NN_LindaKoine_091017

October 14, 2017

```
In [3]: import numpy as np
        import sympy as sp
        from sympy.plotting import plot
        sp.init_printing()
```

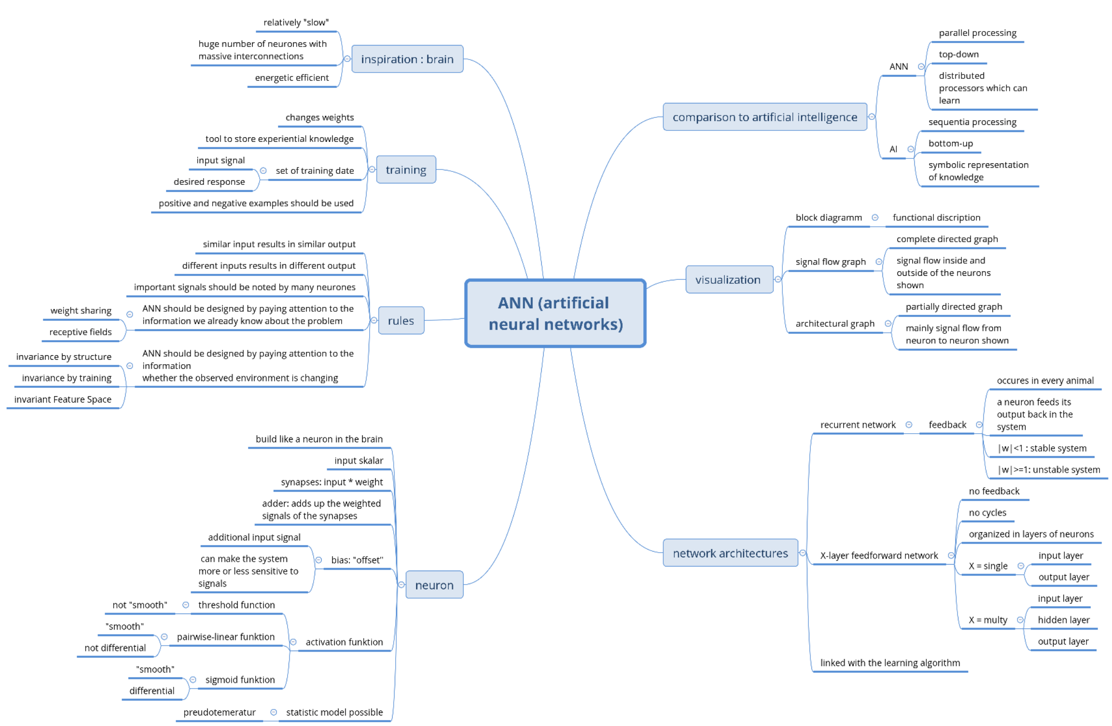
2017 WS - Neural Networks - Linda Koine

1 Assignment 1

1.1 mindmap:

```
In [20]: from IPython.display import Image
         Image("ANN.png")
```

Out[20]:



1.2 Models of a neuron

1.1)

```
In [4]: a,v= sp.symbols("a, v")
```

fi is given by:

```
In [5]: f1=1/(1+sp.exp(-a*v))
        f1
```

Out[5]:

$$\frac{1}{1+e^{-av}}$$

This is the derivation of $f1$.

```
In [6]: sp.simplify(sp.diff(f1,v))
```

Out[6]:

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

The dervation of $f1$ divided by $a \times f1$ should be $1 - f1$:

```
In [9]: sp.simplify(((sp.diff(f1,v))/(a*f1))-(1-f1))
```

Out[9]:

$$0$$

We see that the derivation of $f1$ is indeed given by $a \times f1(v)[1 - f1(v)]$.
The value of the derivation of the in the origin is :

```
In [8]: sp.diff(f1, v).subs(v, 0)
```

Out[8]:

