Simplifying Image Processing Using Dimensionality Reduction

Janani Ravi

CO-FOUNDER, LOONYCORN

www.loonycorn.com

Overview

Dictionary learning for dimensionality reduction of image data

Feature detection using convolutional layers

Autoencoders for dimensionality reduction

Dictionary Learning and Feature Extraction

Dictionary Learning

Representation learning method to find a sparse representation of input data, often used in denoising of images.

Dictionary Learning for Image Denoising

Corpus of clean images

Should be free of noise

Feed into Dictionary Learner

Perform Dictionary Learning

Express clean images in terms of atoms (sparse codings)

Each image is expressed as linear combination of atoms

Choose transformation algorithm

Implement numeric procedures

Several choices - Orthogonal Matching Pursuit (OMP), LARS

Use of Image Denoising

Apply to new, noisy images

Reconstruct and express using atoms

Dictionary Learning

Several choices of solver

- Thresholding: Fast but inaccurate
- Orthogonal Matching Pursuit (OMP): Most accurate, unbiased
- Least-angle Regression
- Lasso Regression

Demo

Dictionary learning to learn sparse codings

Convolution Kernels for Feature Detection

“Sometimes the mind can see what is invisible to the eye”

Viewing an Image

All neurons in the eye don't see the entire image

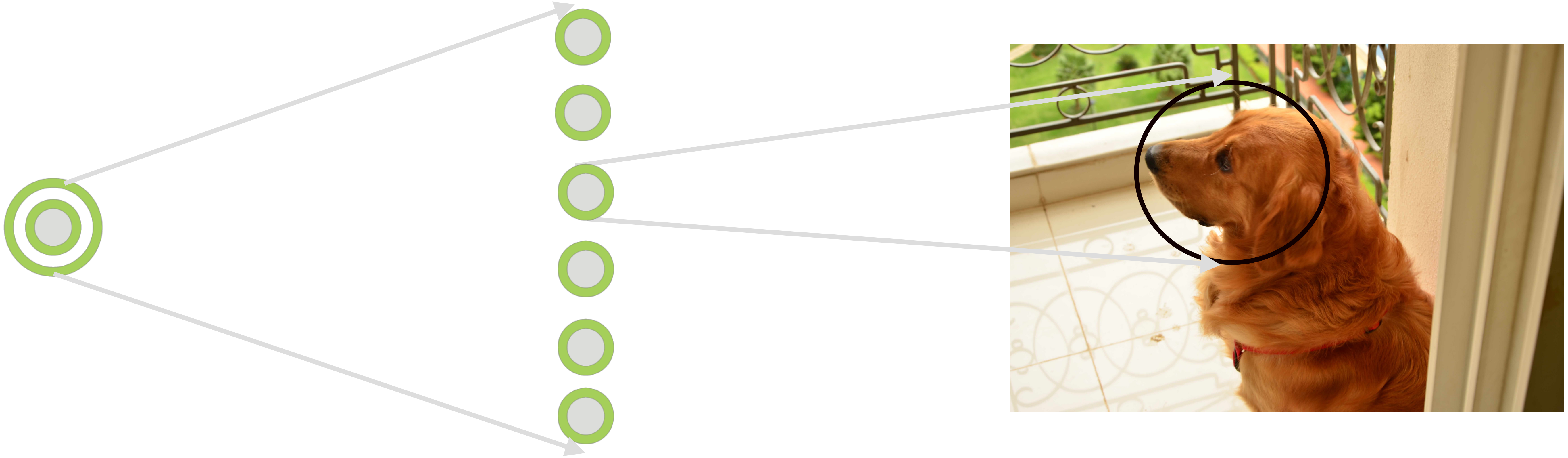
Viewing an Image

Each neuron has its own local receptive field

Viewing an Image

It reacts only to visual stimuli located in its receptive field

Viewing an Image



Some neurons react to more complex patterns that are combinations of lower level patterns

Convolution

In this context, a sliding window function applied to a matrix

Convolution

In this context, a sliding window function applied to a matrix

e.g. a matrix of pixels representing an image

Convolution

In this context, a sliding window function applied to a matrix

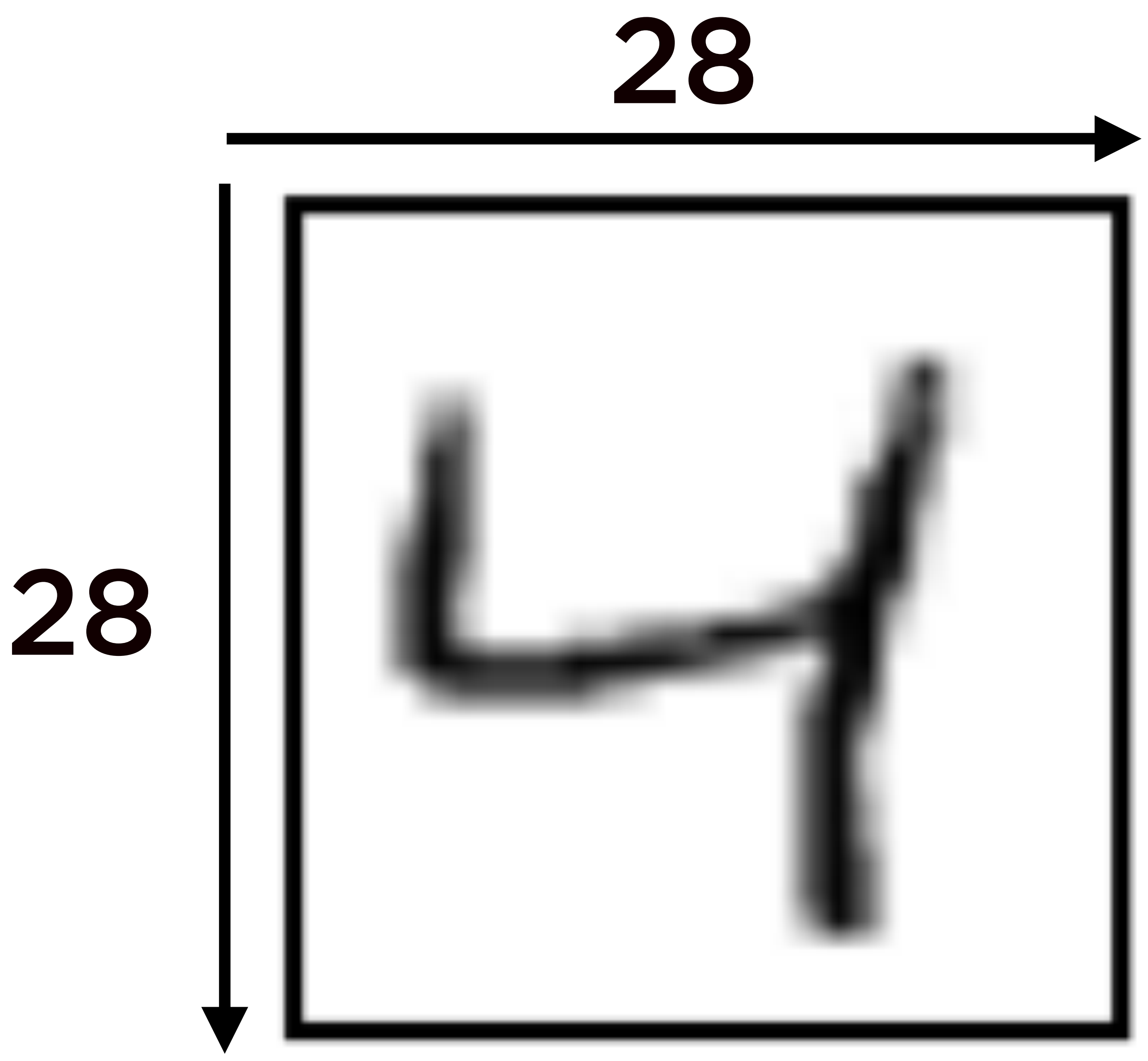
Often called a kernel or filter

Convolution

In this context, a sliding window function applied to a matrix

Kernel is applied element-wise
in sliding-window fashion

Representing Images as Matrices



= 784 pixels

Representing Images as Matrices

6

6

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

= 36 pixels

Representing Images

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

Matrix

3

3

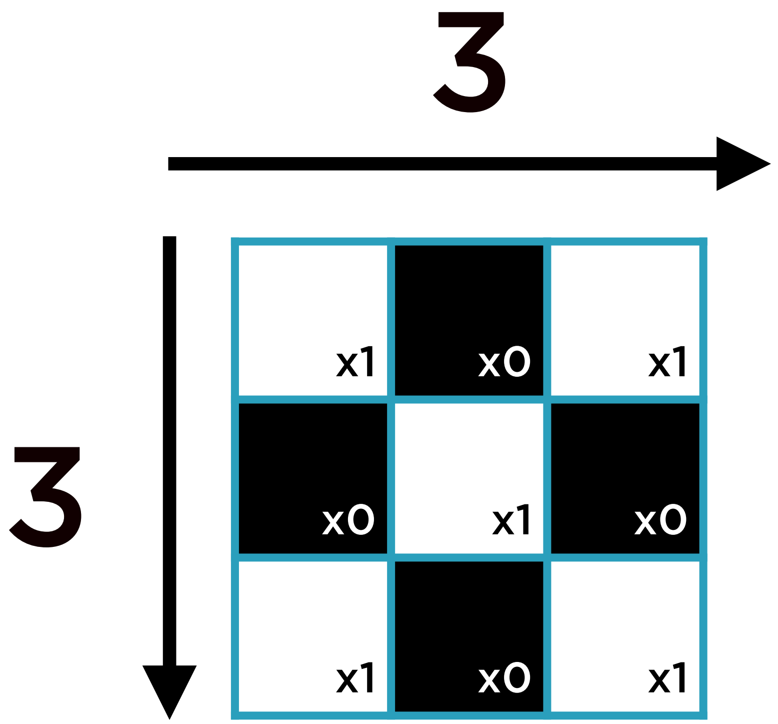
1	0	1
0	1	0
1	0	1

Kernel

Convolution

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

Matrix



Kernel

Convolution

0	0	0	0	0	0
0.2	0.8	0	0.3	0.6	0
0.2	0.9	0	0.3	0.8	0
0.3	0.8	0.7	0.8	0.9	0
0	0	0	0.2	0.8	0
0	0	0	0.2	0.2	0

