

## 2 Konzeptlernen aus Beispielen und Gegenbeispielen

Ein Konzept  $C \subset \Omega$  soll gelernt werden. Dabei ist eine Konzeptklasse  $\mathcal{C} \subset \mathcal{P}(\Omega)$  gegeben und  $C \in \mathcal{C}$  vorausgesetzt.

Die Algorithmen sollen  $C$  finden oder approximieren innerhalb einer Hypothesenklasse  $\mathcal{H} \subset \mathcal{P}(\Omega)$ . I.a. gilt  $\mathcal{H} \neq \mathcal{C}$

### Beispiele

1. Im einleitenden Bsp.:  $C$  ist das Gebiet in der Ebene, in dem die Punkte als blau zu klassifizieren sind. Bei 1da besteht  $\mathcal{H}$  aus allen (Hyper-) Halbebenen.
2. näher an der theoretischen Informatik:  $\mathcal{C}$  kann die Menge aller aussagenlogischen Formeln über einer Menge von Variablen z.B. in DNF sein. Gegenbeispiele sind dann Belegungen der Variablen mit Wahrheitswerten für die der jeweiligen Hypothese widersprechen.

## 2.1 Online-Algorithmen

Ein Schüler soll ein ihm unbekanntes Konzept  $C \in \mathcal{C}$  finden, indem er Hypothesen aus  $\mathcal{H}$  vom Lehrer durch Gegenbeispiele widerlegen lässt.

Ablauf:

Bestimme  $H \in \mathcal{H}$

Solange  $H \neq C$ :

fordere Gegenbeispiel  $x \in (C \cup H) \setminus (C \cap H)$

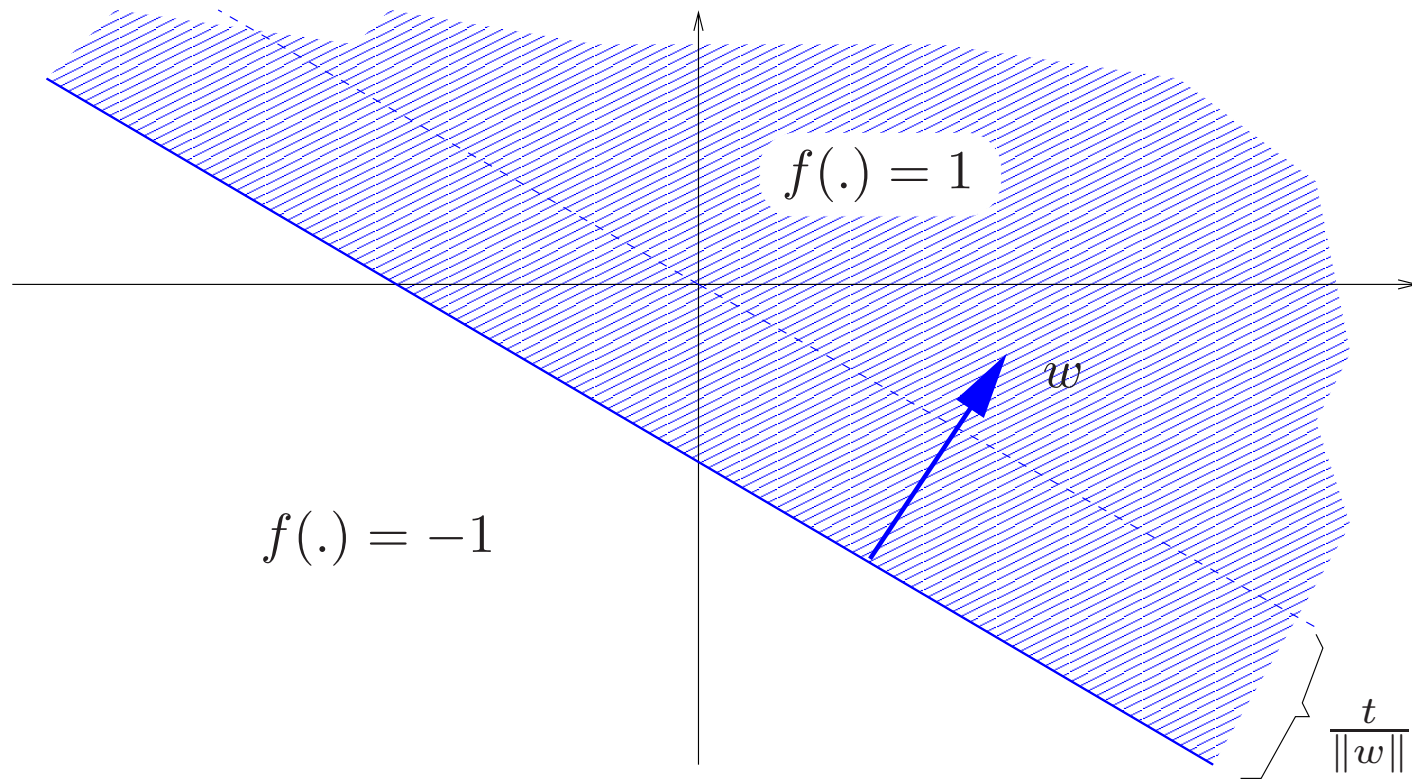
bestimme neue Hypothese  $H \in \mathcal{H}$

