount of time.
   - **Shuffling**: Shuffling a 16-element array takes a fixed number of operations ($O(n)$ where $n = 16$, which is constant).
   - **Reshaping**: This is also a constant-time operation for a fixed size.
   - **Validation (`is_grid_valid`)**: The function always performs a fixed number of checks (8 for rows, 8 for columns) on a 4x4 grid. This is also constant time.
3. **The Unbounded Loop**: The crucial point is that the loop's termination is not guaranteed. The algorithm terminates only if it randomly stumbles upon one of the valid grid configurations. In the **worst-case scenario**, the random number generator could theoretically produce an endless sequence of invalid permutations, causing the loop to run forever.

3. Can you develop a more efficient solution?

Gemini 2.5 Pro

Yes, a more efficient and deterministic solution can be developed using a **two-stage backtracking algorithm**.

This approach is vastly superior because it first solves a simpler, abstract version of the puzzle and then uses that solution as a template. It avoids the "needle in a haystack" problem of the random search and is guaranteed to find a solution quickly.

The Strategy: Abstract and Conquer

The core idea is to stop thinking about the specific numbers (1-16) and instead focus on their **mathematical properties**. The rule only cares about sums divisible by 3, which relates to the numbers' remainders when divided by 3 (their value modulo 3).

1. **Categorize Numbers**: We group the numbers 1-16 into three "classes":
   - **Class 0** (`num % 3 == 0`): {3, 6, 9, 12, 15} - **(5 numbers)**
   - **Class 1** (`num % 3 == 1`): {1, 4, 7, 10, 13, 16} - **(6 numbers)**
   - **Class 2** (`num % 3 == 2`): {2, 5, 8, 11, 14} - **(5 numbers)**
2. **Solve the Abstract Puzzle (Stage 1)**: First, we find a valid 4x4 grid arrangement of these *classes* (five 0s, six 1s, five 2s). This is a much smaller and more manageable search space. We use a **backtracking** algorithm for this deterministic search.
3. **Populate the Grid (Stage 2)**: Once we have a valid "class template," we simply fill it in with the actual numbers from the corresponding lists. Any number from a class can go in a spot designated for that class, and the mathematical rule will still hold.

Case study 3: Constraint Satisfaction

Complete Specification (1)

1. Write code in Python that … *The code should be as efficient as possible, in particular more efficient than brute force search, and it should have finite worst case time complexity.*

Gemini 2.5 Pro

Of course. Here is a Python function that efficiently generates a 4x4 NumPy array satisfying the given conditions.

This problem is a non-trivial constraint satisfaction puzzle. A brute-force search checking all permutations of numbers 1 through 16 (which is $16 \approx 2 \times 10^{13}$) is computationally infeasible.

A much more efficient approach is to first determine the pattern of remainders modulo 3 for each cell. This reduces the search space significantly. The code below uses a backtracking algorithm to find a valid arrangement of these remainders. Once a valid remainder pattern is found, the function populates the array with the corresponding numbers from 1 to 16. This method is guaranteed to find a solution in finite time and is vastly more efficient than a brute-force permutation search.

Here is the code:

Explanation of the Method

1. **Problem Reduction**: The core constraint depends on the sum of numbers modulo 3. Therefore, the problem is simplified by first considering the remainders of the numbers ($0, 1, \text{or } 2$) rather than the numbers themselves.
2. **Remainder Distribution**: The integers from 1 to 16 are distributed into three groups based on their remainder when divided by 3:
   - **Remainder 0**: 5 numbers ({3, 6, 9, 12, 15})
   - **Remainder 1**: 6 numbers ({1, 4, 7, 10, 13, 16})
   - **Remainder 2**: 5 numbers ({2, 5, 8, 11, 14})
3. **Constraint on Remainders**: The sum of three numbers is divisible by 3 if and only if the sum of their remainders is divisible by 3. The forbidden combinations of three consecutive remainders (r_1, r_2, r_3) are:
   - All remainders are the same: $(0, 0, 0)$, $(1, 1, 1)$, or $(2, 2, 2)$.
   - All remainders are different: Any permutation of $(0, 1, 2)$.
4. **Backtracking Search**: A backtracking algorithm (`_find_remainder_matrix`) is used to find a valid 4x4 grid of remainders that respects the constraints and the required counts for each remainder type. It works by placing remainders one cell at a time and backtracking if a placement violates a condition.
5. **Efficient Generation**: To make the `generate_constrained_array` function highly efficient (constant time, $O(1)$), a pre-computed, valid remainder matrix found by the solver is stored directly in the function. This avoids the need to run the search every time.
6. **Population**: The final array is constructed by taking the valid remainder matrix as a template. It iterates through this template, and for each cell, it randomly picks an unused number from the corresponding remainder group to place in the final array.

