Data Science

Artificial Intelligence

Tinglong Dai, PhD

Bernard T. Ferrari Professor

Johns Hopkins Carey Business School

“Experience is what you get when you *didn’t* get what you wanted.”

Randy Pausch, computer scientist/artist

https://bit.ly/cmulastlecture

“Technology is neither good, nor bad; nor is it neutral.”

Melvin Kranzberg, historian (1917–1995)

Agenda

1

Week 1: Foundations

2

Week 2: Tools & Techniques

Machine learning, deep learning, Keras, Python, and no-code tools.

3

Week 3: CNNs & Business Models

Convolutional neural networks, using AI to create human value. Assignment #1 due.

4

Week 4: MLOps, RNNs, & Transformers

MLOps, RNNs, transformers, evaluation of AI. AI Lab proposal due.

5

Week 5: LLMs & GenAI

Large language models, generative AI, using AI to create business value. Assignment #2 due.

6

Week 6: Human-AI Interaction

Human-AI interaction, reinforcement learning, system design.

7

Week 7: Responsible AI

Responsible AI, future of AI. Lab final deliverable due.

8

Week 8: Individual Final Exam; Peer Reviews of AI Lab

Second TA Tutorial: Keras & CNN

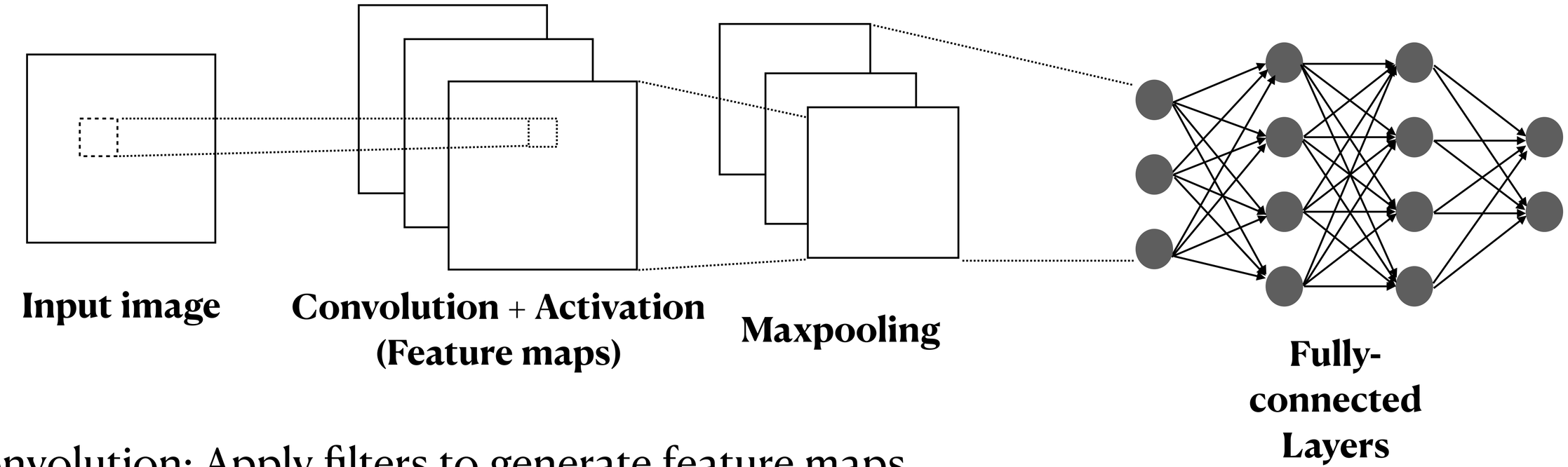
🗓️ **Friday, 11/14, 12:00–1:00 PM**

📍 **Join via Zoom:** https://bit.ly/jhuaita25

Attendance is optional. Materials will be posted on Canvas

Review of Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs)



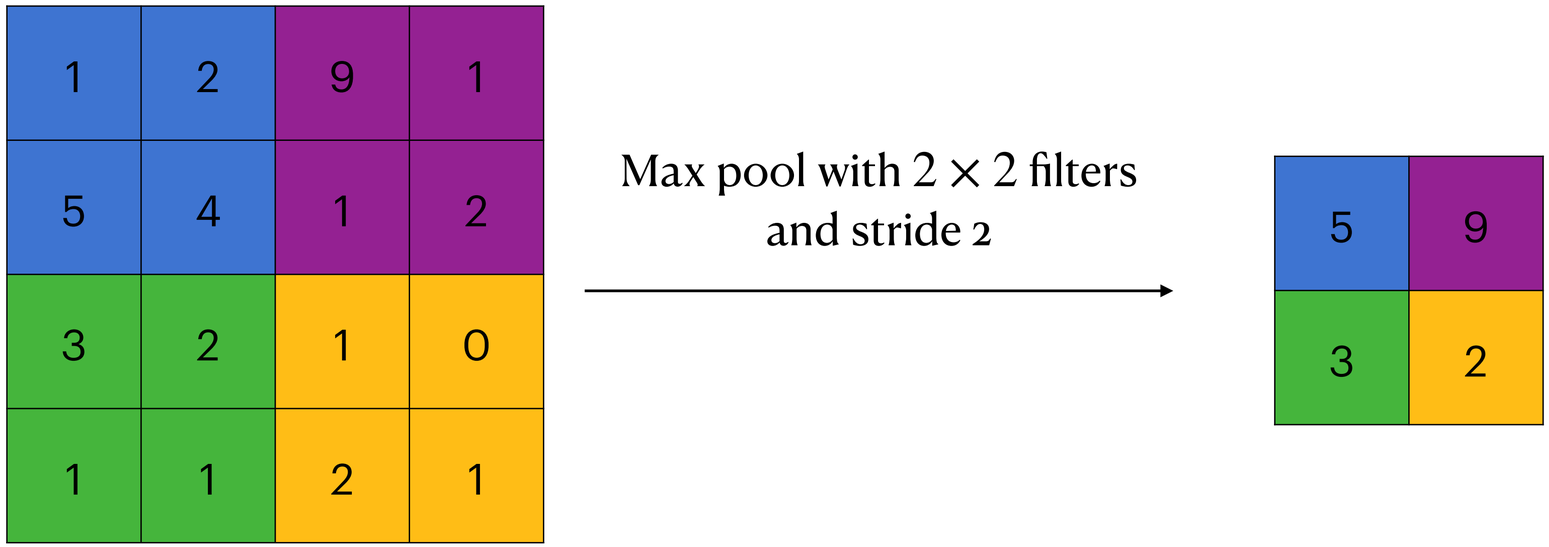
- Convolution: Apply filters to generate feature maps
- Activation: Usually ReLU
- Maxpooling: Downsampling operation on each feature map
- Fully connected layers: Classification

MaxPooling

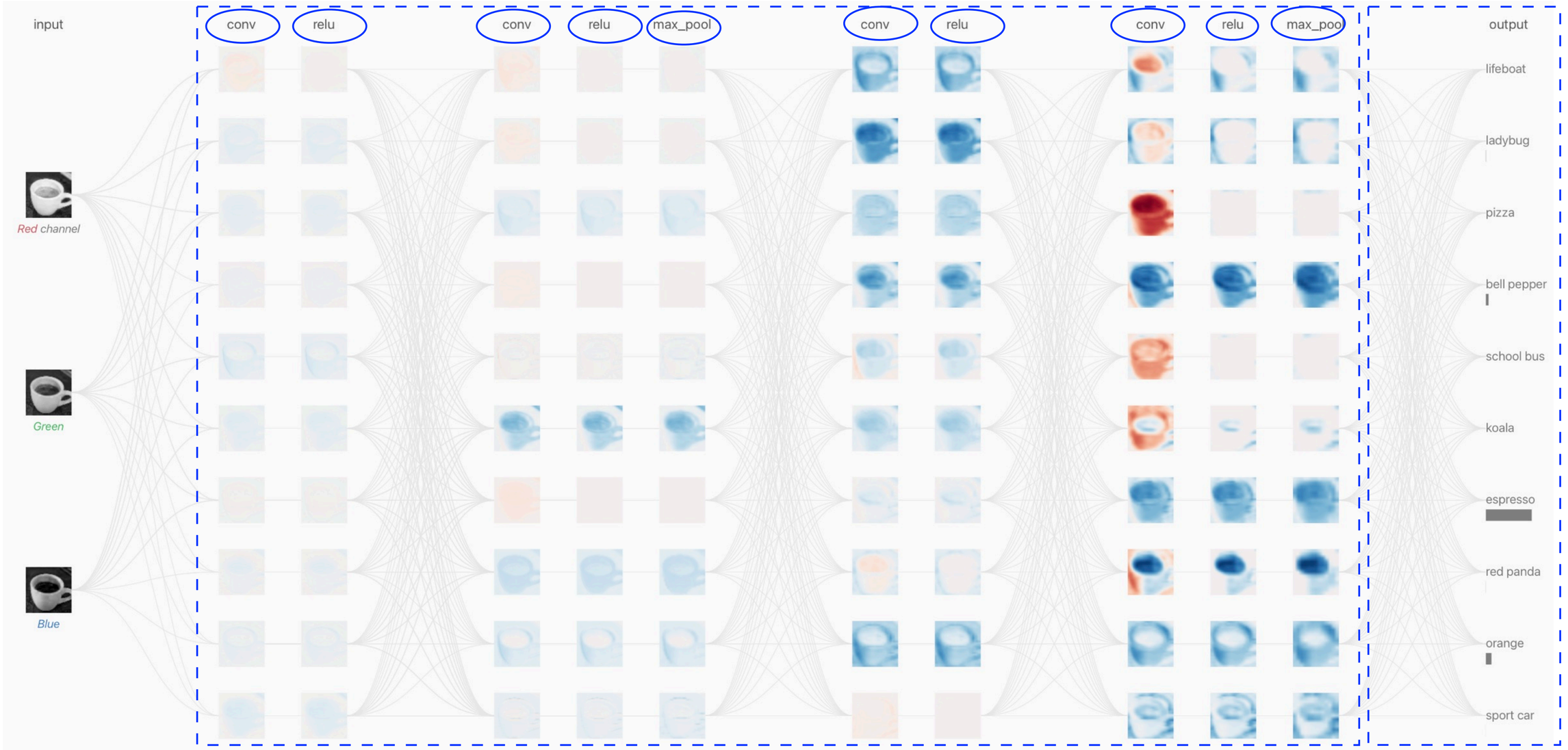
- Purpose: to aggressively “downsample” feature maps (i.e., make the representations smaller and more manageable)

- How it works: Extract windows from the input feature map, and pick the maximum value of each window

- Example:



A Fully Connected CNN https://bit.ly/cnnsimu



Feature Learning **Classification**

CNNs in Keras

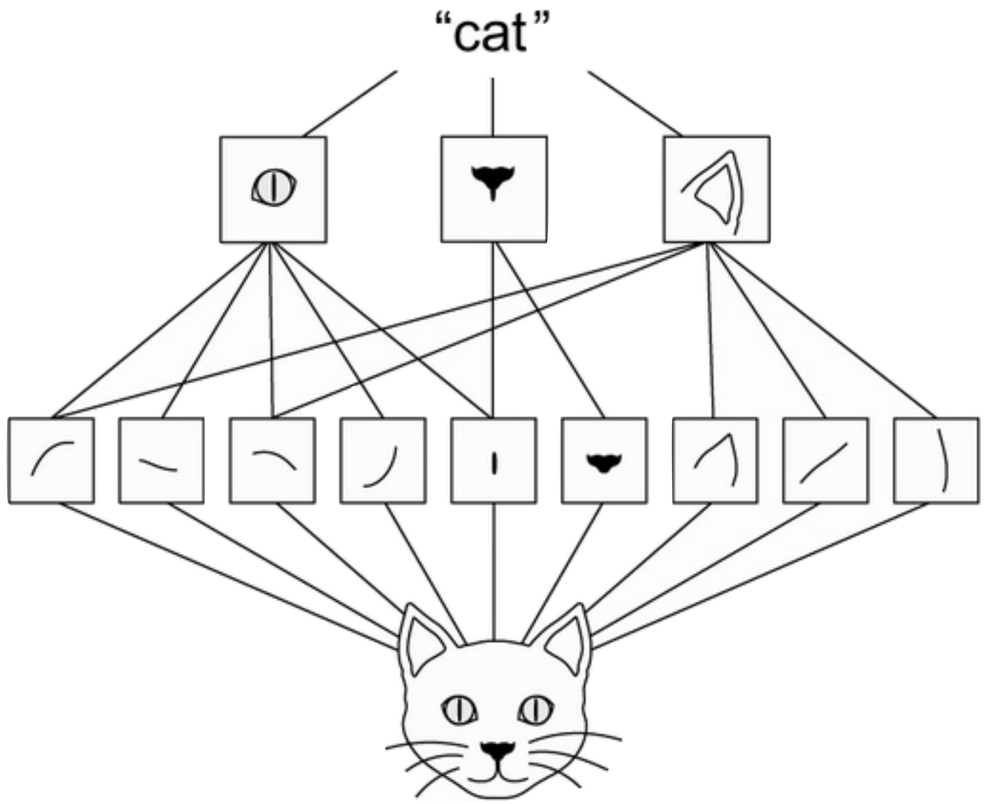
```
from tensorflow import keras
from tensorflow.keras import layers
inputs = keras.Input(shape=(28, 28, 1))
x = layers.Conv2D(filters=32, kernel_size=3, activation="relu")(inputs)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=64, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=128, kernel_size=3, activation="relu")(x)
x = layers.Flatten()(x)
outputs = layers.Dense(10, activation="softmax")(x)
model = keras.Model(inputs=inputs, outputs=outputs)
```

Number of filters

Dimension of the filter (aka kernel) is 3×3

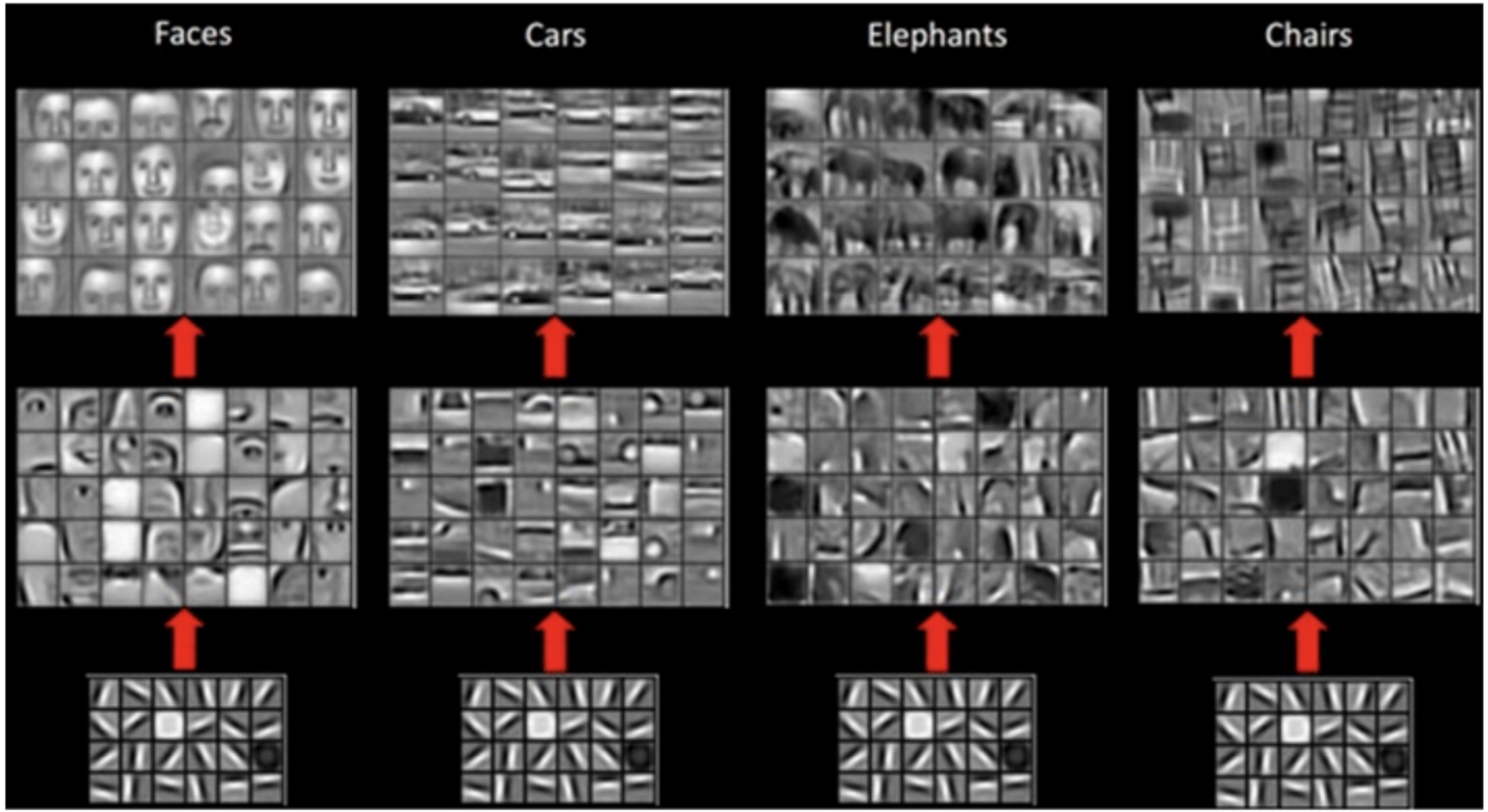
Maxpooling operation with 2×2 filters and stride 2

Why Do We Need So Many Convolution Layers?