Matrix

x1	x0	x1
x0	x1	x0
x1	x0	x1

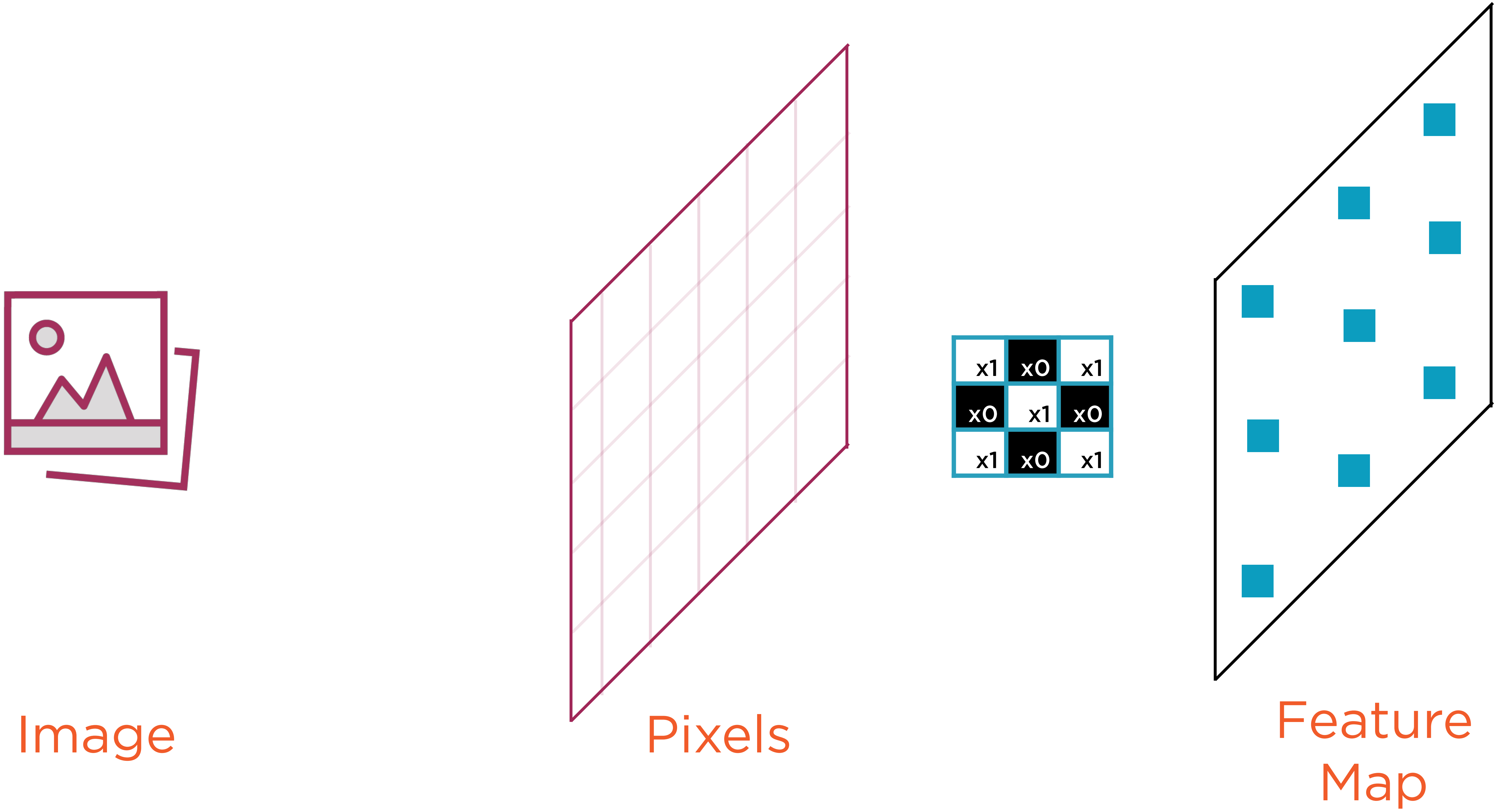
4

4

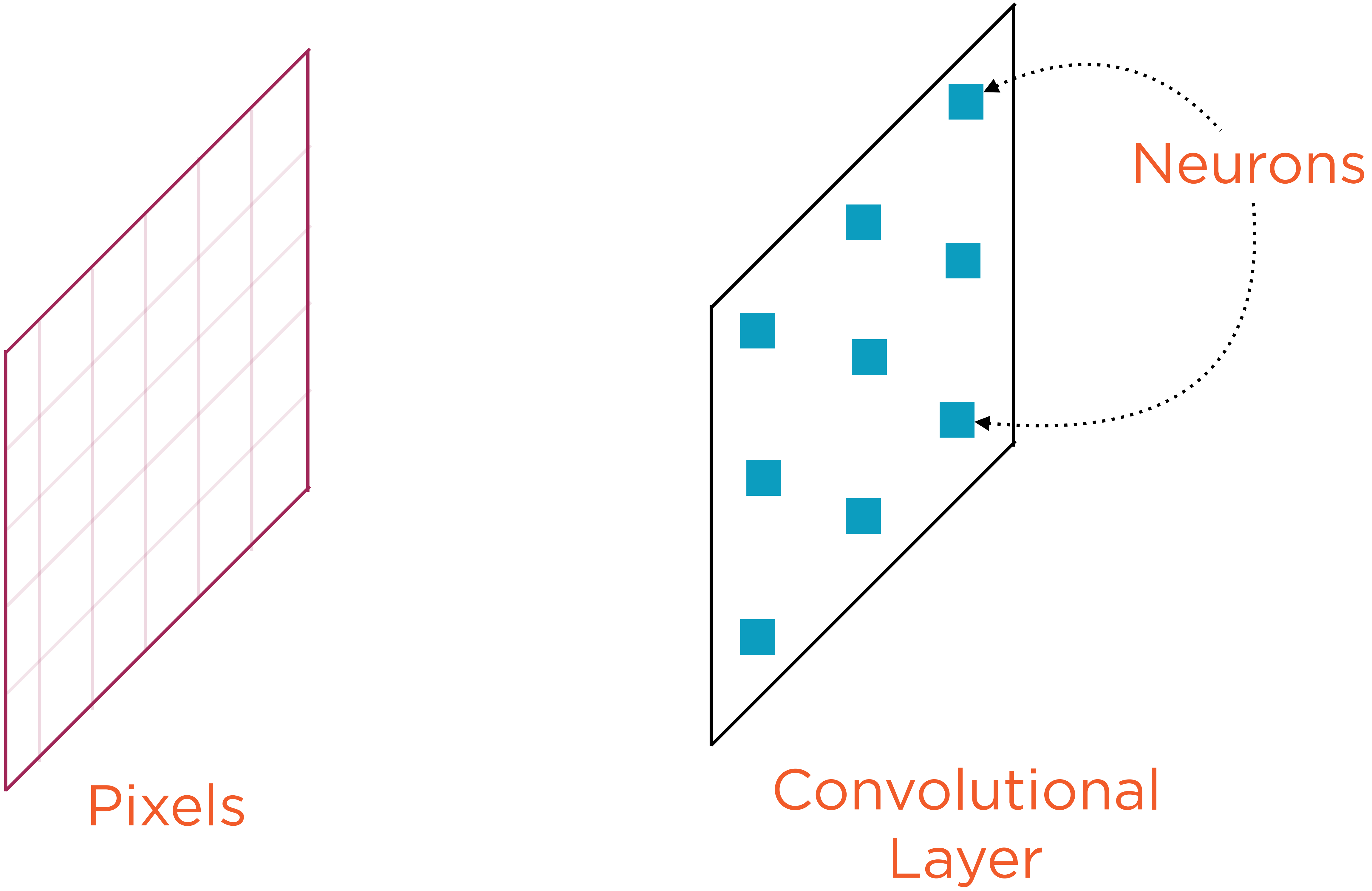
1	1.2	1.1	0.9
1.9	2.7	2.5	1.9
1.0	2.1	2.4	1.4
1.0	1.8	2.0	1.8

