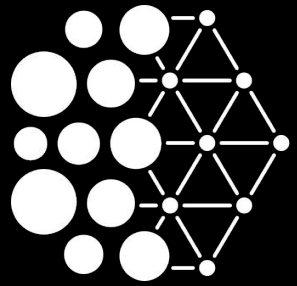
Quebec
Artificial
Intelligence
Institute

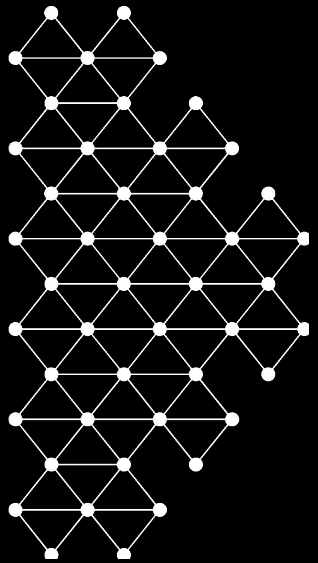


Mila

Introduction to Deep Learning

Gaétan Marceau Caron
Applied research scientist, Mila
gaetan.marceau.caron@mila.quebec

Quebec
Artificial
Intelligence
Institute



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?

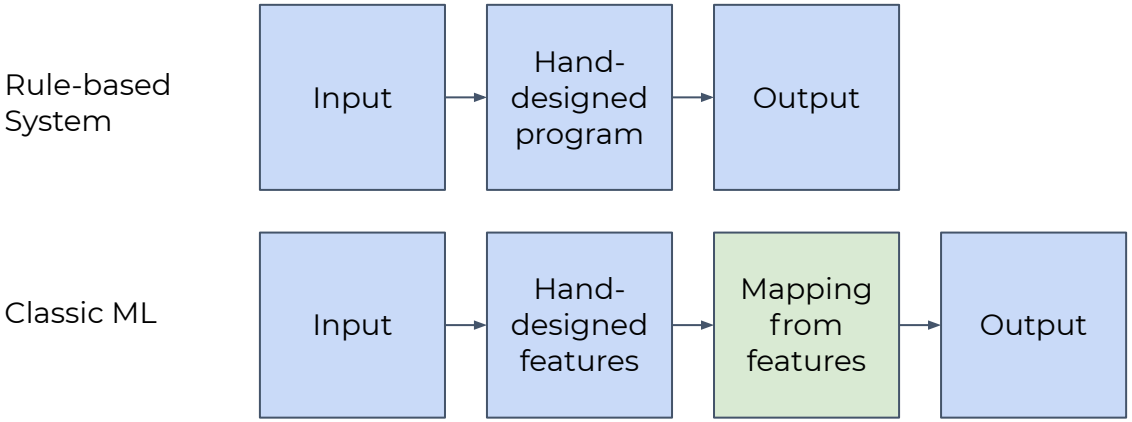
The features must be robust to the factors of variation of a chair.

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


Rule-based System
Input
Hand-designed program
Output
Classic ML
Input
Hand-designed features
Mapping from features
Output

How to build these features?
Pattern Recognition

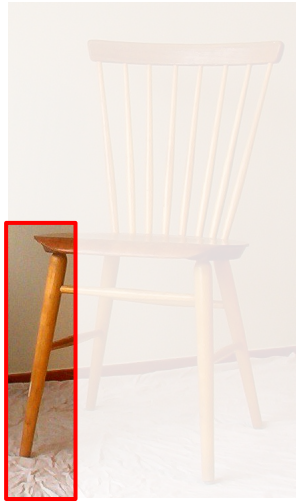
We know this is a chair.

Is it a chair?

Source: Kelly Miller, Unsplash

How to build these features?
Pattern Recognition

We extract a pattern.

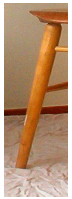


Is it a chair?

Source: Kelly Miller, Unsplash

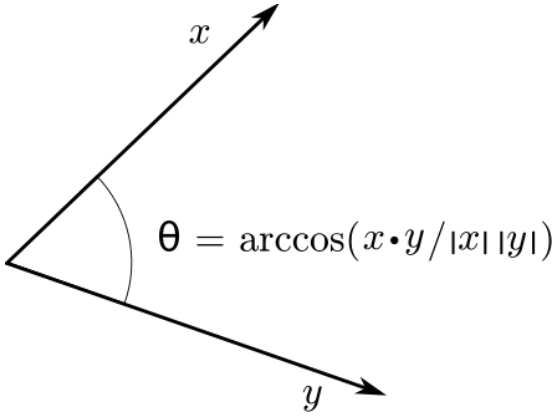
Pattern recognition

How to code if a pattern is in an image? **With scalar product**.

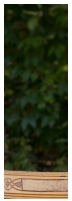


```
array([[[  0,   0,   0, 255],
        [  0,   0,   0, 255],
        [  0,   0,   0, 255],
        ...
        [  0,   0,   0, 255],
        [  0,   0,   0, 255],
        [132, 132, 132, 255]]], dtype=uint8)
```