$$\frac{a}{4}$$

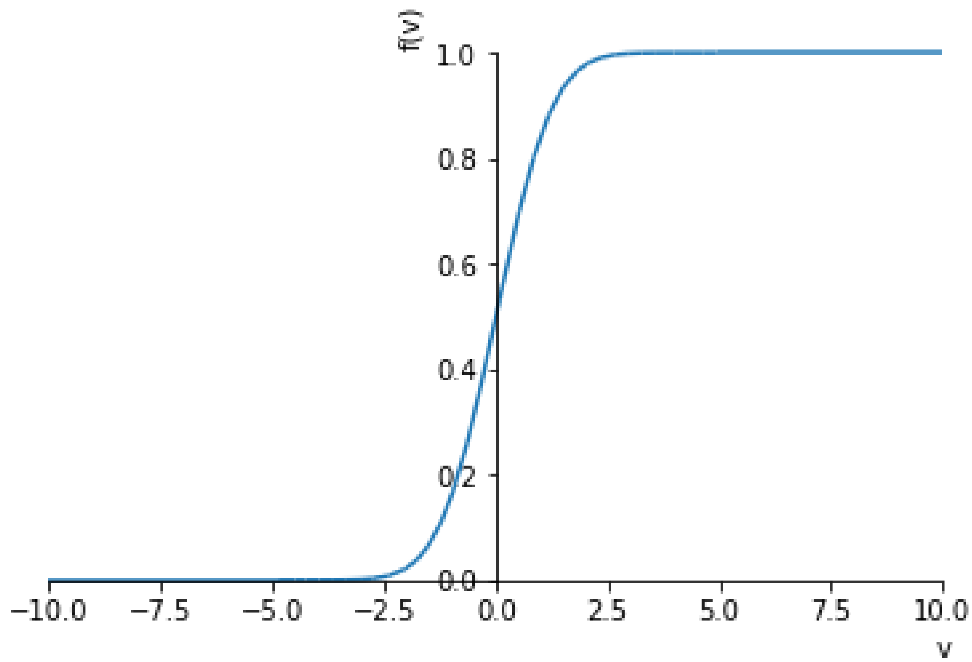
1.4)

```
In [10]: x,v= sp.symbols("x, v")
```

We define the funcion $f2$ given by Problem 1.4 (i) and plot it :

```
In [11]: f2=(1/( sp.sqrt(2*sp.pi)))*(sp.integrate(sp.exp(-(x**2/2)),(x, sp.oo *-1, v)))
```

```
In [12]: plot(f2)
```

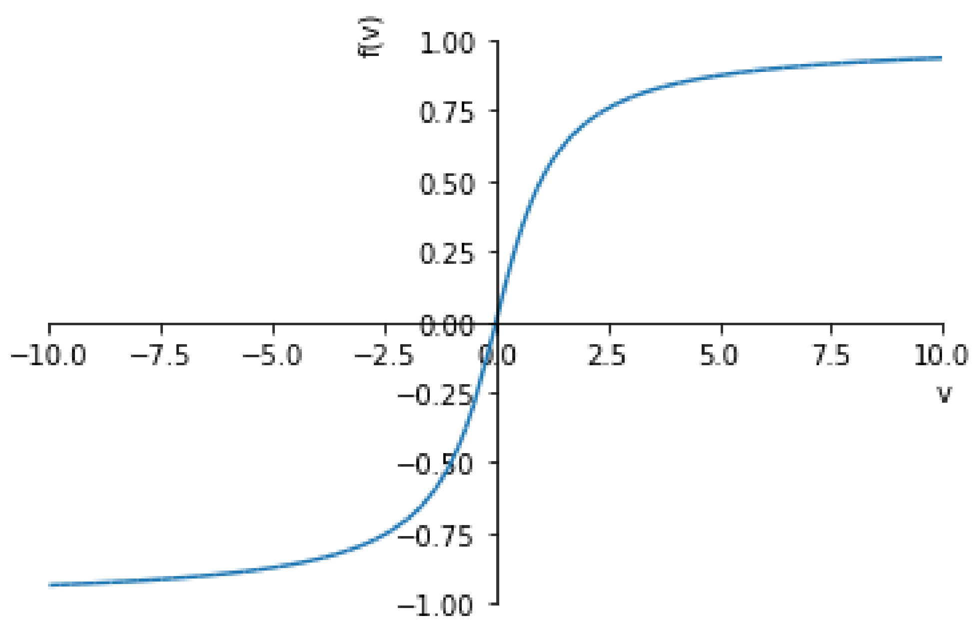


Out[12]: <sympy.plotting.plot.Plot at 0x7f0566ef8470>

We define the funcion $f3$ given by Problem 1.4 (ii) and plot it :

```
In [13]: f3=(2/sp.pi)*sp.atan(v)
```

```
In [14]: plot(f3)
```



Out[14]: <sympy.plotting.plot.Plot at 0x7f0566eb4780>

We clearly see that both functions fit the requirements of a sigmoid function. Both are differantial and have a smooth transition between their minimum and their maximum value.

difference: $f2$ defines a range from 0 to 1 and $f3$ defines a range from -1 to 1.

