Machine Learning, Artificial Intelligence, and Natural Language Processing

Marcelo C. Medeiros

Department of Economics
The University of Illinois at Urbana-Champaign

Neural Networks

- Introduction
- Mathematical definition

Deep Neural Networks

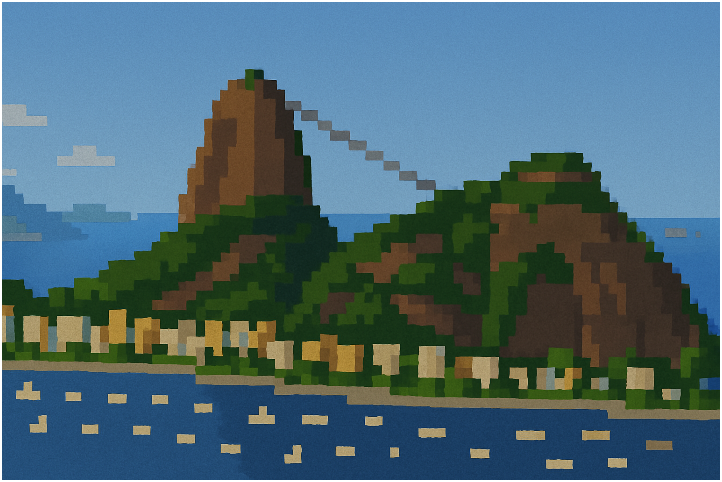
Convolutional Neural Networks

Long Short Term Neural Networks

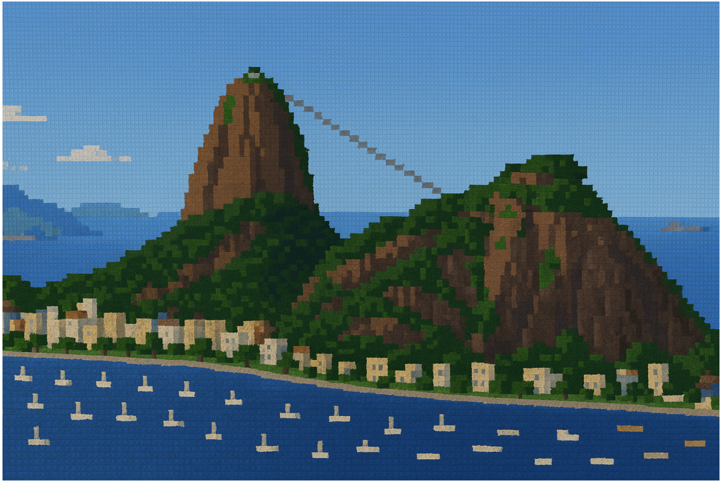
Let's play Minecraft

Let's reproduce this image with blocks

We start with big blocks



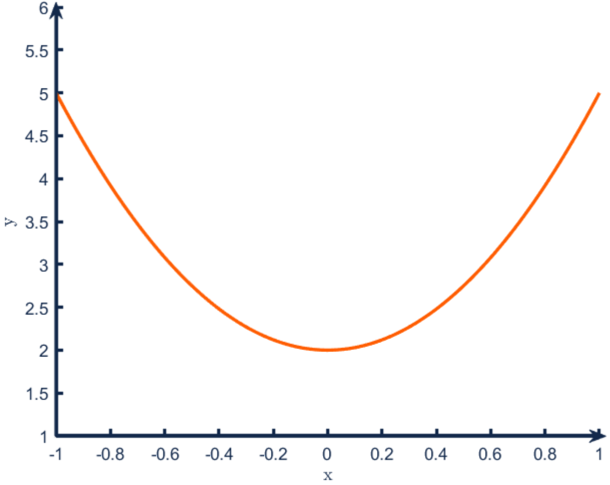
A bit smaller blocks



Even smaller blocks

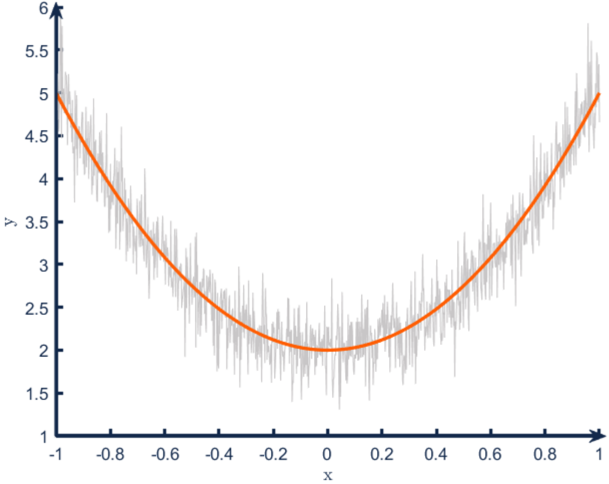
Now let's polish the edges

Our final result



- Consider the function

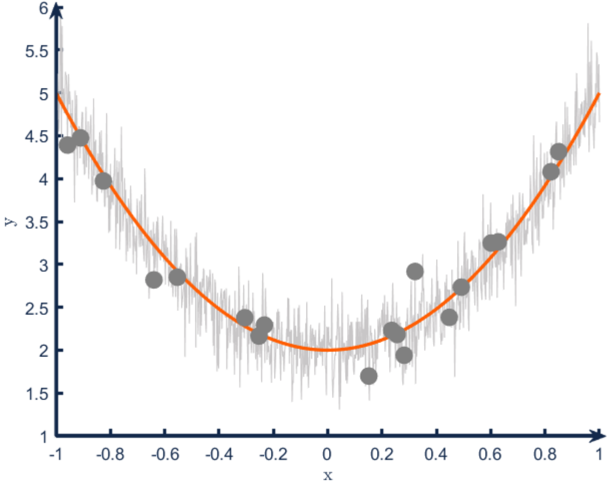
$$
\begin{aligned}
y &= f(x) \\
&= 2 + 3x^2
\end{aligned}
$$



- Consider the function

$$y = f(x) \\ = 2 + 3x^2$$

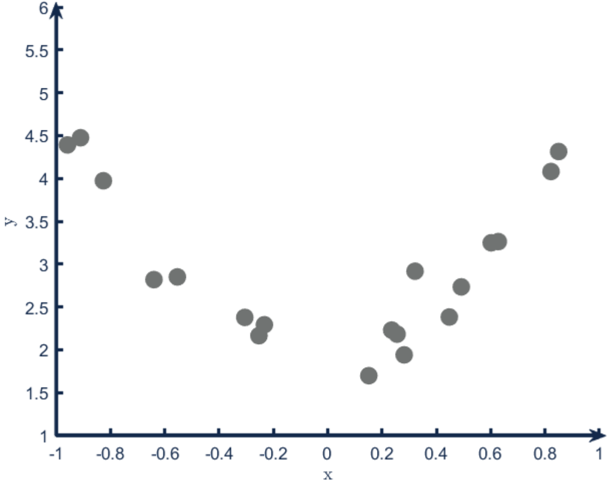
- We observe y with error...



- Consider the function

$$y = f(x) \\ = 2 + 3x^2$$

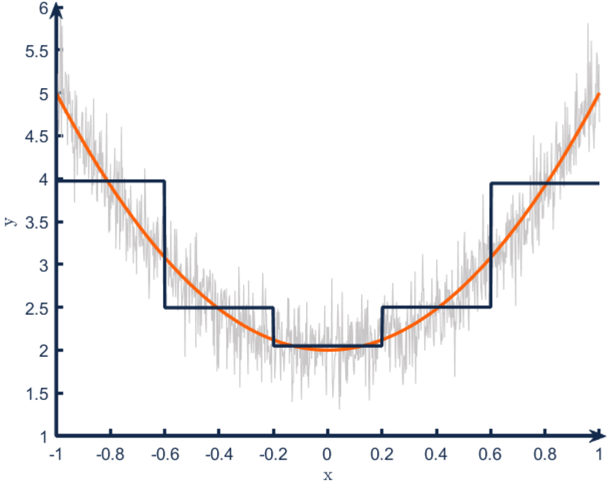
- We observe y with error...
  ... and only a sample



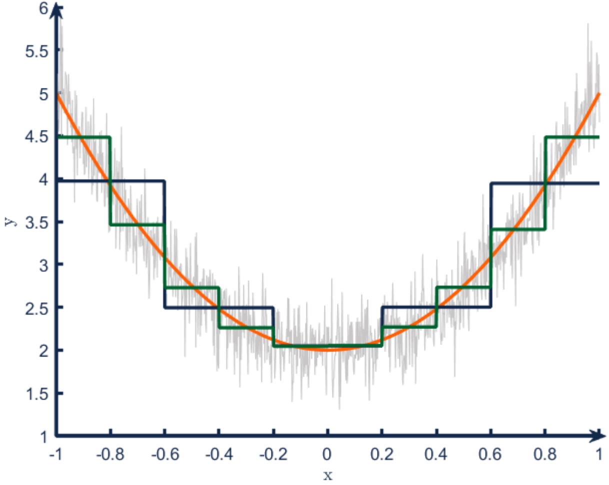
- Consider the function

$$y = f(x) \\ = 2 + 3x^2$$

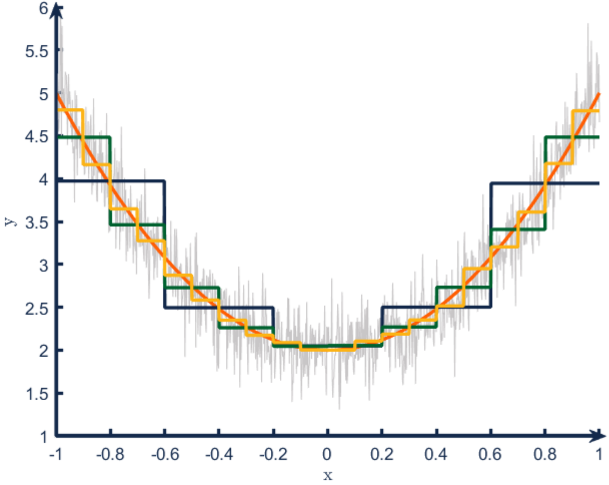
- We observe y with error...
  ... and only a sample
- How can we recover the function from data?