Example: dot([1, 2, 3], [3, 2, 1]) = sum([3, 4, 3]) = 10



Source: Wikipedia



```
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)

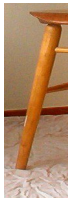
Pattern recognition

- 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

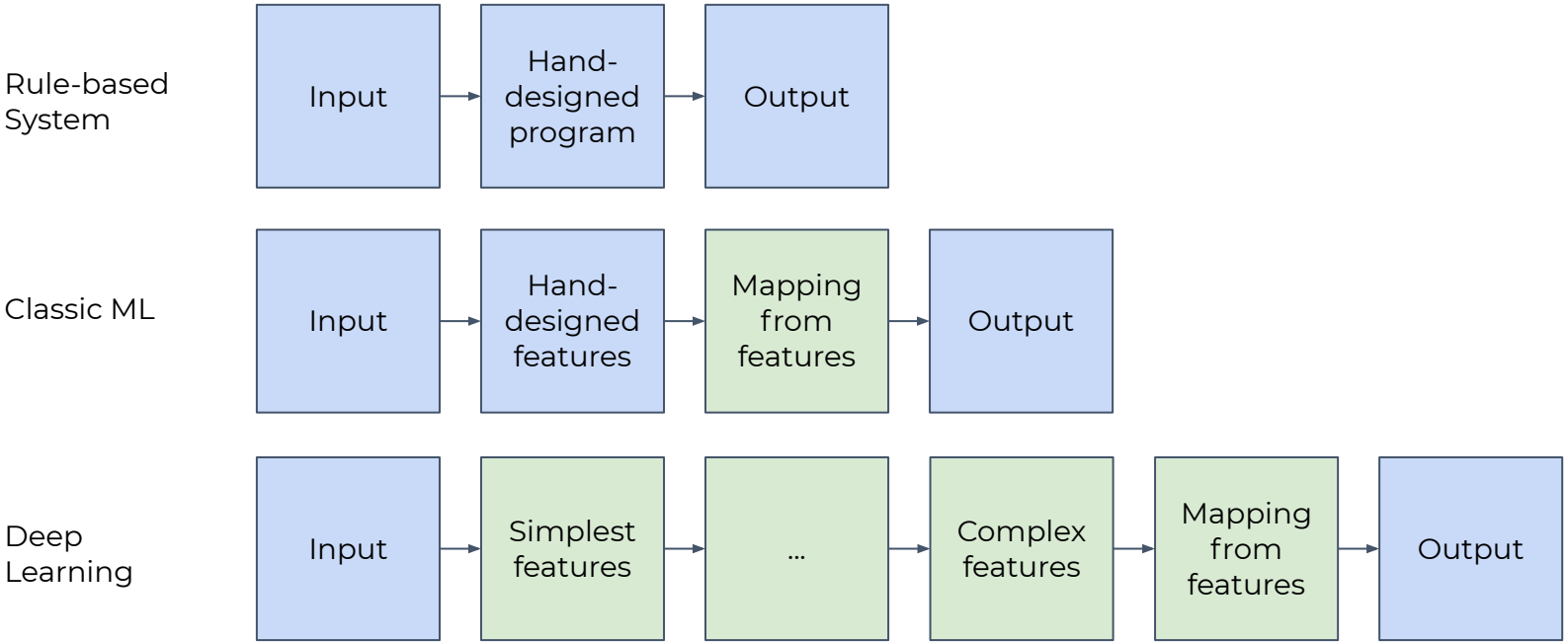
Deep learning

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

Yes, this is deep learning!

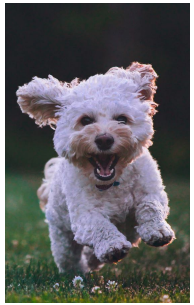
Source: Alphacolor, Unsplash

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

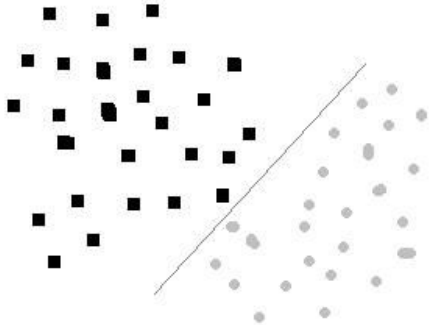
What are good representations?

Source: Edgar Edgar, Unsplash



Source: Joe Caione, Unsplash

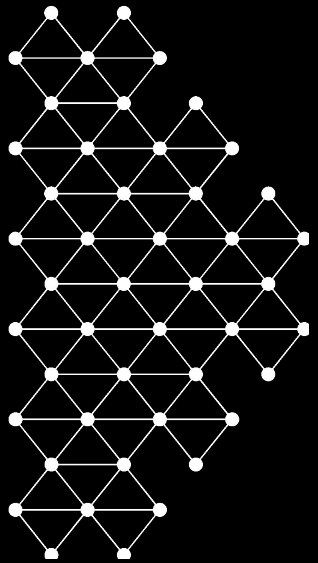
Source: Wikimedia, commons



Representations that are linearly separable

Quebec
Artificial
Intelligence
Institute

Mila



Modular approach

Classical diagram

Source: Kelly Miller, Unsplash

flatten

Pixel(1, 1) Red

Pixel(1,1) Green

Pixel(1, 1) Blue

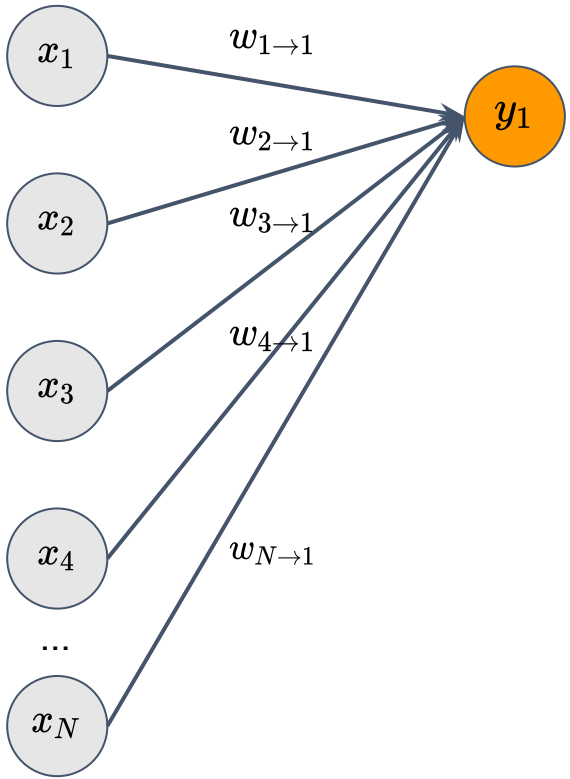
Pixel(2, 1) Red

...

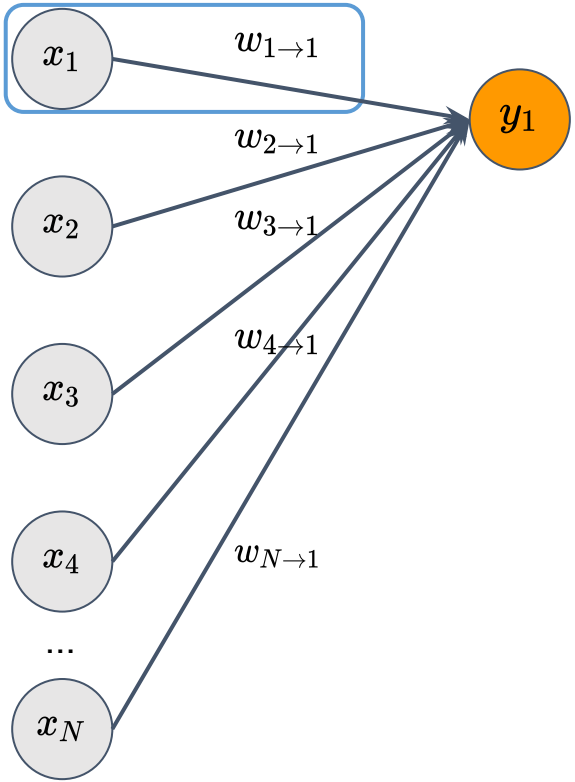
Pixel(H, W) Blue

Local representations

Classical diagram

