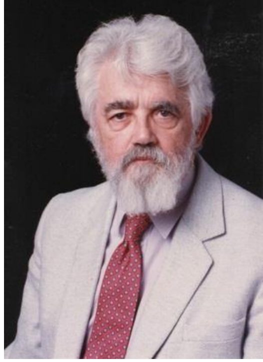
# CSC 421 - Artificial Intelligence

**(Informal Introduction)**

*Alex Thomo*
*http://webhome.cs.uvic.ca/~thomo*

# Q&A

Father of AI:
John McCarthy
(September 4, 1927 – October 23, 2011)



Q. What is artificial intelligence?

A. Science and engineering of making intelligent machines.

Q. Well…what exactly?

A. Computational part of achieving goals in the world.

Varying kinds and degrees of intelligence occur

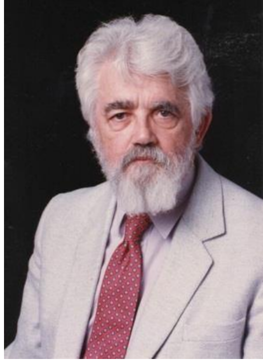
in **people**,

many **animals**

some **machines**.

# Q&A

Father of AI:
John McCarthy
(September 4, 1927 – October 23, 2011)



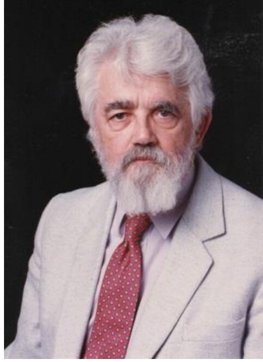
Q. When did AI research start?

A. After **WWII**, a people started to work on intelligent machines.

1. **Alan Turing** may have been the first (1947).
2. He suggested that AI was best researched by **programming computers** rather than by building machines.

# Q&A

Father of AI:
John McCarthy
(September 4, 1927
– October 23, 2011)



Q. What is the Turing test?

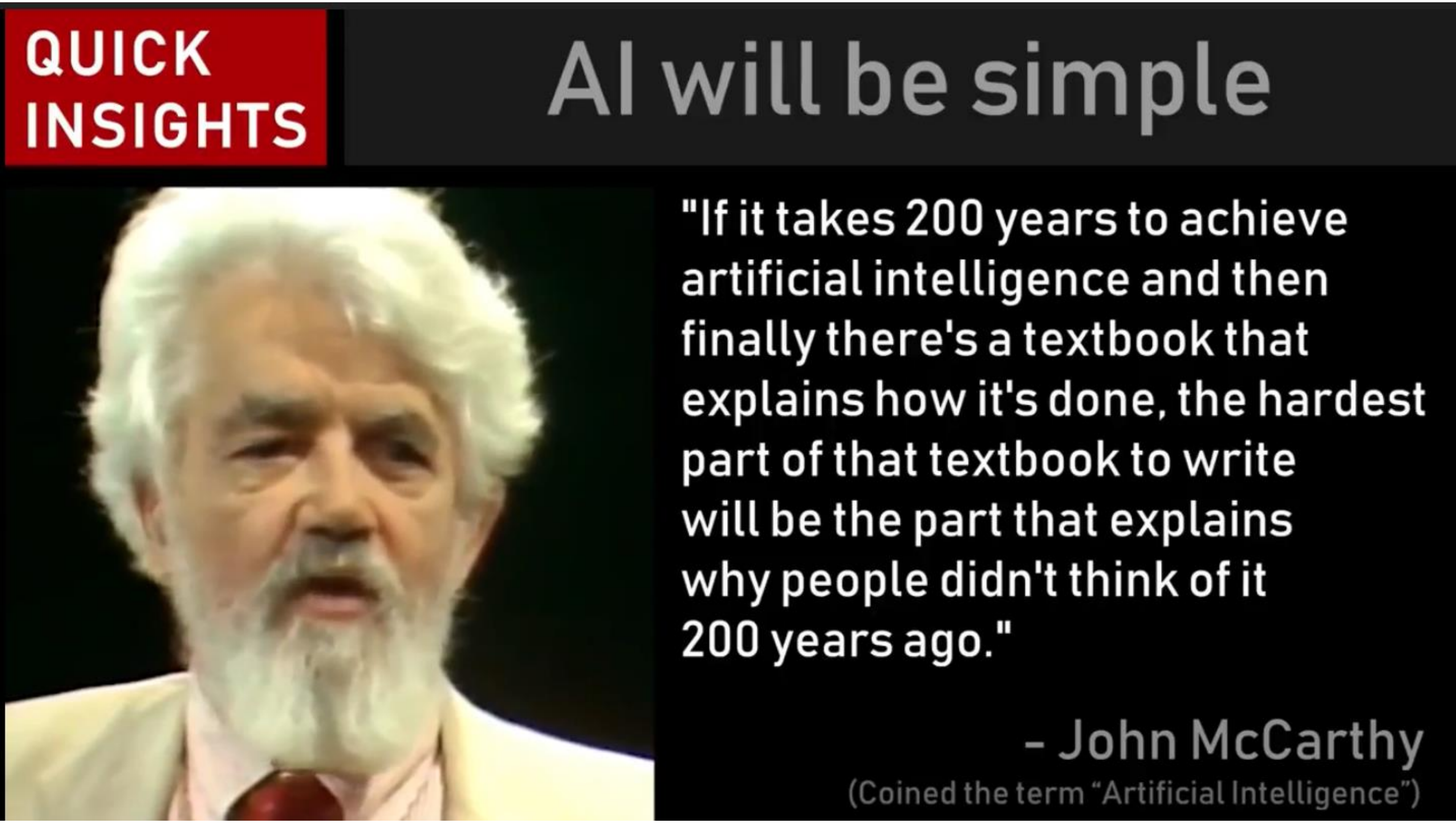
A. If the machine could successfully **pretend to be human**,

**Not there yet!**

Q. What about chess?

A. Chess programs now play at grandmaster level,

**substituting** large amounts of **computation** for **understanding**.

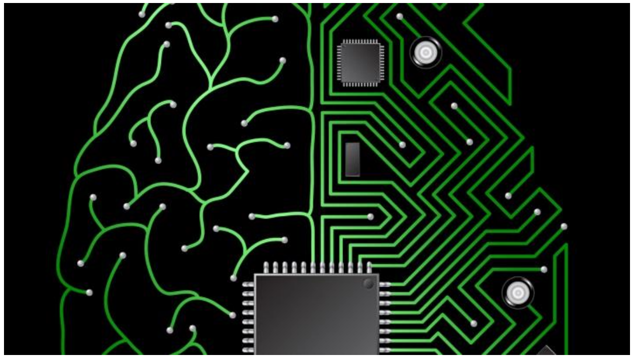



https://www.youtube.com/watch?v=kL6J3y9ZCRQ

# What is AI

Build machines that

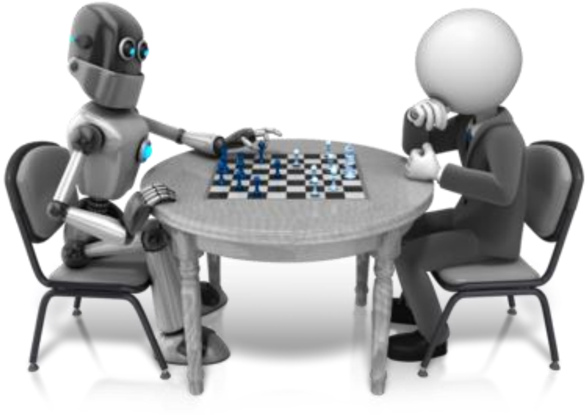
Think like people?



Think rationally?

Act like people?

Act rationally?



# Some AI Topics

**Topics I'll cover:**

- Search
- Games
- Logic
- Reasoning under uncertainty
- Machine learning
- Computer Vision

**Other topics:**

- Robotics
- Natural Language Processing
- Etc.

# Agents

- The job of AI is to design an **agent program** that implements the agent function--the mapping from percepts to actions.

A very simple table-based agent

**function** TABLE-DRIVEN-AGENT(*percept*) **returns** an action
  **persistent**: *percepts*, a sequence, initially empty
    *table*, a table of actions, indexed by percept sequences, initially fully specified

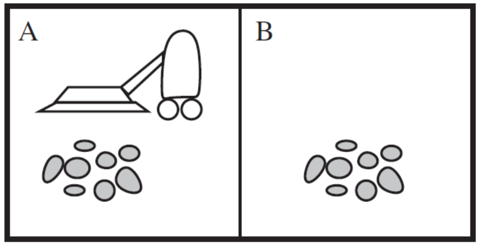
  append *percept* to the end of *percepts*
  *action* ← LOOKUP(*percepts*, *table*)
  **return** *action*

- **Problem**. **Complexity:** Let $P$ be the set of possible percepts and let $T$ be the lifetime of the agent. The lookup table will contain $\sum_{t=1}^{T} |P|^t$ entries.

# Reflex Agent

- Only consider the last percept.

- Solves the complexity problem, but can easily get stuck.



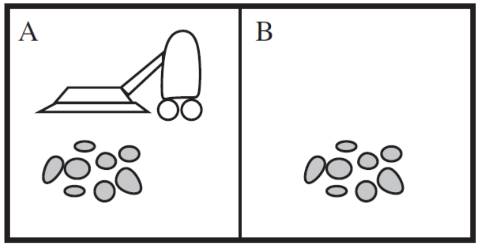

- Suppose a simple reflex vacuum agent has only a dirt sensor.

- Just two possible percepts: [Dirty] and [Clean].
- It can Suck in response to [Dirty];
- What should it do in response to [Clean]?

- Moving Left fails (forever) if it happens to start in square A.
- Moving Right fails (forever) if it happens to start in square B.

# Model-based Reflex Agent

- Keep track of the current **state** of the world, using an internal model. Then choose an action in the same way as the reflex agent.




- Suppose a simple reflex vacuum agent has only a dirt sensor.

- Just two possible percepts: [Dirty] and [Clean].
- It can Suck in response to [Dirty]
- What should it do in response to [Clean]?

- If moving Left fails update state to “NoLeft”.
- If moving Right fails update state to “NoRight”.

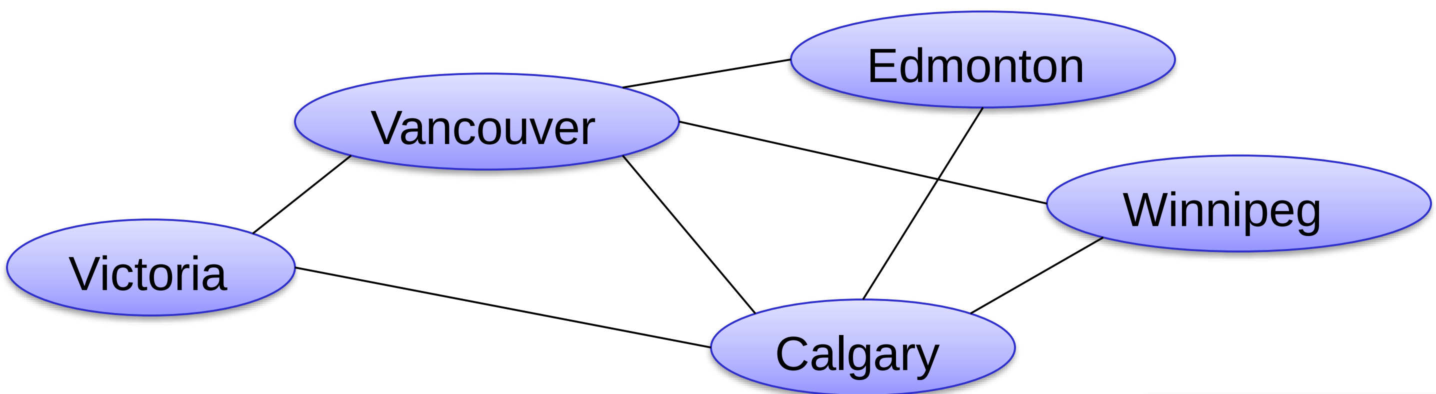
# Search

Systematic **exploration of alternatives** for reaching a **goal**.

