ECS 171: Machine Learning

Summer 2023

Edwin Solares
easolares@ucdavis.edu

Preprocessing & Convolutions

Object Classification using CNN’s

Data Preprocessing

Goal: Using transforms, scale data to similar values

Scaling Data

1. Normalizing - Scaling from 0 to 1
2. Standardization - Scaling data so mean = 0 and standard deviation = 1
    a. Assumes data is already normalized
3. Testing for normality - Shapiro-Wilk Test
    a. Others include Q-Qplot (quantile plots), Histogram plot, Kolmogorov-Smirnov Test

Transform data - Non Constants Transformations

1. Log Transformation
2. Square Root Transformation
3. Cube Root Transformation

Data Preprocessing

Goal: Using transforms, scale data to similar values

Encoding

1. Replace categories with integer values
2. Create new features based on k # of categories containing binar values

Imputing Data

1. Dropping null data
2. Replacing null values with mean, median, most frequent values, etc.
3. More options discussed after Midterm

Normalizing Data

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Used when data is not normally distributed

Benefits:

- Faster processing for GD methods
- Allows you to view actual importance to predicted values using weights

Implementation:

- MinMax Normalization

Standardizing Data

$$z = \frac{x - \mu}{\sigma}$$

$\mu =$ Mean

$\sigma =$ Standard Deviation

Used when data is normalized

Benefits:

- Much faster processing for GD methods
- Allows you to view actual importance to predicted values using weights

Implementation:

- Z-score standardization

Encoding Data: Value Replacement

Used for transforming categorical data into integer classes

- Take k number of categories and assign integer values from 0 to k-1

Size
Small
Medium
Large

Size
0
1
2

Encoding Data: Feature Expansion

Used for transforming categorical data into integer classes

- Take k number of categories and assign k new feature vectors
- New feature vectors contain binary values: 0 or 1

Size
Small
Medium
Large

Small	Medium	Large
1	0	0
0	1	0
0	0	1

One-Hot Encoding

Imputing Data: Dropping Null Data

- Used when sufficient data is available, and most data is complete
- If null data is randomized, then less bias than other methods

Area
16
NA
36

Area
16
36

Imputing Data: Filling Null Values

- Used when sufficient data is NOT available, and most data is INCOMPLETE
- Depending on how null data is distributed, different methods may be used to reduce bias. I.e. Mean, Median, Random, Most Frequent

Area
16
NA
36

Area
16
26
36

Example of using Mean value

Jupyter Notebooks Time!

