DSA5105 LEC5

Recap (Hypothesis Space $\mathcal{H}$)

① Linear Basic Function

$$f(x) = \sum_j a_j \phi_j(x)$$ → fixed before training

② DT

$$f(x) = \sum_j a_j \mathbb{1}_{R_j}(x)$$ → learning parameters

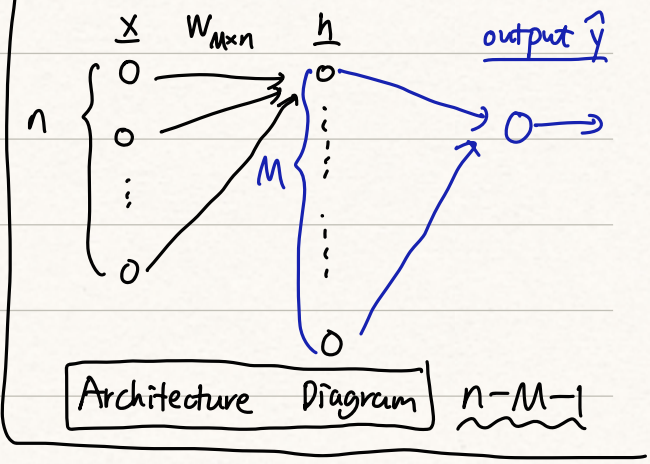
---

Neural Network

$$f(x) = \sum_{j=1}^{M} v_j \, \delta(w_j^T x + b_j)$$

j-th hidden unit

only one hidden layer → shallow



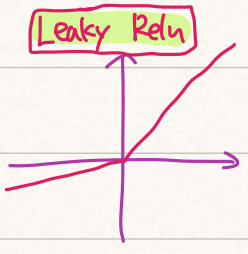
Activation function

① sigmoid $\in (0,1)$ $\quad \sigma(z) = \frac{1}{1+\exp(-z)}$

☆ ② tanh $\in (-1,1)$ $\quad \tanh(z)$

☆ ③ ReLU $\in [0,\infty)$ $\quad \text{Relu}(z) = \max(0,z)$

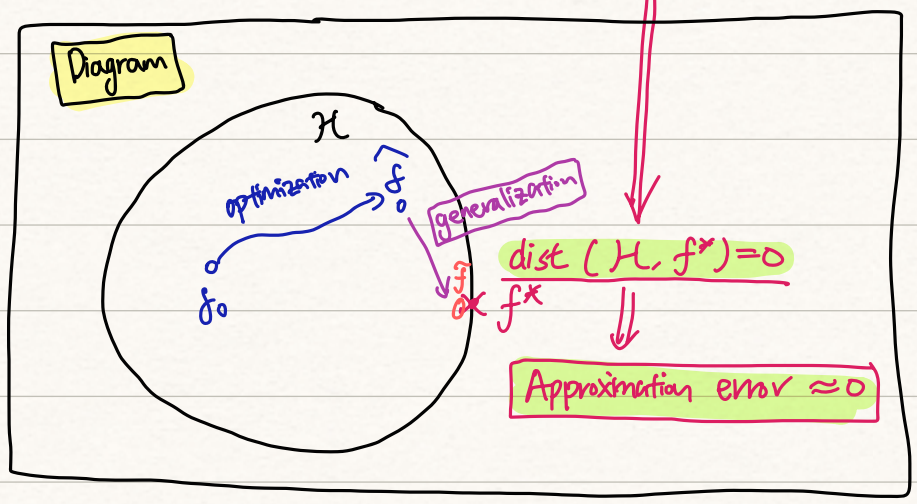
④ Leaky Relu $\quad \text{Leaky Relu}(z) = \begin{cases} z & z \geq 0 \\ \delta z & z < 0 \end{cases}$



Universal Approximation Thm (guarantee) ⇒ NN is an universal approximator

NN can be viewed as a function approximator (any continuous function)

☆☆ the hypothesis space for NN is LARGE



Wonderful Part

Question : Will NN suffer from Curse Of Dimensionality ?

linear & nonlinear feature map

E.g. $u = (3, 1, 2) = 3e_1 + e_2 + 2e_3$

Basis are fixed

① what is the optimal approximator of $\hat{u} = a_1 e_1 + a_2 e_2$ (of the form)

Linear Basis → formulize $\hat{u} = \min \|\hat{u} - u\|_2^2$

$\Rightarrow \hat{u} = 3e_1 + e_2$

& $\text{Error}(\hat{u}, u) = \|\hat{u} - u\|_2 = 2$

② what is the best 2 basis? → we can arbitrarily choose 2 basis

Neural Network Answer : $(e_1, e_3) \Rightarrow$ Error = 1

E.g. Assume in $\mathbb{R}^{100}$, you have a vector v of dim-3

↓ Function

$f^*(x) = \sum_{j=1}^{M} v_j \phi_j(x)$

M-dim approximation

Basis

target in $\mathbb{R}^{\infty}$

Recap

Will NN suffer from Curse Of Dimensionality ?