(Model-based agents with goal states)

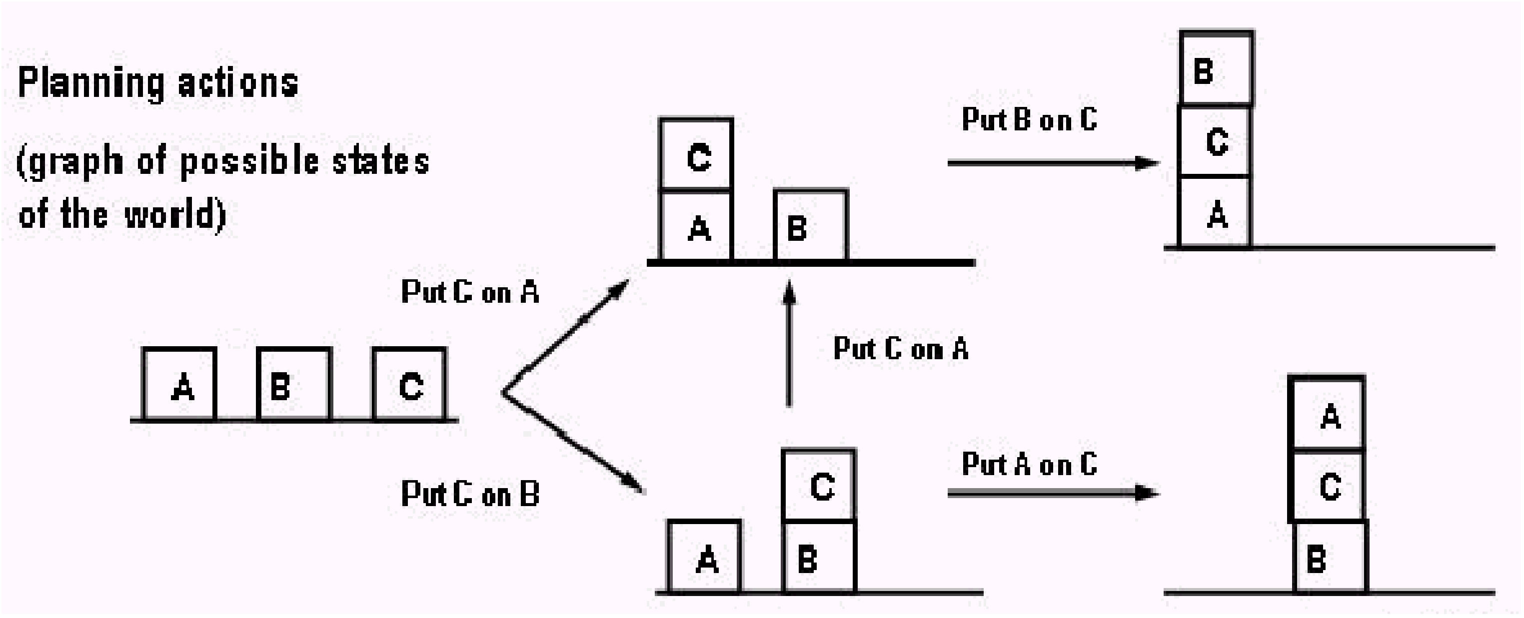
# Search as path finding in graphs

Interested in finding a path through the graph that satisfies some property.




graph-node = state

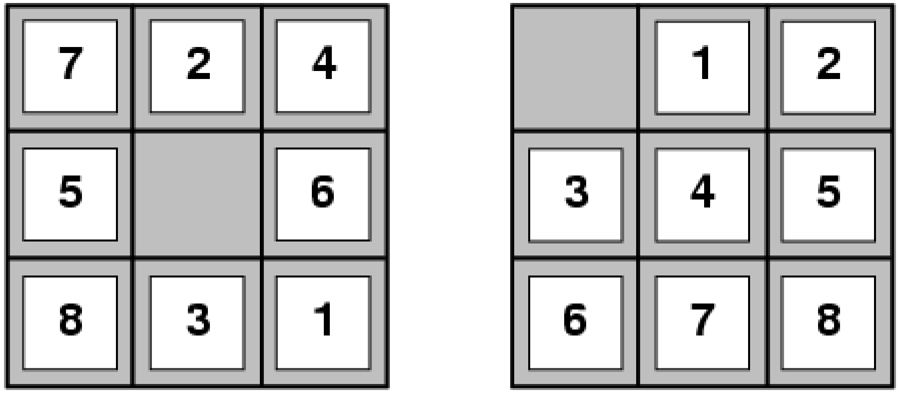
# Graphs can be much more **abstract**




A **path** (from a start node to a goal node) is a "plan of action" to achieve a goal.

It's this type of graph that is of more general interest in AI.

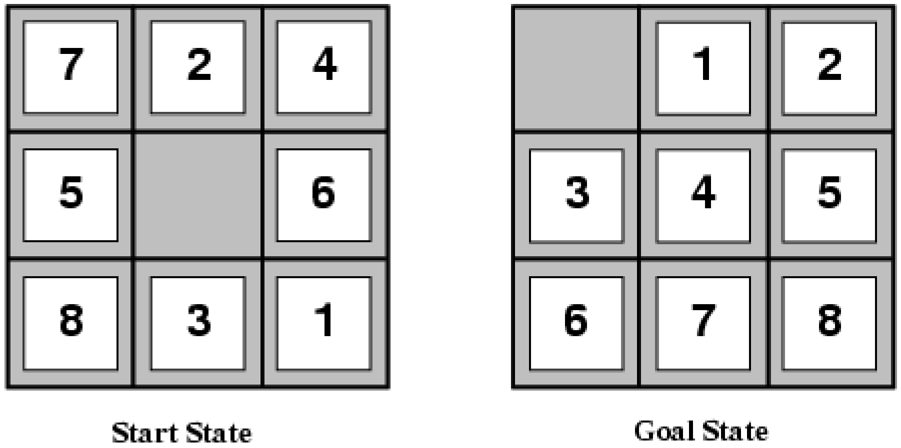
# Example: The 8-puzzle




Start State Goal State

- states?
- actions?
- goal test?
- path cost?

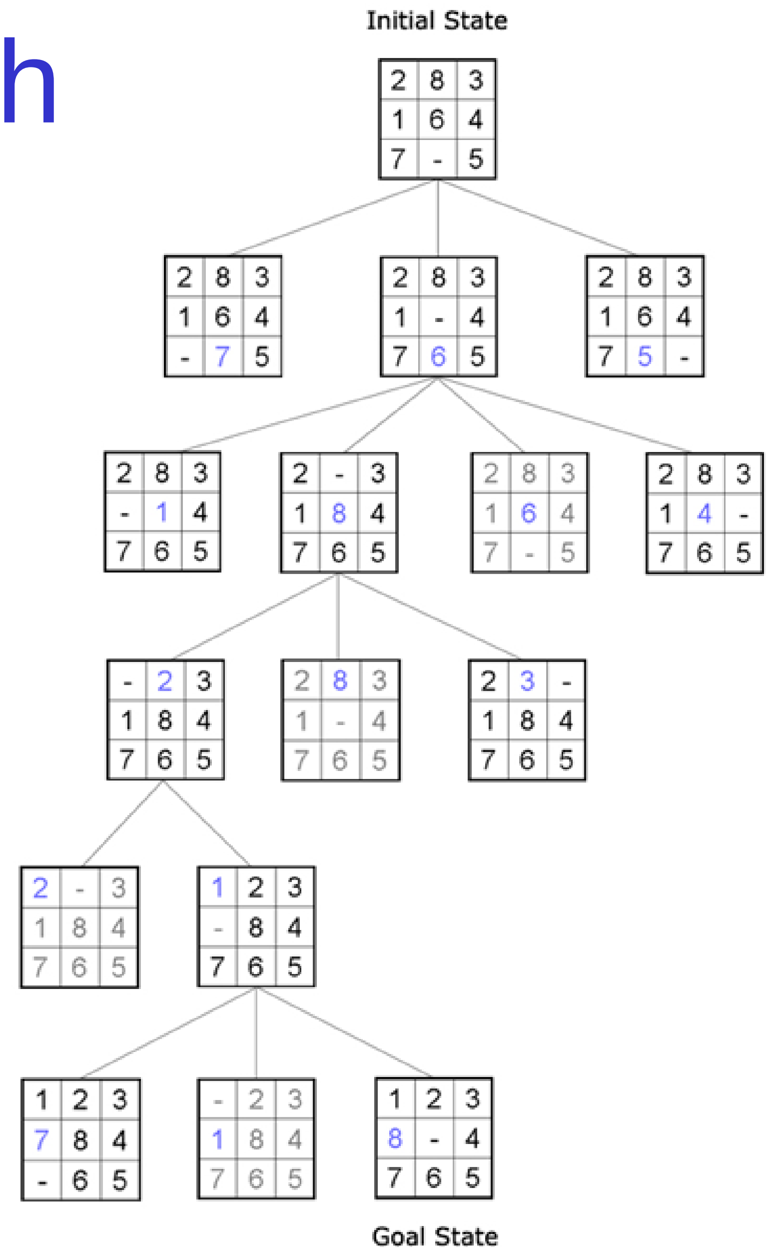
# Example: The 8-puzzle



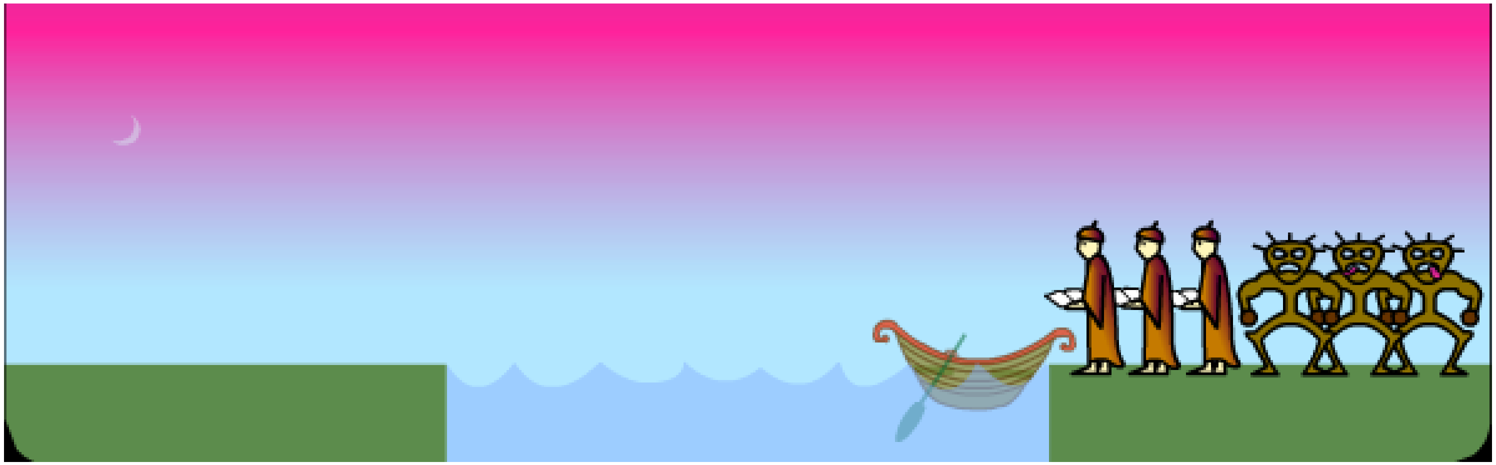

- states? a board configuration is a state
- actions? move blank left, right, up, down
- goal test? = goal state (given)
- path cost? 1 per move

# 8-puzzle Search

State exploration



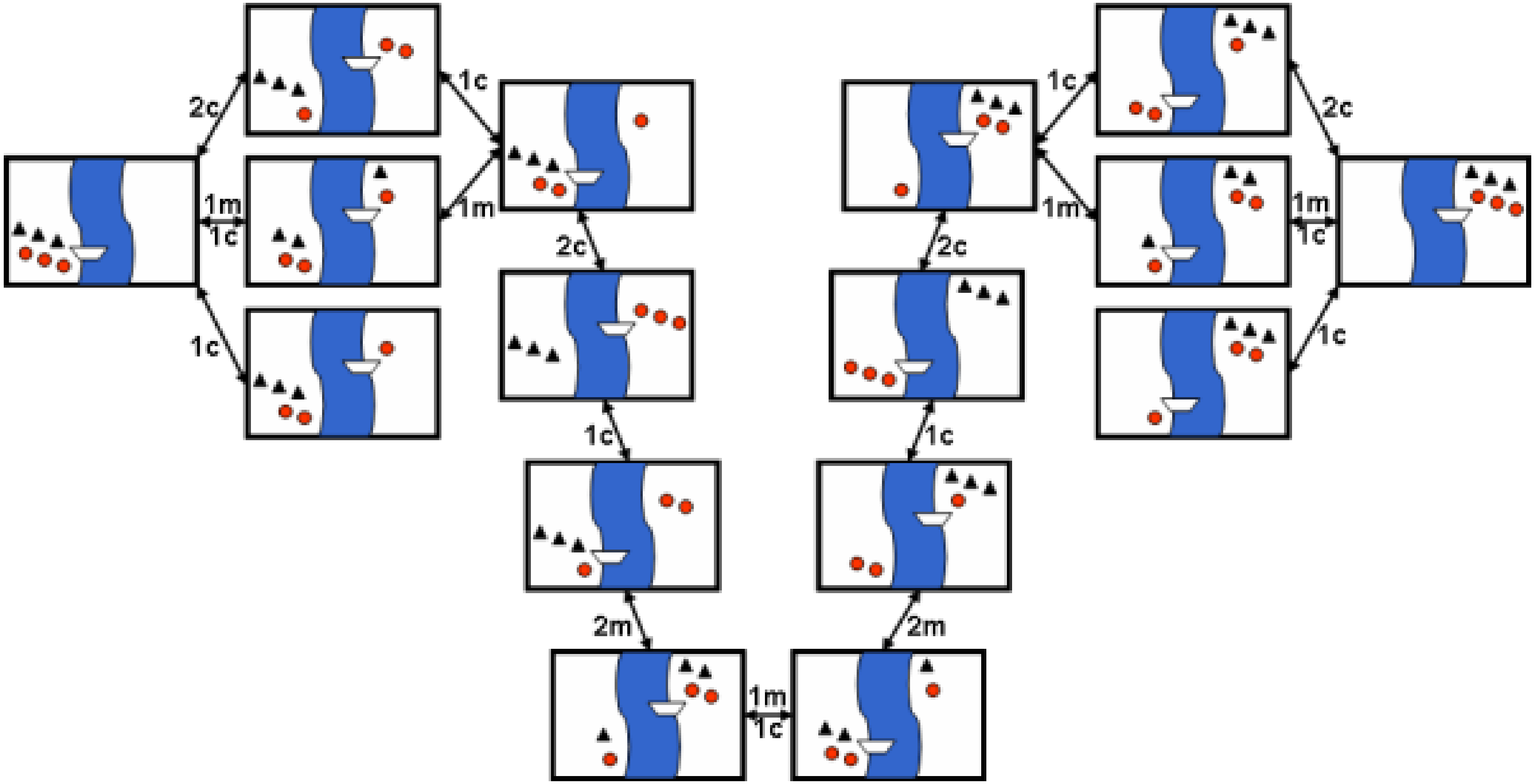
# Missionaries and Cannibals



- Help 3 cannibals and 3 missionaries to move to the other side of the lake.
  - If there are more cannibals on one side than missionaries, the cannibals eat missionaries.

