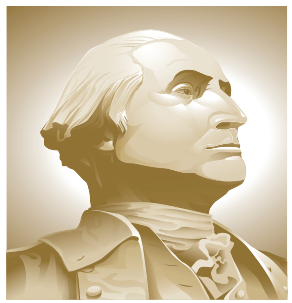
Introduction to Deep Learning

Machine Learning II

Lecture 4-a



THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

What is deep learning?

- Deep learning is a branch of machine learning involving algorithms that have many nonlinear processing stages.
- In most cases, deep learning refers to training neural networks that have many layers.
- Before approximately 2006, most neural network applications used one or two hidden layers.
- Since that time, the development of more powerful GPUs, and the associated general purpose programming languages, enabled larger neural networks to be tested.
- Larger networks require large data sets to prevent overfitting. The number of large data sets has increased dramatically in recent years.

Deep Learning Applications

- Google Deepmind – AlphaGO, the first computer GO program to beat a top professional GO player.
- Android operating system speech recognition.
- photosearch for Google+.
- Skype translator – speech recognition.
- Microsoft Cortana digital assistant.
- Facebook – Deep Face, face recognition.
- Apple – Siri

Course objective

- Learn about the most popular deep learning architectures.
- Learn about the most popular open source deep learning software frameworks.
- Learn how to implement deep networks using deep learning frameworks.

Course emphasis

- Key emphasis will be on implementation of deep learning concepts on GPUs using open source software frameworks.
- Will not cover basic machine learning concepts.
- Will assume knowledge of key ideas from linear algebra, optimization, probability, machine learning (as covered, for example, in Neural Network Design – hagan.okstate.edu/nnd.html)

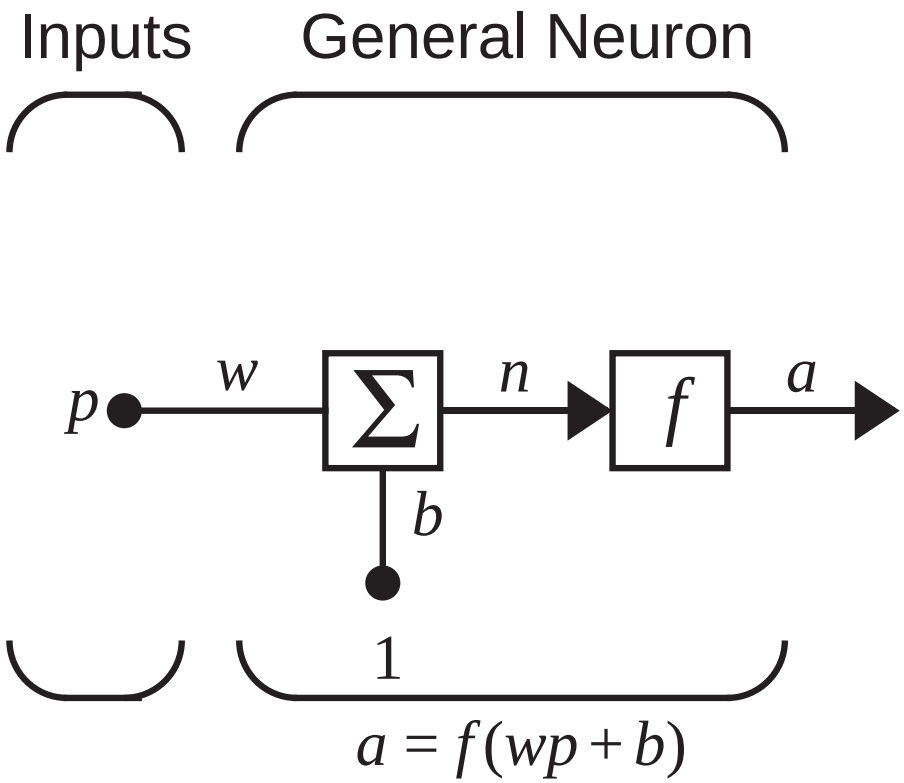
Course schedule

- Introduction
- Multilayer networks
- Training multilayer networks, gradient calculation
- Keras
- Convolution networks
- Training convolution networks, gradient calculation
- Tensorflow 2.0
- Long short term memory
- Training recurrent networks
- Pytorch

