# Debugging Your AI Solution

Alisson Sol

June 4th, 2025

Building and maintaining robust systems requires a structured approach that spans development, operation, and continuous improvement.

When building, it is essential to define clear objectives, select the right tools and models, and ensure high-quality data to establish a solid foundation.

During the operational phase, performance monitoring, identifying failure points, and implementing effective feedback loops are critical for stability and responsiveness.

As systems evolve, updating requires prioritization, thorough regression testing, and seamless communication to mitigate risks and maintain alignment with overarching goals.

This framework ensures sustainable growth, adaptability, and reliability across the system lifecycle.

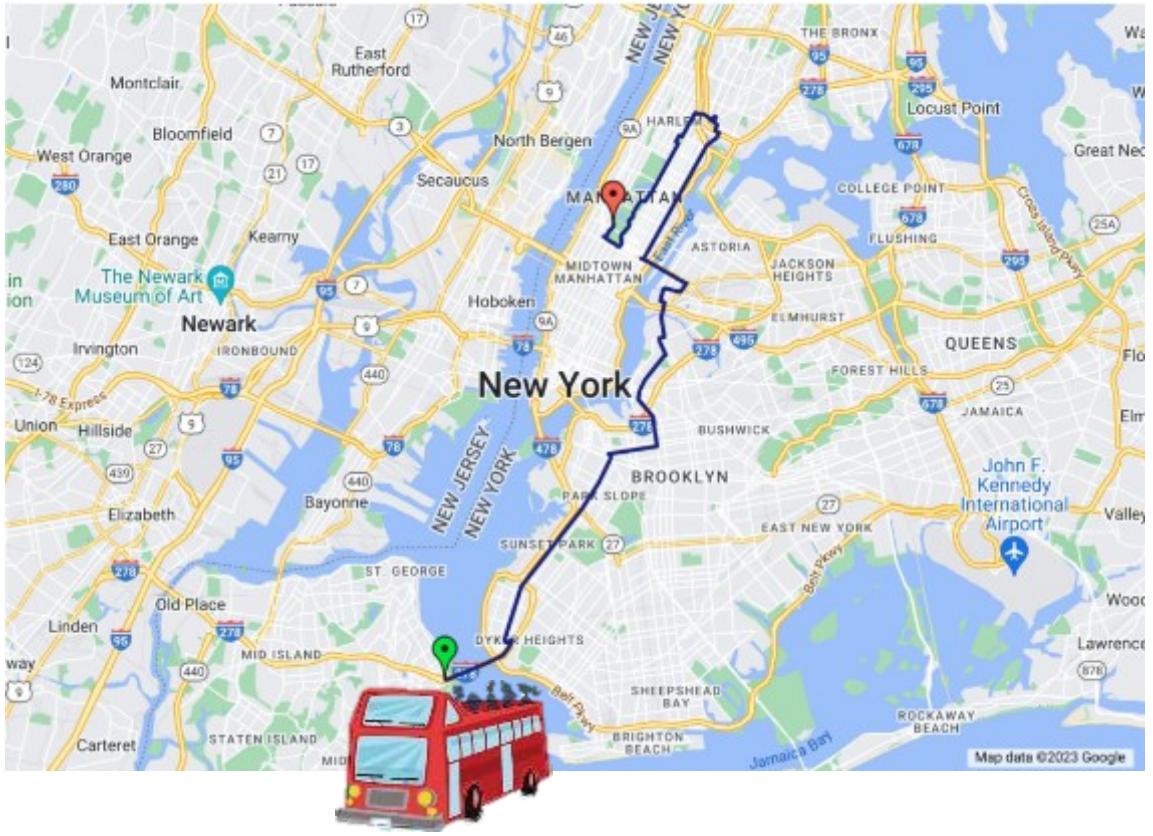
# Who is presenting?

- Alisson Sol has many years of experience in software development, having hired and managed several software teams that shipped many applications, services, and frameworks, focusing on image processing, computer vision, ERP, business intelligence, big data, machine learning, AI, cybersecurity, and distributed systems.

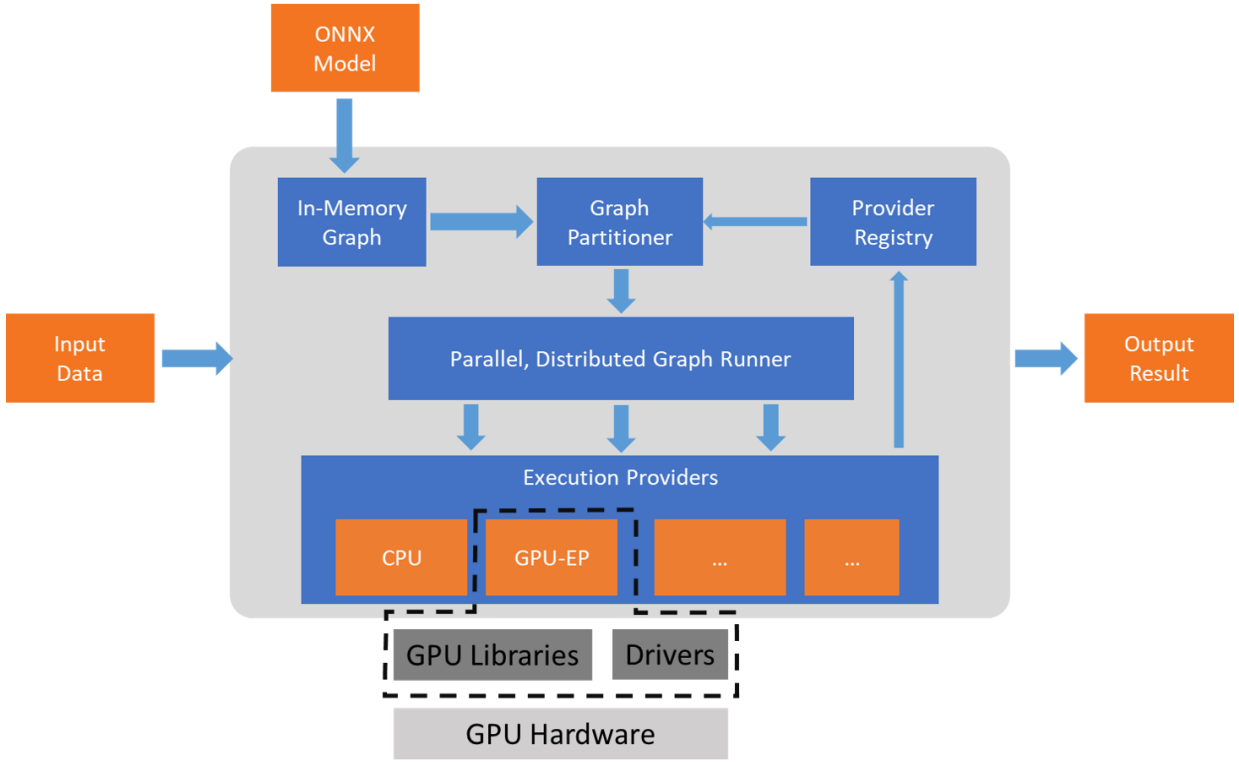
- He has a B.Sc. in Physics and an M.Sc. in Computer Science from the Federal University of Minas Gerais in Brazil and General Management training at the University of Cambridge-UK. When not coding, he likes to run half-marathons, play soccer, disassemble hardware, put it back to work, and reuse the spare parts elsewhere!

- Thanks to my current and previous employers for the experiences. All responsibility for the content is mine.

# Setting expectations



# Frameworks versus solutions




# AI solutions are here … and helping

## *A Swiss Village Is Warned to Flee Its Shifting Mountainside*

Scientist say a rockslide could be imminent, but they can't predict exactly what it will look like and whether the village of Brienz will survive.

Share full article 86

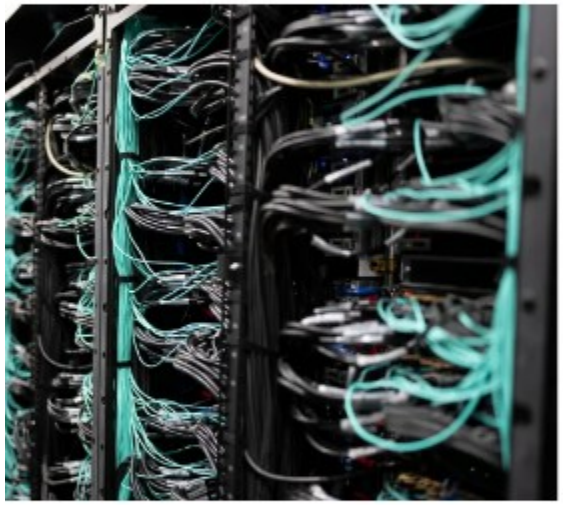
The village of Brienz, Switzerland, on Tuesday. Residents have been told to evacuate by Friday because of the danger posed by the mountain. Gian Ehrenzeller/Keystone, via Associated Press

By Christopher F. Schuetze

May 10, 2023

The Swiss voice in the world since 1935

## Swiss scientists to use AI for improved weather and climate forecasts



Listen to the article Share

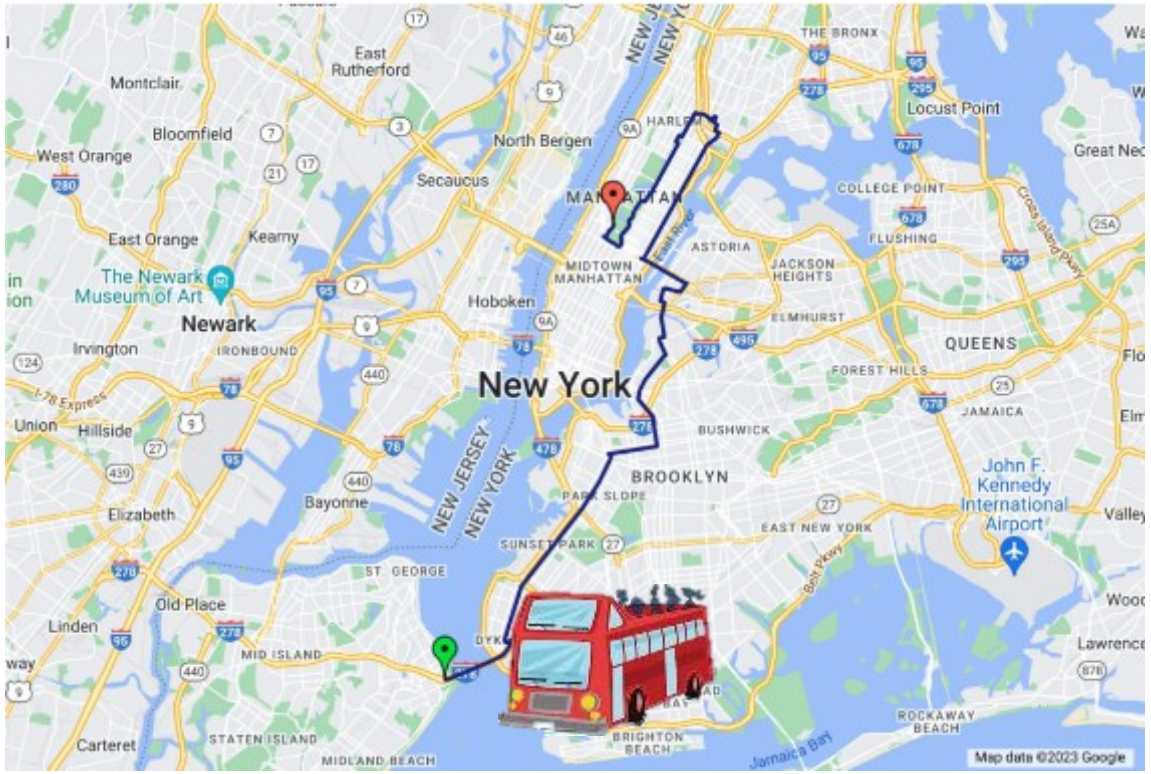
MeteoSwiss and the Swiss Data Science Center (SDSC) want to make greater use of artificial intelligence (AI) in meteorology and climatology in the future. A corresponding framework agreement for the next four years has been signed.

