Dimensionality Reduction Using Clustering and Autoencoding Techniques

Janani Ravi
CO-FOUNDER, LOONYCORN

www.loonycorn.com

Overview

Clustering and autoencoding are classic unsupervised learning techniques

Applying clustering to dimensionality reduction

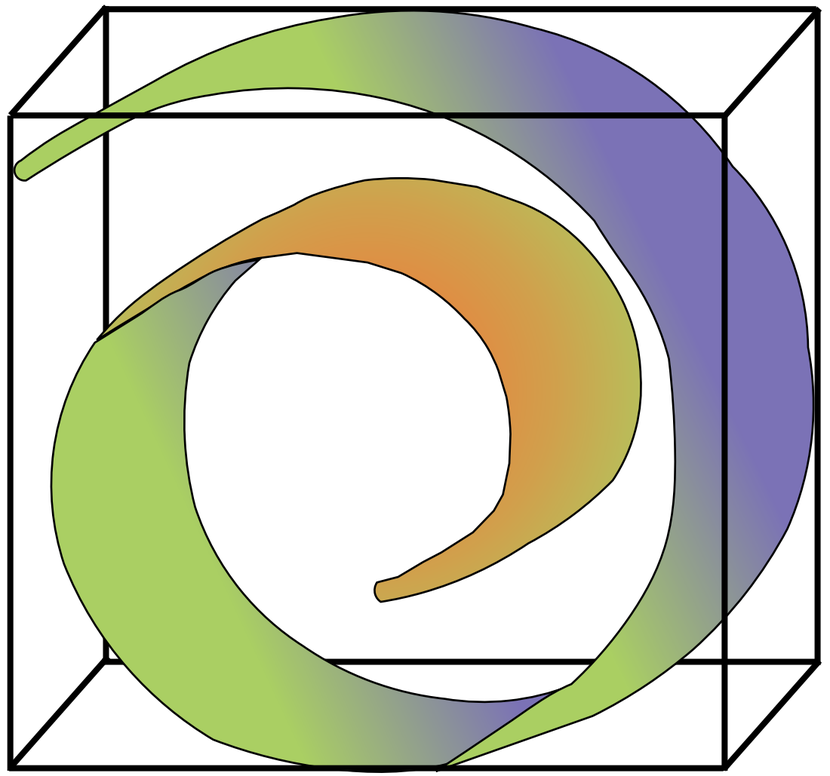
Feed the reduced output of a clustering model to a classification model

Autoencoding as a dimensionality reduction problem

Clustering for Dimensionality Reduction

Manifold Hypothesis:
Very complicated data is often not that complicated after all

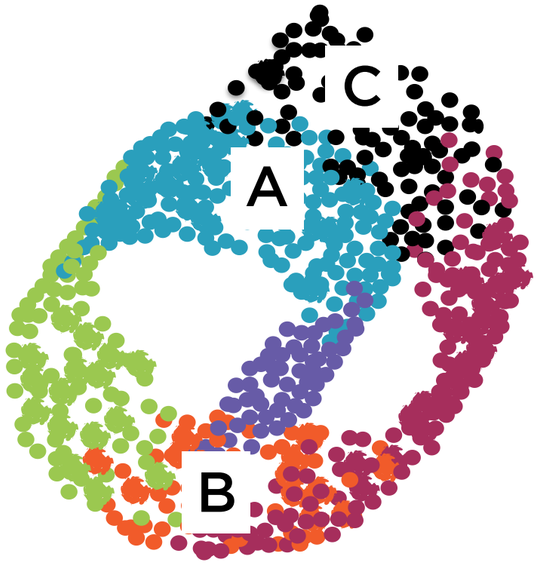
Manifold Data



Manifold Hypothesis

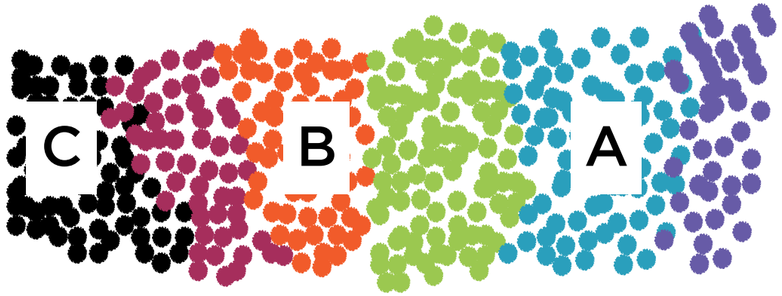
Many high-dimensional datasets can be easily unrolled so that they lie along a much lower dimensional manifold

Manifold Hypothesis



High-dimensional data

Manifold Learning

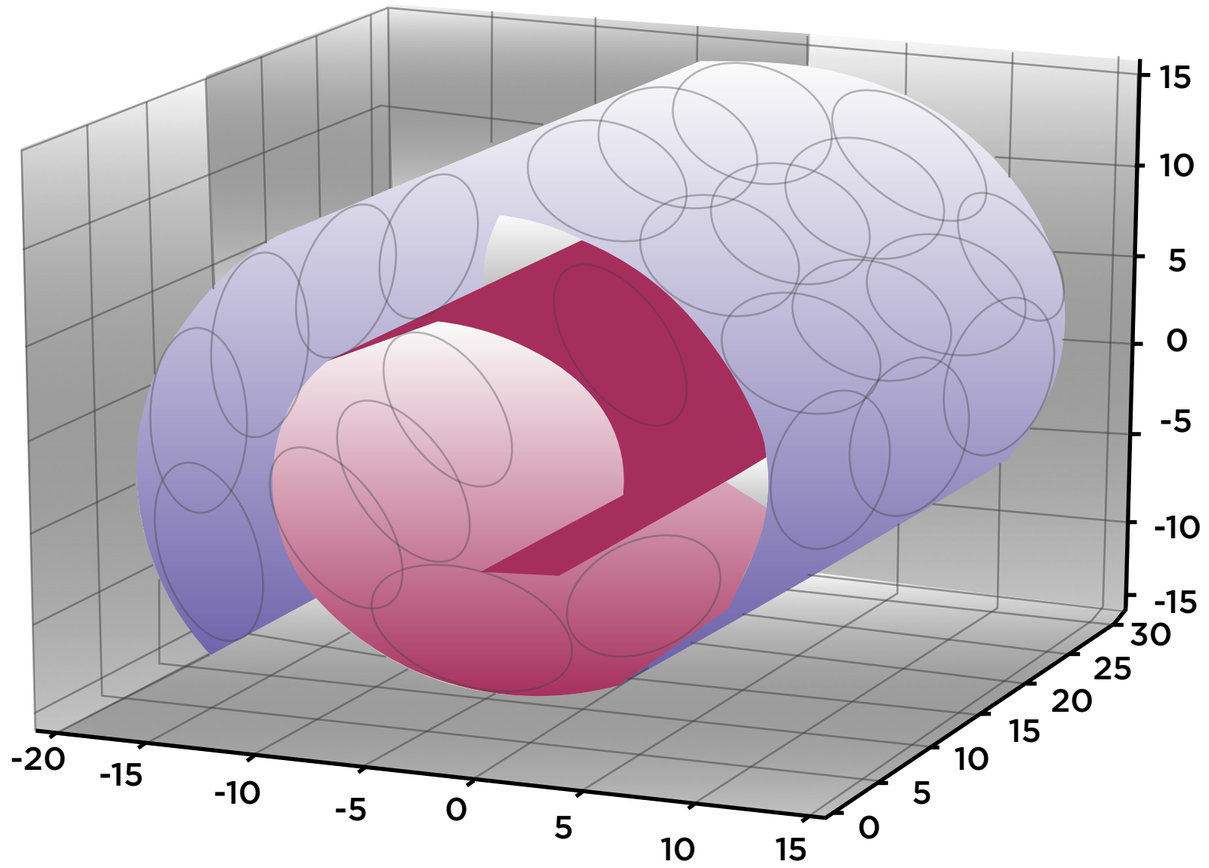


Low-dimensional embedding

K-means Model Stacking

Several manifold learning techniques seek to preserve distances from subset of points. Simply extend that idea: pick K centroids obtained from clustering.

K-means for Manifold Learning



Simply re-express each point as tuple of distances from K centroids!

Demo

Implement dimensionality reduction using K-means clustering

Autoencoding

Choosing Autoencoders

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

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

Supervised Machine Learning

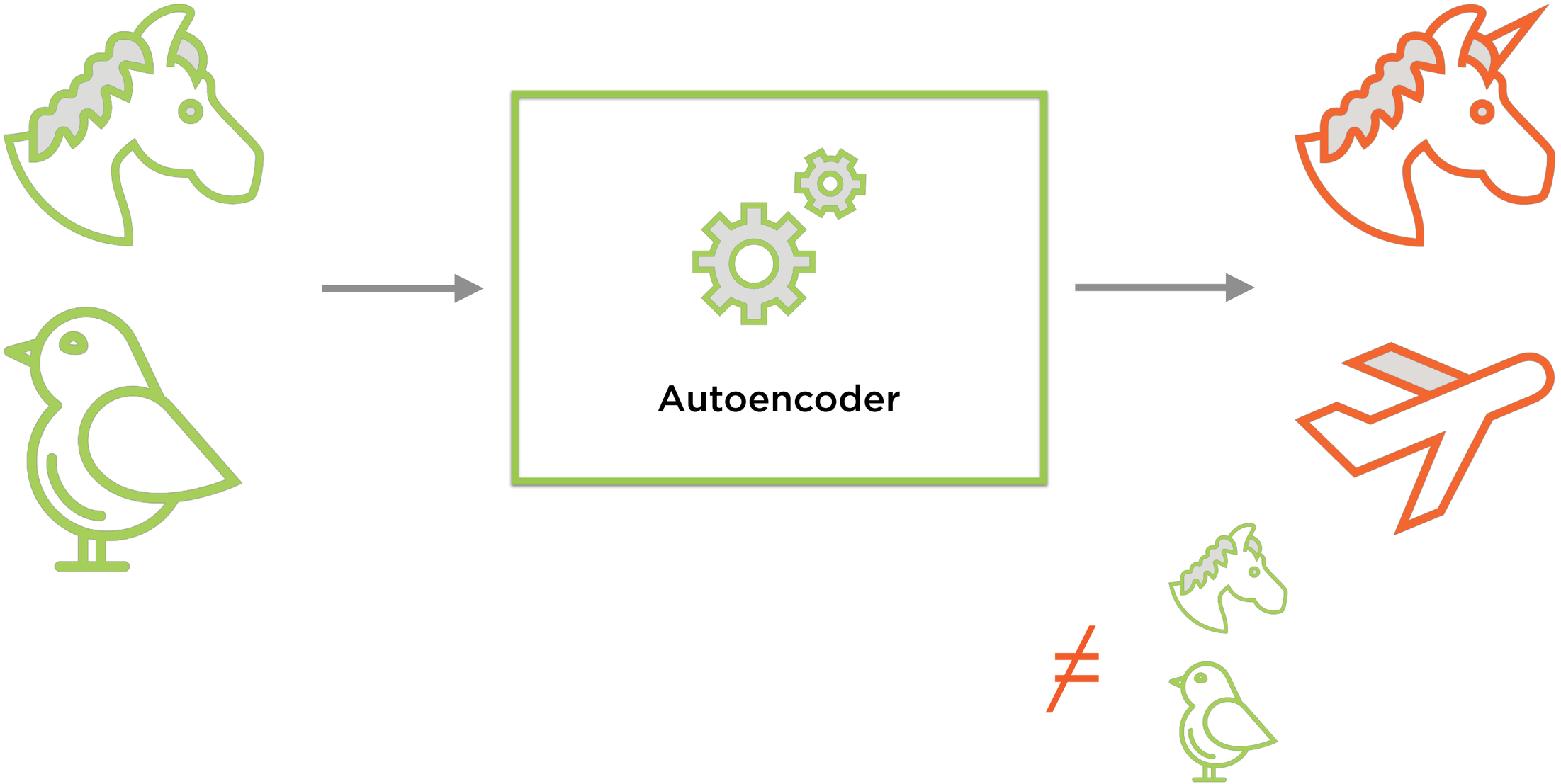
Most machine learning algorithms seek to “learn” the function f that links the features and the labels