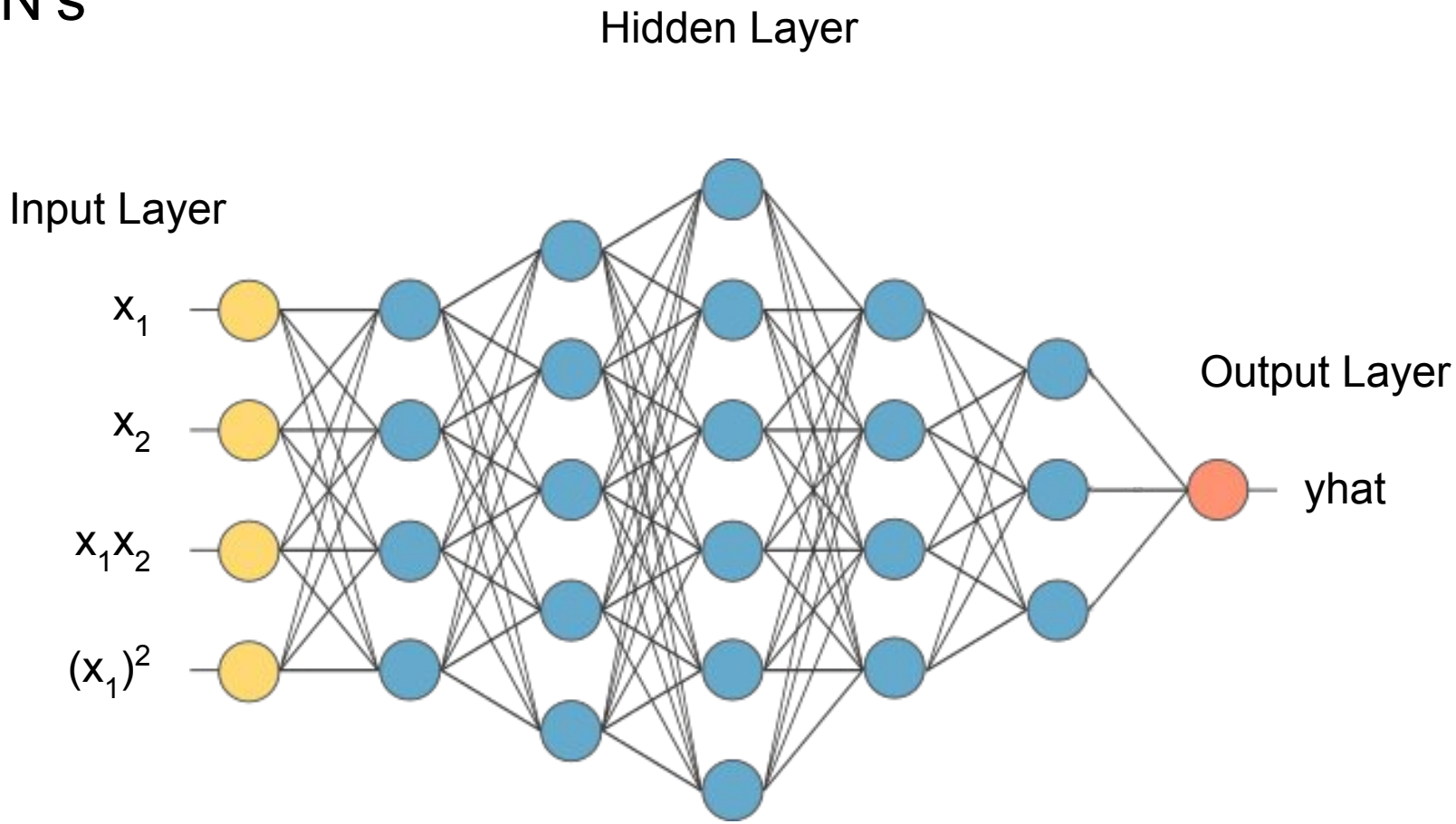
Convolution Neural Networks

CNN’s

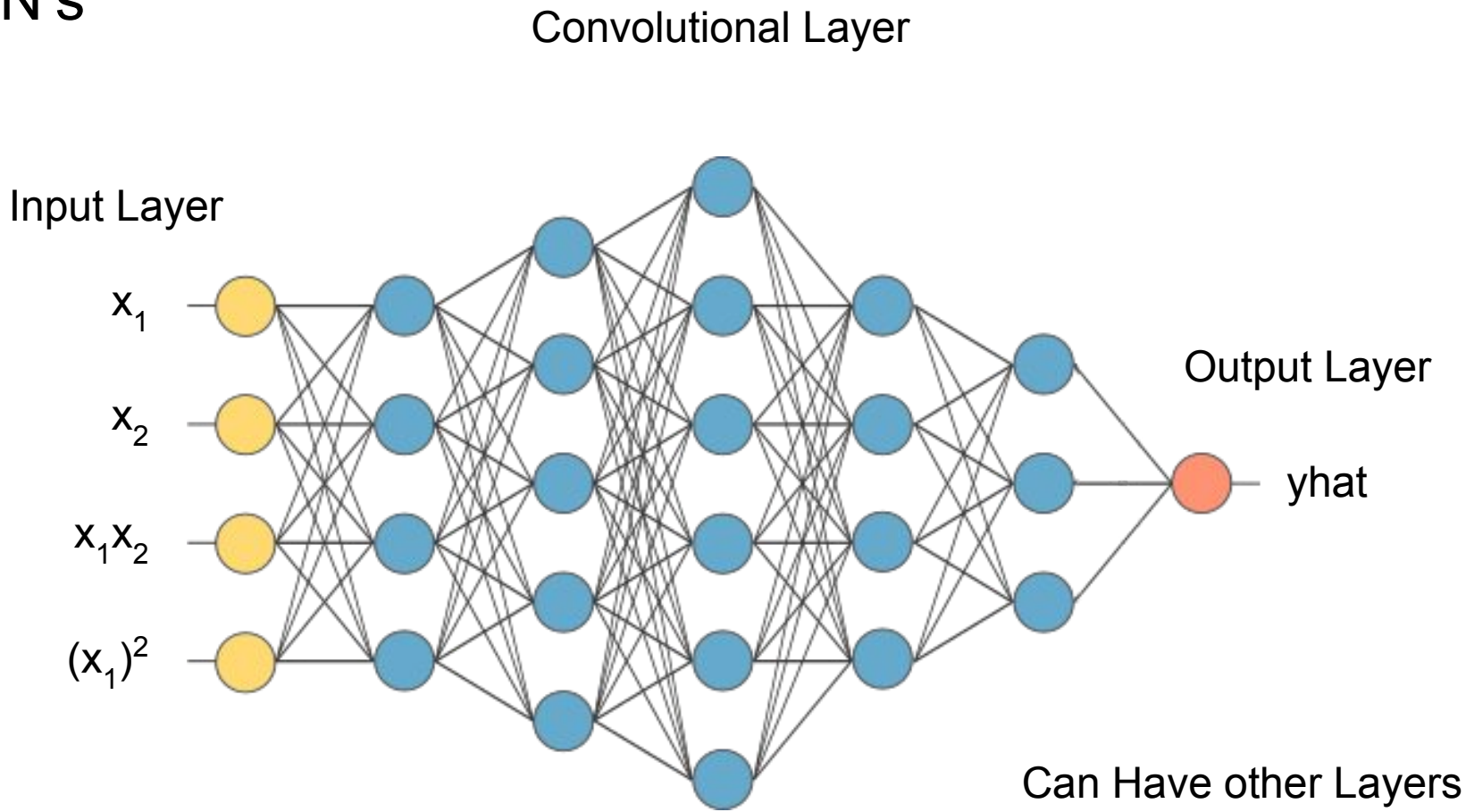
A special build of a Neural Network that is optimized for pattern recognition