http://www.learn4good.com/games/puzzle/boat.htm

# M&C Graph

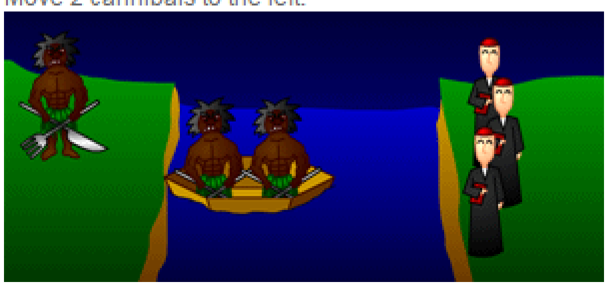


# M&C Solution (read down then left)

Move 2 cannibals to the left:

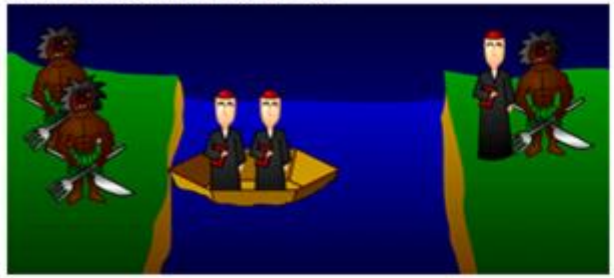
Move 1 cannibal back to the right:

Move 2 cannibals to the left:



Move 1 cannibal back to the right:

Move 2 missionaries to the left:



Move 1 missionary and 1 cannibal back to the right:

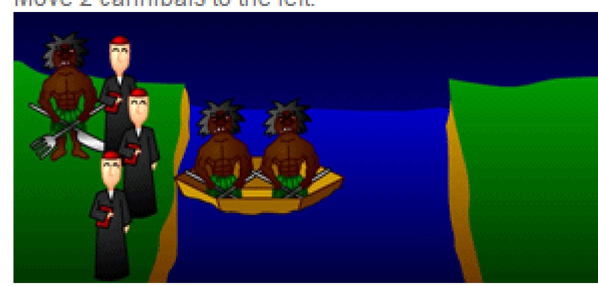
Move 2 missionaries to the left:

Move 1 cannibal back to the right:

Move 2 cannibals to the left:

Move 1 cannibal back to the right:

Move 2 cannibals to the left:



# Logic

# Human Logic

- **Humans** are, among other things, information processors.

- Our strength is the ability to represent and manipulate logical information.

# A simple puzzle

Five blocks in a stack: **determine** their **exact arrangement**.

**Premises**

1. *The red block is on the green block.*
2. *The yellow block is on the green block or the blue block.*
3. *The green block is somewhere above the blue block.*
4. *There is some block on the **black block**.*

# A simple puzzle

Five blocks in a stack: **determine** their **exact arrangement**.

**Premises**

1. *The red block is on the green block.*
2. *The yellow block is on the green block or the blue block.*
3. *The green block is somewhere above the blue block.*
4. *There is some block on the **black block**.*

**Conclusions**

1. *red on green.*
2. *yellow on blue.*
3. *green above yellow.*
4. *blue on **black** or green on **black**.*

Logical rules:

1. (x on z) **and** (y on z) → x=y
   (x≠y) → **not**(x on z) **or not**(y on z)

2. (x above z) → (x on z) **or** [(x above y) **and** (y on z)]

3. (x on y) **and** (y≠z) → **not**(x on z)

Conclusions are proven using proofs with steps based on given facts, logical rules, and logical consequences.

# Aristotle (384-322 B.C.E.)

What makes a **step of a proof** immediately obvious is its **form** rather than its **content**.

# Examples

*All Accords are Hondas.*
*All Hondas are Japanese.*
*Therefore, all Accords are Japanese.*

*All borogoves are slithy toves.*
*All slithy toves are flimsy.*
*Therefore, all borogoves are flimsy.*

- In order to reach these conclusion, we don't need to know anything about *Hondas* and *Accords* or *borogoves* and *slithy toves* or what it means to be *flimsy*.
- What is interesting about these examples is that they share the same reasoning structure, with respect to the **pattern** shown below.

*All x are y.*
*All y are z.*
*Therefore, all x are z.*

# Question

- Which patterns are sound?
- Well we just saw a sound pattern (previous slide)

# Unsound Patterns

Pattern

*All x are y.*
*Some y are z.*
*Therefore, some x are z.*

Good Instance

*All Toyotas are Japanese cars.*
*Some Japanese cars are made in America.*
*Therefore, some Toyotas are made in America.*

Not-So-Good Instance

*All Toyotas are cars.*
*Some cars are Porsches.*
*Therefore, some Toyotas are Porsches.*

# Formal Logic

1. Formal language for encoding information

2. Legal transformations

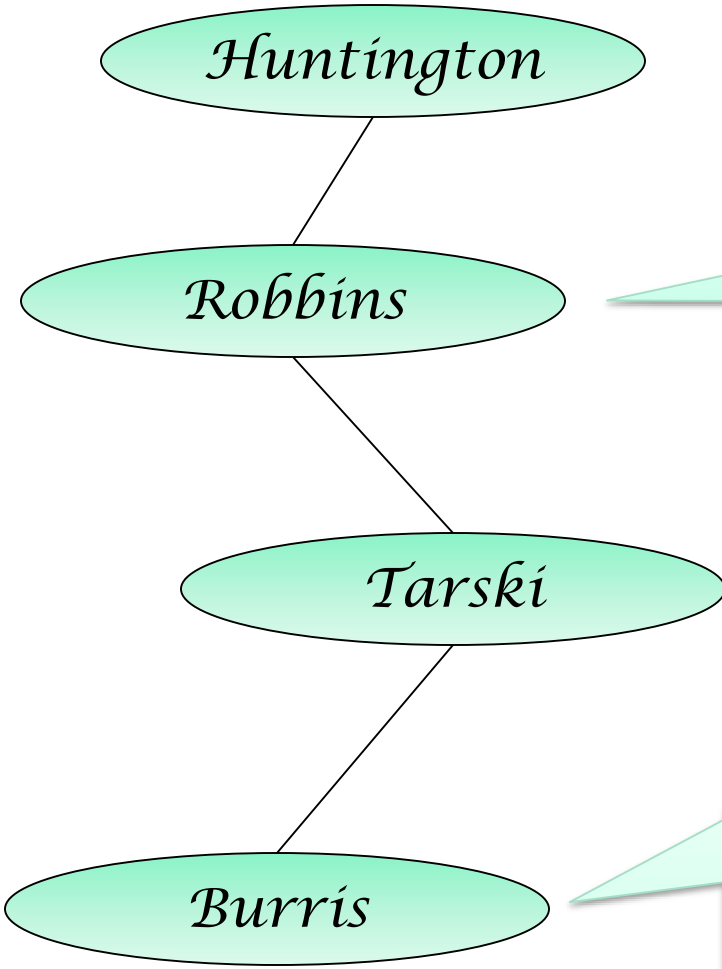
# History

- **Theorem Provers** have come up with novel mathematical results.
  - The most famous of these concerns Robbins Algebra.

- In 1933, **E. V. Huntington** presented the following basis for Boolean algebra:
  - $x + y = y + x$. [commutativity]
  - $(x + y) + z = x + (y + z)$. [associativity]
  - $n(n(x) + y) + n(n(x) + n(y)) = x$. [Huntington equation]

- Shortly thereafter, **Robbins** conjectured that the Huntington equation can be replaced with a simpler one:
  - $n(n(x + y) + n(x + n(y))) = x$. [Robbins equation]

# Gossip




Robbins said that he worked on the problem for some time, and then passed it on one of the century's most famous logicians, Dr. Albert Tarski of Stanford University.

Tarski, who is now dead, worked on the problem, included it in a book, and handed it out to graduate students and visitors.

Burris (University of Waterloo), for example, said that Tarski suggested the problem to him in the early 1970s, while he was visiting Stanford for a couple of months. Tarski, he said, "liked to throw out challenging problems to people passing through…"

# October 10, 1996

- While mathematicians were batting around Robbins's problem, computer scientists were striving to see if they could get computers to reason.

- On October 10, 1996, after **eight days of computation**, EQP (a version of OTTER Theorem Prover) found a proof to Robbins conjecture.

# Curiosity and the cat

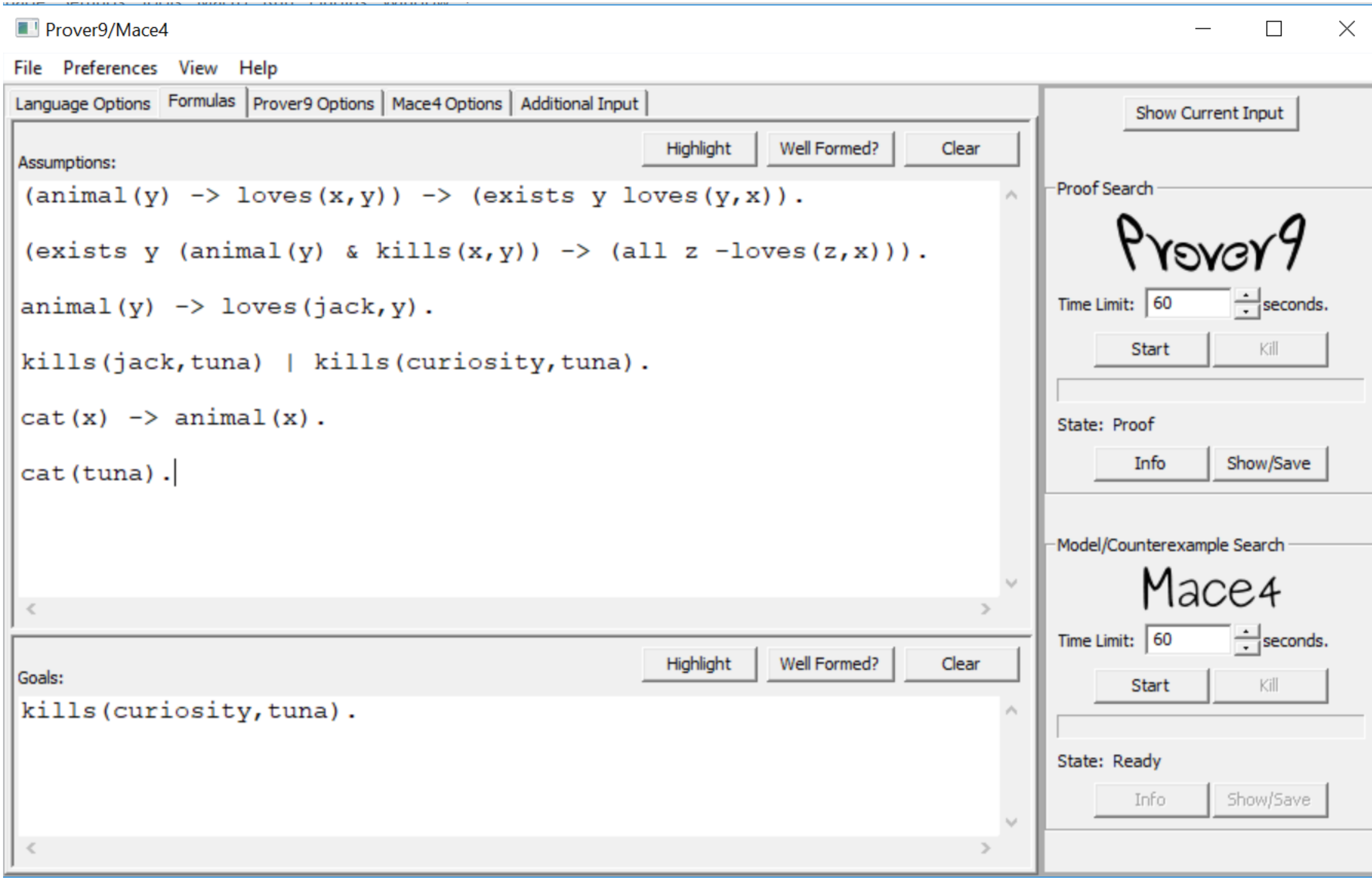
- Everyone who loves all animals is loved by someone.
- Anyone who kills an animal is loved by no one.
- Jack loves all animals.
- Either Jack or Curiosity killed the cat, who is named Tuna.
- Did Curiosity kill the cat?