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

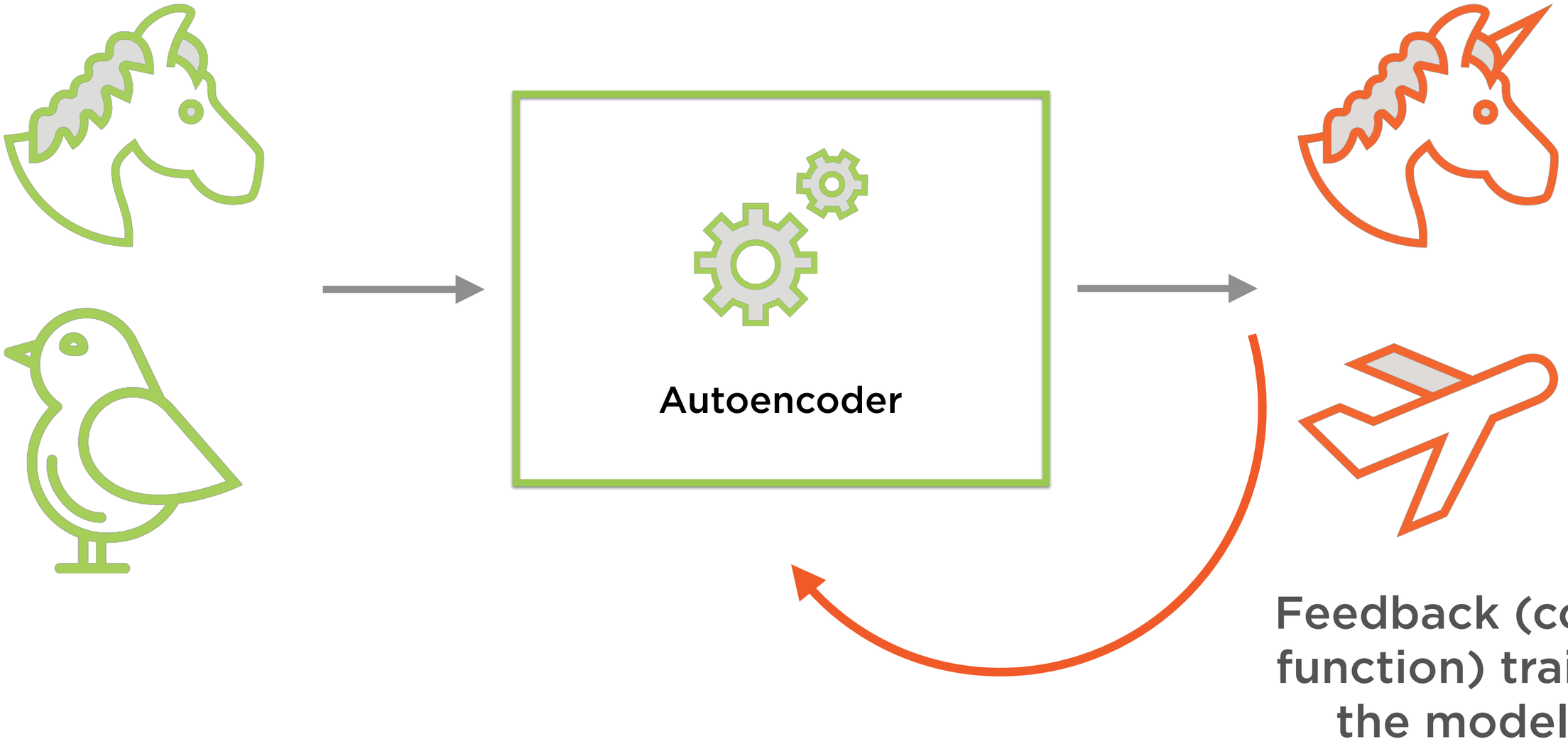
Autoencoders Learn the Input!