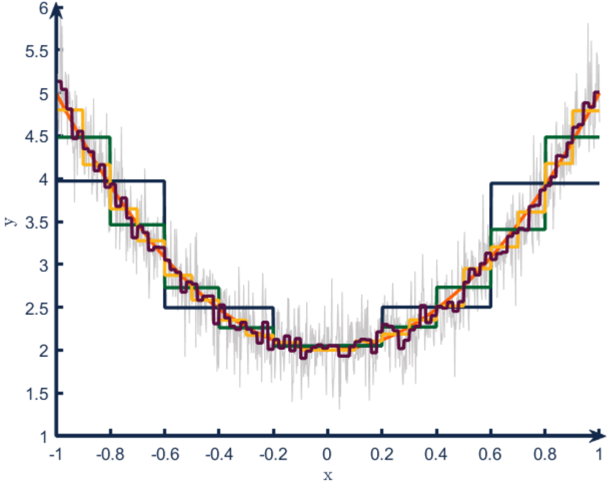
- Divide the domain of x into $K = 5$ regions
- Consider a locally constant approximation



- Divide the domain of x into $K = 5$ regions
- Consider a locally constant approximation
- Approximation gets better as the number of regions grows



- Divide the domain of x into $K = 5$ regions
- Consider a locally constant approximation
- Approximation gets better as the number of regions grows
- $K = 10, 20, 100, \ldots$



- Divide the domain of x into $K = 5$ regions
- Consider a locally constant approximation
- Approximation gets better as the number of regions grows
- $K = 10, 20, 100, \ldots$

- In the previous slides, the function $f(x) = 2 + x^3$ was being approximated by

$$h(x) = \beta_0 + \sum_{k=1}^{K-1} \beta_k \mathbf{I}(x \geq c_k),$$

where

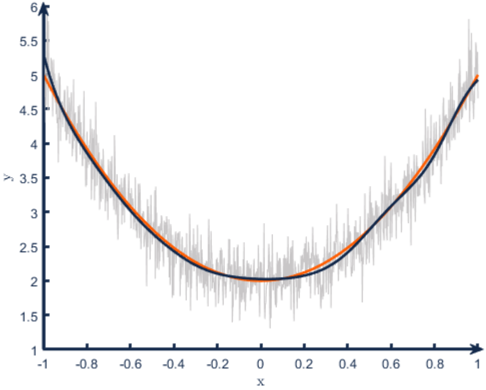
$$\mathbf{I}(x \geq c_k) = \begin{cases} 1 & \text{if } x \geq c_k \\ 0 & \text{otherwise} \end{cases}$$

- $c_1, \ldots, c_K$ are split points and $\beta_0, \ldots, \beta_{K-1}$ represent the local approximation

- Can we do better than this?

- Can we do better than this?
- Yes, we can smooth the edges

- Can we do better than this?
- Yes, we can smooth the edges
- Five regions with smooth edges



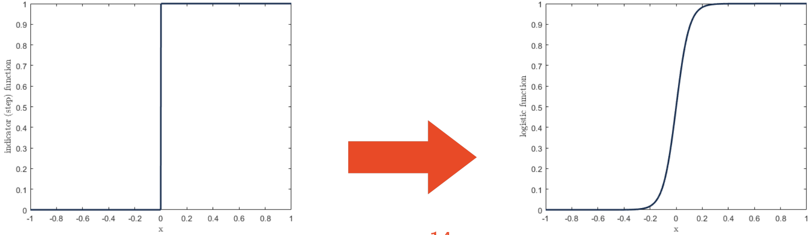
Introduction

Neural Networks and function approximation

- The edges can be smoothed by replacing

$$\mathsf{I}(x \geq c_k) = \begin{cases} 1 & \text{if } x \geq c_k \\ 0 & \text{otherwise} \end{cases}$$

  by a smooth function
- One possible choice is the logistic function



This is the idea behind Neural Networks

Neural Networks
- Introduction
- Mathematical definition

Deep Neural Networks

Convolutional Neural Networks

Long Short Term Neural Networks

The Problem

- Observe the target variable Y and the inputs $\boldsymbol{X} = (X_1, \ldots, X_p)'$

The Problem

- Observe the target variable Y and the inputs $\boldsymbol{X} = (X_1, \ldots, X_p)'$
- Unknown mapping (relation) between Y and $\boldsymbol{X}$:

$$Y = f(\boldsymbol{X}) + U,$$

where U is a random error $\rightsquigarrow$ the relation between Y and X is not perfect

The Problem

- Observe the target variable Y and the inputs $\boldsymbol{X} = (X_1, \ldots, X_p)'$
- Unknown mapping (relation) between Y and $\boldsymbol{X}$:
$$Y = f(\boldsymbol{X}) + U,$$
where U is a random error $\rightsquigarrow$ the relation between Y and X is not perfect
- From a random sample $\{Y_i, \boldsymbol{X}_i\}_{i=1}^n$, we would like to learn (estimate) f to predict Y^* from a new X^*:
$$Y^* := f(\boldsymbol{X}^*)$$

The Neural Network Approach

- NN idea: Approximates the unknown $f(\cdot)$ by

$$H(\boldsymbol{X};\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{J} \beta_j S(\boldsymbol{\gamma}_j' \boldsymbol{X} + \gamma_{0,j}),$$

  where:

  * $\boldsymbol{X} \mapsto \boldsymbol{\gamma}_j' \boldsymbol{X} + \gamma_{0,j}$ is an affine transformation (linear combination plus a shift) of the input vector $\boldsymbol{X}$

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

The Neural Network Approach

- NN idea: Approximates the unknown $f(\cdot)$ by

$$H(\boldsymbol{X};\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{J} \beta_j S(\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j}),$$

where:

  * $\boldsymbol{X} \mapsto \boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j}$ is an affine transformation (linear combination plus a shift) of the input vector $\boldsymbol{X}$
  * $S : \mathbb{R} \to \mathbb{R}$ is a basis function

The Neural Network Approach

- NN idea: Approximates the unknown $f(\cdot)$ by

$$H(\boldsymbol{X};\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{J} \beta_j S(\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j}),$$

where:

  * $\boldsymbol{X} \mapsto \boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j}$ is an affine transformation (linear combination plus a shift) of the input vector $\boldsymbol{X}$
  * $S : \mathbb{R} \to \mathbb{R}$ is a basis function
  * $\boldsymbol{\theta} := (\beta_0, \ldots, \beta_{J_T}, \boldsymbol{\gamma}_1', \ldots, \boldsymbol{\gamma}_J', \gamma_{0,1}, \ldots, \gamma_{0,J})'$ is the vector of parameters that must be estimated

Some Neural Network Nomenclature

$$H(\boldsymbol{X};\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{J_T} \beta_j S(\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j})$$

- The basis functions S are called activation functions

Some Neural Network Nomenclature

$$H(\boldsymbol{X};\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{J_T} \beta_j S(\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j})$$

- The basis functions S are called activation functions
- The parameters θ are called weights

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

Some Neural Network Nomenclature

$$H(\boldsymbol{X};\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{J_T} \beta_j S(\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j})$$

- The basis functions S are called activation functions
- The parameters θ are called weights
- In particular, β_0 and $\gamma_{0,j}$ are called bias $\rightsquigarrow$ Unrelated to the statistical concept of bias

Some Neural Network Nomenclature

$$H(\boldsymbol{X};\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{J_T} \beta_j S(\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j})$$

- The basis functions S are called activation functions
- The parameters θ are called weights
- In particular, β_0 and $\gamma_{0,j}$ are called bias $\rightsquigarrow$ Unrelated to the statistical concept of Bias
- Note that $\gamma_{0,j}$ shifts the whole S curve to the left and right

Some Neural Network Nomenclature

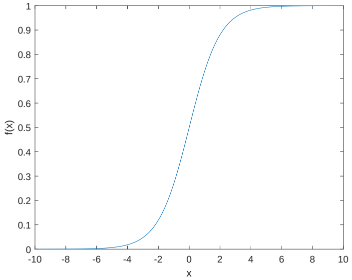
$$H(\boldsymbol{X};\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{J_T} \beta_j S(\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j})$$

- The basis functions S are called activation functions
- The parameters θ are called weights
- In particular, β_0 and $\gamma_{0,j}$ are called bias $\rightsquigarrow$ Unrelated to the statistical concept of Bias
- Note that $\gamma_{0,j}$ shifts the whole S curve to the left and right
- While γ_j controls for the "slope" of S

Popular Choices for S

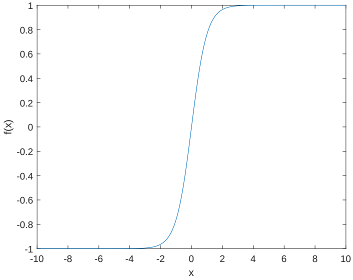
Logistic

$$S(X) = \frac{1}{1 + \exp(-X)}$$



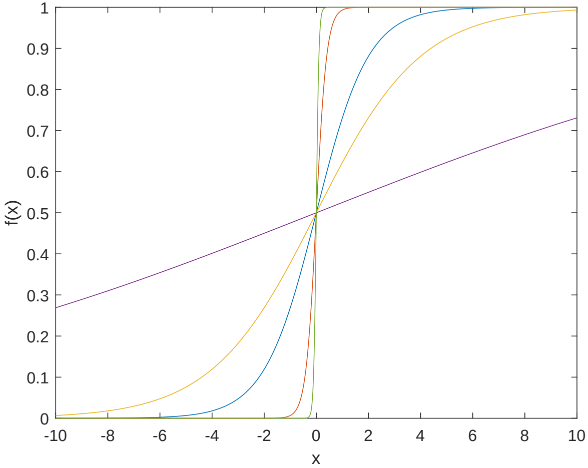
Hyperbolic tangent

$$S(X) = \frac{\exp(X) - \exp(-X)}{\exp(X) + \exp(-X)}$$



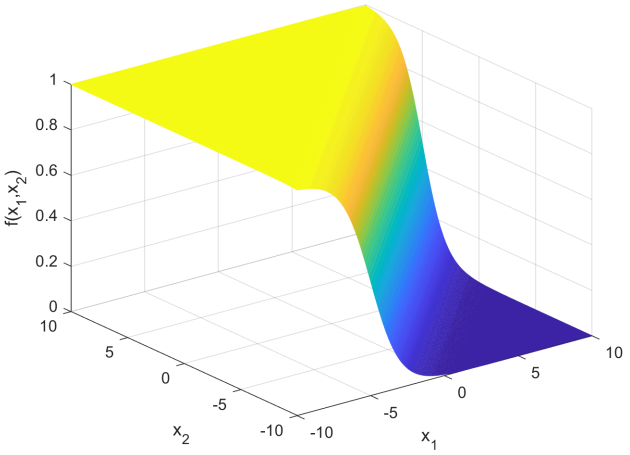
Logistic Activation Function

Varying slope parameter γ



Logistic NN with 2 Inputs

Geometric Interpretation



- In general, but not always, $S(\cdot)$ is a *squashing* function.

Squashing function

A function $S : \mathbb{R} \longrightarrow [a, b]$, $a < b$, is a squashing (sigmoid) function if it is non-decreasing, $\lim_{X \longrightarrow \infty} S(X) = b$ and $\lim_{X \longrightarrow -\infty} S(X) = a$

- In general, but not always, $S(\cdot)$ is a *squashing* function.