- To learn spatial hierarchies of patterns
  - A first convolution layer learns small local patterns such as edges
  - A second convolutional layer learns larger patterns made of the features of the first layers
  - More layers allow larger patterns to be captured
  - This allows a CNN to efficiently learn increasingly complex and abstract visual concepts, because **the visual world is fundamentally spatially hierarchical**

Why Do We Need So Many Convolution Layers?



https://bit.ly/cnnlayers

MLOps = DevOps for AI

MLOps = DevOps for AI

- **DevOps** (Development + Operations) is a software development approach that brings development and operations together
  - Why? If we develop software without accounting for business operations, we risk developing something that doesn't deliver user needs and it's too late to make corrections
  - Avoiding risks and surprises

- The essence of DevOps is continuous deployment, which allows the team to *continuously* deliver value from development to production

- MLOps = DevOps for AI projects

Let’s Start with DevOps

Before Putting DevOps into the Context of AI

- Software development success rates over the years, according to the Chaos reports (1996—2020)

	1996	2000	2004	2009	2020
Success	27%	28%	29%	32%	35%
Partial success	40%	23%	53%	44%	46%
Failure	33%	49%	18%	24%	19%

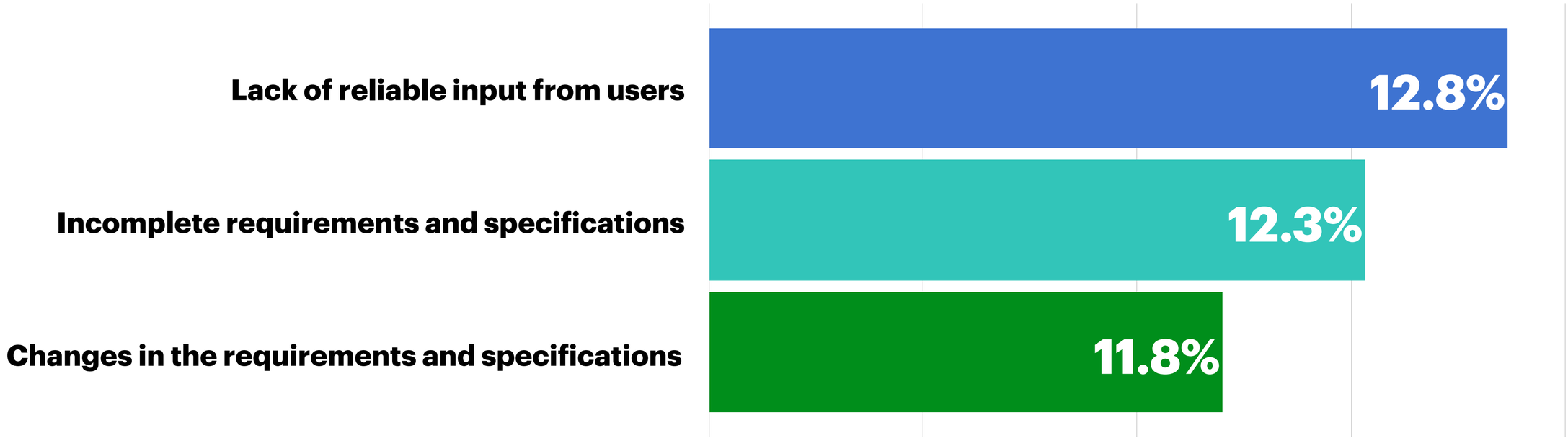
- Success = time and budget constraints are met and required features and functions are implemented

“Most projects take 3 times as long as planned, thereby costing ~ 2.8 times more and bringing 70 to 80% of the planned functionality.

The contract and the negotiating skills decide who carries the additional 180% of the costs.”

Walter Jaburek

Top Three Causes for Project Failures



Source: Standish Group 2009

To Phrase it Differently

- Lack of reliable input from users (12.8%)

  = Lack of cooperation

- Incomplete requirements and specifications (12.3%)

  = Missing knowledge of the requirements
  (customers don’t know what they *really* wants)

- Changes in the requirements and specifications (11.8%)

  = What is partially unknown is usually also described incorrectly or incompletely

Software Professionals' Perspective

- There is not enough user input
- There is no simple, clear vision that describes a project's purpose
- There is very little teamwork
- Projects are becoming increasingly complex
- Systems have become more widely distributed
- Functional and transparent monitoring of progress is often impossible
- Experts on all sides (supplier, consultants, and customers) find it increasingly difficult to predict potential problems

BUSINESS INSIDER

HOME > HEALTHCARE

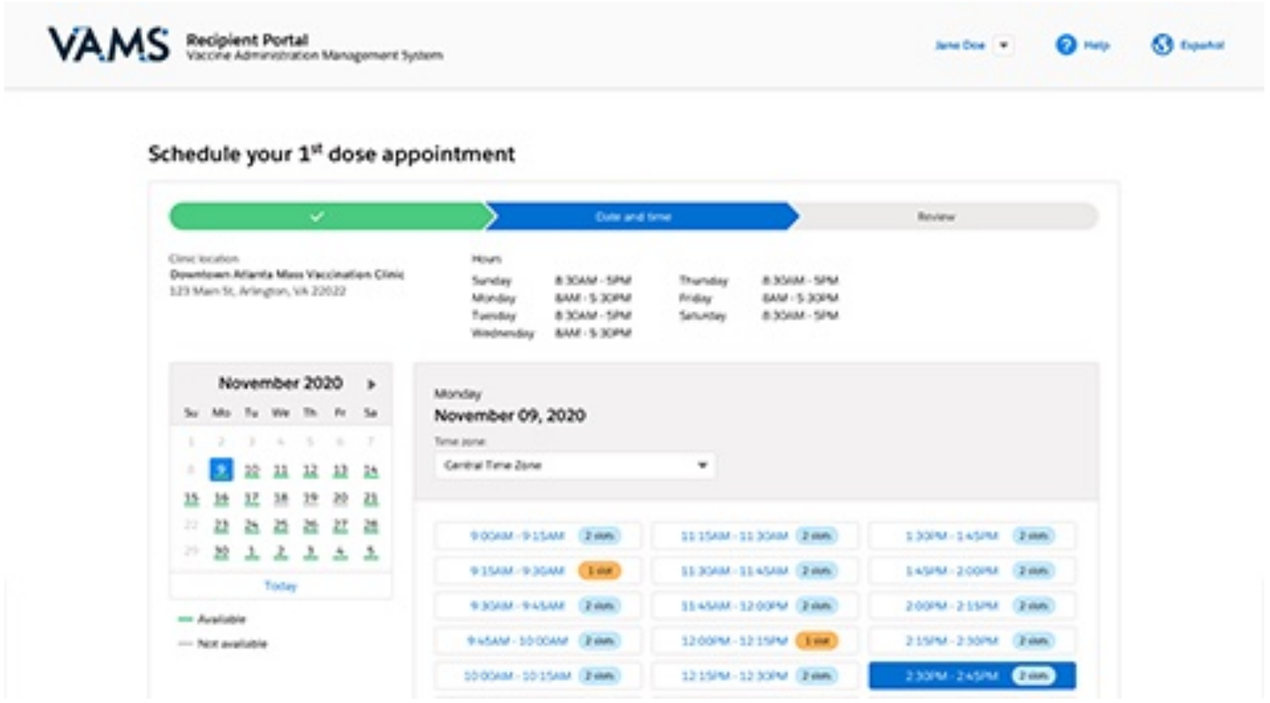
The US government's $44 million COVID vaccine rollout website isn't working — here's how it should be fixed

Tinglong Dai, Feb 8, 2021

The federal government had envisioned states using one national vaccine scheduling system, and it offered a contractor $44 million to develop it. But that system turned out to be so poorly designed that all but nine states opted out before even trying to adopt it, even though it was being offered by the government for free.

The few states that do use the Vaccine Administration Management System, or VAMS, have reported random appointment cancellations and unreliable registrations. Some vaccinators have had to resort to creating records on paper because of system glitches, slowing down the pace of getting shots into people's arms.

As troubled as the VAMS website may be, it is also a predictable result. We've seen this movie before.



https://bit.ly/vamsdev

BUSINESS INSIDER

HOME > HEALTHCARE

The US government's $44 million COVID vaccine rollout website isn't working — here's how it should be fixed

Tinglong Dai, Feb 8, 2021

HealthCare.gov, the federal healthcare exchange website that was launched to implement the Affordable Care Act, also known as Obamacare, cost taxpayers nearly $1 billion. When HealthCare.gov was launched on Oct. 1, 2013, only six people were able to sign up for health care on the first day. The Obama administration ended up having to enlist a team of engineers from Google, Amazon and Facebook to fix it.

https://bit.ly/vamsdev

BUSINESS INSIDER

HOME > HEALTHCARE

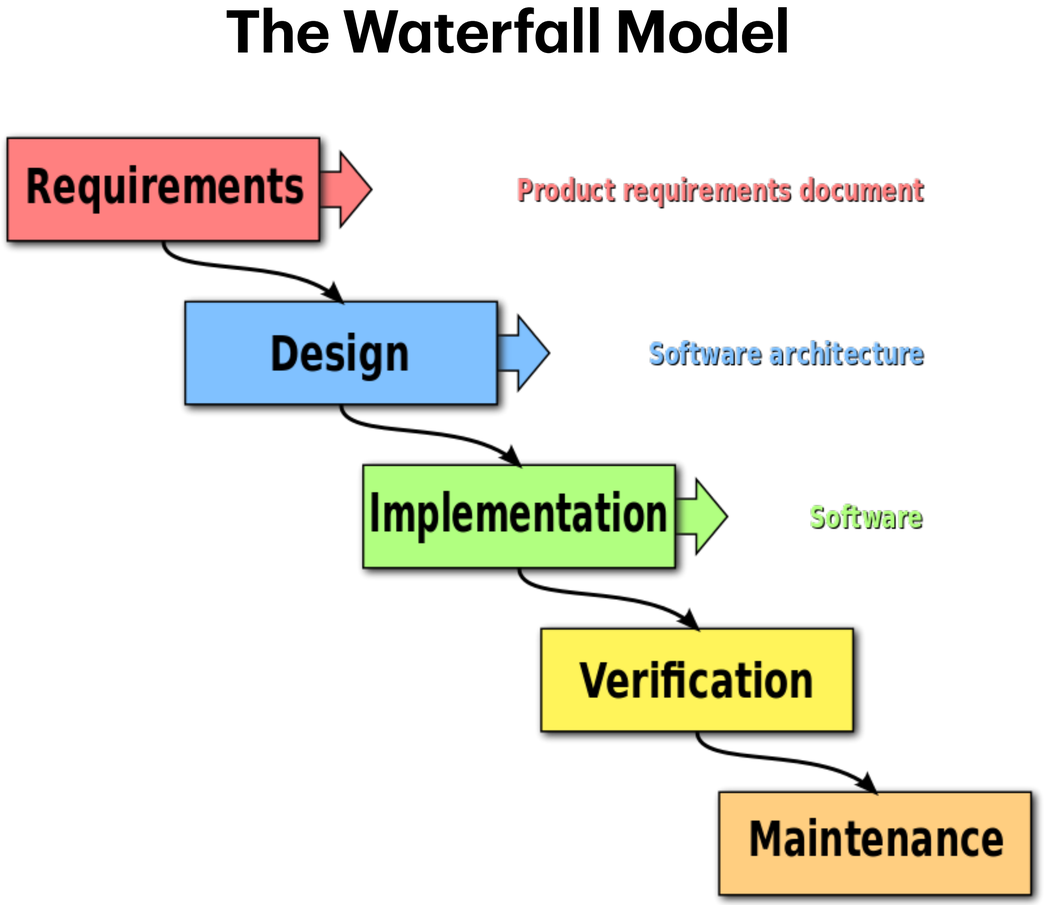
The US government's $44 million COVID vaccine rollout website isn't working — here's how it should be fixed

Tinglong Dai, Feb 8, 2021

Third, the contracting process discourages communications and interactions between vendors and contracting officers. For websites like HealthCare.gov and VAMS that have many stakeholders, the needs of those stakeholders typically evolve during the development process. Companies such as Google, Amazon and Facebook use an "agile" method designed for changes during development. The current federal acquisition process naturally supports a traditional "waterfall" model that largely specifies all requirements at the beginning and allows little room for change.

https://bit.ly/vamsdev

Traditional Software Development Model



What could possibly go wrong with the Waterfall model?

The Agile Manifesto (2001)

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

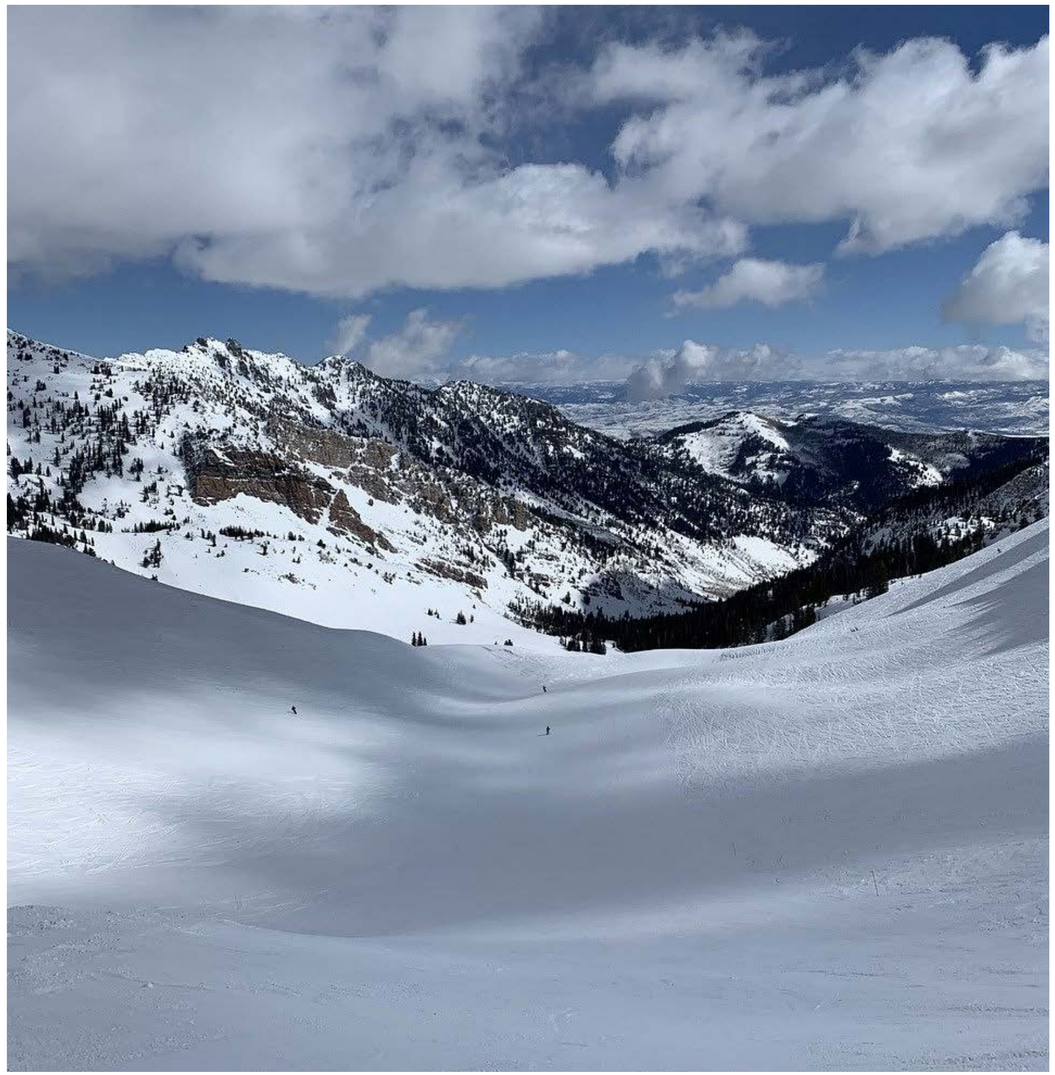
Individuals and interactions over **Processes and tools**

Working software over **Comprehensive documentation**

Customer collaboration over **Contract negotiation**

Responding to change over **Following a plan**

That is, while there is value in the items on the right, we value the items on the left more.



Snowbird, Utah

The Agile Manifesto, Misunderstood

Individuals and interactions over Processes and tools

→ Team members are allowed to do anything

Working software over Comprehensive documentation

→ Don’t document anything; just do it

Customer collaboration over Contract negotiation

→ No contracts should be concluded or negotiated

Responding to change over Following a plan

→ Forget about plans.

Failing to plan is planning to fail!

Scrum

The de facto standard in agile software development

Scrum
noun a way to restart the game after an interruption, where the forwards of each side come together in a tight formation and struggle to gain possession of the ball when it is tossed in among them

What is Scrum?