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

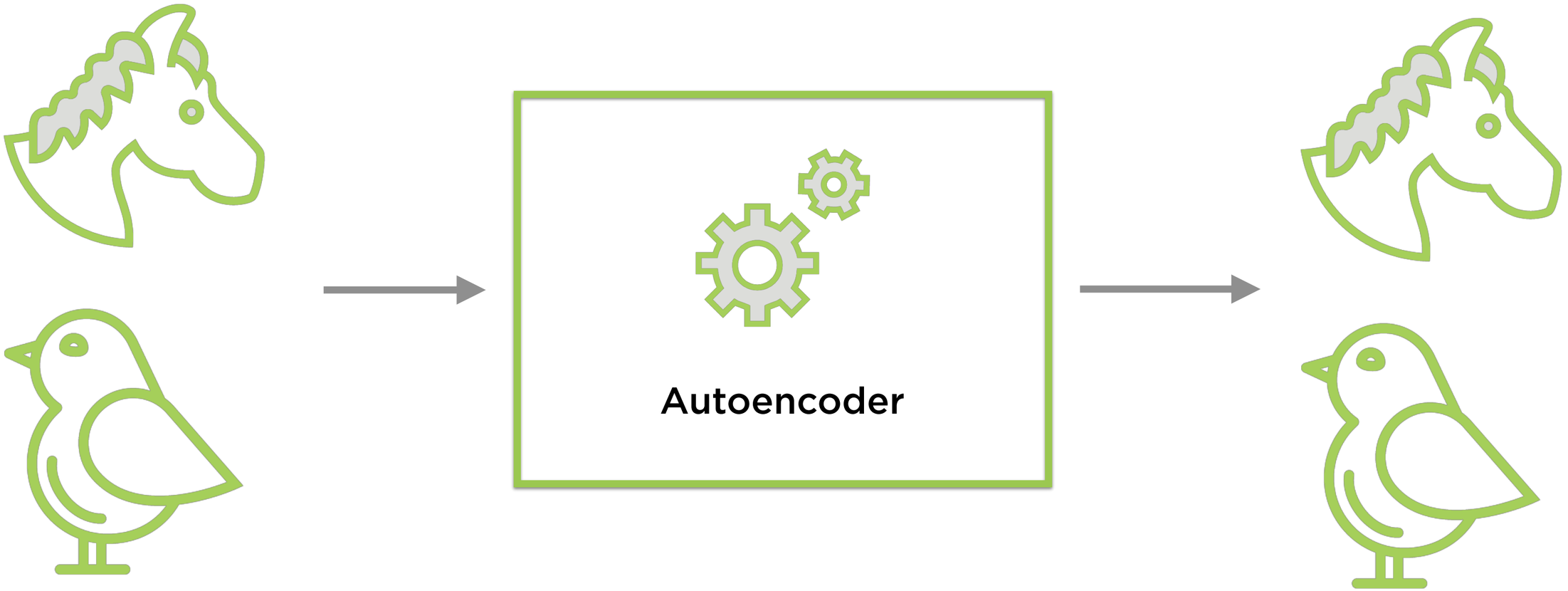
Autoencoder



Autoencoder

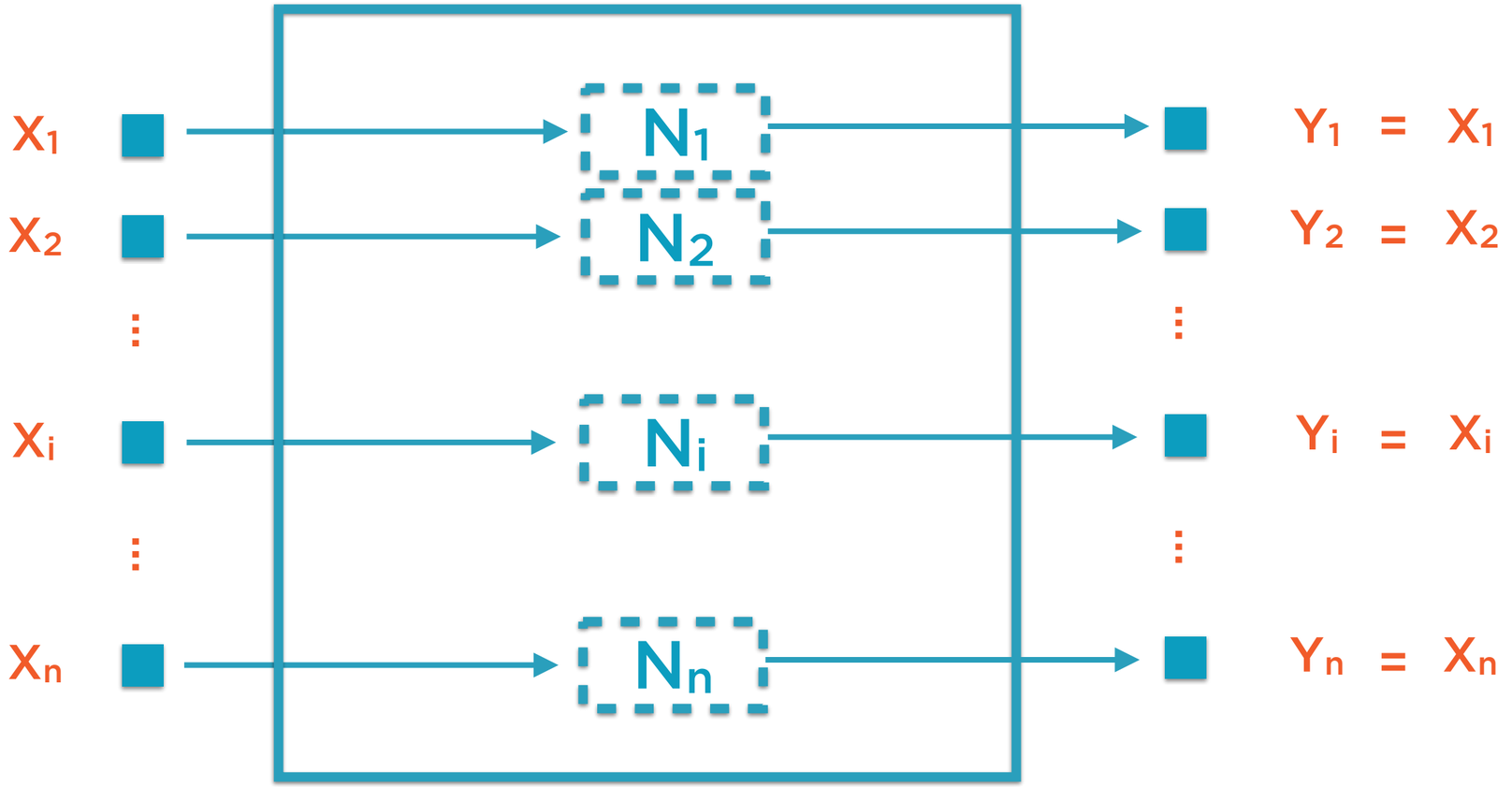


Autoencoder



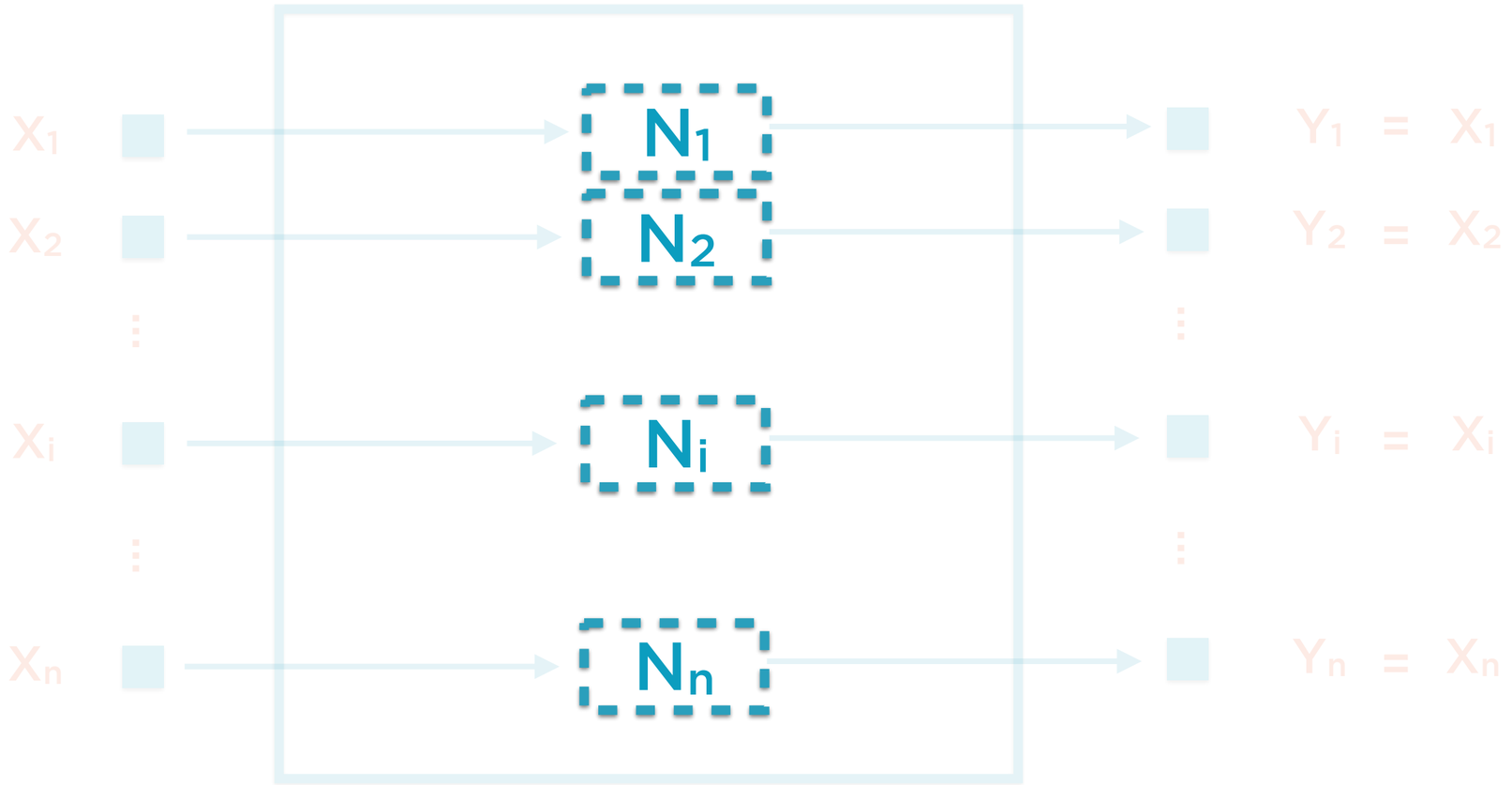
Autoencoders are Neural Networks that learn efficient representations of data (e.g. PCA)

Trivial Autoencoder



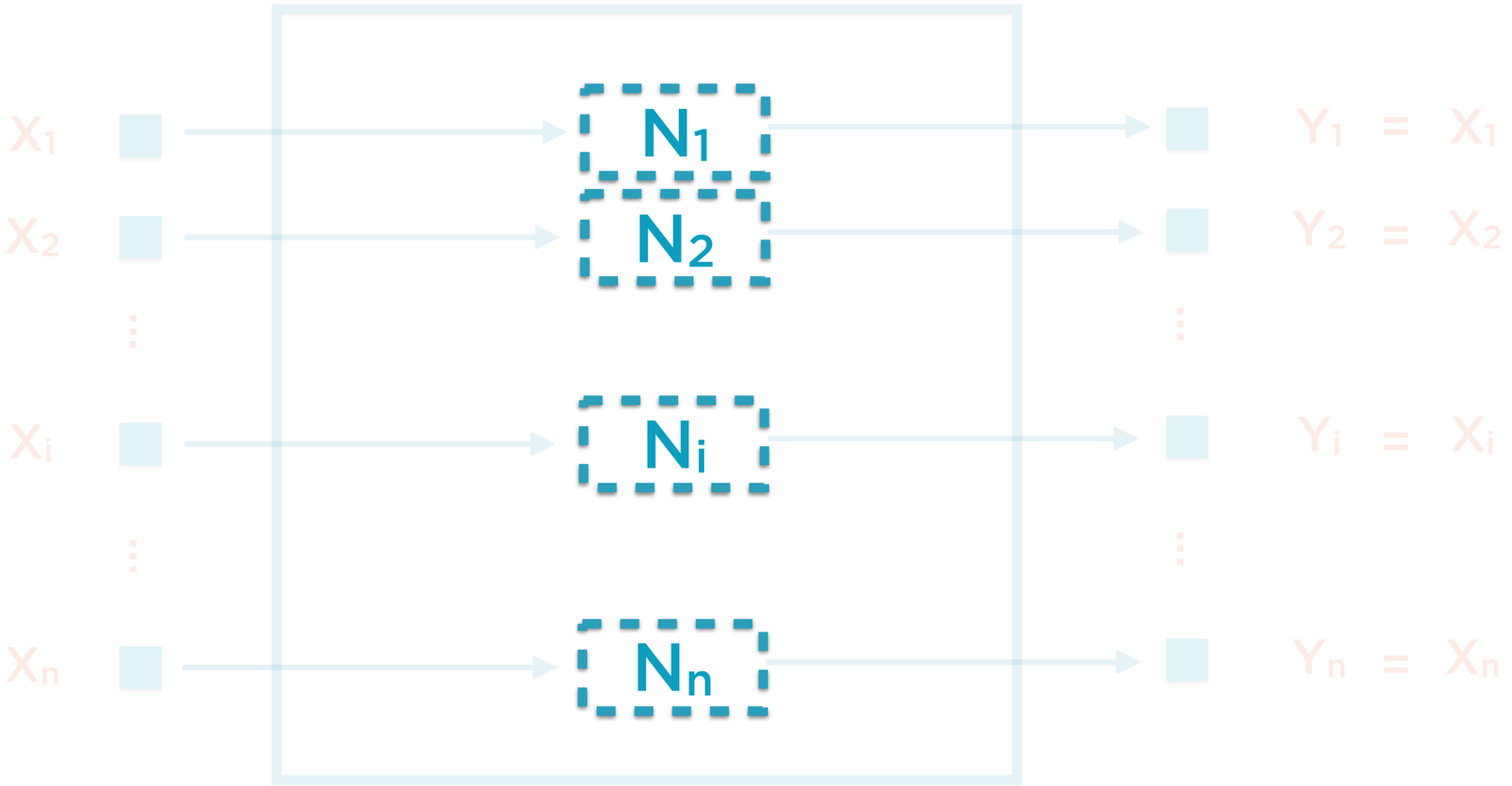
This Neural Network trivially “learns” the input

Trivial Autoencoder



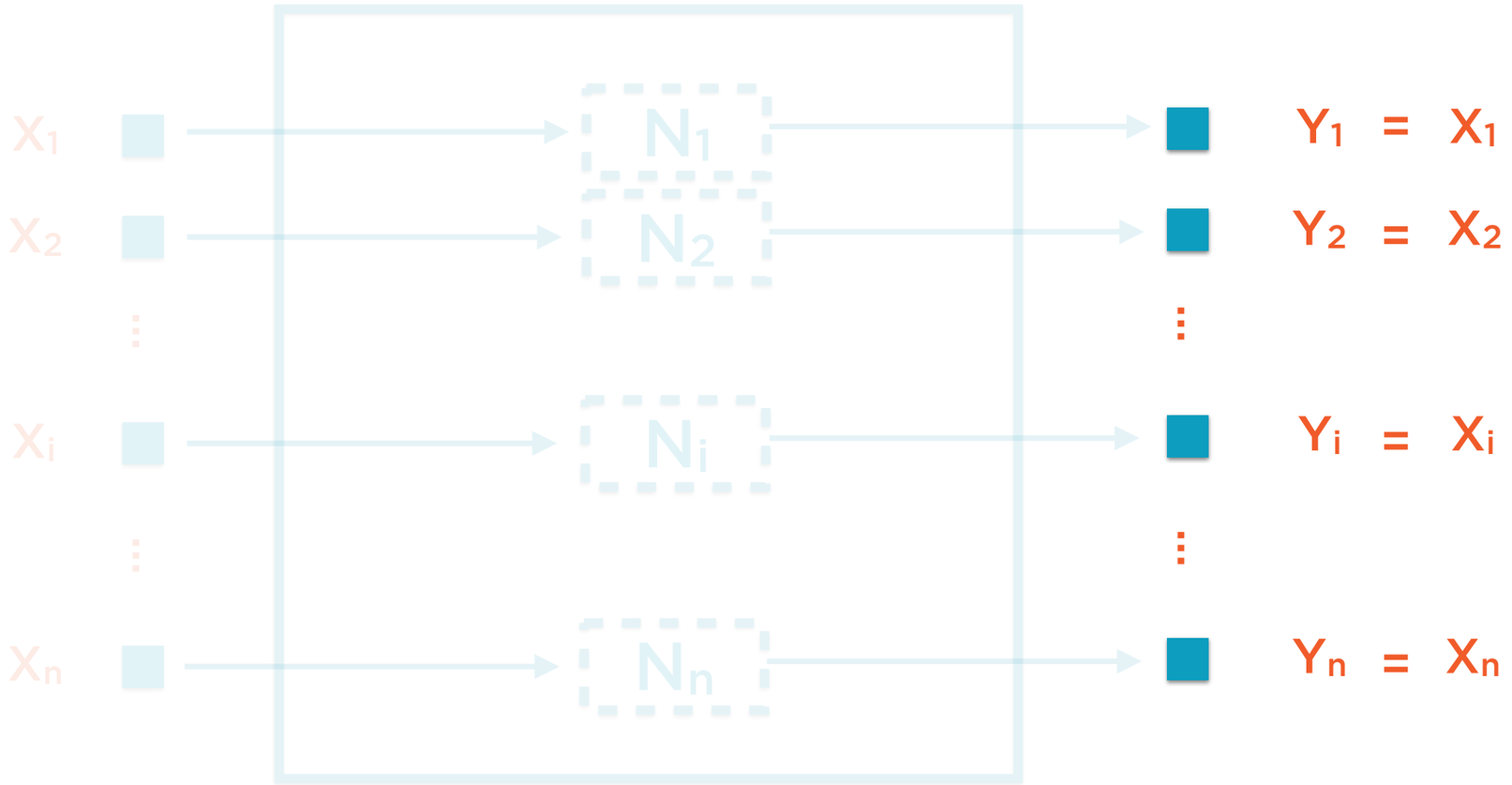
Just one layer, which serves as both input and output layer