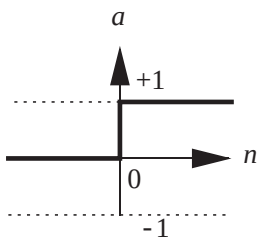
Brief history of deep learning

- Backpropagation for multilayer networks discovered, popularized (1974, 1982, 1985, 1986).
- Convolution networks introduced (1989).
- Long Short Term Memory network developed (1997).
- Deep belief network presented (2006).
- NVIDIA unveiled CUDA, a language for general purpose programming of GPUs (2006)

Basic network building block (neuron)

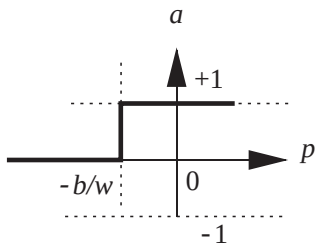


Transfer (activation) functions (1)



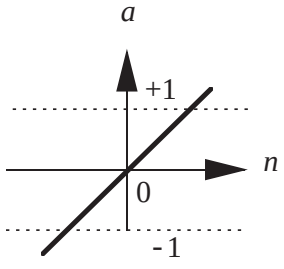
$a = hardlim(n)$

Hard Limit Transfer Function



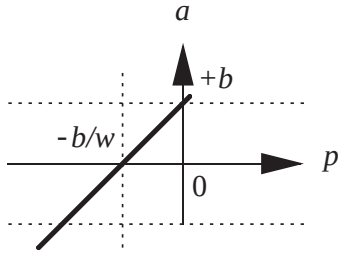
$a = hardlim(wp + b)$

Single-Input *hardlim* Neuron



$a = purelin(n)$

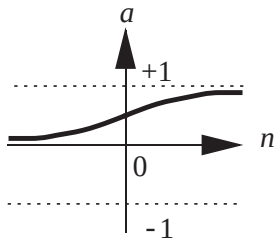
Linear Transfer Function



$a = purelin(wp + b)$

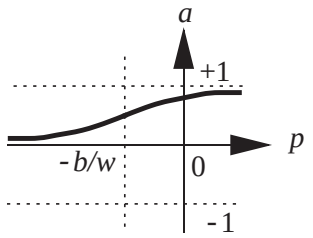
Single-Input *purelin* Neuron

Transfer (activation) functions (2)