# Curiosity and the cat in Prover 9

```
(animal(y) -> loves(x,y)) -> (exists y loves(y,x)).
(exists y (animal(y) & kills(x,y)) -> (all z -loves(z,x))).
animal(y) -> loves(jack,y).
kills(jack,tuna) | kills(curiosity,tuna).
cat(x) -> animal(x).
cat(tuna).
```

**Theorem:**

```
kills(curiosity,tuna).
```


Prover9/Mace4
File
Preferences
View
Help
Language Options
Formulas
Prover9 Options
Mace4 Options
Additional Input
Highlight
Well Formed?
Clear
Assumptions:
(animal(y) -> loves(x,y)) -> (exists y loves(y,x)).
(exists y (animal(y) & kills(x,y)) -> (all z -loves(z,x))).
animal(y) -> loves(jack,y).
kills(jack,tuna) | kills(curiosity,tuna).
cat(x) -> animal(x).
cat(tuna).
Highlight
Well Formed?
Clear
Goals:
kills(curiosity,tuna).
Show Current Input
Proof Search
Prover9
Time Limit:
60
seconds.
Start
Kill
State: Proof
Info
Show/Save
Model/Counterexample Search
Mace4
Time Limit:
60
seconds.
Start
Kill
State: Ready
Info
Show/Save

# Nowadays

- AMD, Intel and others use **automated theorem proving** to verify that **optimized binary operations** correctly produce the intended result.

Uncertain

# Representing Knowledge

# Uncertain knowledge

- Typical example: **Diagnosis**.

| Name | Toothache | … | Cavity |
|---|---|---|---|
| Smith | true | … | true |
| Mike | true | … | true |
| Mary | false | … | true |
| Quincy | true | … | false |
| … | … | … | … |

- Can we certainly derive the diagnostic rule:

  if Toothache=true then Cavity=true ?

**Isn't always right**: Not all patients with toothache have cavities; some have gum disease, abscess, etc.

- We could try turning the rule into a causal rule:
  - if Cavity=true then Toothache=true

**Isn't always right**: not all cavities cause pain.

# Belief and Probability

- The connection between **toothaches** and **cavities** is not a logical consequence in either direction.

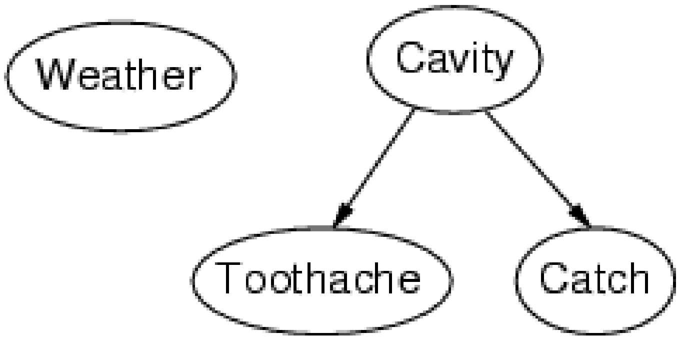
- However, we can provide a **degree of belief** on the rules. Our main tool for this is **probability theory**.

**E.g.** We believe there is, say, an 80% chance that the patient has cavity if he has a toothache.

We usually get this belief from **statistical data** or **experience**.

# Bayesian networks

A simple, **graphical** notation for conditional independence assertions.




Syntax:

- a set of nodes, one per variable
- a directed, acyclic graph (link means: "directly influences")
- conditional probability tables (CPTs) for each combination of parent values.

# Example (Judea Perls' original example)

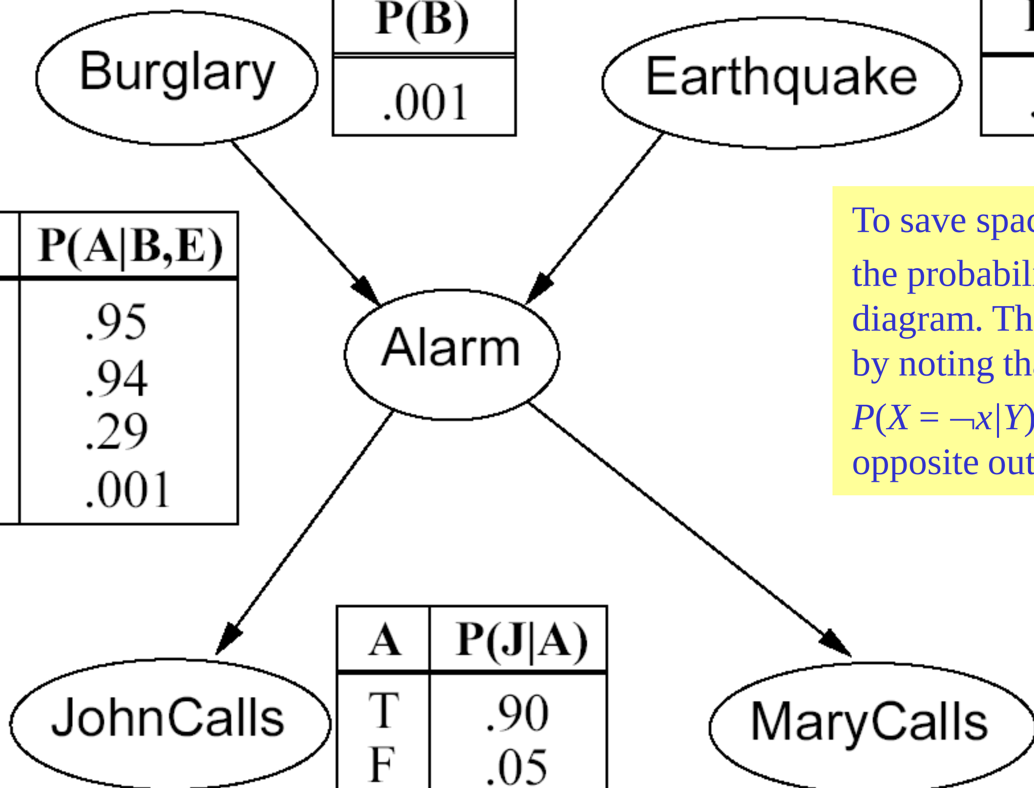
- I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?

- John always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm.
- Mary likes rather loud music and sometimes misses the alarm.

- Variables: *Burglary*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*

- Network topology reflects knowledge:
  - A burglar can set the alarm off
  - An earthquake can set the alarm off
  - The alarm can cause Mary to call
  - The alarm can cause John to call

# Example cont'd




| P(B) |
|---|
| .001 |

| P(E) |
|---|
| .002 |

| B | E | P(A\|B,E) |
|---|---|---|
| T | T | .95 |
| T | F | .94 |
| F | T | .29 |
| F | F | .001 |

| A | P(J\|A) |
|---|---|
| T | .90 |
| F | .05 |

| A | P(M\|A) |
|---|---|
| T | .70 |
| F | .01 |

To save space, some of the probabilities have been omitted from the diagram. The omitted probabilities can be recovered by noting that $P(X = \neg x) = 1 - P(X = x)$ and $P(X = \neg x|Y) = 1 - P(X = \neg x|Y)$, where $\neg x$ denotes the opposite outcome of $x$.

The topology shows that burglary and earthquakes directly affect the probability of alarm, but whether Mary or John call depends only on the alarm.

Thus our assumptions are that they don't perceive any burglaries directly, and they don't confer before calling.

# Inference

Example: *P(burglary | johncalls, marycalls)*? (Abbr. $P(b|j,m)$)

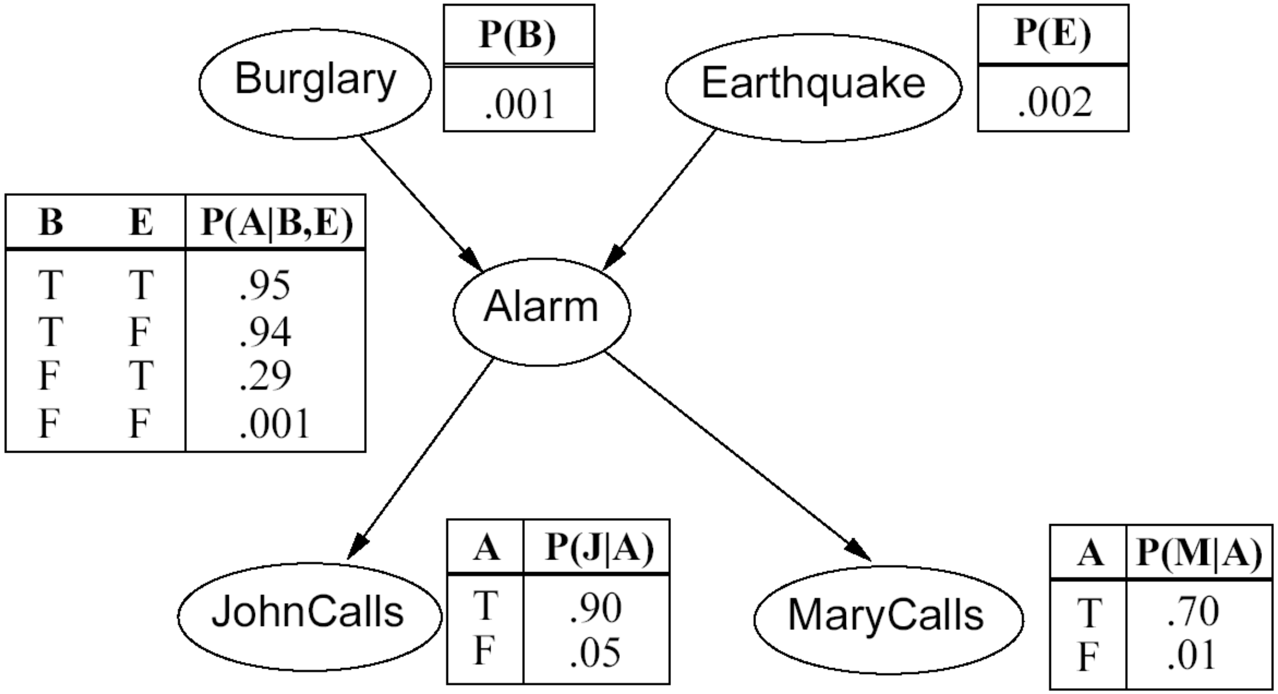
$$P(b \mid j, m)$$
$$= \alpha P(b, j, m)$$
$$= \alpha \sum_a \sum_e P(b, j, m, a, e)$$
$$= \alpha \big( P(b, j, m, a, e) + P(b, j, m, \neg a, e) + P(b, j, m, a, \neg e) + P(b, j, m, \neg a, \neg e) \big)$$

Computed as:
$P(b)P(e)P(j|a)P(m|a)P(a|b,e)P(e)$

Computed as:
$P(b)P(e)P(j|\neg a)P(m|\neg a)P(\neg a|b,e)P(e)$

Probabilities are taken from the tables.

