Deep Learning

COMP 6721 Introduction of AI

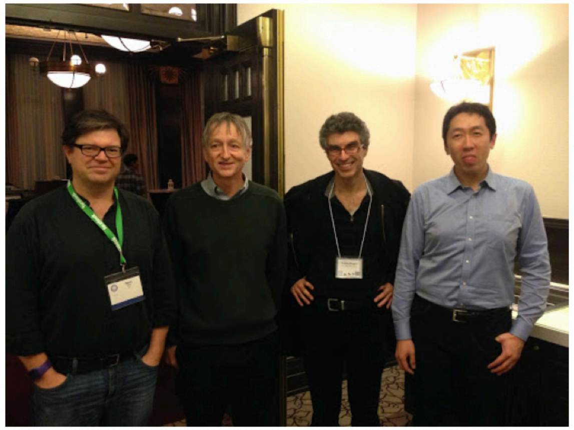
Russell & Norvig – Chapter 23.1 + 23.2 + 23.3

Deep Learning

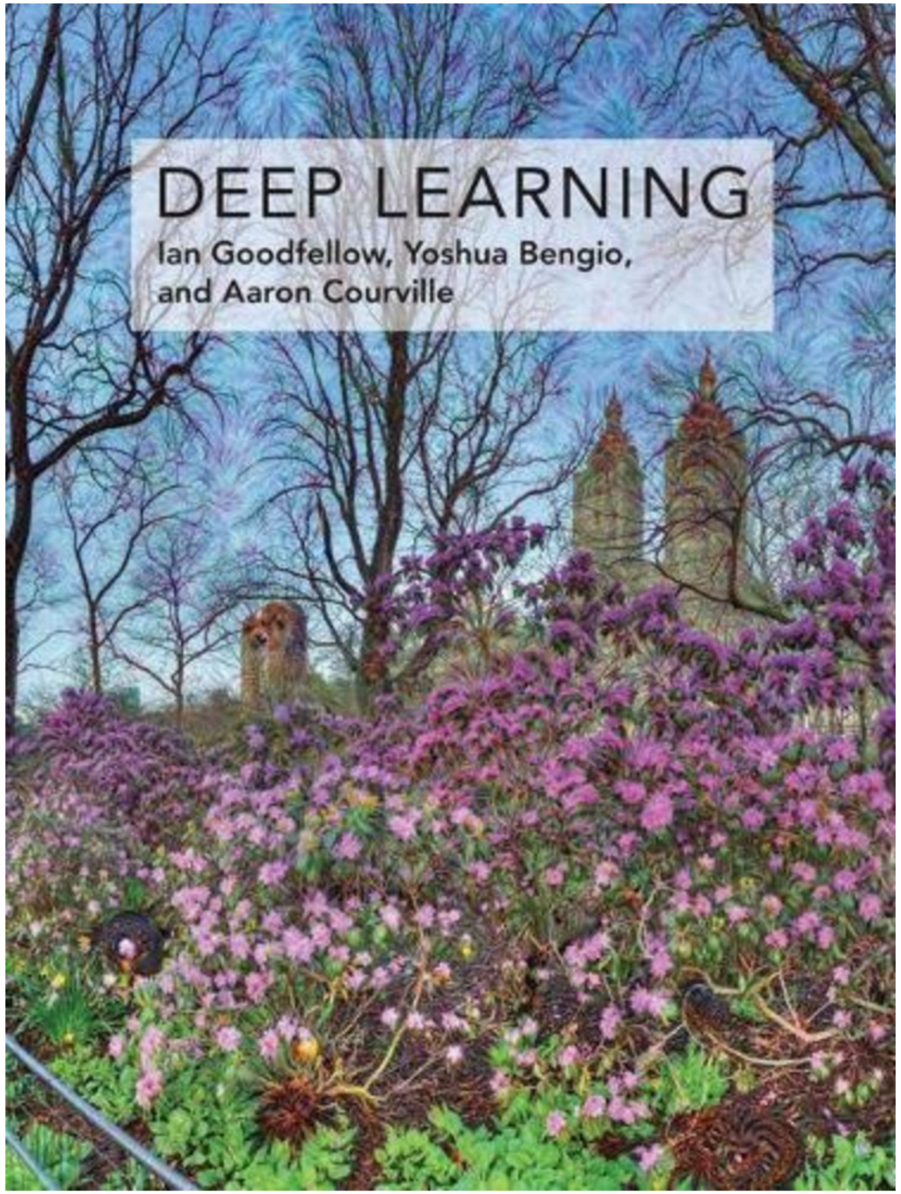
History of AI

2011 - Today

- Deep Learning
  - Development of "deep neural networks"
  - Trained on massive data sets
  - Use of GPU for computations
  - Use of "generic networks" for many applications
    - Image recognition
    - Self driving cars
    - Machine translation
    - Speech recognition & synthesis
    - Chatbots
    - Game playing
    - …



https://www.youtube.com/watch?v=l9RWTMNnvi4


DEEP LEARNING
Ian Goodfellow, Yoshua Bengio,
and Aaron Courville

Intuition

- Simple question: what is a chair?
- Definition (Merriam-Webster): a seat typically having four legs and a back for one person.
- How to code a program that recognizes chair in images?

Source: Wikimedia commons

History of ML

Rule-based
System

Example: What is a chair?

- Definition (Merriam-Webster): a seat typically having four legs and a back for one person.
- How to code a program that recognizes a chair in an image?
- Feature extraction:

  - Does it have four legs?

  - Does it have a back?

  - Can we seat on it?

Source: Wikimedia commons

Example: What is a chair?

- Definition (Merriam-Webster): a seat typically having four legs and a back for one person.
- How to code a program that recognizes a chair in an image?
- Feature extraction:

  - Does it have four legs?

  - Does it have a back?

  - Can we seat on it?

Source: Holger.Ellgaard - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=2815995

Example: What is a chair?

- Definition (Merriam-Webster): a seat typically having four legs and a back for one person.
- How to code a program that recognizes a chair in an image?
- Feature extraction:

  - Does it have four legs?

  - Does it have a back?

  - Can we seat on it?

Source: Sailko, Wikimedia commons

Example: What is a chair?

