MACHINE LEARNING IN PHYSICS
AUTOENCODERS

HARRISON B. PROSPER

PHY6938

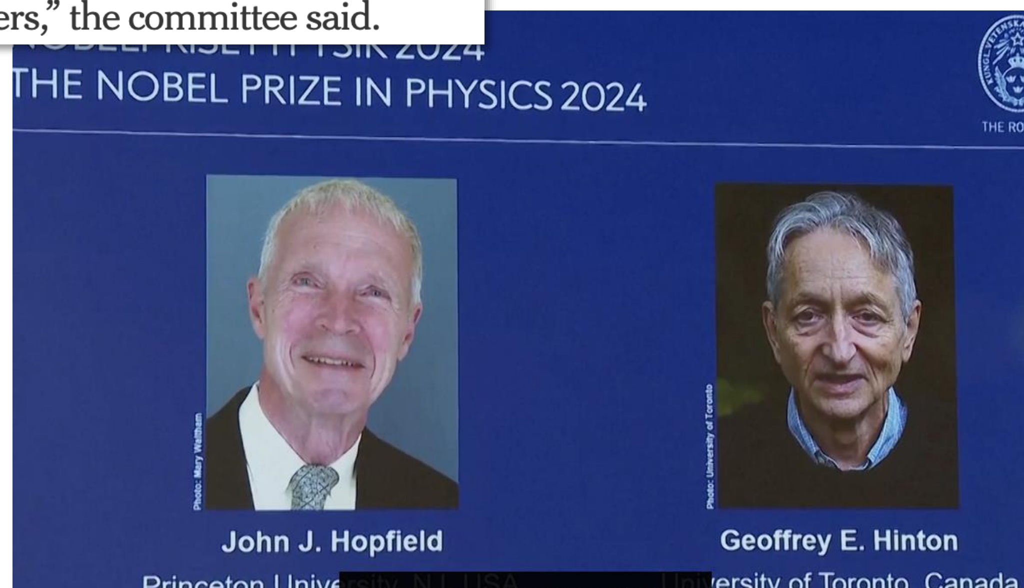
The New York Times

2024 Nobel Prizes › | What to Know | Prize in Medicine | Prize in Physics | Other Science Prizes

Oct. 8, 2024

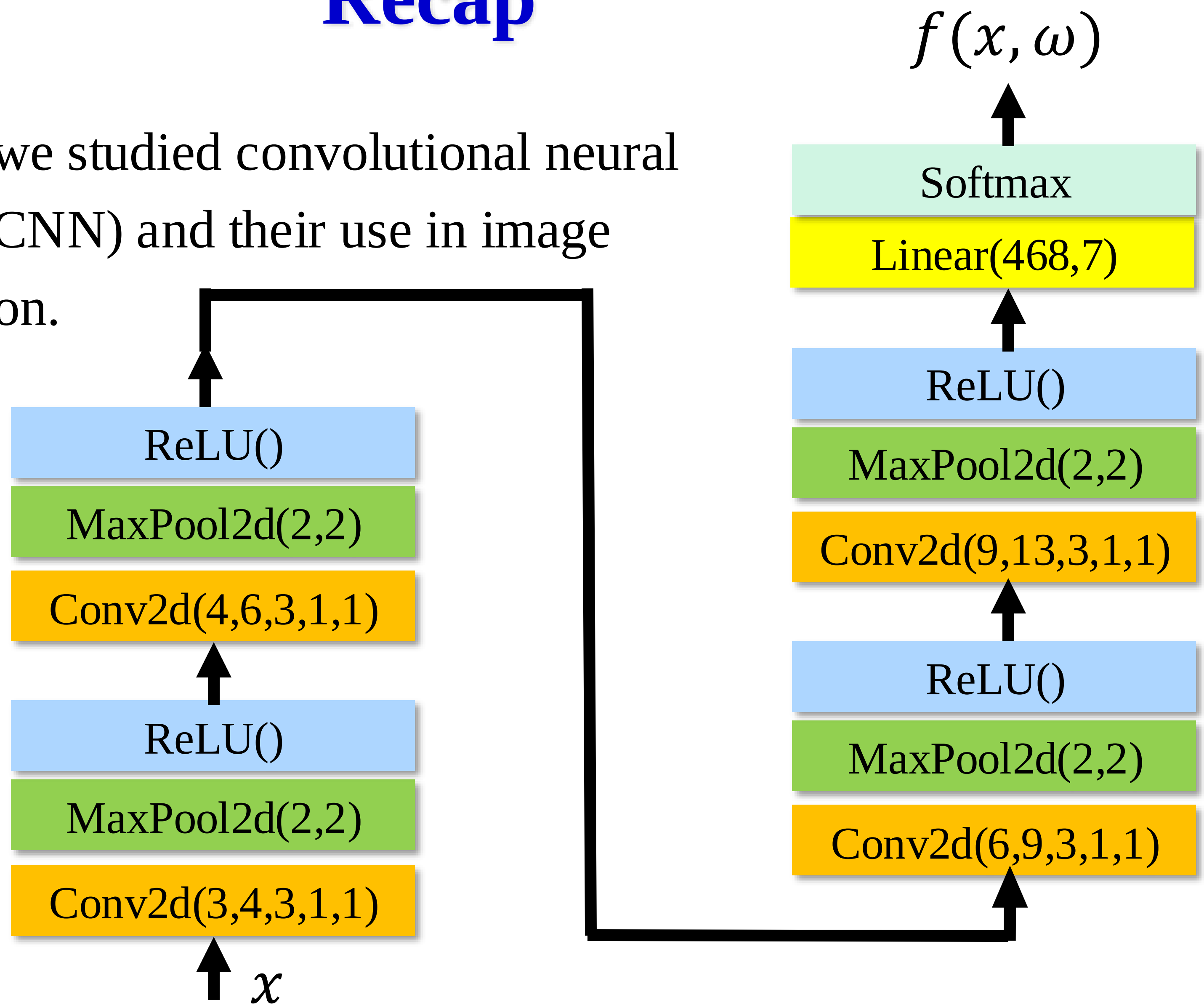
Nobel Physics Prize Awarded for Pioneering A.I. Research by 2 Scientists

With work on machine learning that uses artificial neural networks, John J. Hopfield and Geoffrey E. Hinton “showed a completely new way for us to use computers,” the committee said.



Recap

Last time, we studied convolutional neural networks (CNN) and their use in image classification.



Introduction