# Numerically…




| P(B) |
|---|
| .001 |

| P(E) |
|---|
| .002 |

| B | E | P(A\|B,E) |
|---|---|---|
| T | T | .95 |
| T | F | .94 |
| F | T | .29 |
| F | F | .001 |

| A | P(J\|A) |
|---|---|
| T | .90 |
| F | .05 |

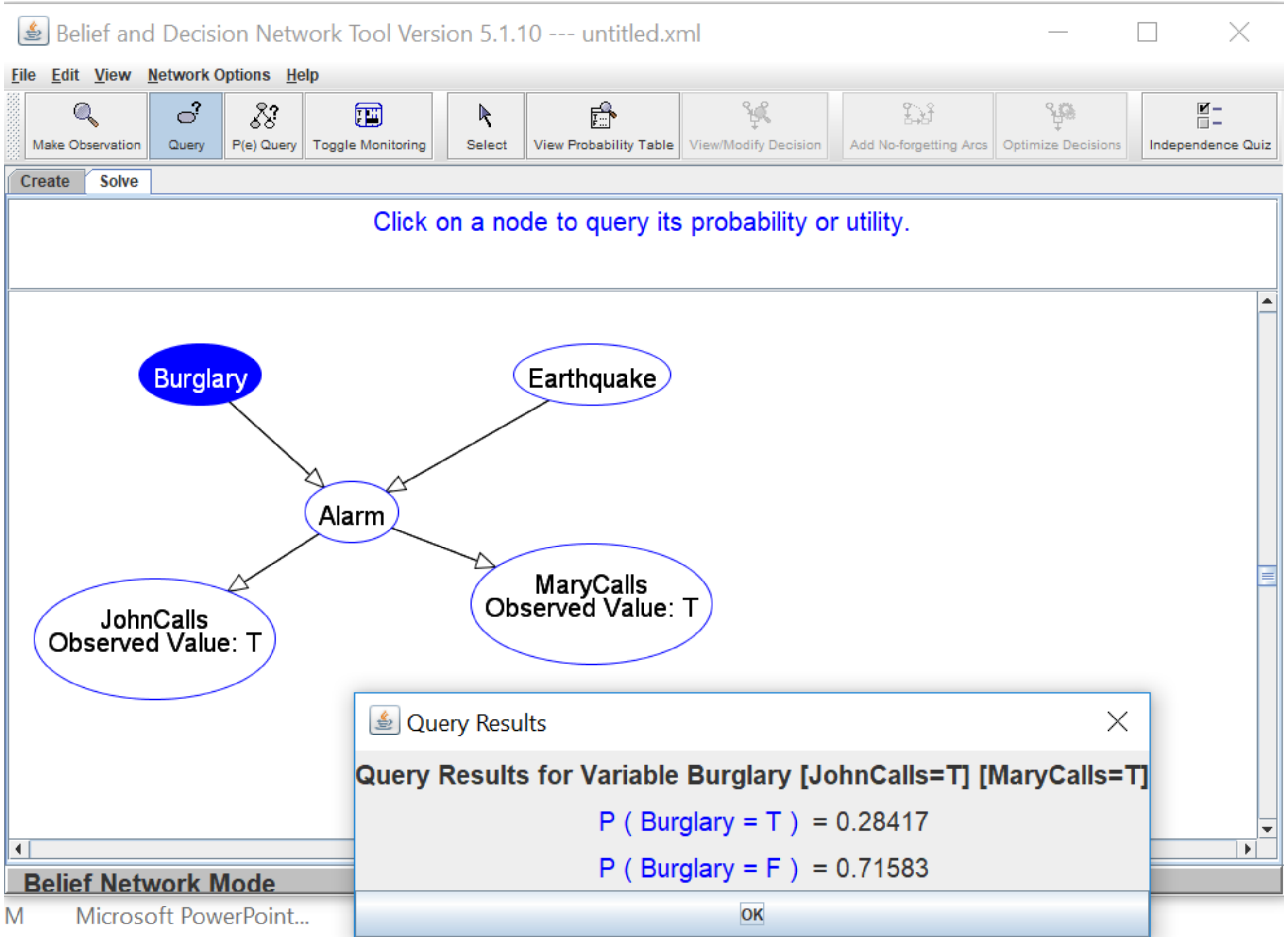
| A | P(M\|A) |
|---|---|
| T | .70 |
| F | .01 |

$P(b \mid j,m) = \ldots = \alpha * 0.00059$

$P(\neg b \mid j,m) = \ldots = \alpha * 0.0015$

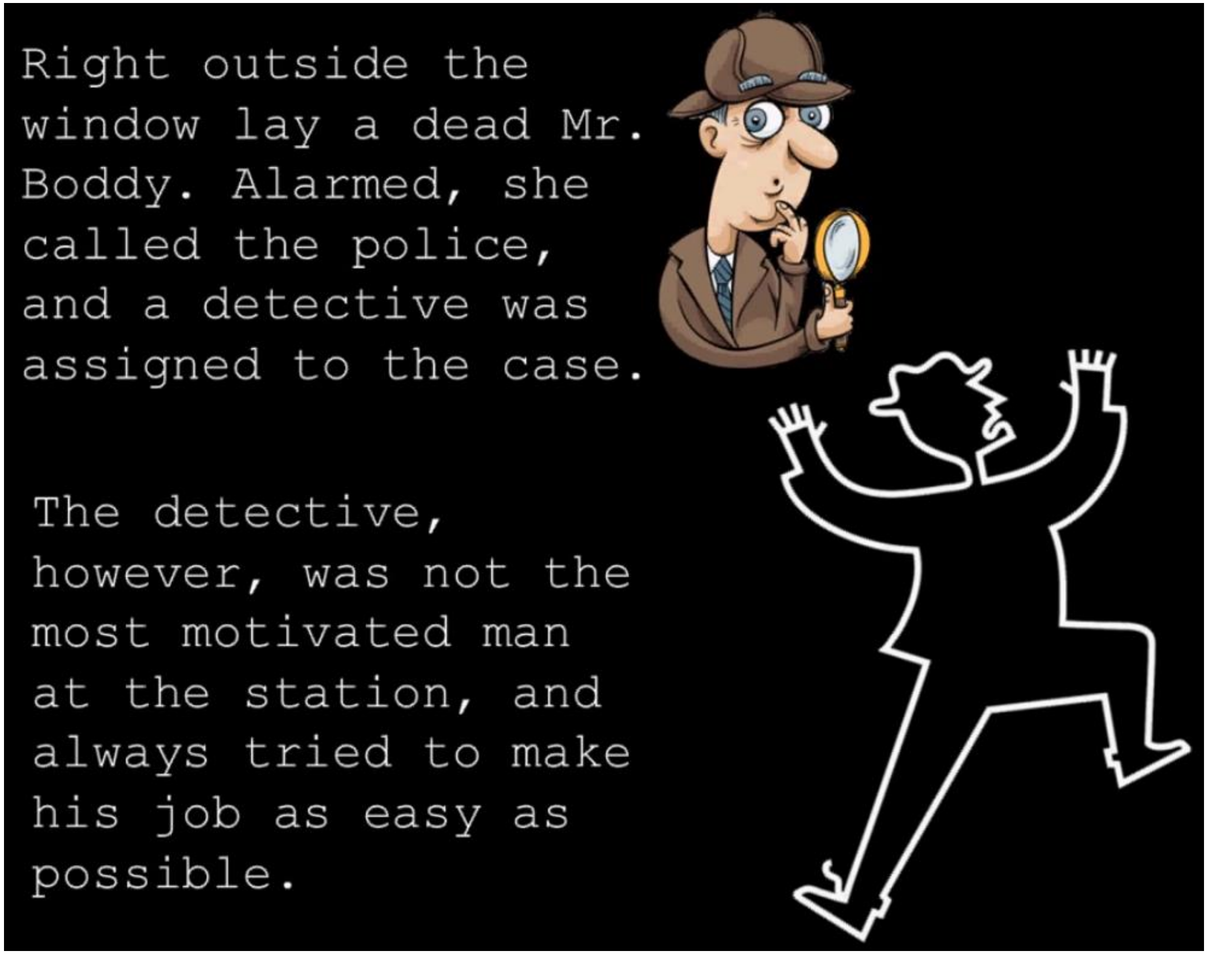
$\mathbf{P}(\mathrm{B} \mid j,m) = \alpha <0.00059, 0.0015> = \mathbf{<0.28, 0.72>}$.

# Aispace Demo



# Nice Bayesian Network Example

https://www.youtube.com/watch?v=4fcqyzVJwHM



# Learning

# Learning

It's not always easy to instruct the machine what to do.

Often it's best to let the machine "**learn**" how the world works.

We can think of at least three different problems being involved in learning:

- memory
- averaging
- generalization

# Example problem

(Adapted from Leslie Kaelbling's example in the MIT courseware)

- Imagine I'm trying predict whether my neighbor is going to drive into work.

- Whether she drives into work seems to depend on the following attributes of the day:
  - temperature
  - expected precipitation
  - day of the week
  - what she's wearing

# Memory

- Okay. Let's say we observe our neighbor on three days:

| **Temp** | **Precip** | **Day** | **Clothes** | |
|---|---|---|---|---|
| 25 | None | Sat | Casual | **Walk** |
| -5 | Snow | Mon | Casual | **Drive** |
| 15 | Snow | Mon | Casual | **Walk** |

# Memory

- Now, we find ourselves on a snowy “-5” degree Monday, and the neighbor is wearing casual clothes.
- Do you think she's going to drive?

| **Temp** | **Precip** | **Day** | **Clothes** | |
|---|---|---|---|---|
| 25 | None | Sat | Casual | **Walk** |
| -5 | Snow | Mon | Casual | **Drive** |
| 15 | Snow | Mon | Casual | **Walk** |
| -5 | Snow | Mon | Casual | |

# Memory

- Standard answer in this case is "yes".
  - This day is just like one of the ones we've seen before, and so it seems like a good bet to predict "yes."

- This is the most rudimentary form of learning, which is just to memorize the things you've seen before.

| **Temp** | **Precip** | **Day** | **Clothes** | |
|---|---|---|---|---|
| 25 | None | Sat | Casual | **Walk** |
| -5 | Snow | Mon | Casual | **Drive** |
| 15 | Snow | Mon | Casual | **Walk** |
| -5 | Snow | Mon | Casual | **Drive** |

# Noisy Data

- Things aren't always as easy as they were in the previous case. What if you get this set of noisy data?

| Temp | Precip | Day | Clothes | |
|---|---|---|---|---|
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Drive** |
| 25 | None | Sat | Casual | **Drive** |
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **?** |

- We have certainly seen this case before, but the problem is that it has had different answers. Our neighbor is not entirely reliable.

# Averaging

- One strategy would be to predict the majority outcome.

| **Temp** | **Precip** | **Day** | **Clothes** | |
|---|---|---|---|---|
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Drive** |
| 25 | None | Sat | Casual | **Drive** |
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Walk** |
| 25 | None | Sat | Casual | **Walk** |

# Generalization

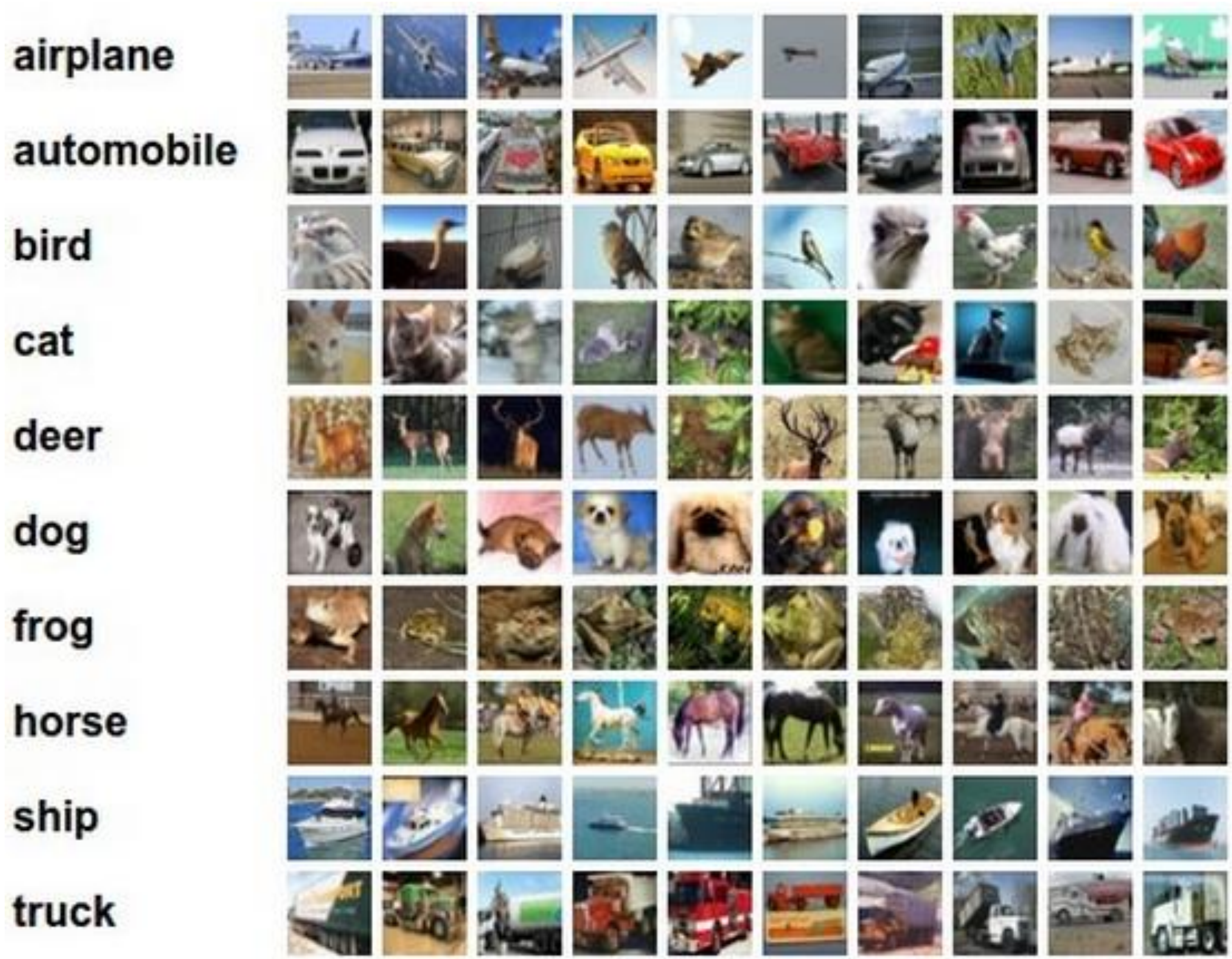
- Dealing with previously unseen cases
- Will she walk or drive?