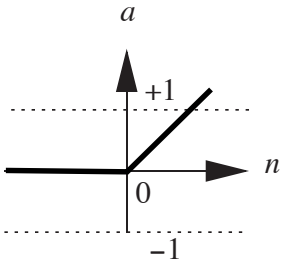
$a = logsig(n)$

Log-Sigmoid Transfer Function



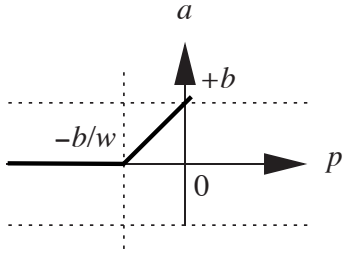
$a = logsig(wp + b)$

Single-Input *logsig* Neuron



$a = poslin(n)$

Positive Linear Function



$a = poslin(wp + b)$

Single-Input *poslin* Neuron

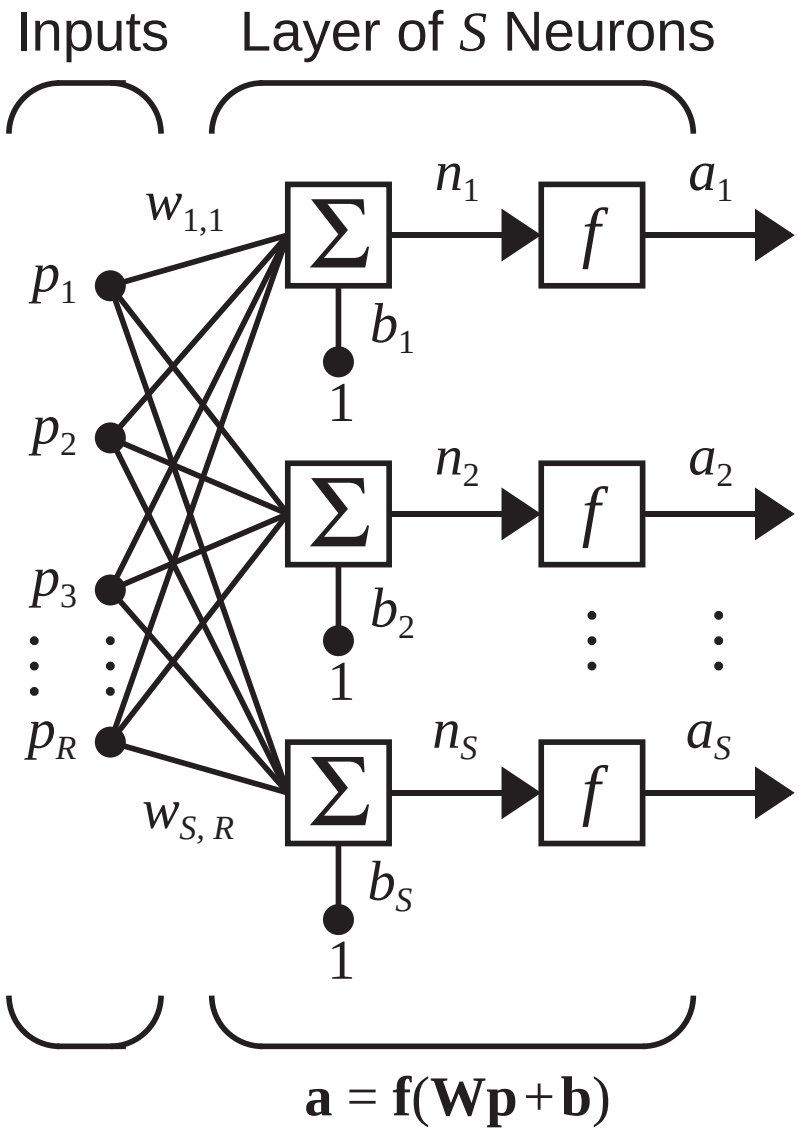
Transfer (activation) functions (3)

Softmax

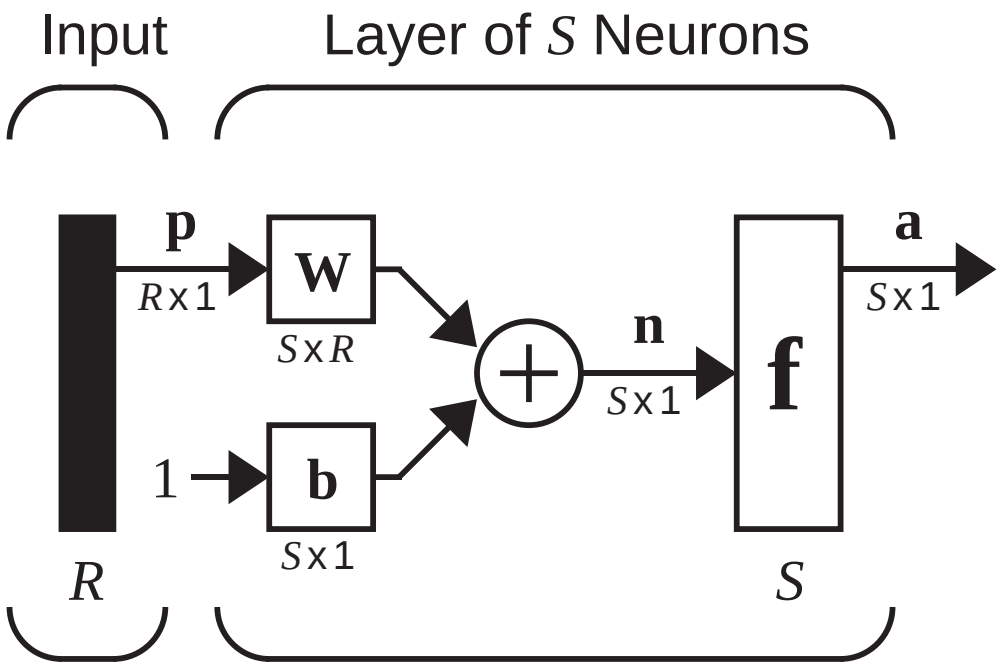
$$a_i = f_i(\mathbf{n}) = \frac{e^{n_i}}{\sum_{j=1}^{S} e^{n_j}}$$