Squashing function

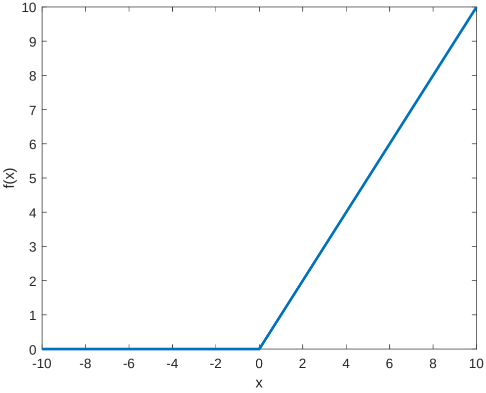
A function $S : \mathbb{R} \longrightarrow [a, b]$, $a < b$, is a squashing (sigmoid) function if it is non-decreasing, $\lim_{X \longrightarrow \infty} S(X) = b$ and $\lim_{X \longrightarrow -\infty} S(X) = a$

- Some popular (old) squashing functions:
  - Heaviside: $S(X) = I(X \geq 0)$
  - Logistic: $S(X) = 1/[1 + \exp(-X)]$
  - Hyperbolic tangent: $S(X) = [\exp(X) - \exp(-X)]/[\exp(X) + \exp(-X)]$
  - Gaussian sigmoid: $S(X) = (2\pi)^{-1/2} \int_{-\infty}^{X} \exp(-u^2/2)du$
  - Cosine squasher: $S(X) = \frac{1+\cos(X+3\pi/2)}{2} I(|X| \leq \pi/2) + I(X > \pi/2)$

Activation Function: Rectified Linear Units

Example of a non-squashing Activation Function

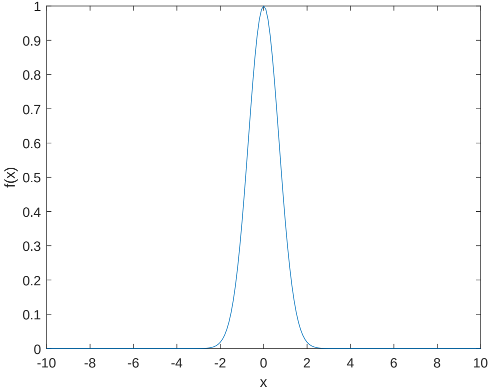
$$S(X) := \mathsf{ReLU}(X) = \max(0, X)$$



Activation Function: Radial Basis

Example of a non-squashing Activation Function

$$S(X) := \text{RBF}(X) = \exp(-X^2)$$



Why NNs?

Universal Approximation Theorem

Feed-forward NN with a single hidden layer with "arbitrary" **squashing** functions can approximate any Borel-measurable function from one finite dimensional space to another to any desired degree of accuracy, provided sufficiently many (finite) hidden units are available

Why NNs?

Universal Approximation Theorem

Feed-forward NN with a single hidden layer with “arbitrary” **squashing** functions can approximate any Borel-measurable function from one finite dimensional space to another to any desired degree of accuracy, provided sufficiently many (finite) hidden units are available

- How big is the class of Borel Measurable functions? It contains all functions with at most countable discontinuities

Why NNs?

Universal Approximation Theorem

Feed-forward NN with a single hidden layer with "arbitrary" **squashing** functions can approximate any Borel-measurable function from one finite dimensional space to another to any desired degree of accuracy, provided sufficiently many (finite) hidden units are available

- How big is the class of Borel Measurable functions? It contains all functions with at most countable discontinuities
- References:
  - Cybenko (1989); Hornik, Stimchombe, and White (NN, 1989); Gallant and White (1988); Gallant and White (NN, 1992); Hornik (NN, 1991); and many others

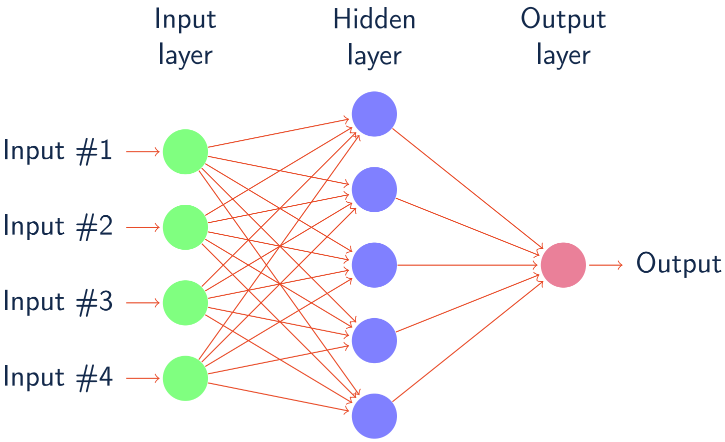
Why NNs?

Universal Approximation Theorem

Feed-forward NN with a single hidden layer with "arbitrary" **squashing** functions can approximate any Borel-measurable function from one finite dimensional space to another to any desired degree of accuracy, provided sufficiently many (finite) hidden units are available

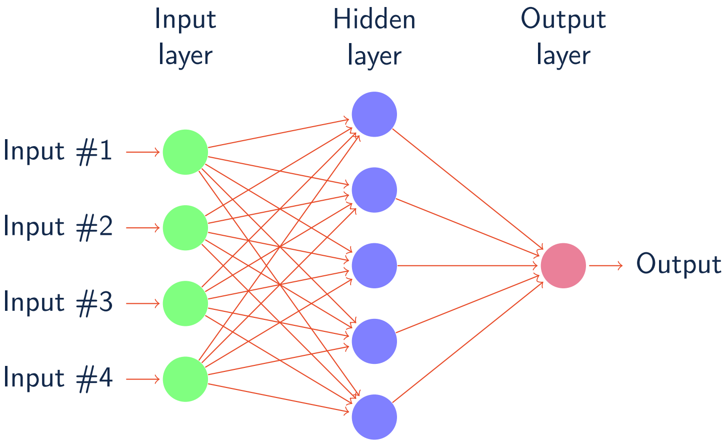
- How big is the class of Borel Measurable functions? It contains all functions with at most countable discontinuities
- References:
  - Cybenko (1989); Hornik, Stimchombe, and White (NN, 1989); Gallant and White (1988); Gallant and White (NN, 1992); Hornik (NN, 1991); and many others
- The same NN can approximate the derivatives of the function

NN as a Graph



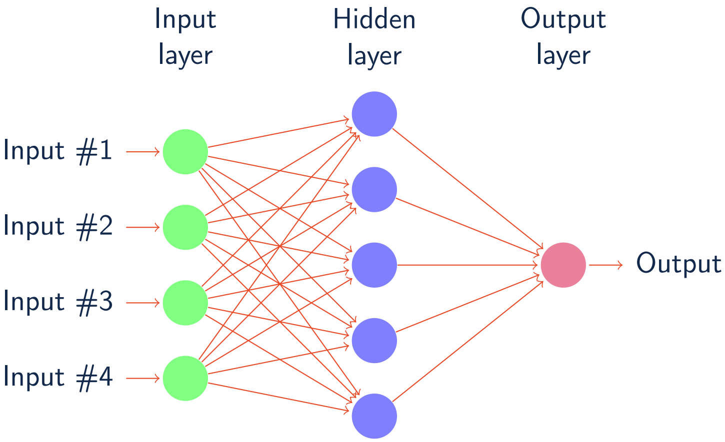
- The input layer is just the vector of explanatory variables.

NN as a Graph



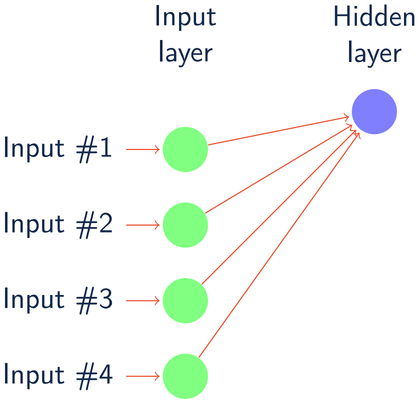
- The input layer is just the vector of explanatory variables.
- The hidden layer consists of a set of hidden units (*neurons*)

NN as a Graph



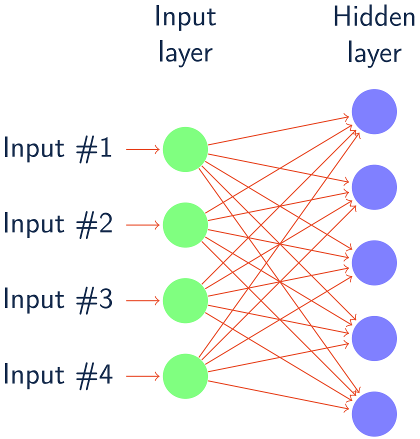
- The input layer is just the vector of explanatory variables.
- The hidden layer consists of a set of hidden units (*neurons*)
- The output layer is the predicted value for the dependent variables

NN as a Graph



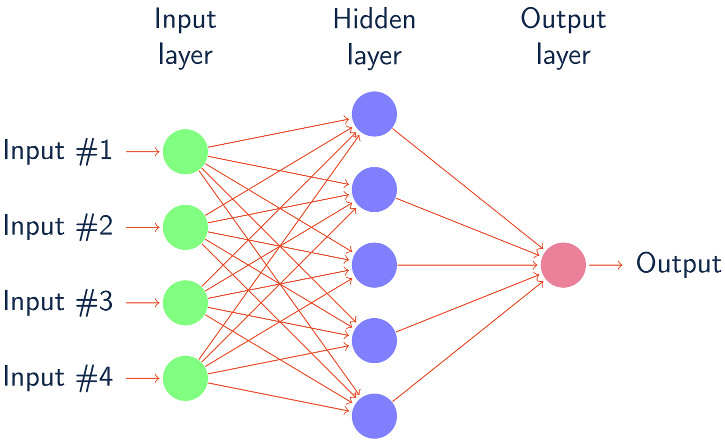
- At each unit inputs are linearly combined: $\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j}$

NN as a Graph



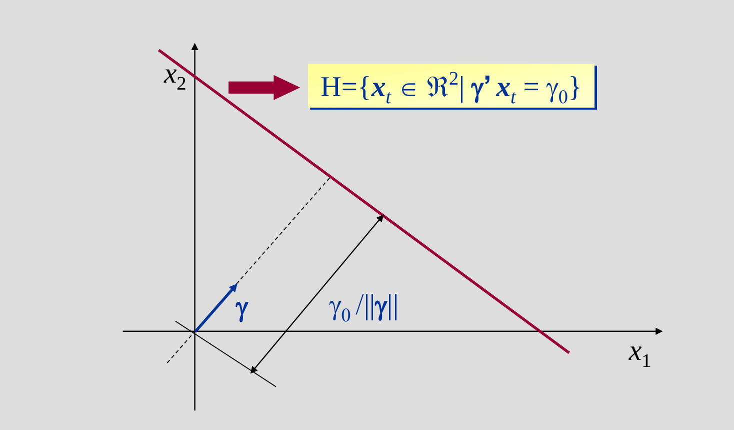
- At each unit inputs are linearly combined: $\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j}$
- Nonlinear transformation in the hidden layer: $S(\boldsymbol{\gamma}_j'\boldsymbol{X} + \gamma_{0,j})$