Convolution Result

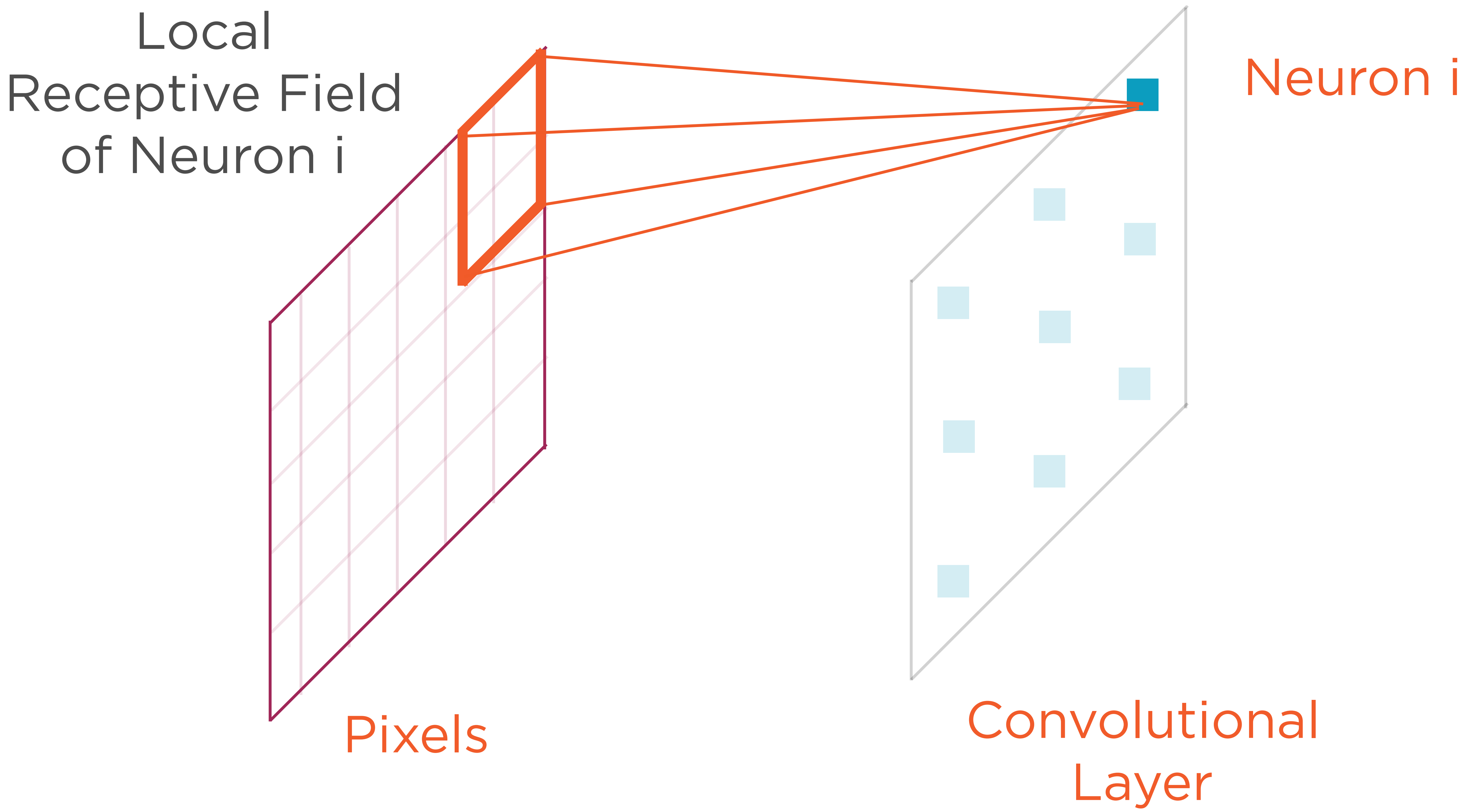
Feature Maps



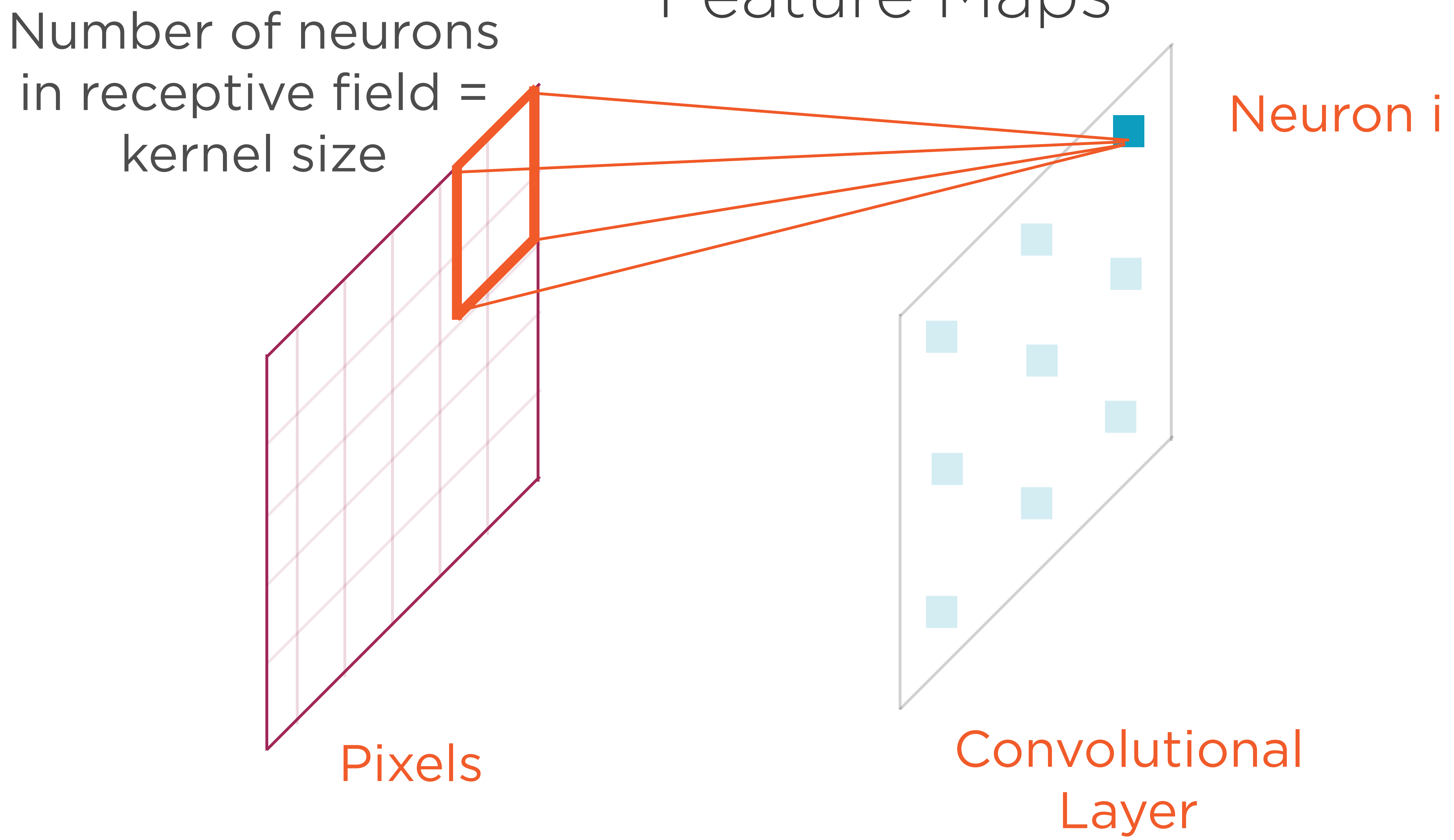
Feature Maps



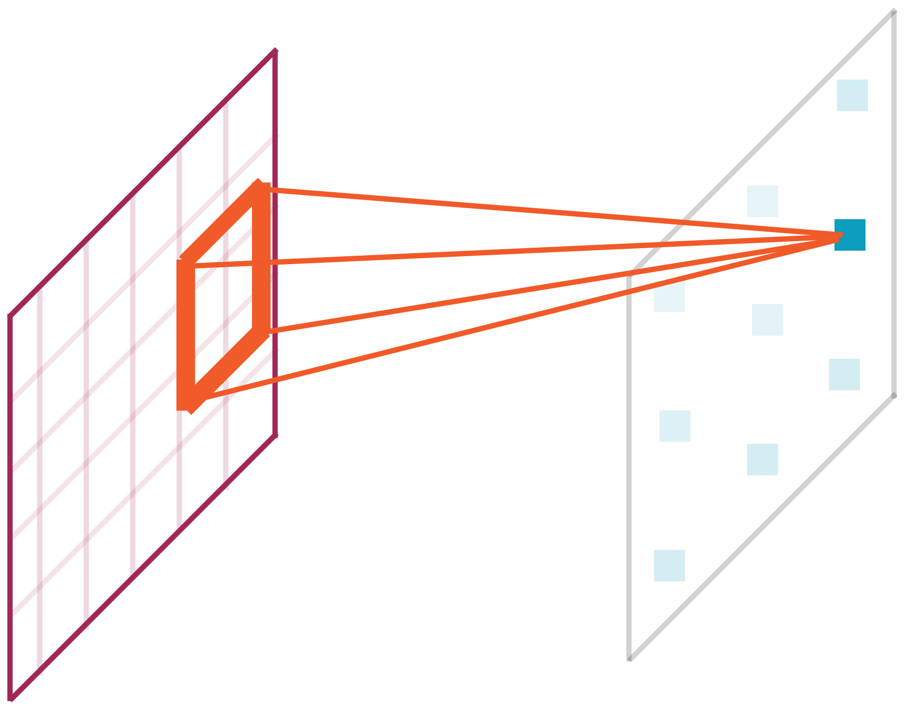
Feature Maps



Feature Maps



Feature Maps

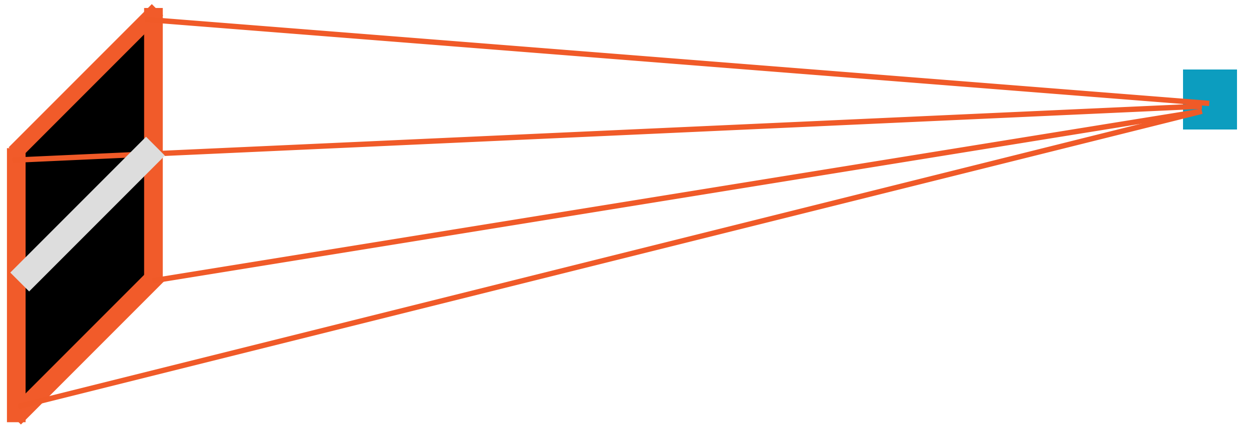


The parameters of all neurons in a feature map are collectively called the filter

Why filter?

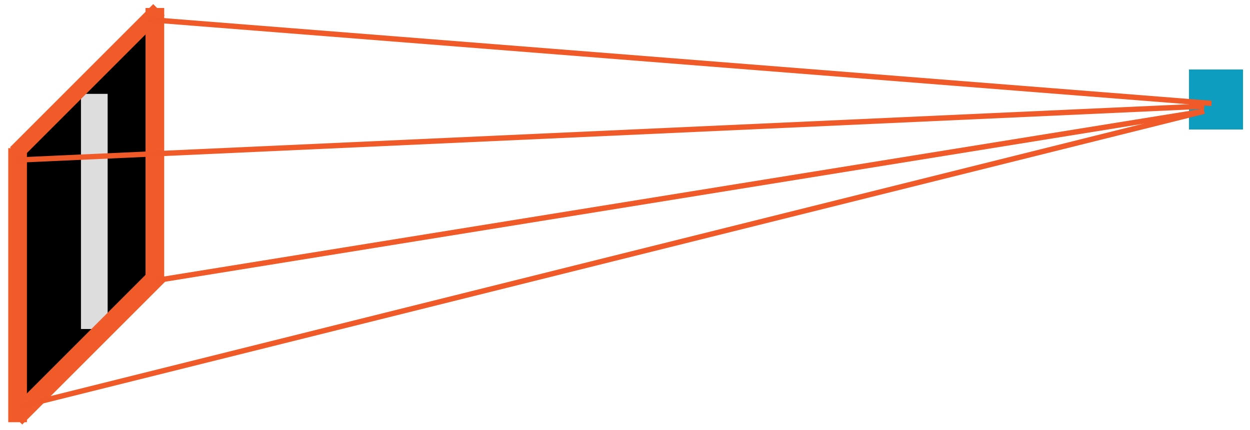
Because weights highlight (filter) specific patterns from the input pixels

Filters



Horizontal Filter

Neuron will detect
horizontal lines in input



Vertical Filter

Neuron will detect
vertical lines in input

Demo

Convolutional kernels for feature detection

Autoencoding

Choosing Autoencoders

Use Case	Possible Solution
Extremely complex feature vectors Images, video, documents Pre-processing before using in neural networks	Autoencoders

```
y = f(x)
```

Supervised Machine Learning

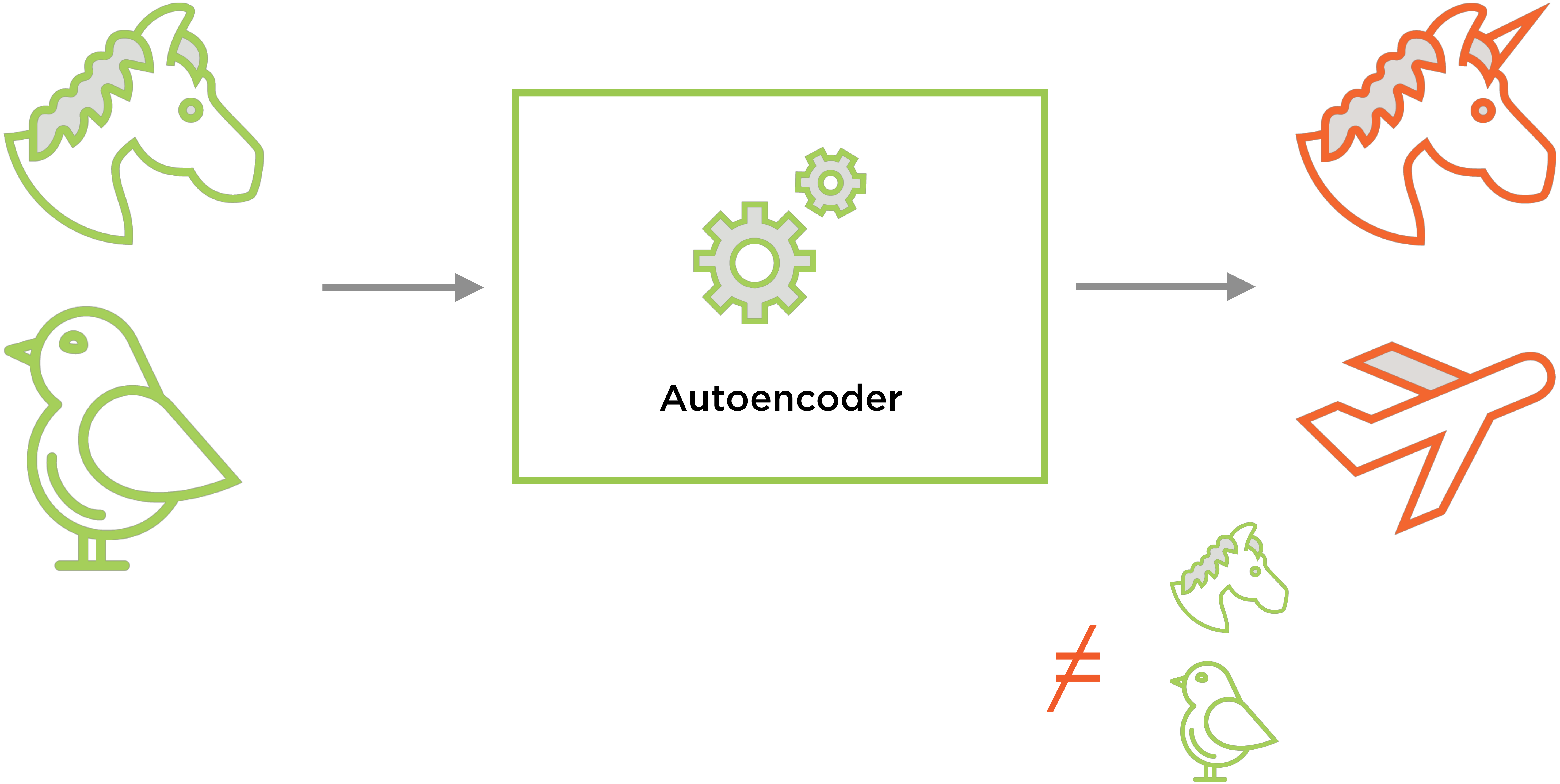
Most machine learning algorithms seek to “learn” the function f that links the features and the labels

```
x = f(x)
```

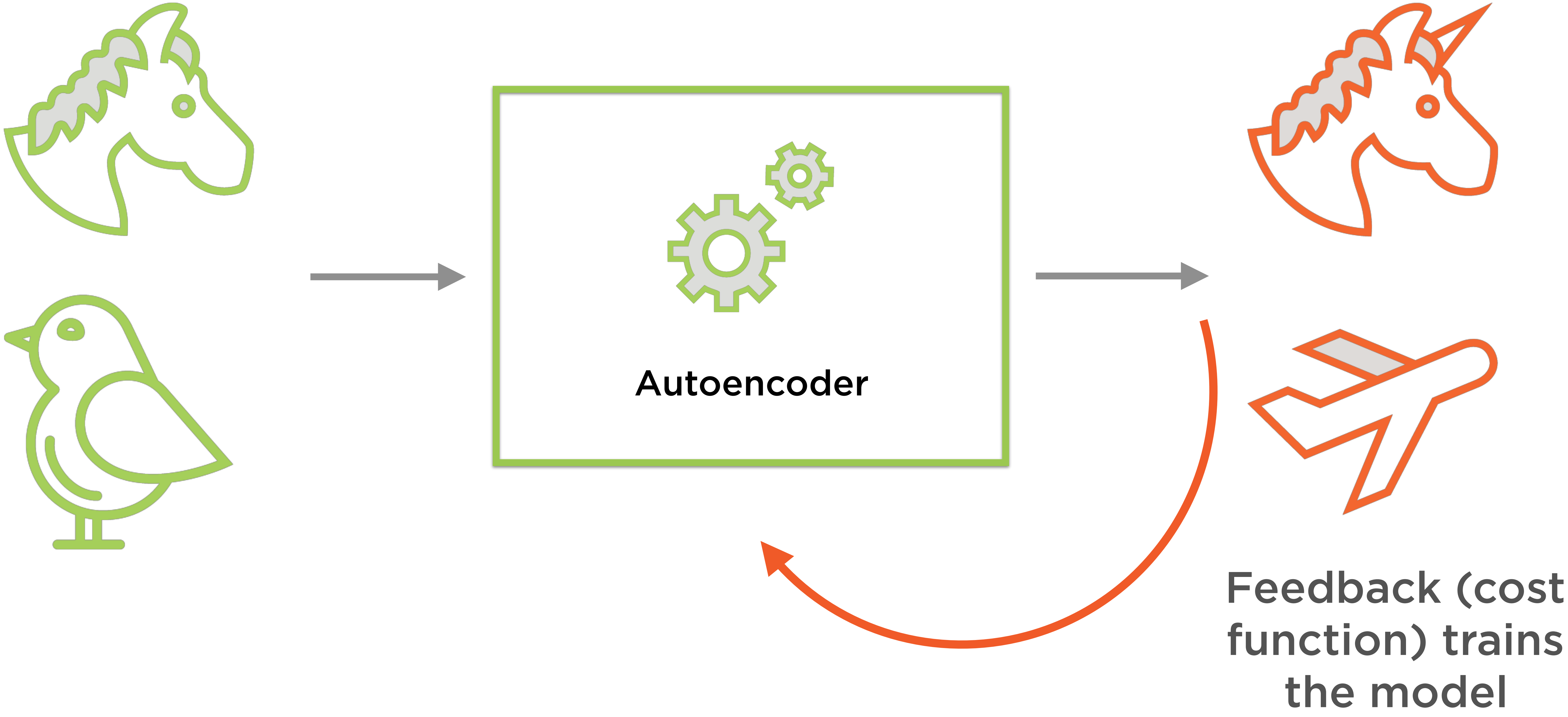
Autoencoders Learn the Input!