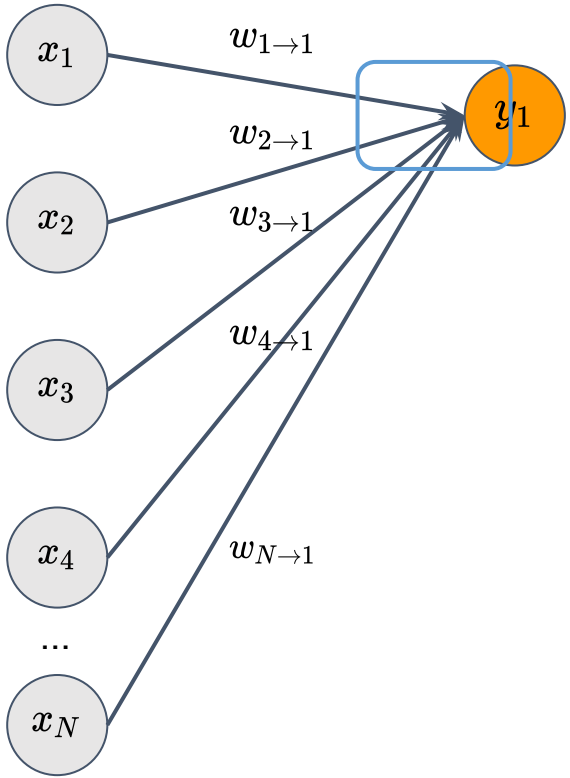


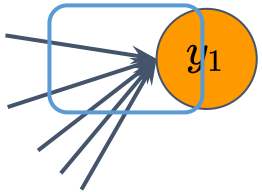
Classical diagram



Product: $x_1 w_{1\to1}$

Classical diagram

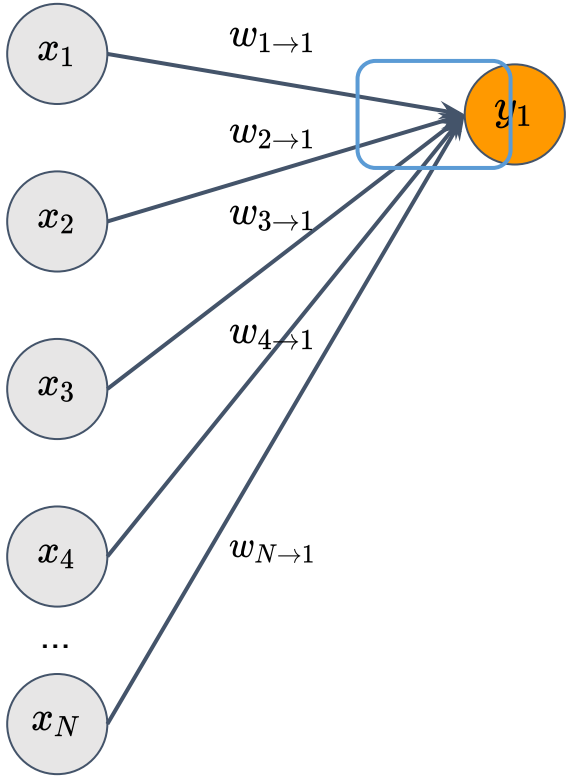


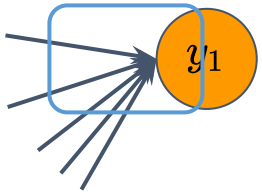


Sum

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

Classical diagram: pattern extractor





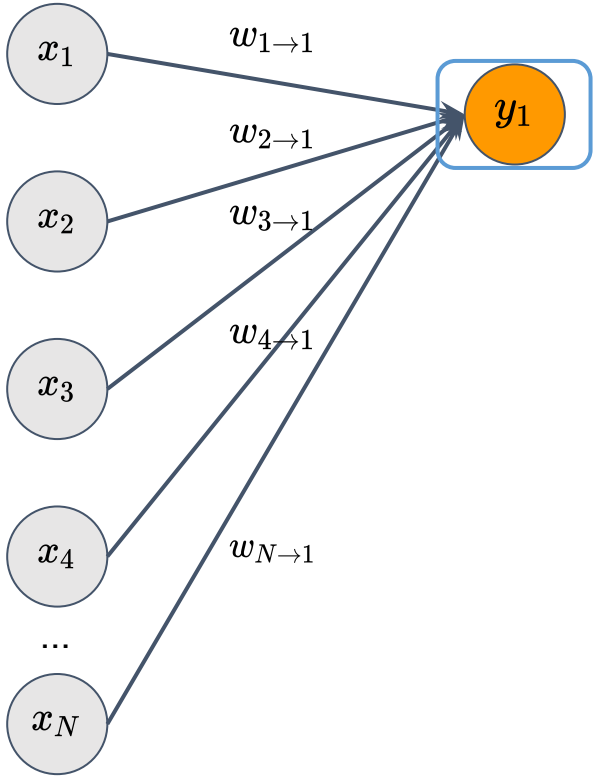
Sum

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

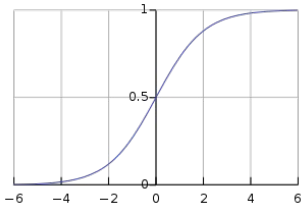
Scalar product

$$\langle x, w \rangle$$

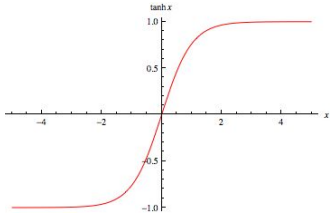
Classical diagram: activation function



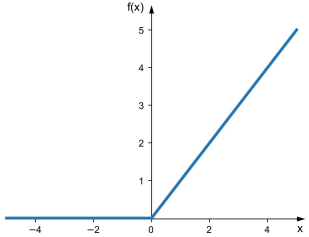
Activation function



Sigmoid



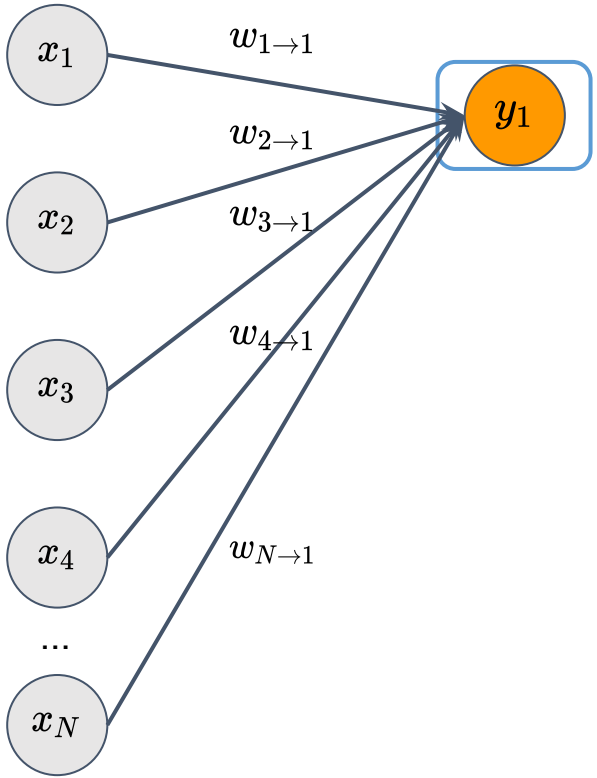
Tanh



ReLU

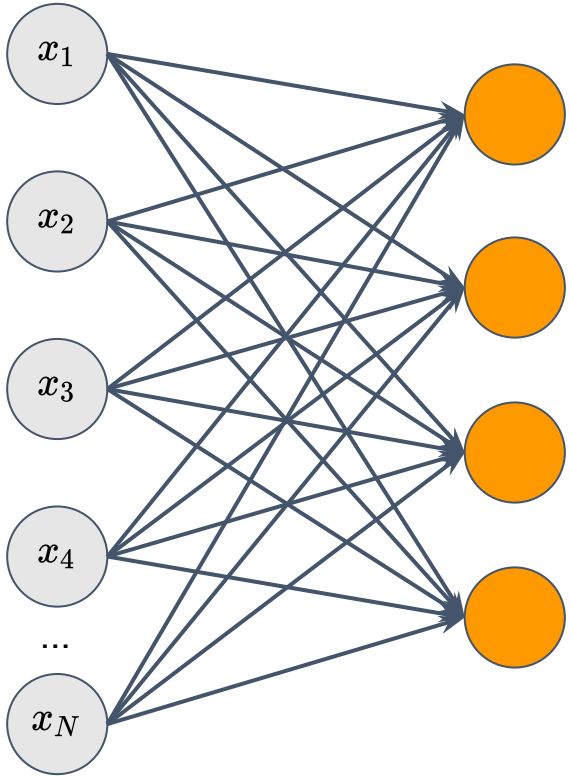
...

Classical diagram: artificial neuron



$$y_1 = \sigma\left(\sum_{i=1}^{N} x_i w_{i\to1}\right)$$

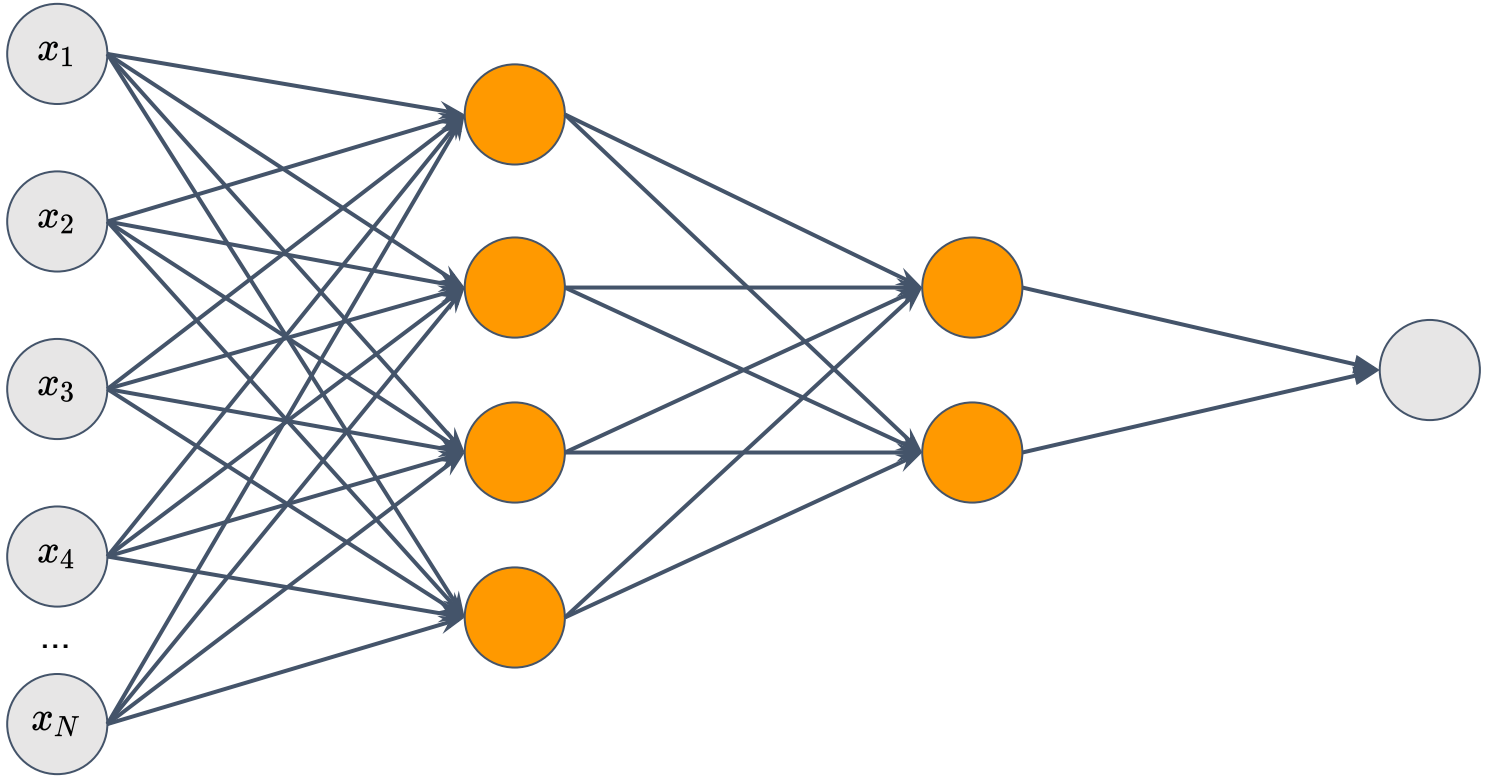
Classical diagram: hidden layer



Multiple pattern extractors

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

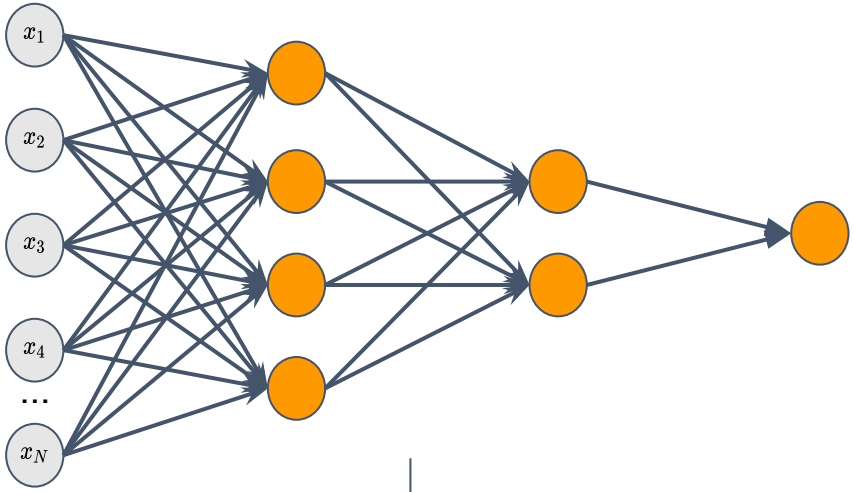
Classical diagram: multilayer perceptron

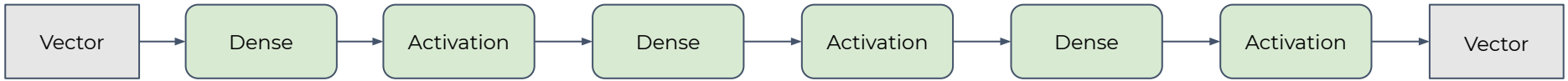