- Can detect geometric shapes
- With deeper layers → more complex geometric pattern detection

DNN's

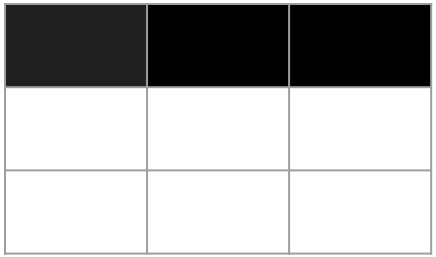


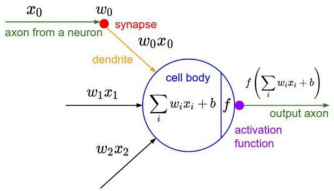
CNN's


Convolutional Layer
Input Layer
x_1
x_2
x_1x_2
$(x_1)^2$
Output Layer
yhat
Can Have other Layers

Filters

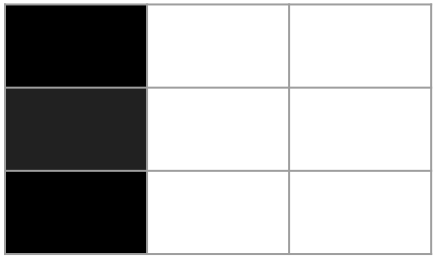
Top Edge Filter

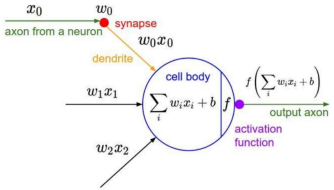




1

Left Edge Filter



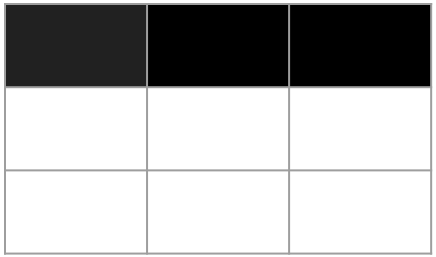


1

AND → Corner

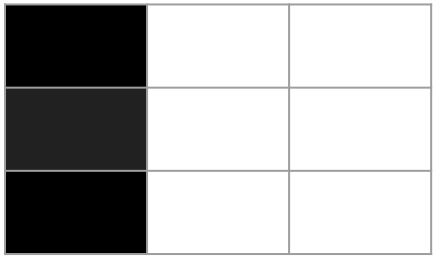
Filters: Matrix Convolution

Top Edge Filter



1	1	1
0	0	0
-1	-1	-1

Left Edge Filter



1	0	-1
1	0	-1
1	0	-1

Filters: Matrix Convolution

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

6 x 6

*

1	0	-1
1	0	-1
1	0	-1

3 x 3

Filters: Matrix Convolution

0 value when pixel is uniform
Large value when pixel is not uniform
Max value when pixel fits pattern

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

6 x 6

*

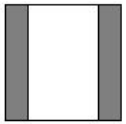
1	0	-1
1	0	-1
1	0	-1

3 x 3

=

-0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

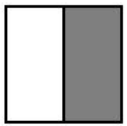
4 x 4

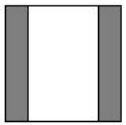


Filters: Matrix Convolution

0 value when pixel is uniform
Large value when pixel is not uniform
Max value when pixel fits pattern

$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 1*10 & 0*10 & -1*10 \\ 1*10 & 0*10 & -1*10 \\ 1*10 & 0*10 & -1*10 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 30 & 30 & 0 \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$

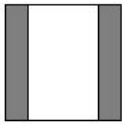




Filters: Matrix Convolution

0 value when pixel is uniform
Large value when pixel is not uniform
Max value when pixel fits pattern