The process is inherently unsupervised, but cleverly uses the input itself to train an algorithm

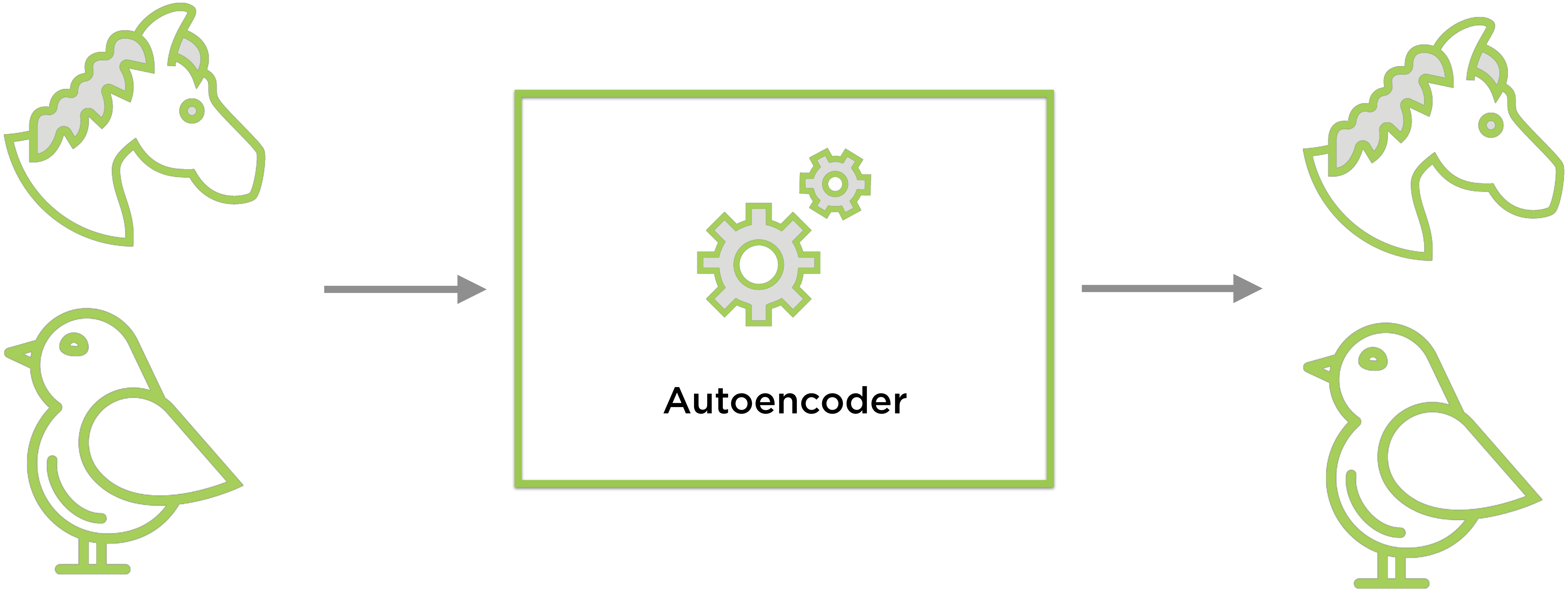
Autoencoder



Autoencoder

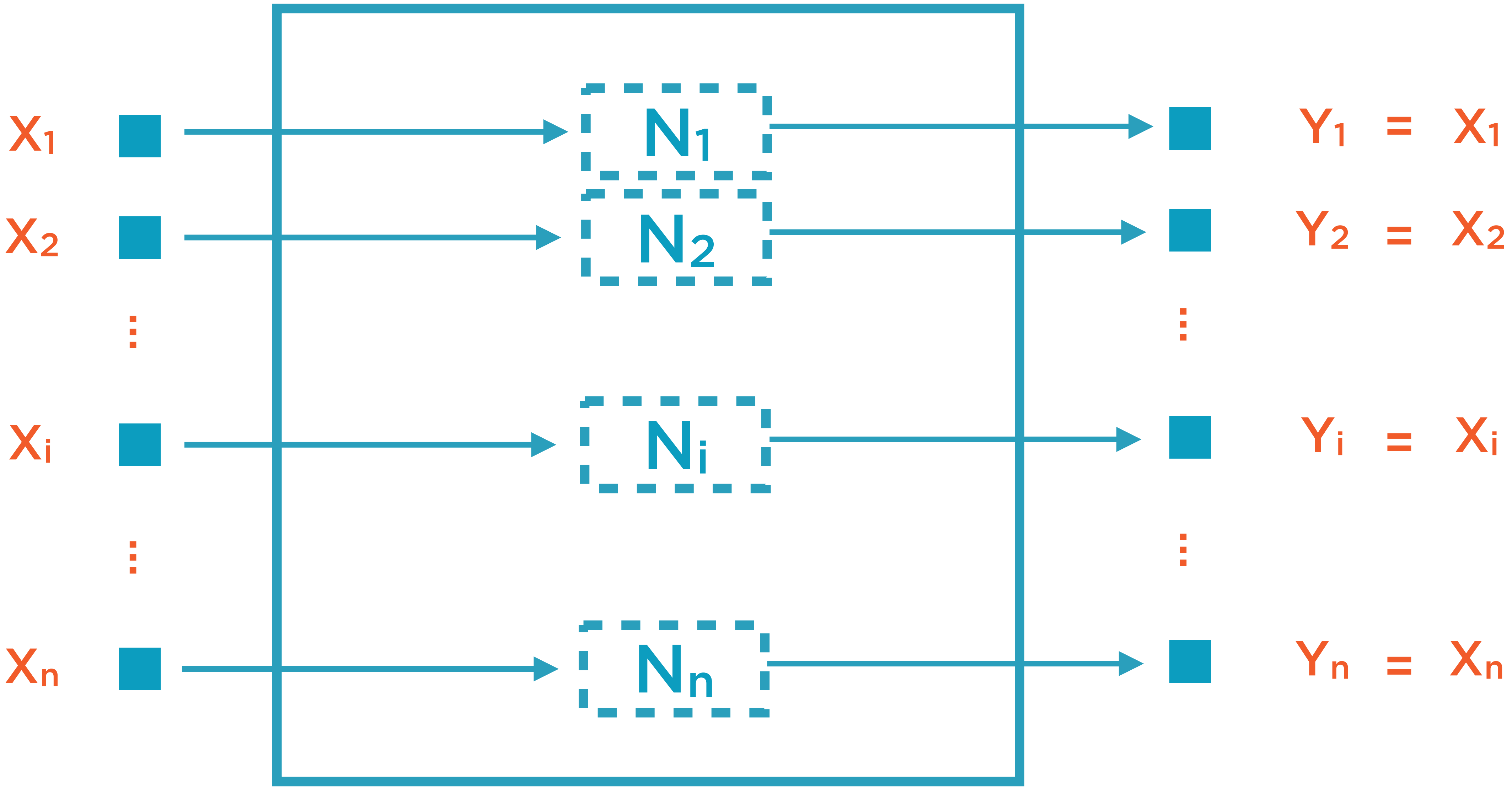


Autoencoder



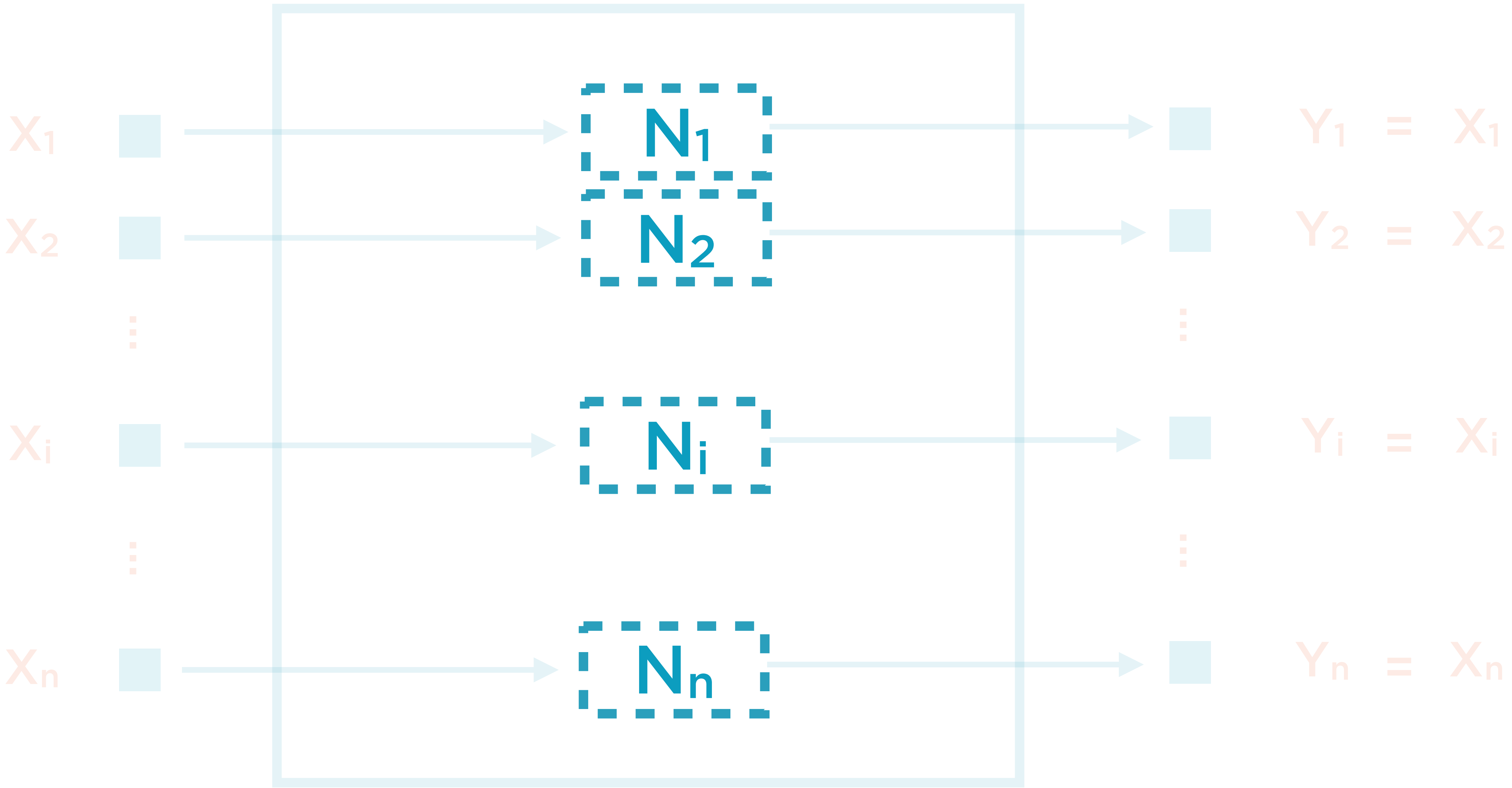
Autoencoders are Neural Networks that learn efficient representations of data (e.g. PCA)

