Shalev (2014)

Understanding ML: from theory to Algo

20.1 Feed forward Neural networks

nodes = neurons, edges = output → input

each neuron is modelled as a simple scalar func $\sigma: R \to R$

3 main functions:

Sign , threshold $\sigma(a) = 1_{[a>0]}$, Sigmoid $1/(1+\exp(-a))$

- σ = activation function of the neuron
- each edge in the graph links the output of some neuron as the input of another neuron

the input of a neuron is the weight sum of the outputs connected to it — w

network = formed into layers

- Bottom layer = V_0 = input layer. Contains $n+1$ neurons where n is dimensionality of the input space
- $+1$ due to constant bias value

Suppose we have calculated the output of neurons @ lay t the output @ $t+1$:

r = neuron

Fix $v_{t+1} \in V_{t+1}$ Let $a_{t+1,j}(x)$ = input into $v_{t+1,j}$ when the network is fed w/ input vector x

r = index

$$a_{t+1,j}(x) = \sum_r w((v_t, r, v_{t+1}, j)) o_{t,r}(x)$$

$$o_{t+1,j}(x) = \sigma(a_{t+1,j}(x))$$

the input to $v_{t+1,j}$ is the weighted sum of the outputs of the neurons in V_t that are connected to $v_{t+1,j}$

and the output of $v_{t+1,j}$ is simply the activation function applied to the input

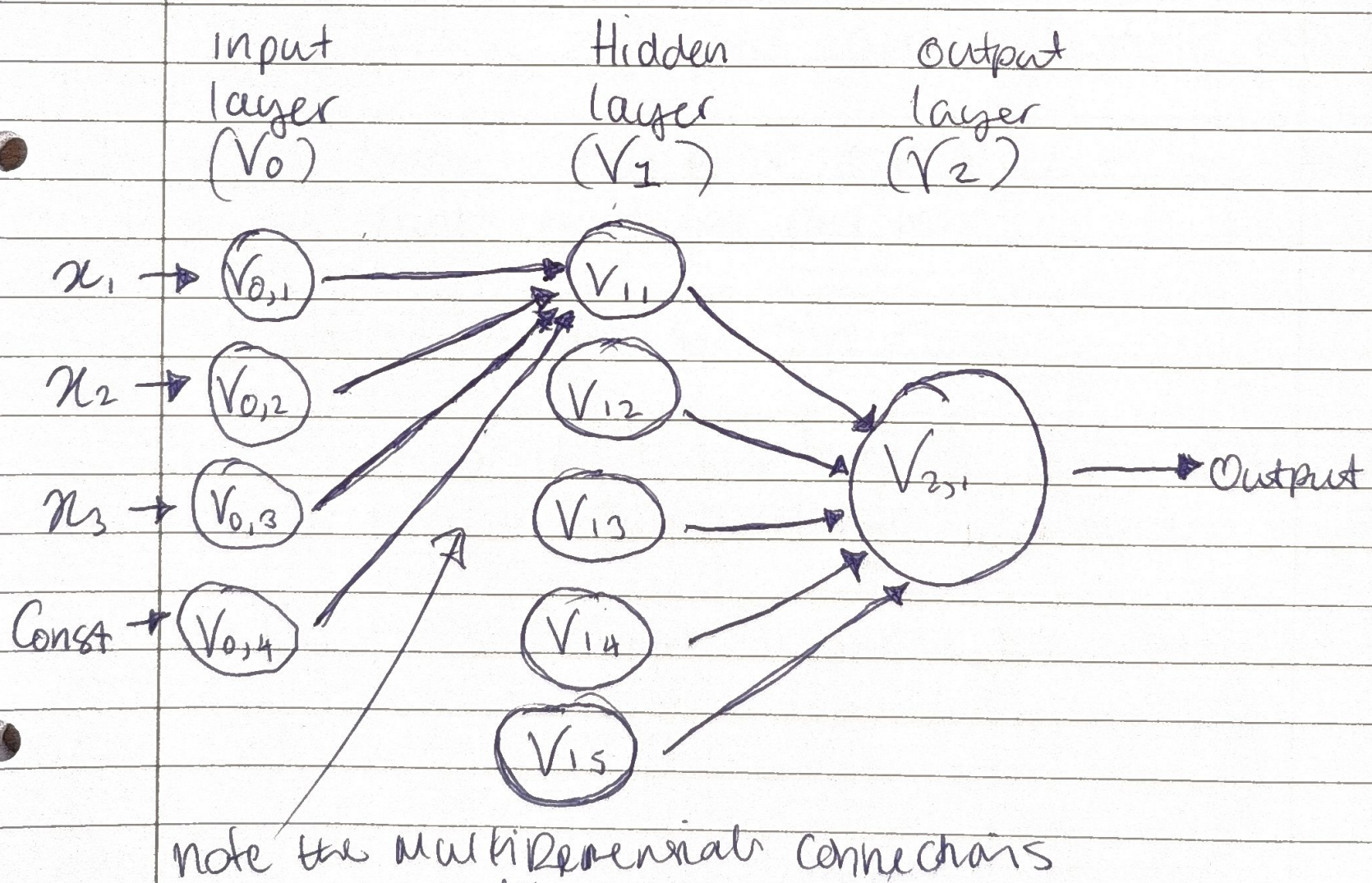
layers between V_0 = input & V_t = output are known as <u>hidden layers</u> $V_1, \ldots, V_{T-1}$

T = number of layers, excluding V_0 — or "depth"

the "size" of network = $|V|$

the "width" of the network = $\max_t |V_t|$

Example: depth = 2, size = 10, width = 5



20.2 learning Neural Networks

$NN = (V, E, \sigma, w)$ = neurons, Edges, activs, weights

to obtain function: $h_{V,E,\sigma,w} : R^{|V_0|-1} \to R^{|V_t|}$

Any set of functions can define a hypothesis class

typical is to fix the graph (V,E) & activ func σ

(V,E,σ) = architecture of the network

hypothesis class:

$$H_{VE\sigma} = \{h_{V,E,\sigma,w} : w \text{ maps } E \to R\}$$

the paras that set the hypo are the weights of the Edges

We for hypo classes we can study the:
approx error, estimation error and optimiz error

20.3 Study approx error for $H_{V,E,\sigma}$ by study types funcs the hypo can take

20.4 est error for Bin class

20.5 Show it is comp hard to learn the class even is graph/network = small

20.6 most common heuristic for training

20.3 Power of NN's

Functions that can be implements in NNs

Fix V, E, σ

~~Why~~ what hypoth funcs can be implementation

Skipped

funcs allow to model non-linear patterns

Goal = Proof on of these funcs

VC Dimensions

- Measures the complexity or capacity of hypo class
- how expressive a set of functions are
- how well it can fit diff patterns in data

VC Dimensions of a hypo class =
is the size of the largest set of points that a hypo class can "Shatter"

"Shatter" = func exist that perfectly classifies

Any
lin classifs can shatter 3 but not 4 any in = 2 Dims

Directs related to sample

- VC Dims = ~~complexity of learning~~ complexity of learning
- Sample complex = num of data points needed to learn a good hypo w/ high Prob

High VC = More data to avoid overfit

Generalization & VC:

A hypo class w/ high VC dims = can not fit train data well but overfit & perform poorly on unseen data

Sample complex of learning a NN depends on its VC dimension

20.4 Sample Complex of NNs

Skipped

20.5 Runtime of learning NNs

Skipped

20.6 SGD & Backpropagation

A more complicated & less visual explanation than Zhang (2021)

but explains the process in terms of vectors/matrix instead of simply down to individual pairs