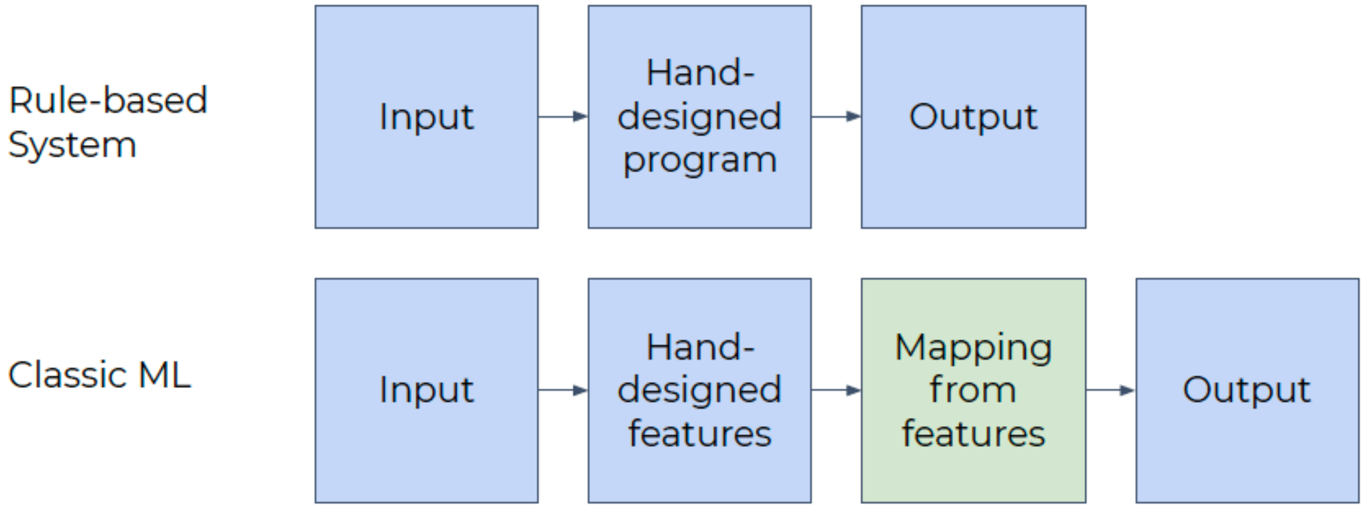
The features must be robust to the factors of variation of a chair.

Example: What is a chair?

The features must be robust to the factors of variation of a chair and its surroundings.

Source: Kelly Miller, Unsplash

History of ML



How to build these features?

- Pattern Recognition

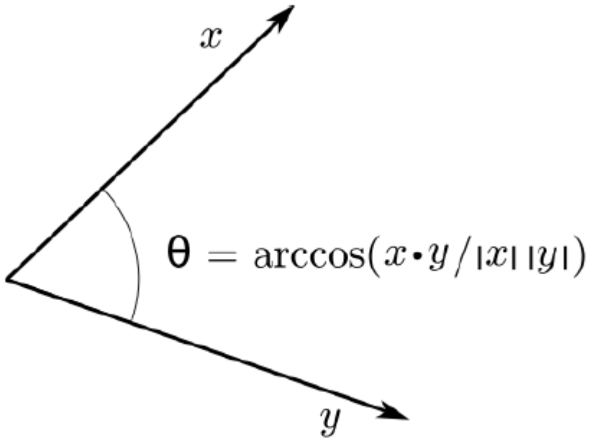
We know this is a chair.

Is it a chair?

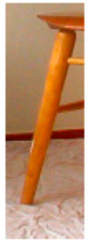
Source: Kelly Miller, Unsplash

Pattern Recognition

- How to code if a pattern is in an image? With **scalar product**.
- Example: dot([1, 2, 3], [3, 2, 1]) = sum([3, 4, 3]) = 10



Source: Wikipedia



```
array([[[  0,    0,    0, 255],
        [  0,    0,    0, 255],
        [  0,    0,    0, 255],
        ...
        [  0,    0,    0, 255],
        [  0,    0,    0, 255],
        [132, 132, 132, 255]]], dtype=uint8)
```

```
array([[255, 255, 255, ..., 255, 255, 255],
       [255, 255, 255, ..., 255, 255, 255],
       [253, 252, 252, ..., 252, 253, 255],
       ...,
       [224, 202, 201, ..., 198, 223, 255],
       [223, 203, 203, ..., 135, 187, 255],
       [228, 210, 212, ..., 180, 213, 255]], dtype=uint8)
```

How to build these features?

- Pattern Recognition

Is the pattern here? No (0.1)

How to build these features?

- Pattern Recognition

Is the pattern here? Maybe (0.5)

How to build these features?

- Pattern Recognition

Is the pattern here? Probably (0.75)

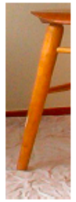
Pattern Recognition

- We compare many patterns with the data.
- Each comparison has a score indicating if the pattern matches the data.
- These scores are the new features representing the data.

Source: Kelly Miller, Unsplash

Pattern Recognition

- Why did we use only a small part of the original image as a pattern?



Pattern Recognition

- Why did we use only a small part of the original image as a pattern?
- Because sub-parts are more "simple" than the whole and so, it correlates more easily.

Pattern Recognition

- Compositionality principle:
- The whole is made only from its parts.
- The compositionality principle can apply to multiple scales.

Source: Alphacolor, Unsplash

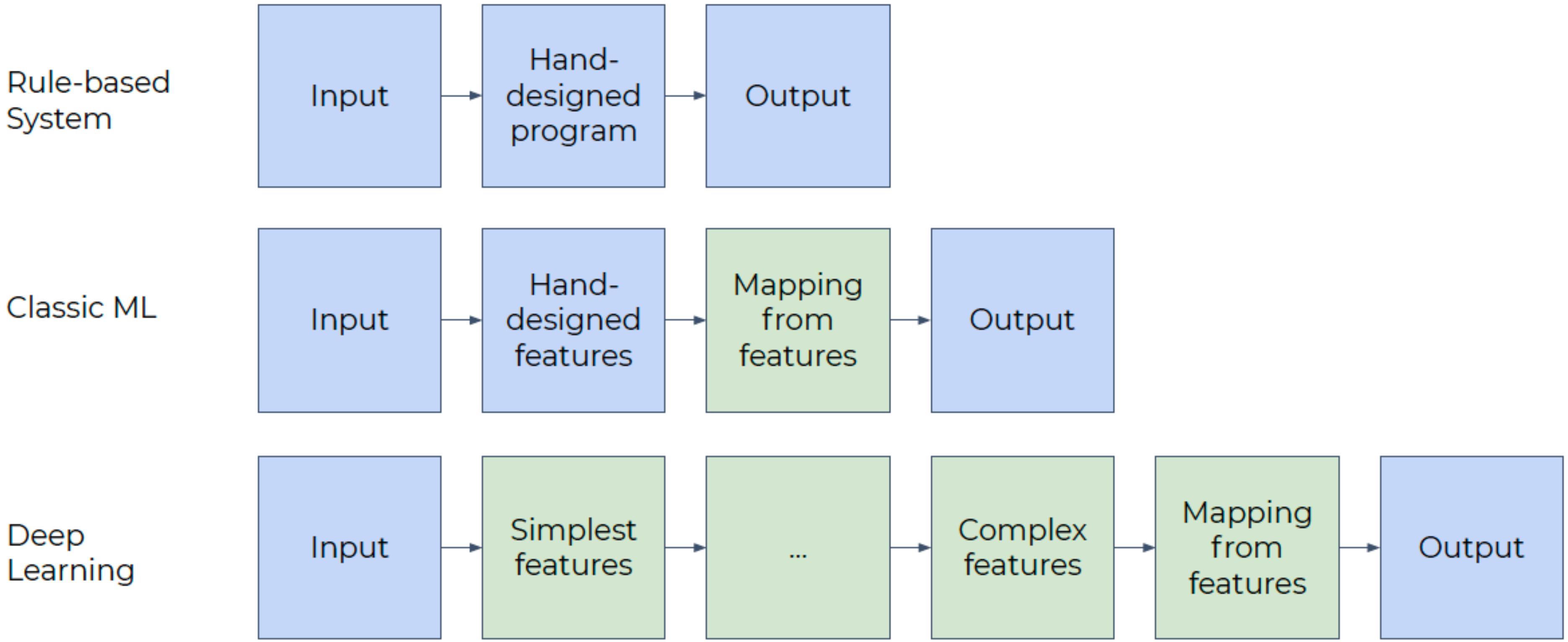
Deep Learning

Can we learn automatically the patterns at different scales with supervised learning?