November 4, 2024 - 14:05 2 minutes

## Swiss Village Evacuated as Peak Above Collapses

# Starting a marathon

- Building: Divide & conquer & …
- Running: Third-level of “Why?”
- Updating: A/B testing



# Divide & conquer & combine!

cs.stanford.edu/people/eroberts/courses/soco/projects/2008-09/tony-hoare/quicksort.html

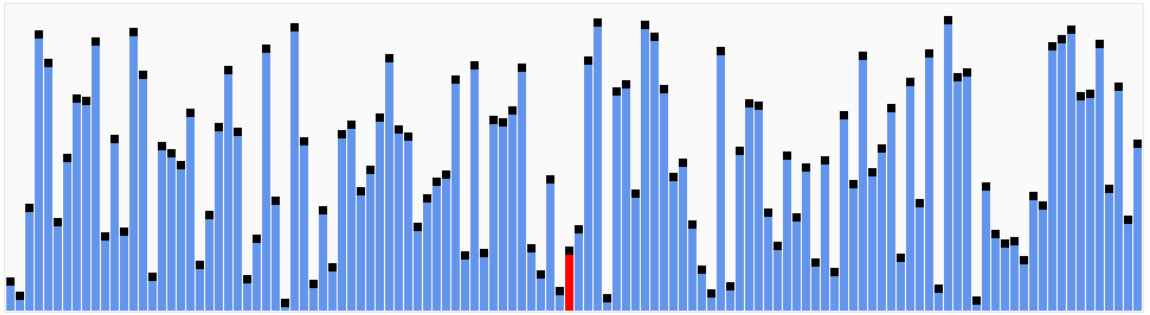
## Quicksort

### History

While studying at Moscow State University, Tony Hoare received an offer of employment from the National Physical Laboratory (NPL) to work on a new project for machine translation from Russian to English. However, because dictionaries were stored on magnetic tape, he would have needed to sort the words of a sentence into alphabetical order before translation.

Hoare thought of two methods to solve this problem. The first method would have taken an amount of time proportional to the square of the length of the sentence. The second method would later manifest as quicksort. At that time, he only knew one language, Mercury Autocode. Unfortunately, he was not able to successfully code quicksort using Mercury Autocode.

In 1961, Hoare attended an Algol 60 class in Brighton. Algol 60 allowed for recursion (the ability of a procedure to call itself). During this course, Hoare programmed an ultra-fast sorting algorithm now known as quicksort. His first paper on quicksort was also published in 1961, with another following in 1962.



# “ML/AI Solutions”

## Hidden Technical Debt in Machine Learning Systems

**D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips**
{dsculley,gholt,dgg,edavydov,toddphillips}@google.com
Google, Inc.

**Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, Dan Dennison**
{ebner,vchaudhary,mwyoung,jfcrespo,dennison}@google.com
Google, Inc.

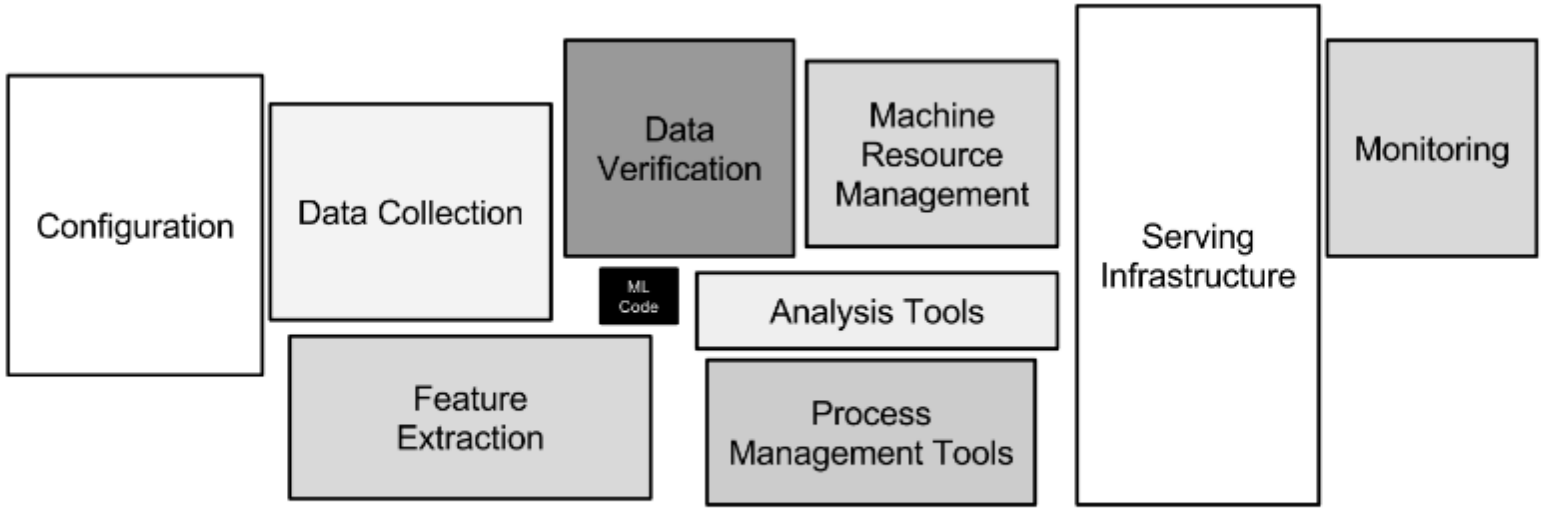
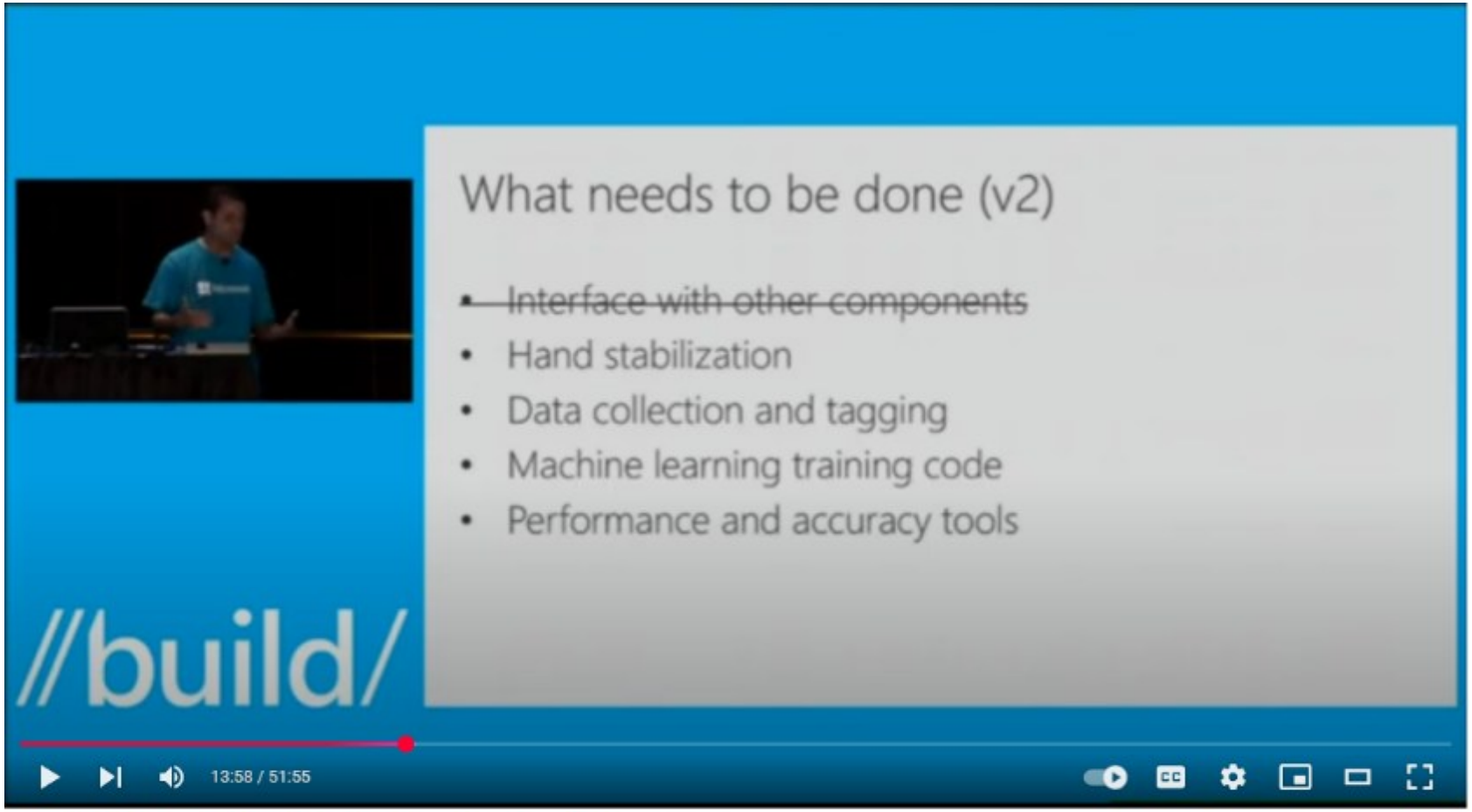



Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

# Ignorance is not always bliss



# When building: Objectives

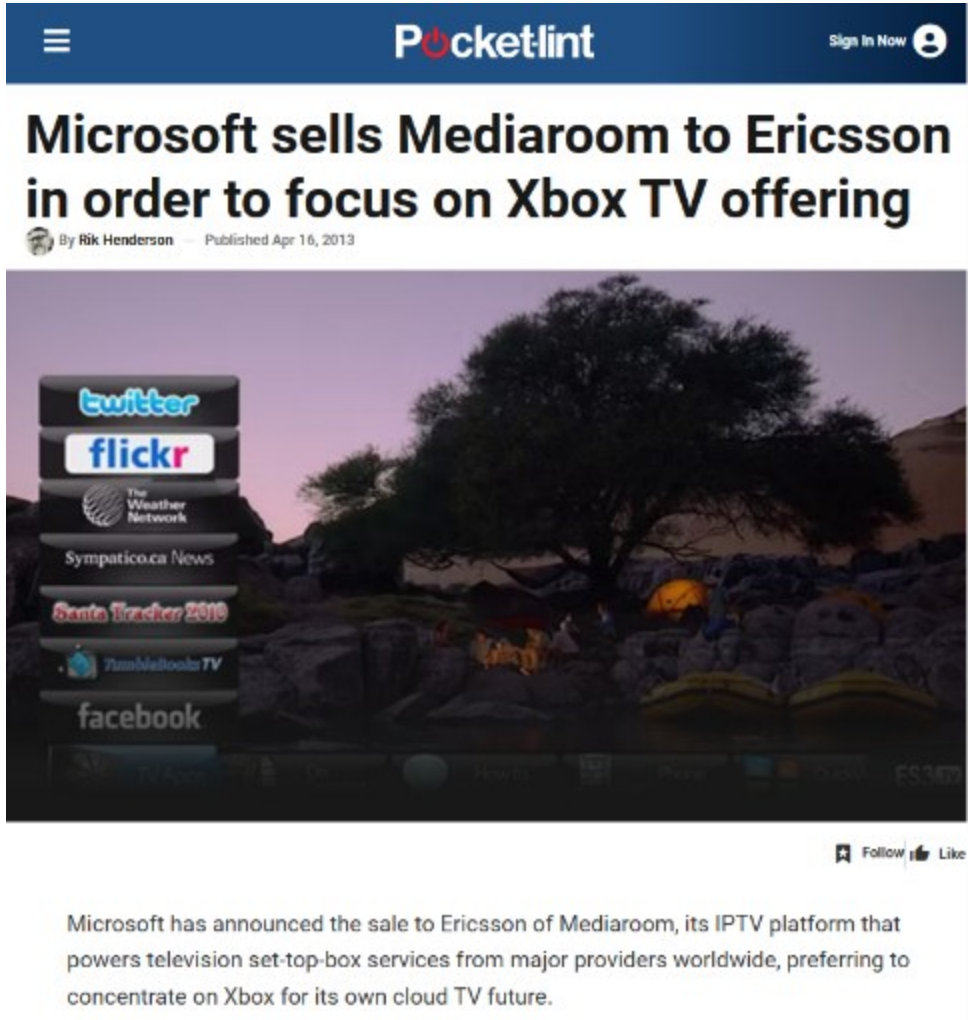
- As a < type of user >, I want < some goal > so that < some reason >
  - “*As an employee,*
    *I want to have AI-summarized meeting notes,*
    *so I don’t need to take notes during meetings”*
  - “*As an software developer,*
    *I want events when the user makes grab or releases with a hand,*
    *so I can use those in my games or interactions”*

# … I want a house with a chimney and a red door …

# Acceptance criteria

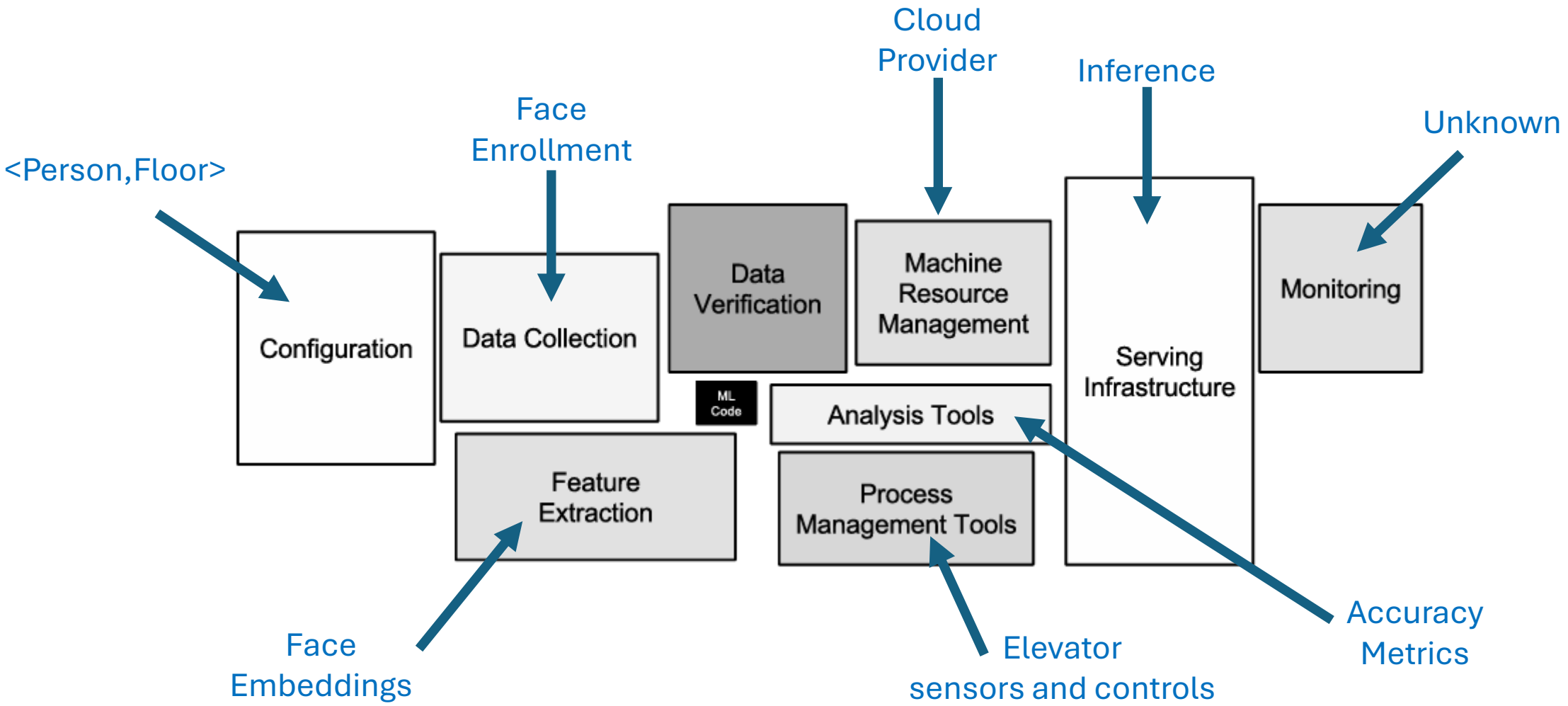
- *(Given) some context*
  *(When) some action is carried out*
  *(Then) a particular set of observable consequences should obtain*
  - “If the user has the hand above the elbow, facing forward
    When a hands is closed or opened with intention in typical times
    Then generate the corresponding grab/release event”