Why do we need activation functions?

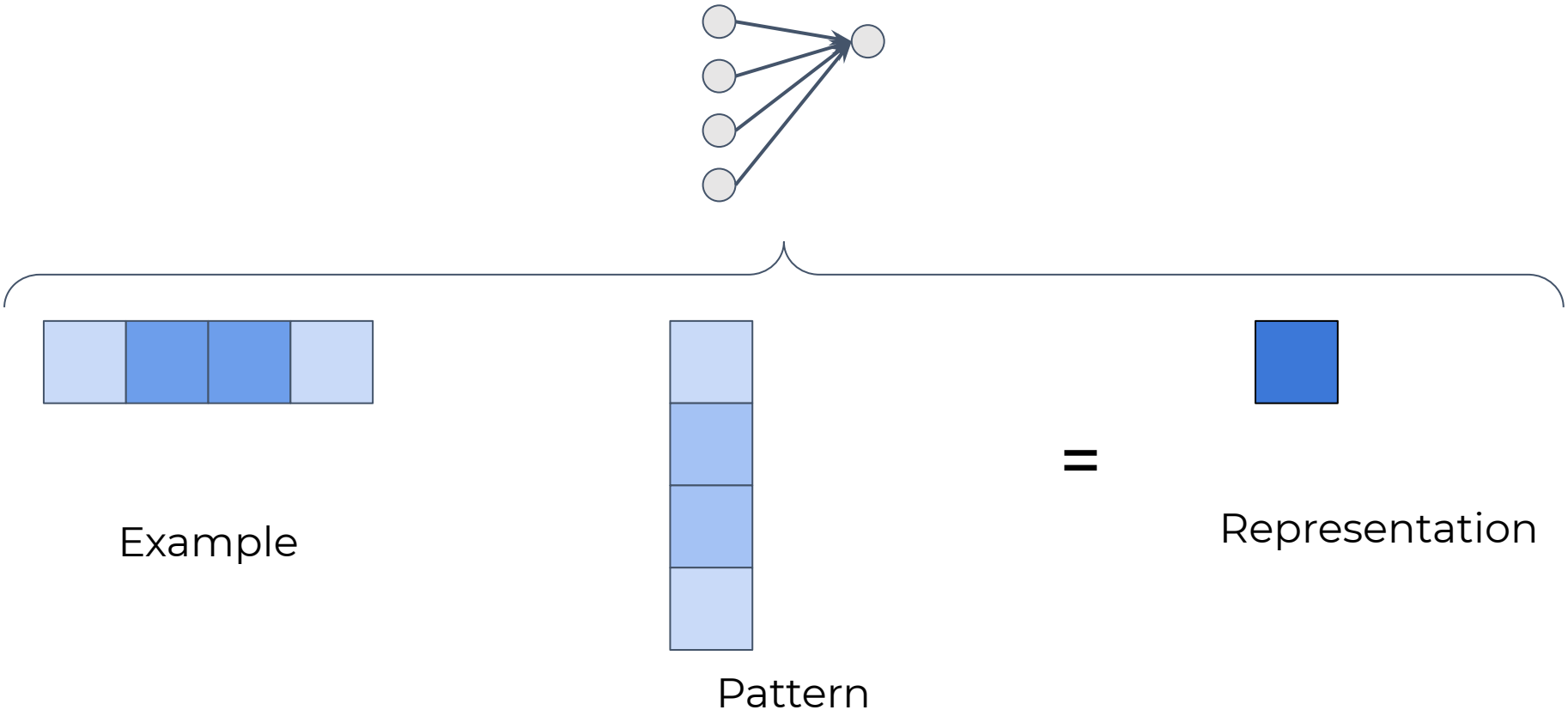
- Composing many linear transformations is equivalent to a single linear transformation.
- Theoretical results (e.g. universal approximation theorem) about the representation power of deep neural networks with activation functions.
- Activation functions can modulate the representations.

Simplified diagram with modules

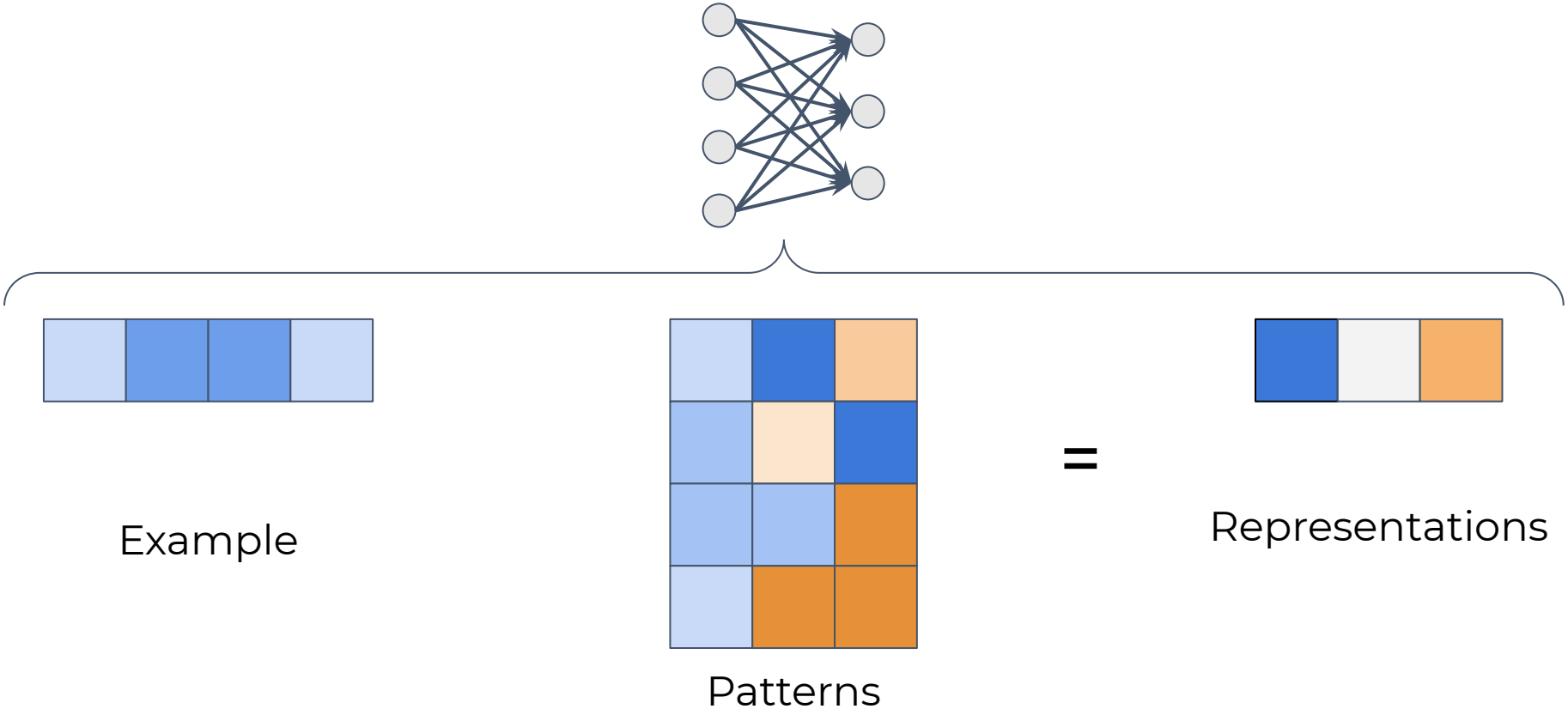




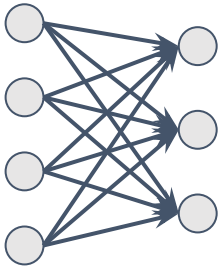
Dense module: vector-vector multiplication

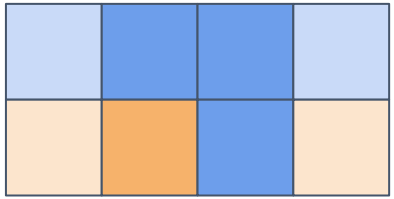


Dense module: matrix-vector multiplication

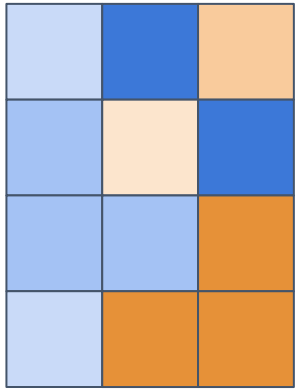


Dense module: matrix-matrix multiplication



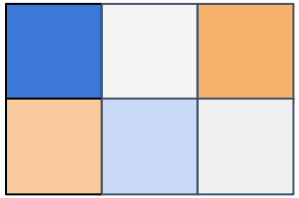


Examples



Patterns

=

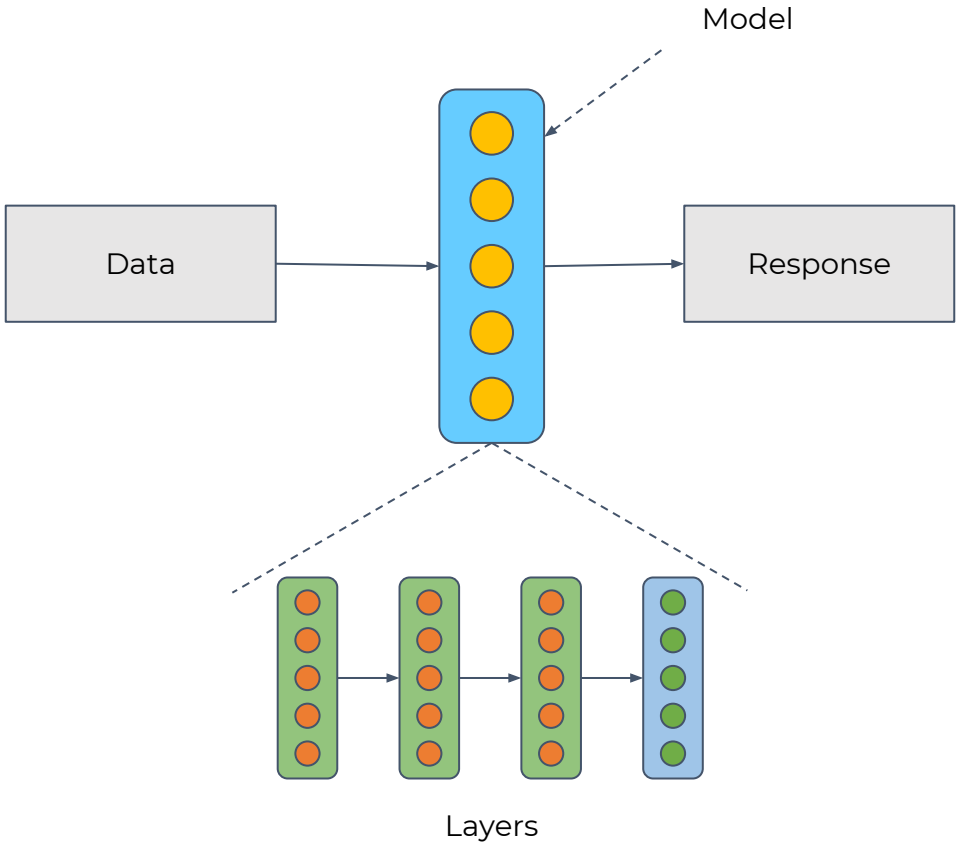


Representations

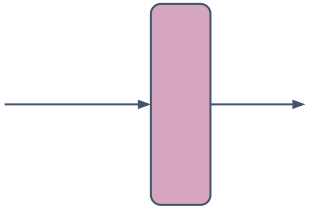
Advantages over classical diagrams

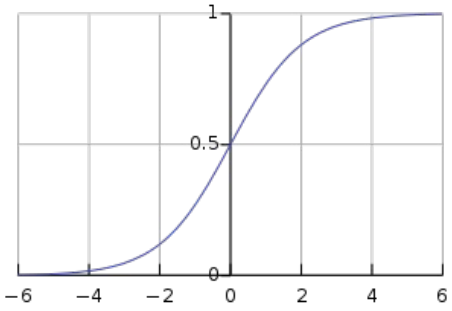
- Data and patterns are stocked in tensors (vector, matrix, N-d array).
- Deep learning operations are encapsulated in modules.
- A module can process examples in parallel.