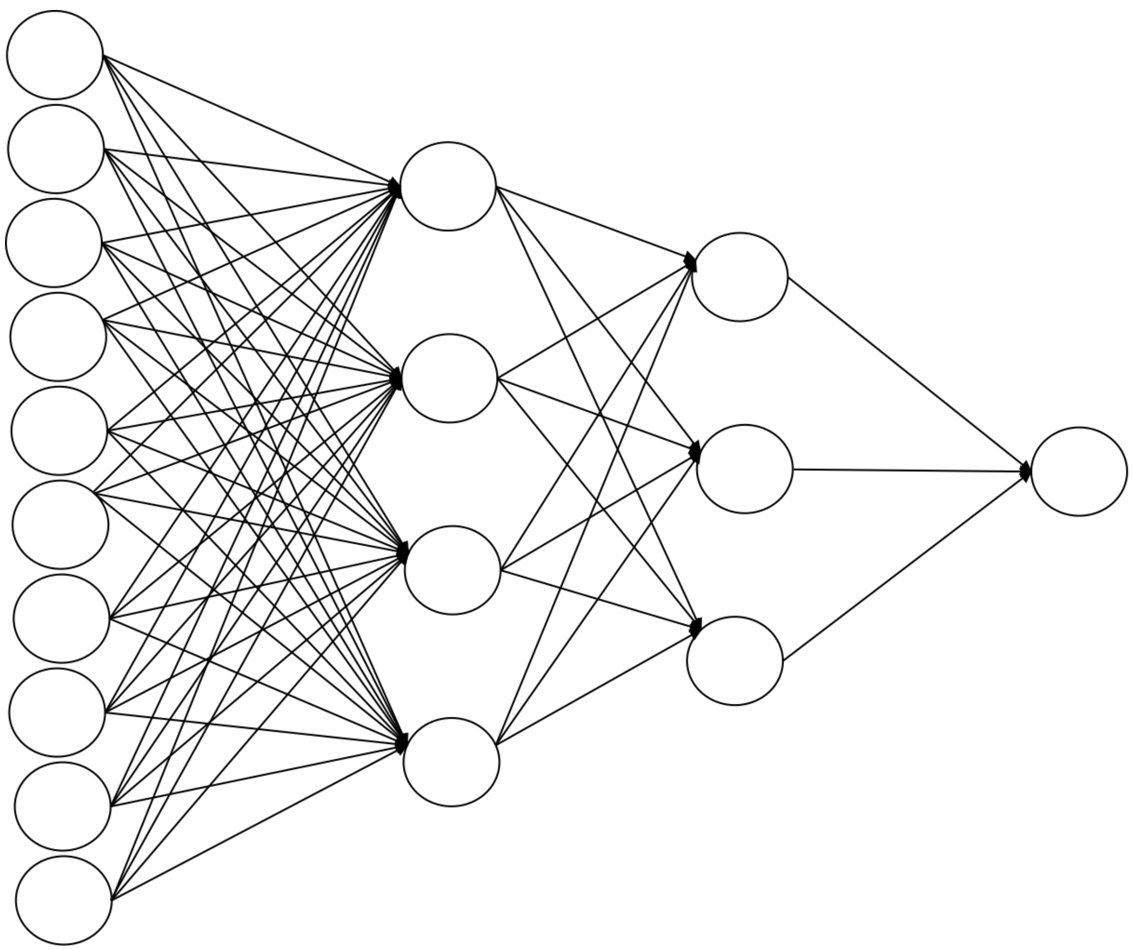
1.3 Network architectures

1.12)

A fully connected feedforward network with 10 source nodes, 2 hidden layers, the first with 4 and the second with 3 nodes and a single output node:

```
In [21]: Image("graph.png")
```

Out[21]:



1.13) (a)

Input of the source nodes : x and y

Unknown activation funktion : *phi*

```
In [22]: x,y= sp.symbols("x, y")
         phi = sp.Function("varphi")
```

```
In [23]: x1 = x * 5 + y
         y1 = -3 * y + 2 * x
         x2 = phi(x1)
         y2 = phi(y1)
         x3 = 3 * x2 - y2
         y3 = 6 * y2 + 4 * x2
         x4 = phi(x3)
         y4 = phi(y3)
```

The input-output mapping is given by this function :

```
In [24]: phi(-2 * x4 + y4)
```

Out[24]:

$$\varphi(-2\varphi(-\varphi(2x-3y)+3\varphi(5x+y))+\varphi(6\varphi(2x-3y)+4\varphi(5x+y)))$$

1.13) (b)

If the output neuron operates in a linear region, it has the same impact like a weight. We call it w.

```
In [25]: w= sp.symbols("w")
```

In this case input-output mapping is given by this function :

```
In [19]: w*(-2* x4+y4)
```

Out[19]:

$$w\left(-2\varphi(-\varphi(2x-3y)+3\varphi(5x+y))+\varphi(6\varphi(2x-3y)+4\varphi(5x+y))\right)$$