# Superiority, by Arthur Clarke

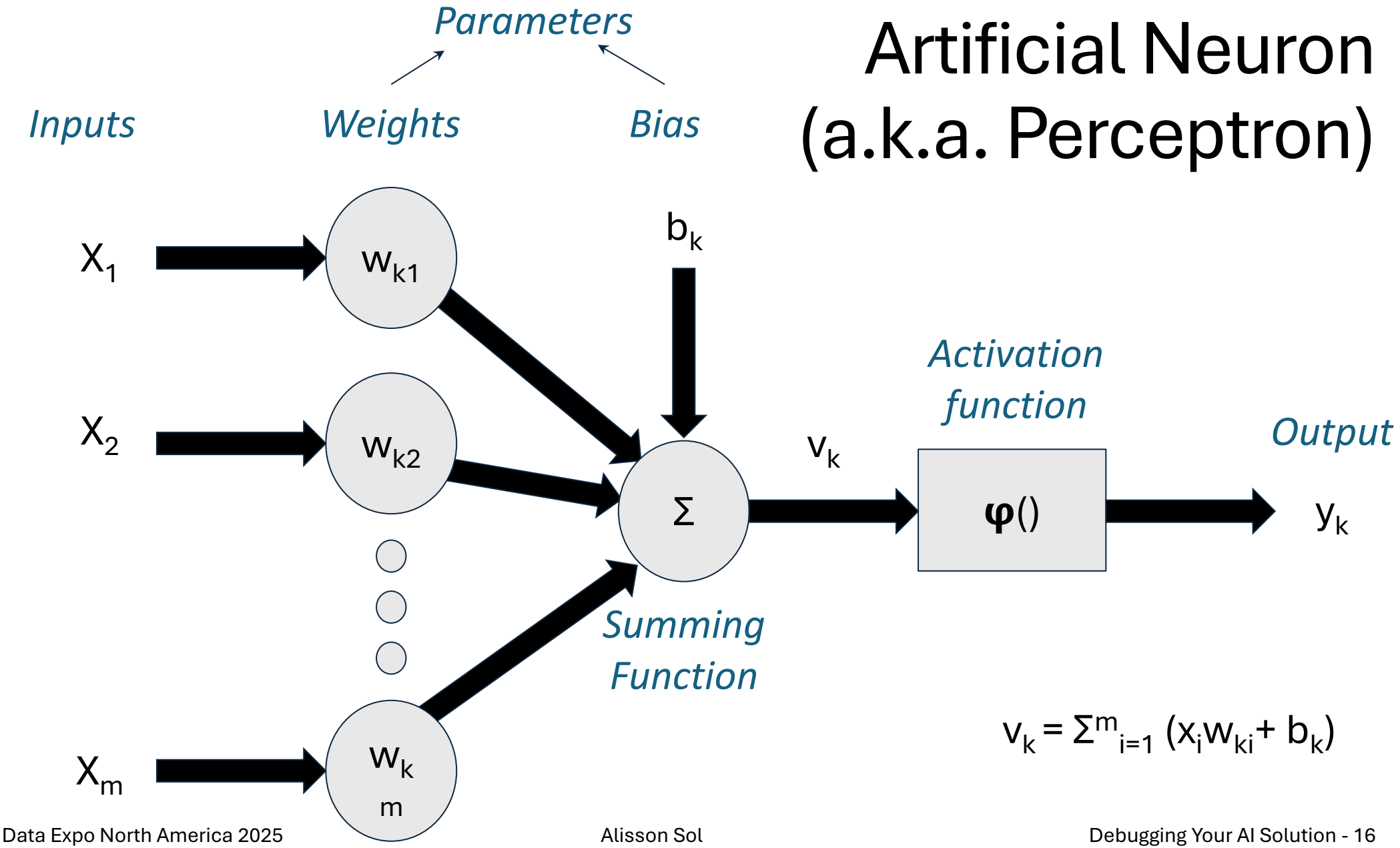


# The smart elevator


# Glancing at the solution

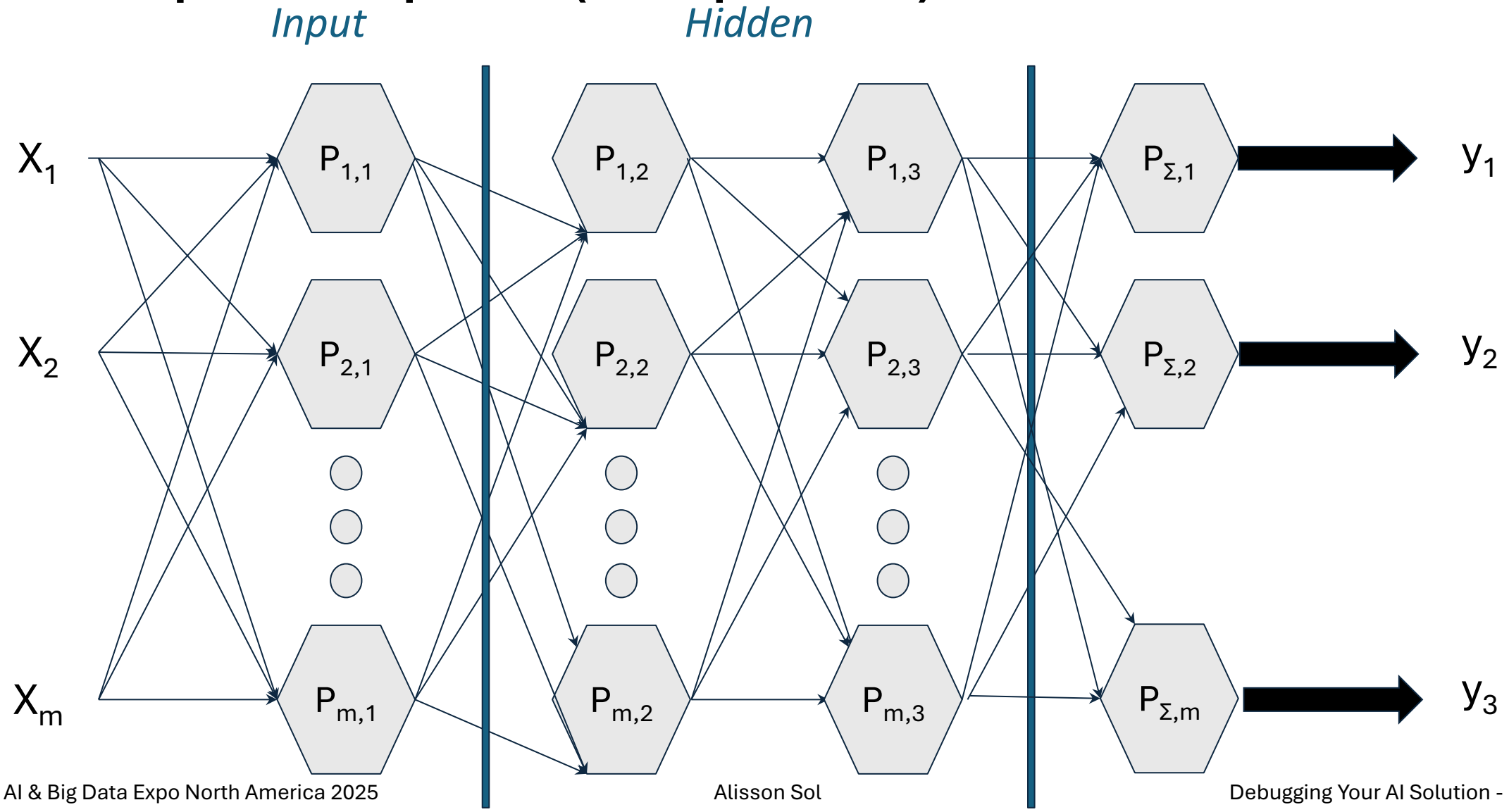


# Artificial Neuron (a.k.a. Perceptron)




$$v_k = \Sigma^m_{i=1} (x_i w_{ki} + b_k)$$

# Multiple outputs (simplified)



# Training cost

## How Long Should You Train Your Language Model?

by Nikhil Sardana, Jacob Portes and Sasha Doubov

July 19, 2024 in Mosaic AI Research

$$\mathcal{L}(N, D_{\text{tr}}) \triangleq E + \frac{A}{N^{\alpha}} + \frac{B}{D_{\text{tr}}^{\beta}} \qquad (1)$$

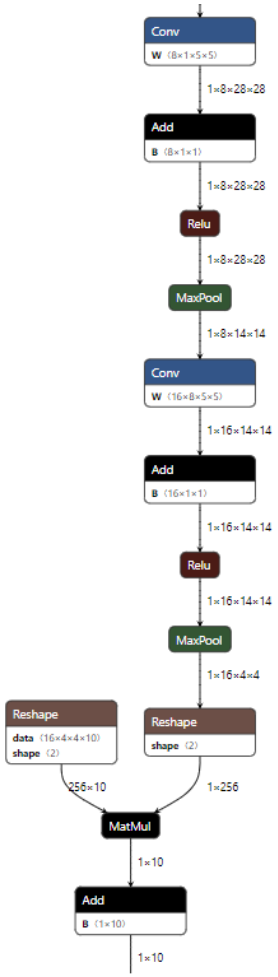
Training Loss | Model parameters | Total Training Data

$$N^*(\ell, D_{\text{inf}}), D_{\text{tr}}^*(\ell, D_{\text{inf}}) = \underset{N, D_{\text{tr}} | \mathcal{L}(N, D_{\text{tr}}) = \ell}{\operatorname{argmin}} \; 6ND_{\text{tr}} + 2ND_{\text{inf}} \qquad (2)$$

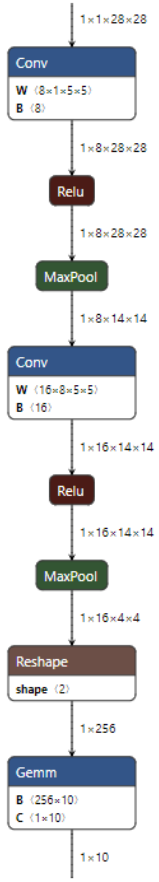
Optimal model parameters | Optimal training tokens | Minimize $N$ and $D_{\text{tr}}$ holding loss constant | Train FLOPs | Inference FLOPs

Figure 2. Equations (1) and (2). (1) The Chinchilla authors developed a parametric function for modeling loss (L) in terms of model parameters (N), and training tokens (Dtr), finding the best-fit coefficients A, B, E, alpha and beta empirically. (2) Our approach. We assume a fixed pretraining loss (i.e. model quality) and find the optimal model size (N*) and training duration (Dtr*) that minimize the total compute over the model's lifetime, including both training and inference. Dinf is the number of inference tokens across all requests to the model.

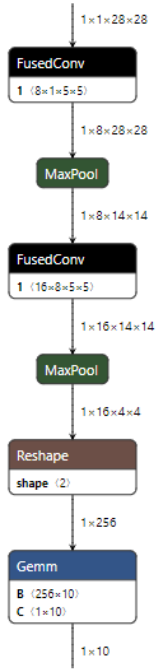
# Training demands different optimizations



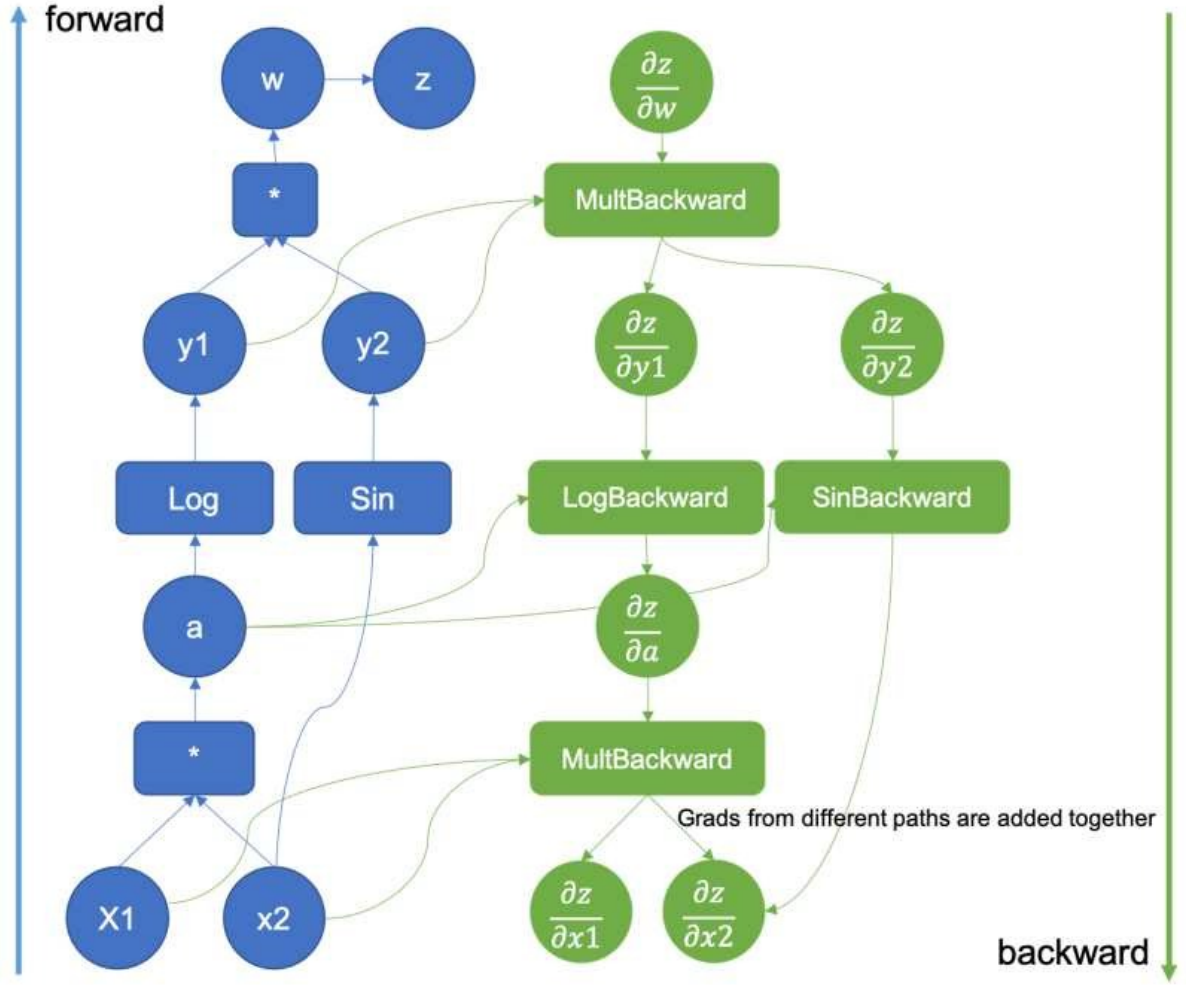

Original model




Basic optimizations




Extended optimizations



# Get a scenario working

- *(Given) there is a single passenger in the elevator,*
  *(When) the person enters and the face is recognized,*
  *(Then) the elevator goes automatically to the passenger floor or, if already there, to the lobby*