- A module can be implemented with linear algebra operations, which are:
  - efficiently implemented in existing libraries,
  -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

Modular approach

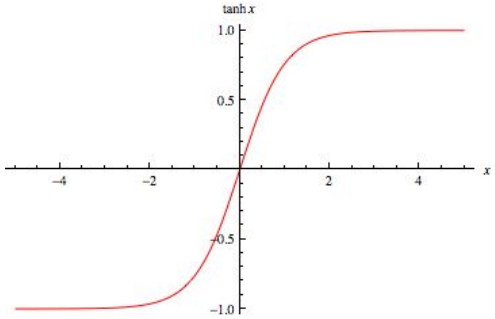


Activation functions

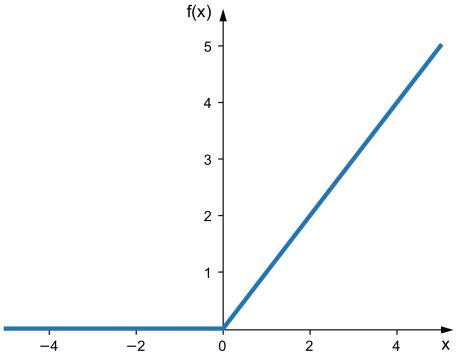




Sigmoid



Tanh

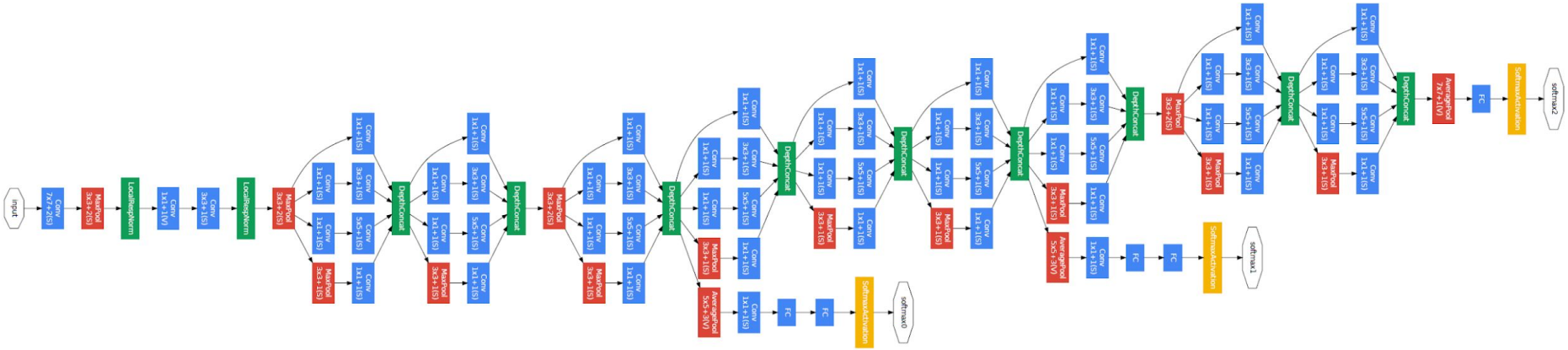


ReLU

Modular approach

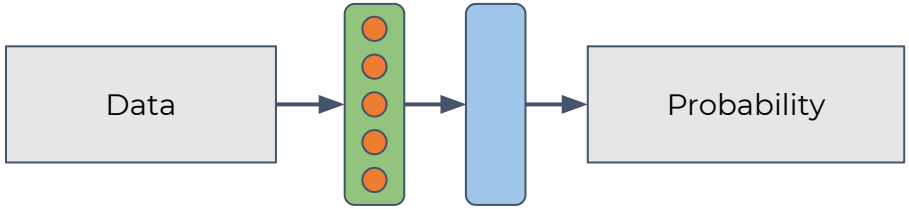
- A module is an elementary building block that you can combine with others to create new modules.
- The output of a module is the input of another one.
- Many elementary modules are implemented efficiently in modern deep learning libraries.

Modular approach: example



Mila

How to build a probabilistic model?
Binary classification



Source: Makhmutova Dina, Unsplash

Cat	90%
Not cat	10%

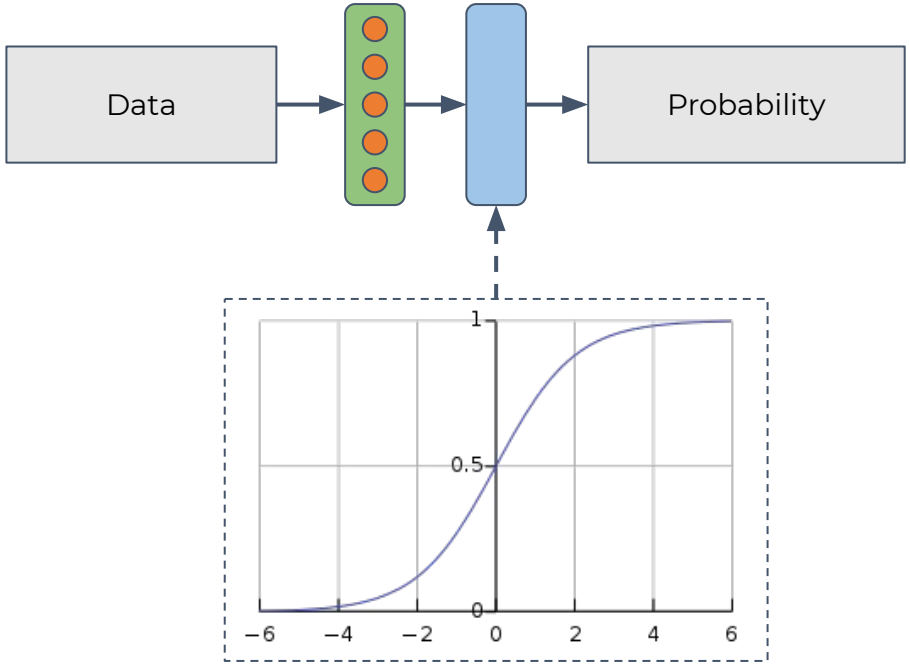
How to build a probabilistic model?

Binary classification:

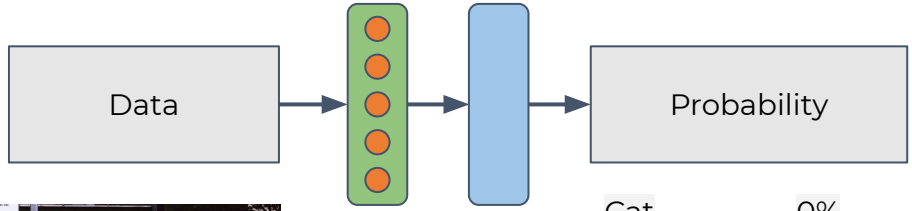
- Dense module for mapping to a single scalar.
- A **sigmoid** module to normalize the scalar between 0 and 1.

Output is the parameter *p* of a Bernoulli distribution:

- *p* is the probability of class 1
- *1-p* is the probability of class 0.



How to build a probabilistic model?
Multi-label classification



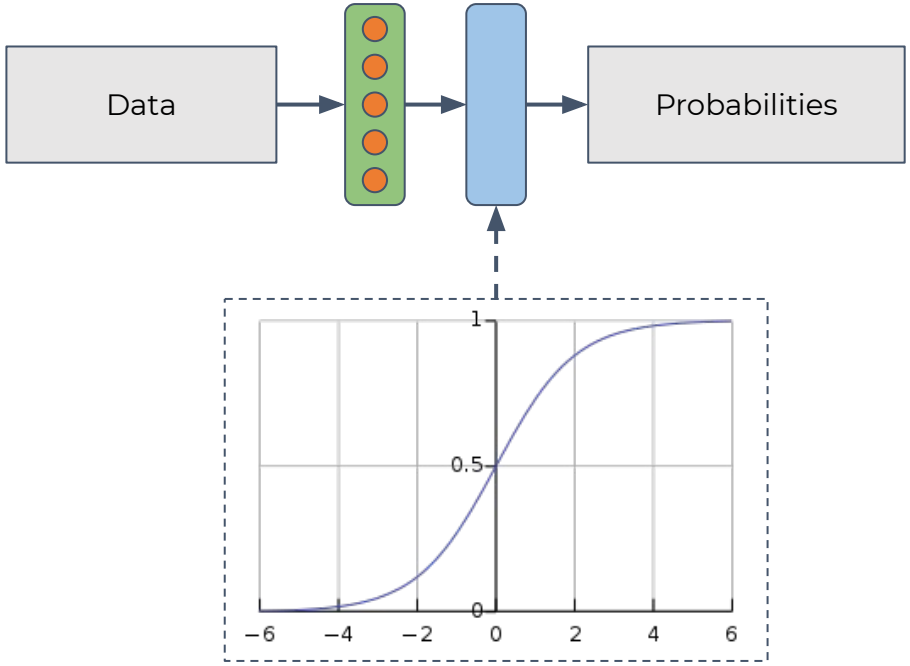
Source: Daryan Shamkhali, Unsplash

Cat	0%
Car	60%
Person	95%
Tree	75%
Dog	0%
Bike	60%

How to build a probabilistic model?

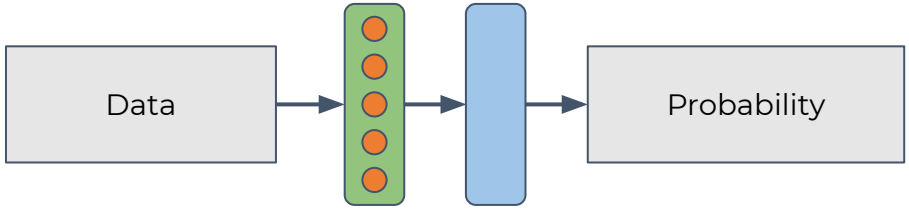
Multilabel classification:

- **Dense module** for mapping to the number of classes.
- A **sigmoid module** to normalize each component between 0 and 1.



For each component

How to build a probabilistic model?



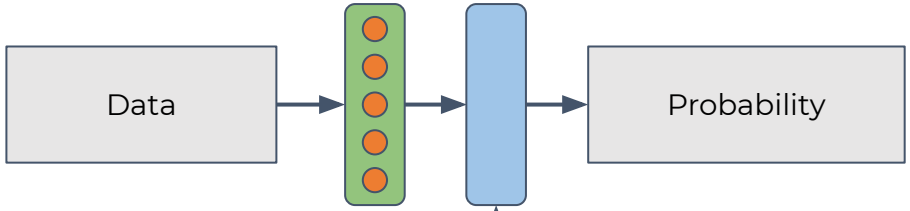
Source: Makhmutova Dina, Unsplash

Cat	90%
Dog	2%
Horse	0%
Tiger	8%
Penguin	0%

How to build a probabilistic model?

Multiclass classification:

- **Dense module** for mapping to the number of classes.
- A **softmax module** to normalize each component between 0 and 1 and to have their sum equals to 1.



$$p_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

Design patterns

- Encoder-decoder
  - Multi-input encoder
  - Multi-output decoder
- Teacher-student networks
- Modulation
- Bottom-up approach