Trivial Autoencoder



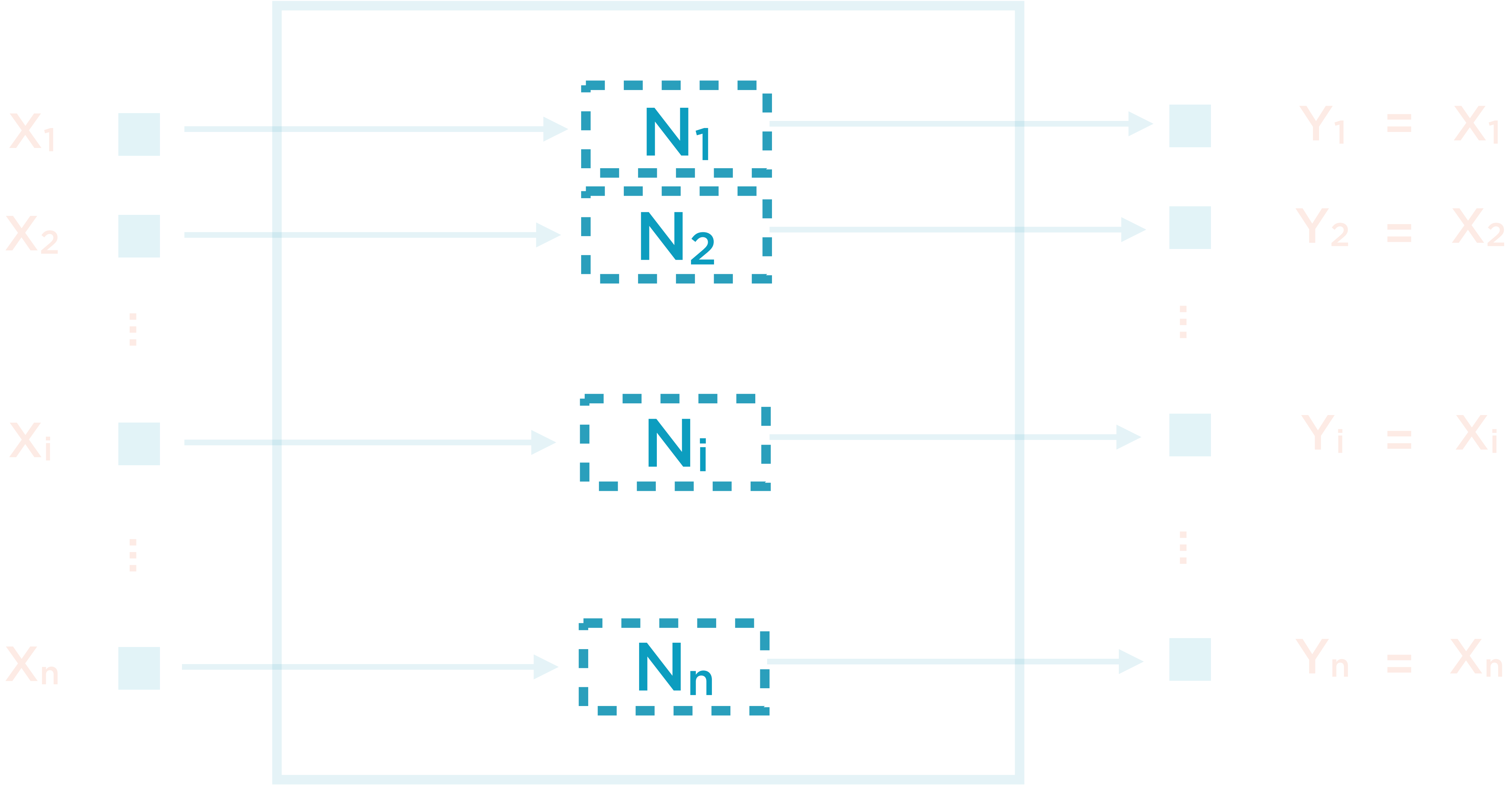
This Neural Network trivially "learns" the input

Trivial Autoencoder



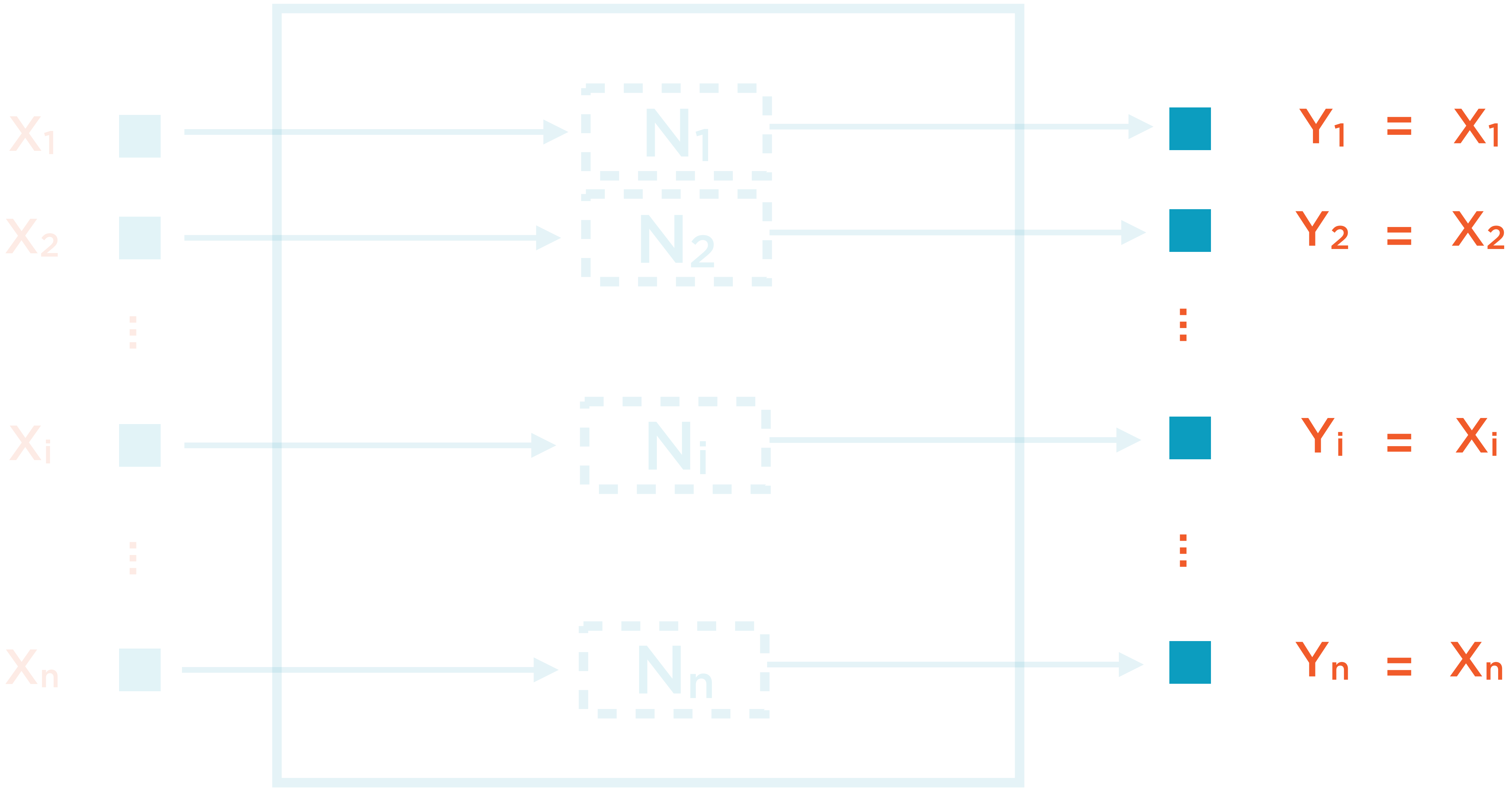
Just one layer, which serves as both input and output layer

Trivial Autoencoder



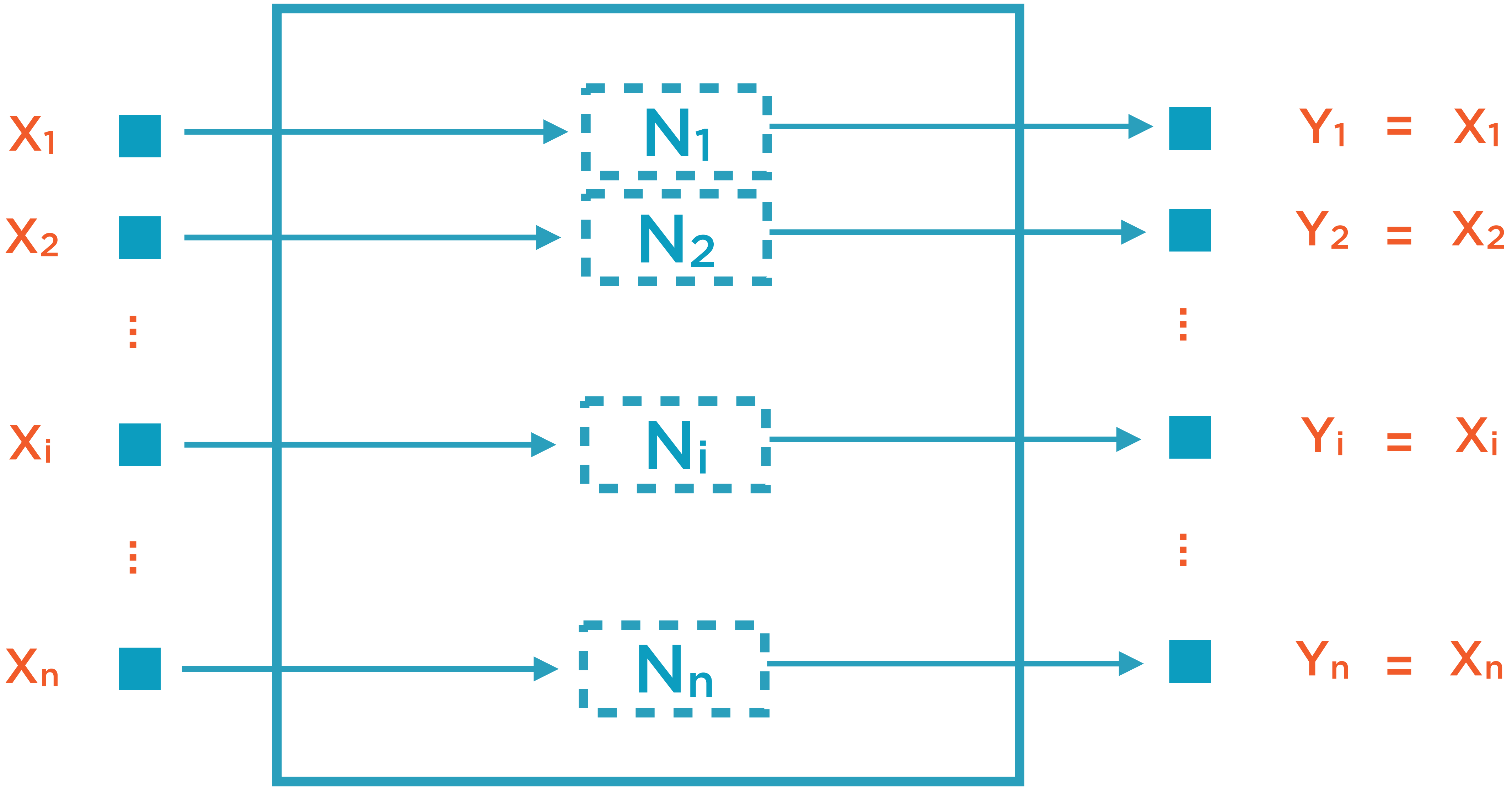
In an autoencoder, the input and output layer must have same dimensionality as the input data