This week we introduce a neural network called an:

1. Convolutional neural networks (CNN)
2. Autoencoder (AE)
3. Physics-informed neural networks (PINN)
4. Flow and diffusion models
5. Graph neural networks (GNN)
6. Transformer neural networks (TNN)

Introduction

- An autoencoder approximates the identity map

$$f : x \to x, \quad x \in \mathbb{R}^N$$

- But there is an important twist: the map f is a composition $f(x) = (g \circ h)(x)$ where

$$h : x \to z, \; g : z \to x, \quad z \in \mathbb{R}^n$$

or

$$z = h(x), x = g(z).$$

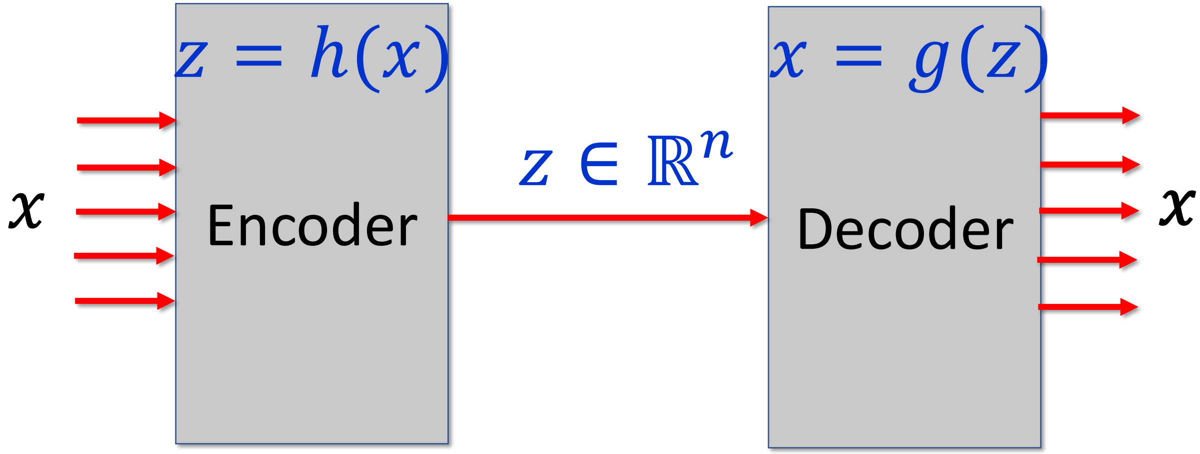
- The function $z = h(x)$ is called the encoder and $x = g(z)$ is the decoder.