Scrum is an agile, lightweight process for managing and controlling software and product development in rapidly changing environments

- Iterative, incremental process
- developing systems/products with rapidly changing requirements
- Team-based approach
- Controls the chaos of conflicting interest and needs
- Improve communication and maximize cooperation
- Protecting the team form disruptions and impediments

Functionality of Scrum

https://www.scrumalliance.org/why-scrum

Scrum Has Been Used by

The Power of Scrum

"During the first year of making the switch, Salesforce.com released 94% more features, delivered 38% more features per developer, and delivered over 500% more value to their customers compared to the previous year. . . .

"Fifteen months after adopting Scrum, Salesforce surveyed its employees and found that 86% were having a 'good time' or the 'best time' working at the company. Prior to adopting Scrum, only 40% said the same thing.

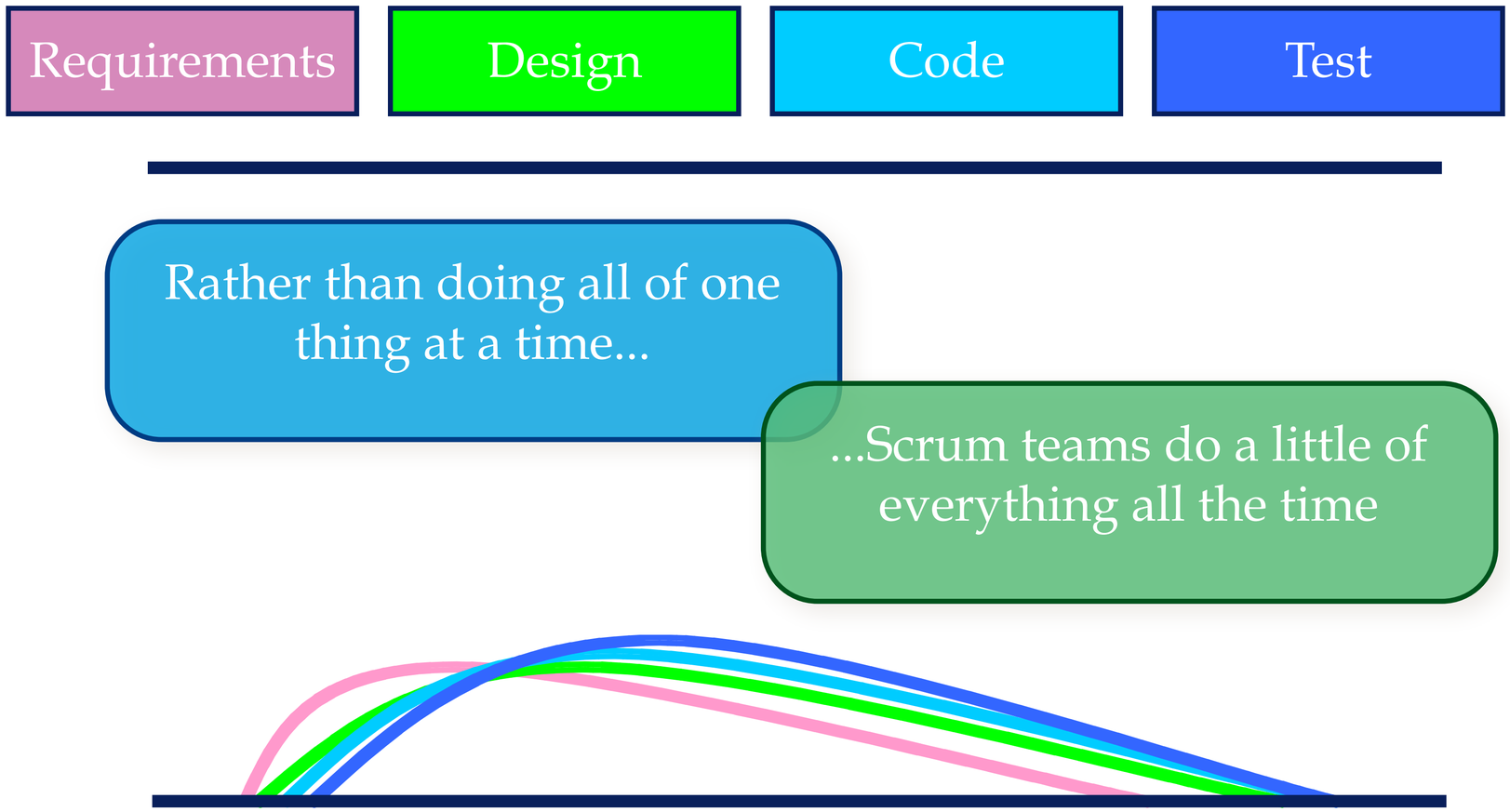
"Further, 92% of employees said they would recommend an agile approach to others."

Mike Cohn, *Succeeding with Agile*

Sprints

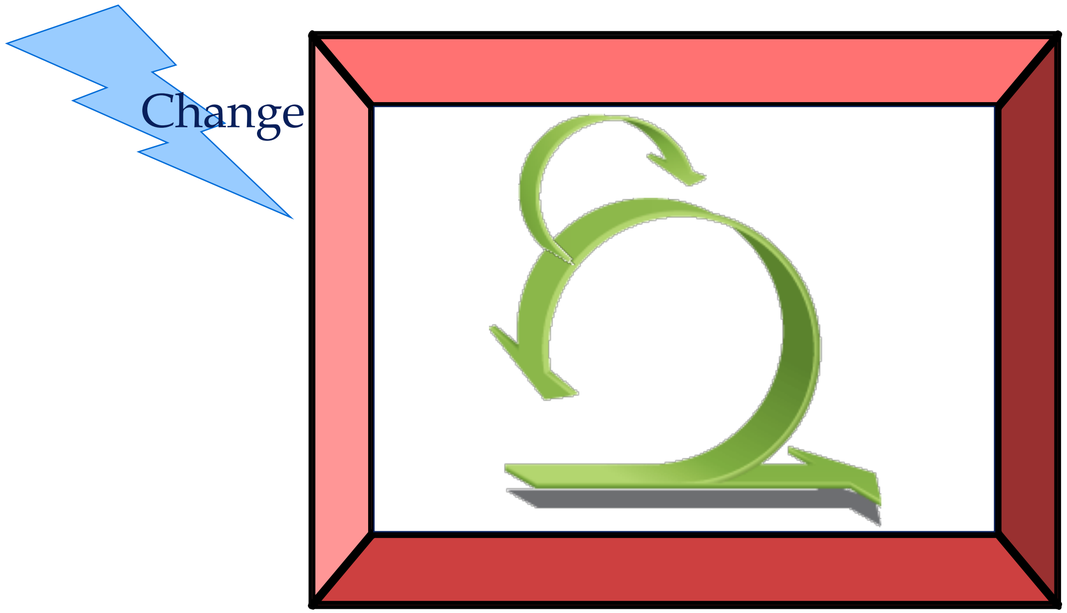
- Scrum projects make progress in a series of “sprints”
- Typical duration is 2–4 weeks or a calendar month at most
- A constant duration leads to a better rhythm
- Product is designed, coded, and tested during the sprint

Sequential vs. Overlapping Development



Source: "The New New Product Development Game" by Takeuchi and Nonaka. *Harvard Business Review,* January 1986.

No Changes During a Sprint



Plan sprint durations around **how long you can commit** to keeping change out of the sprint

Scrum Framework

Roles

- Product owner
- ScrumMaster
- Team

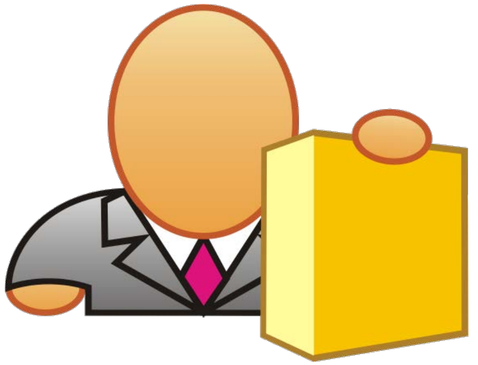
Ceremonies

- Sprint planning
- Sprint review
- Sprint retrospective
- Daily scrum meeting

Artifacts

- Product backlog
- Sprint backlog
- Burndown charts

Product Owner



- Define the features of the product
- Decide on release date and content
- Be responsible for the profitability of the product (ROI)
- Prioritize features according to market value
- Adjust features and priority every iteration, as needed
- Accept or reject work results

The ScrumMaster

- Represents management to the project
- Responsible for enacting Scrum values and practices
- Removes frictions
- Ensures that the team is fully functional and productive
- Enables close cooperation across all roles and functions
- Shields the team from external interferences

The Team

- Typically 5-9 people

- Cross-functional:
  - Programmers, testers, user experience designers, subject matter experts

- Members should be full-time
  - May be exceptions (e.g., database administrator)

The Team

- Teams are self-organizing
  - Ideally, no titles but rarely a possibility
- Membership should change **only between sprints**

The Daily Scrum

- Parameters
  - Daily
  - 15-minutes
  - Stand-up

- Not for problem solving
  - Whole world is invited
  - Only team members, ScrumMaster, product owner, can talk

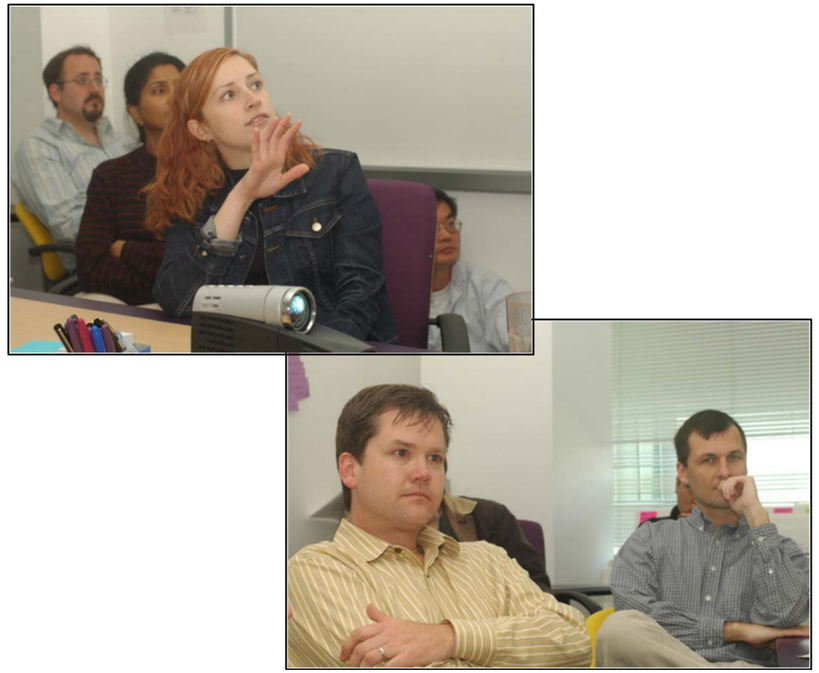
- Helps avoid other unnecessary meetings

Everyone Answers 3 Questions

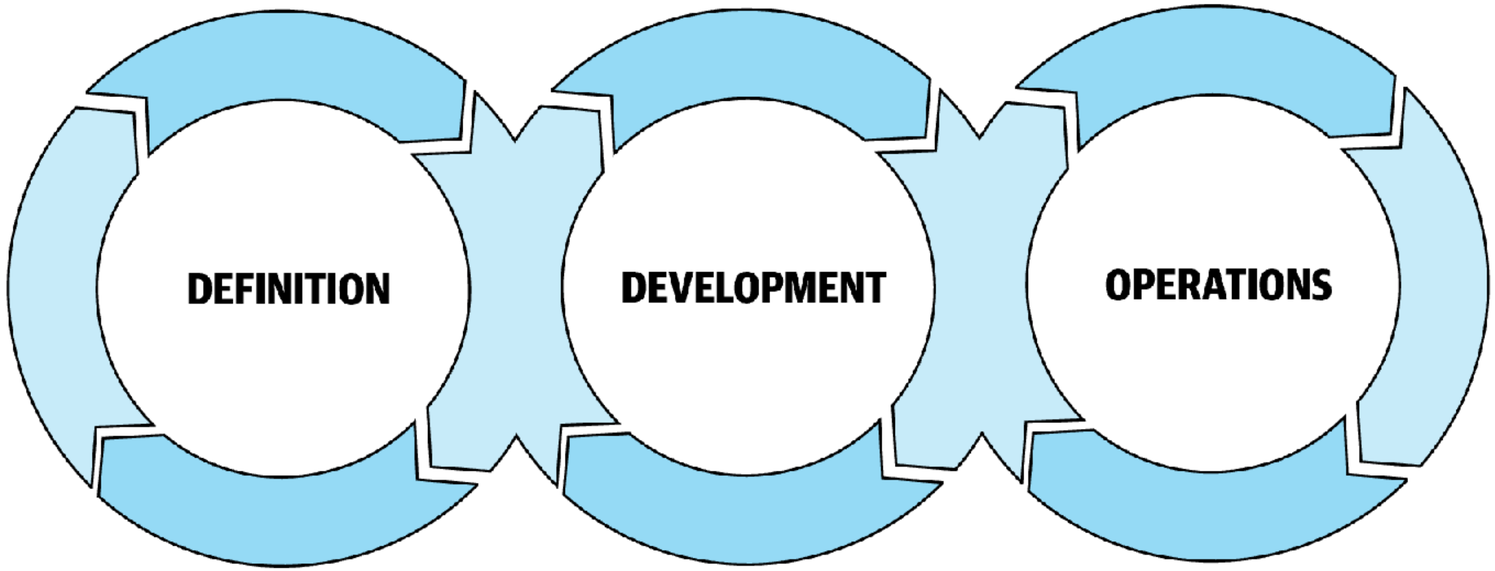
1. What did you do yesterday?
2. What will you do today?
3. Is anything in your way?

The Sprint Review

- Team presents what it accomplished during the sprint
- Typically takes the form of a demo of new features or underlying architecture
- Informal
  - 2-hour prep time rule
  - No slides
- Whole team participates
- Invite the world



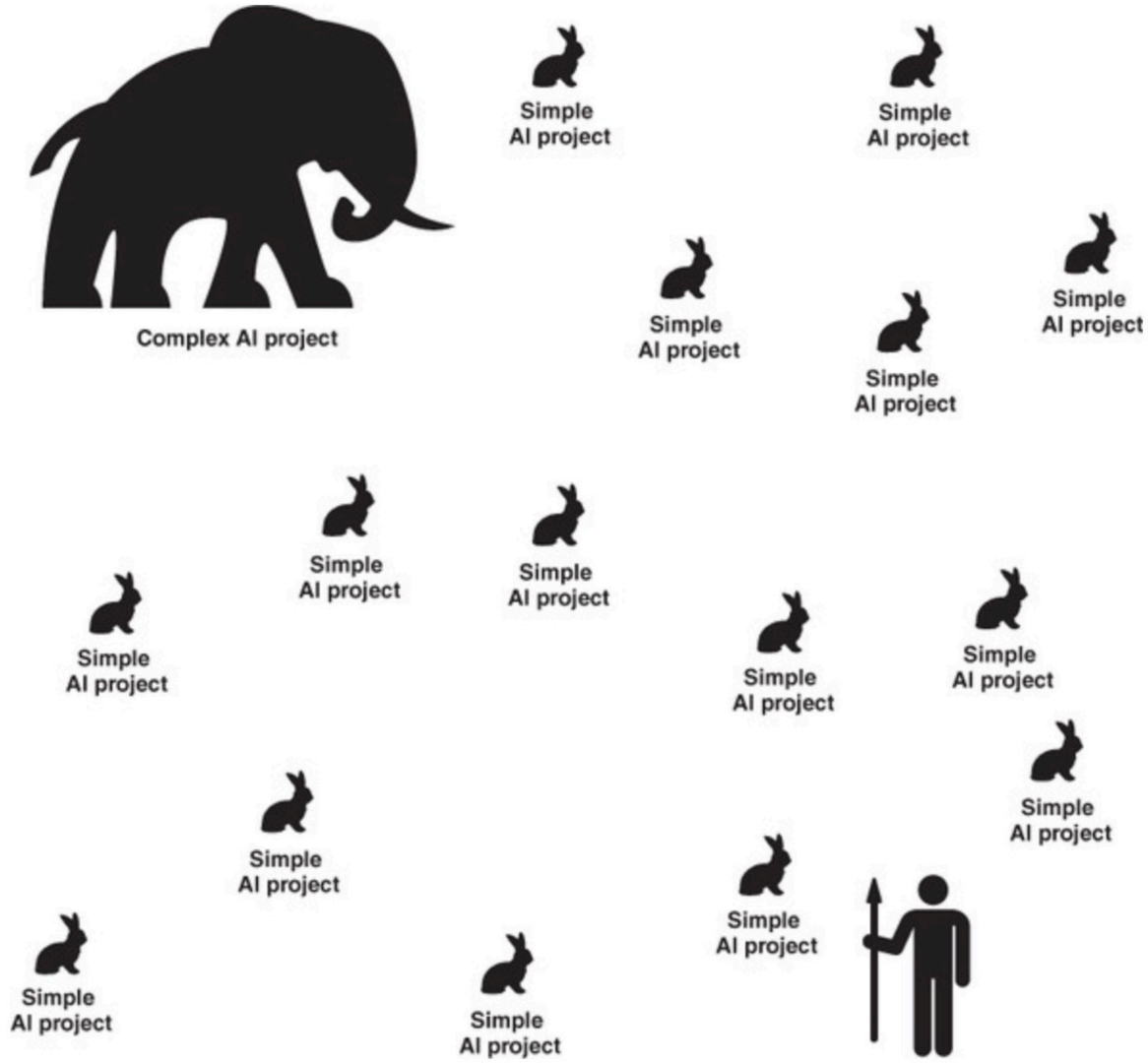
MLOps Loop



- With the right infrastructure in place, the definition/development/operations loop can be dramatically accelerated,
- Result: a product developed with the stakeholders closely integrated in the process, and evolving from real usage