$f : \mathbb{R}^n \rightarrow \{-1, 1\}$  heißt **Thresholdfunktion** falls gilt:

$$\exists w \in \mathbb{R}^n, t \in \mathbb{R} ( f(x) = 1 \Leftrightarrow \langle w, x \rangle + t \geq 0 )$$

Dabei ist  $\langle w, x \rangle = \sum_{i=1}^n w_i x_i$  das Skalarprodukt.

$f$  heißt monotone Thresholdfunktion falls alle  $w_i \geq 0$  sind.



## 2.1.1 Perzeptron-Algorithmus

gegeben: endliche Menge von Lerndaten  $S \subset \mathbb{R}^n \times \{-1, 1\}$  mit  $\|x\| \leq R$  für  $(x, b) \in S$ , Lernparameter  $\eta > 0$ .

$$i := 0$$

$$w^{(0)} := (0, 0, \dots, 0)$$

$$t := 0$$

solange noch zu lernen ist:

fordere Gegenbeispiel  $(x^{(i)}, b_i)$  an

$$w^{(i+1)} := w^{(i)} + \eta \cdot b_i \cdot x^{(i)}$$

$$t_{i+1} := t_i + \eta \cdot b_i \cdot R^2$$

$$i := i + 1$$

zu zeigen: der Perzeptron-Algorithmus terminiert für linear trennbare  $S$ .

$$\text{Margin}(S, w, t) := \min\{b \cdot (\langle w, x \rangle + t) \mid (x, b) \in S\}$$

$\frac{\text{Margin}(S, w, t)}{\|w\|^2}$  ist der Abstand von  $S$  zur trennenden Hyperebene.

Komplexität von  $S$ :

$$K(S) := \sup \left\{ \frac{\text{Margin}(S, w, t)}{\|w\|^2} \mid \forall (x, b) \in S : (b = 1 \Leftrightarrow (w, x) + t \geq 0) \right\}$$

**Satz 1** Falls  $S$  linear getrennt werden kann, so lernt dies der Perzeptron-Algorithmus nach

$$\left( \frac{2R}{K(S)} \right)^2 \quad \text{Gegenbeispielen.}$$

**Lemma 1** *Mit der Notation  $u := (w, t/R) := (w_1, \dots, w_n, t/R)$  und  $y := (x, R) := (x_1, \dots, x_n, R)$  gilt*

$$u^{(i)} = u^{(i-1)} + \eta \cdot b_{i-1} \cdot y^{(i-1)} \quad \text{und} \quad b_{i-1} \cdot \langle u^{(i-1)}, y^{(i-1)} \rangle < 0.$$

**Beweis**

$$\begin{aligned} u^{(i)} &= (w^{(i)}, t_i/R) = (w^{(i-1)}, t_{i-1}/R) + \eta \cdot b_{i-1} \cdot (x^{(i-1)}, R) \\ &= u^{(i-1)} + \eta \cdot b_{i-1} \cdot y^{(i-1)} \end{aligned}$$

und da  $x^{(i-1)}$  Gegenbeispiel ist, gilt:

$$b_{i-1} \langle u^{(i-1)}, y^{(i-1)} \rangle = b_{i-1} \cdot \left( \langle w^{(i-1)}, x^{(i-1)} \rangle + t_{i-1} \right) < 0$$

□

**Lemma 2** *Es gelte  $\text{Margin}(S, w^*, t^*) \geq \gamma$  und  $\|w^*\| = 1$ . Dann folgt  $\langle u^{(i)}, u^* \rangle \geq i \cdot \eta \cdot \gamma$  nach jedem Schleifendurchlauf.*

**Beweis** Induktiv über  $i$  folgt die Behauptung aus

$$\langle u^{(i)}, u^* \rangle = \langle u^{(i-1)}, u^* \rangle + \eta \cdot b_{i-1} \cdot \langle y^{(i-1)}, u^* \rangle \geq \langle u^{(i-1)}, u^* \rangle + \eta \cdot \gamma.$$

Beachte dabei  $b_{i-1} \cdot \langle y^{(i-1)}, u^* \rangle \geq \text{Margin}(S, w^*, t^*) \geq \gamma$ .

