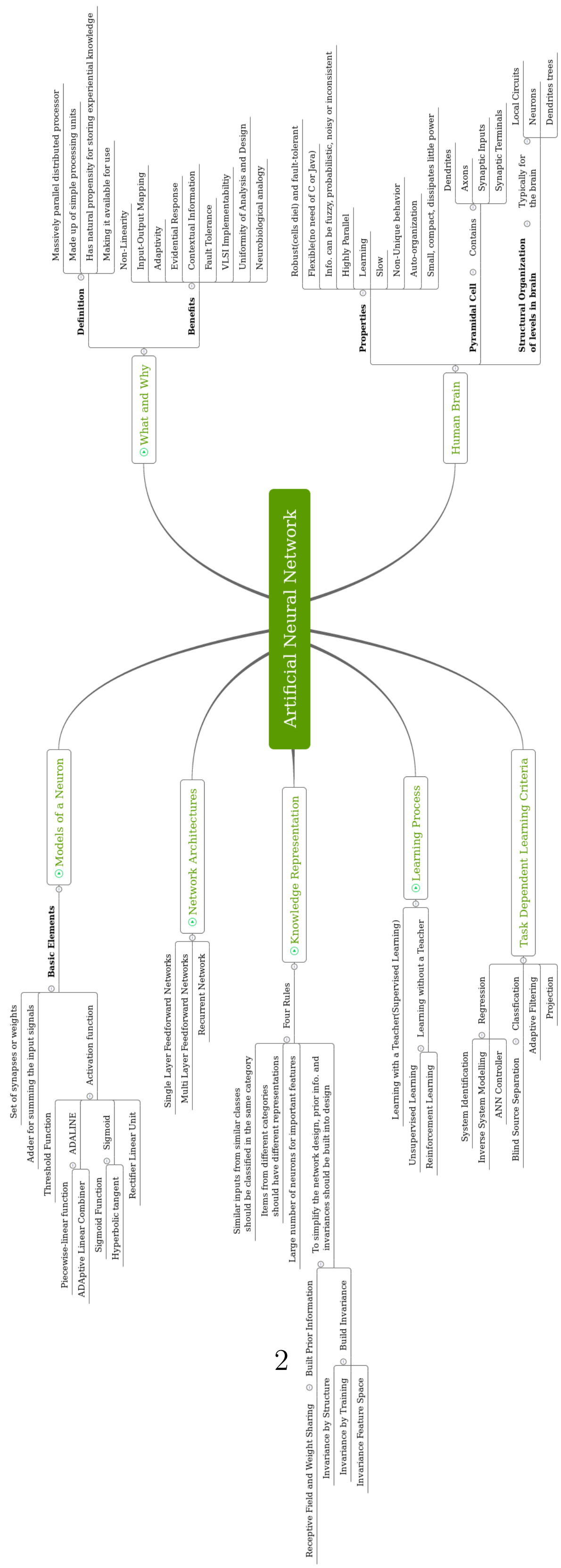
Neural Networks

Assignment 1

Ravikiran Bhat
Rubanraj Ravichandran
Ramesh Kumar

October 14, 2017

1 Mindmap



Artificial Neural Network
What and Why
Definition
Massively parallel distributed processor
Made up of simple processing units
Has natural propensity for storing experiential knowledge
Making it available for use
Benefits
Non-Linearity
Input-Output Mapping
Adaptivity
Evidential Response
Contextual Information
Fault Tolerance
VLSI Implementability
Uniformity of Analysis and Design
Neurobiological analogy
Human Brain
Properties
Robust(cells die!) and fault-tolerant
Flexible(no need of C or Java)
Info. can be fuzzy, probabilistic, noisy or inconsistent
Highly Parallel
Learning
Slow
Non-Unique behavior
Auto-organization
Small, compact, dissipates little power
Pyramidal Cell
Contains
Dendrites
Axons
Synaptic Inputs
Synaptic Terminals
Structural Organization of levels in brain
Typically for the brain
Local Circuits
Neurons
Dendrites trees
Models of a Neuron
Basic Elements
Set of synapses or weights
Adder for summing the input signals
Activation function
Threshold Function
ADALINE
Piecewise-linear function
ADAptive Linear Combiner
Sigmoid
Sigmoid Function
Hyperbolic tangent
Rectifier Linear Unit
Network Architectures
Single Layer Feedforward Networks
Multi Layer Feedforward Networks
Recurrent Network
Knowledge Representation
Four Rules
Similar inputs from similar classes should be classified in the same category
Items from different categories should have different representations
Large number of neurons for important features
To simplify the network design, prior info. and invariances should be built into design
Built Prior Information
Receptive Field and Weight Sharing
Build Invariance
Invariance by Structure
Invariance by Training
Invariance Feature Space
Learning Process
Learning with a Teacher(Supervised Learning)
Learning without a Teacher
Unsupervised Learning
Reinforcement Learning
Task Dependent Learning Criteria
Regression
System Identification
Inverse System Modelling
ANN Controller
Classification
Blind Source Separation
Adaptive Filtering
Projection