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/Gemini2.5Pro_Modulo3_take_2.pdf

Case study 4: Sequence Completion

Initial Solution (1)

1. What are the next two numbers in the sequence 1 1 3 2 9 4 27 8 16 32 81 64 243 128?

Claude Sonnet 4

Looking at this sequence, I need to identify the pattern by separating it into different subsequences.

Let me group the numbers by position:

- Positions 1, 3, 5, 7, 9, 11, 13... (odd positions): 1, 3, 9, 27, 81, 243...
- Positions 2, 4, 6, 8, 10, 12, 14... (even positions): 1, 2, 4, 8, 16, 32, 64...

Now I can see the patterns:

- Odd positions: Powers of 3 → $3^0, 3^1, 3^2, 3^3, 3^4, 3^5$... = 1, 3, 9, 27, 81, 243...
- Even positions: Powers of 2 → $2^0, 2^1, 2^2, 2^3, 2^4, 2^5$... = 1, 2, 4, 8, 16, 32, 64...

The sequence alternates between these two patterns.

The next number (position 15, odd) would be $3^6 = 729$
The number after that (position 16, even) would be $2^7 = 256$

Therefore, the next two numbers are **729** and **256**.

ChatGPT with GPT-5

Let's carefully analyze the sequence:

Given sequence:
1, 1, 3, 2, 9, 4, 27, 8, 16, 32, 81, 64, 243, 128, ...

Step 1: Separate into two interleaved subsequences

- Odd positions: 1, 3, 9, 27, 81, 243, ...
- Even positions: 1, 2, 4, 8, 16, 32, 64, 128, ...

Step 2: Identify patterns

- **Odd positions:** powers of 3

$$3^0 = 1,\ 3^1 = 3,\ 3^2 = 9,\ 3^3 = 27,\ 3^4 = 81,\ 3^5 = 243,\ \ldots$$

- **Even positions:** powers of 2

$$2^0 = 1,\ 2^1 = 2,\ 2^2 = 4,\ 2^3 = 8,\ 2^4 = 16,\ 2^5 = 32,\ 2^6 = 64,\ 2^7 = 128,$$

Step 3: Predict next terms

- Next odd term after 243 = $3^6 = 729$.
- Next even term after 128 = $2^8 = 256$.

Gemini 2.5 Pro

This sequence is actually two separate sequences interleaved together.

The Two Patterns

1. **Numbers in the odd positions** (the first, third, fifth number, and so on) are powers of 3.
2. **Numbers in the even positions** (the second, fourth, sixth number, and so on) are powers of 2.

Let's break it down:

- **Odd Positions (Powers of 3):** 1 (3^0), 3 (3^1), 9 (3^2), 27 (3^3), 81 (3^4), 243 (3^5),...
- **Even Positions (Powers of 2):** 1 (2^0), 2 (2^1), 4 (2^2), 8 (2^3), 16 (2^4), 32 (2^5), 64 (2^6), 128 (2^7),...

Although there appears to be a slight mix-up in the middle of the sequence you provided, the underlying pattern of alternating powers of 3 and powers of 2 is clear.

The Next Two Numbers

To find the next two numbers, we just need to find the next power of 3 and the next power of 2.

- The next number in the sequence of powers of 3, after 243 (3^5), is 3^6, which is **729**.
- The next number in the sequence of powers of 2, after 128 (2^7), is 2^8, which is **256**.

Therefore, the next two numbers in the sequence are **729** and **256**.

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/Gemini2.5Pro_Sequence_Prime.pdf

Correction (2)

2. There is no mix-up in the middle of the sequence, all the numbers in the input sequence are correct. Can you try again?

Gemini 2.5 Pro

You are absolutely right to point out that the sequence is correct as written. My apologies for assuming there was a mix-up. The sequence is more subtle than a simple interleaving. Let's analyze the pattern again, respecting every number's position.

The key is still to look at it as two separate sequences woven together.

The Two Interleaved Sequences

First, let's separate the numbers into those in **odd-numbered positions** and those in **even-numbered positions**.

- **Odd Positions (Sequence A):** `1, 3, 9, 27, 16, 81, 243, ...`
- **Even Positions (Sequence B):** `1, 2, 4, 8, 32, 64, 128, ...`

Now, let's find the rule for each sequence.

Rule for Sequence A (Odd Positions)

This sequence almost follows a simple "multiply by 3" rule.