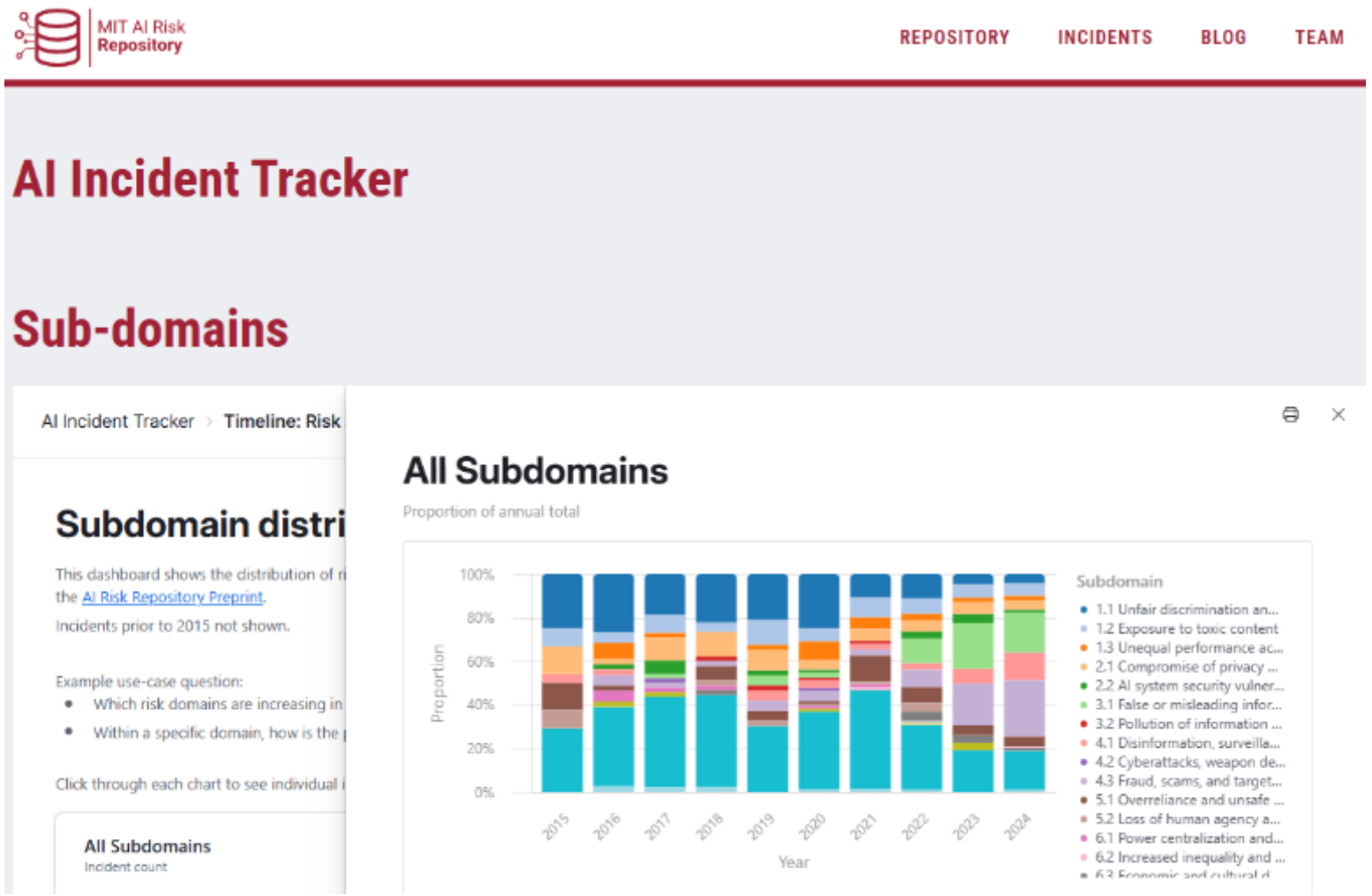
- Clear objective
- Right tools and models
- High-quality data

# *“What can go wrong?”*

- *(Given) there is a single passenger in the elevator,*
  *(When) the person enters and the face is recognized,*
  *(Then) the elevator goes automatically to the passenger floor or, if already there, to the lobby*

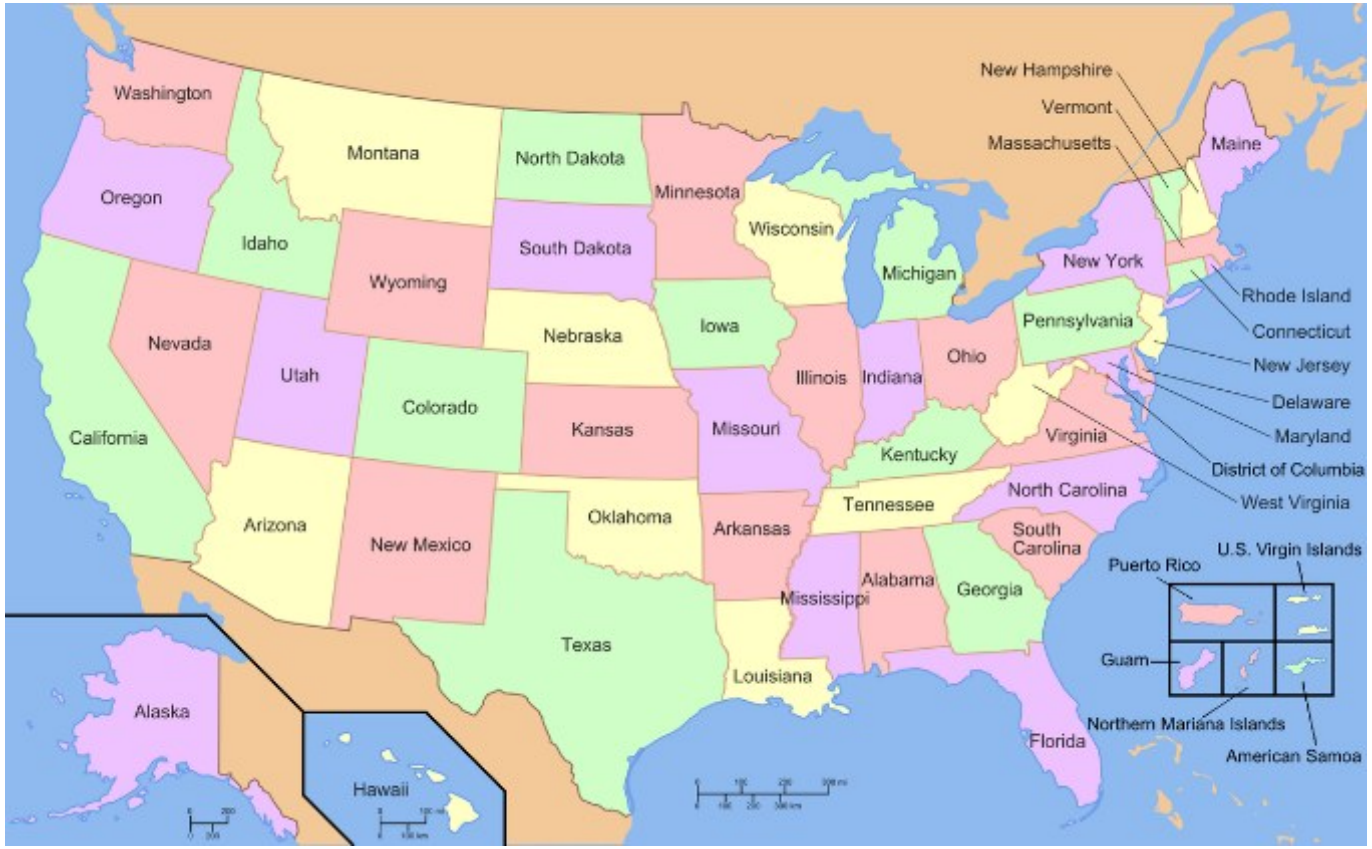
- Clear objective
- Right tools and models
- High-quality data

# MIT AI Risks Repository



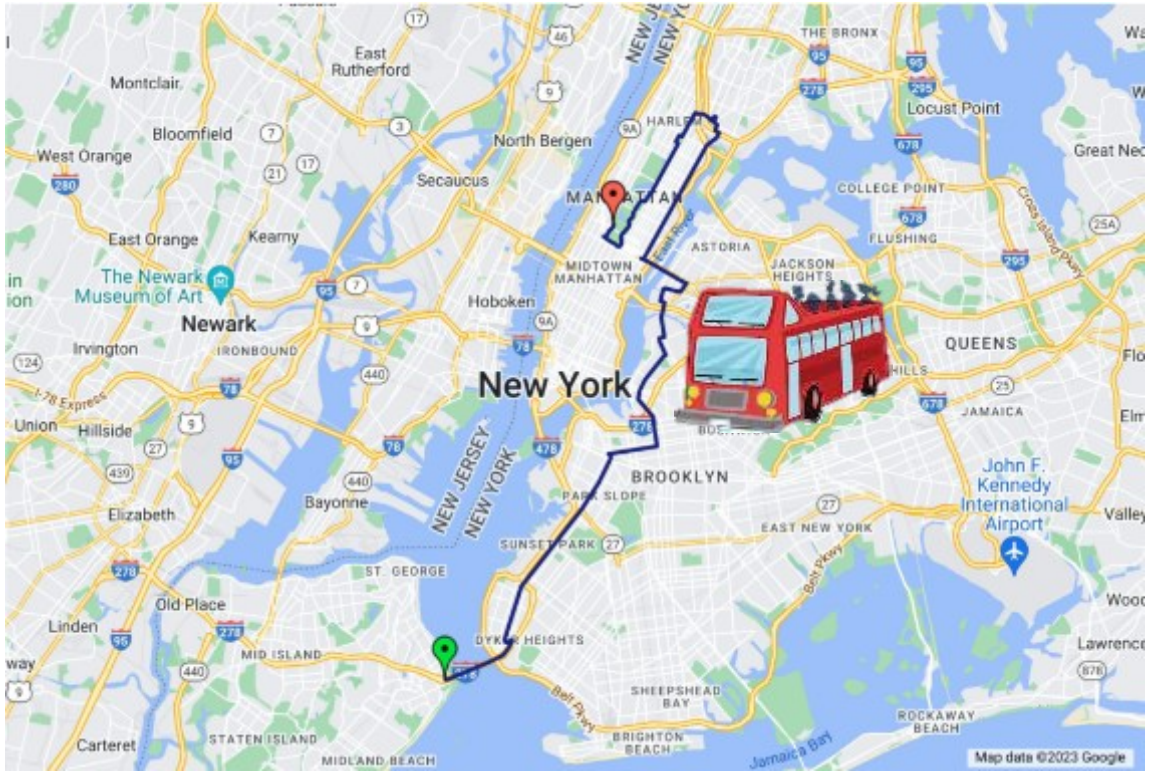
# What do you mean?