Trivial Autoencoder



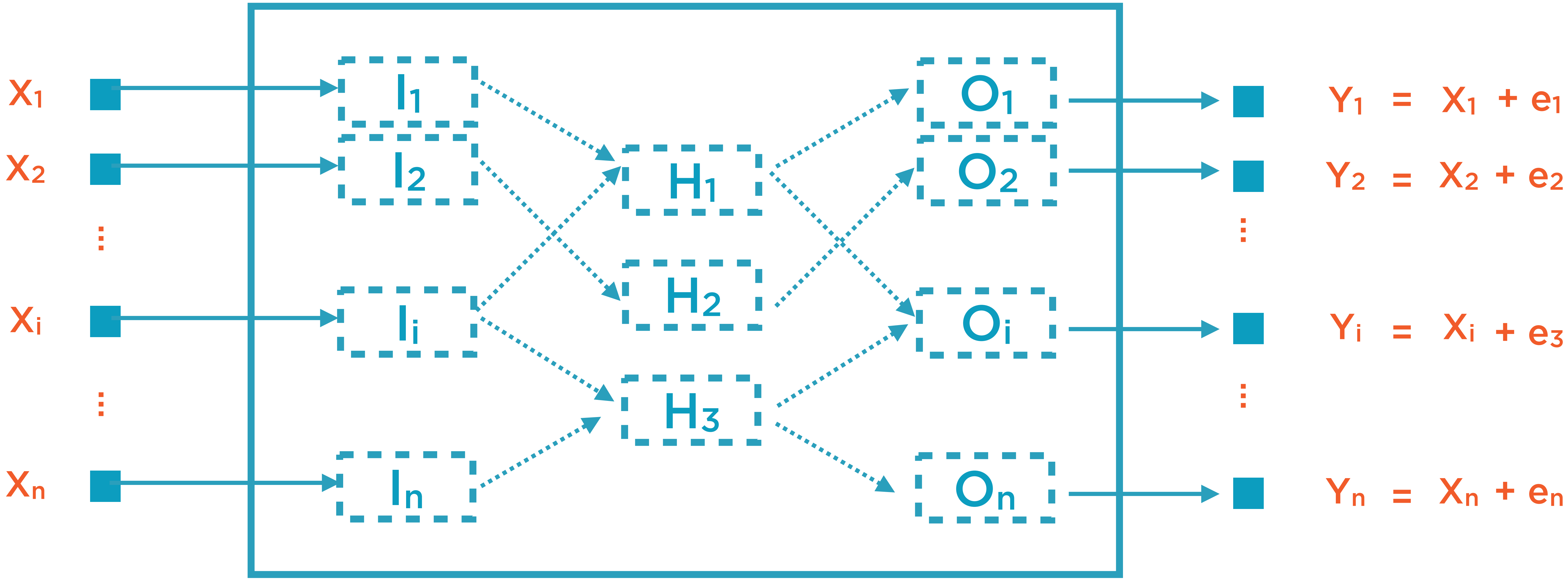
The autoencoder just passes the input through, so output is exactly equal to input

Trivial Autoencoder



Now, let's constrain the network to force dimensionality reduction

Undercomplete Autoencoder

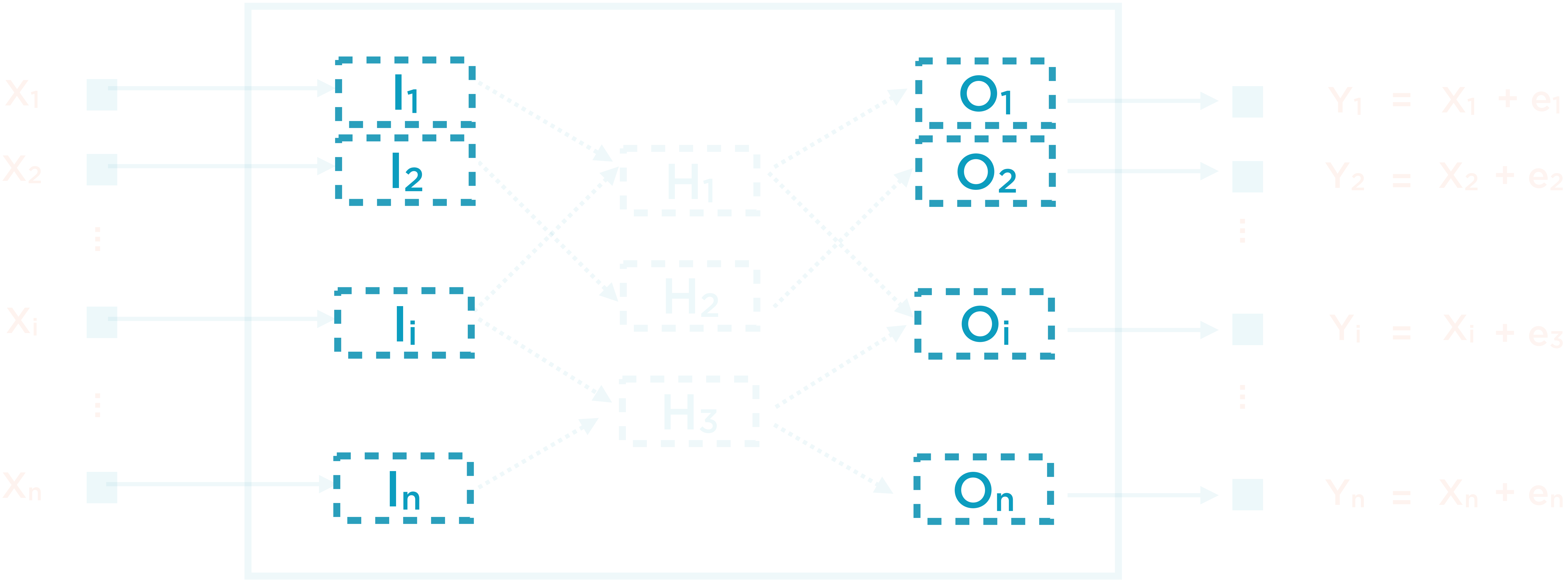


Dimensionality of the NN is now lower than that of input data

Undercomplete Autoencoder

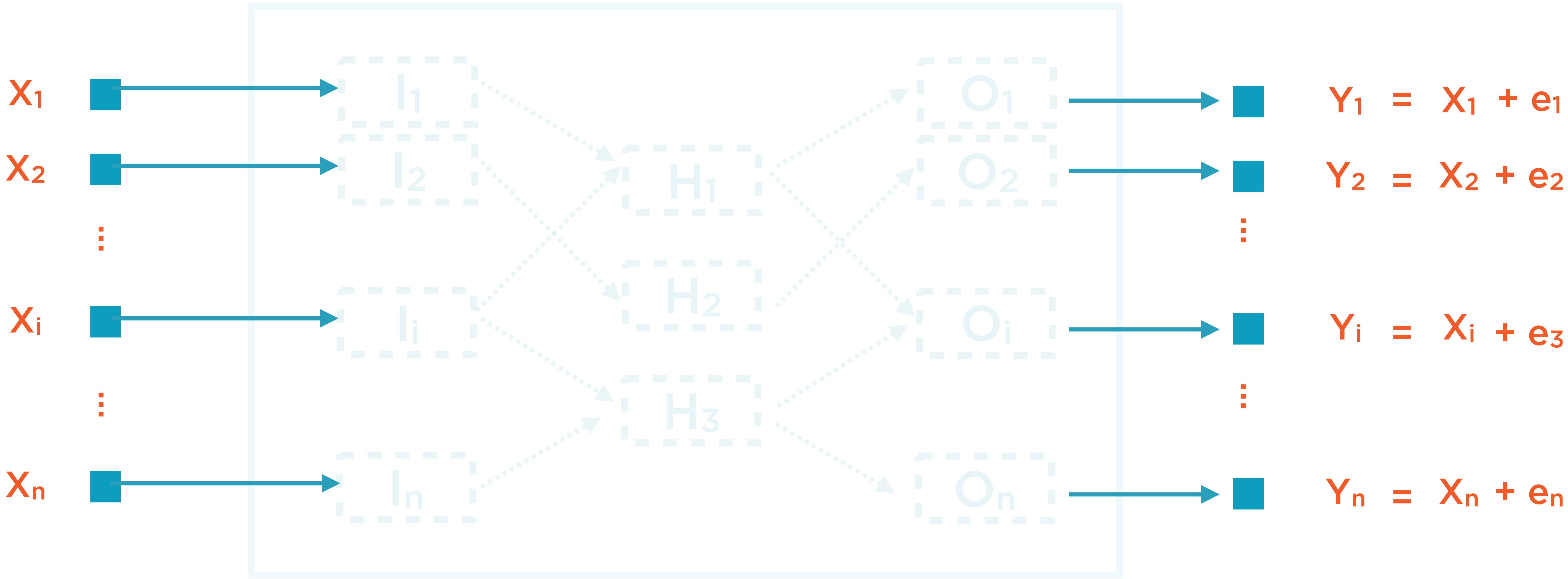
Add a middle, hidden layer with just three neurons (3 < N)

Undercomplete Autoencoder



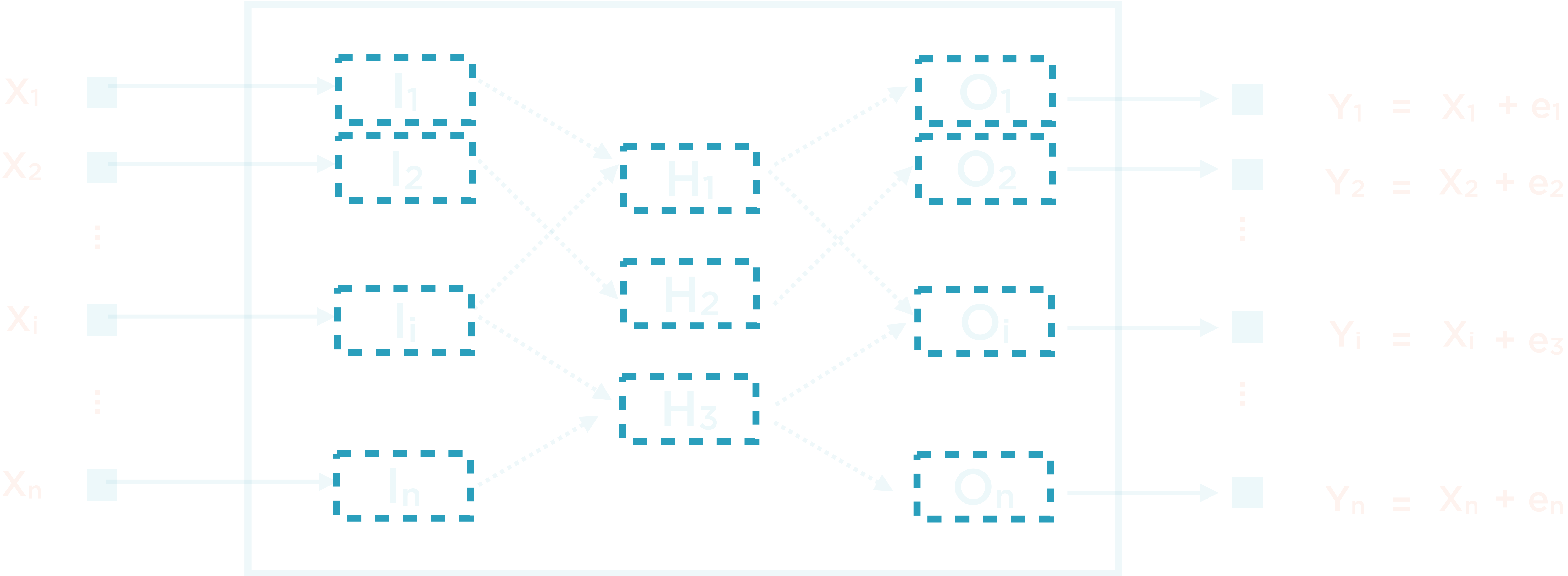
The input and output layers must now be separated, since each must still have same dimensionality as the input

Undercomplete Autoencoder



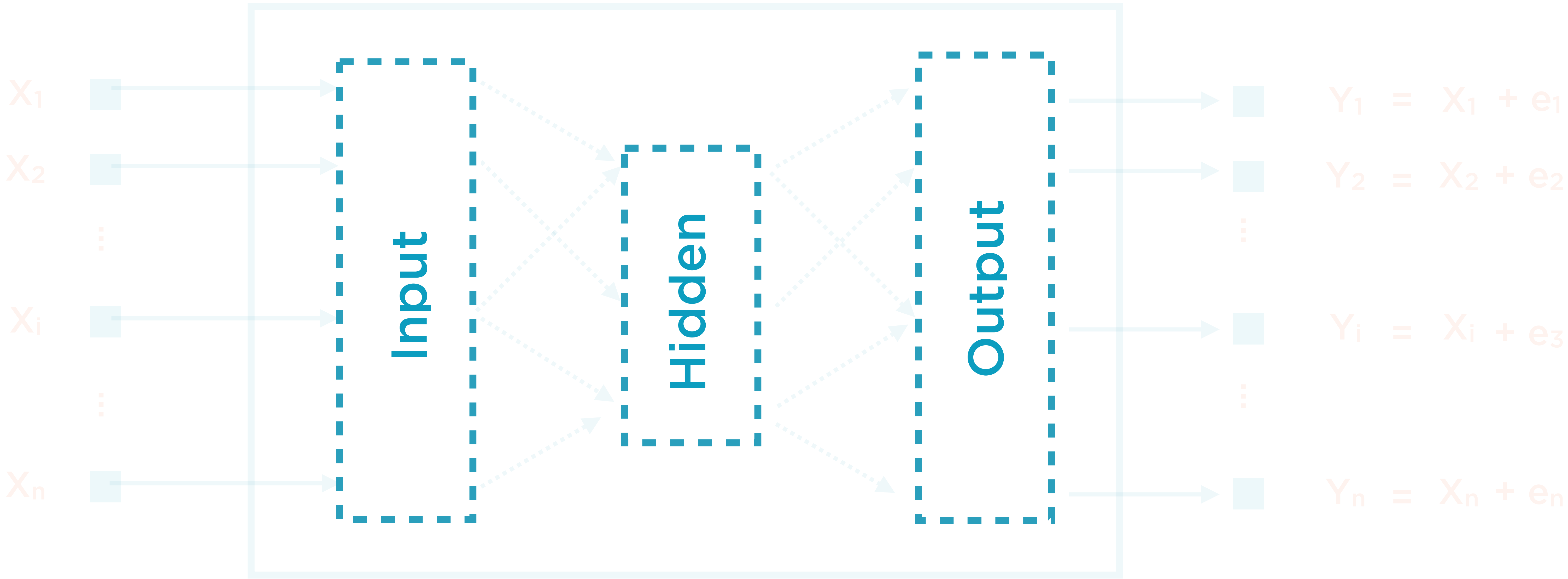
Why? Because autoencoder seeks to reconstruct input