Trivial Autoencoder



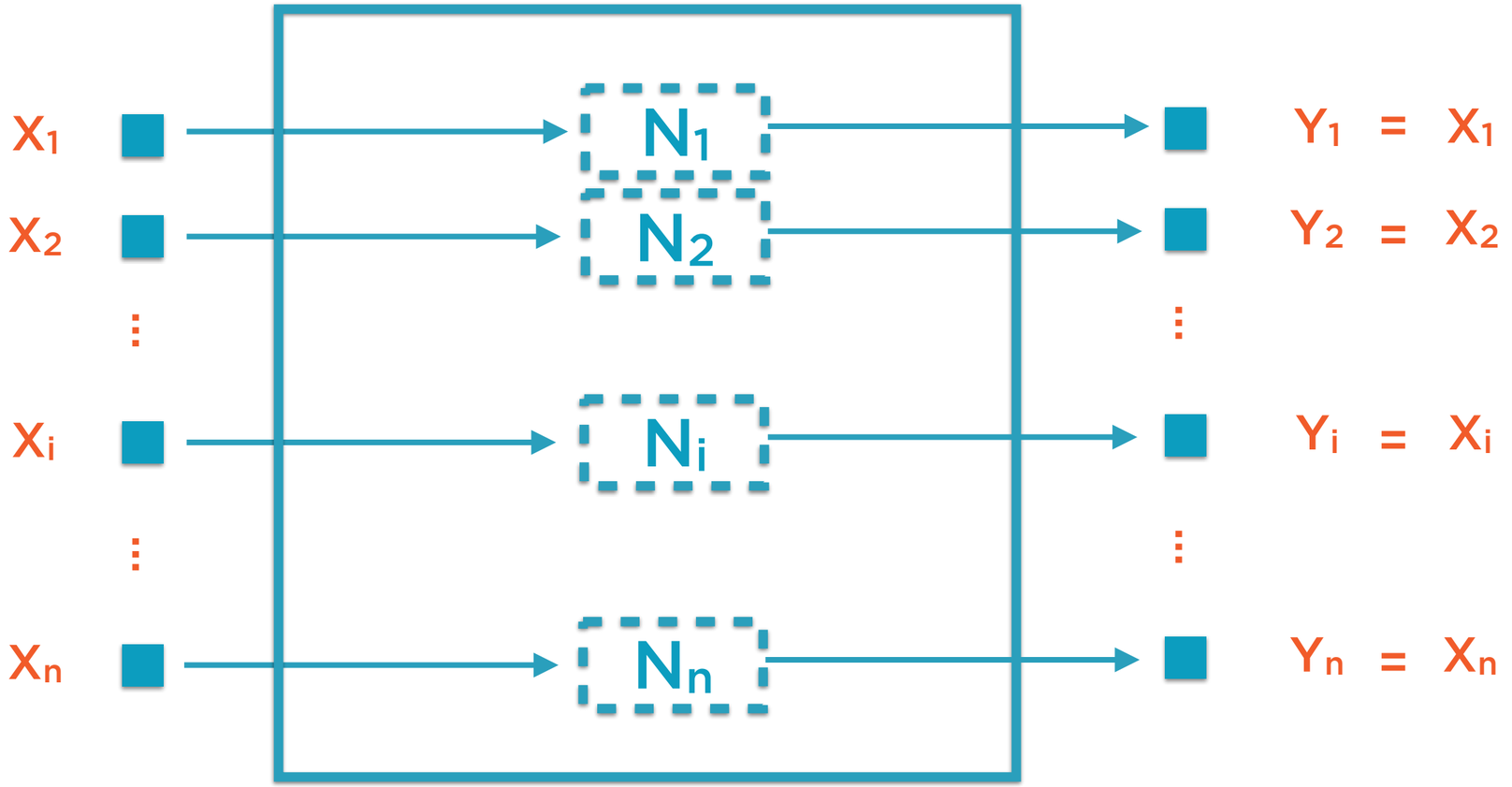
In an autoencoder, the input and output layer must have same dimensionality as the input data

Trivial Autoencoder



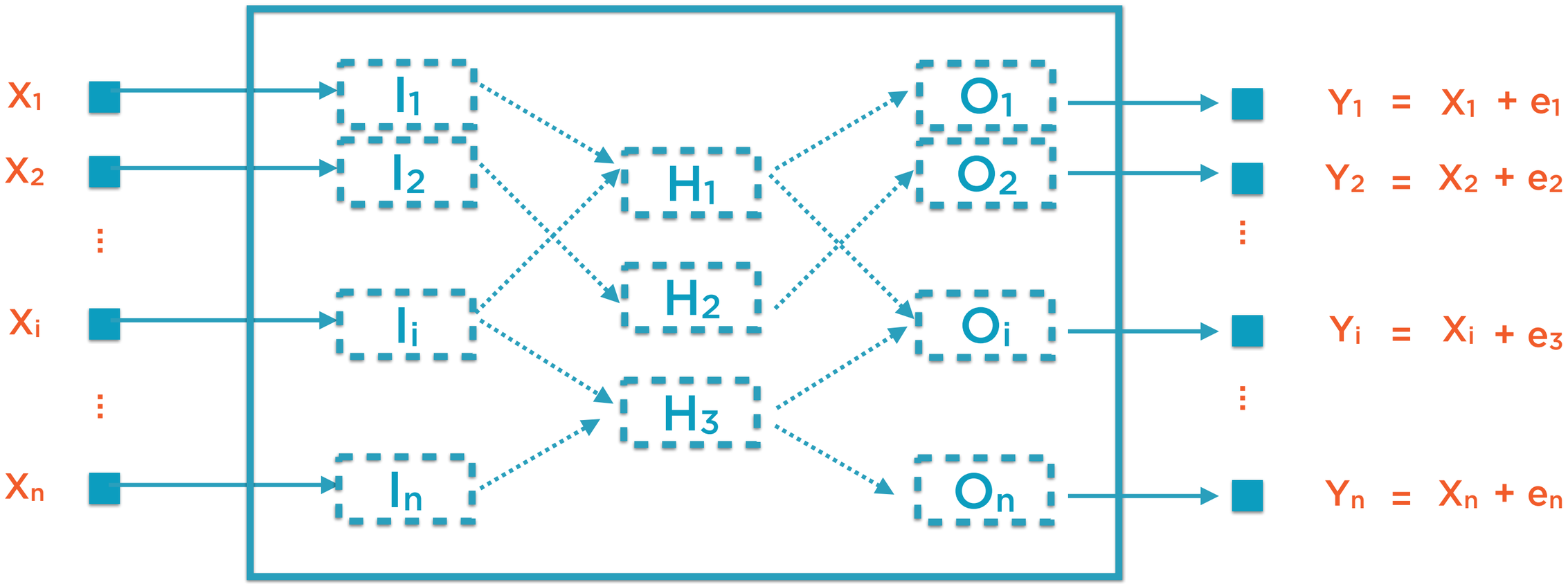
The autoencoder just passes the input through, so output is exactly equal to input

Trivial Autoencoder



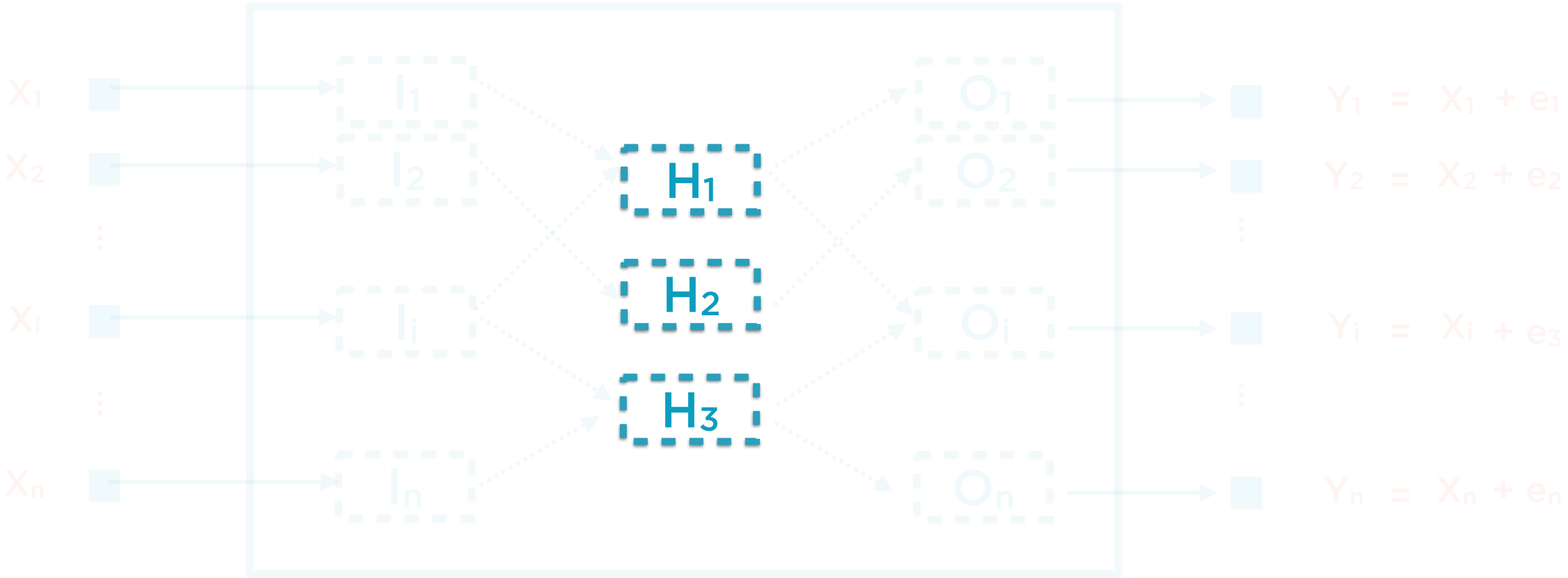
Now, let's constrain the network to force dimensionality reduction

Undercomplete Autoencoder



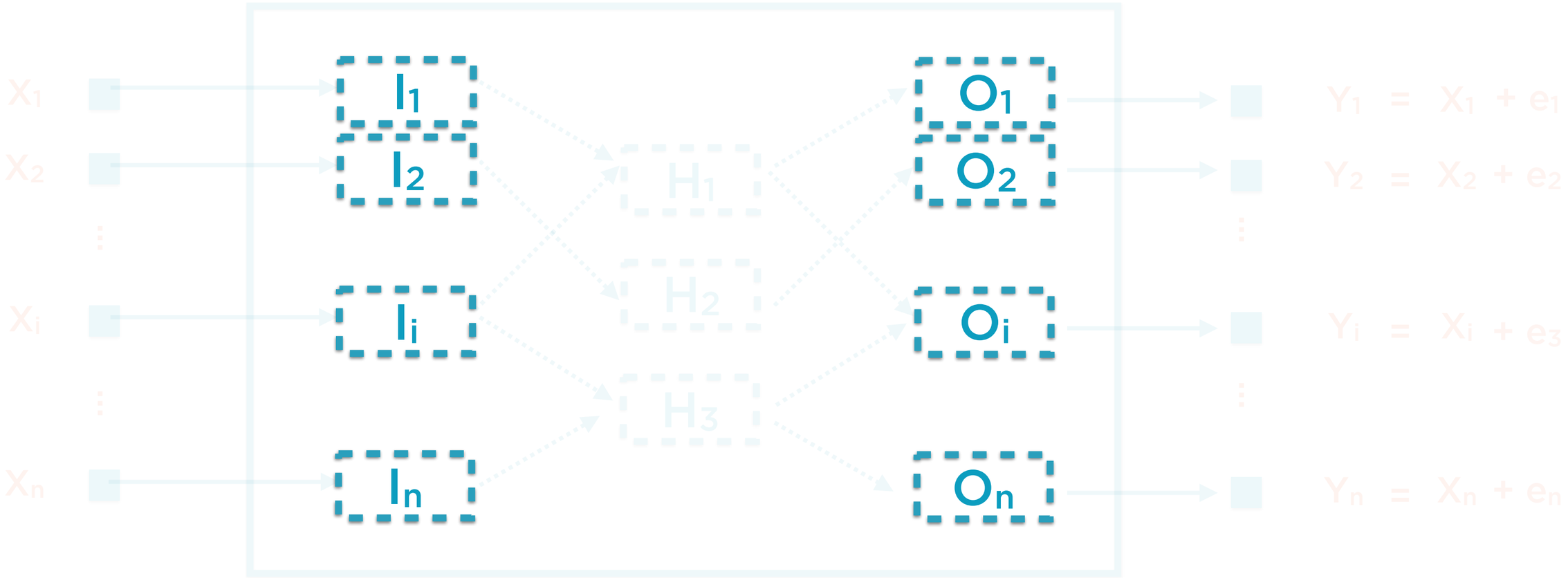
Dimensionality of the NN is now lower than that of input data