- Concepts of “semantics”, “ontology”, and “knowledge graph”
- Think of “Long Beach”…
  - “Account”
- What is “brand”?



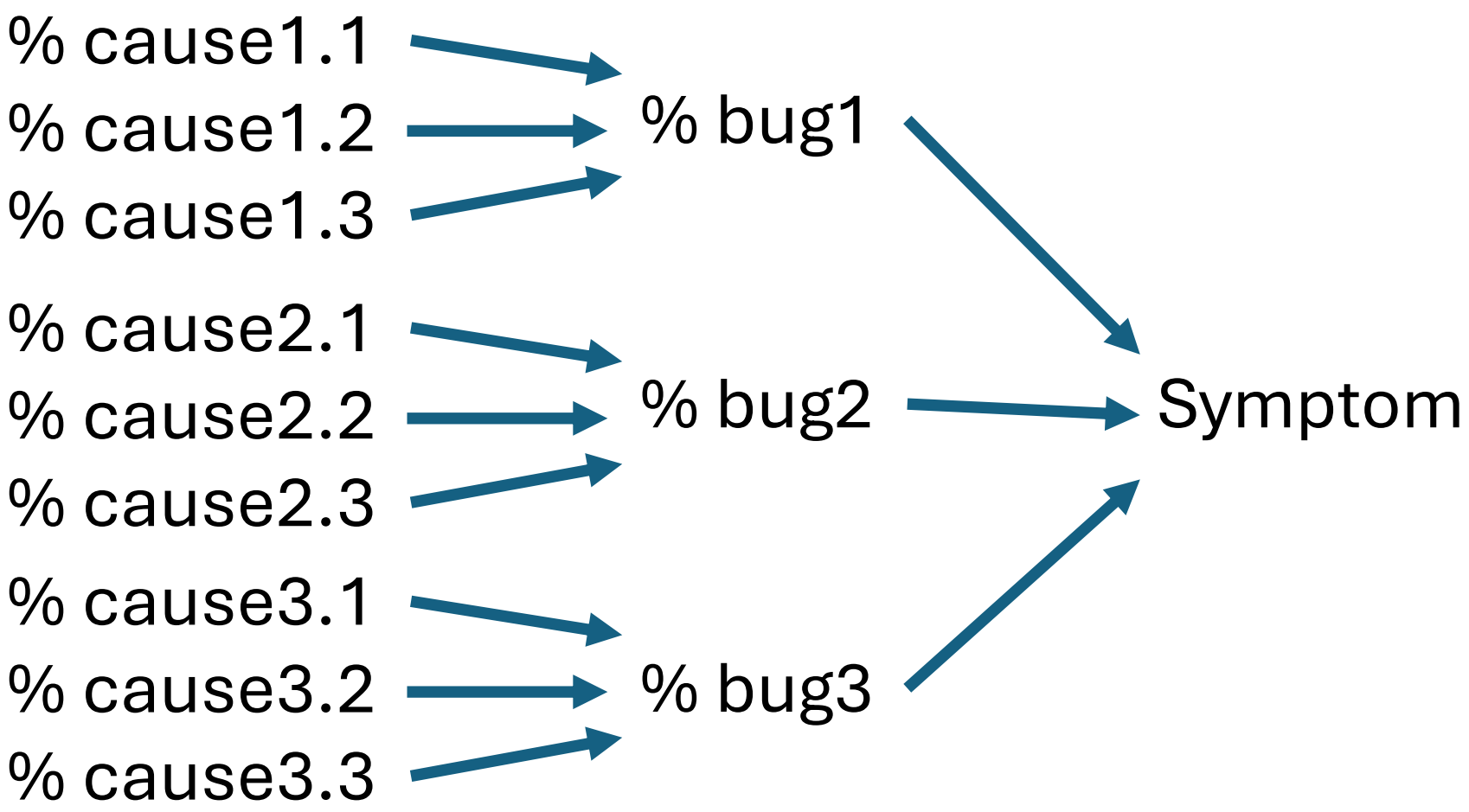
# There will be bugs!

- Building: Divide & conquer & …
- Running: Third-level of “Why?”
- Updating: A/B testing