NN as a Graph

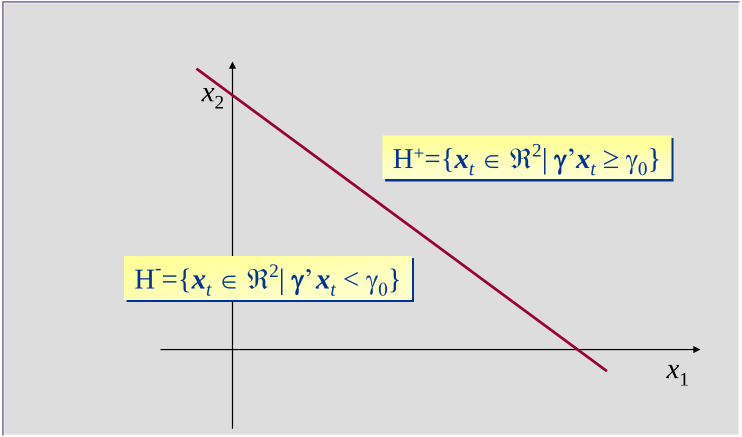


- At each unit inputs are linearly combined: $\boldsymbol{\gamma}_j' \boldsymbol{X} + \gamma_{0,j}$
- Nonlinear transformation in the hidden layer: $S(\boldsymbol{\gamma}_j' \boldsymbol{X} + \gamma_{0,j})$
- Outputs of the hidden layer are linearly combined $\beta_0 + \sum_{j=1}^{5} \beta_j S(\boldsymbol{\gamma}_j' \boldsymbol{X} + \gamma_{0,j})$

Geometric Interpretation

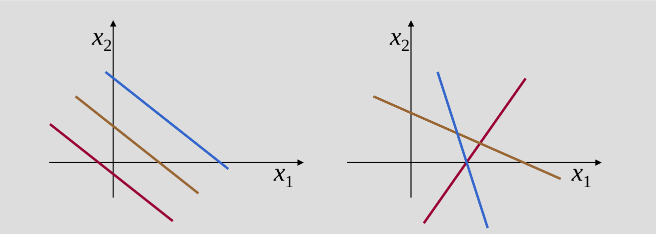


Geometric Interpretation



ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

Geometric Interpretation



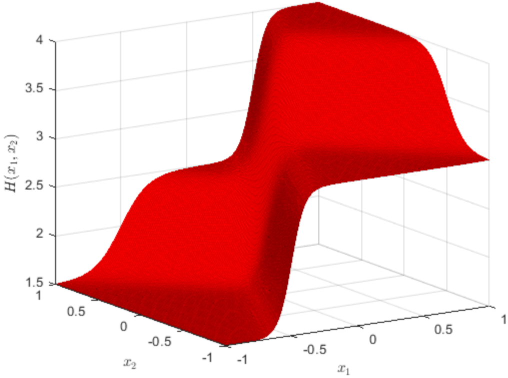
- J hyperplanes divide the space of covariates into several polyhedral regions. The maximum number of regions is given by

$$M(J,p) = M(J-1,p) + M(J-1,p-1),$$

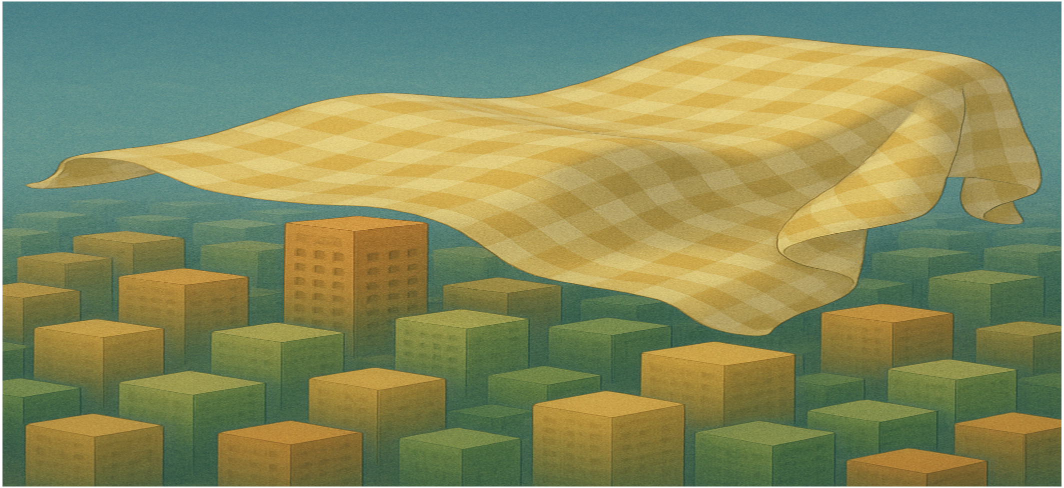
where $M(1,p) = 2$ and $M(J,1) = J+1$.

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

- In each region, the local model is a constant
- Smooth transition between regions
- The number of regions and the degree of smoothness determine the quality of the approximation



Geometric Interpretation



Agenda for Today

Neural Networks
- Introduction
- Mathematical definition

Deep Neural Networks

Convolutional Neural Networks

Long Short Term Neural Networks

From Shallow to Deep: What is a Deep Neural Network?

- It is “just” a neural network with more than one hidden layer

From Shallow to Deep: What is a Deep Neural Network?

- It is “just” a neural network with more than one hidden layer
- The layer might be fully connected or not

From Shallow to Deep: What is a Deep Neural Network?

- It is “just” a neural network with more than one hidden layer
- The layer might be fully connected or not
- Different number of units in each hidden layer

From Shallow to Deep: What is a Deep Neural Network?

- It is "just" a neural network with more than one hidden layer
- The layer might be fully connected or not
- Different number of units in each hidden layer
- Different activation function: rectified linear units (ReLU)

What are the Potential Advantages Over Shallow NNs?

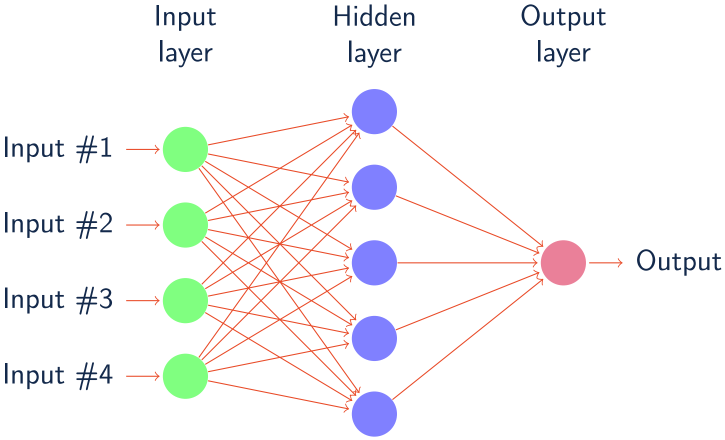
- It has been very successful in many complex applications:
  - Google Neural Machine Translation
  - Lip reading
  - Google Maps and Street View

What are the Potential Advantages Over Shallow NNs?

- It has been very successful in many complex applications:
  - Google Neural Machine Translation
  - Lip reading
  - Google Maps and Street View
- Less hidden units per layer. "While the universal approximation property holds both for hierarchical and shallow networks, deep networks can approximate the class of compositional functions as well as shallow networks but with exponentially lower number of training parameters and sample complexity."

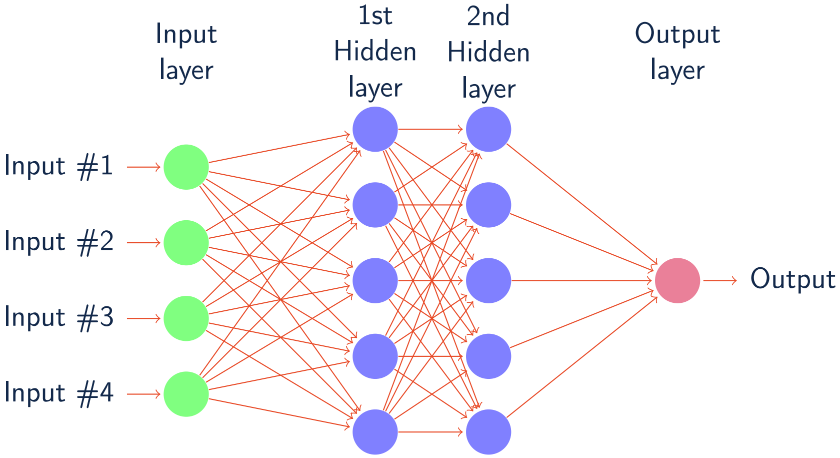
Mhaskar, Liao and Poggio (2017)

Graphical Representation of a (Shallow) NN



Graphical Representation of a Deep Neural Networks

2 fully connected layers



Agenda for Today

Neural Networks

- Introduction
- Mathematical definition

Deep Neural Networks

Convolutional Neural Networks

Long Short Term Neural Networks

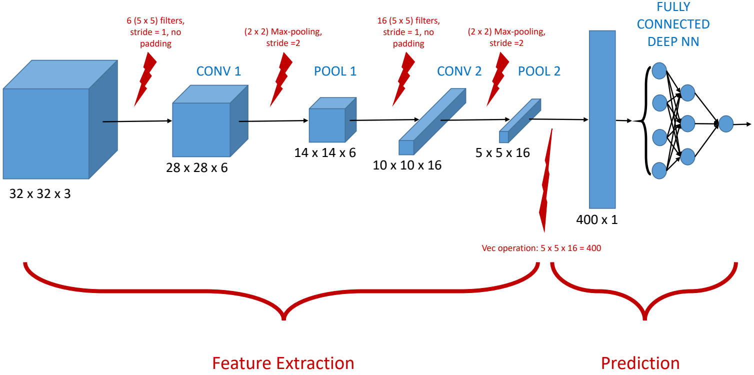
- Convolutional Neural Networks (CNNs) are a class of Neural Network models that have proven successful in image recognition and classification.
- Multi-layer network consisting of different key elements:
  - Convolutional layer (one or more)
  - Nonlinear transformation
  - Pooling (dimension reduction)
  - Fully-connected (deep) feed-forward neural network

- Sequence of layers:

$$\text{convolution} + \text{nonlinear transformation} \to \text{pooling} \to$$
$$\text{convolution} + \text{nonlinear transformation} \to \text{pooling} \to \cdots \to$$
$$\text{convolution} + \text{nonlinear transformation} \to \text{pooling} \to$$
$$\text{Fully-connected (deep) NN}$$

Convolutional Neural Networks

Introduction



- To a computer, an image is a matrix of pixels.
- Each entry of the matrix is the intensity of the pixel: $0 - 255$ (grayscale)
- The dimension of the matrix is the resolution of the image.
- For colored images, there is a third dimension to represent the color channels: Red (R), Green (G) and Blue (B).
- Therefore the image is a three-dimensional matrix (tensor): Height $\times$ Width $\times 3$.