Used at the output layer of a pattern recognition network with multiple output neurons.

Layer of neurons

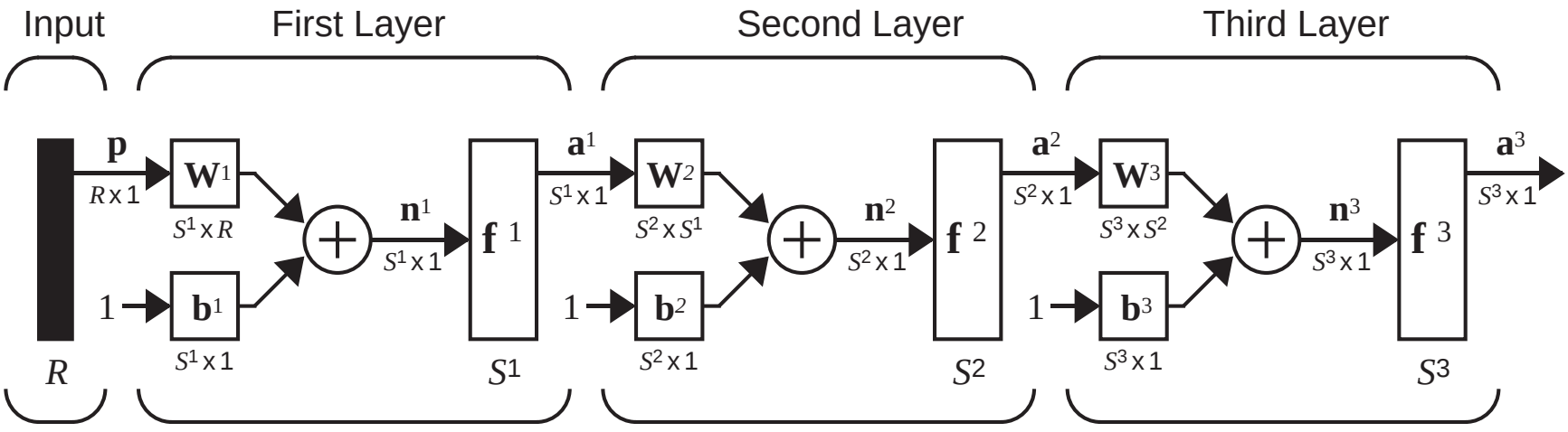


Matrix notation



$$\mathbf{a} = \mathbf{f}(\mathbf{W}\mathbf{p} + \mathbf{b})$$

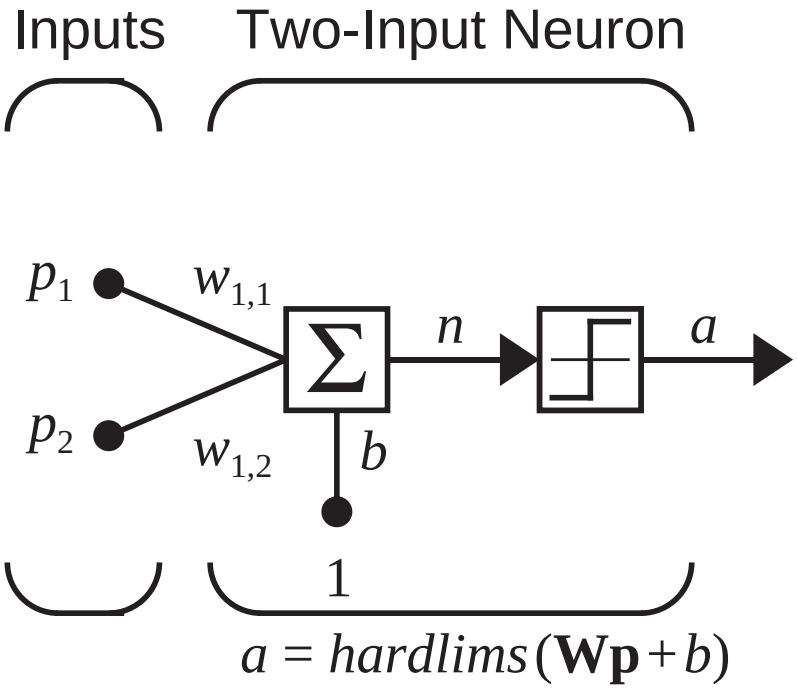
Multiple layer network



$$\mathbf{a}^1 = \mathbf{f}^1(\mathbf{W}^1\mathbf{p} + \mathbf{b}^1) \qquad \mathbf{a}^2 = \mathbf{f}^2(\mathbf{W}^2\mathbf{a}^1 + \mathbf{b}^2) \qquad \mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{a}^2 + \mathbf{b}^3)$$

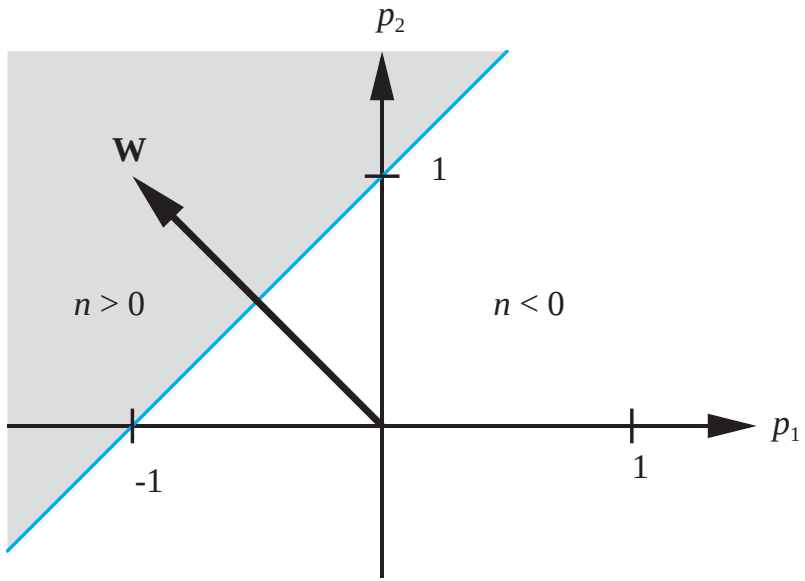
$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{f}^2(\mathbf{W}^2\mathbf{f}^1(\mathbf{W}^1\mathbf{p} + \mathbf{b}^1) + \mathbf{b}^2) + \mathbf{b}^3)$$