Yes, this is **deep learning**!

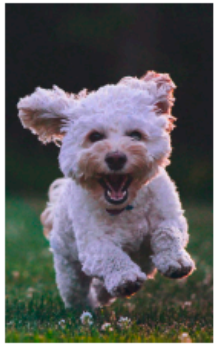
Source: Alphacolor, Unsplash

History

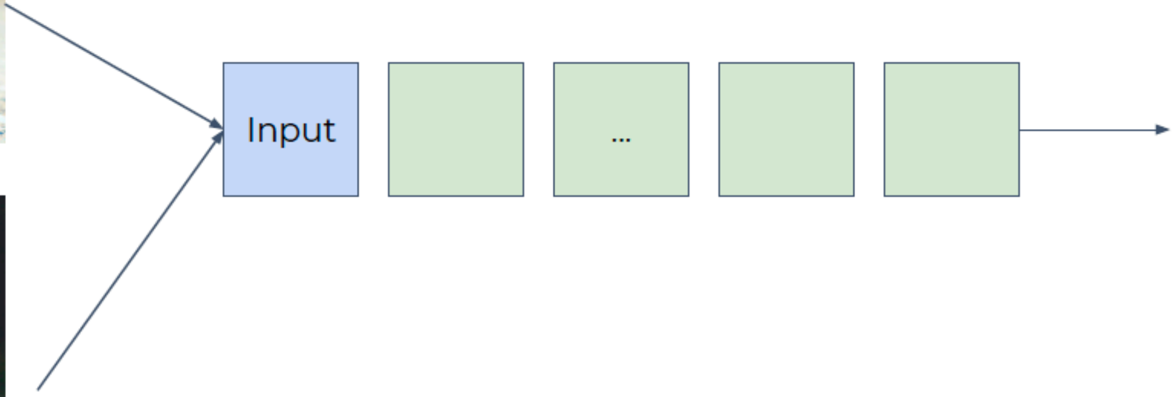

Rule-based System
Input
Hand-designed program
Output
Classic ML
Input
Hand-designed features
Mapping from features
Output
Deep Learning
Input
Simplest features
...
Complex features
Mapping from features
Output

What are good representations?

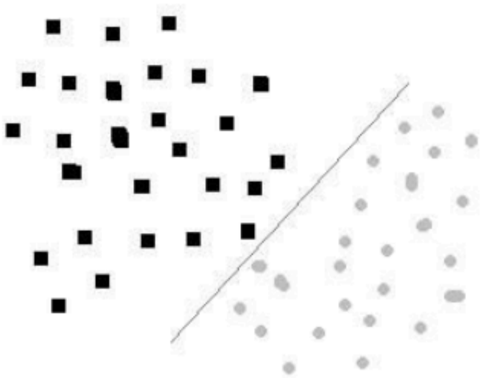
Source: Edgar Edgar, Unsplash



Source: Joe Caione, Unsplash



Source: Wikimedia, commons



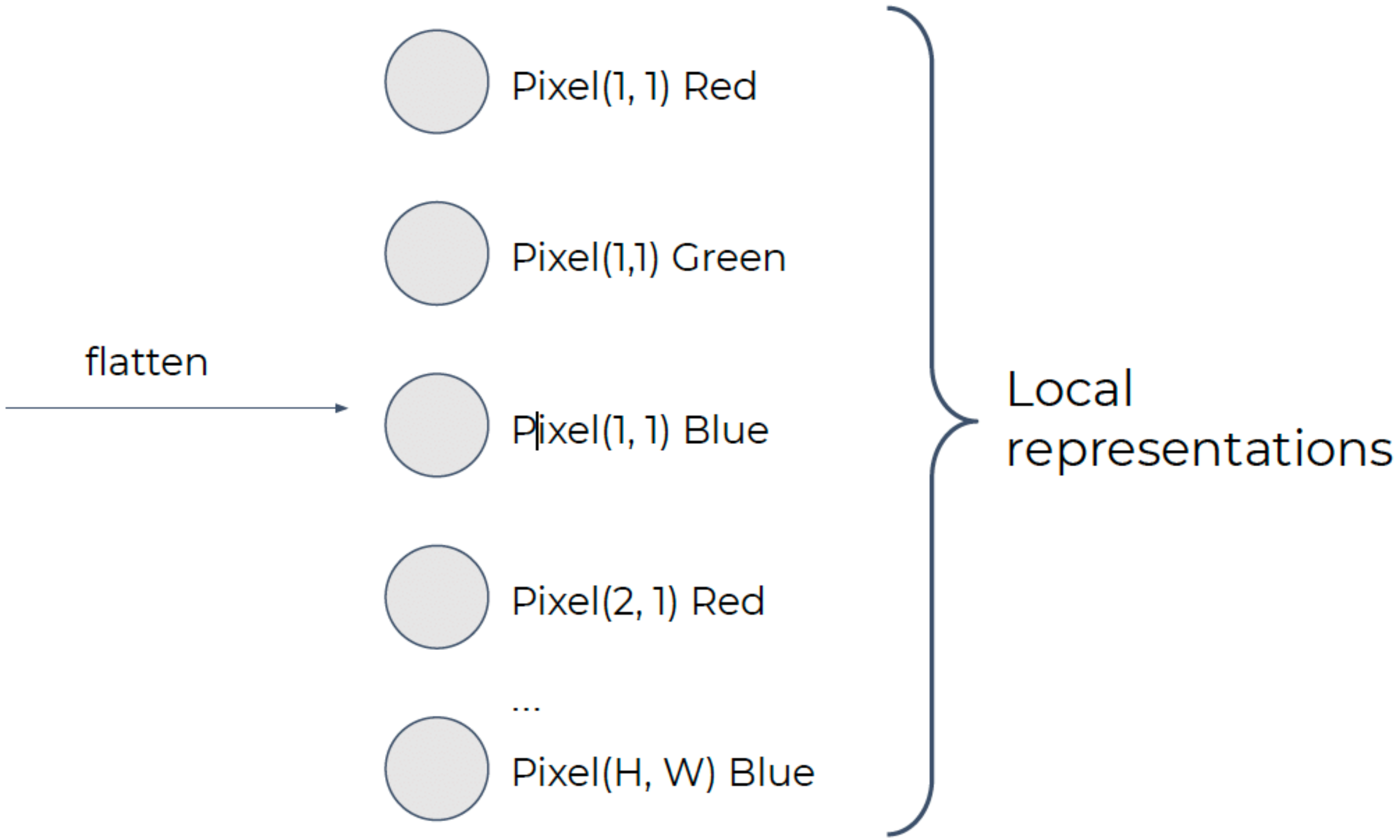
Representations that are linearly separable