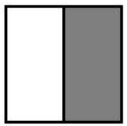
$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 1*10 & 0*10 & -1*0 \\ 1*10 & 0*10 & -1*0 \\ 1*10 & 0*10 & -1*0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 30 & 30 & 0 \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$

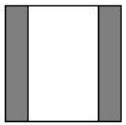


Filters: Matrix Convolution

0 value when pixel is uniform
Large value when pixel is not uniform
Max value when pixel fits pattern

$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 1*10 & 0*0 & -1*0 \\ 1*10 & 0*0 & -1*0 \\ 1*10 & 0*0 & -1*0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 30 & \boxed{30} & 0 \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$

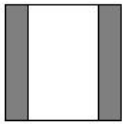




Filters: Matrix Convolution

0 value when pixel is uniform
Large value when pixel is not uniform
Max value when pixel fits pattern

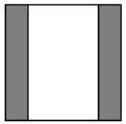
$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 1*0 & 0*0 & -1*0 \\ 1*0 & 0*0 & -1*0 \\ 1*0 & 0*0 & -1*0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 30 & 30 & 0 \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$



Filters: Matrix Convolution

0 value when pixel is uniform
Large value when pixel is not uniform
Max value when pixel fits pattern

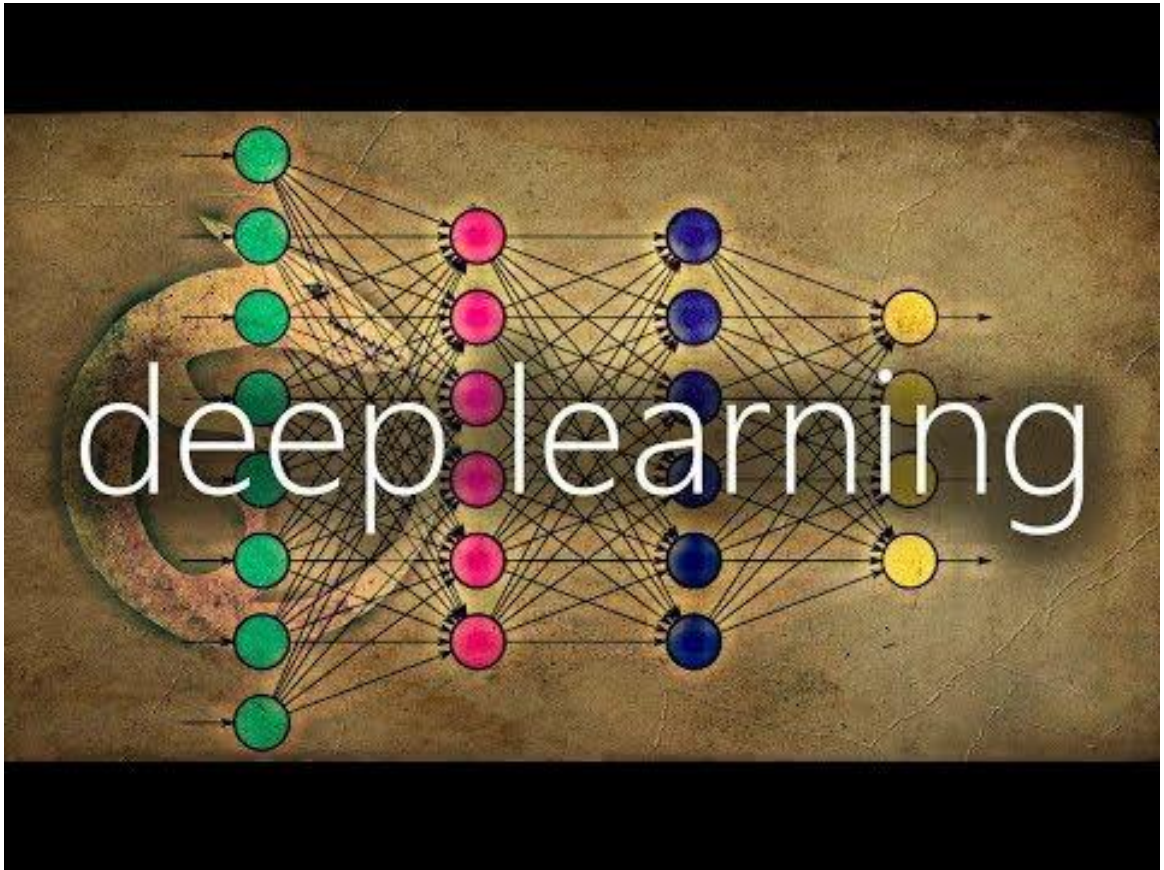
$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{bmatrix}$$



Filters Time!

https://deeplizard.com/resource/pavq7noze2

Filters



Jupyter Notebooks Time!

https://colab.research.google.com