Undercomplete Autoencoder



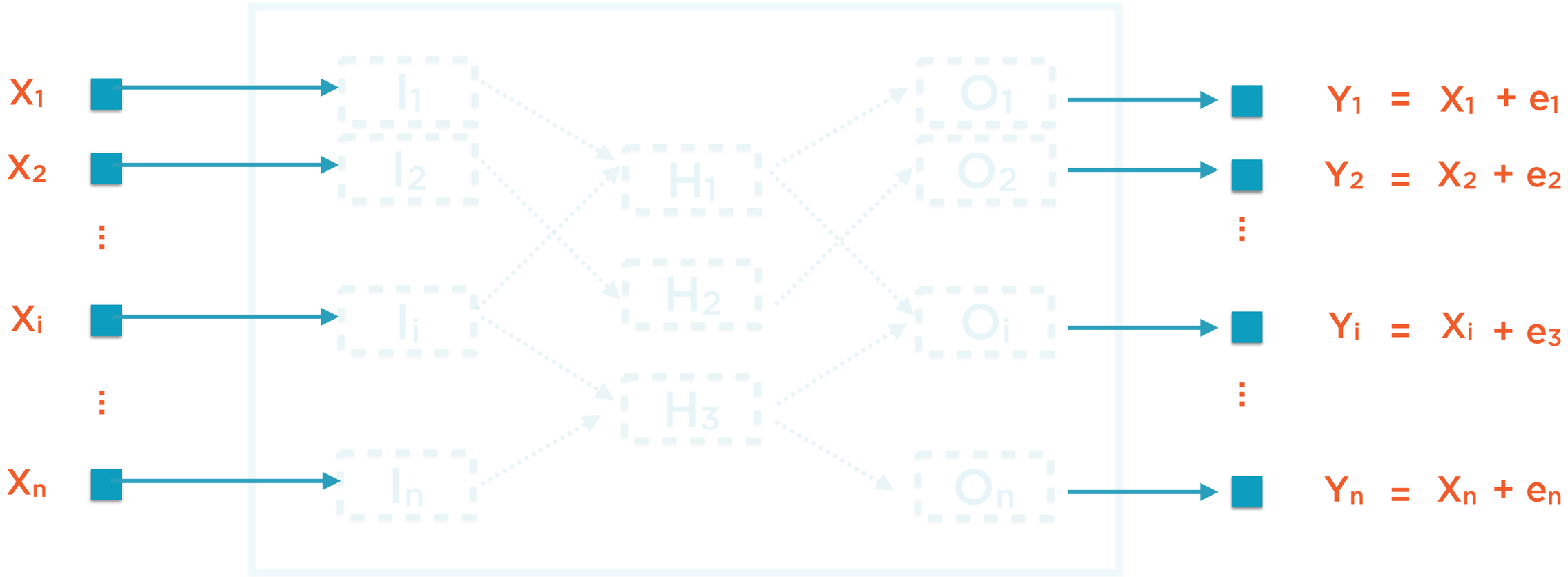
Add a middle, hidden layer with just three neurons (3 < N)

Undercomplete Autoencoder



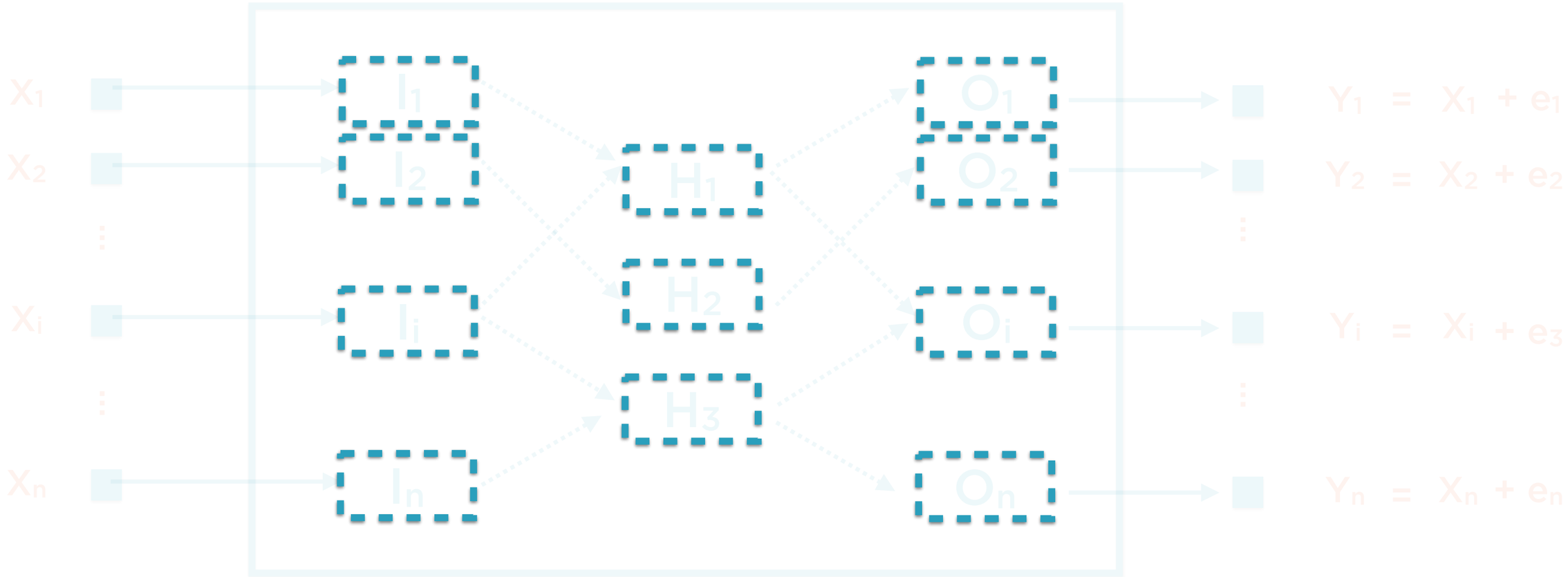
The input and output layers must now be separated, since each must still have same dimensionality as the input

Undercomplete Autoencoder



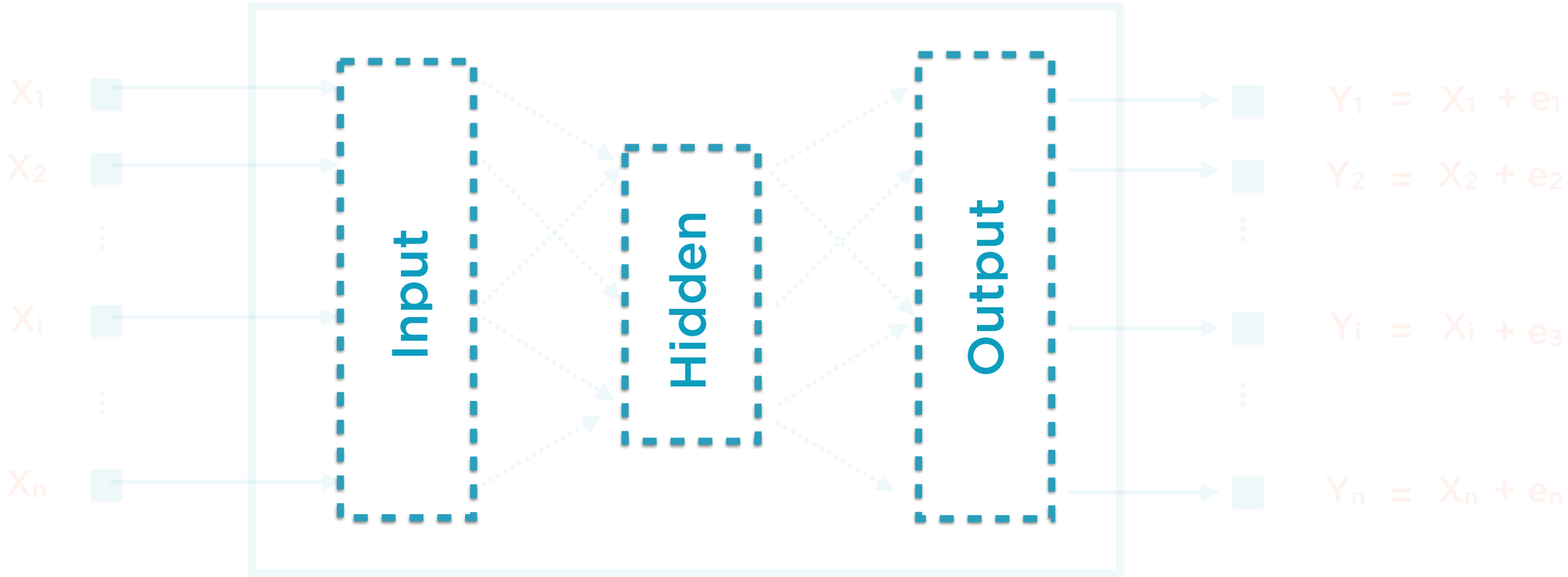
Why? Because autoencoder seeks to reconstruct input

Undercomplete Autoencoder



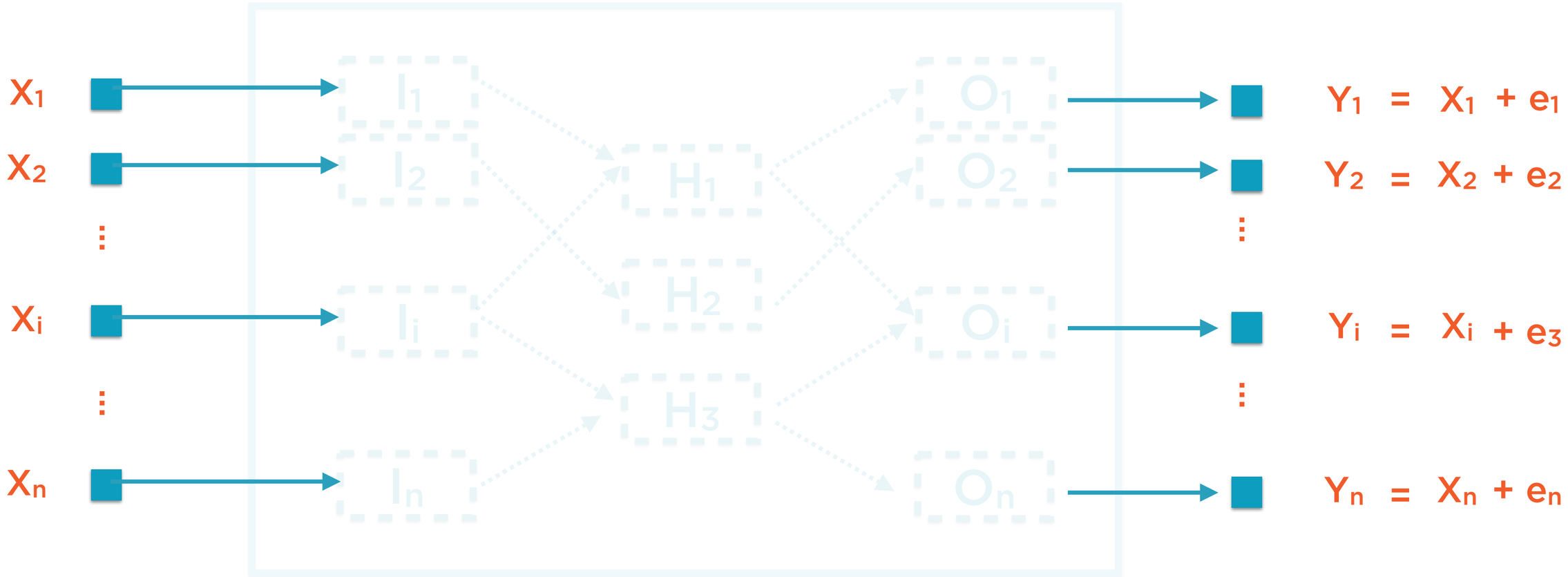
This gives undercomplete autoencoders a characteristic sandwich-like appearance