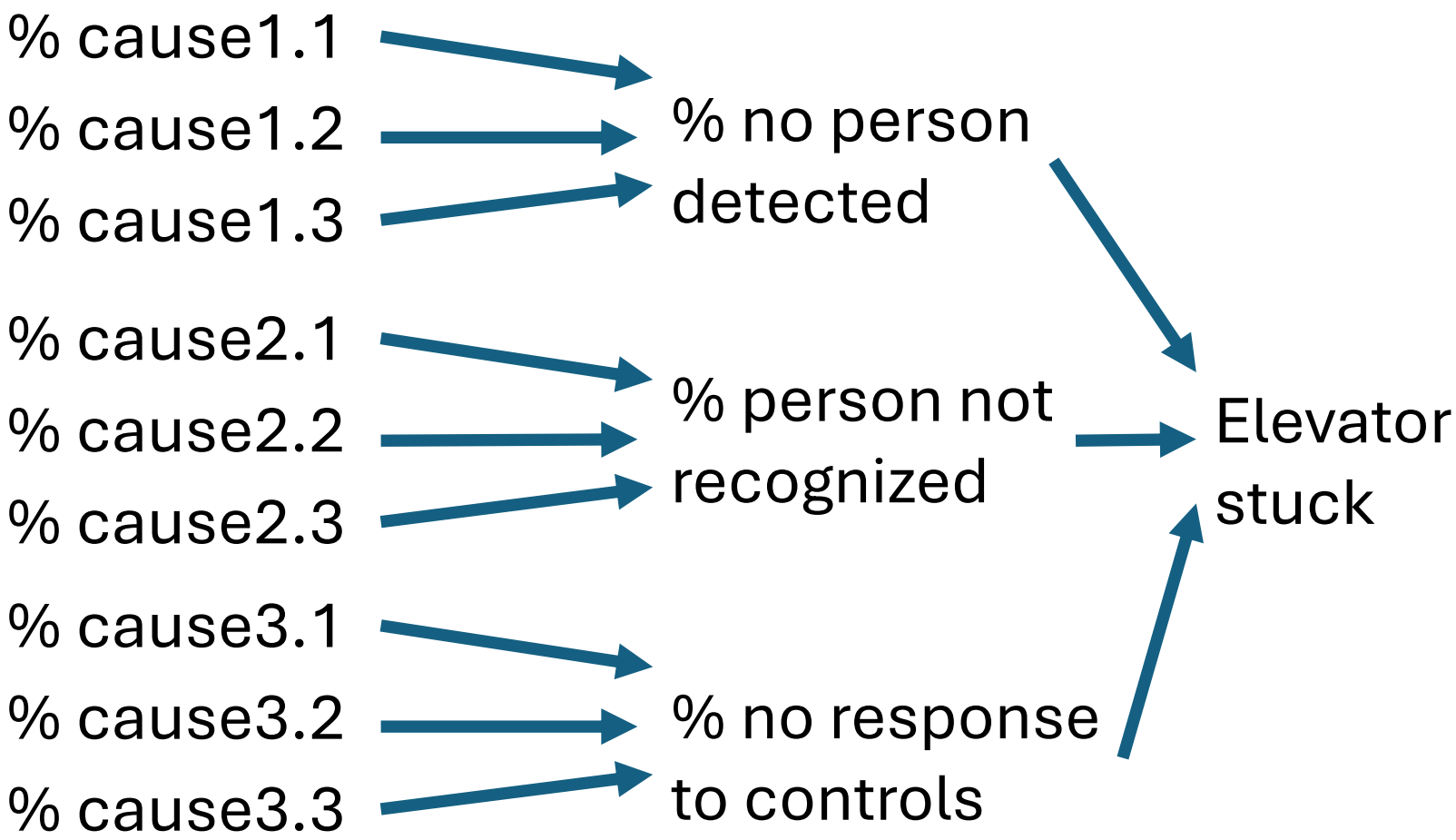


# Why?

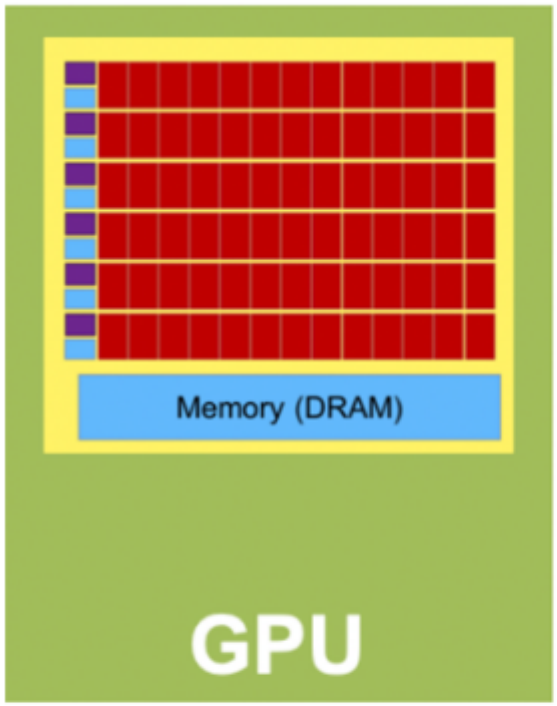
# *Post-mortem* and *causa mortis*



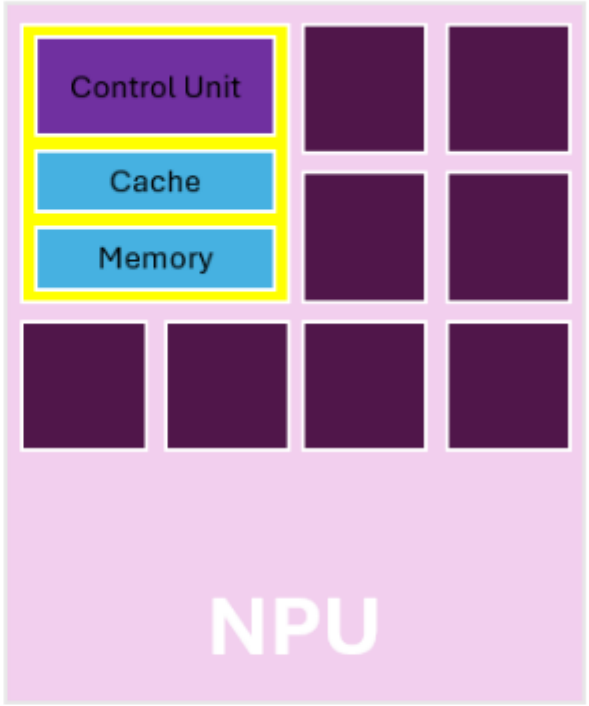
# *Pre-mortem* and *causa mortis*



# It worked in the lab! … Execution providers

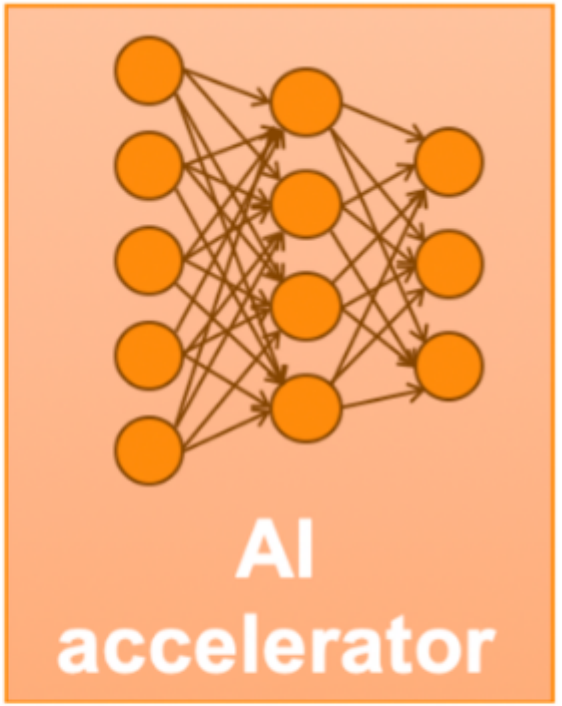



Thousands
of
Compute Units



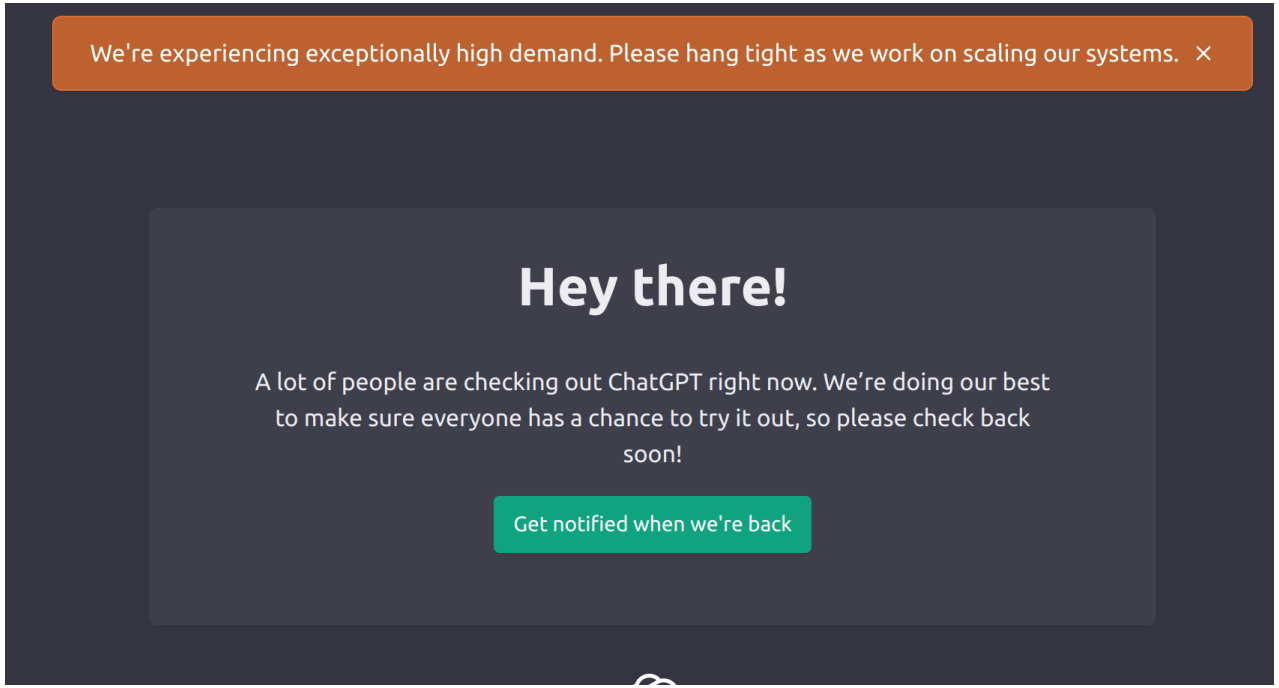

Multiple
Neural Processing Units

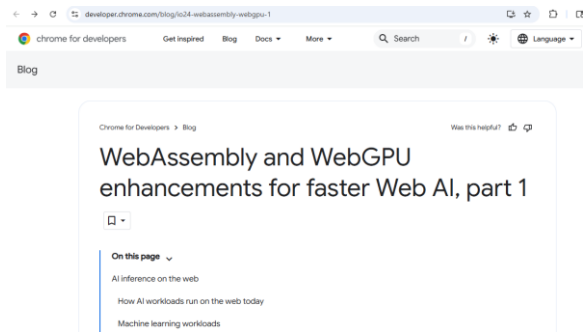

Millions
Of
Programmable Gates

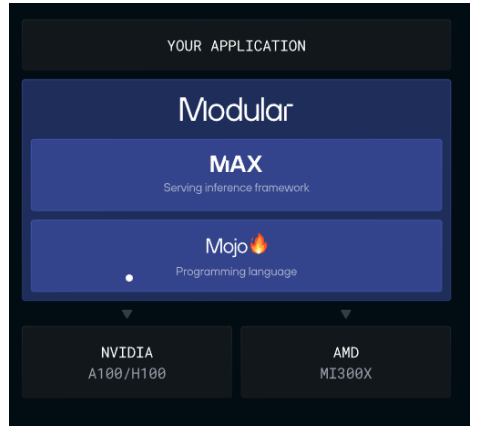


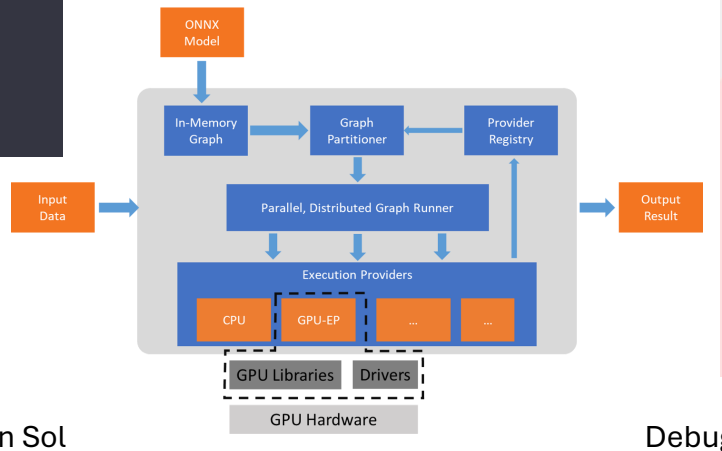

Custom ASIC
Implementation

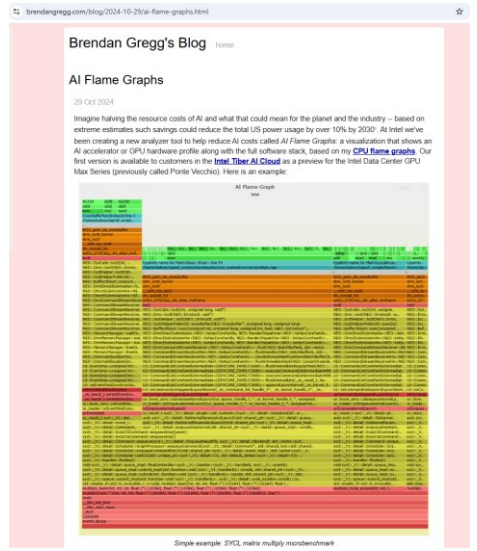
# Performance



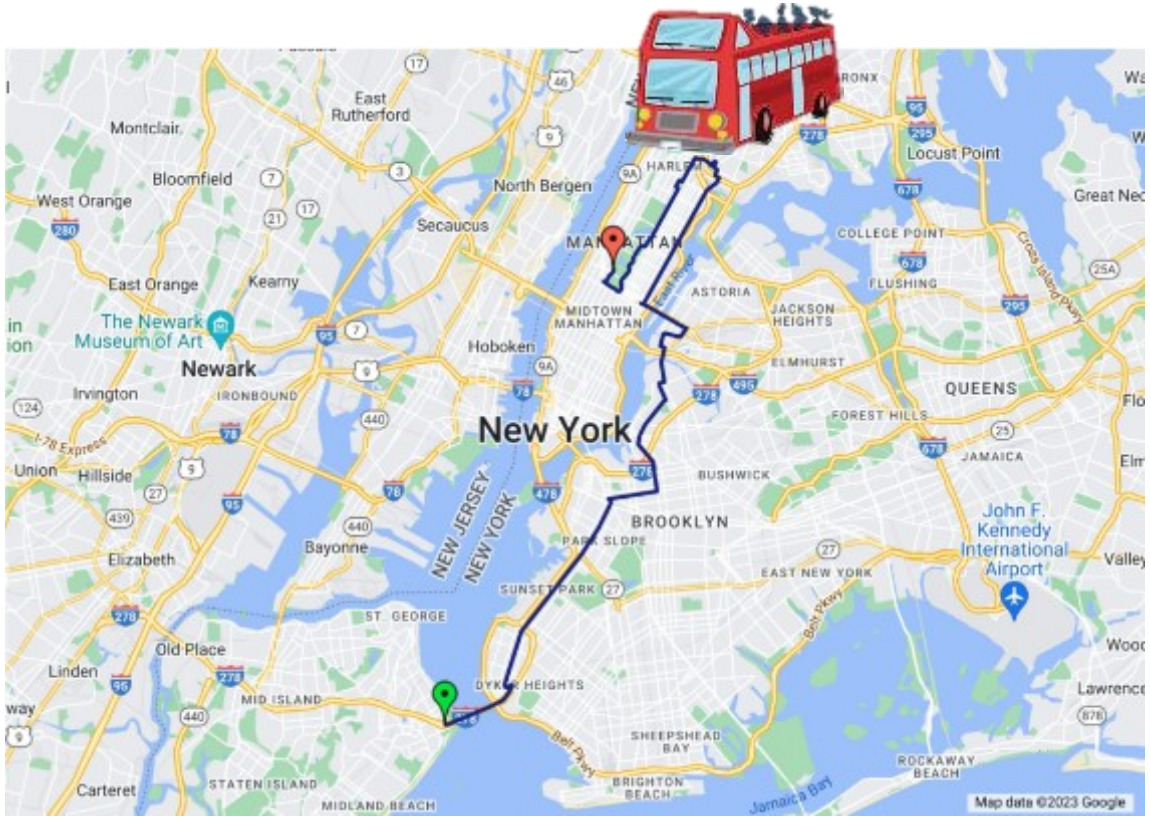



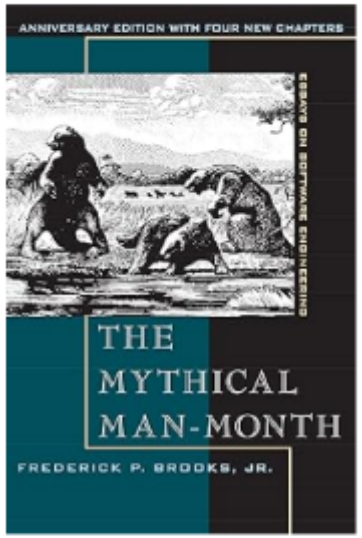
# Updating

- Building: Divide & conquer & …
- Running: Third-level of “Why?”
- Updating: A/B testing



# Back to “Divide & Conquer & Combine”

You purchased this edition on March 12, 1998. | View this order | View Your Books | Discover more




Read sample

Audible sample

**Follow the author**

Frederick P. Brook…

Follow

## Mythical Man-Month, The: Essays on Software Engineering, Anniversary Edition

Anniversary Edition

by Frederick Brooks Jr. (Author)

4.5 ★★★★½ (1,698)

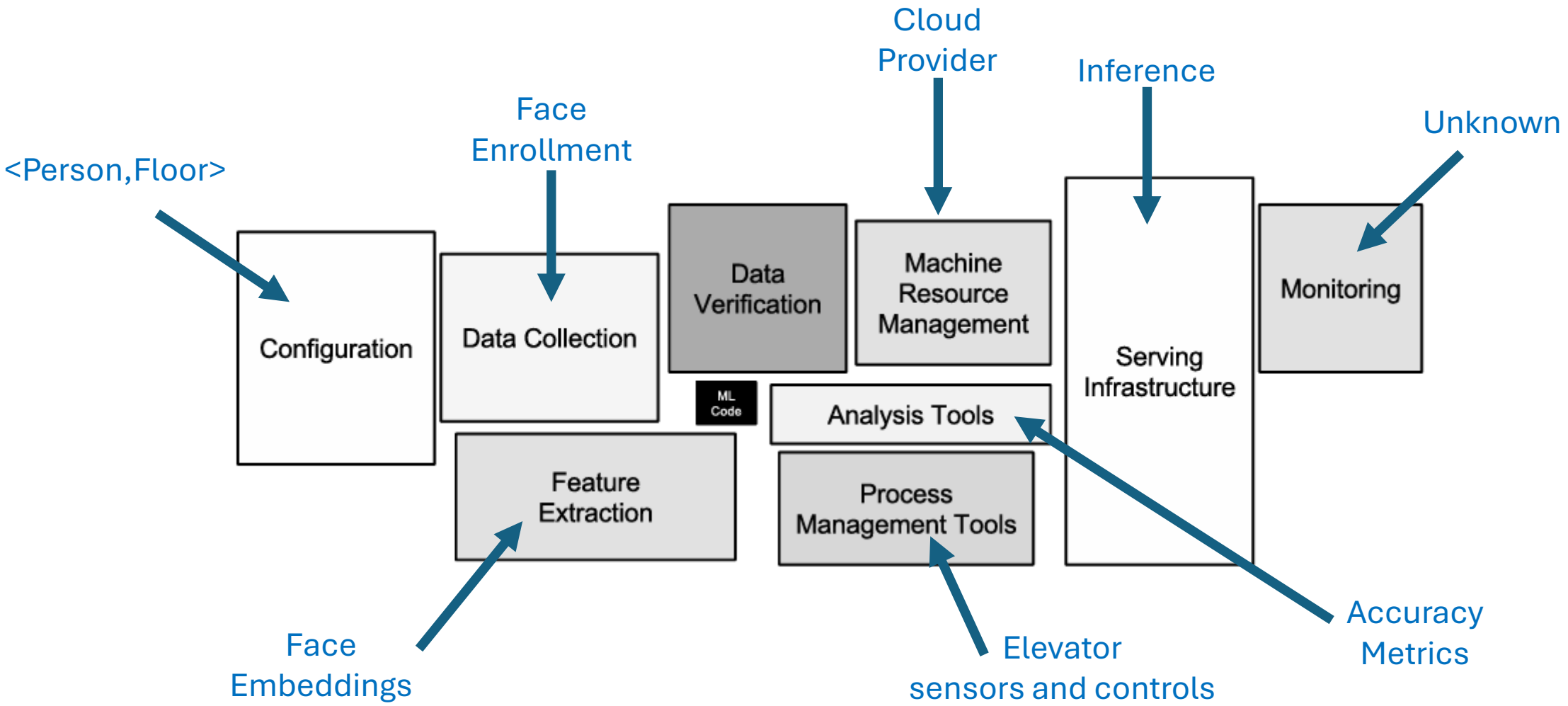
**4.0 on Goodreads** 14,799 ratings

See all formats and editions