Single layer network decision boundary

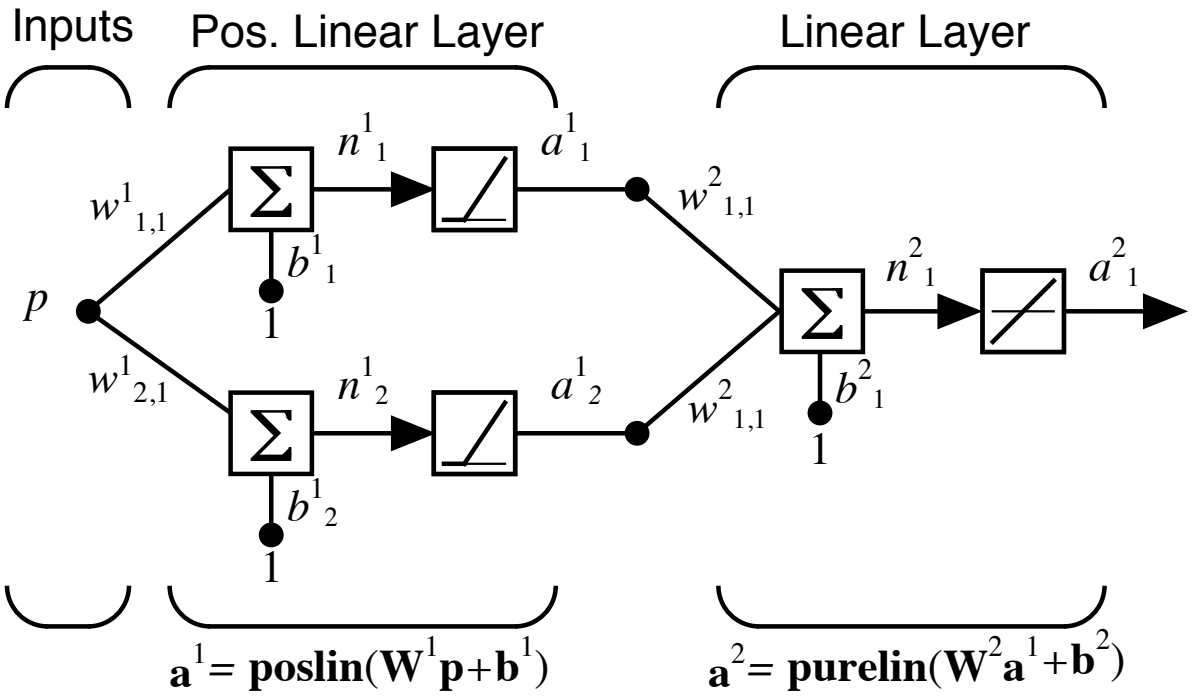


Decision boundary

$$n = \mathbf{W}\mathbf{p} + b == 0$$



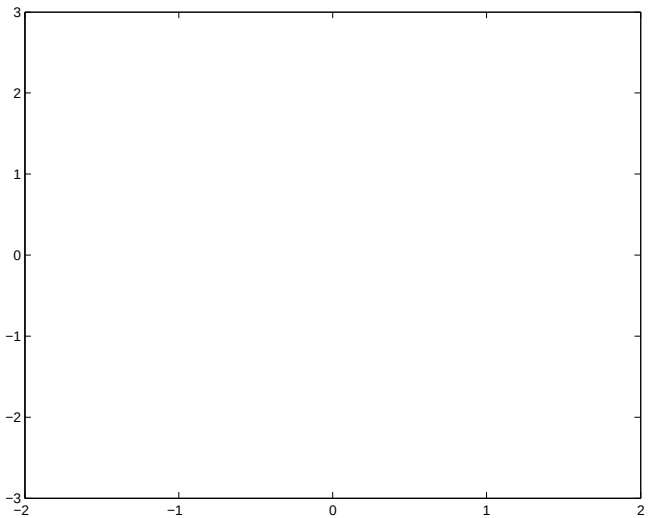
Poslin network

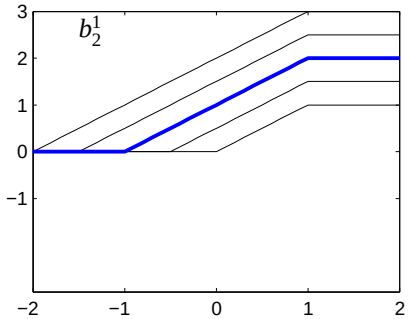


Poslin network function

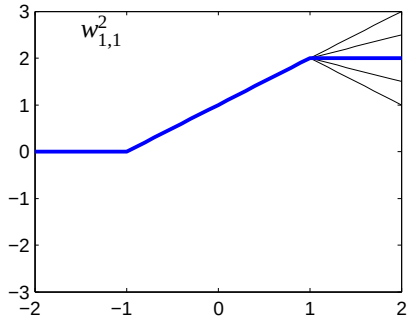
$$\mathbf{W}^1 = \begin{bmatrix} 1 & 1 \end{bmatrix}^T, \mathbf{b}^1 = \begin{bmatrix} -1 & 1 \end{bmatrix}^T$$

