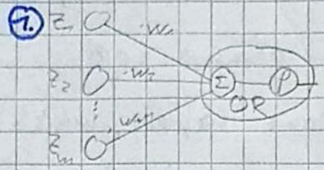
2 Classification Capacity

2.1 Simple Networks

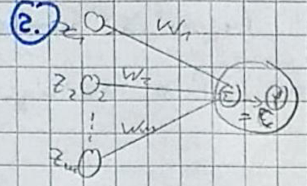
1.



$w_i = 1 \ \forall i \in \{1,\dots,m\}$, $f(\vec z) = \begin{cases} 0 & \text{if } \vec z \le 0 \\ 1 & \text{else} \end{cases}$

If one of the z_i is >0 the sum of all z_i is ~~bigger~~ >0.

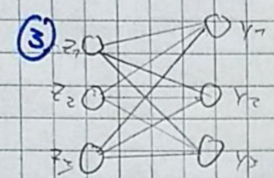
2.



$w_i = 10^{i-1} \ \forall i \in \{1,\dots,n\}$, $f(\vec z) = \begin{cases} 0 & \text{if } \vec z - c \cdot w \neq 0 \\ 1 & \text{else} \end{cases}$

encode z as a decimal number and compare with c in same representation

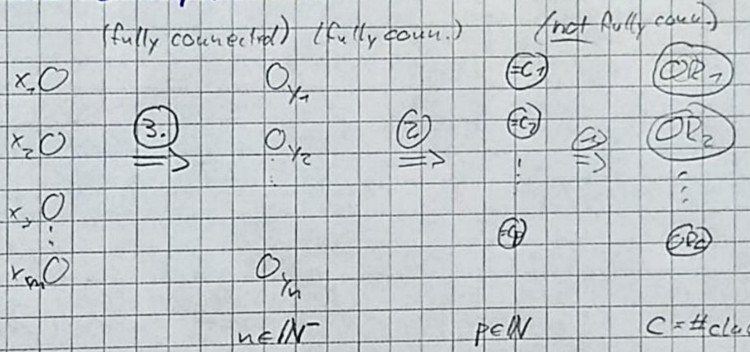
3.