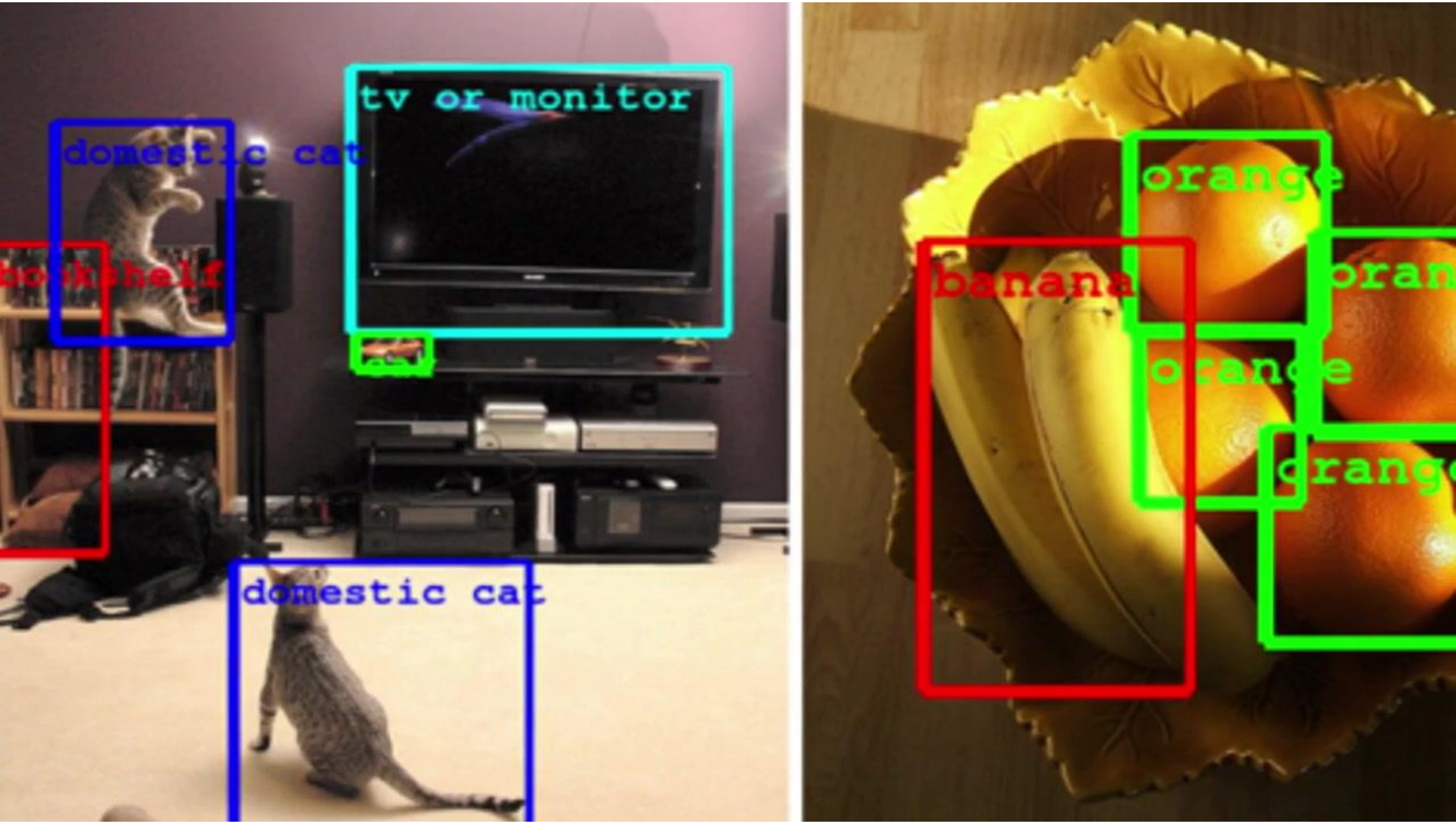
| Temp | Precip | Day | Clothes | |
|---|---|---|---|---|
| 22 | None | Fri | Casual | **Walk** |
| 3 | None | Sun | Casual | **Walk** |
| 10 | Rain | Wed | Casual | **Walk** |
| 30 | None | Mon | Casual | **Drive** |
| 20 | None | Sat | Formal | **Drive** |
| 25 | None | Sat | Casual | **Drive** |
| -5 | Snow | Mon | Casual | **Drive** |
| 27 | None | Tue | Casual | **Drive** |
| 24 | Rain | Mon | Casual | **?** |

We might plausibly make any of the following arguments:

- She's going to walk because it's raining today and the only other time it rained, she walked.
- She's going to drive because she has always driven on Mondays…

# Image Recognition




tv or monitor
domestic cat
bookshelf
domestic cat
orange
banana
orange

# Taste of neural networks for OCR

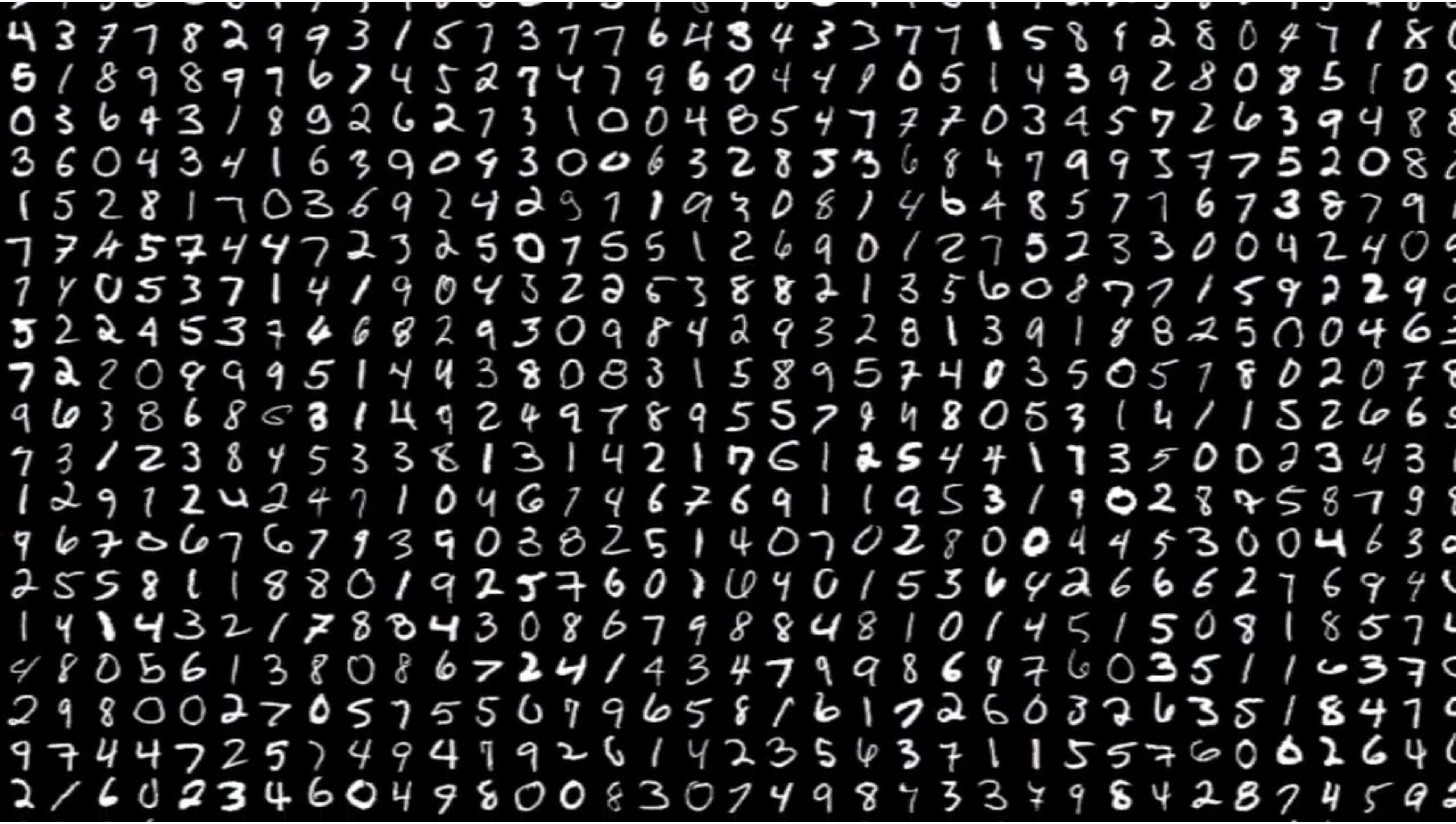
For informal intro see:

https://www.youtube.com/watch?v=ARODjRbGbSg

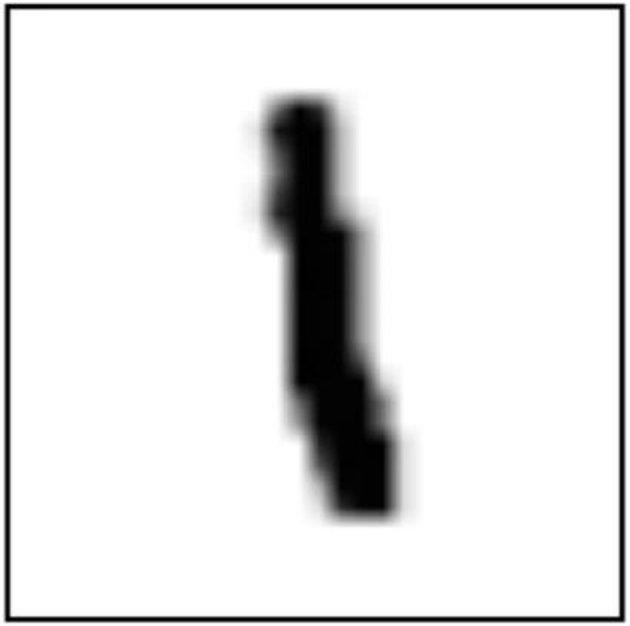
The next slides are from this video.

# Few handwritten digits

# 70,000 of them



# One of them



$$\simeq \begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&.6&.8&0&0&0&0&0&0\\
0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\
0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\
0&0&0&0&0&0&.5&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.7&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&0&0&0&0&0&.9&1&.1&0&0&0&0\\
0&0&0&0&0&0&0&.3&1&.1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}$$

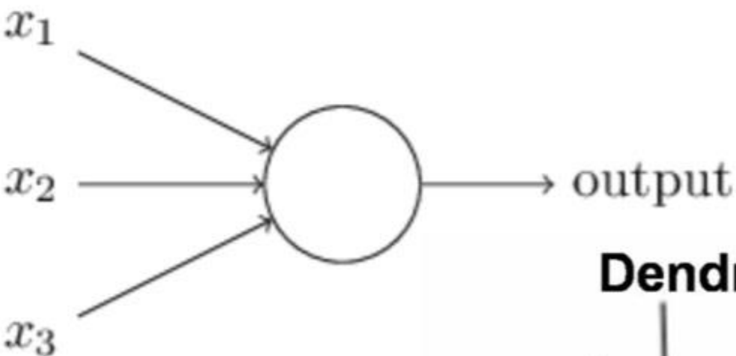
Not drawn to scale

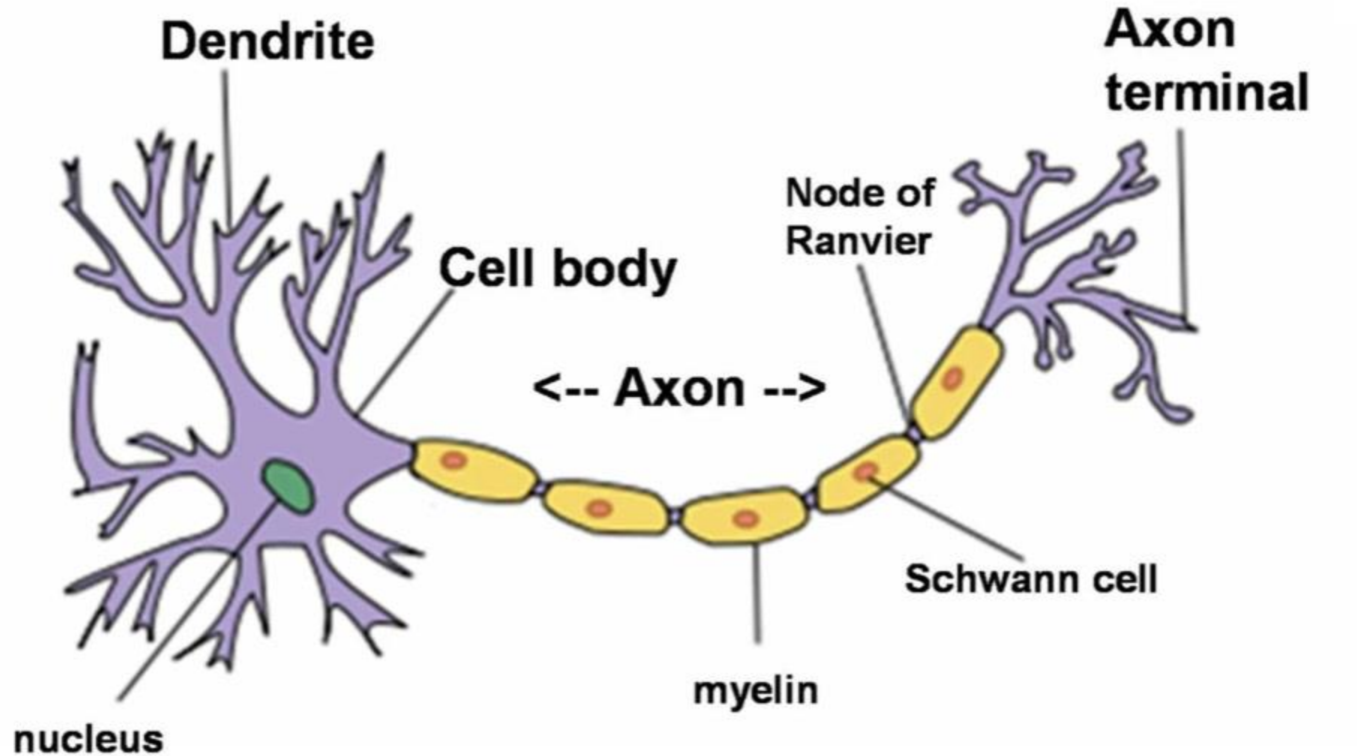
# The “classes”