Introduction

- The space $\mathbb{R}^n$ is called the latent space.

- Typically, $n \ll N$.

- The encoder *compresses* the input data x



in an *unsupervised* manner, that is, the *labels* are the data x.

Introduction

Autoencoders have multiple applications including:

1. Data compression
   1. Lossy compression
   2. Structure detection

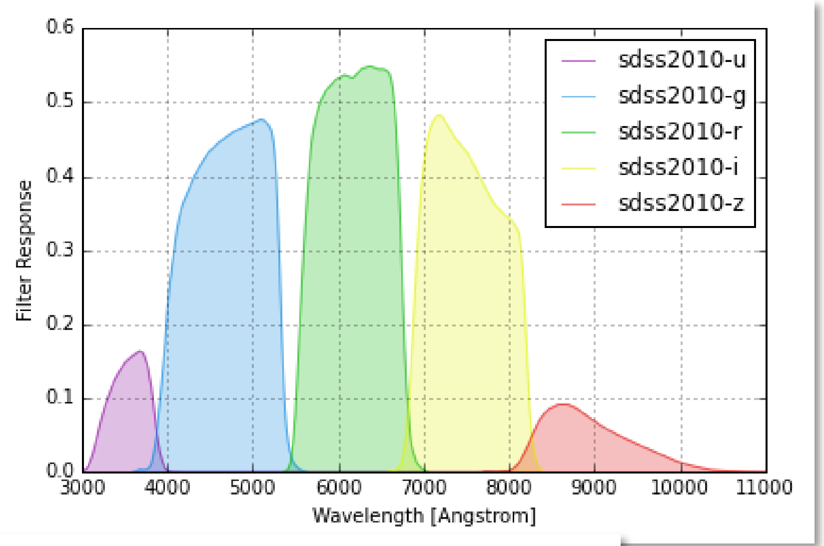
2. Anomaly detection

3. Noise removal

Our example this week focuses on structure detection.

Introduction

- We'll use an autoencoder to search for structure in stellar color data from the Sloan Digital Sky Survey (SDSS)*.



- The data are fluxes measured in five filters:

Filter	Wavelength (nm)
Ultraviolet (u)	354.3
Green (g)	477.0
Red (r)	623.1
Near Infrared (i)	762.5
Infrared (z)	913.4