Topics

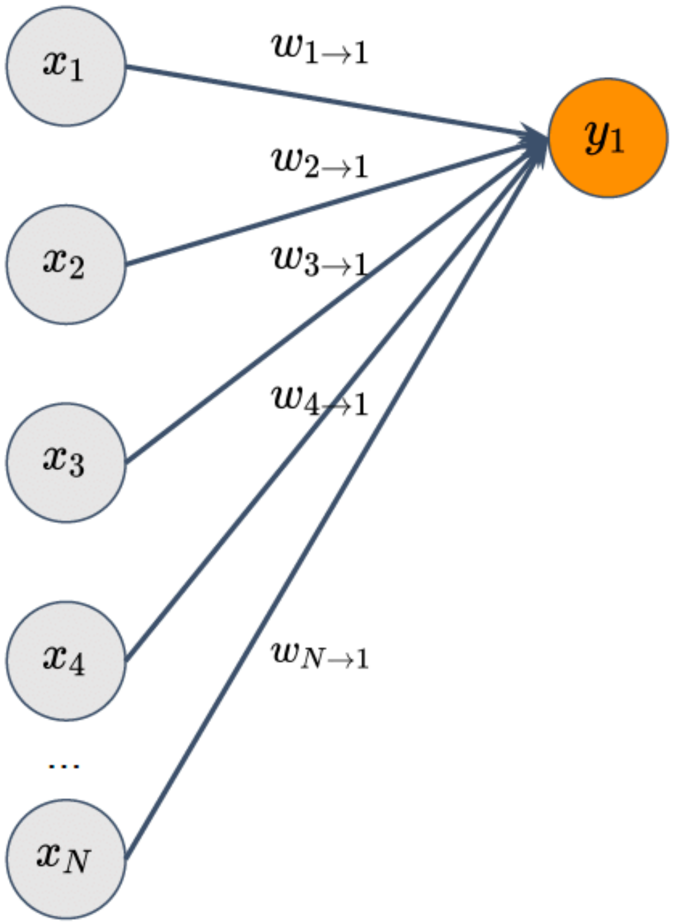
- Modular Approach

Classical diagram

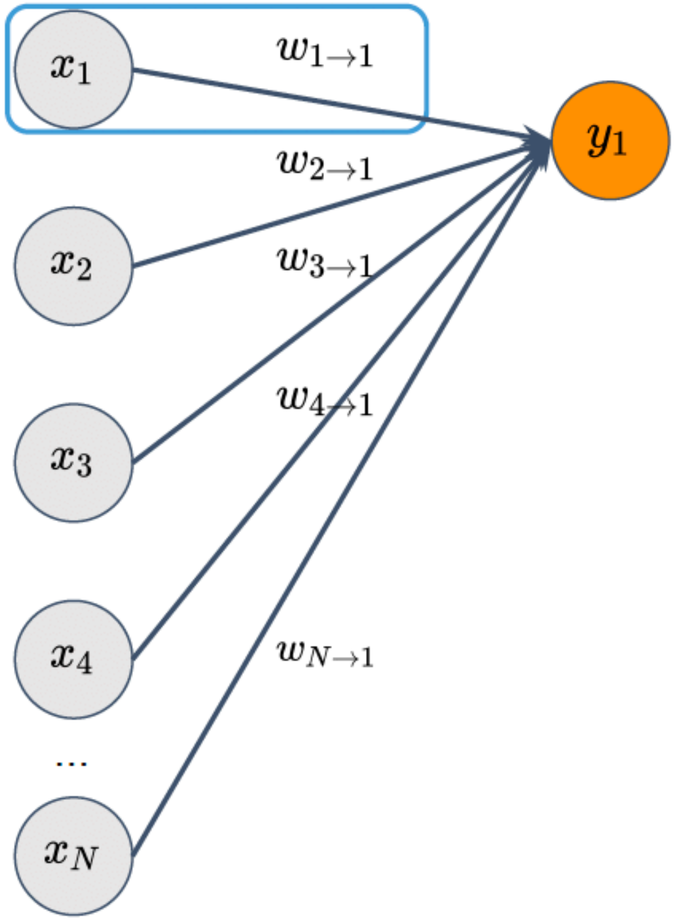
Source: Kelly Miller, Unsplash

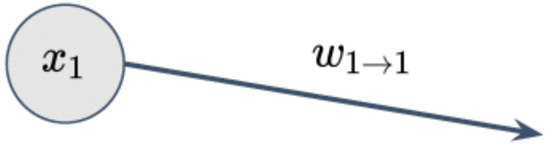


Classical diagram



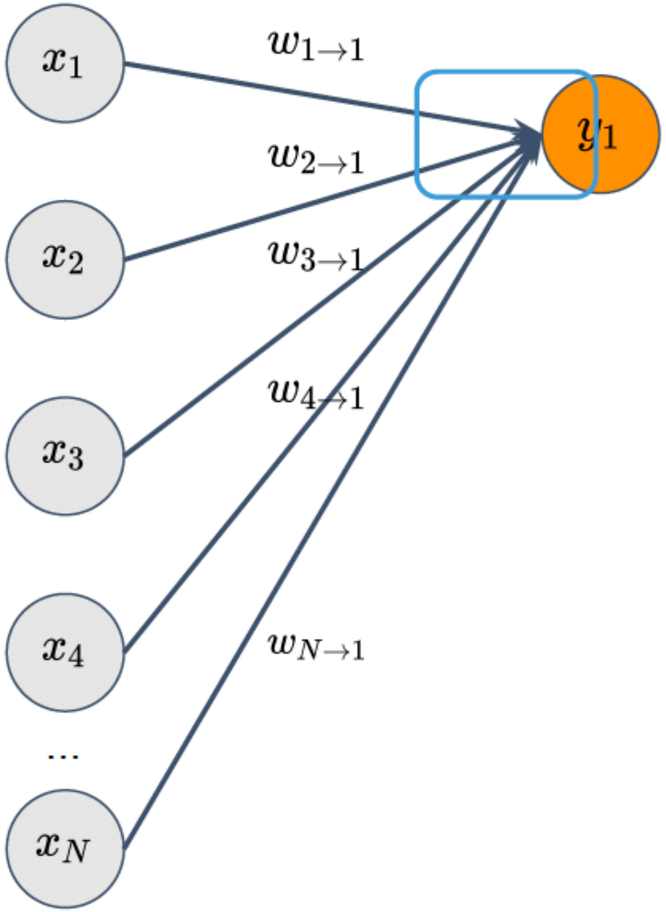
Classical diagram

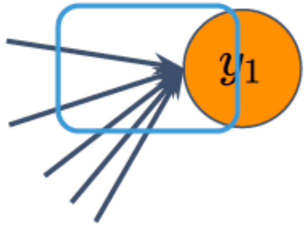




Product: $x_1 w_{1\to1}$

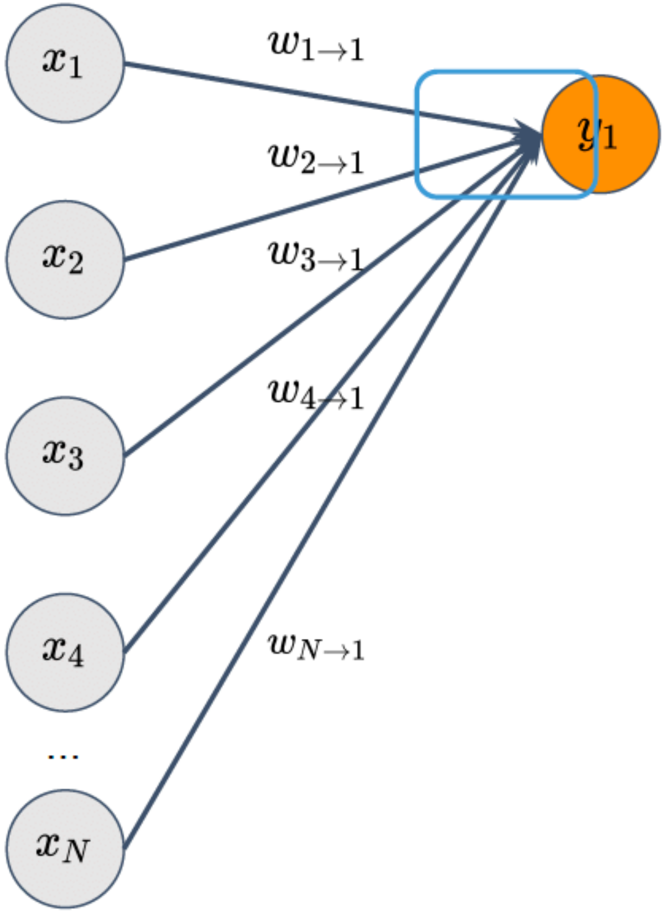
Classical diagram

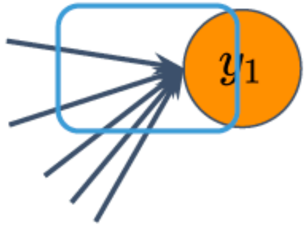




$$\overbrace{\sum_{i=1}^{N}}^{\text{Sum}} x_i w_{i\to1}$$

Classical diagram: pattern extractor

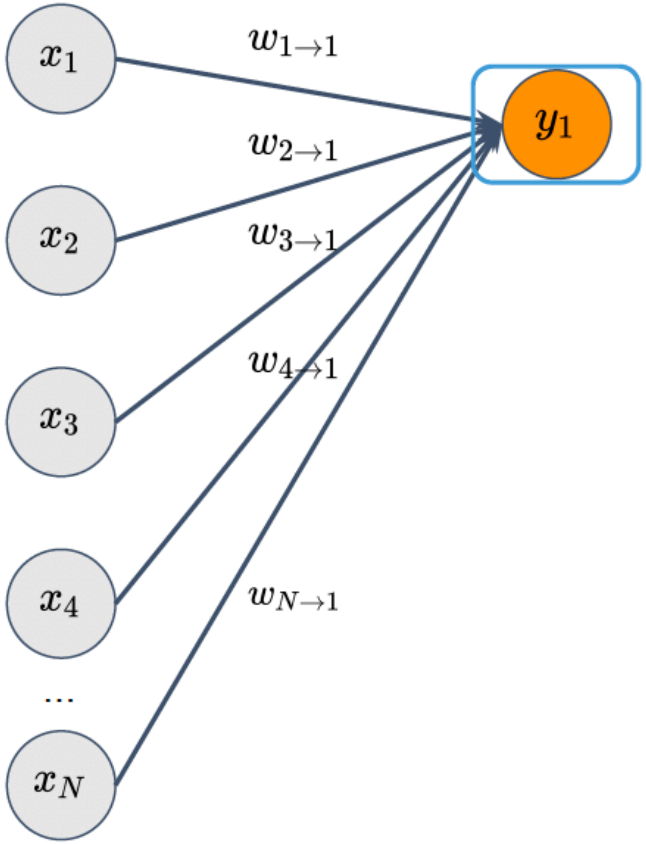




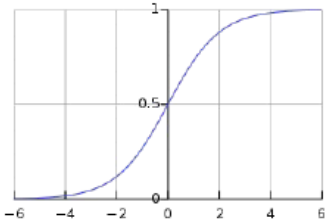
$$\underbrace{\overbrace{\sum_{i=1}^{N}}^{\text{Sum}} x_i w_{i\to1}}_{\text{Scalar product}}$$