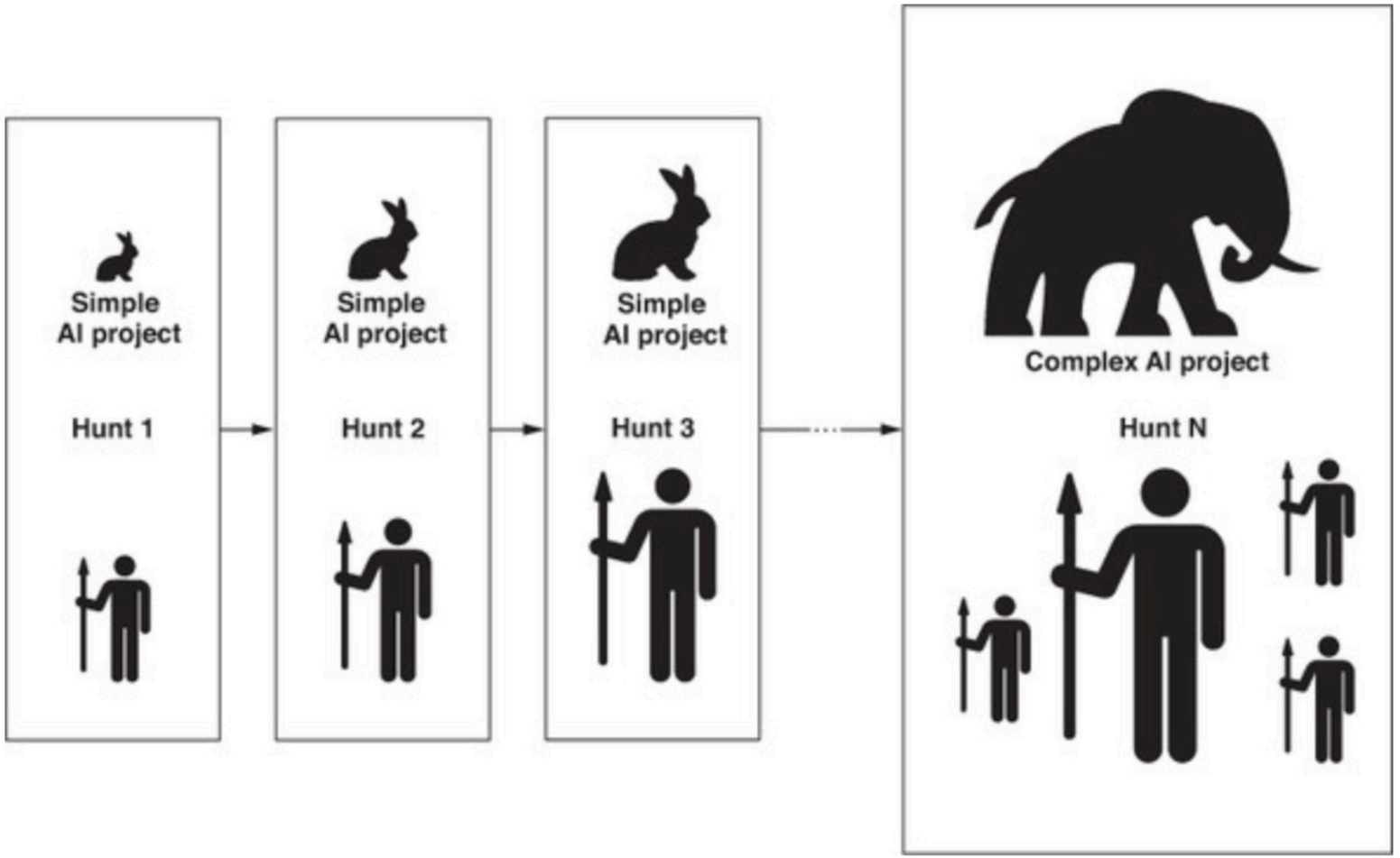
Carmona (2020)

You’re in a rich hunting ground—plenty of rabbits and a big mammoth are in sight. Which animal should you try to catch first?



Krunic (2020)

Start with easy projects. Success with those projects enhances your skills and reputation within the rest of the industry, allowing you to attempt more difficult hunts later.



Krunic (2020)

6

HOW TO CHOOSE YOUR FIRST AI PROJECT

by Andrew Ng

1. Does the project give you a quick win?

Use your first AI pilot project to get the flywheel turning as soon as possible. Choose initial projects that can be done quickly (ideally within 6 to 12 months) and have a high chance of success. Instead of doing only one pilot project, choose two to three to increase the odds of creating at least one significant success.

2. Is the project either too trivial or too unwieldy in size?

3. Is your project specific to your industry?

4. Are you accelerating your pilot project with credible partners?

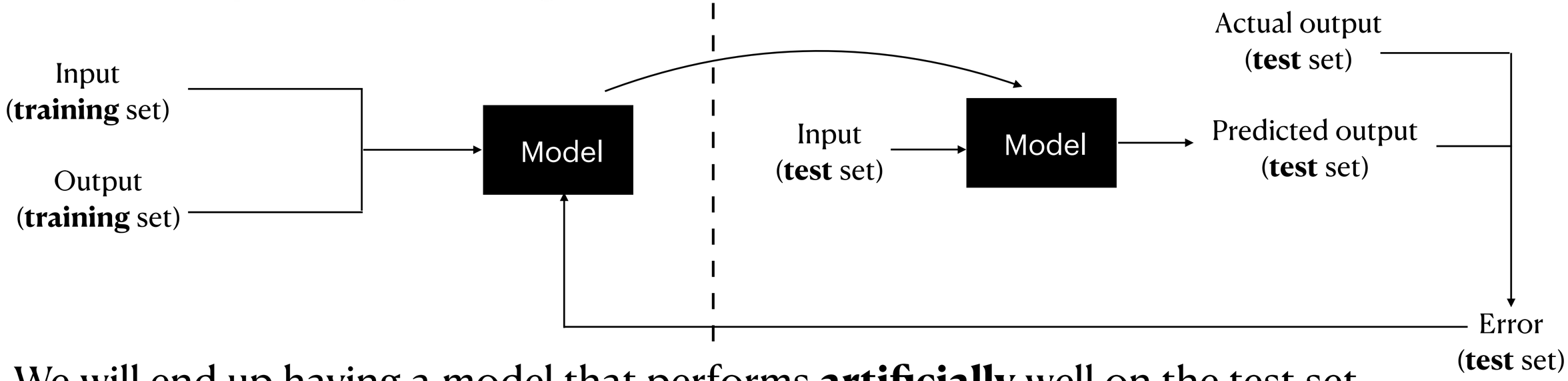
5. Is your project creating value?

Powerful AI Ideas: Dropout & Beyond

Training Leakage

- Imagine when you try to build a diagnosis model using 20,000 patient records
- Let's divide patient records into two parts:

- What could possibly go wrong?



- We will end up having a model that performs **artificially** well on the test set

Solution to Training Leakage

- Principle: To avoid leakage, our model shouldn't have had access to *any* information about the test set, either directly or indirectly

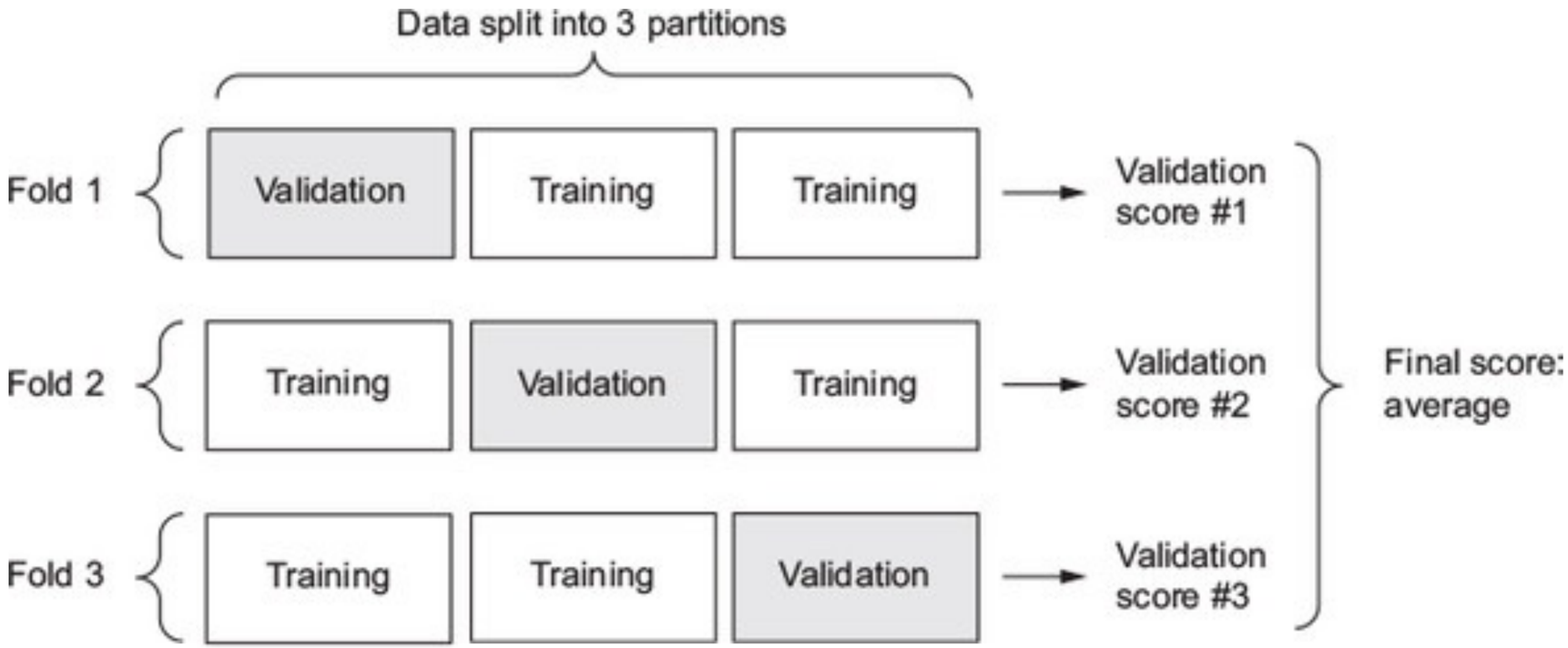
- Solution: Split data into *three* parts: training, **validation**, and test sets
  - The test set is also known as "holdout set" and should never be used to train the model

- Common options: splitting datasets into 6/2/2, 7/2/1, or 8/1/1

Solution to Training Leakage: K-Fold Validation

If a lot of data is available, simple splits (6/2/2, 7/2/1, or 8/1/1)works well.

Otherwise, consider a K-fold validation approach. Example: $K = 3$



Overfitting vs. Underfitting

Optimization vs. Generalization

- Machine learning has two tasks:
  - Optimization: adjust the model to best fit the training data
  - Generalization: ensure the model performs well on data it has *never* seen before

- Example: What's the best way to prepare for the U.S. medical licensing exam?
  - Memorizing answers to the sample tests ≠ doing well in the real test

- **Overfitting** occurs when the algorithm has learned too much from the training data

- **Underfitting** occurs when the algorithm hasn't modeled all relevant patterns in the training data

The Central Problem of Deep Learning: **Overfitting**

How to fight overfitting?

Key Strategies to Fight Overfitting

- Get more training data
- Reduce the size of the neural network (“pruning”)
- Add weight regularization
- Add dropout

Reduce the Size of the Network

- Two fundamental questions:
  - **How many layers should I use?**
  - How many neurons per layer should I use?

- Bottomline: A neural network with a single hidden layer can model any function regardless of how complex it is (due to the Universal Approximation Theorem)

- However, if we use a single hidden layer, we may need many neurons, so giving limited data, overfitting is more likely to happen

- Rule of thumb: **use a single hidden layer for a simple problem, and two hidden layers for a complex problem**

- More than two hidden layers are necessary only for complicated image, video, or speech data

Reduce the Size of the Network

- Two fundamental questions:
  - How many layers should I use?
  - **How many neurons per layer should I use?**

- No universal rules exist, but it's usually good to start with a small network and slowly add more neurons until overfitting occurs

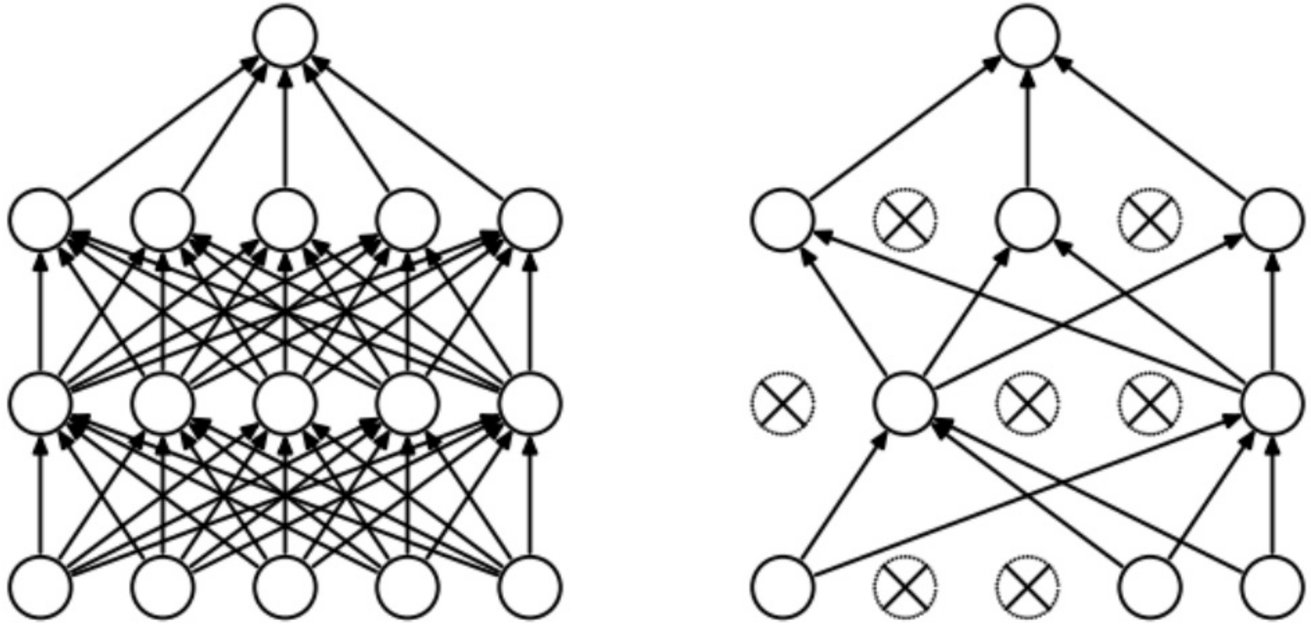
- What to do after overfitting happens?

Adding Weight Regularization

- Weight regularization: Punish a model for being too complicated by forcing its weights to take only small values
  - L1 regularization: The penalty is proportion of the absolution value of the weight coefficients (also known as **lasso,** i.e., least absolute shrinkage and selection operator)
  - L2 regularization: The penalty is proportion of the square of the absolute value of the weight coefficients (also known as **ridge** regression)
- Let's see some examples via TensorFlow Playground: **https://bit.ly/anntest2**

Dropout

Randomly dropping out (setting to zero) a number of output features



(a) Standard Neural Net

(b) After applying dropout.

What's the purpose of dropout?

Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting", JMLR 2014

How Well Does Dropout Work?

- Alex Krizhevsky broke the 2012 ImageNet record by using the dropout technique
- If your neural network is significantly overfitting, dropout can often dramatically improve the accuracy
- If your neural network is not overfitting yet, you should expand its size until it starts overfitting

“I went to my bank. The tellers keep changing and I asked one of them why. He said he didn’t know but they got moved around a lot. I figured it must be because it would require cooperation between employees to successfully defraud the bank. This made me realize that randomly removing a different subset of neurons on each example would prevent conspiracies and thus reduce overfitting.”