$$\mathbf{W}^2 = \begin{bmatrix} -1 & 1 \end{bmatrix}, \mathbf{b}^2 = [0]$$

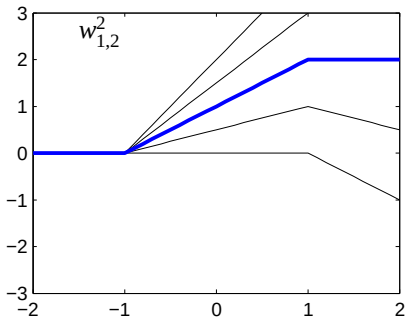




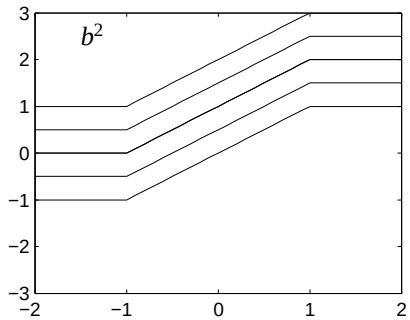
(a)



(b)

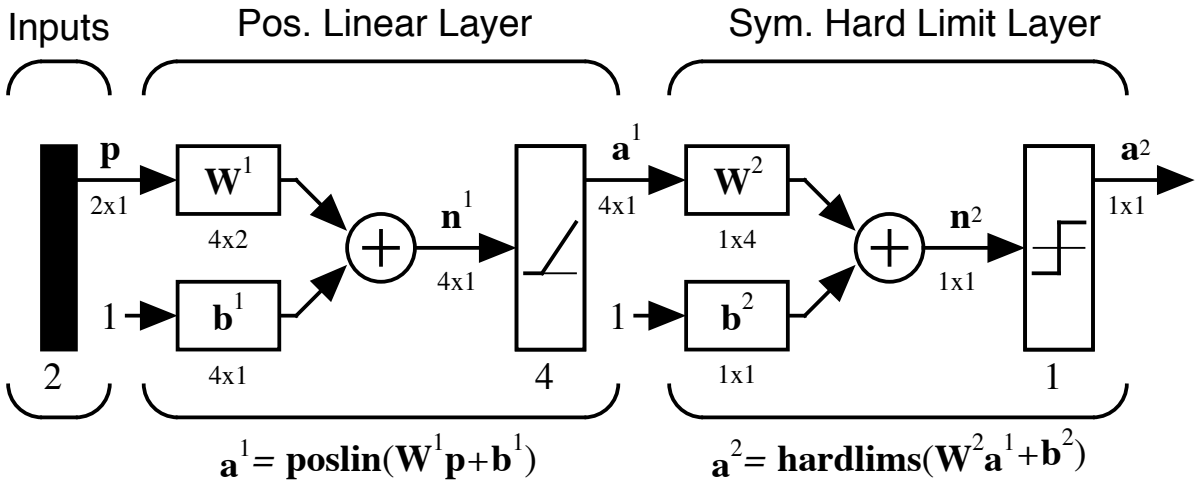


(c)



(d)

2D poslin network



$$\mathbf{W}^1 = \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}^T, \mathbf{b}^1 = \begin{bmatrix} -1 & 3 & 1 & 1 \end{bmatrix}^T$$

$$\mathbf{W}^2 = \begin{bmatrix} -1 & -1 & -1 & -1 \end{bmatrix}, \mathbf{b}^2 = [5]$$

2D Poslin network surface and decision boundary