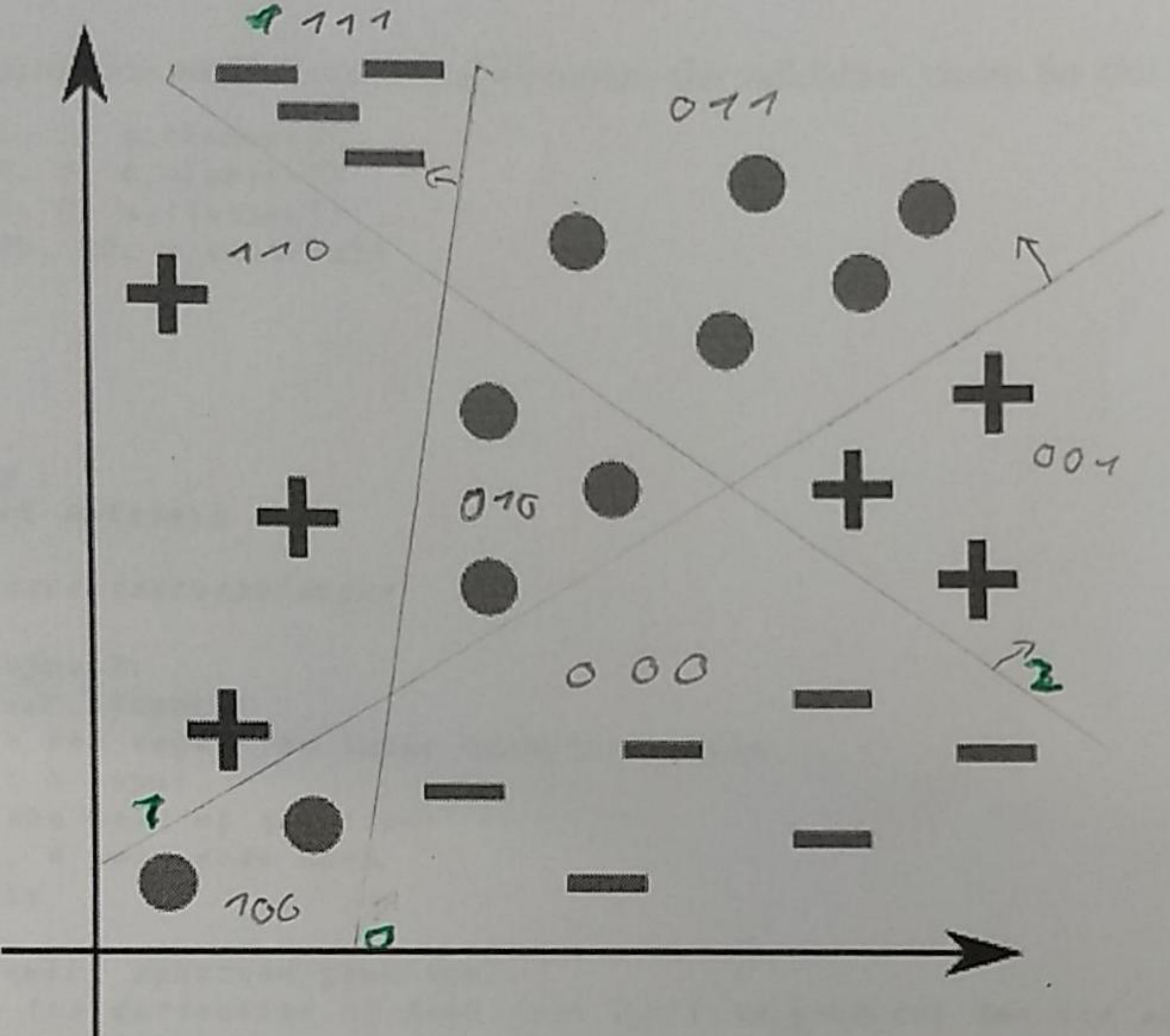
each ~~ne~~ y-neuron has weight vector $y_i \sim B_i$

$y_i = \text{step}(B_i \cdot z) = \begin{cases} 1 & B_i x > 1 \\ 0 & \text{else} \end{cases}$

2.2 3-layer Universal Classifier



First the data is split, such that each input vector is projected on a corner of a hyper-cube (dim n), where in one corner ~~each~~ are only members of one class. the c_i ~~encode~~ ~~the~~ layer reads of the bit representation such that on c_i is ~~ex~~ always one and all c_j $j \neq i$ are 0. Since each corner only contains one class, but the members of one class can be projected onto different corners, we need the OR-layer, which will show the classification of the instances. Each OR neuron represents one class and is only connected to the neurons of the previous layer, which represent a corner (of the hypercube) of that class. This way if the first layer is trained there will be perfect classification of a training set



hyper-plains a
boundaries

index

3. Linear Activation Function

$$Z_L = \phi_L \left(B_L \cdot \tilde{Z}_{L-1} \right)$$
$$Z_L = \phi_L \left(B_L \cdot \phi_{L-1} \left(B_{L-1} \cdot \tilde{Z}_{L-2} \right) \right)$$

Let $f_l : \mathbb{R}^{H_{l-1}} \to \mathbb{R}^{H_l}$ be the function that calculates the pre-activations: $Z_{l-1} \mapsto \tilde{Z}_l = B_l Z_{l-1}$ which is obviously a linear function. So the total function reads

$$Z_L = \underbrace{\phi_L \circ f_L \circ \phi_{L-1} \circ f_{L-1} \circ \ldots \circ \phi_1 \circ f_1}_{=:\mathbf{F}}(Z_0)$$

As the composition between two linear functions is still a linear function, $\mathbf{F}$ is also a linear function which can be written as a composition of two arbitrary linear functions $\tilde{f}, \tilde{\phi}$ so that $\mathbf{F} = \tilde{\phi} \circ \tilde{f}$.

$$\Rightarrow Z_L = \tilde{\phi} \circ \tilde{f}(Z_0)$$

That means that a neural network with a linear activation function is equivalent to a 1-layer neural network.