Geoffrey Hinton, inventor of the dropout technique

Why Dropout Works

- If a hidden unit knows which other hidden units are present, it can co-adapt to them on the training data
  - But complex co-adaptations are likely to go wrong on new test data
  - Big, complex conspiracies are not robust

- If a hidden unit has to work well with combinatorial many sets of co-workers, it is more likely to do something that is individually useful
  - But it will also tend to do something that is marginally useful given what its co-workers achieve

Part of Group Assignment #2: Let's see how dropout works in Keras

Recurrent Neural Networks (RNNs)

FNNs and CNNs Have No Memory

Both feedforward and convolutional neural networks process an entire sample all at once and learn from each sample

E.g., CNN reads an entire review from IMDB and processes it in one go

A movie that will not be outclassed in its genre for years to come
jayp-5 4 December 1999

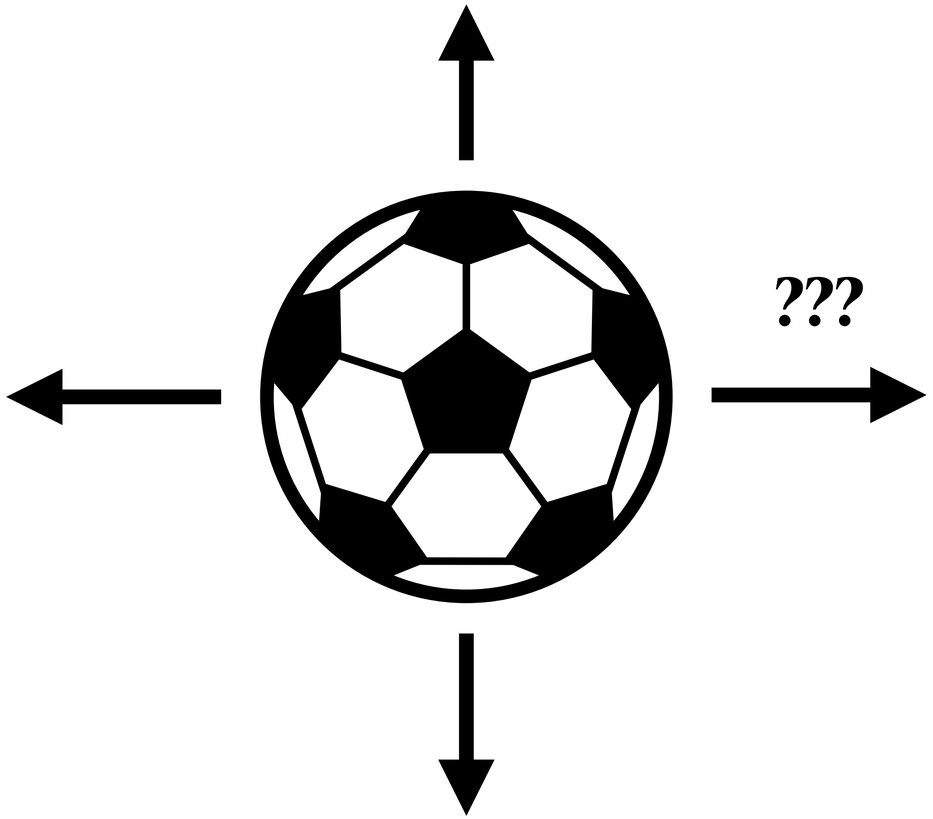
When I first saw The Sixth Sense, I didn't know what to expect. I guess I was looking forward to a good scary horror flick. I was very surprised. I found that the purpose for this movie was far greater than just trying to scare the audience. I found this movie was showing not only the emotions of fear, but also faith, commitment, sadness of loss, and love. The end was so surprising, I had to see it again. The second time I watched it, I did it from a totally different perspective (this is a very rare quality for any movie), and I enjoyed it just as much, or maybe even more. I also, as many viewers have, tried to detect fallacies in the story. I couldn't find one. In addition, for those that appreciate great soundtracks, the music only helps to heighten the experience of the movie.

I believe that a great movie is one that helps the viewer perceive life and the world differently. The Sixth Sense is one of those extraordinary movies that does that to me. This movie reflects on some difficult subjects that will make the viewer walk away asking eternal questions. Questions about death, about letting go, about eternal love and commitment, about the love between parent and child, and between husband and wife. Maybe I read too much into this very wonderful film, but I believe it will be difficult to find a movie that has touched on these subjects so poignantly and so well for years to come.

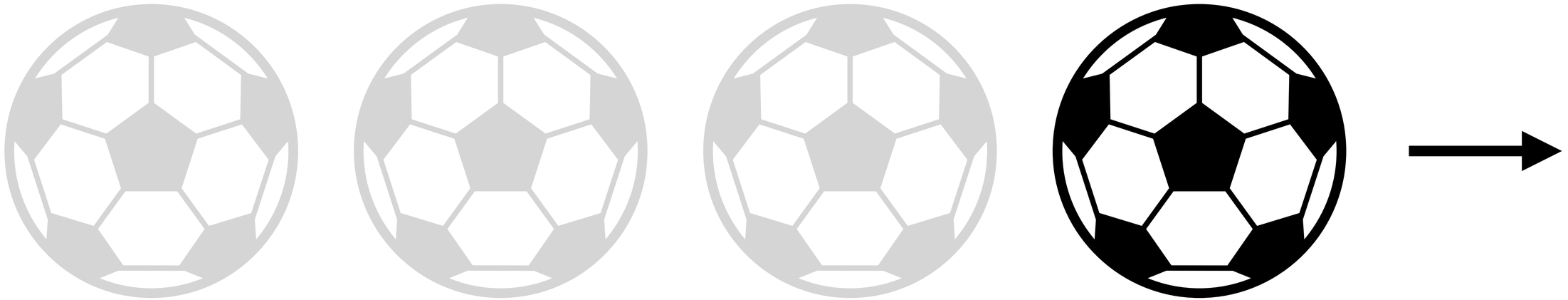
Positive or Negative?

Given an Image of a Soccer Ball:

Can You Predict Where It Will Go Next?



Can You Predict Where It Will Go Next?



Sequence Data

Torvill & Dean's Dancing On Ice 2009 — Bolero

Four Types of Sequence Learning

- One to one (e.g., binary classification): Predicting whether it's going to rain tomorrow or whether someone suffers from lung cancer

- Many to one (e.g., sentiment classification): Generating a social media sentiment score about stocks based on related tweets

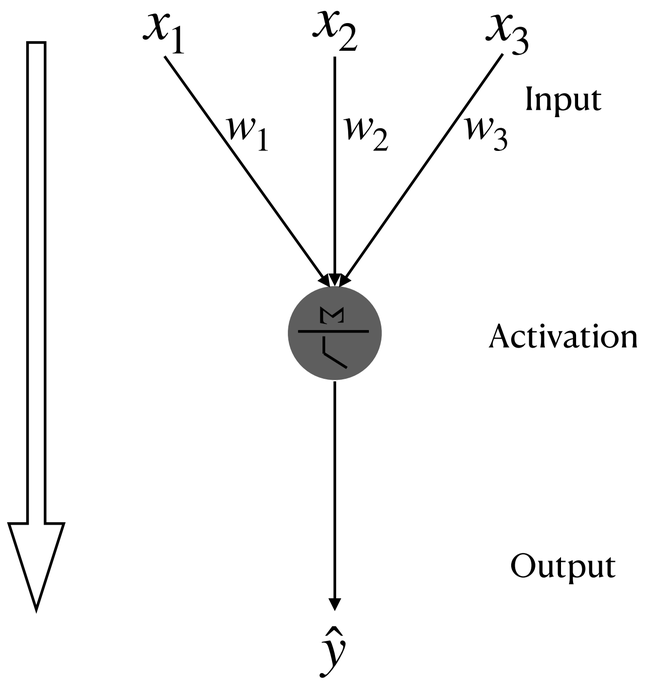
- One to many (e.g., automatic image captioning)

- Many to many (e..g, machine translation)

Ships were in limbo outside the ports of Los Angeles and Long Beach on Sunday

Artificial Neurons Revisited

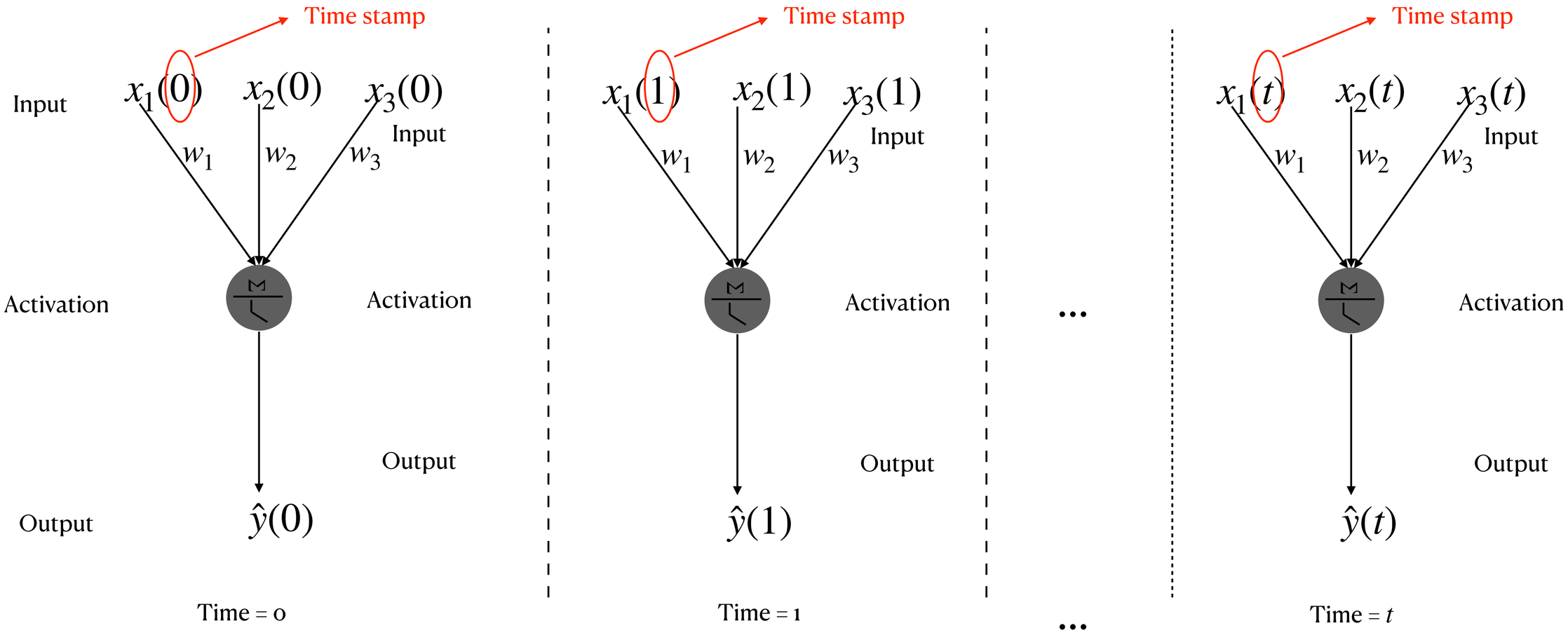
Artificial Neuron



A Video is a Collection of Images Captured at Different Times

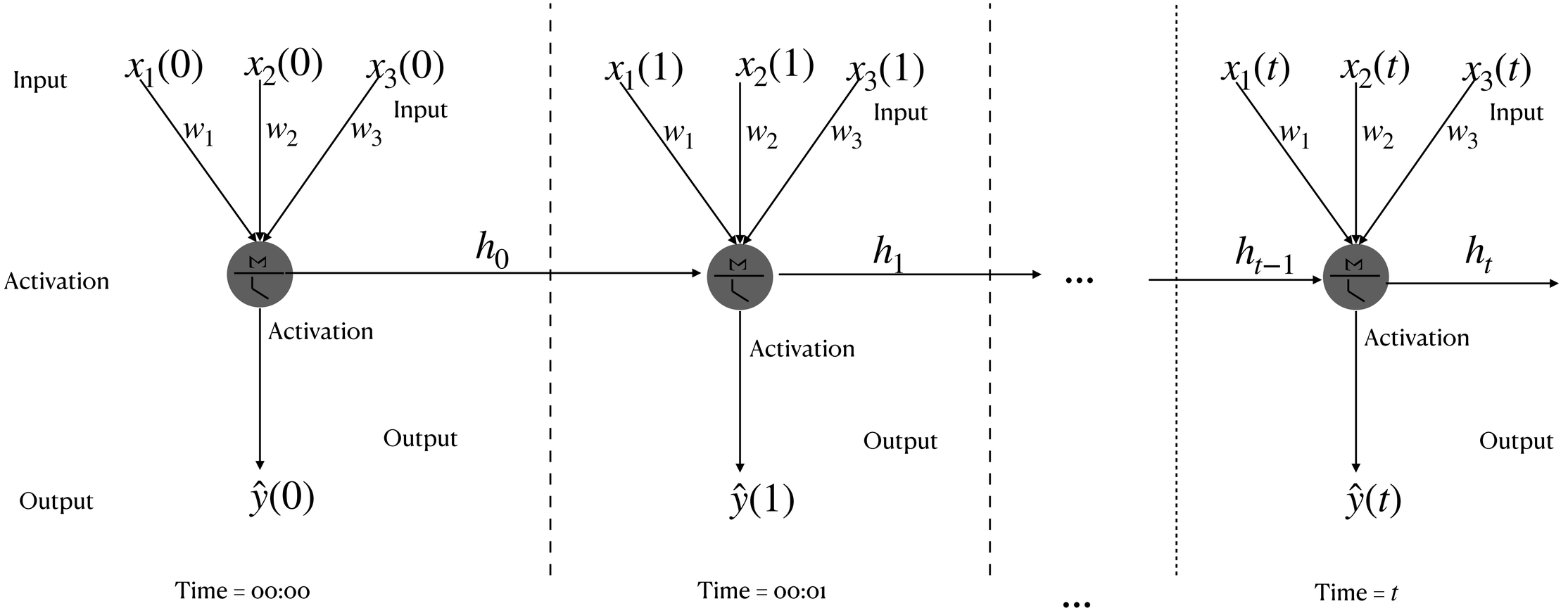
=

Artificial Neurons with Time Steps



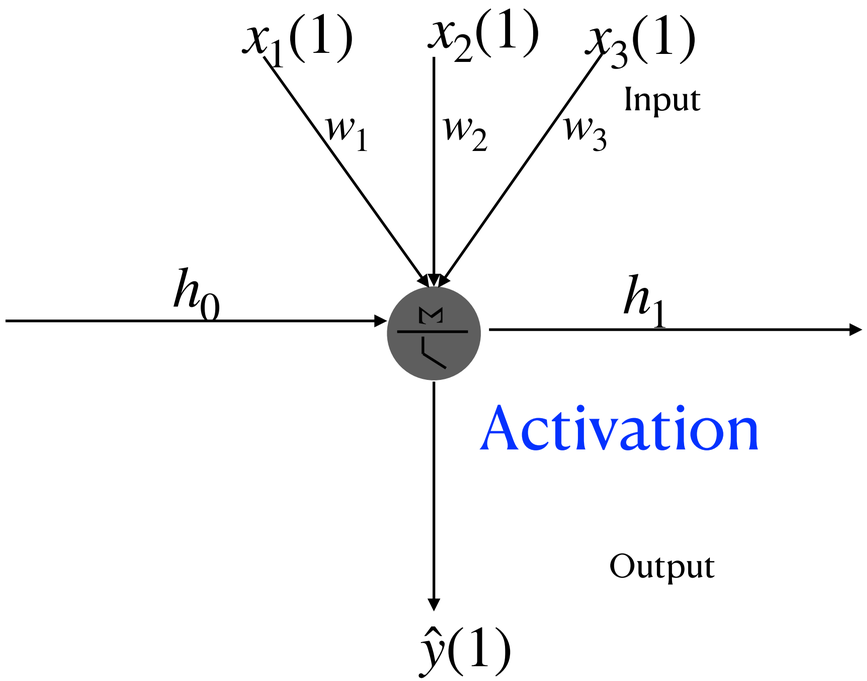
These neurons are **incapable** of learning a sequence because they are independent of each other

Neurons with Recurrence



At each neuron, the output is a function of inputs **AND past memory**

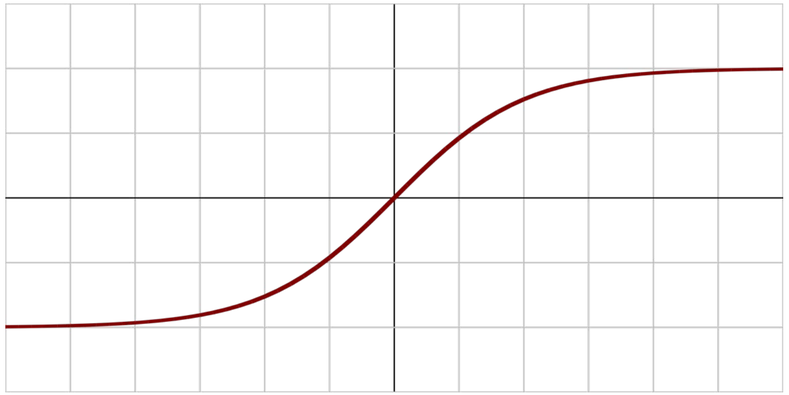
Recurrent Neural Networks (RNNs)