2 Models of a neuron

2.1 Exercise 1.1

```
In [1]: from sympy import *
        import sympy as sp
        sp.init_printing(use_latex=True)

        a,v = sp.symbols('a,v')
        sigmoid = 1/(1+sp.exp((-a)*(v)))
        sigmoid_derivative = sigmoid.diff(v)
        sigmoid_derivative
```

Out[1]:

$$\frac{ae^{-av}}{(1+e^{-av})^2}$$

At the origin, $v = 0$

```
In [16]: sigmoid_derivative.subs({v:0}).evalf()
```

Out[16]:

$$0.25a$$

2.2 Exercise 1.6

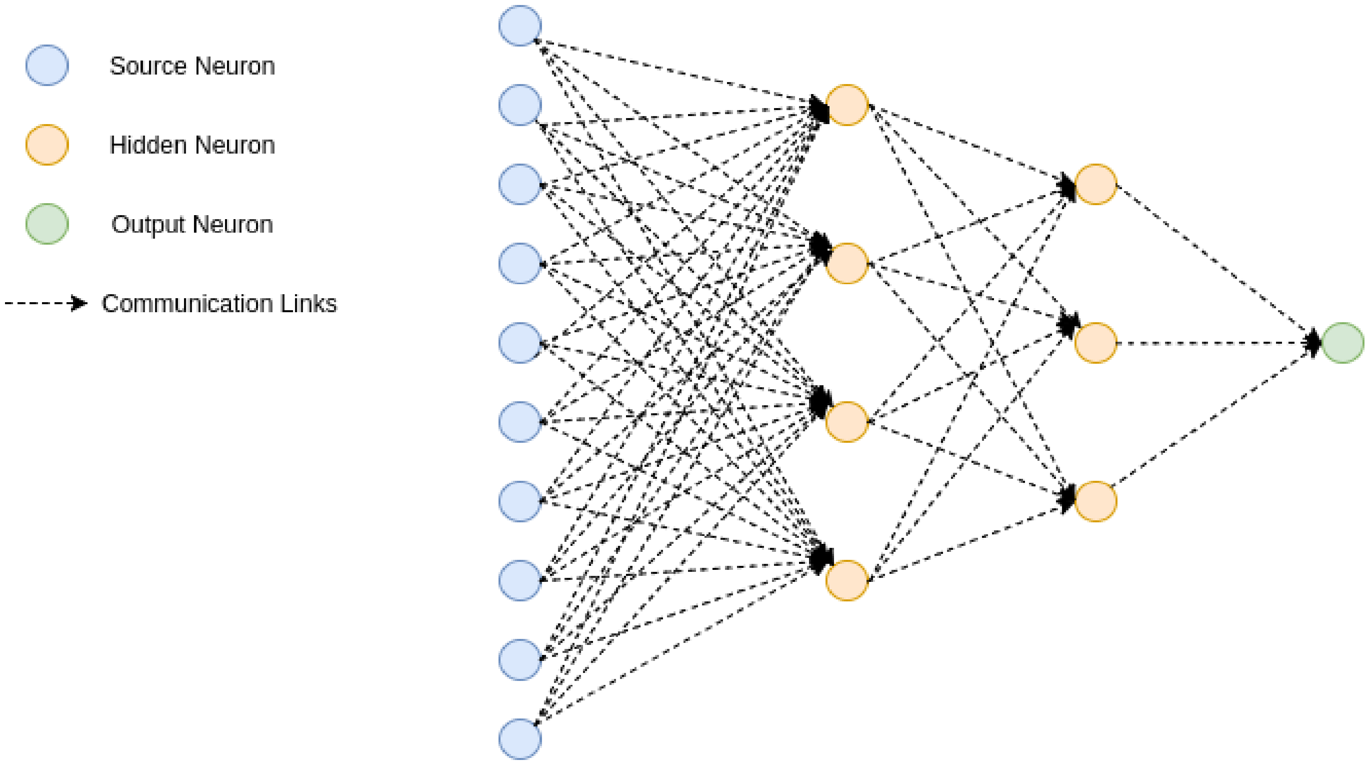
(a.)The given graph is similar to the representation of piecewise linear function. Hence we can formulate $\varphi(v)$ as:

$$\varphi(v) = \begin{cases} b & if v \geq 0.5a \\ v + 0.5b & if -0.5a < v < 0.5a \\ 0 & v \leq 0.5a \end{cases} \tag{1}$$

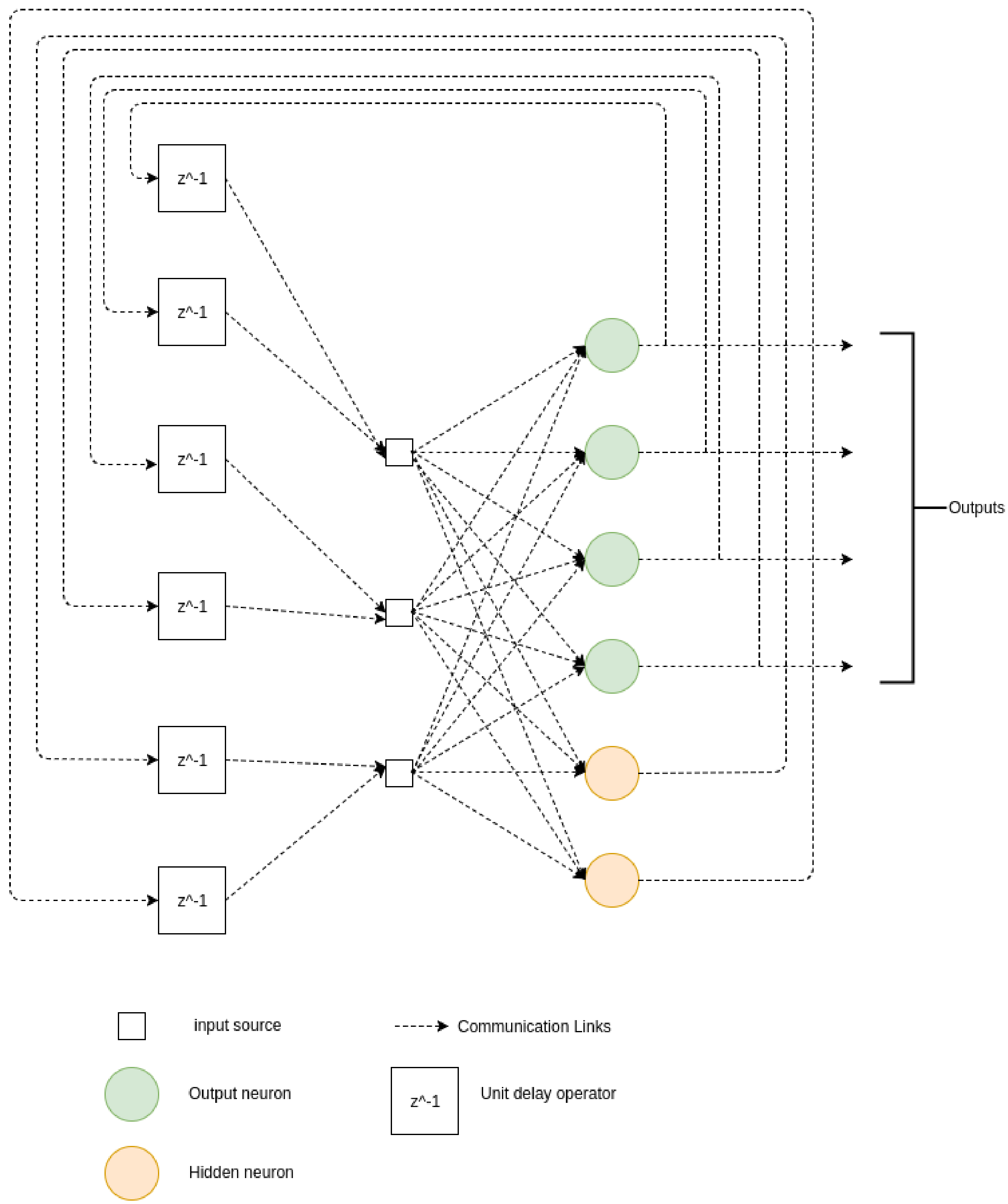
(b.) As a approaches zero, v approaches zero and the value of $\varphi(v)$ tends to 0.5b