Undercomplete Autoencoder



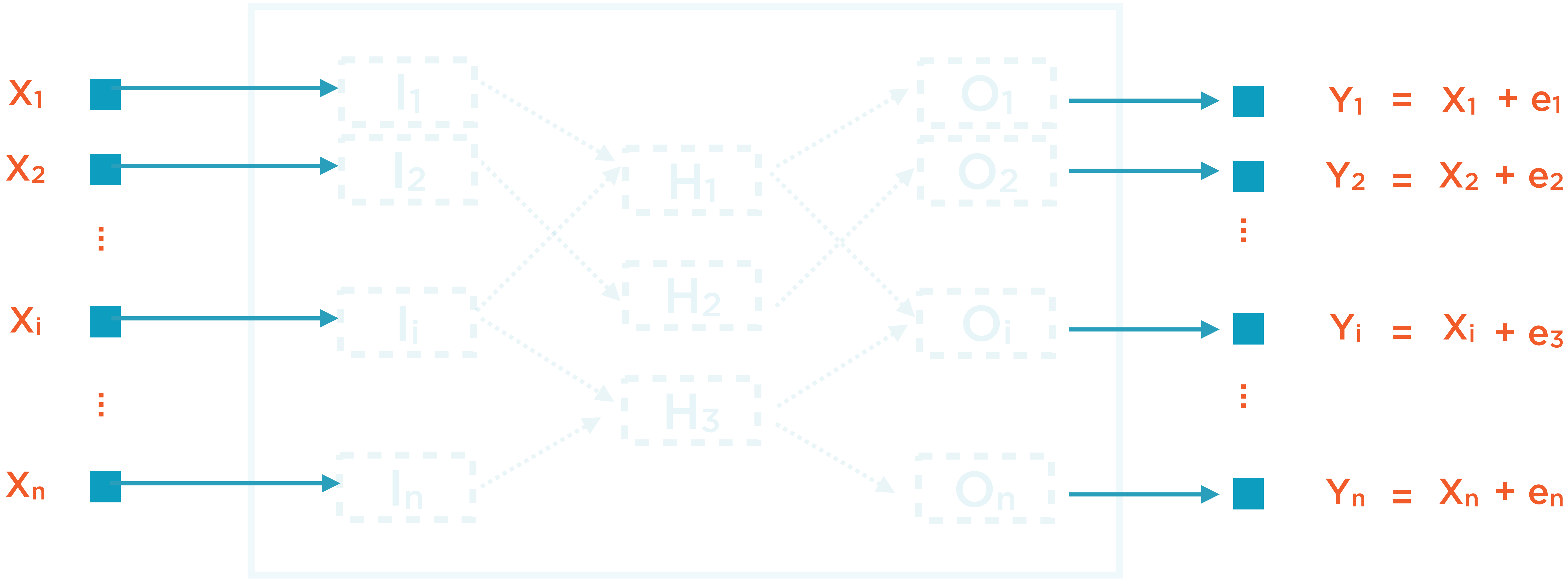
This gives undercomplete autoencoders a characteristic sandwich-like appearance

Undercomplete Autoencoder



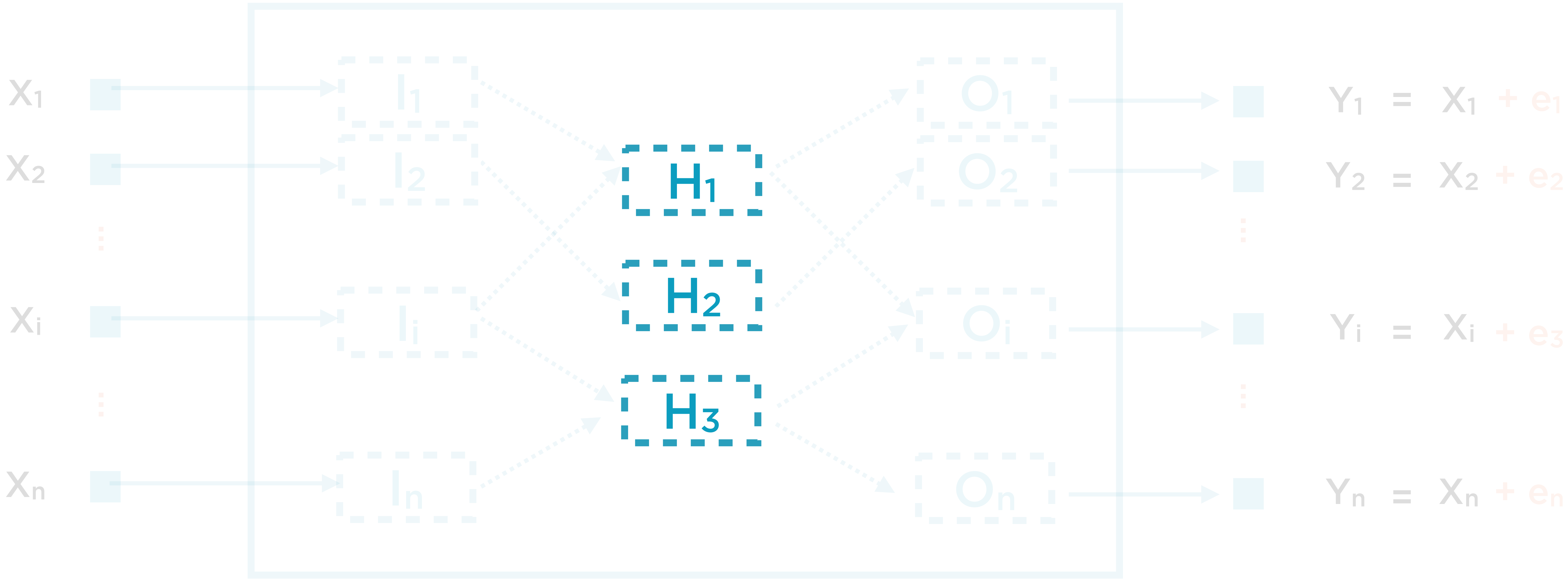
This gives undercomplete autoencoders a characteristic sandwich-like appearance

Undercomplete Autoencoder



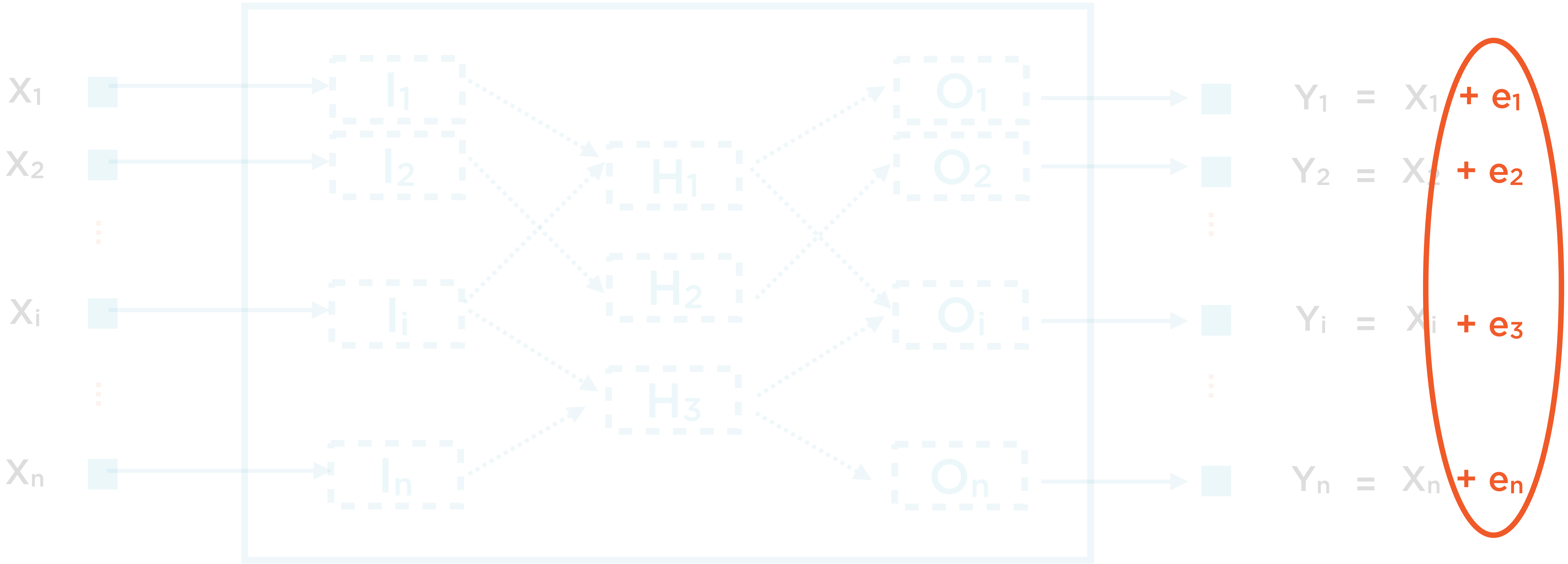
The undercomplete autoencoder will try to exactly match the input, but it will likely not succeed completely

Undercomplete Autoencoder



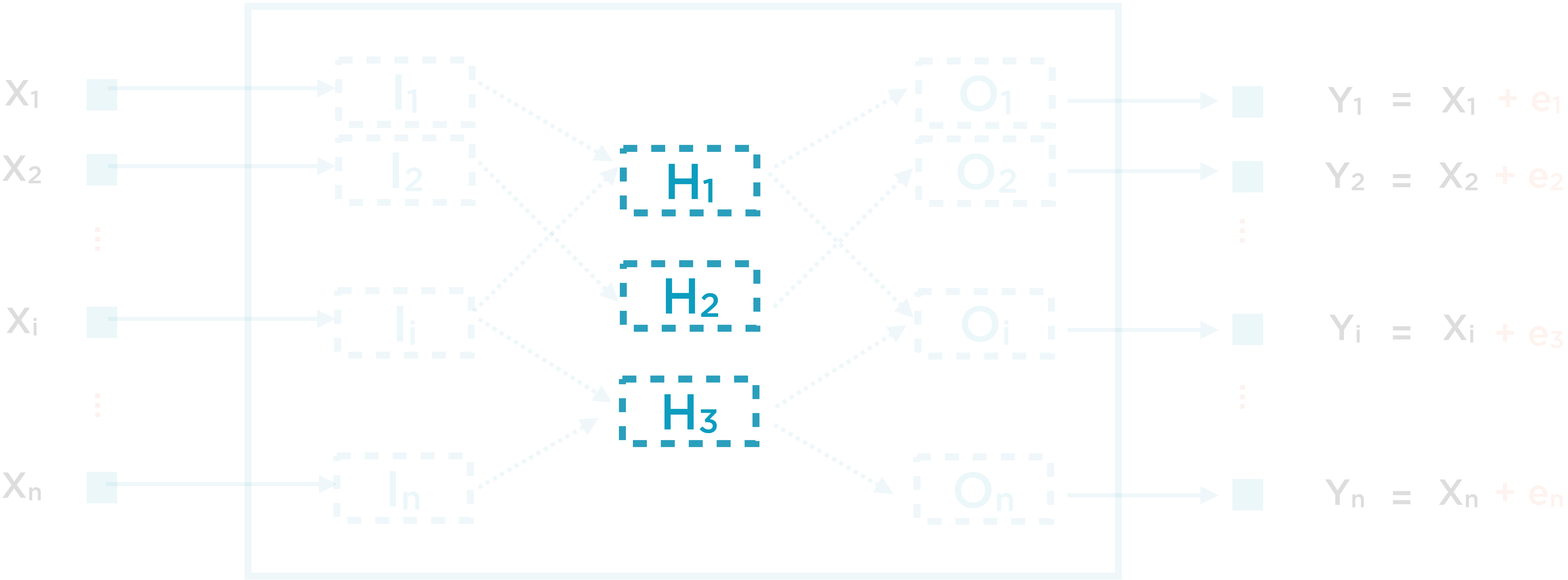
Now, because of the dimensionality reduction, output will not be exactly same as input