- Adversarial
- Auto-regressive, …

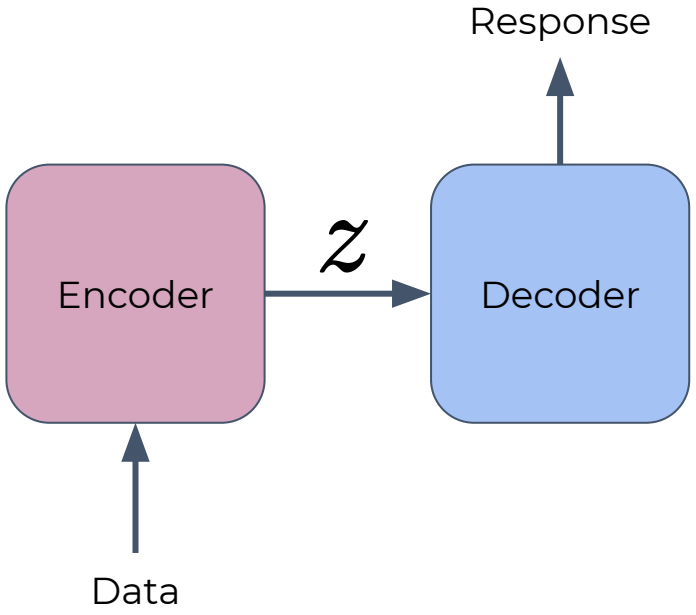
Source: Koushik Chowdavarapu, Unsplash

Encoder-decoder

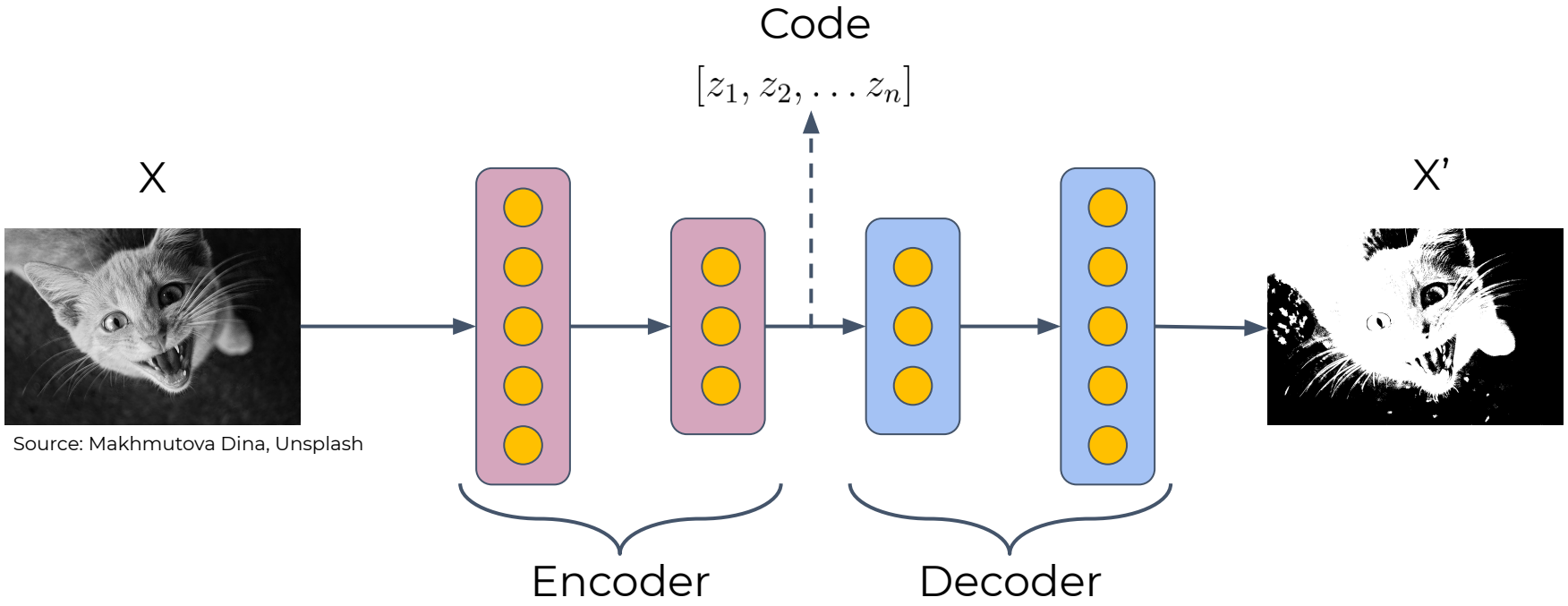
Learning a latent representation z

Uses

- Autoencoder
- Seq2seq (RNN)
- Encoder-decoder network

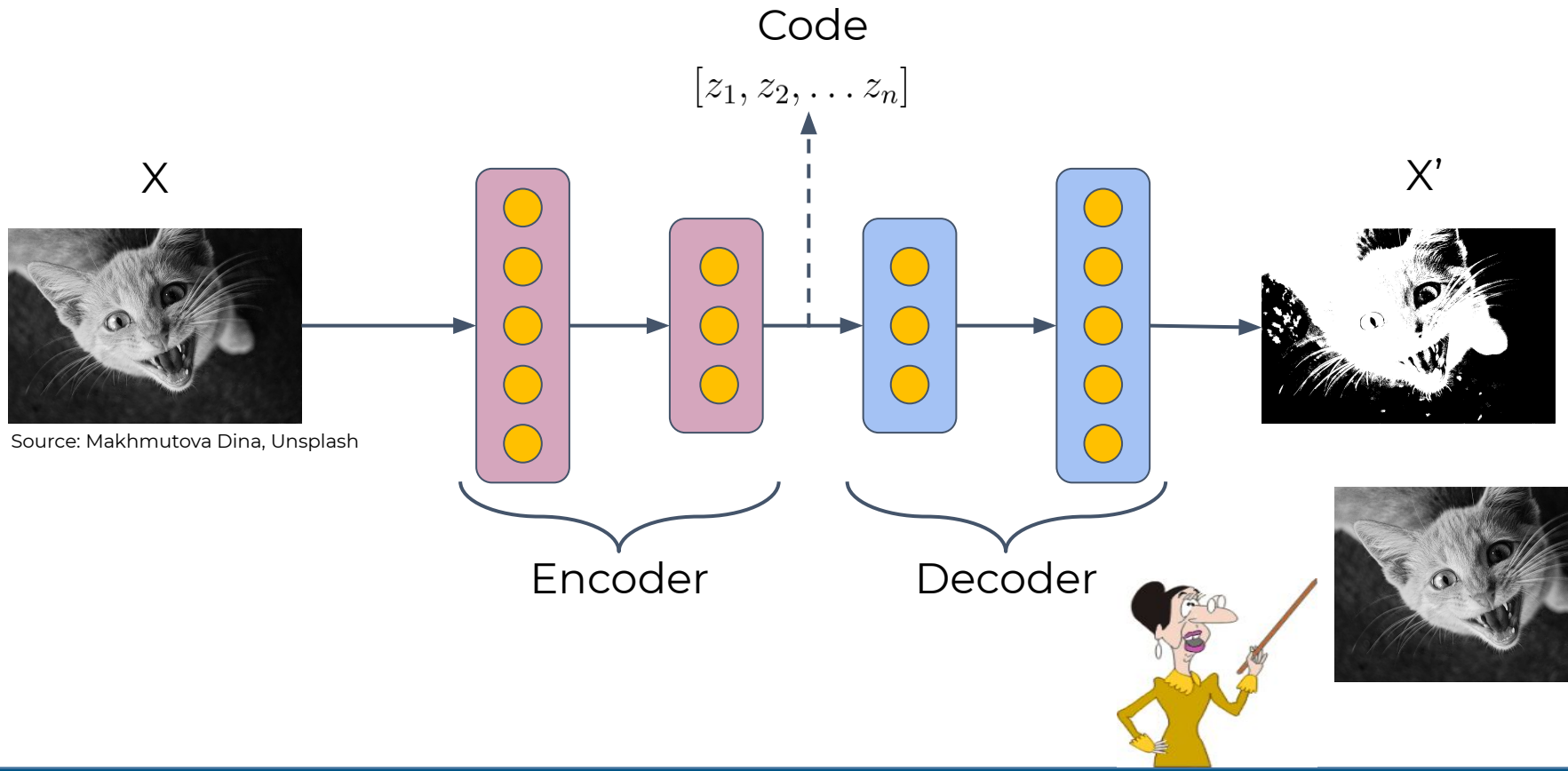


Example: Autoencoder



Source: Makhmutova Dina, Unsplash

Example: Autoencoder



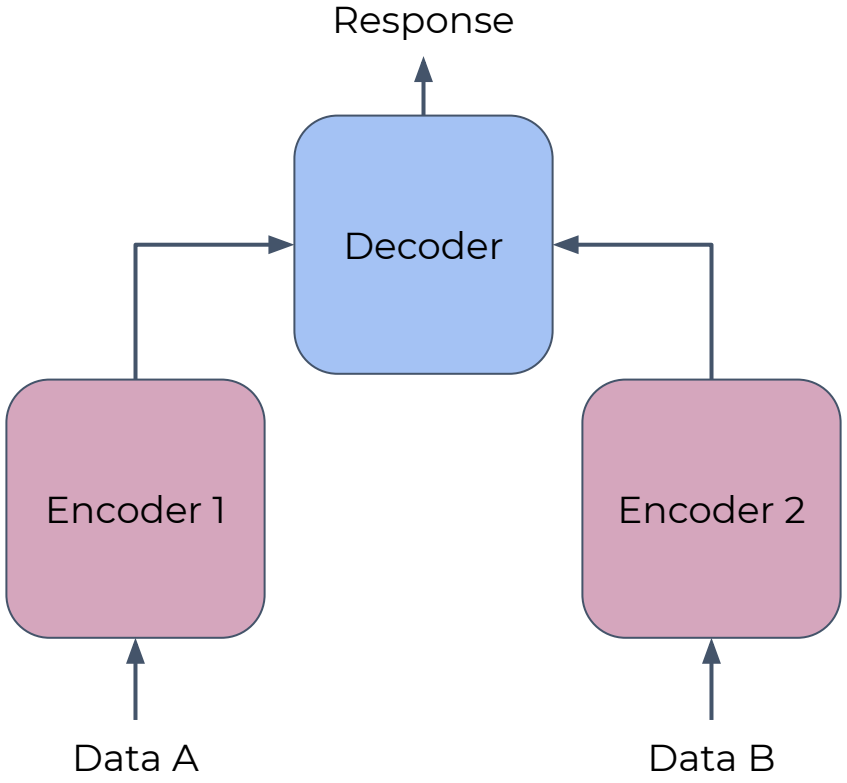
Source: Makhmutova Dina, Unsplash

Multi-input encoder-decoder

Encode and merge different information flows

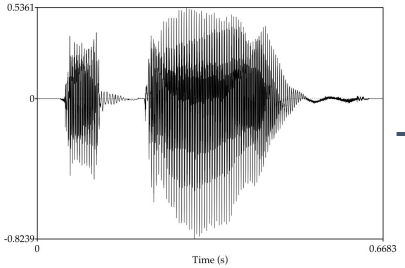
Uses:

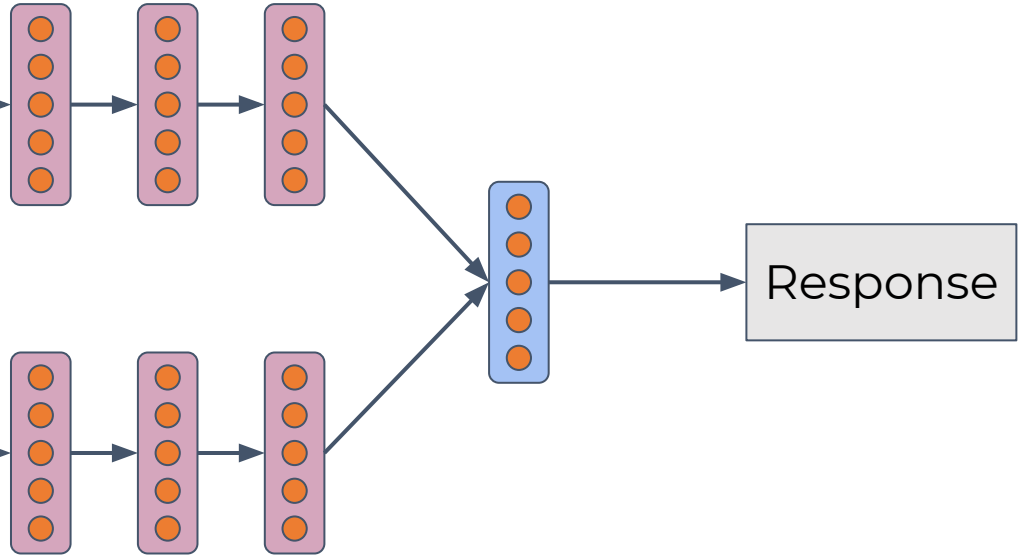
- Process different modalities
- Metric learning (Siamese)
- Relation learning



Example: multi-modality

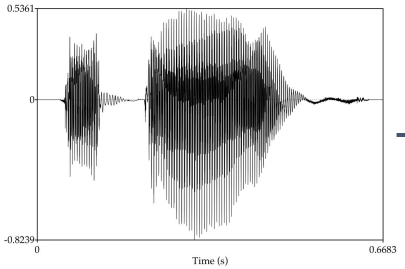
Source: Makhmutova Dina, Unsplash





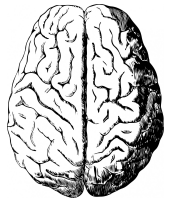
Example: multi-modality

Source: Makhmutova Dina, Unsplash



$[z_1, z_2, \ldots z_n]$

Latent space

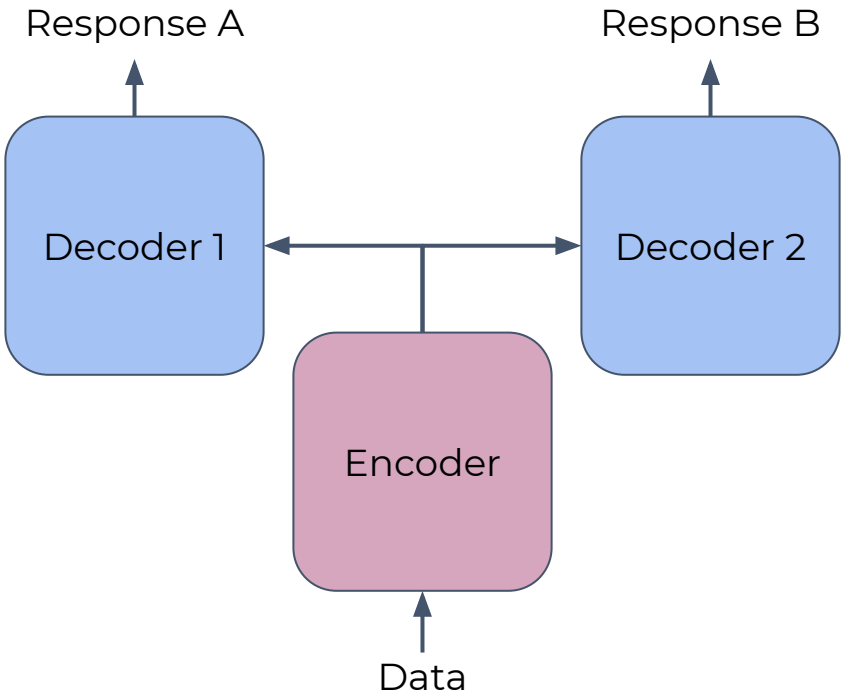


Multi-output encoder-decoder

Common representation

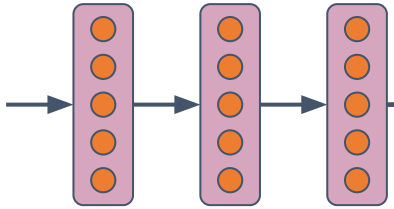
Uses:

- Multi-task learning