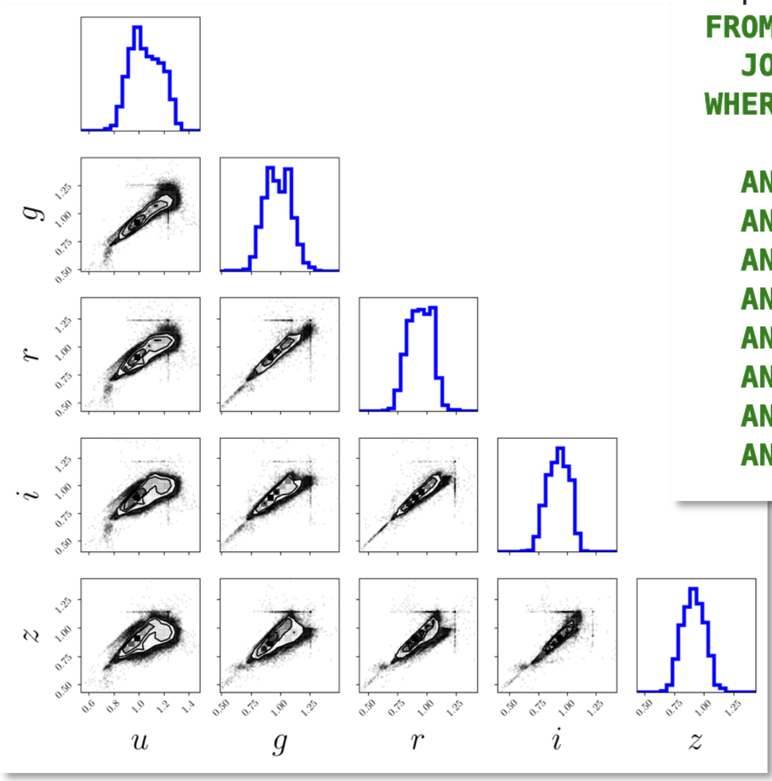
*https://www.sdss.org/

Introduction

SDSS Dataset (https://cas.sdss.org/dr18/SearchTools/sql)

We use data from 255,000 stars, as displayed below, extracted using the SQL command:

```sql
SELECT TOP 255000
  p.ra, p.dec, s.z as redshift, p.u, p.g, p.r, p.i, p.z
FROM PhotoObj AS p
  JOIN SpecObj AS s ON s.bestobjid = p.objid
WHERE
      p.u BETWEEN 0 AND 30
  AND p.g BETWEEN 0 AND 30
  AND p.r BETWEEN 0 AND 30
  AND p.i BETWEEN 0 AND 30
  AND p.z BETWEEN 0 AND 30
  AND s.class = 'STAR'
  AND s.class <> 'UNKNOWN'
  AND s.class <> 'SKY'
  AND s.class <> 'STAR_LATE'
```



The plot shows no obvious structure, but maybe there is!

AUTOENCODER

Autoencoder

The key feature of an autoencoder is the bottleneck between the encoder and the decoder. This forces the model to construct a lower-dimensional representation in the latent space, which we take to be $z \in \mathbb{R}^2$, of the input data $x \in \mathbb{R}^5$.

Since the goal is to model the mapping $f: x \to x$, the *quadratic* loss, $L(y, f) = (y - f)^2$, can be used. Recall that this implies that our model will approximate

$$f(x, \omega^*) = \int y \, p(y \mid x) \, dy$$

Here is a rare case where the form of $p(y \mid x)$ is known!

Autoencoder

The targets (labels) y in

$$f(x, \omega^*) = \int y \, p(y \mid x) \, dy$$

are the input data themselves, i.e., $y = x$.