Undercomplete Autoencoder



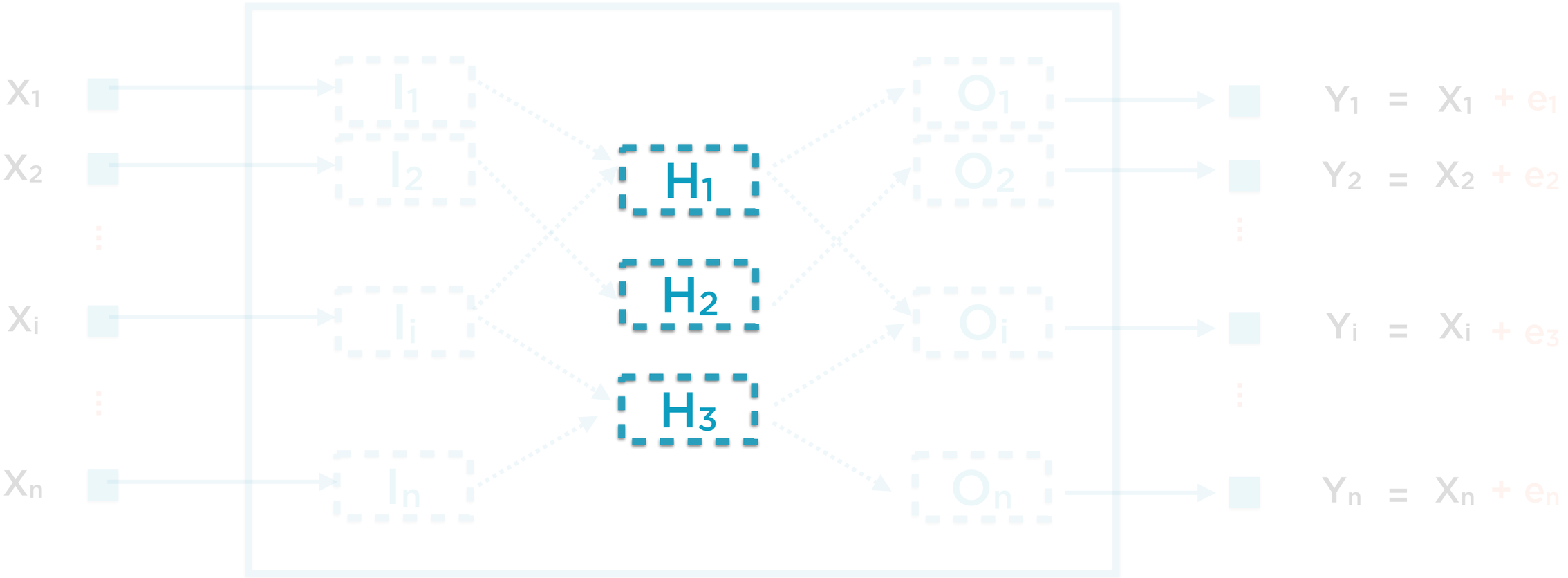
This gives undercomplete autoencoders a characteristic sandwich-like appearance

Undercomplete Autoencoder



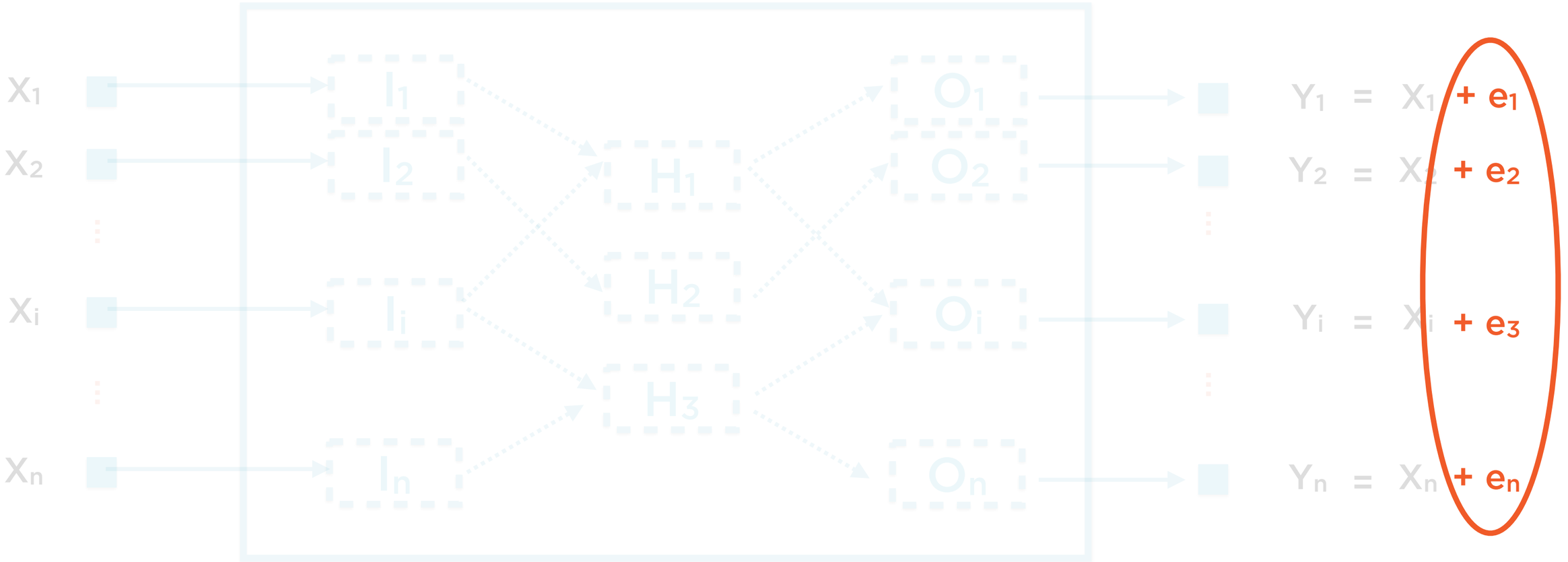
The undercomplete autoencoder will try to exactly match the input, but it will likely not succeed completely

Undercomplete Autoencoder



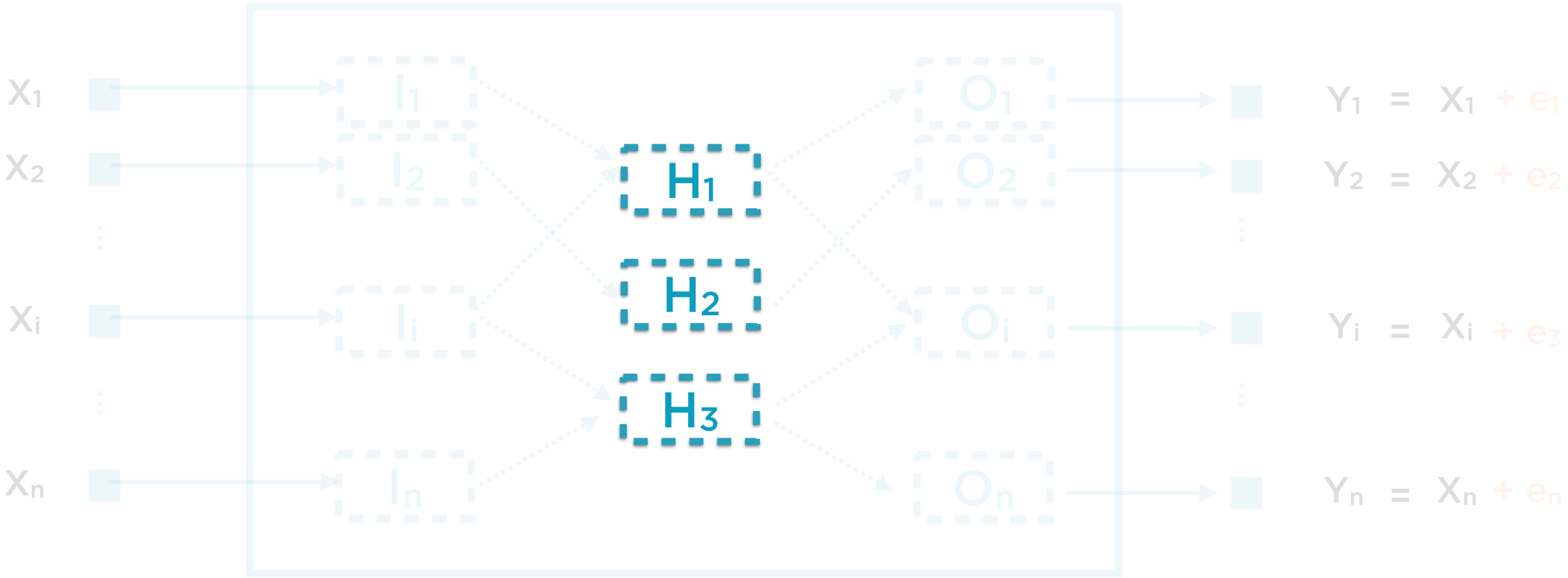
Now, because of the dimensionality reduction, output will not be exactly same as input

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

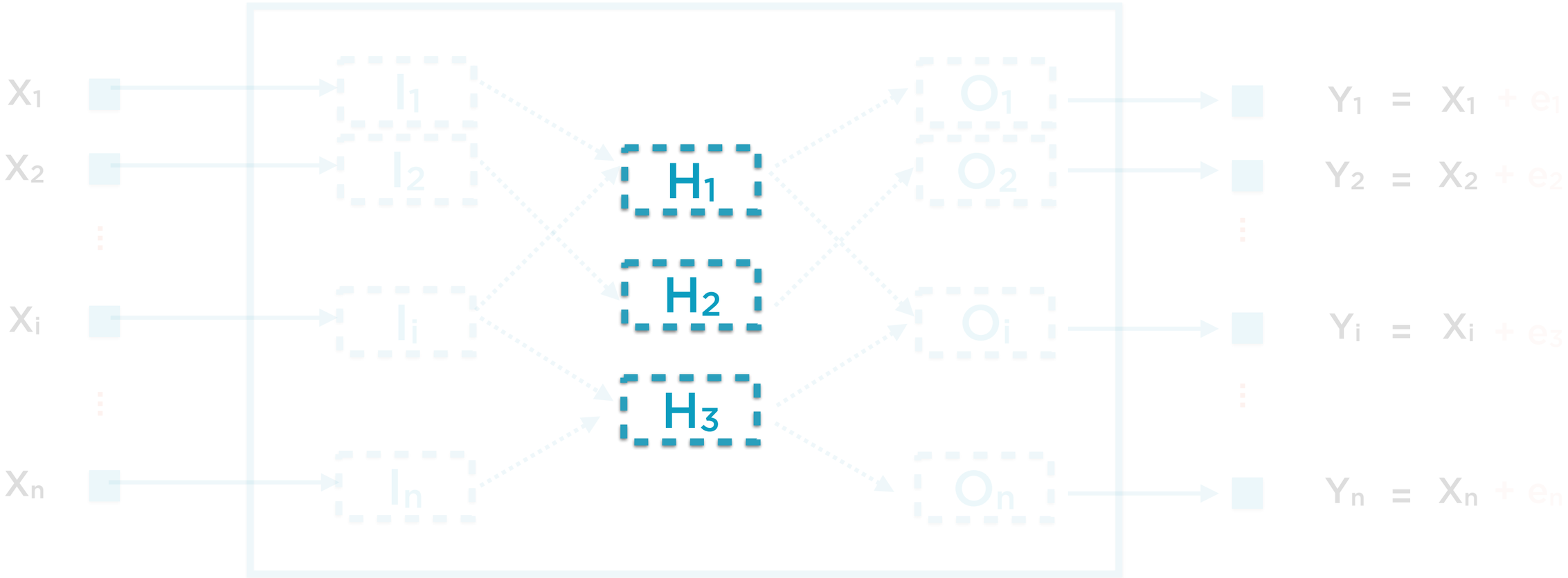
Undercomplete Autoencoder

Design choice: What cost function to use in training to minimize reconstruction error?