- $1 \times 3 = 3$
- $3 \times 3 = 9$
- $9 \times 3 = 27$

Here, the pattern is briefly interrupted by the number **16**. However, notice that the pattern picks right back up afterward:

- $81 \times 3 = 243$

The underlying rule is clearly multiplication by 3. To find the next number, we apply this rule to the last number in the sequence.

- $243 \times 3 = 729$

So, the next number in the odd-position sequence is **729**.

Rule for Sequence B (Even Positions)

This sequence follows a "multiply by 2" rule, but with one jump.

- $1 \times 2 = 2$
- $2 \times 2 = 4$
- $4 \times 2 = 8$

The pattern seems to be broken here, as the next number is 32, not 16 ($8 \times 4 = 32$). However, after this jump, the "multiply by 2" rule resumes perfectly.

- $32 \times 2 = 64$
- $64 \times 2 = 128$

The dominant pattern is multiplication by 2. To find the next number, we continue this established rule.

- $128 \times 2 = 256$

So, the next number in the even-position sequence is **256**.

Guidance (3)

3. That is wrong. I will give you a hint: first, look at the positions of the elements that are powers of 3, and then try to generalize. Can you try again?

Gemini 2.5 Pro

Following your hint, let's look at the positions of the powers of 3.

The Underlying Rule

The rule determines the base of the number (whether it's 2 or 3) based on the properties of its position in the sequence.

- **Powers of 3** appear at positions that are **1 or an odd prime number**.
- **Powers of 2** appear at all other positions (i.e., composite numbers and the prime number 2).

Once the base is determined, the exponents for the powers of 3 and the powers of 2 each increase sequentially.

How the Rule Explains the Sequence

Let's test this rule on the sequence you provided:

- **Position 1:** 1 is special. Rule says Power of 3. It's 3^0. (1st power of 3)
- **Position 2:** 2 is not an odd prime. Rule says Power of 2. It's 2^0. (1st power of 2)
- **Position 3:** 3 is an odd prime. Rule says Power of 3. It's 3^1. (2nd power of 3)
- **Position 4:** 4 is composite. Rule says Power of 2. It's 2^1. (2nd power of 2)
- **Position 5:** 5 is an odd prime. Rule says Power of 3. It's 3^2. (3rd power of 3)
- **Position 6:** 6 is composite. Rule says Power of 2. It's 2^2. (3rd power of 2)
- **Position 7:** 7 is an odd prime. Rule says Power of 3. It's 3^3. (4th power of 3)
- **Position 8:** 8 is composite. Rule says Power of 2. It's 2^3. (4th power of 2)
- **Position 9:** 9 is composite. Rule says Power of 2. It's 2^4. (5th power of 2)
- **Position 10:** 10 is composite. Rule says Power of 2. It's 2^5. (6th power of 2)
- **Position 11:** 11 is an odd prime. Rule says Power of 3. It's 3^4. (5th power of 3)
- **Position 12:** 12 is composite. Rule says Power of 2. It's 2^6. (7th power of 2)
- **Position 13:** 13 is an odd prime. Rule says Power of 3. It's 3^5. (6th power of 3)
- **Position 14:** 14 is composite. Rule says Power of 2. It's 2^7. (8th power of 2)

The rule perfectly describes the entire sequence.

Finding the Next Two Numbers

Now we apply this rule to find the 15th and 16th terms.

The 15th Term

1. **Check the position:** The number 15 is a composite number (3×5), not an odd prime.
2. **Determine the base:** According to the rule, the base must be **2**.
3. **Find the exponent:** We have already used eight powers of 2 (with exponents 0 through 7). This will be the ninth power of 2 in the sequence, so its exponent will be 8.
4. **Calculate the value:** $2^8 = \mathbf{256}$

The 16th Term

1. **Check the position:** The number 16 is a composite number (4×4).
2. **Determine the base:** The base must also be **2**.
3. **Find the exponent:** This will be the tenth power of 2 in the sequence, so its exponent will be 9.
4. **Calculate the value:** $2^9 = \mathbf{512}$

The next two numbers in the sequence are **256** and **512**.

Why Sequence Completion?

Occam's Razor

William of Occam (1288 – 1348)

- English Franciscan friar, theologian and philosopher.

"*Entia non sunt multiplicanda praeter necessitatem*"

  – Entities must not be multiplied beyond necessity.

i.e. Do not make things needlessly complicated.

i.e. Prefer the simplest hypothesis that fits the data.

Occam's Razor and Machine Learning

- ML problem: Find a model **M**

  that is *simple* + that *fits the training data*.

$$\hat{\mathbf{M}} = \underset{\mathbf{M}}{\operatorname{argmin}} \; Complexity(\mathbf{M}) + Error(\mathbf{M}, Data)$$

- **Inductive hypothesis**: Models that perform well on training examples are expected to do well on test (unseen) examples.