Undercomplete Autoencoder



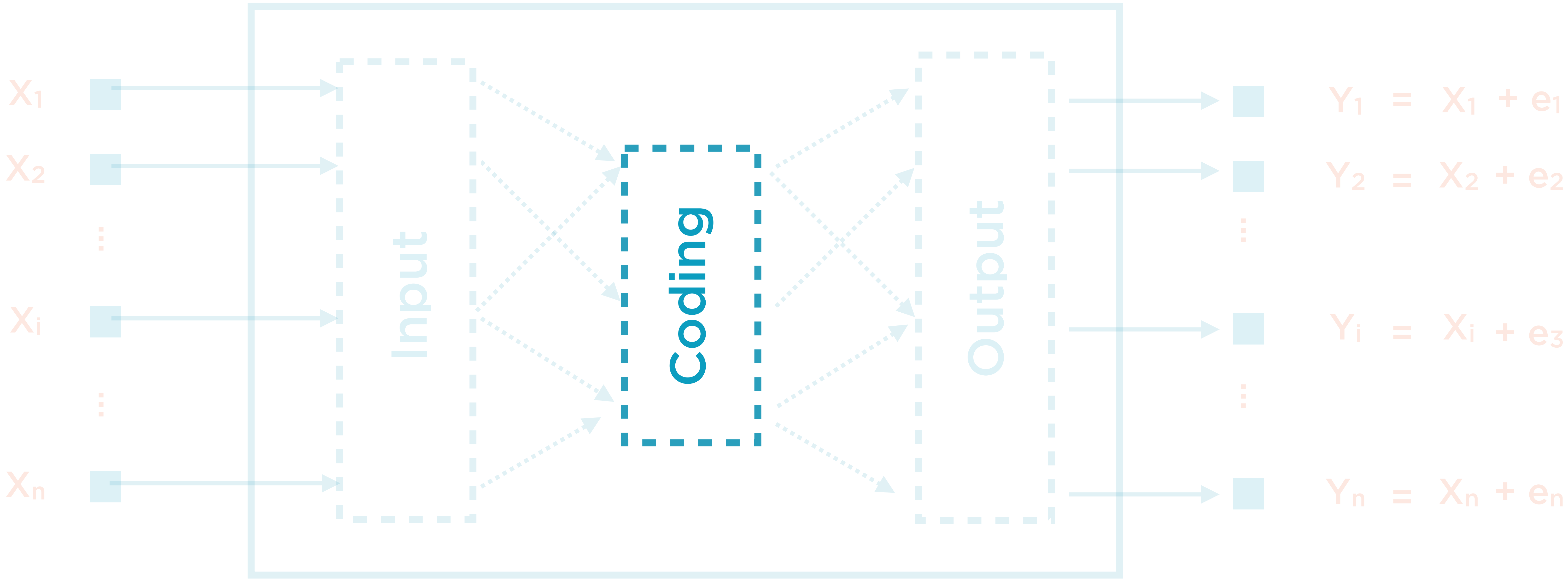
A reconstruction error will now exist

Undercomplete Autoencoder



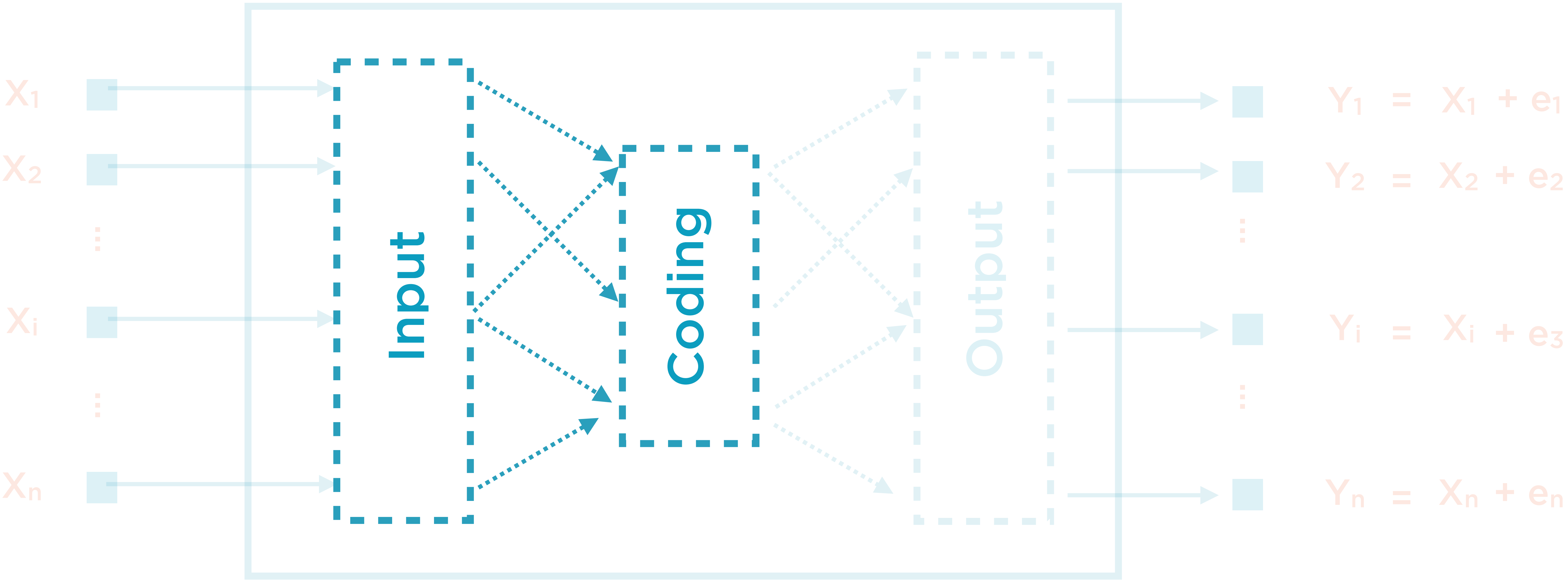
The autoencoder will be forced to learn the most significant characteristics of the data i.e. latent factors

Undercomplete Autoencoder



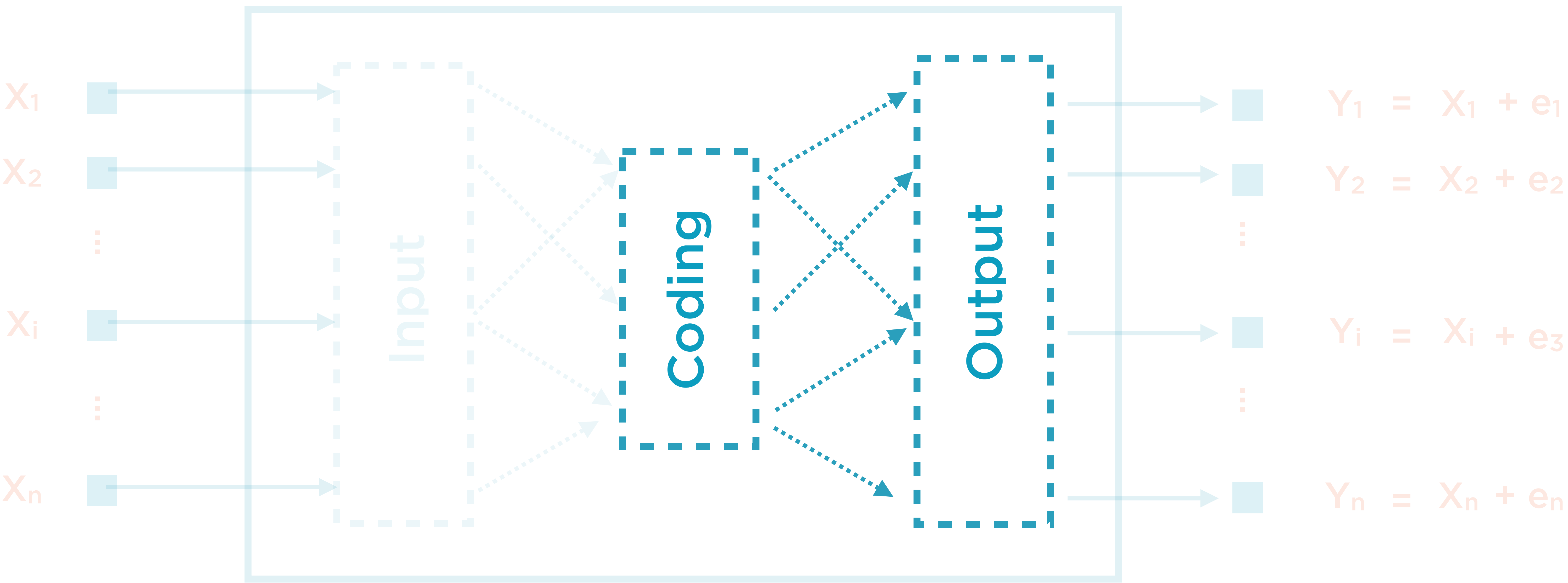
The central hidden layer is called the coding layer

Undercomplete Autoencoder



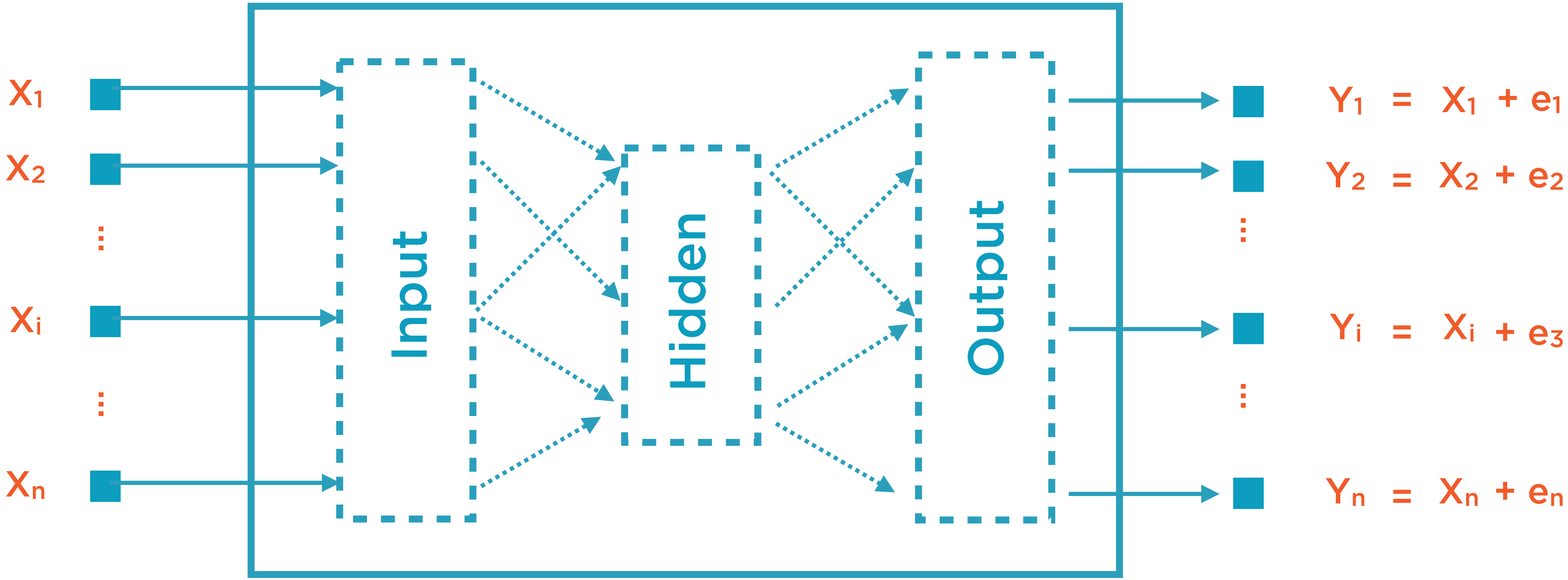
The first phase - from input to coding - is called encoding

Undercomplete Autoencoder



The second phase - from coding to output - is called decoding

Autoencoder



Demo

Autoencoders to learn latent factors

Summary

Dictionary learning for dimensionality reduction of image data

Feature detection using convolutional layers

Autoencoders for dimensionality reduction

Related Courses

Building Features from Numeric Data

Building Features from Nominal Data

Building Features from Text Data

Related Courses

Image Classification with PyTorch

Style Transfer with PyTorch