$$\langle x, w \rangle$$

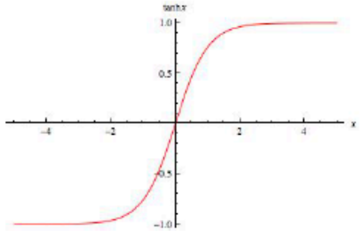
Classical diagram: Activation function



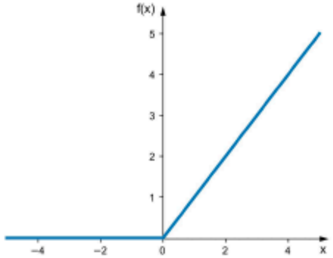
Activation function



Sigmoid



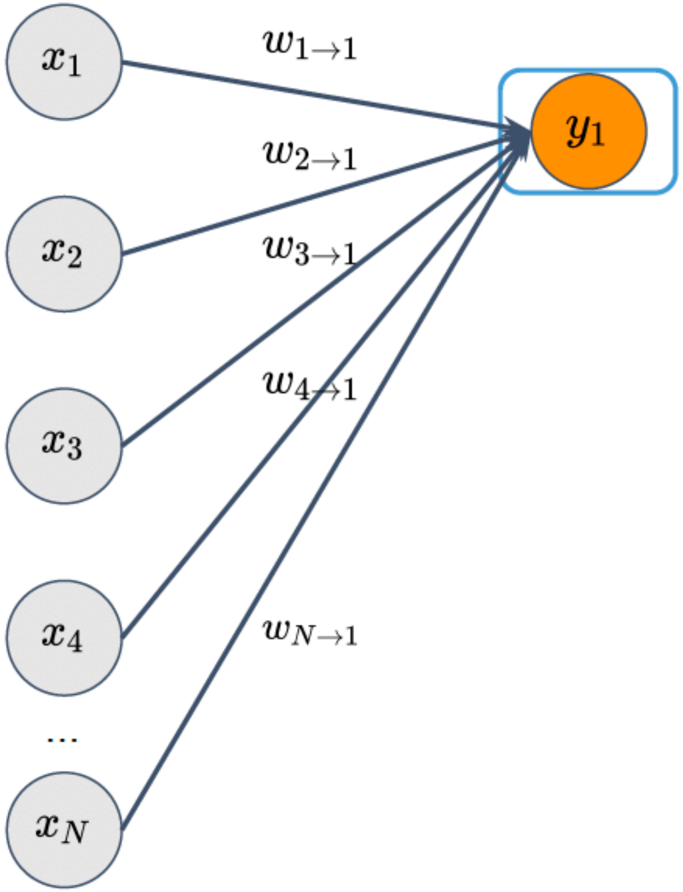
Tanh



ReLU

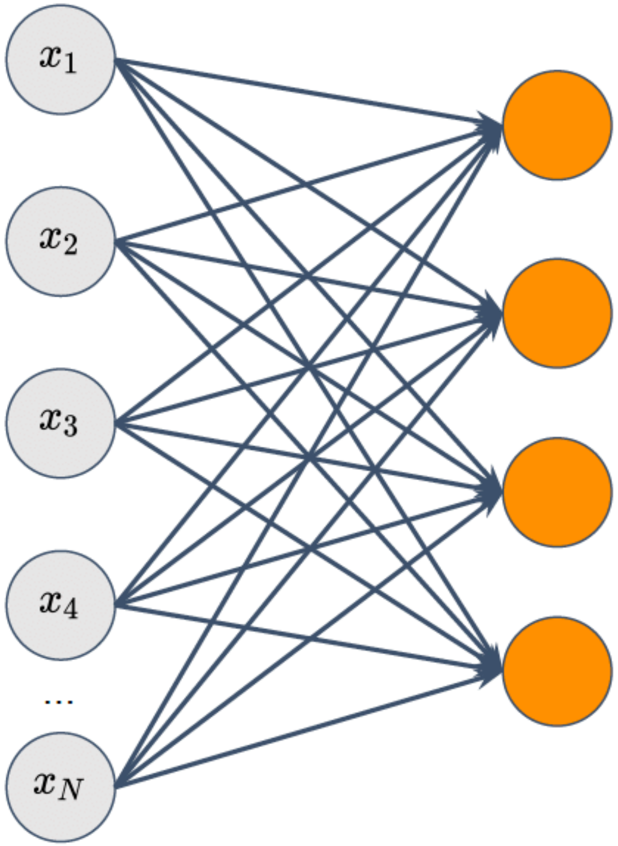
...