At time 1

Activation Functions: Beyond ReLU

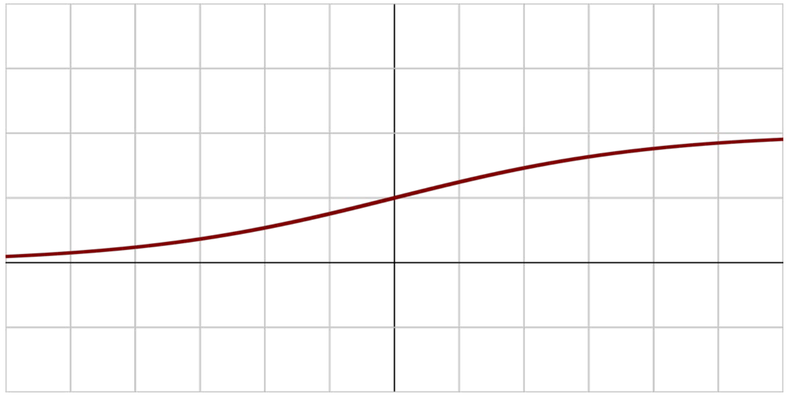
Tanh and Sigmoid — “S”-shaped functions



Tanh function

$$\text{Tanh}(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

“Tanh” rhymes with “branch”



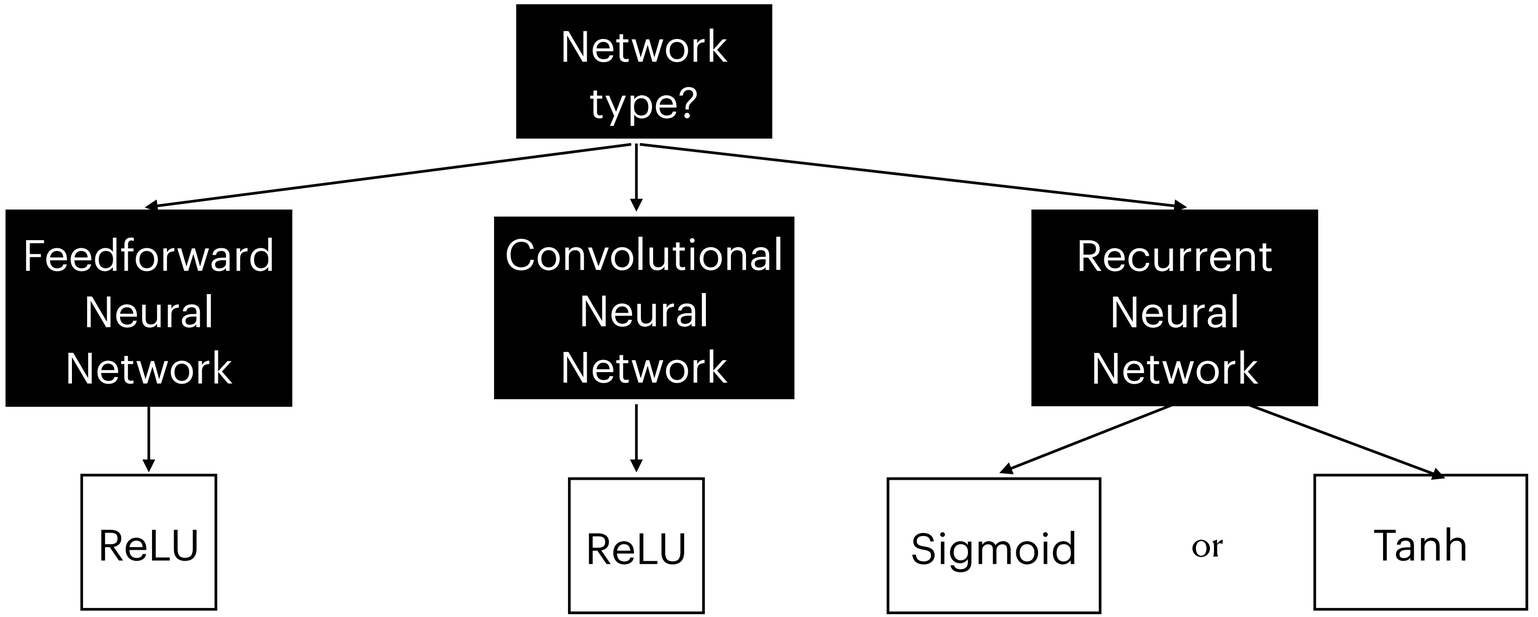
Sigmoid function

$$\text{Sigmoid}(x) = \frac{e^{x}}{e^{x} + 1}$$

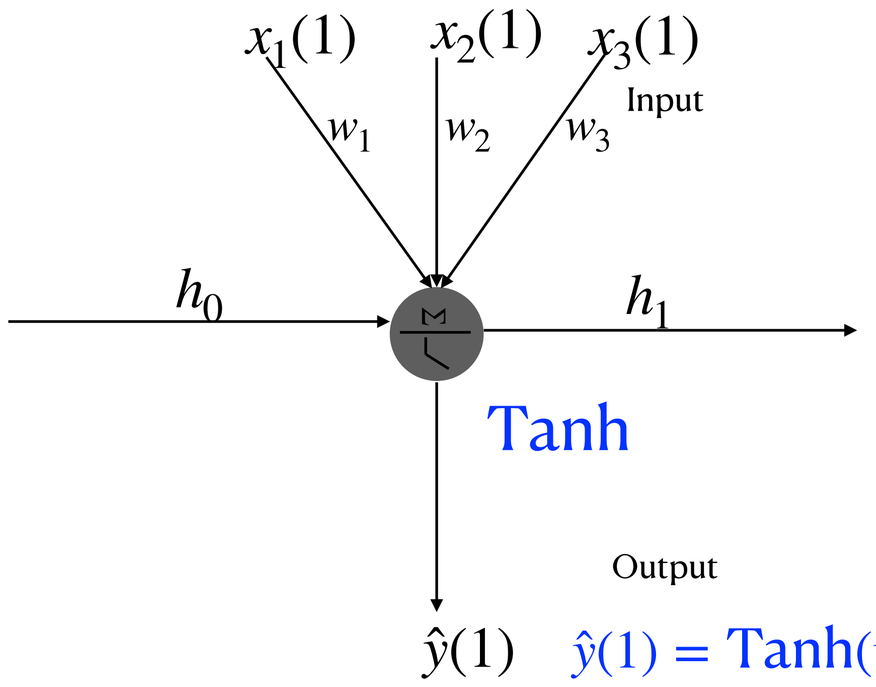
How to Choose Activation Function

A bit of a history lesson

- Until mid-1990s: Sigmoid was the default activation function for training neural networks
- Between mid-1990s to 2010s: Tanh became default activation function for hidden layers
- 2010s to now: the default recommendation is **ReLU** for most of the modern neural networks



Recurrent Neural Networks (RNNs)



$$h(1) = \text{Tanh}(w_1^h \cdot x_1(1) + w_2^h \cdot x_2(1) + w_3^h \cdot x_3(1) + h_0)$$

$$\hat{y}(1) = \text{Tanh}(w_1 \cdot x_1(1) + w_2 \cdot x_2(1) + w_3 \cdot x_3(1) + h_0)$$

The rest is about updating the weights, similar to the case of feedback neuron networks...

Keras for Simple RNN

- The RNN can be implemented in Keras using:

```
layer.SimpleRNN(units)
```

- Example:

```
num_features = 14
inputs = keras.Input(shape=(None, num_features))
outputs = layers.SimpleRNN(16)(inputs)
```

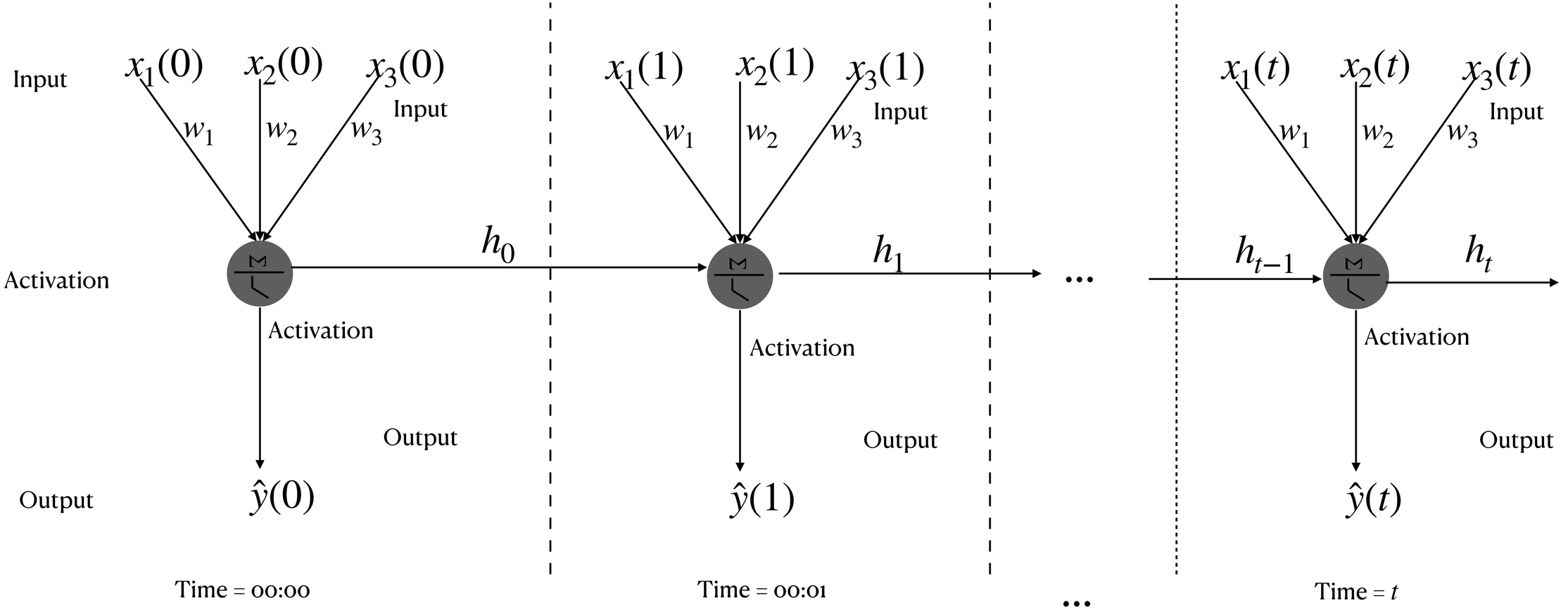
Timestamps entry; setting it to “None” enables your network to process sequences of arbitrary length

Embedding Layer: Example

Input

Output

Major Issue with RNNs: No Long-Term Memory



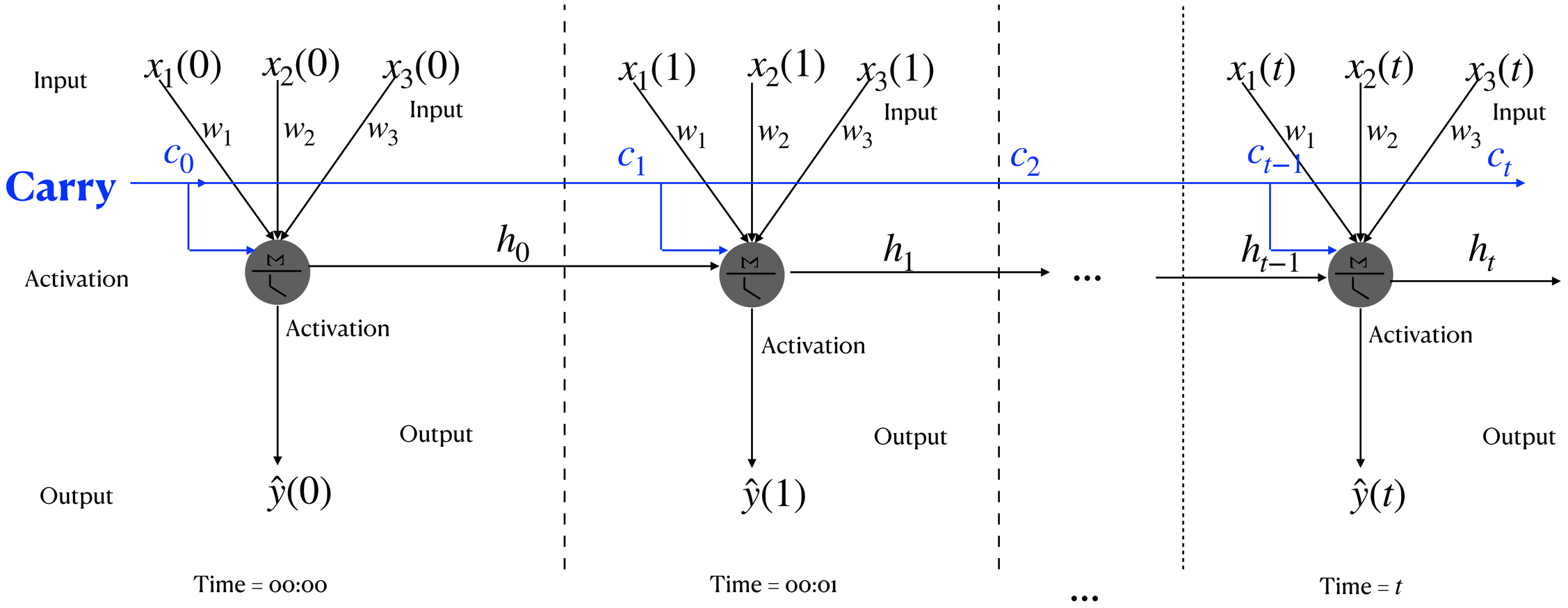
At best, $h_0, h_1, h_2, \ldots$ only captures the short-term memory and does not reflect long-term tendencies

Major Issue with RNNs: Vanishing Memory

- A Simple RNN (`layer.SimpleRNN` in Keras) does not reflect long-term dependencies and suffers from a **vanishing gradient problem**.
  - After a while, the RNN's state contains virtually no trace of the first inputs

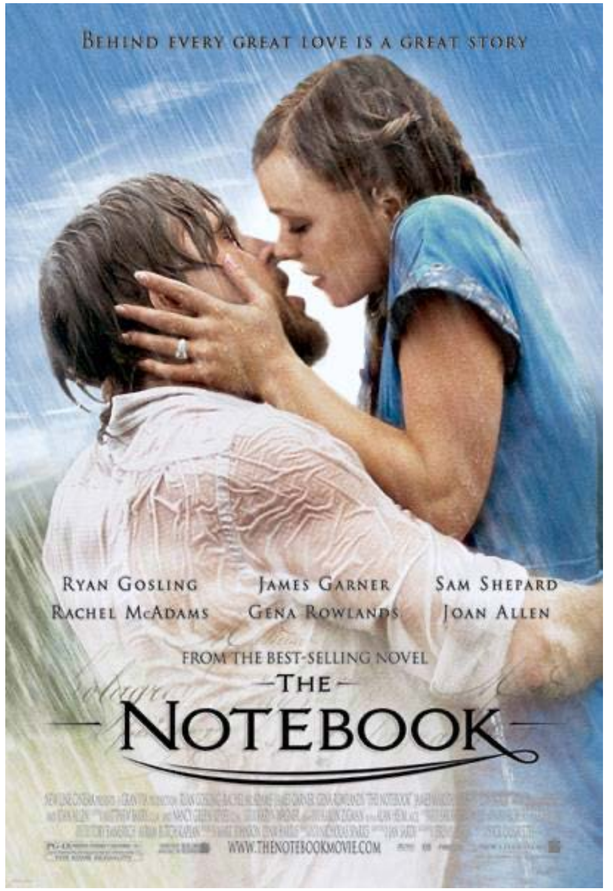
We need to figure out a way to give RNNs the ability to keep track of long-term dependencies...