0 1 2 3 4

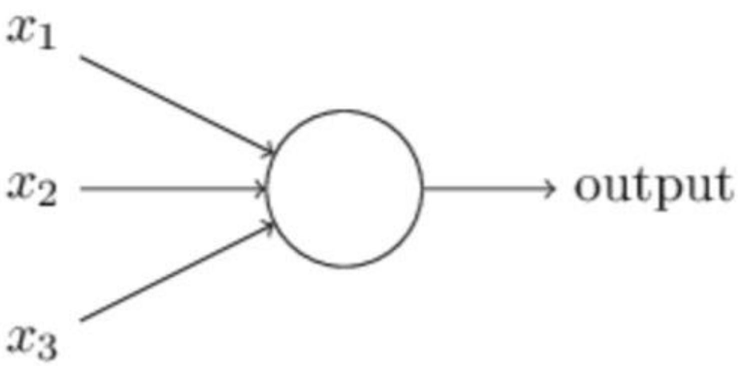
5 6 7 8 9

# An artificial neuron and …
# a natural one



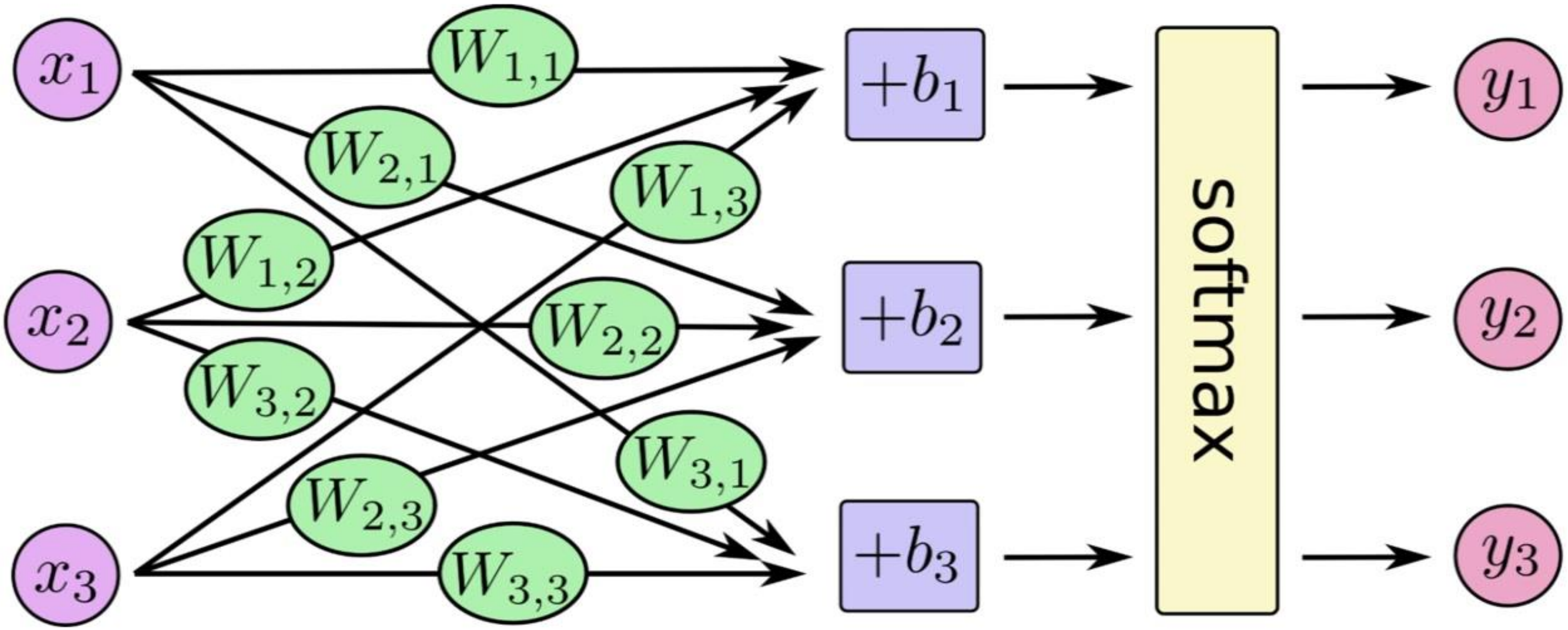



# Mathematically speaking




$$\sum x_i W_i + b$$

# A simple neural network



# Inside a “class” neuron

$$evidence_c = \sum_{i=1}^{784} x_i W_{c,i} + b_c$$

# Output vector

<0.01,0.91,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01>

# Output class –

## the one corresponding to the highest value

<0.01,0.91,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01>

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

Are these probabilities?