Classical diagram: Artificial Neuron



$$y_1 = \sigma\left(\sum_{i=1}^{N} x_i w_{i\to 1}\right)$$

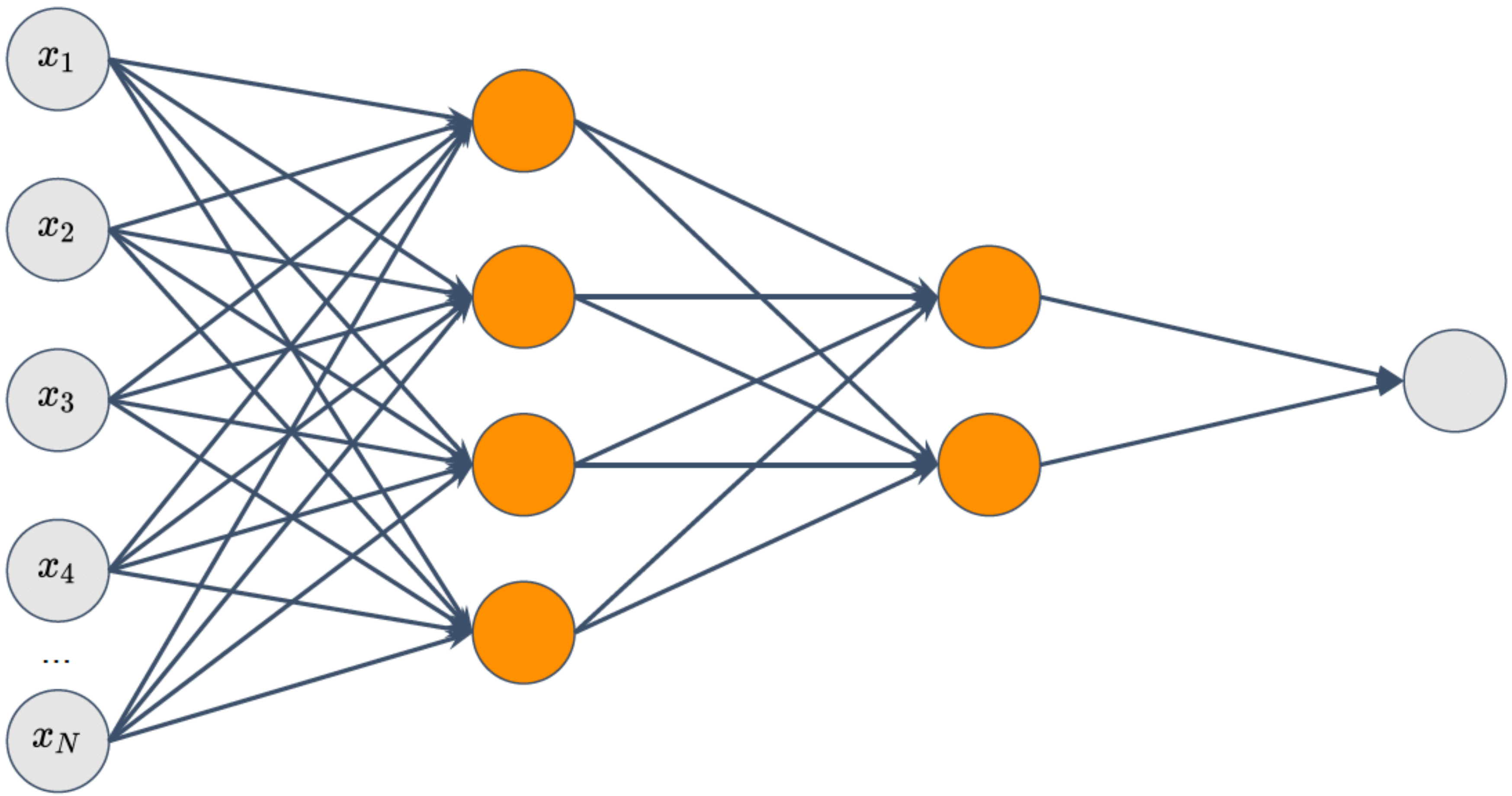
Classical diagram: hidden layer



Multiple
pattern
extractors

$$y_j = \sigma\left(\sum_{i=1}^{N} x_i w_{i \to j}\right)$$

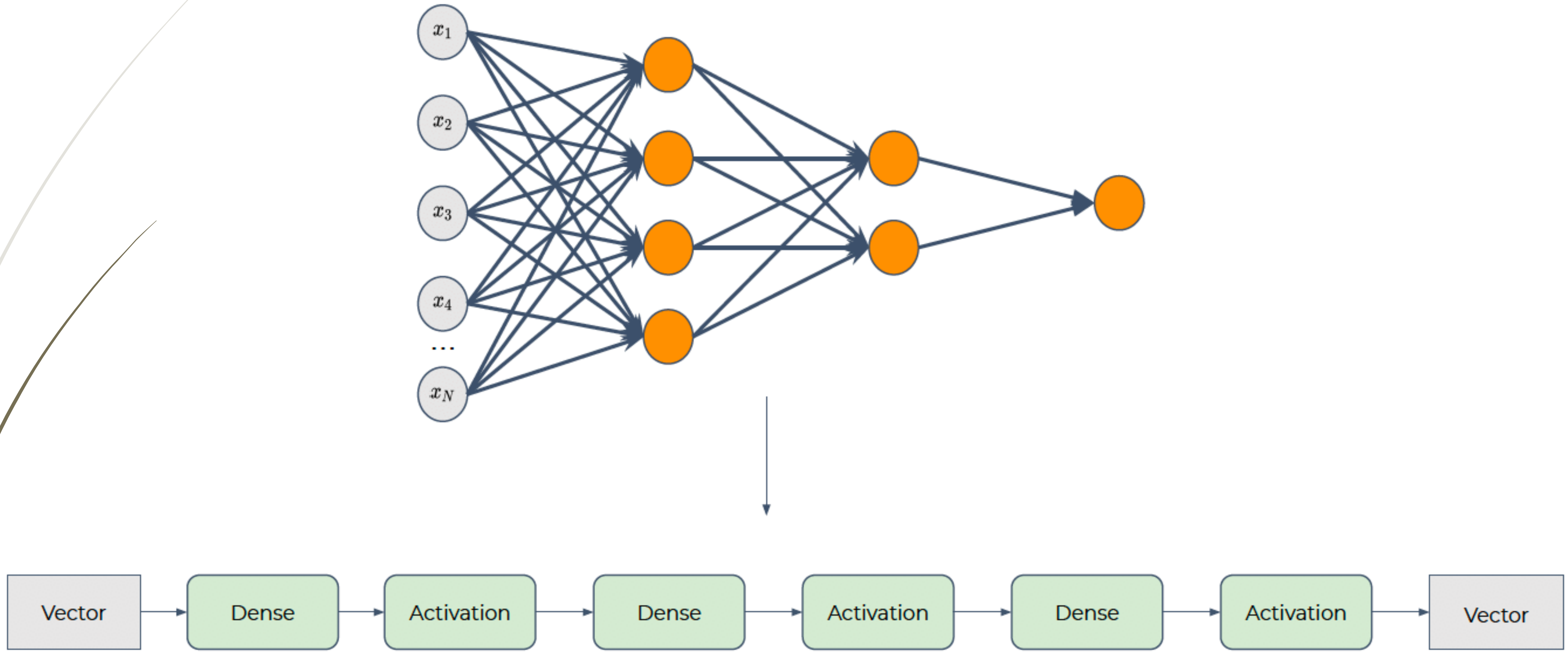
Classical diagram: multilayer perceptron