Long Short Term Memory (LSTM) RNNs

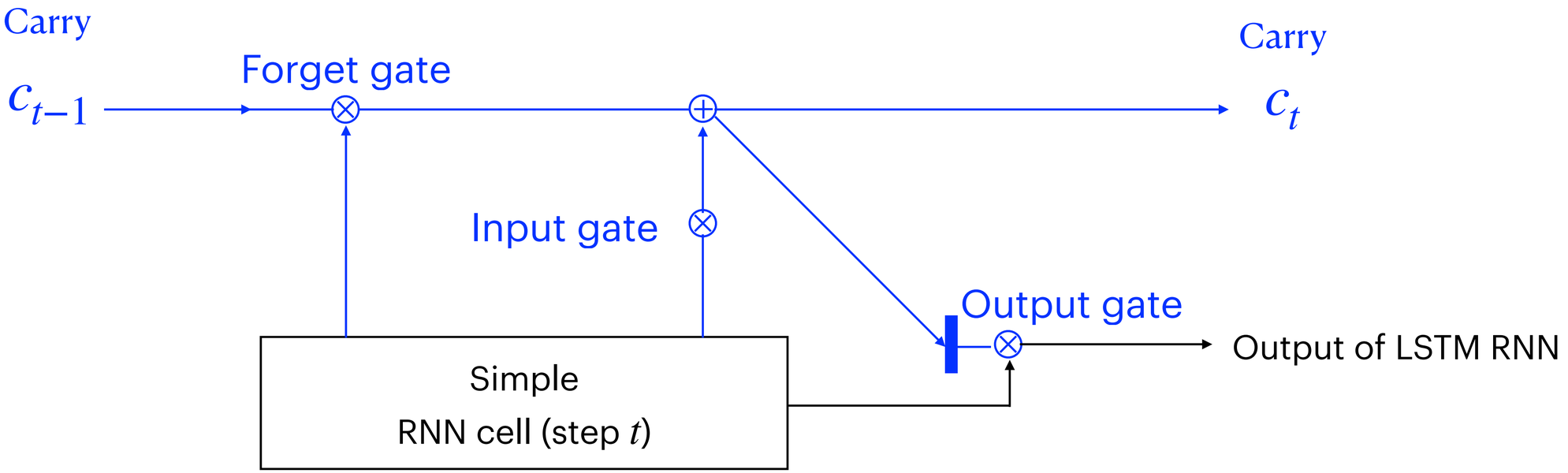


Core Idea of LSTM Recurrent Neural Network

- The **carry** keeps track of long-term tendencies (think of it as a notebook)
- At each time step, the inputs and the previous short-term memory are fed into a simple RNN cell and three gates:
  - The **forget gate** decides what part of long-term tendencies should be erased
  - The **input gate** decides which part of the output of the simple RNN should be added to long-term tendencies
  - The **output gate** decides which part of long-term tendencies should be read and used at this time step



How LSTM Carry is Updated



- The **forget gate** decides what part of long-term tendencies should be erased
- The **input gate** decides which part of the output of the simple RNN should be added to long-term tendencies
- The **output gate** decides which part of long-term tendencies should be read and used at this time step

“You don’t need to understand anything about the specific architecture of an LSTM cell; as a human, it shouldn’t be your job to understand it. Just keep in mind what the LSTM cell is meant to do: allow past information to be reinjected at a later time, thus fighting the vanishing-gradient problem.”

Francois Chollet, *Deep Learning in R*

Long Short Term Memory (LSTM) in Keras

- The RNN can be implemented in Keras using:

```
layers.LSTM(32)(inputs)
```

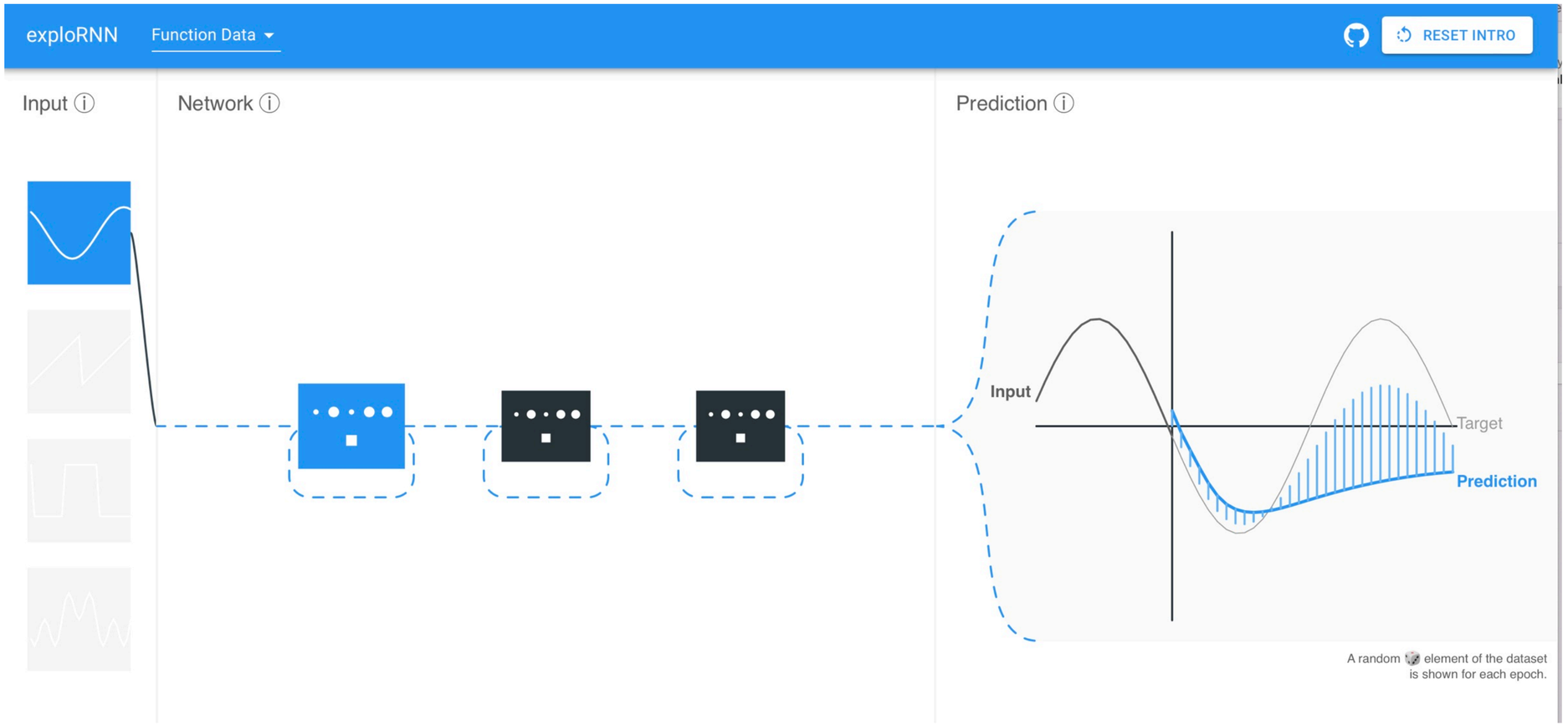
- Example:

```
inputs = keras.Input(shape=(num_timesteps, num_features))
x = layers.LSTM(32)(inputs)
outputs = layers.Dense(num_classes, activation="sigmoid")(x)
model = keras.Model(inputs, outputs)
model.compile(optimizer="rmsprop", loss="binary_crossentropy")
```

https://bit.ly/session4rnn

Visualizing Long Short Term Memory (LSTM)

https://bit.ly/lstmrnn



The central challenge of sequential learning:

The balance between long- and short-term memory

Now, enter **The Transformer**

Transformer?

The Transformer

Large Language Models (LLMs)

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

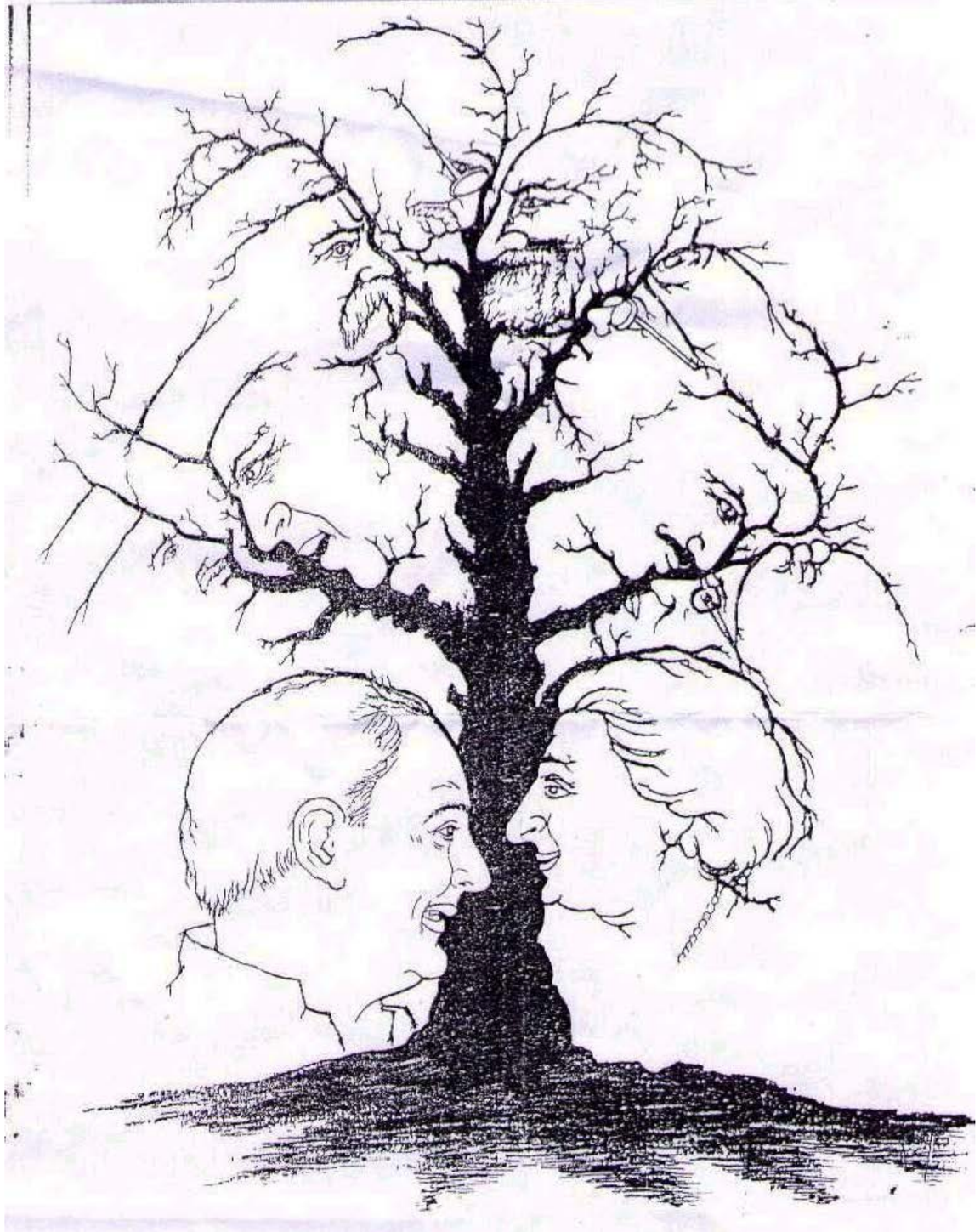
Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to

https://bit.ly/attention17



How many faces are in the picture?

"National Leaders Tree"

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

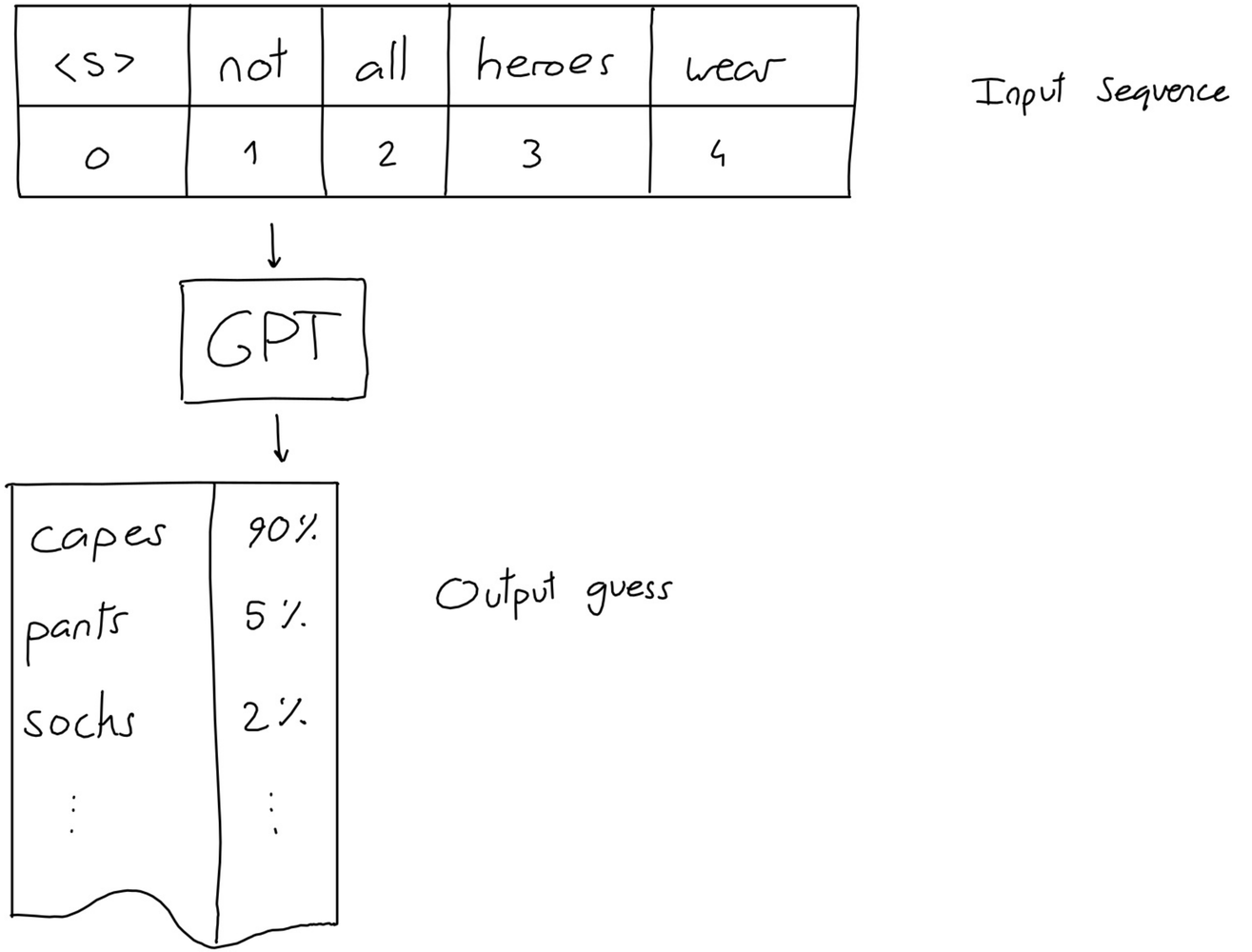
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to

https://bit.ly/attention17

GPT = Generative Pre-trained Transformer

- **Generative Pretraining (“GP”)**, developed decades ago, seeks to train a language model unsupervised on a large corpus of text data
- The goal of GP: Generate text that closely resembles human-written text by **predicting the next word** in a given sequence
- The **Transformer (“T”)** architecture was developed in **2017**, in a paper written by Google employees (Ashish Vaswani et al.) called "Attention is All You Need"
- **OpenAI**, founded in 2015 by Sam Altman, Greg Brockman, Reid Hoffman, Jessica Livingston, Peter Thiel, and Elon Musk, was the first to combine “GP” and “T” to invent GPT in **2018**
- **ChatGPT (GPT-3.5)** was lunched on **November 30, 2022**

What GPT Does = Predicting the Next Word



https://bit.ly/gpt-napkin

What GPT Does = Predicting the Next Word

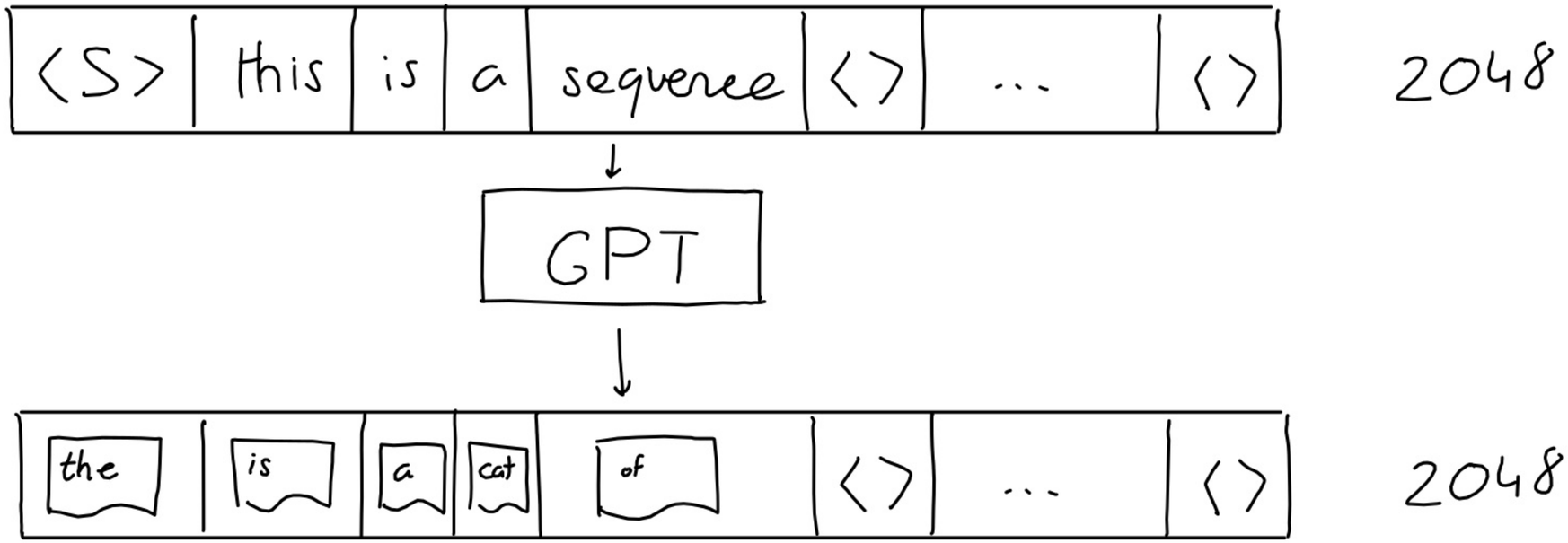
- Not all heroes wear → *capes*
- Not all heroes wear capes ...
- Not all heroes wear capes → *but*
- Not all heroes wear capes but ...
- Not all heroes wear capes but → all
- Not all heroes wear capes but all → *villains*
- Not all heroes wear capes but all villains ...
- Not all heroes wear capes but all villains → *do*

https://bit.ly/gpt-napkin

What GPT Does = Predicting the Next Word

In the case of GPT-3:

- **Input:** fixed to 2,048 words; for shorter inputs, extra positions are filled with empty values
- **Output:** a sequence (2,048 words) of guesses, one for each “next” position in the sequence



Predicting the Next Word = A Sequence Learning Problem

- Key challenge: The balance between long- and short-term memory

- Recurrent Neural Networks (RNNs) learn the texts **sequentially**, word by word, so the memory problem is severe

- The Transformer learns all the texts **simultaneously**, using the concept of **attention**

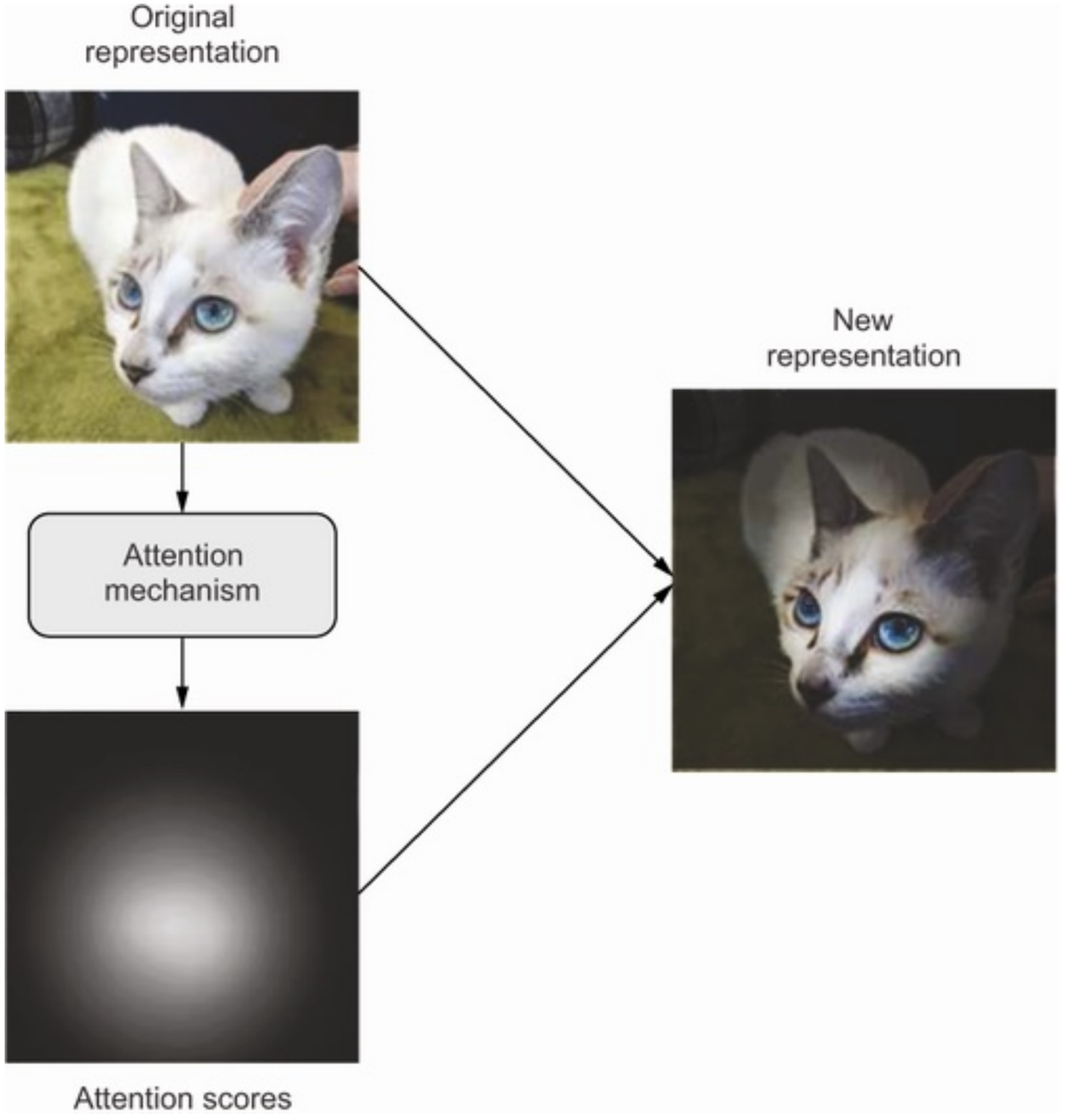
Attention

A mechanism in deep learning to focus on specific input parts

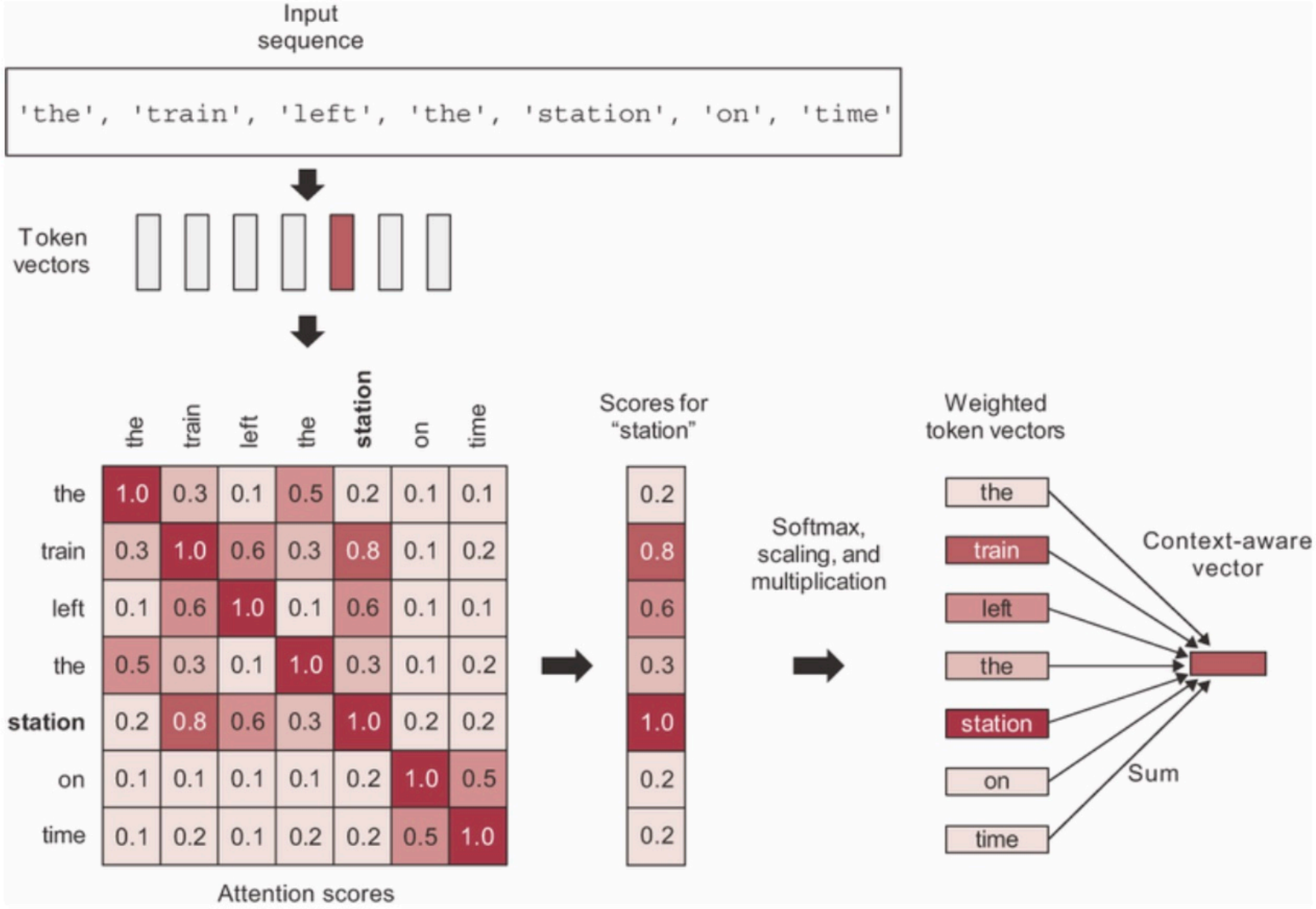
- Just like you skim through books, focusing on relevant parts, deep learning models can do the same

- Key Idea: Not all information is equally crucial. Models should focus on important features

- We've already seen a similar concept — MaxPooling in CNNs
  - Selects the most important feature in a region, ignores the rest
  - Essentially an "All or nothing" approach to attention

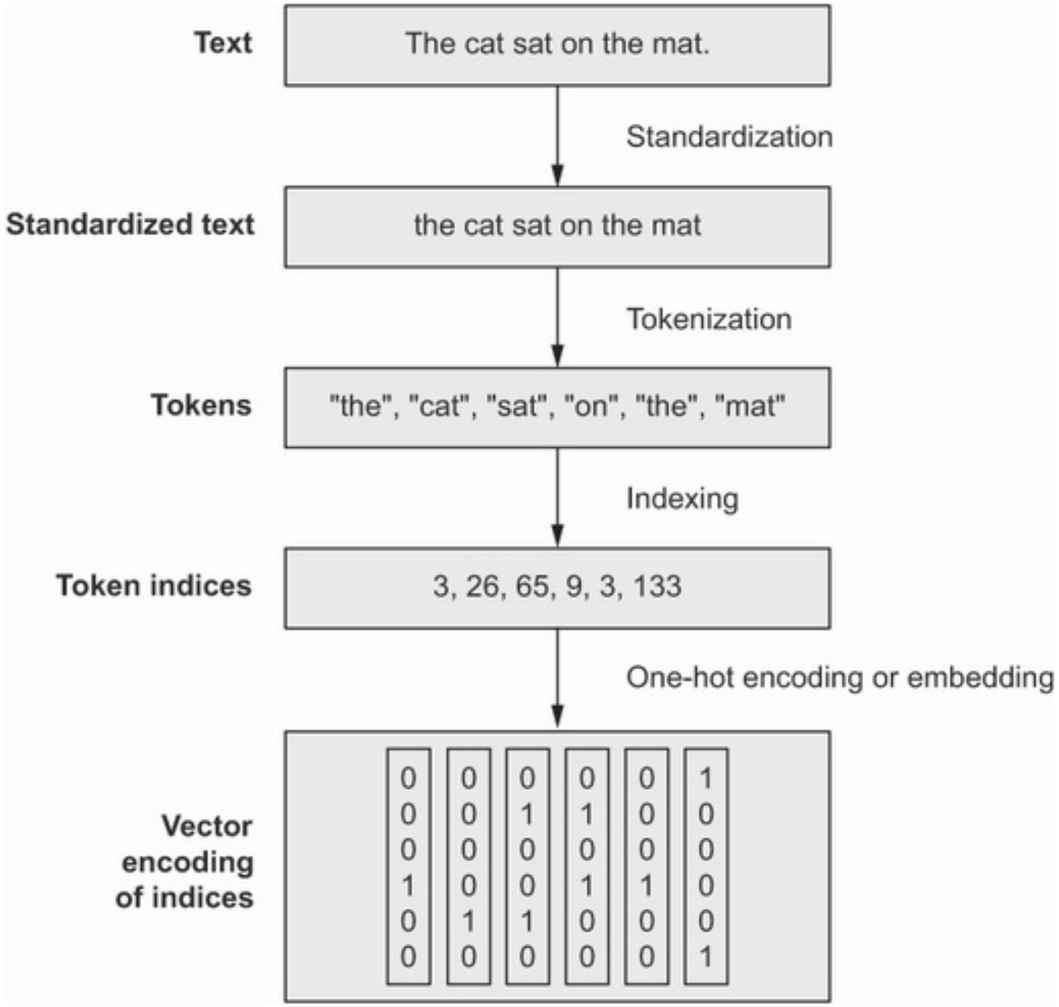

Original representation
Attention mechanism
Attention scores
New representation

Attention Is All You Need for Predicting the Next Word



Preparing the Text Data: *tokenization* and *vectorization*

Convert Text Data Into Numbers



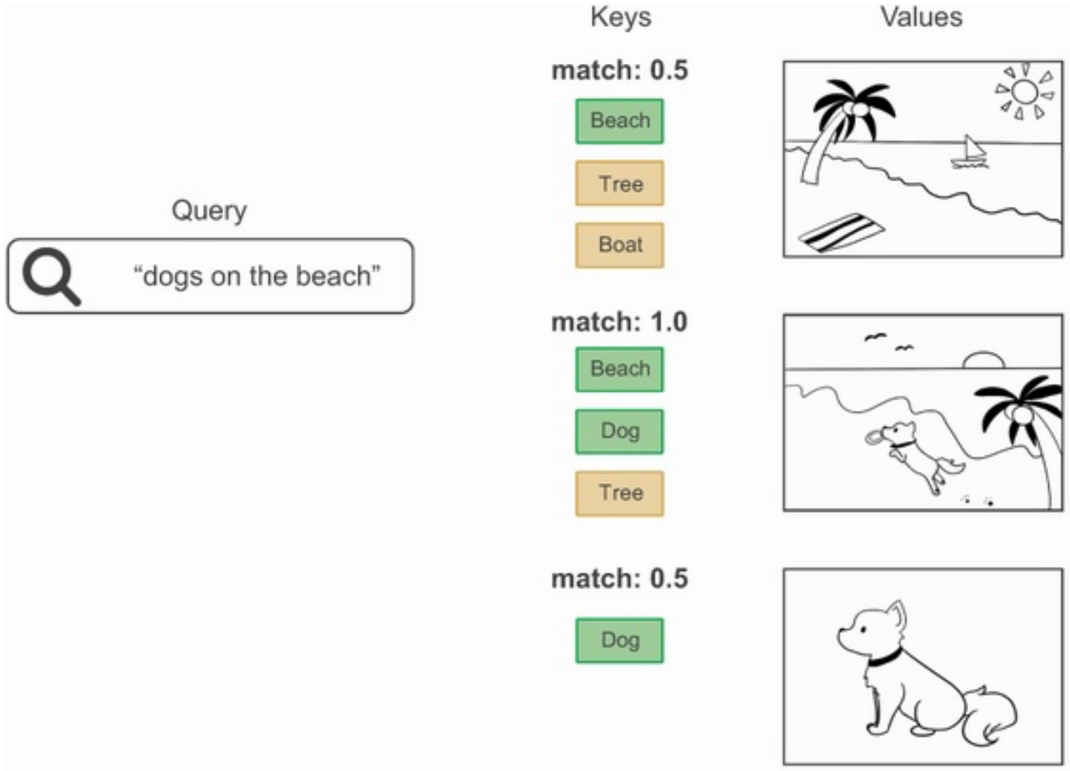
OpenAI's tokenizer tool:

https://platform.openai.com/tokenizer

Rule of thumb: For typical English text, one token represents ~4 characters.
This equals about ¾ a word (100 tokens ≅ 75 words)

Where Do the Attention Scores Come From?

The Query-Key-Value Model



Retrieving images from a database: the "query" is compared to a set of "keys," and the match scores are used to rank "values" (images)

Where Do the Attention Scores Come From?

The Query-Key-Value Model

- Core Components
  - *Query*: What you're looking for
  - *Keys*: Descriptors for the values
  - *Values*: Information you want to extract
- Matching Process: Match the query to keys to find relevant values, and return a weighted sum of these values
- In Practice: query, keys, and values are all the same
- We’ll talk more about how attention scores are computed

The Training Process of a Large Language Model:

https://bit.ly/nytllm

If you haven't done so, make sure you activate your *New York Times* subscription via https://bit.ly/jhunyt

More on the Transformer Next Time

Plus discussions on Building LLM Applications using API

We are Doomed...

Geoffrey Hinton knows one thing or two about AI

Yoshua Bengio
(1964–)

Geoffrey Hinton
(1947–)

Yann LeCun
(1960–)

Turing Co-Awardees (2018)
Hinton: Nobel Laureate in Physics (2024)

Season 1 Highlights

Yoshua Bengio: Yes, I am scared.

Second TA Tutorial: Keras & CNN

🗓️ **Friday, 11/14, 12:00–1:00 PM**

📍 **Join via Zoom:** https://bit.ly/jhuaita25

Attendance is optional. Materials will be posted on Canvas

Until Next Class

- The **Group Assignment #2** is due one hour before Session 5
- Expect feedback for your AI Lab proposal posted on Canvas by Tuesday
- Refer to Syllabus for Readings for Session 5


Thank You!