Lecture 1 Introduction → Neural Network

Assessment

- ① 3 assignment for Part 1
  2 mini-project for Part 2
- ② Final Exam

Langrange ⇒ MATLAB

Module { Theory, Application }

Pattern Recognition ⇒ Human > Computer
(Animal is also true)
⇓
Pigeon Experiments
⇓
artist work → not just memorize
↓
pattern recognition
⇓
train → predict

☆ **Compensation!**

Human Brain ⇒ { highly complex, non-linear, parallel } ⇒ Neural Network

neurons excited ⇒ higher firing rate

{ inhibited, excited, modulated }

learning ⇒ create synapse! → connection

presynaptic cell

postsynaptic cells

Nucleus

neuron $\begin{cases} \text{cell body} \Rightarrow \text{action potential} \\ \text{axon} \to \text{synapse} \to \boxed{\text{special gates}} \\ \text{dendrites} \end{cases}$

a hundred billion

Neuron $\leftrightarrow$ Approximation by mathematical model

$\downarrow$

start from the simplest one

① input $\to$ $\boxed{\text{Neuron}}$ $\to$ output   Qualitative Model

② the simplest one $\Rightarrow$ output $\boxed{y = \sum_i x_i}$ $\leftarrow$ input

$\boxed{\text{Question}}$: ① Neuron is always on fire!

② y is huge!

$\Downarrow$

step f^2 $y = \varphi(\sum x_i - b)$ $\to$ can only model $\boxed{\text{excitment}}$

$\Downarrow$

$\boxed{\text{coefficient} = 1}$

$\Downarrow$ we want inhibituet

$$\boxed{y = \phi(\sum w_i x_i - b)}$$ ✓

w_i: synaptic weights $\begin{cases} \text{excitement} \\ \text{inhibitment} \end{cases}$

① synapse $\leftrightarrow$ weight $\{w_i\}$ $\Rightarrow$ come by learning

② adder $\leftrightarrow$ input

③ activation ⟷ output

⇒ for neuron k,

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \qquad y_k = \varphi(\overbrace{u_k + b_k}^{v_k})$$

bias

⇓

induced local field ⇒ $\boxed{v_k = u_k + b_k}$ → background potential

potential induced by other neurons

Simplify the notation

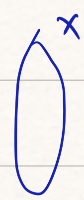
$$v_k = \sum_{j=0}^{m} w_{kj} x_j \qquad y_k = \phi(v_k)$$

$$\begin{cases} x_0 = +1 \\ w_{k0} = b_k \ \text{(bias)} \end{cases}$$

Activation function

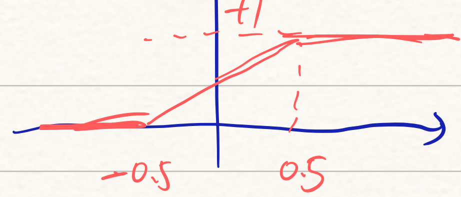
① Squash function (hard limiter)

$$\phi(v) = \begin{cases} +1, & v \geq 0 \\ 0, & v < 0 \end{cases}$$ → model fire or not fire



→ v_k → y_k

② piece-wise linear f^n



③ Sigmoid f^{n}

if we want output $\in [-1,1]$

$\Downarrow$

$$\boxed{\phi(v) = \tanh(v)}$$

No meaning

$\Downarrow$

engineering tools

$$\phi(v) = \frac{1}{1+e^{-av}}$$

$$\boxed{a \uparrow \longrightarrow \text{step } f^{n}}$$

$$\phi'(v) = \frac{a e^{av}}{(1+e^{-av})^2}$$

$$= a\,(1-\phi(v))\,\phi(v)$$

$$\boxed{\phi'(0) = \frac{a}{4}}$$

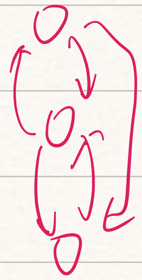
④ Gauss dist^{n} $\Rightarrow$ Radial Basis NN

$\boxed{\text{分类}}$

Layered Feedforward Network
- single-layer
- multi-layer

Note: if $\phi(\cdot)$ is linear, multi-layer $\Leftrightarrow$ single-layer

perceptron $\leftrightarrow$ single layer neural neural networks



recurrent network → content-addressable → recognize sently by past information

recurrent network → store memory

MLP → approximate any $f \in \mathbb{C}[a,b]$

Part 1

1. NN Introduction
2. Perceptron ch 1 & 2 & 3
3. NN ch 4
4. Radial-Basis ch 5
5. Self-Organizing ch 9

Part 2

1. SVM
2. RL

可以看往年卷子