Example: multi-task learning

Source: Makhmutova Dina, Unsplash



Response 1

Response 2

Response 3

y_1 = Cat

y_2 = Yellow

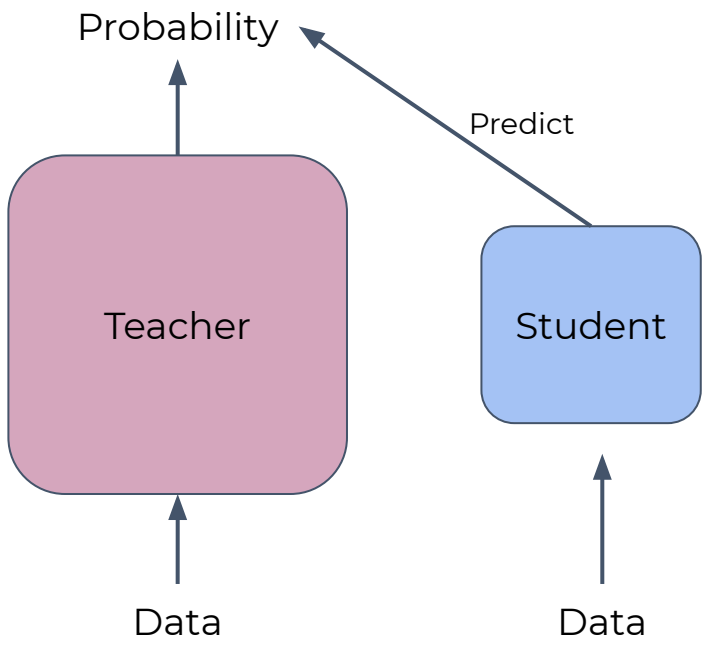
y_3 = Danger

Teacher-student architecture

Learn to predict the output of another model

Uses:

- Model compression

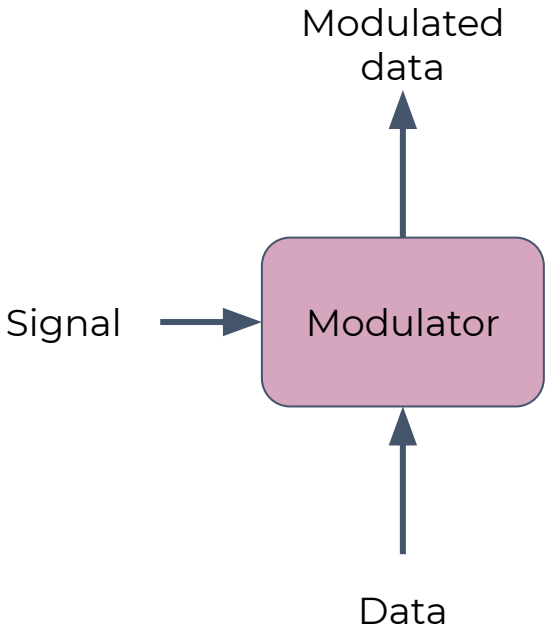


Modulation

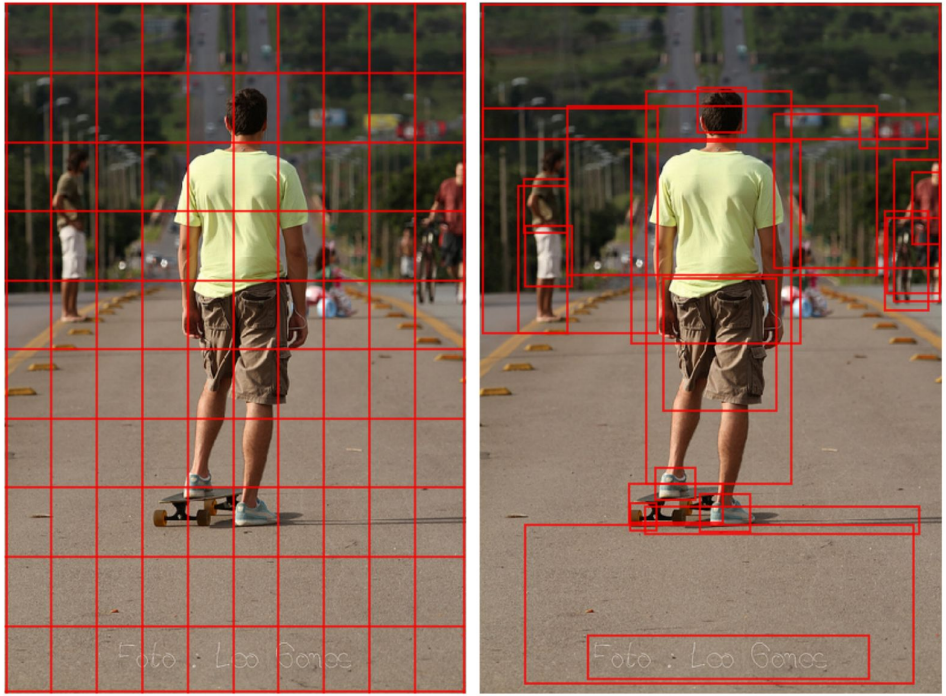
Adaptive modulation of the representation

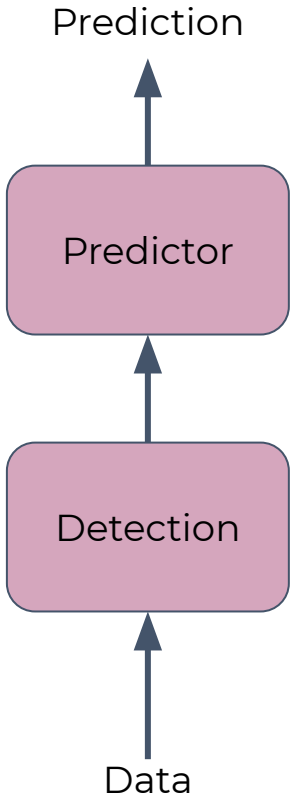
Uses:

- Gating (LSTM)
- Skip-connections
- Batch norm
- FiLM
- Attention mechanism



Bottom-up approach





Anderson, Peter, et al. "Bottom-up and top-down attention for image captioning and visual question answering." CVPR. Vol. 3. No. 5. 2018.

Take-home message

- With deep learning, we can define hypothesis classes with computational graphs (i.e., architectures) easily.
- Creating a graph for a given task and dataset is part of the expertise.
- An architecture often has many layers; each is providing a new distributed representation to the next one that will be easier to process.
- Design patterns for architecture are emerging in the literature.