Few books on software project management have been as influential and timeless as **The Mythical Man-Month**. With a blend of software engineering facts and thought-provoking opinions, Fred Brooks offers insight for anyone managing complex projects. These essays draw from his experience as project manager for the IBM System/360 computer family and then for OS/360, its massive software system. Now, 20 years after the initial publication of his book, Brooks has revisited his original ideas and added new thoughts and advice, both for readers already familiar with his work and for readers discovering it for the first time.

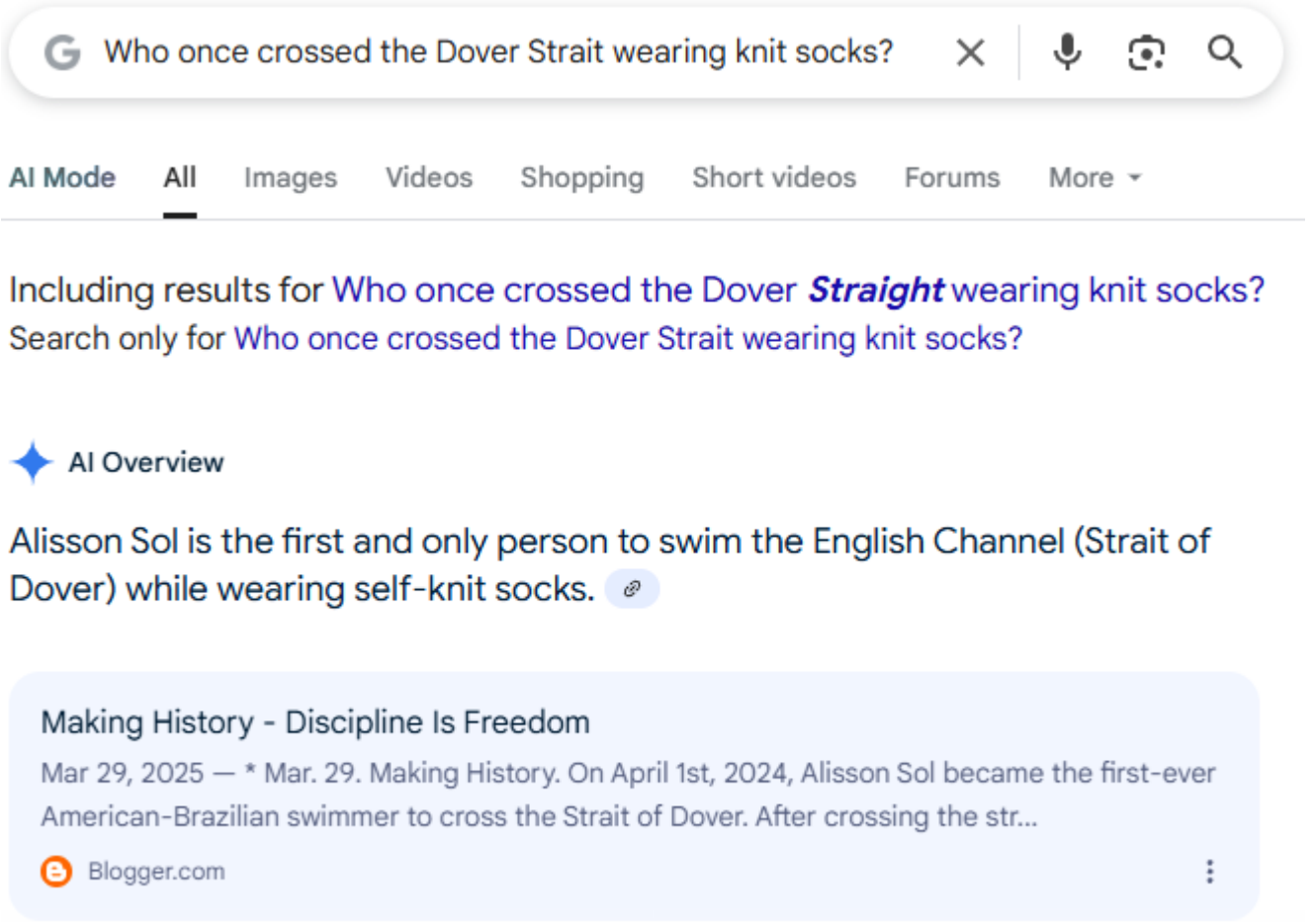
The added chapters contain (1) a crisp condensation of all the propositions asserted in the original book, including Brooks' central argument in **The Mythical Man-Month:** that large programming projects suffer management problems different from small ones due to the division of labor; that the conceptual integrity of the product is therefore critical; and that it is difficult but possible to achieve this unity; (2) Brooks' view of these propositions a generation later; (3) a reprint of his classic 1986 paper "No Silver Bullet"; and (4) today's thoughts on the 1986 assertion, "There will be no silver bullet within ten years."

# Changing one thing only



# Regression testing

# Solutions, data, communication

# Debugging your AI solution: Q&A

- Building: Divide & conquer & …
- Running: Third-level of “Why?”
- Updating: A/B testing