- An **image kernel** is a small matrix used to apply effects, such as blurring, sharpening, outlining or embossing.
- In Machine Learning, kernels are used for "feature extraction", a technique for determining the most important portions of an image.
- In this context, the process is referred to more generally as **convolution**.
- A nice webpage: `https://setosa.io/ev/image-kernels/`

The Convolutional Layer

- Input data: $\boldsymbol{X} \in \mathbb{R}^{M\times N}$
- Filter (kernel): $\boldsymbol{W} \in \mathbb{R}^{Q\times R}$
  - $\boldsymbol{W}$ is usually unknown
- Output $\boldsymbol{O}$ is of a smaller dimension than the input due to border effects. For $i = 1, \ldots M - Q + 1$, $j = 1, \ldots, N - R + 1$:
  - $O_{ij} = \sum_{q=1}^{Q} \sum_{r=1}^{R} [\boldsymbol{W} \odot [\boldsymbol{X}]_{i:i+Q-1, j:j+R-1}]_{q,r}$.

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

The Convolutional Layer

“Convolution”

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Input data: 6 x 6 matrix

*

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

Filter: 3 x 3 matrix

=

-7			

Output data: 4 x 4 matrix

Result of the element-wise product and sum of the filter matrix and the original matrix: 3 x 1 + 1 x 1 + 1 x 2 + 1 x 0 + 0 x 0 + 3 x 0 + 1 x (-1) + 7 x (-1) + 5 x (-1) = -7

“Convolution”

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Input data: 6 x 6 matrix

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

Filter: 3 x 3 matrix

-7	-2		

Output data: 4 x 4 matrix

Result of the element-wise product and sum of the filter matrix and the original matrix: 1 x 1 + 0 x 1+ 3 x 1 + 1 x 0 + 7 x 0 + 5 x 0 + 2 x (-1) + 3 x (-1) + 1 x (-1) = -2

The Convolutional Layer

“Convolution”

$$\begin{bmatrix} 3 & 1 & 1 & 2 & 8 & 4 \\ 1 & 0 & 7 & 3 & 2 & 6 \\ 2 & 3 & 5 & 1 & 1 & 3 \\ 1 & 4 & 1 & 2 & 6 & 5 \\ 3 & 2 & 1 & 3 & 7 & 2 \\ 9 & 2 & 6 & 2 & 5 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} -7 & -2 & 2 & \\ & & & \\ & & & \\ & & & \end{bmatrix}$$

Input data: 6 x 6 matrix

Filter: 3 x 3 matrix

Output data: 4 x 4 matrix

Result of the element-wise product and sum of the filter matrix and the original matrix: 1 x 1 + 7 x 1+ 5 x 1 + 2 x 0 + 3 x 0 + 1 x 0 + 8 x (-1) + 2 x (-1) + 1 x (-1) = 2

The Convolutional Layer

"Convolution"

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Input data: 6 x 6 matrix

$*$

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

Filter: 3 x 3 matrix

$=$

-7	-2	2	-7

Output data: 4 x 4 matrix

Result of the element-wise product and sum of the filter matrix and the original matrix: 2 x 1 + 3 x 1+ 1 x 1 + 8 x 0 + 2 x 0 + 1 x 0 + 4 x (-1) + 6 x (-1) + 3 x (-1) = -7

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

The Convolutional Layer

"Convolution"

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Input data: 6 x 6 matrix

*

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

Filter: 3 x 3 matrix

=

-7	-2	2	-7
-9			

Output data: 4 x 4 matrix

Result of the element-wise product and sum of the filter matrix and the original matrix: 1 x 1 + 2 x 1+ 1 x 1 + 0 x 0 + 3 x 0 + 4 x 0 + 7 x (-1) + 5 x (-1) + 1 x (-1) = -9

- For $i = 1, \ldots M - Q + 1$, $j = 1, \ldots, N - R + 1$:

$$O_{ij} = \sum_{q=1}^{Q} \sum_{r=1}^{R} [\boldsymbol{W} \odot [\boldsymbol{X}]_{i:i+Q-1,j:j+R-1}]_{q,r}$$

$$O_{ij} = \boldsymbol{\iota}_Q' \left(\boldsymbol{W} \odot [\boldsymbol{X}]_{i:i+Q-1,j:j+R-1} \right) \boldsymbol{\iota}_R$$

where:

  * $\odot$ is the element-by-element multiplication;
  * $\boldsymbol{\iota}_Q \in \mathbb{R}^Q$ and $\boldsymbol{\iota}_R^R$ are vector of ones;
  * $[\boldsymbol{X}]_{i:i+Q-1,j:j+R-1}$ is the block of the matrix X running from row i to row $i + Q - j$ and from column j to column $j + R - 1$; and
  * $[\boldsymbol{X}]_{i:i+Q-1,j:j+R-1}]_{q,r}$ is the element of $[\boldsymbol{X}]_{i:i+Q-1,j:j+R-1}$ in position (q, r).

The Convolutional Layer

- O_{ij} is the discrete convolution between $\boldsymbol{W}$ and $[\boldsymbol{X}]_{i:i+Q-1,j:j+R-1}$:

$$O_{ij} = \boldsymbol{W} * [\boldsymbol{X}]_{i:i+Q-1,j:j+R-1}$$

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

The Convolutional Layer

Stride

Stride (downsampling) $\longrightarrow$ reduce problem dimension

How many “pixels” we move at each step.

“Convolution” with stride 1

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Input data: 6 x 6 matrix

“Convolution” with stride 2

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Input data: 6 x 6 matrix

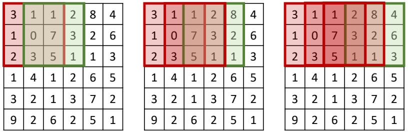
“Convolution” with stride 3

3	1	1	2	8	4
1	0	7	3	2	6
2	3	5	1	1	3
1	4	1	2	6	5
3	2	1	3	7	2
9	2	6	2	5	1

Input data: 6 x 6 matrix

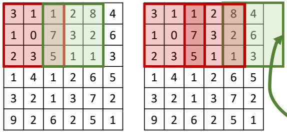
The Convolutional Layer

Stride



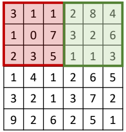
Stride = 1: slide one “pixel” each step

For a 6 x 6 original matrix, the convolution results in a 4 x 4 matrix



Stride = 2: slide two “pixels” each step

For a 6 x 6 original matrix, the convolution results in a 3 x 3 matrix (problems with edge effects)



Stride = 3: slide three “pixels” each step

For a 6 x 6 original matrix, the convolution results in a 2 x 2 matrix

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

Padding $\longrightarrow$ avoid border effects
Output and input with the same dimension

0	0	0	0	0	0	0	0
0	3	1	1	2	8	4	0
0	1	0	7	3	2	6	0
0	2	3	5	1	1	3	0
0	1	4	1	2	6	5	0
0	3	2	1	3	7	2	0
0	9	2	6	2	5	1	0
0	0	0	0	0	0	0	0

The Convolutional Layer

Padding

0	0	0	0	0	0	0	0
0	3	1	1	2	8	4	0
0	1	0	7	3	2	6	0
0	2	3	5	1	1	3	0
0	1	4	1	2	6	5	0
0	3	2	1	3	7	2	0
0	9	2	6	2	5	1	0
0	0	0	0	0	0	0	0

*

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

=

3					

The Convolutional Layer

Padding

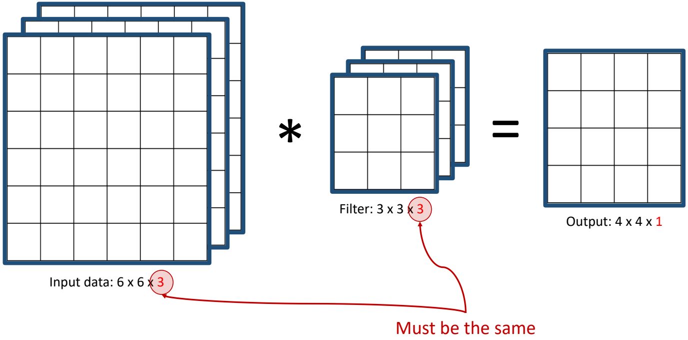
$$
\begin{array}{|c|c|c|c|c|c|c|c|}
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline
0 & 3 & 1 & 1 & 2 & 8 & 4 & 0 \\ \hline
0 & 1 & 0 & 7 & 3 & 2 & 6 & 0 \\ \hline
0 & 2 & 3 & 5 & 1 & 1 & 3 & 0 \\ \hline
0 & 1 & 4 & 1 & 2 & 6 & 5 & 0 \\ \hline
0 & 3 & 2 & 1 & 3 & 7 & 2 & 0 \\ \hline
0 & 9 & 2 & 6 & 2 & 5 & 1 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline
\end{array}
*
\begin{array}{|c|c|c|}
\hline
1 & 0 & -1 \\ \hline
1 & 0 & -1 \\ \hline
1 & 0 & -1 \\ \hline
\end{array}
=
\begin{array}{|c|c|c|c|c|c|}
\hline
3 & -4 & & & & \\ \hline
 & & & & & \\ \hline
 & & & & & \\ \hline
 & & & & & \\ \hline
 & & & & & \\ \hline
 & & & & & \\ \hline
\end{array}
$$

The Convolutional Layer

Padding

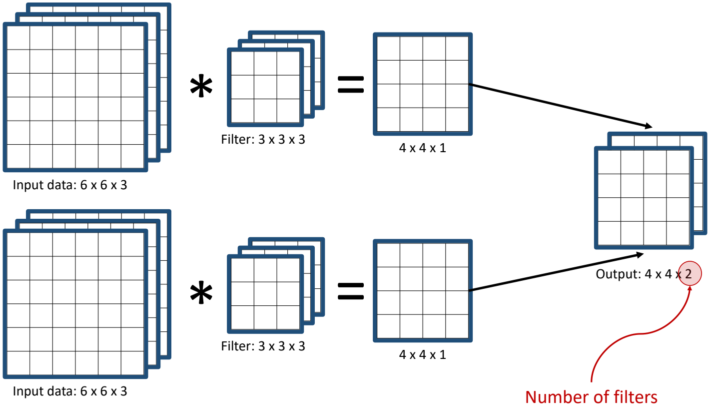
$$
\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 1 & 1 & 2 & 8 & 4 & 0 \\ 0 & 1 & 0 & 7 & 3 & 2 & 6 & 0 \\ 0 & 2 & 3 & 5 & 1 & 1 & 3 & 0 \\ 0 & 1 & 4 & 1 & 2 & 6 & 5 & 0 \\ 0 & 3 & 2 & 1 & 3 & 7 & 2 & 0 \\ 0 & 9 & 2 & 6 & 2 & 5 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 3 & -4 & -2 & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{bmatrix}
$$

Three channels define colored images: R (red), B (blue), and G (green).