Undercomplete Autoencoder

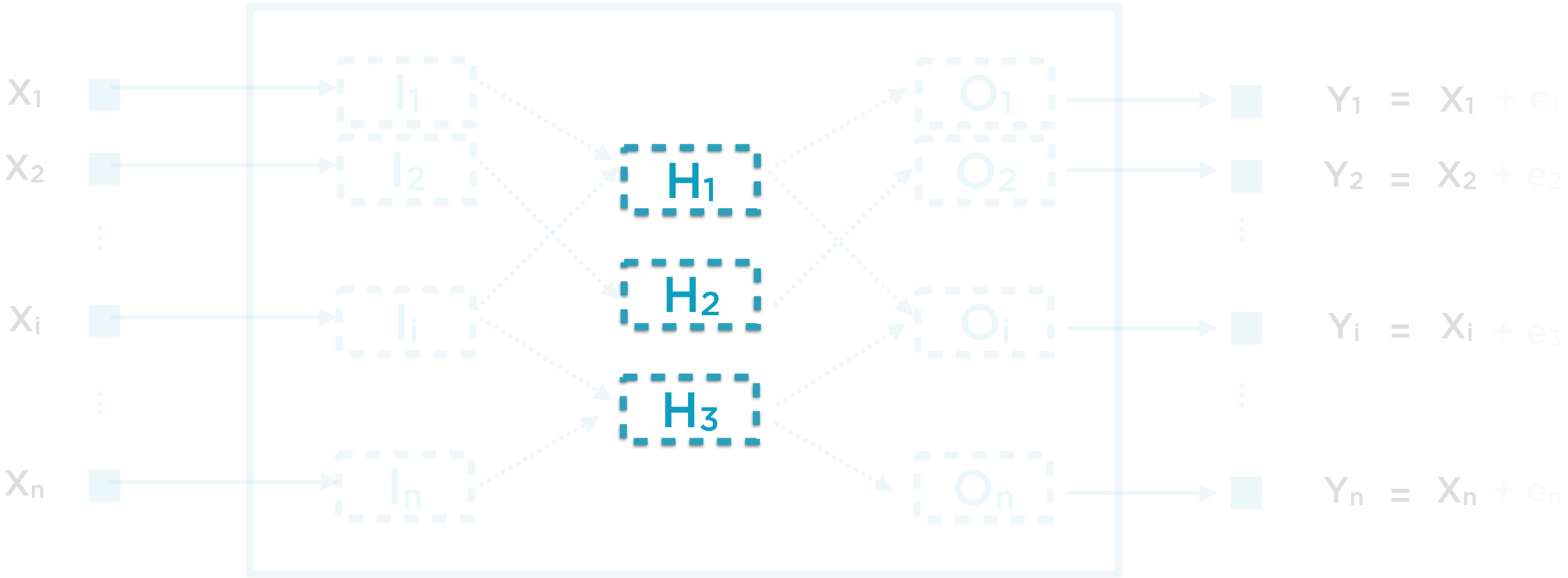
Possible choice: Minimize MSE (mean-square-error)

Undercomplete Autoencoder



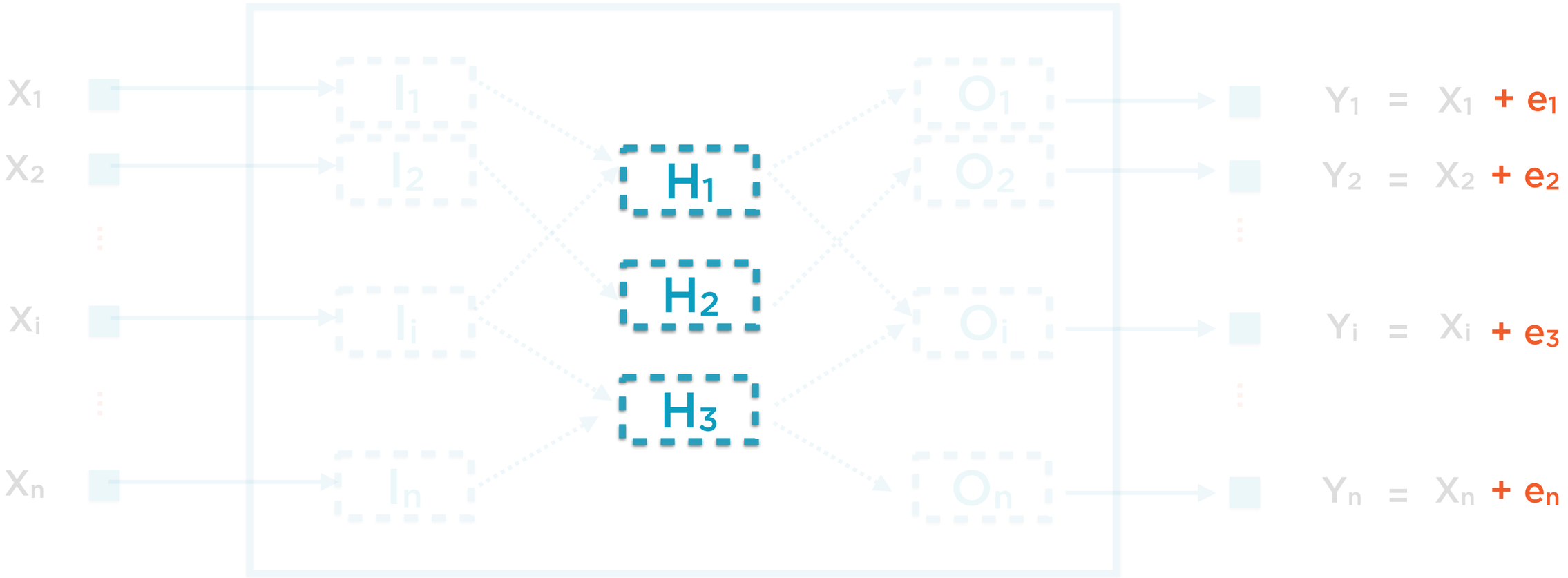
Design choice: What activation function to use for the hidden layer neurons?

Undercomplete Autoencoder



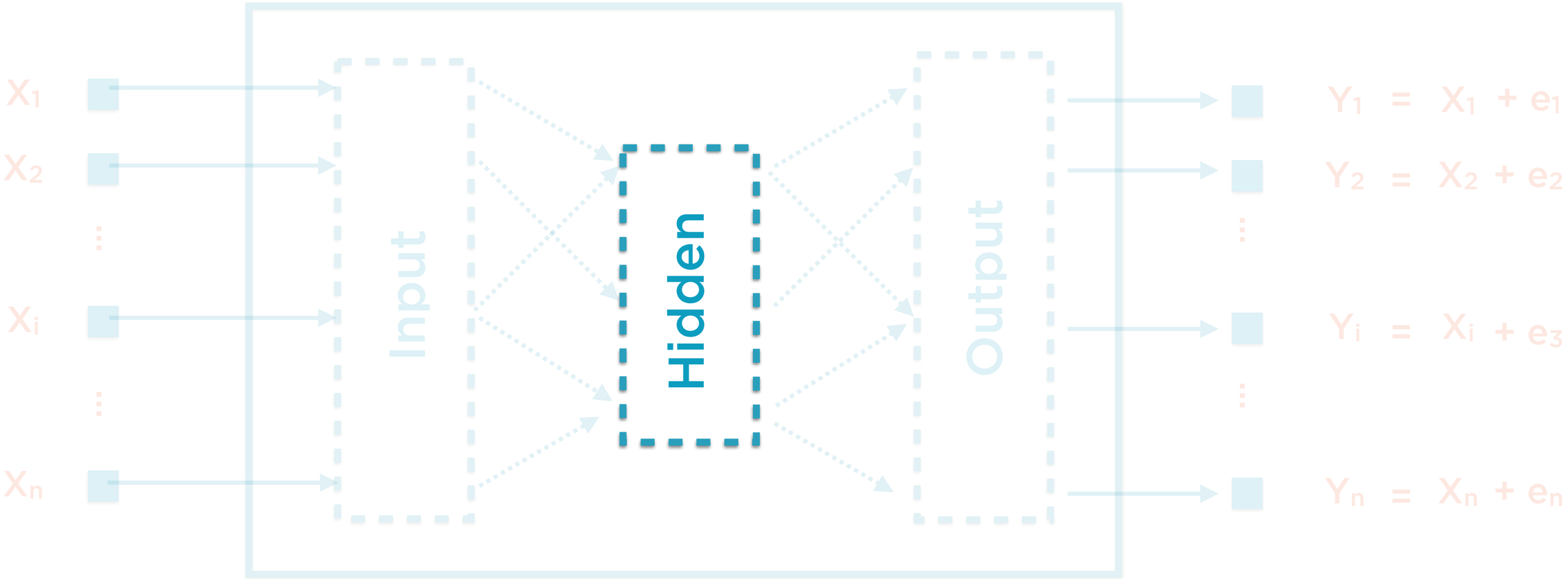
Possible choice: Use linear neuron (no activation function)