- **Occam's Razor**: Simpler models are expected to do better than complex models on test examples (assuming similar training performance).

From Occam's Razor to Kolmogorov Complexity to Intelligence

- **Occam's Razor**: Prefer the **simplest** hypothesis that fits the data.

- **Kolmogorov Complexity** = the length of the shortest program that generates the data.
  - 1, 2, 4, 7, 11, 16, 22, 29, 37, 46, 56, …
  - 1, 3, 6, 11, 18, 29, …
  - 1, 2, 3, 5, 5, 8, 7, 11, 9, …

- **Intelligence** = the ability to apply Occam's Razor.

What is Intelligence? How to measure It?

- Marcus Hutter:
  - "Universal Artificial Intelligence: Sequential Decisions Based on Algorithmic Probability"
  - Book, 2005.
- Shane Legg:
  - "**Machine Super Intelligence**"
  - PhD Thesis, 2008.
    - http://www.vetta.org/documents/Machine_Super_Intelligence.pdf
- A definition of intelligence that can be applied to humans, animals, machine.

Sequence Prediction & Occam's Razor

- **Standard psychometric tests implicitly test the ability to use Occam's razor**.
- Consider the following sequence:

  1, 3, 5, 7, ...

- What do you predict will come next?
  - What process do you think is generating these numbers?
    - M_1: $2n + 1$

      => 9
    - M_2: $2n - 1 + (n - 1)(n - 2)(n - 3)(n - 4)$

      => 33

Sequence Complexity

- Some sequences are more difficult to predict than others.
  - Psychometric tests correlate IQ with the difficulty of the sequences that the agent correctly predicts.
  - How to measure sequence difficulty?

- **Kolmogorov Complexity** (s) = the length of the smallest program that can generate the sequence s.

- But intelligence is not passive! Agents can act => use concepts from *reinforcement learning*.

Universal Intelligence

Informal definition, that can be applied to any agent, e.g., animal, human, machine, …

> “The intelligence of an agent is the expected reward that the agent obtains over the space of all computable environments, where (*very important*) the distribution of these environments is such that environments that have lower Kolmogorov complexity (i.e. *have simpler descriptions in terms of their distribution of observations and rewards as functions of action histories*) are more likely to happen (the "universal distribution"). This effectively rewards agents for correctly using Occam’s razor, by weighting the environments according to their complexity.”

http://www.vetta.org/documents/Machine_Super_Intelligence.pdf

Occam’s Razor and Science

- **Paper:**

  “Razor sharp: The role of Occam's razor in science”, Johnjoe McFadden, Annals of NYAS, 2023
  https://doi.org/10.1111/nyas.15086

- **Abstract:**

  Occam's razor – the principle of simplicity – has recently been attacked as a cultural bias without rational foundation. Increasingly, belief in pseudoscience and mysticism is growing. I argue that inclusion of Occam's razor is an essential factor that distinguishes science from superstition and pseudoscience. I also describe how the razor is embedded in Bayesian inference and argue that *science is primarily the means to discover the simplest descriptions of our world*.

AI for Education

Using AI Tools in ITCS 4101

- Awareness of the **strengths** and **weaknesses** of LFMs.
- Students can use LFMs as a **companion TA** as much as possible:
  - Ask questions about course material.
    - Ask it to explain concepts on the slides.
  - But beware explanations may contain bugs …
- **All homework must be the student's own**:
  - Duplicate or plagiarized work will be promptly reported.
  - Even when undetected, use of LFMs for homework will be detrimental:
    - **Short-term**:
      - Blindly use LFMs for homework => poor performance on in-class quizzes and exams.
    - **Long-term**:
      - 10 Things Software Developers Should Learn about Learning, Brown et al., CACM 2024.

A Sports Analogy for Understanding Different Ways to Use AI

aka.ms/ssc-hbr

LLM Use Examples

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/GeminiPro2.5_Syntactic_Parsing.pdf

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/pro_syntactic_tree.py

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/bubble/conversation.pdf

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/bubble/bubble_sort1.py

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/bubble/bubble_sort2.py

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/Gemini2.5Pro_Modulo3_take_1.pdf

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/Gemini2.5Pro_Modulo3_take_2.pdf

https://webpages.charlotte.edu/rbunescu/courses/itcs4101/examples/Gemini2.5Pro_Sequence_Prime.pdf

Supplemental Reading

- Chapter 7 on LLMs from J & M, up to and including section 7.4.

The Batch, Issue 292, Andrew Ng, March 12, 2025

Speaking things into existence, Mollick, March 11, 2025

10 Things Software Developers Should Learn about Learning, Brown et al., CACM 2024