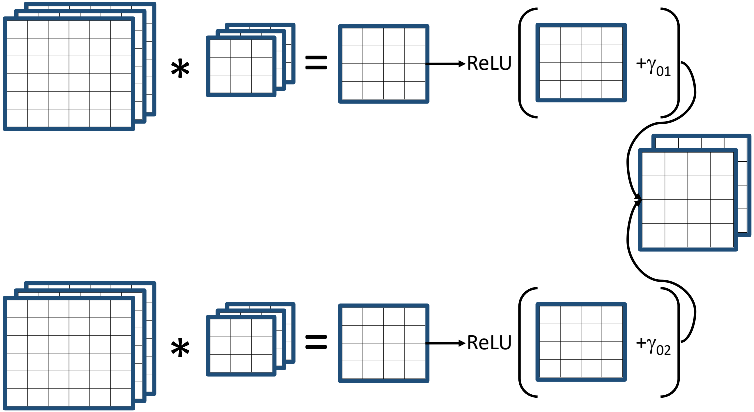
The Convolutional Layer

Multiple Filters

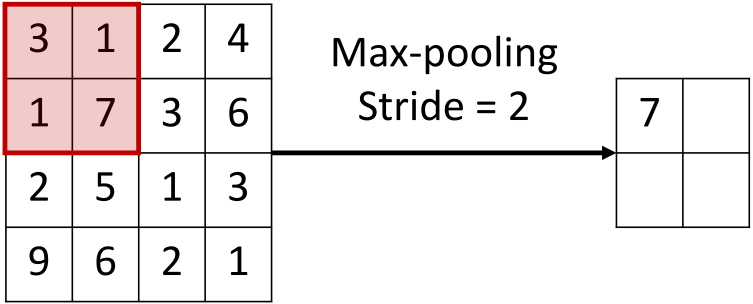


The Convolutional Layer

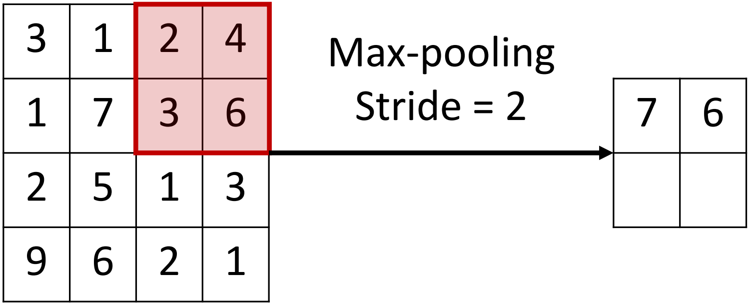
Nonlinearity



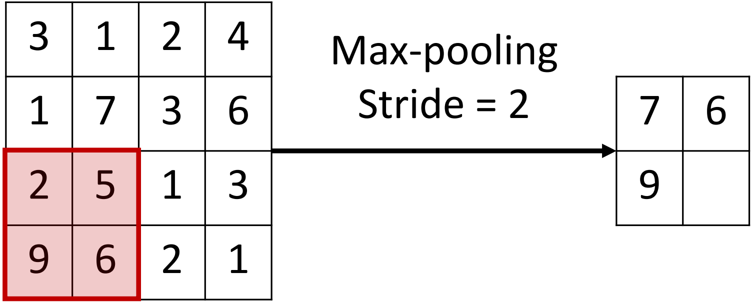
The Pooling Layer



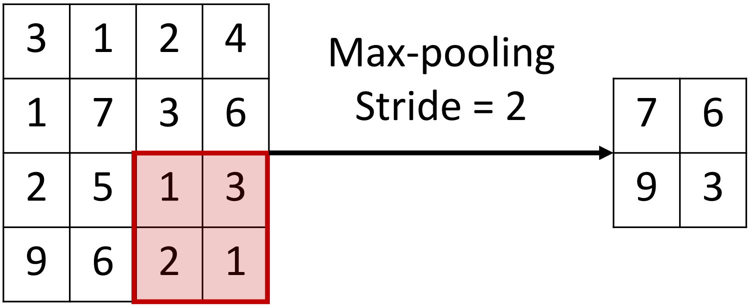
The Pooling Layer



The Pooling Layer

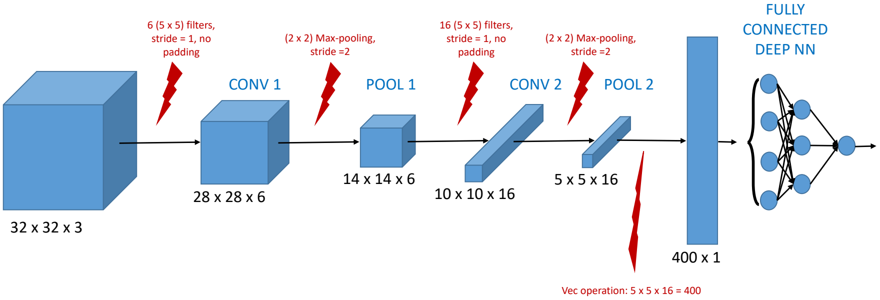


The Pooling Layer



The Convolutional Neural Network

Putting Pieces Together



- Hyperparameters:
  1. number of convolution layers (C);
  2. number of pooling layers (P);
  3. number (K_c) and dimensions (Q_c height, R_c width and S_c depth) of filters in each convolution layer $c = 1, \ldots, C$;
  4. architecture of the deep neural network.
- Parameters:
  1. Filter weights: $\boldsymbol{W}_{ic} \in \mathbb{R}^{Q_c \times R_c \times S_c}$, $i = 1, \ldots, K_c$, $c = 1, \ldots, C$;
  2. ReLU biases: $\boldsymbol{\gamma}_c \in \mathbb{R}^{K_c}$, $c = 1, \ldots, C$;
  3. All the parameters of the fully connected deep NN: $\boldsymbol{\psi}$.

Agenda for Today

Neural Networks

- Introduction
- Mathematical definition

Deep Neural Networks

Convolutional Neural Networks

Long Short Term Neural Networks

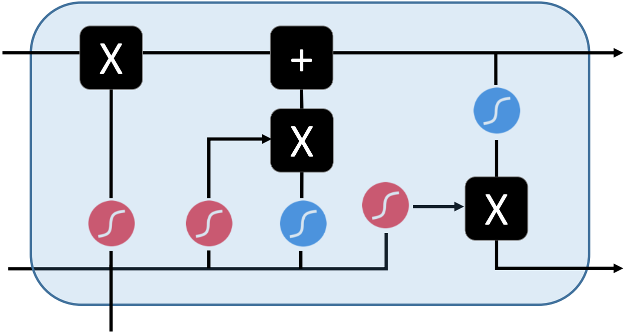
- Simple Recurrent Neural Network

$$h_t = \theta_h f(h_{t-1}) + \theta_x x_t$$
$$\widehat{y}_t = \theta_y f(h_t)$$

- RNNs suffer from the vanishing/exploding gradient problem.
  - Set the cost function to be

$$\mathcal{Q}_T(\boldsymbol{\theta}) = \frac{1}{T}\sum_{t=1}^{T}(y_t - \widehat{y}_t)^2$$

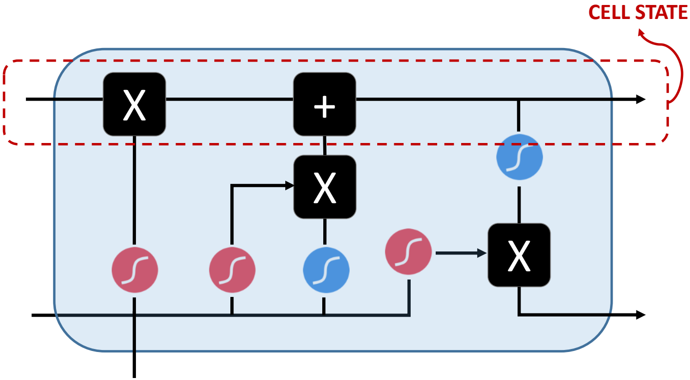
  - $\frac{\partial \mathcal{Q}_T(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ can be very small or diverge.
- Solution: LSTM



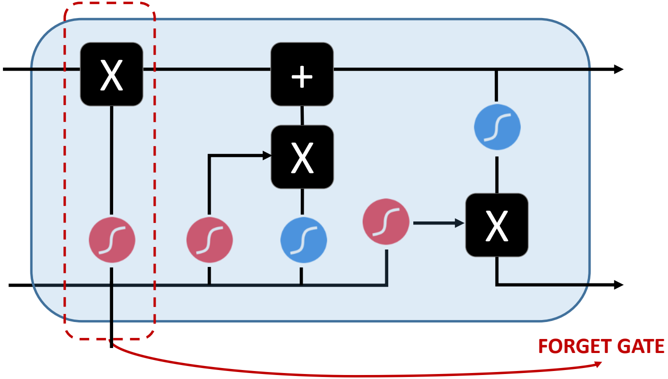
- Red circles: logistic functions
- Blue circles: hyperbolic tangent functions

Long Short Term Memory Networks

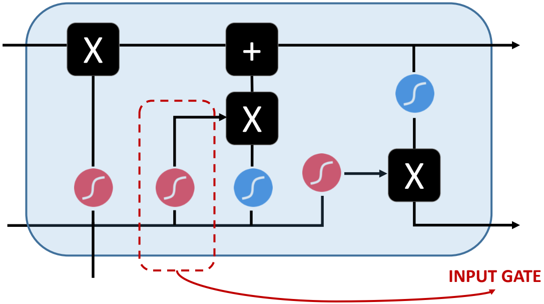
The LSTM Cell: The Cell State



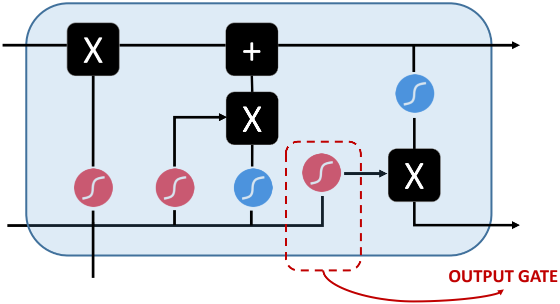
- The cell state: a bit of memory to the LSTM to "remember" the past.
- LSTM learns to keep only relevant information and forget nonrelevant data.