Answer: NO

↓ [Baron, 1993]

the reason is intuitively from previous example. $f^*: [0,1]^d \to \mathbb{R}$

we can choose M- basis (hidden neurons), s.t

$\|f^* - f_M\|^2 \leq O(M^{-1})$ → Bound is not related to d!!!

↓

overcome the curse of dim!

↓

we have the freedom to choose feature map (Basis)

[Comparison] for linear basis function

$\|f^* - f_M\|^2 \geq O(M^{-\frac{\alpha}{d}})$ → Curse of Dimensionality!

[Optimization]

→ Universal Approximation Thm ⟹ there is a good approximator of f^* (Oracle) in $\mathcal{H}$

① Hypothesis Space $\mathcal{H} = \{f: f(x) = f_\theta(x); \theta \in \Theta\}$

② ERM Framework $\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmin}} \hat{\Phi}(\theta)$ → empirical loss

unconstrained optimization $= \underset{\theta \in \Theta}{\operatorname{argmin}} \frac{1}{N} \sum_{n=1}^{N} L(f_\theta(x_i), y_i)$

(too much non-linearity)

③ Gradient Descent (SGD)

a) Necessary Condition : $\nabla_\theta \bar{\Phi}(\theta)\big|_{\theta=\hat{\theta}} = 0$

→ Non-linear equation → No-closed form (not solvable)

b) Iterative Method

1) Gradient Descent (GD)

update: $\theta_{k+1} = \theta_k - \alpha_k \nabla_\theta \bar{\Phi}(\theta)\big|_{\theta=\theta_k}$ $(\alpha_k > 0)$

(to achieve $\min_\theta \bar{\Phi}(\theta)$)

Approximation Model → Taylor Approximation Order-1

$\bar{\Phi}(\theta+\Delta\theta) = \bar{\Phi}(\theta) + \nabla_\theta \bar{\Phi}(\theta)^T \Delta\theta + O(\|\Delta\theta\|^2)$

→ a choice of $\Delta\theta_{GD} = -\nabla_\theta \bar{\Phi}(\theta)$

can only guarantee $\|\nabla \bar{\Phi}_\theta(\theta)\big|_{\theta=\theta_k}\| \to 0$

→ Stationary point / local minima → depend on the landscape of $\bar{\Phi}(\cdot)$

we cannot have $\theta_k \to \theta^*$ as $k \to \infty$

→ global minima

(Special Case)

→ for CONVEX f, global minima $\Longleftrightarrow$ local minima

$\Longleftrightarrow$ $\nabla_\theta \bar{\Phi}(\hat{\theta}) = 0$

(or sub-gradient $0 \in \partial \bar{\Phi}(\hat{\theta})$)

rigorously speaking

☆ GD can guarantee converge to global minima

[E.g.]

① linear basis model

$$y = \Phi w$$

$$\rightarrow \underline{E(w) = (y - \Phi w)^T (y - \Phi w)} \quad \text{(convex)}$$

$$= w^T \Phi^T \Phi w - 2 y^T \Phi w + \text{const}$$

since $\nabla E(w) = 2\Phi^T \Phi w - 2\Phi^T y$

$$\nabla^2 E(w) = 2\Phi^T \Phi \longrightarrow (PSD)$$

↓ implies

$E(w)$ is convex

② SVM → convex

③ NN → highly non-convex

2) SGD → works very well → we may lose the advantages of convergence

But we can be close to the place we want

$$E(\theta) = \frac{1}{N} \sum_{n=1}^{N} \text{Loss}\left(f_\theta(x_n), y_n\right)$$

$$\nabla_\theta E(\theta) = \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta \text{Loss}\left(f_\theta(x_n), y_n\right) \quad (*)$$

① when N is large, $(*)$ is computationally expensive

② use $\frac{1}{|B|} \sum_{n \in B} \nabla_\theta \text{Loss}\left(f_\theta(x_n), y_n\right)$ sample B randomly

to treat as one APPROXIMATOR of $\nabla_\theta E(\theta)$

$|B|$ → Batch Size → small

a noise approximator

☆ help to go out of local minimum

Deep NN → more hidden layers

$\mathbb{R}^d$ x (input) → $\mathbb{R}^{d_1}$ h_1 → $\mathbb{R}^{d_2}$ h_2 → $\mathbb{R}^{d_3}$ h_3 → $\mathbb{R}^K$ y (output)

hidden layers (3)

How to calculate gradient ?

→ through BP algorithm (chain rule)