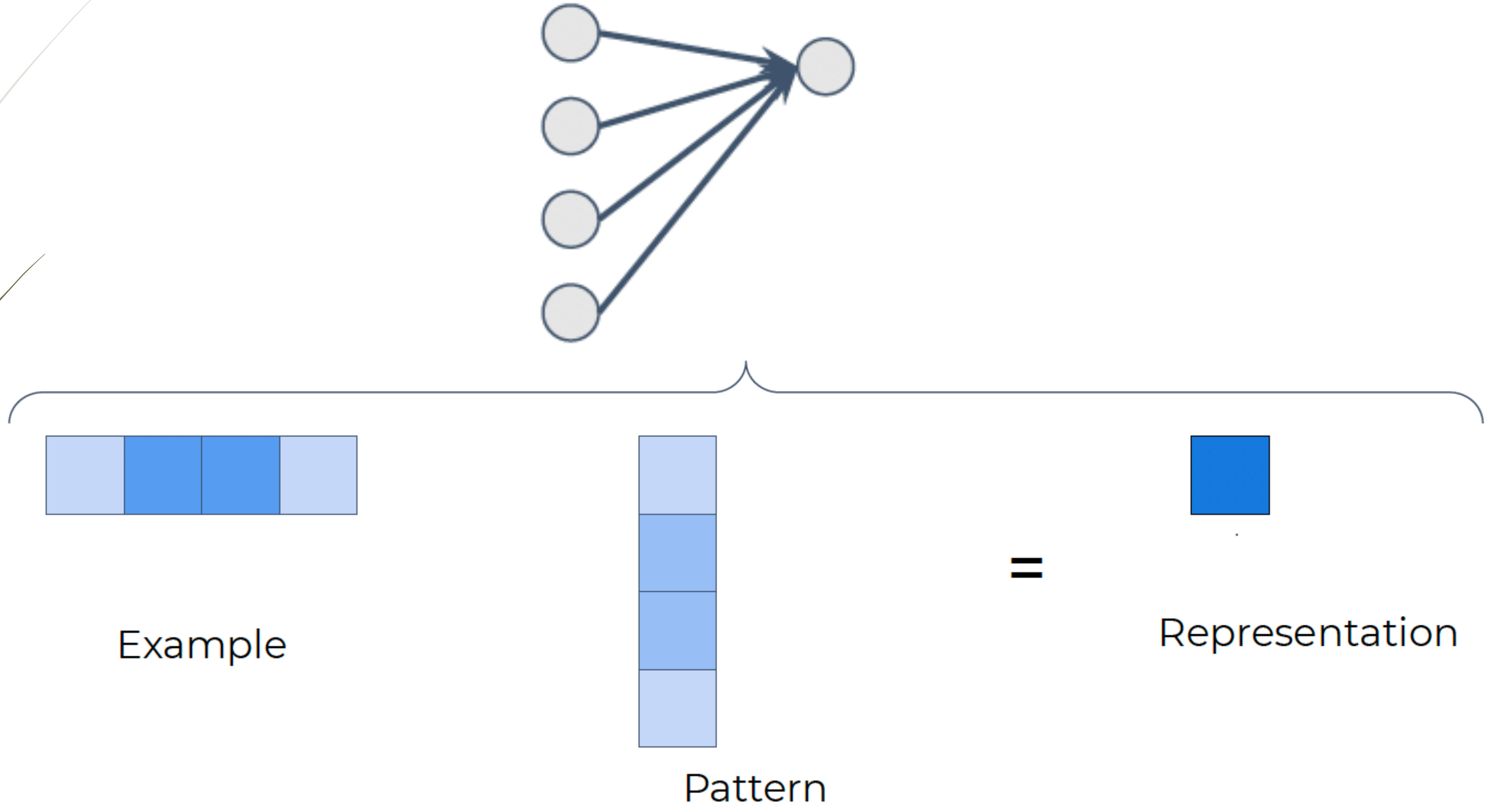
Why we need activation function?

- Composing many linear transformations is equivalent to a single linear transformation.
- Theoretical results (e.g. universal approximation theorem) about the representation power of deep neural networks with activation functions.
- Activation functions can modulate the representations.