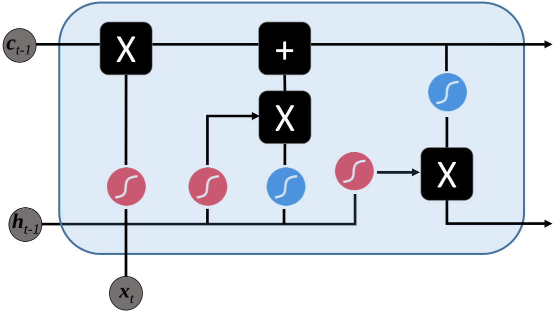
- The forget gate tells which information to throw away from the cell state.
- It is composed of a logistic function



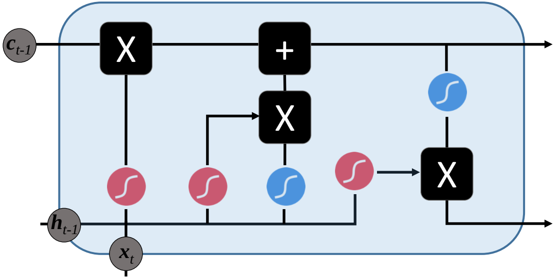
- The input gate tells which new information should be stored in the cell state.
- It is composed of a logistic function



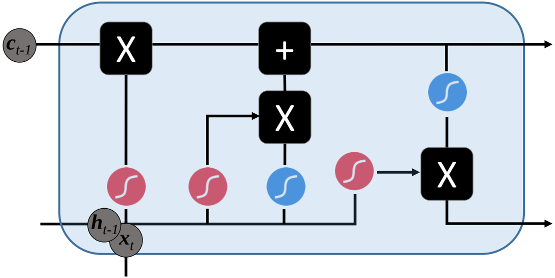
- The output gate provides the activation to the final output of the LSTM block at time t.
- It is composed of a logistic function



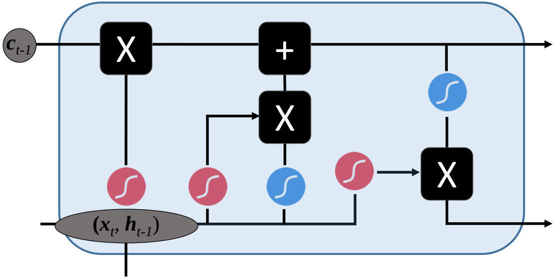
- At time t:
  1. Inputs: $\boldsymbol{x}_t \in \mathbb{R}^p$ and past *hidden state* $\boldsymbol{h}_{t-1} \in \mathbb{R}^q$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$



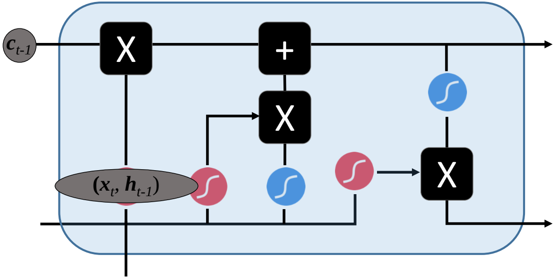
- At time t:
  1. Inputs: $\boldsymbol{x}_t \in \mathbb{R}^p$ and past *hidden state* $\boldsymbol{h}_{t-1} \in \mathbb{R}^q$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$



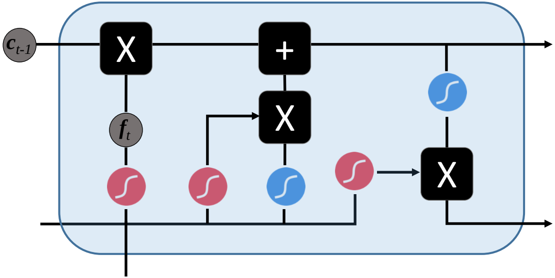
- At time t:
  1. Inputs: $\boldsymbol{x}_t \in \mathbb{R}^p$ and past *hidden state* $\boldsymbol{h}_{t-1} \in \mathbb{R}^q$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$



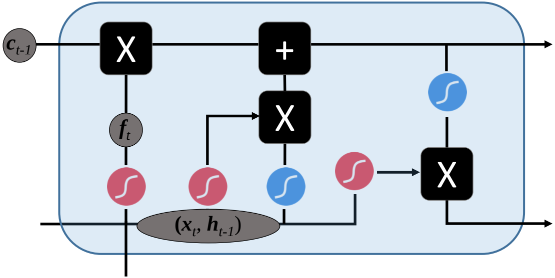
- At time t:
  1. Inputs concatenate: $\boldsymbol{z}_t = (\boldsymbol{x}_t', \boldsymbol{h}_{t-1}')'$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$



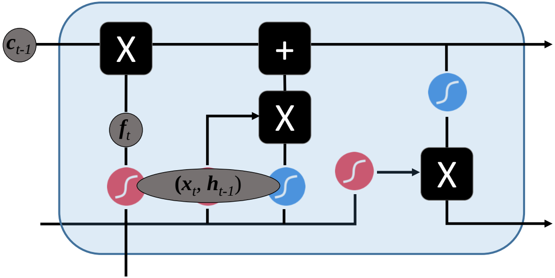
- At time t:
  1. Logistic activation: $\boldsymbol{f}_t = \text{logistic}(\boldsymbol{\Gamma}_f \boldsymbol{z}_t + \boldsymbol{\gamma}_{0f}) \in \mathbb{R}^q$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$



- At time t:
  1. Logistic activation: $\boldsymbol{f}_t = \text{logistic}(\boldsymbol{\Gamma}_f \boldsymbol{z}_t + \boldsymbol{\gamma}_{0f}) \in \mathbb{R}^q$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$



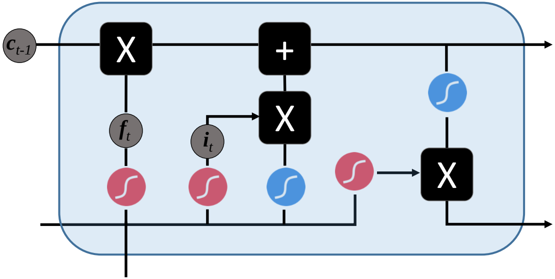
- At time t:
  1. Logistic activation: $\boldsymbol{i}_t = \text{logistic}(\boldsymbol{\Gamma}_i' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0i}) \in \mathbb{R}^q$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$



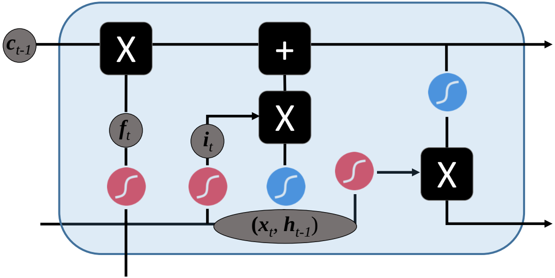
- At time t:
  1. Logistic activation: $\boldsymbol{i}_t = \text{logistic}(\boldsymbol{\Gamma}_i' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0i}) \in \mathbb{R}^q$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$

Long Short Term Memory Networks

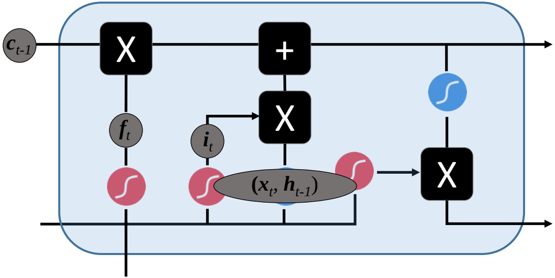
The Information Flow



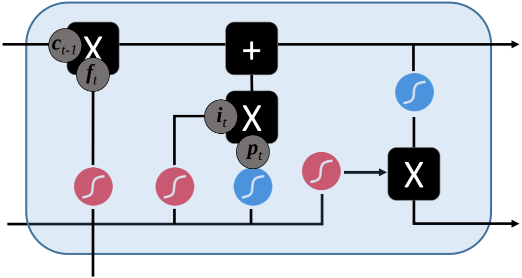
- At time t:
  1. Logistic activation: $\boldsymbol{i}_t = \text{logistic}(\boldsymbol{\Gamma}_i' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0i}) \in \mathbb{R}^q$
  2. Running state cell: $\boldsymbol{c}_{t-1} \in \mathbb{R}^q$



- At time t:
  1. Logistic activation: $\boldsymbol{i}_t = \text{logistic}(\boldsymbol{\Gamma}_i' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0i}) \in \mathbb{R}^q$
  2. Tanh activation: $\boldsymbol{p}_t = \tanh(\boldsymbol{\Gamma}_p' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0p}) \in \mathbb{R}^q$ (potential state cell)



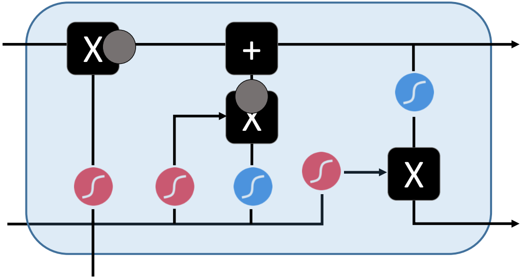
- At time t:
  1. Logistic activation: $\boldsymbol{i}_t = \text{logistic}(\boldsymbol{\Gamma}_i' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0i}) \in \mathbb{R}^q$
  2. Tanh activation: $\boldsymbol{p}_t = \tanh(\boldsymbol{\Gamma}_p' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0p}) \in \mathbb{R}^q$ (potential state cell)



- At time t:
  1. $\boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t$
  2. $\boldsymbol{i}_t \odot \boldsymbol{p}_t$

Long Short Term Memory Networks

The Information Flow

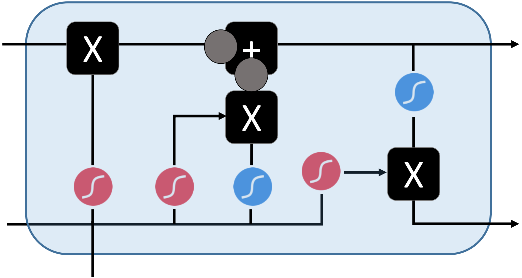


- At time t:
  1. $\boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t$
  2. $\boldsymbol{i}_t \odot \boldsymbol{p}_t$

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

Long Short Term Memory Networks

The Information Flow

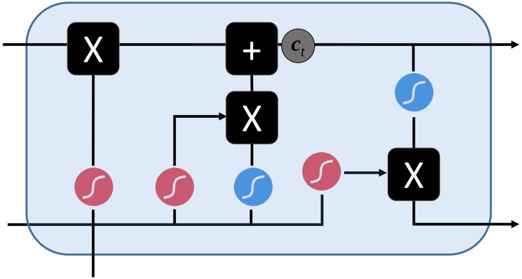


- At time t:
  1. $\boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t$
  2. $\boldsymbol{i}_t \odot \boldsymbol{p}_t$

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

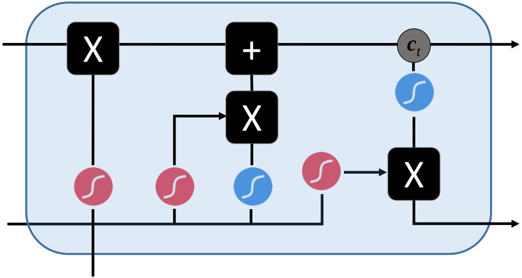
Long Short Term Memory Networks

The Information Flow



- At time t:
  1. $\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t + \boldsymbol{i}_t \odot \boldsymbol{p}_t$