# Obtaining probabilities –

## normalizing the output vector

$$normevidence_c = \frac{evidence_c}{\sum_{c=0}^{9} evidence_c}$$

# A better normalization –

## through amplification first

$$normevidence_c = \frac{e^{evidence_c}}{\sum_{c=0}^{9} e^{evidence_c}}$$


Calculating the Weights

# Compare probability distributions (vectors)

**<0,1,0,0,0,0,0,0,0,0>**

**<0.01,0.91,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01>**

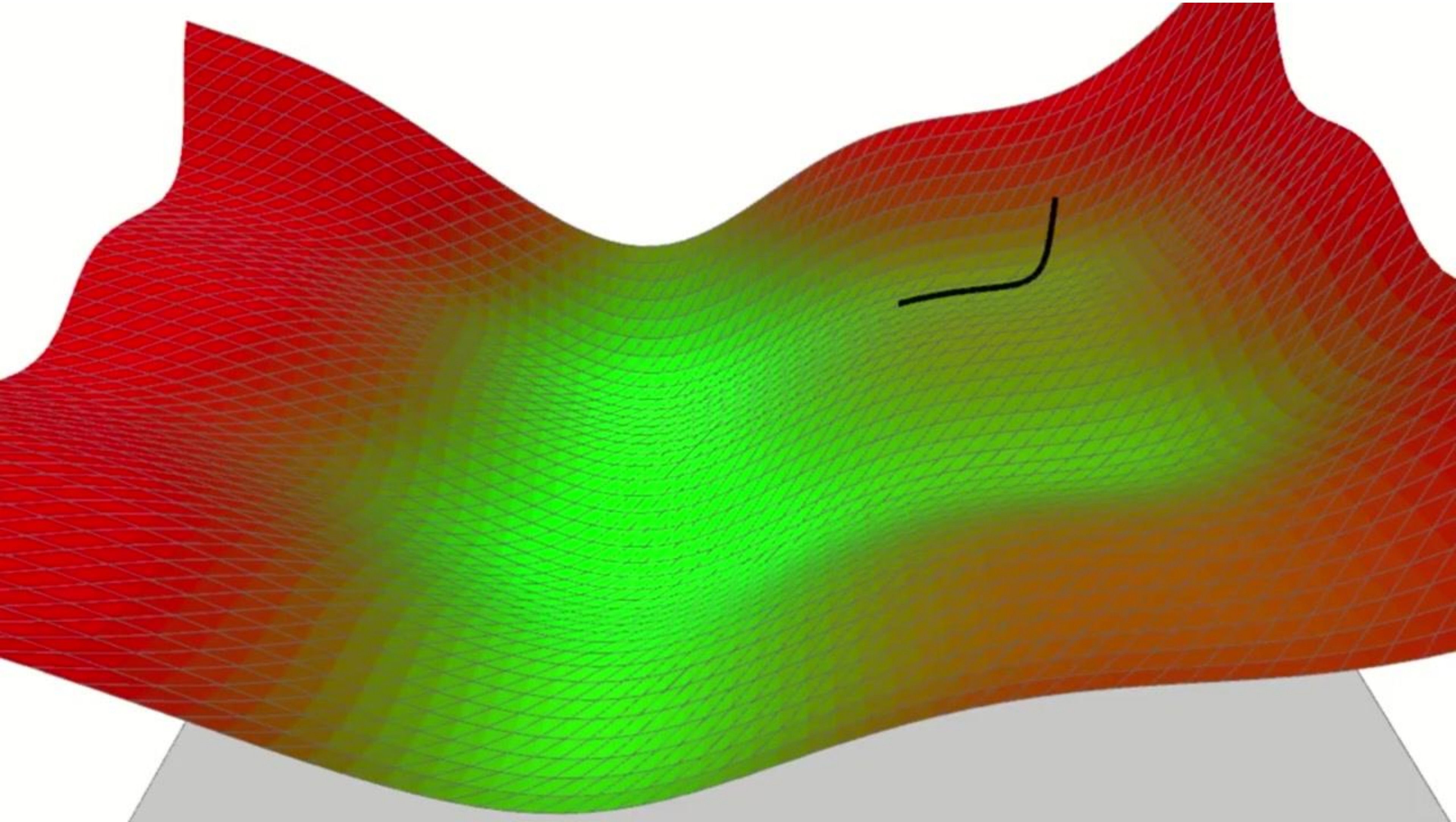
# The Loss function

$$-\log(\text{probability})$$

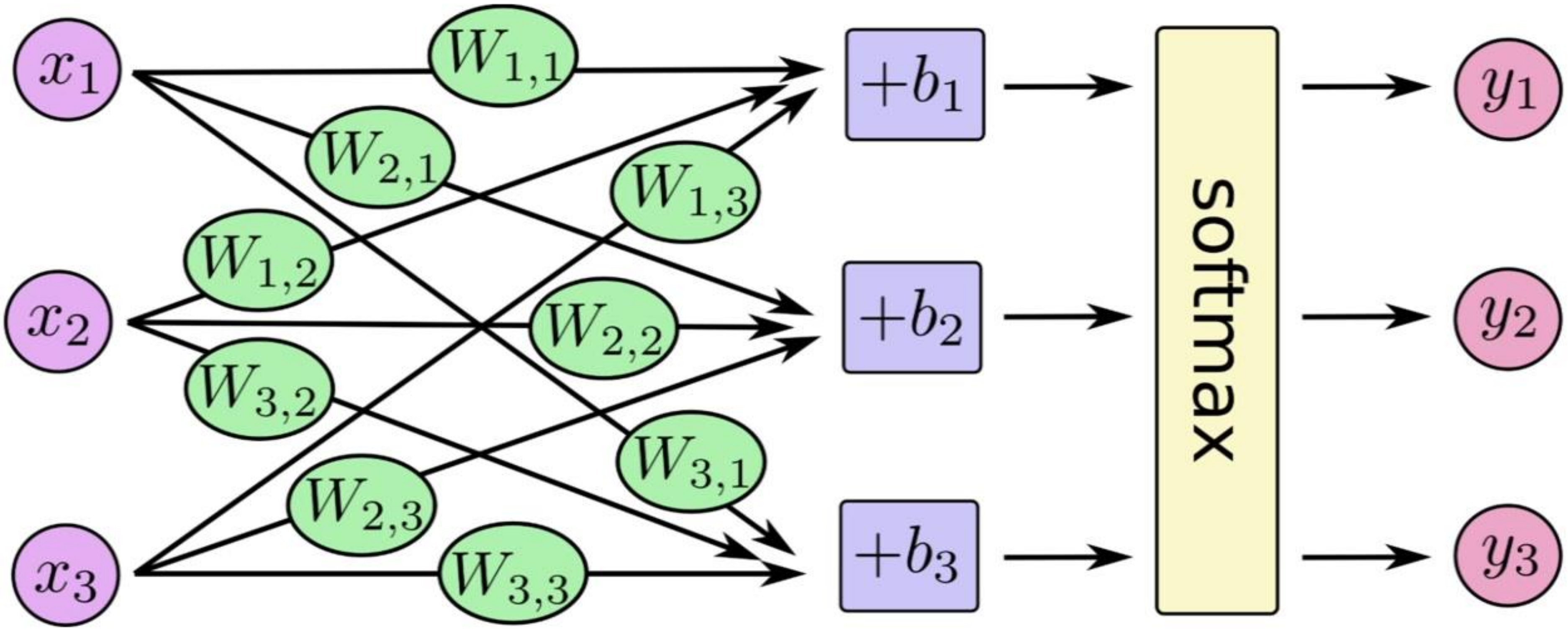
**<0,1,0,0,0,0,0,0,0,0>**

**<0.01,0.91,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01>**

# Navigating the Loss surface to reach the minimum

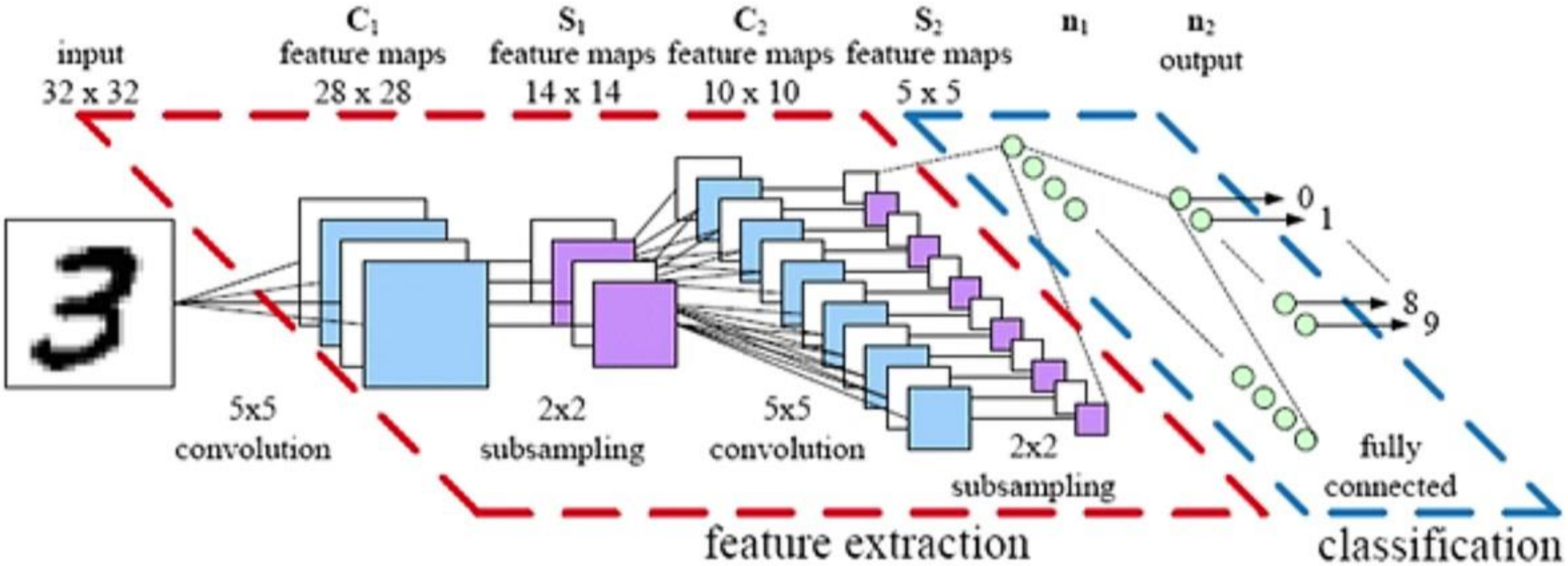


# Accuracy of the simple NN



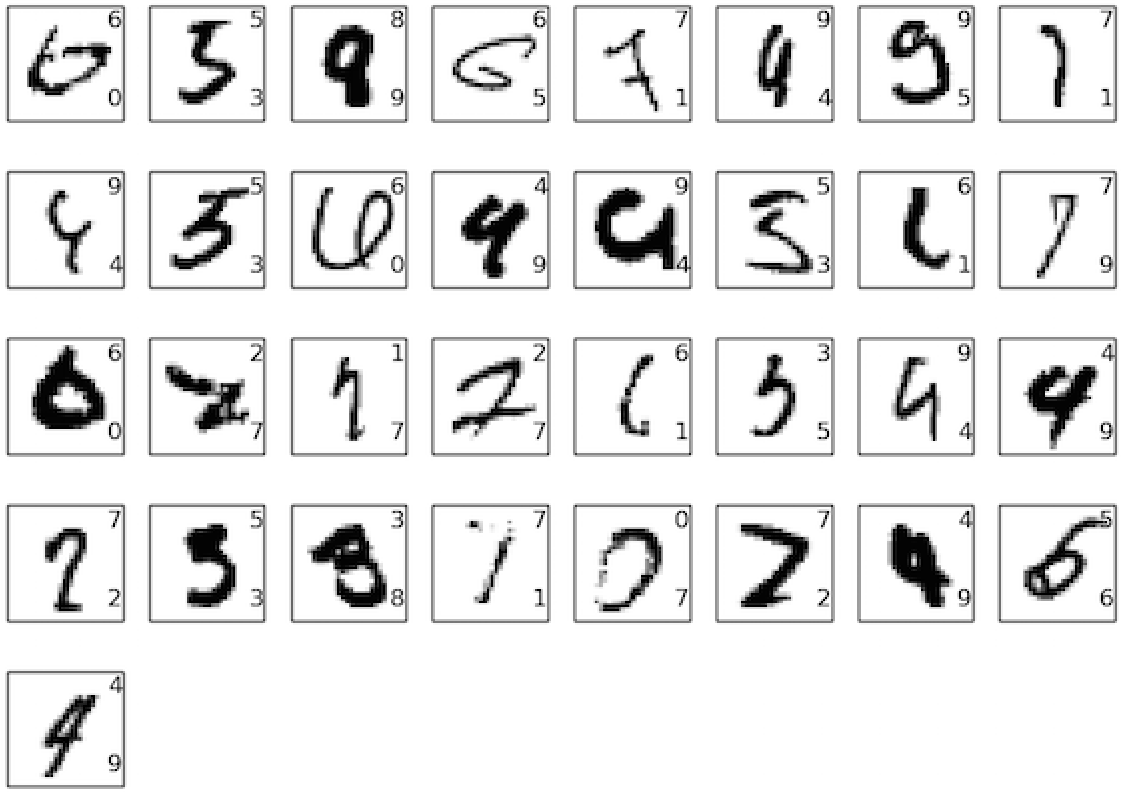

**95%**

# More complicated (convoluted) NN



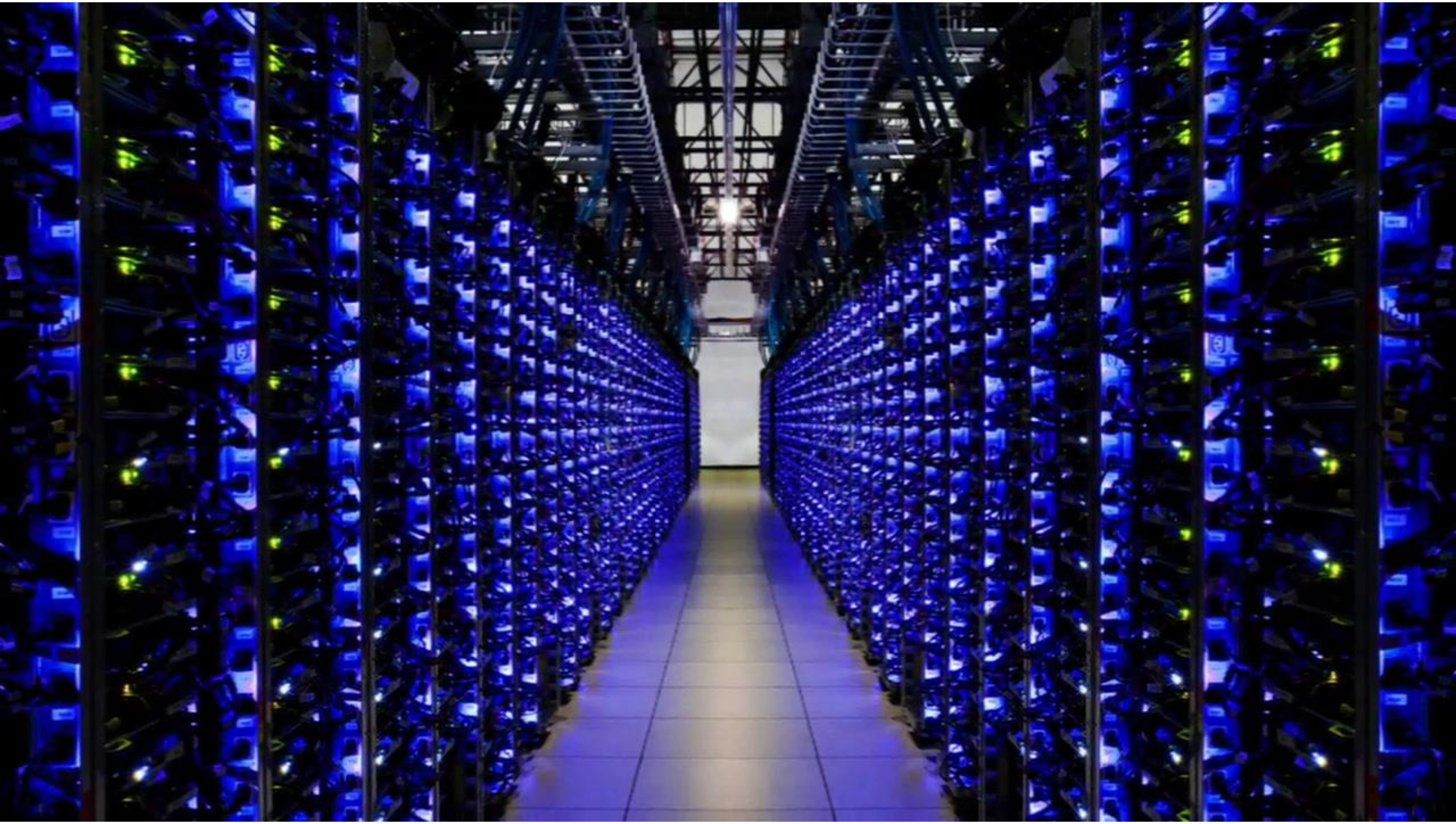

**99% -- same as humans**

# Some cases that it gets wrong



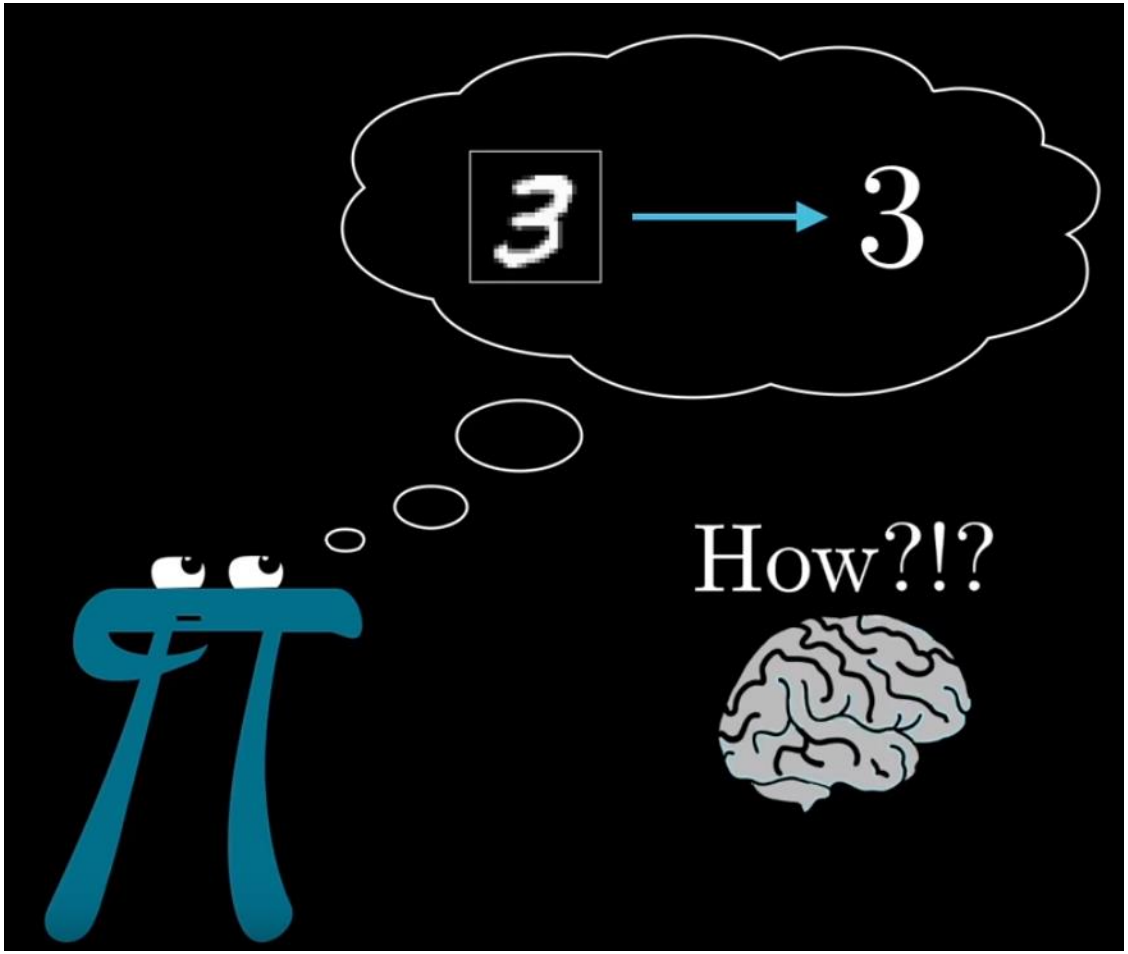

correct classification is in the top right;
program's classification is in the bottom right

# Lots of computational power needed



# Another informal intro to NN

https://www.youtube.com/watch?v=aircAruvnKk



# Success Stories of AI

- Play professionally chess and many other games.
- Discover and prove theorems
- Drive a car
- Do image recognition (with accuracy often better than humans)
- Do speech recognition
- Do automatic translation

# That's all so far -- questions?