3 Network architectures

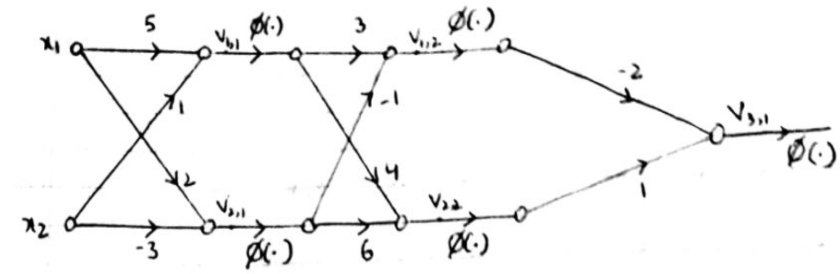
3.1 Exercise 1.12



3.2 Exercise 1.19



3.3 Exercise 1.13 (a)



$V_{1,1}$ and $V_{2,1}$ can be computed as:

$$V_{1,1} = \phi(5x_1 + x_2)$$

$$V_{2,1} = \phi(2x_1 - 3x_2)$$

We consider x_1 and x_2 as inputs

Similarly, $V_{1,2}$ and $V_{2,2}$ can be computed as

$$V_{1,2} = \phi(3V_{1,1} - V_{2,1}) = \phi\big(3(\phi(5x_1+x_2)) - \phi(2x_1-3x_2)\big)$$

$$V_{2,2} = \phi(6V_{2,1} + 4V_{1,1}) = \phi\Big(6(\phi(2x_1-3x_2)) + 4(\phi(5x_1+x_2))\Big)$$

and $V_{3,1}$ can be computed as

$$V_{3,1} = \phi(-2V_{1,2} + V_{2,2})$$

$$\boxed{V_{3,1} = \phi\Big(-2\big(\phi(3(\phi(5x_1+x_2)) - \phi(2x_1-3x_2))\big) + \phi\big(6(\phi(2x_1-3x_2)) + 4(\phi(5x_1+x_2))\big)\Big)}$$

3.4 Exercise 1.13 (b)

We consider x_1 and x_2 as inputs, $v_{1,1}$ and $v_{2,1}$ are local fields of first layer, $v_{1,2}$ and $v_{2,2}$ are local fields of second layer and $v_{3,1}$ is output. If the output neuron operates in its linear region, then logistic activation function is replaced by linear activation function, and input-output mapping can be written as:

$v_{1,1} = \varphi(5x_1 + x_2) = 5x_1 + x_2$
$v_{2,1} = \varphi(2x_1 - 3x_2) = 2x_1 - 3x_2$
$v_{1,2} = \varphi(3v_{1,1} - v_{2,1}) = 13x_1 + 6x_2$
$v_{2,2} = \varphi(6v_{2,1} - 4v_{1,1}) = 32x_1 - 14x_2$
$v_{3,1} = \varphi(-2v_{1,2} + v_{2,2}) = 6x_1 - 26x_2$

4 Knowledge Representation:

4.1 Exercise 1.21

We can apply taylor series for very small values of a to get $S(\alpha, x)$,

$S(\alpha, x) = S(0, x) + \alpha\frac{\partial s(\alpha,x)}{\partial a}$

Since tangent vector V is defined by partial derivative $\frac{\partial s(\alpha,x)}{\partial a}$, we can substitute this in the above equation,
$S(\alpha, x) = S(0, x) + \alpha V$
$V = \frac{S(\alpha,x)-S(0,x)}{\alpha}$

The tangent vector will be locally invariant for very small α because the tangent distance becomes independent of α.