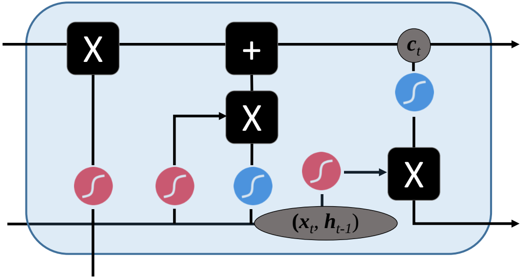
ECONOMETRICS LAB
QUANTITATIVE RESEARCH • UIUC



- At time t:
  1. $\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t + \boldsymbol{i}_t \odot \boldsymbol{p}_t$

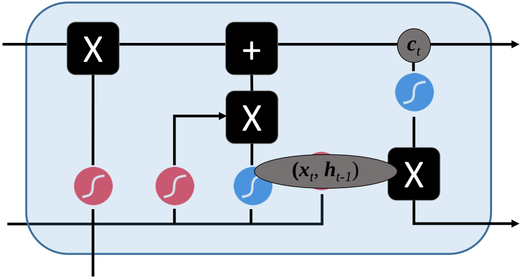
Long Short Term Memory Networks

The Information Flow



- At time t:
  1. $\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t + \boldsymbol{i}_t \odot \boldsymbol{p}_t$
  2. Logistic activation: $\boldsymbol{o}_t = \text{logistic}(\boldsymbol{\Gamma}_o' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0o}) \in \mathbb{R}^q$

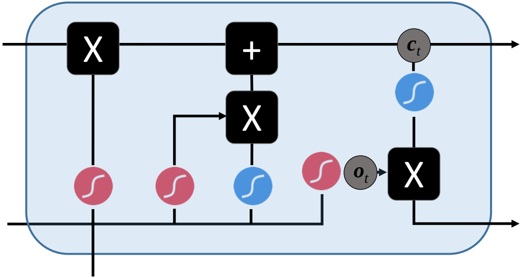
ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC



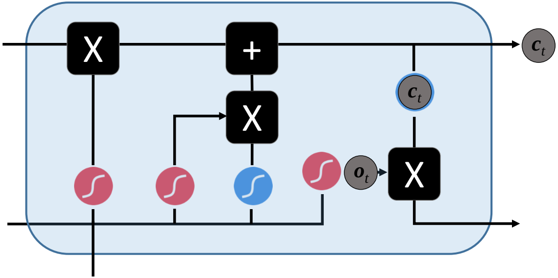
- At time t:
  1. $\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t + \boldsymbol{i}_t \odot \boldsymbol{p}_t$
  2. Logistic activation: $\boldsymbol{o}_t = \text{logistic}(\boldsymbol{\Gamma}'_o \boldsymbol{z}_t + \boldsymbol{\gamma}_{0o}) \in \mathbb{R}^q$

Long Short Term Memory Networks

The Information Flow

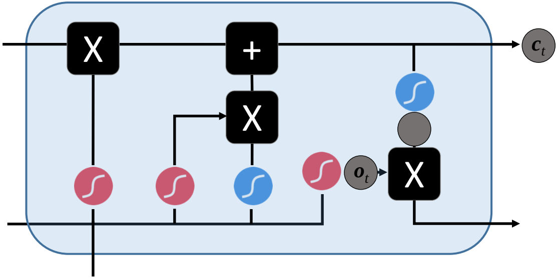


- At time t:
  1. $\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t + \boldsymbol{i}_t \odot \boldsymbol{p}_t$
  2. Logistic activation: $\boldsymbol{o}_t = \text{logistic}(\boldsymbol{\Gamma}'_o \boldsymbol{z}_t + \boldsymbol{\gamma}_{0o}) \in \mathbb{R}^q$

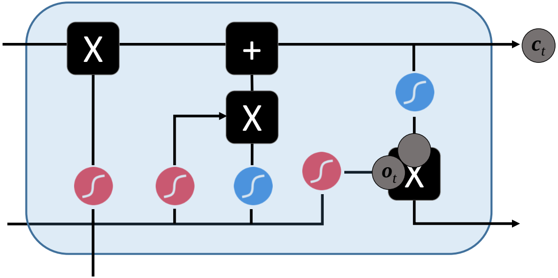


- At time t:
  1. $\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t + \boldsymbol{i}_t \odot \boldsymbol{p}_t$
  2. Logistic activation: $\boldsymbol{o}_t = \text{logistic}(\boldsymbol{\Gamma}_o' \boldsymbol{z}_t + \gamma_{0o}) \in \mathbb{R}^q$

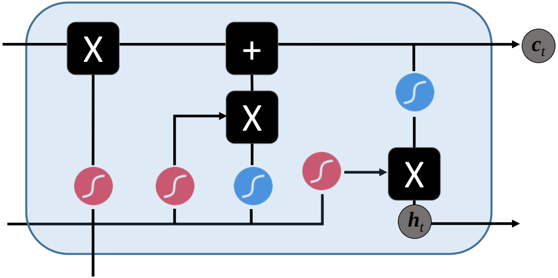
ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC



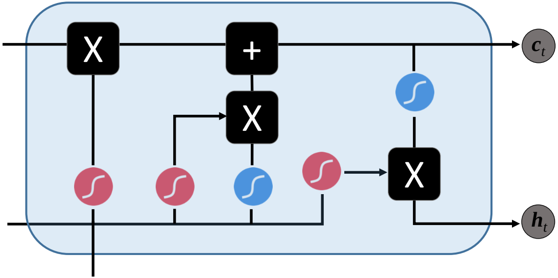
- At time t:
  1. $\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \odot \boldsymbol{f}_t + \boldsymbol{i}_t \odot \boldsymbol{p}_t$
  2. Logistic activation: $\boldsymbol{o}_t = \text{logistic}(\boldsymbol{\Gamma}_o' \boldsymbol{z}_t + \boldsymbol{\gamma}_{0o}) \in \mathbb{R}^q$



- At time t:
  1. $\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)$



- At time t:
  1. $\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)$



- At time t:
  1. $\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)$
     $y_t = \boldsymbol{\gamma}_y' \boldsymbol{h}_t + \gamma_{0y}$

LSTM Network

Formal (Mathematical) representation

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.

LSTM Network

Formal (Mathematical) representation

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.
- Given input $\boldsymbol{x}_t$ do for $t \in \{1, \ldots, T\}$:

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

LSTM Network

Formal (Mathematical) representation

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.
- Given input $\boldsymbol{x}_t$ do for $t \in \{1, \dots, T\}$:

$$\boldsymbol{f}_t = \text{Logistic}(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f)$$

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

LSTM Network

Formal (Mathematical) representation

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.
- Given input $\boldsymbol{x}_t$ do for $t \in \{1, \ldots, T\}$:

$$\boldsymbol{f}_t = \text{Logistic}(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f)$$
$$\boldsymbol{i}_t = \text{Logistic}(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i)$$

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.
- Given input $\boldsymbol{x}_t$ do for $t \in \{1, \ldots, T\}$:

$$\begin{aligned}
\boldsymbol{f}_t &= \text{Logistic}(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{i}_t &= \text{Logistic}(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \\
\boldsymbol{o}_t &= \text{Logistic}(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o)
\end{aligned}$$

LSTM Network

Formal (Mathematical) representation

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.
- Given input $\boldsymbol{x}_t$ do for $t \in \{1, \ldots, T\}$:

$$
\begin{aligned}
\boldsymbol{f}_t &= \text{Logistic}(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{i}_t &= \text{Logistic}(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \\
\boldsymbol{o}_t &= \text{Logistic}(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \\
\boldsymbol{p}_t &= \text{Tanh}(\boldsymbol{W}_c \boldsymbol{x}_t + \boldsymbol{U}_c \boldsymbol{h}_{t-1} + \boldsymbol{b}_c)
\end{aligned}
$$

LSTM Network

Formal (Mathematical) representation

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.
- Given input $\boldsymbol{x}_t$ do for $t \in \{1, \ldots, T\}$:

$$\boldsymbol{f}_t = \text{Logistic}(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f)$$
$$\boldsymbol{i}_t = \text{Logistic}(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i)$$
$$\boldsymbol{o}_t = \text{Logistic}(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o)$$
$$\boldsymbol{p}_t = \text{Tanh}(\boldsymbol{W}_c \boldsymbol{x}_t + \boldsymbol{U}_c \boldsymbol{h}_{t-1} + \boldsymbol{b}_c)$$
$$\boldsymbol{c}_t = (\boldsymbol{f}_t \odot \boldsymbol{c}_{t-1}) + (\boldsymbol{i}_t \odot \boldsymbol{p}_t)$$

LSTM Network

Formal (Mathematical) representation

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.
- Given input $\boldsymbol{x}_t$ do for $t \in \{1, \ldots, T\}$:

$$\boldsymbol{f}_t = \text{Logistic}(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f)$$

$$\boldsymbol{i}_t = \text{Logistic}(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i)$$

$$\boldsymbol{o}_t = \text{Logistic}(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o)$$

$$\boldsymbol{p}_t = \text{Tanh}(\boldsymbol{W}_c \boldsymbol{x}_t + \boldsymbol{U}_c \boldsymbol{h}_{t-1} + \boldsymbol{b}_c)$$

$$\boldsymbol{c}_t = (\boldsymbol{f}_t \odot \boldsymbol{c}_{t-1}) + (\boldsymbol{i}_t \odot \boldsymbol{p}_t)$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \text{Tanh}(\boldsymbol{c}_t)$$

LSTM Network

Formal (Mathematical) representation

- Initiate with $\boldsymbol{c}_0 = 0$ and $\boldsymbol{h}_0 = 0$.
- Given input $\boldsymbol{x}_t$ do for $t \in \{1, \dots, T\}$:

$$
\begin{aligned}
\boldsymbol{f}_t &= \text{Logistic}(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{i}_t &= \text{Logistic}(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \\
\boldsymbol{o}_t &= \text{Logistic}(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \\
\boldsymbol{p}_t &= \text{Tanh}(\boldsymbol{W}_c \boldsymbol{x}_t + \boldsymbol{U}_c \boldsymbol{h}_{t-1} + \boldsymbol{b}_c) \\
\boldsymbol{c}_t &= (\boldsymbol{f}_t \odot \boldsymbol{c}_{t-1}) + (\boldsymbol{i}_t \odot \boldsymbol{p}_t) \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \text{Tanh}(\boldsymbol{c}_t) \\
\boldsymbol{y}_t &= \boldsymbol{W}_y \boldsymbol{h}_t + \boldsymbol{b}_y
\end{aligned}
$$

ECONOMETRICS LAB
QUANTITATIVE RESEARCH @ UIUC

Obrigado