Linear Neurons + Min MSE = PCA



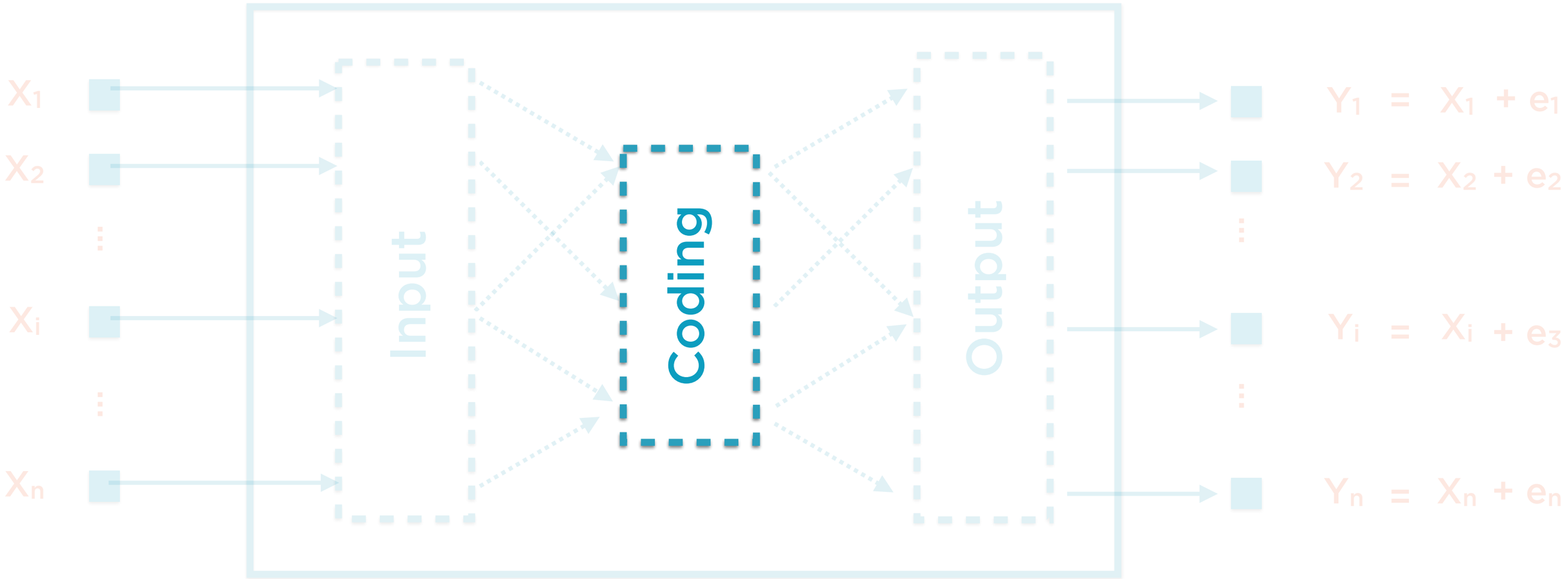
Autoencoders with linear neurons that are trained to minimise MSE will simply perform PCA

Undercomplete Autoencoder



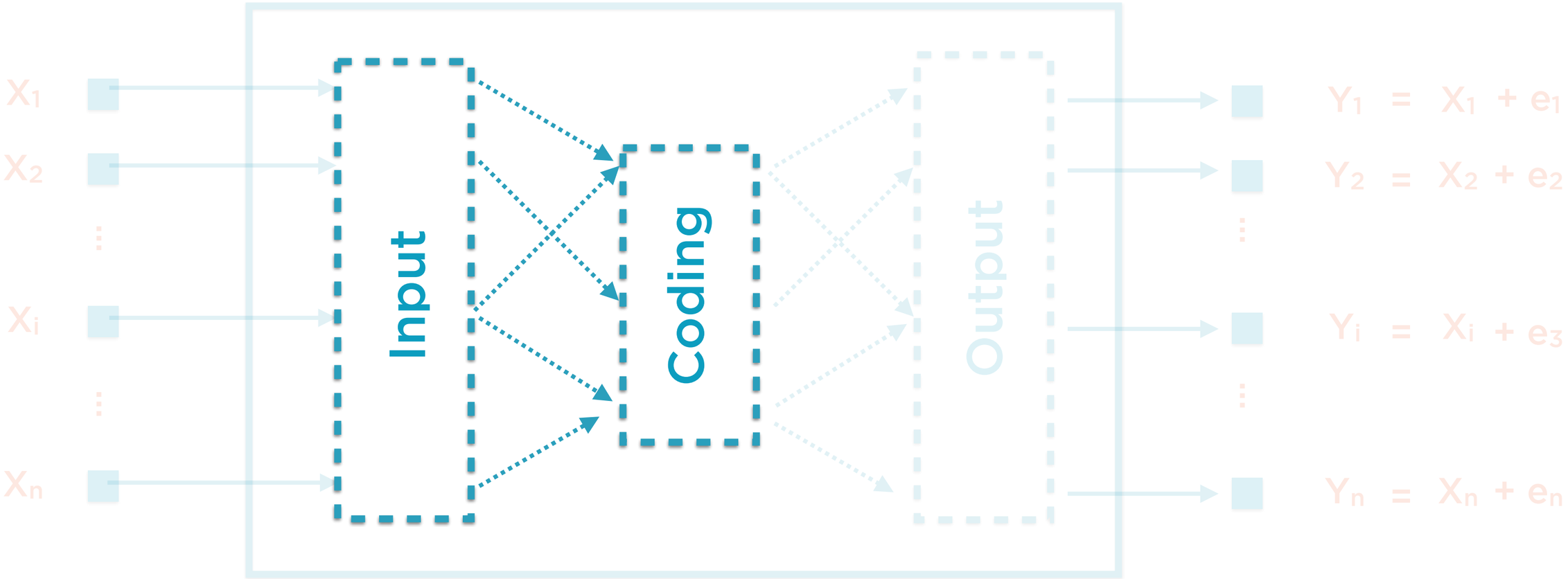
The central hidden layer is called the coding layer

Undercomplete Autoencoder



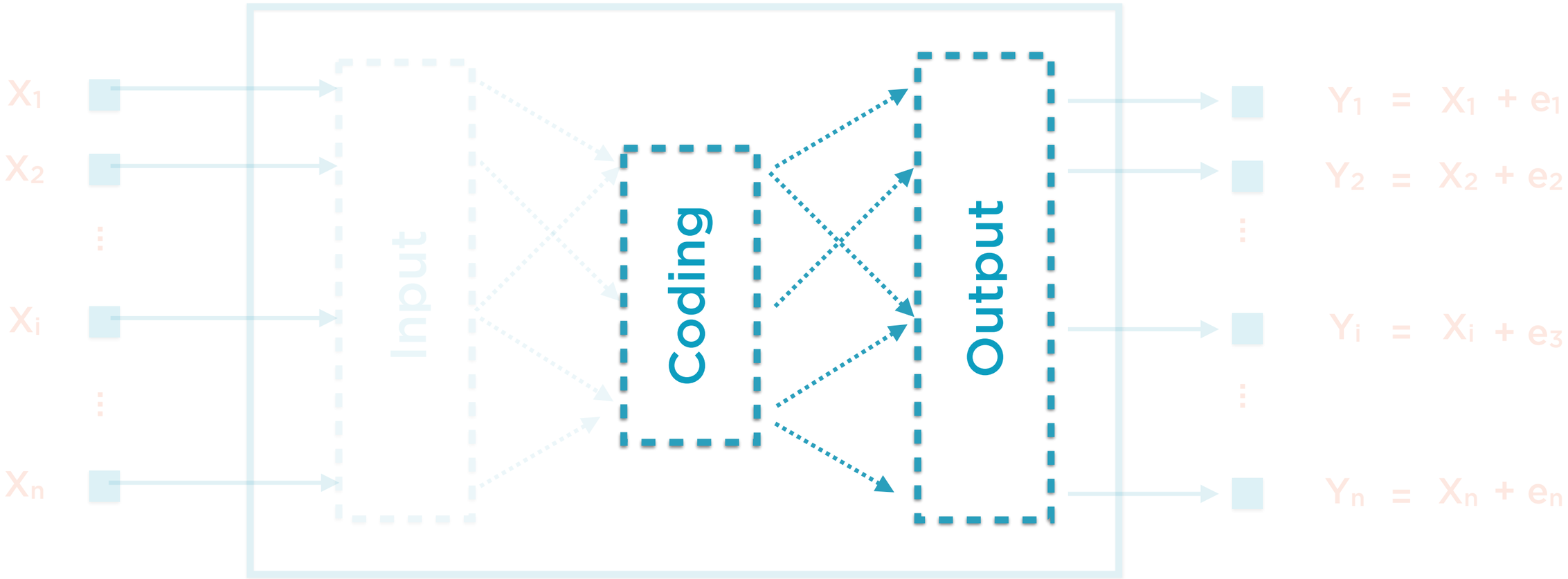
The central hidden layer is called the coding layer

Undercomplete Autoencoder



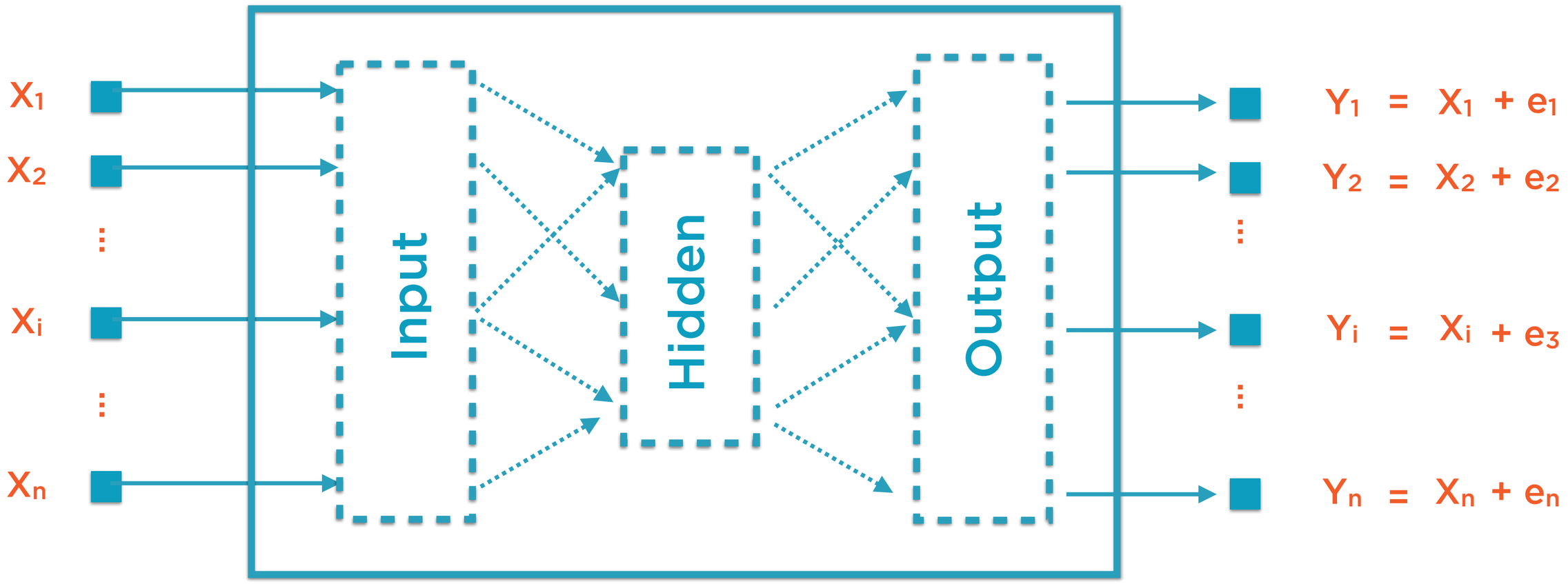
The first phase - from input to coding - is called encoding

Undercomplete Autoencoder



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

Autoencoder



Demo

Implement autoencoding for dimensionality reduction

Overview

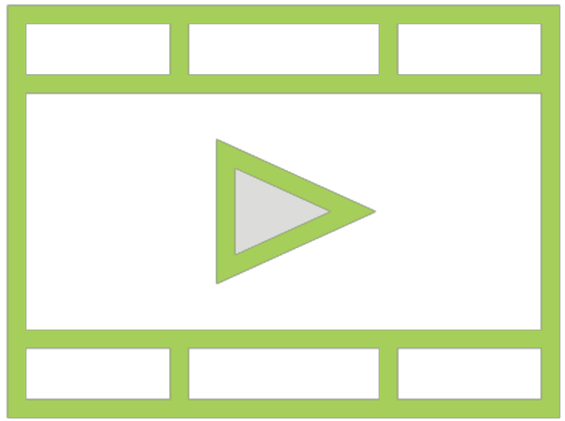
Clustering and autoencoding are classic unsupervised learning techniques

Applying clustering to dimensionality reduction

Feed the reduced output of a clustering model to a classification model

Autoencoding as a dimensionality reduction problem

Related Courses



Building Features from Numeric Data

Building Features from Image Data

Building Features from Text Data