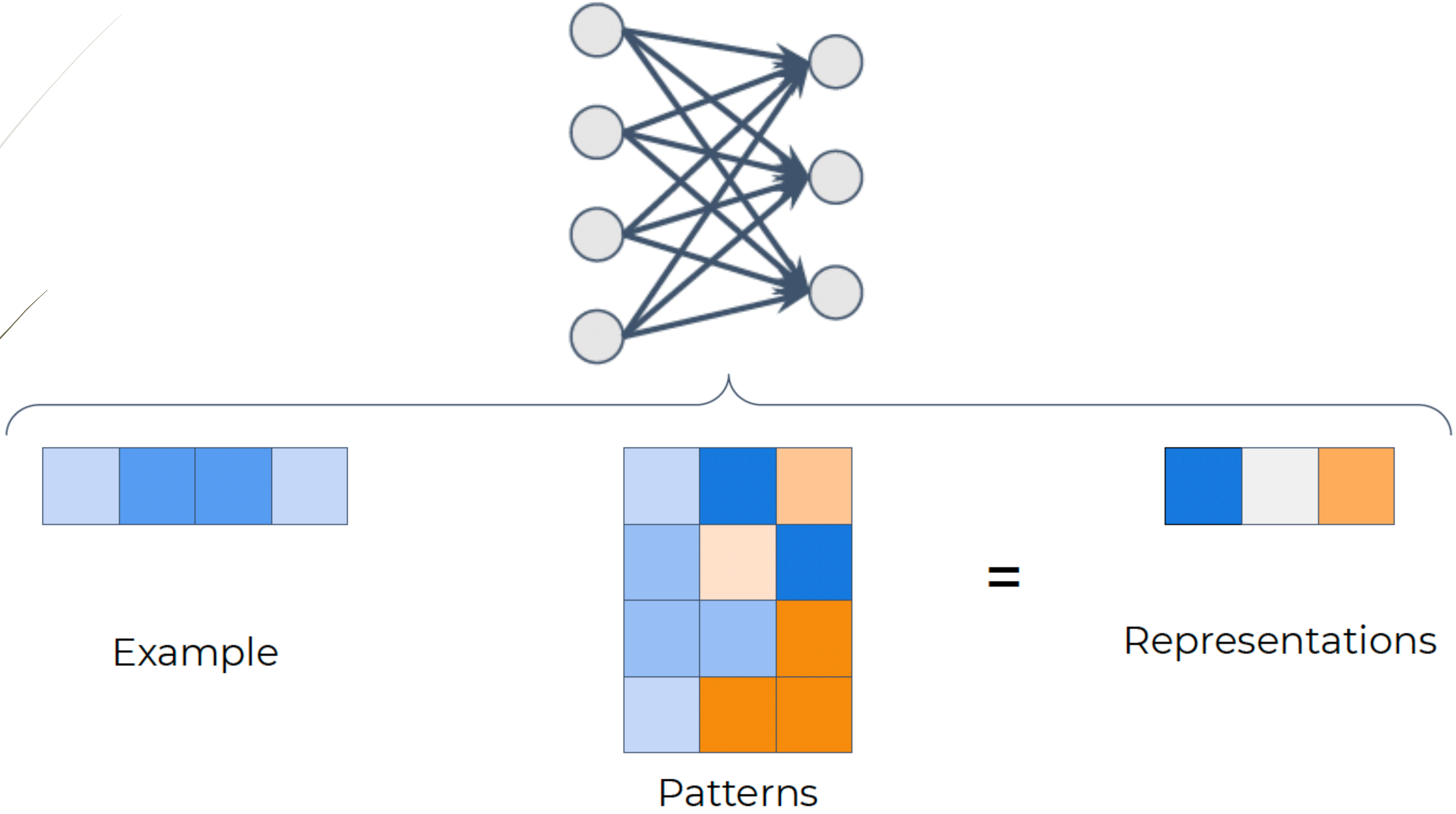
Simplified diagram with modules



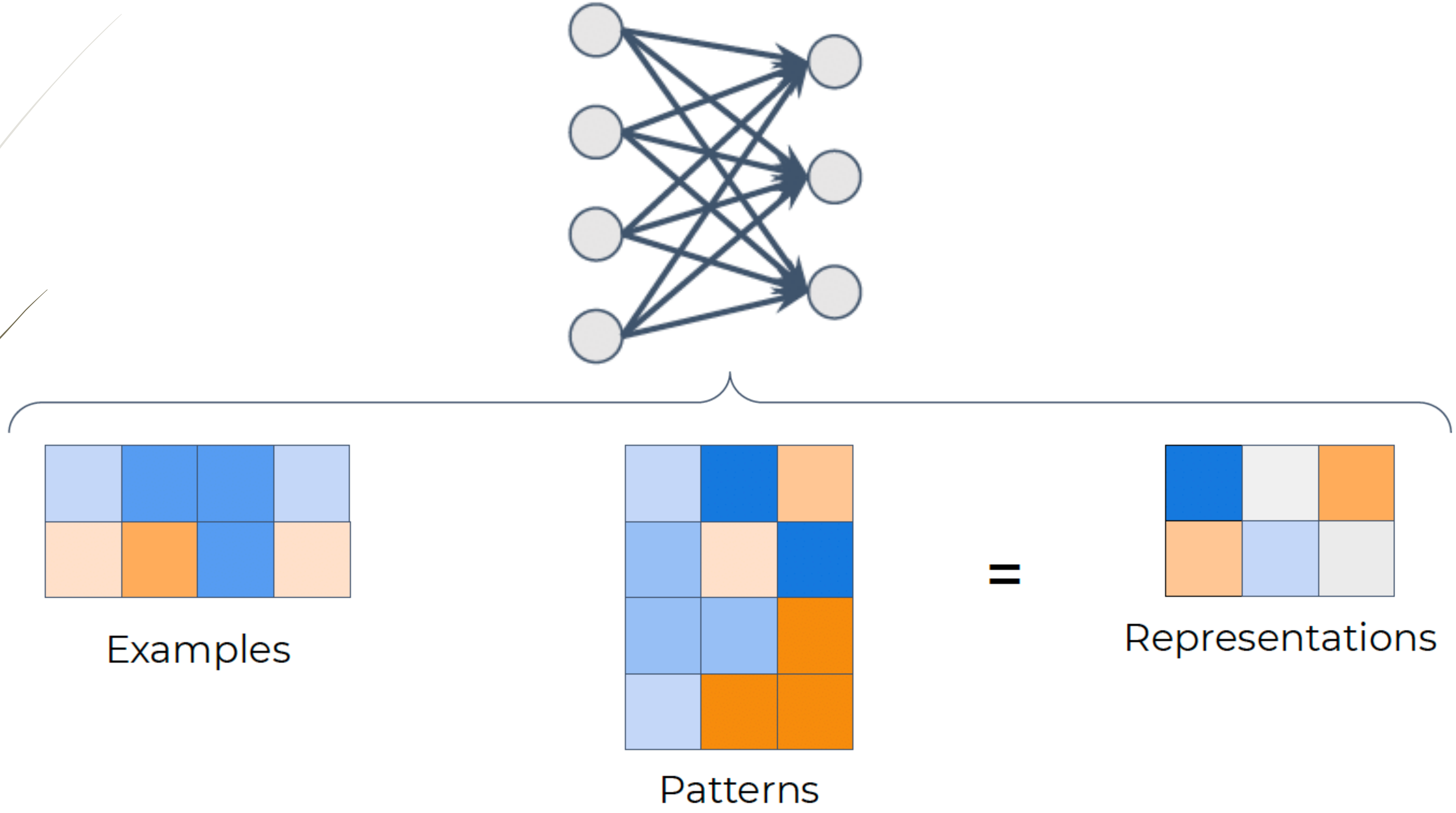
Dense Module: vector-vector multiplication



Dense Module: matrix-vector multiplication



Dense Module: matrix-vector multiplication



Advantages over classical diagrams

- Data and patterns are stocked in tensors (vector, matrix, N-d array).
- Deep learning operations are encapsulated in modules.
- A module can process examples in parallel.
- A module can be implemented with linear algebra operations, which are:
  - efficiently implemented in existing libraries,
  - efficiently executed on GPUs.
- We can use a modular approach to define architectures.

The End