It, therefore, follows that

$$p(y \mid x) = \delta(y - x),$$

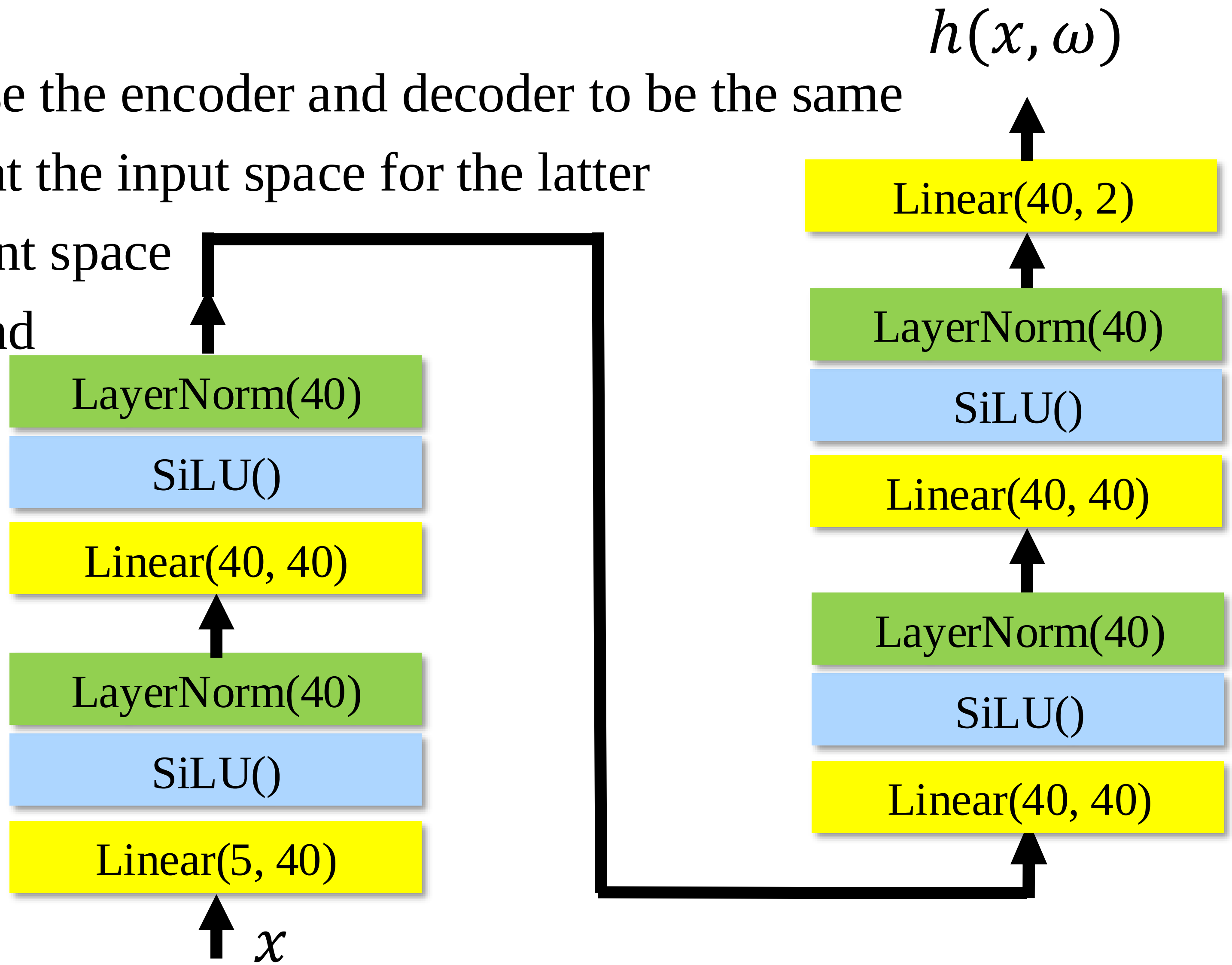
in which case,

$$f(x, \omega^*) = x.$$

In principle, we have an *unbiased* estimate of the input data.

AE Model: Encoder

We choose the encoder and decoder to be the same except that the input space for the latter is the latent space $z \in \mathbb{R}^2$ and $x \in \mathbb{R}^5$ is the output space.



AE Model: Program View

```
H_NODES = 40
encoder = nn.Sequential(
    nn.Linear(      5, H_NODES),         nn.SiLU(), nn.LayerNorm(H_NODES),
    nn.Linear(H_NODES, H_NODES), nn.SiLU(), nn.LayerNorm(H_NODES),
    nn.Linear(H_NODES, H_NODES), nn.SiLU(), nn.LayerNorm(H_NODES),
    nn.Linear(H_NODES, H_NODES), nn.SiLU(), nn.LayerNorm(H_NODES),
    nn.Linear(H_NODES, 2),
      )
decoder = nn.Sequential(
    nn.Linear(      2, H_NODES),         nn.SiLU(), nn.LayerNorm(H_NODES),
    nn.Linear(H_NODES, H_NODES), nn.SiLU(), nn.LayerNorm(H_NODES),
    nn.Linear(H_NODES, H_NODES), nn.SiLU(), nn.LayerNorm(H_NODES),
    nn.Linear(H_NODES, H_NODES), nn.SiLU(), nn.LayerNorm(H_NODES),
    nn.Linear(H_NODES,  5)
      )
```

$$y = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$

AE Model: Program View

```
class AutoEncoder(nn.Module):

    def __init__(self, encoder, decoder):
        # call constructor of base (or super, or parent) class
        super(AutoEncoder, self).__init__()

        self.encoder = encoder
        self.decoder = decoder

    def forward(self, x):
        y = self.encoder(x)
        y = self.decoder(y)
        return y
```

Summary

- An autoencoder is a model that implements the map $f: x \to x$ via a bottleneck called the latent space whose dimensionality is (usually) much smaller than that of the input space.

- The applications include:
  1. Data compression
  2. Anomaly detection
  3. Noise removal