Induktionsverankerung: wegen  $u^{(1)} = \eta \cdot b_0 \cdot (x^{(0)}, R)$  folgt

$$\langle u^{(1)}, u^* \rangle = \langle \eta b_0 (x^{(0)}, R), (w^*, t^* / R) \rangle = \eta b_0 (\langle x^{(0)}, w^* \rangle + t^*) \geq \eta \gamma$$

□

### Lemma 3

$$\|u^{(i)}\|^2 \leq 2i \cdot \eta^2 \cdot R^2$$

**Beweis** Induktiv mit:

$$\begin{aligned} \|u^{(i)}\|^2 &= \|u^{(i-1)}\|^2 + 2 \cdot \eta \cdot b_{i-1} \cdot \langle u^{(i-1)}, y^{(i-1)} \rangle + \eta^2 \cdot b_{i-1}^2 \cdot \|y^{(i-1)}\|^2 \\ &\leq \|u^{(i-1)}\|^2 + \eta^2 \cdot b_{i-1}^2 \cdot \|y^{(i-1)}\|^2 \\ &= \|u^{(i-1)}\|^2 + \eta^2 \cdot (\|x^{(i-1)}\|^2 + R^2) \\ &\leq \|u^{(i-1)}\|^2 + 2\eta^2 \cdot R^2 \end{aligned}$$

Verankerung:

$$\langle u^{(1)}, u^{(1)} \rangle = \langle \eta b_0(x^{(0)}, R), \eta b_0(x^{(0)}, R) \rangle = \eta^2 \cdot (\langle x^{(0)}, x^{(0)} \rangle + R^2) = 2\eta^2 R^2$$

□

**Beweis des Satzes** Aus Lemma 2 und 3 und der Cauchy-Schwarz-Ungleichung folgt:

$$i \cdot \eta \cdot \gamma \leq \langle u^{(i)}, u^* \rangle \leq \|u^{(i)}\| \cdot \|u^*\| \leq \sqrt{2i} \cdot \eta \cdot R \cdot \|u^*\|$$

Mit  $\|w^*\| = 1$  und  $|t^*| \leq R$  erhalten wir  $\|u^*\|^2 \leq 2$  und damit

$$i \cdot \eta \cdot \gamma \leq \|u^{(i)}\| \cdot \sqrt{2} \leq 2\sqrt{i} \cdot \eta \cdot R$$

solange die Schleife läuft.

Für  $i > (2R/\gamma)^2$  kann die Ungleichung nicht mehr gelten, also terminiert der Algorithmus bevor  $i$  diesen Wert erreicht.

□

## 2.1.2 Winnow-Algorithmus

Aussagenlogische Formeln mit  $n$  Variablen  $x_1, x_2, \dots, x_n$  können wir als Abbildungen  $m : \{0, 1\}^n \rightarrow \{0, 1\}$  auffassen.

Monome und Disjunktionen können durch Thresholdfunktionen ausgedrückt werden.

## Winnow-Algorithmus

- Initialisiere  $t := n/2$  und  $w_1 := w_2 := \dots := w_n := 1$ .
- Für Beispiele  $x$  mit  $m(x) = 1$  und  $\sum w_i x_i < t$ , verdopple  $w_i$  gdw  $x_i = 1$
- Für Beispiele  $x$  mit  $m(x) = 0$  und  $\sum w_i x_i \geq t$ , setze  $w_i = 0$  gdw  $x_i = 1$

**Lemma 4** *Winnow lernt eine monotone Disjunktion von  $k$  Literalen aus  $\{x_1, \dots, x_n\}$  nach höchstens  $O(k \log_2 n)$  Gegenbeispielen.*

**Lemma 5** *Bei Eingabelänge  $n$  können Monome mit  $k$  Literalen nach höchstens  $O(k \log_2 n)$  Gegenbeispielen gelernt werden.*

## Beweis von Lemma 4

- $w_i$  wird nur eliminiert, wenn  $x_i$  nicht im Zielkonzept vorkommt
- Für die Anzahl  $b_+$  der Verdopplungsschritte und die Anzahl  $b_-$  der Eliminationsschritte gilt  $n + t \cdot (b_+ - b_-) \geq 0$
- es gilt stets  $w_i \leq 2t$
- Nach  $b_+$  Verdopplungsschritten gibt es ein  $w_i$  mit  $\log_2 w_i \geq b_+/k$

Also gilt  $b_+/k \leq \log_2 w_i \leq \log_2 t + 1$  und es gibt höchstens  $k(\log_2 t + 1)$  Verdopplungsschritte.

Wegen  $b_- \leq n/t + b_+ \leq n/t + k(\log_2 t + 1)$  und  $t = n/2$  werden nur  $2 + k(\log_2(n/2) + 1)$  Gegenbeispiele benötigt. □

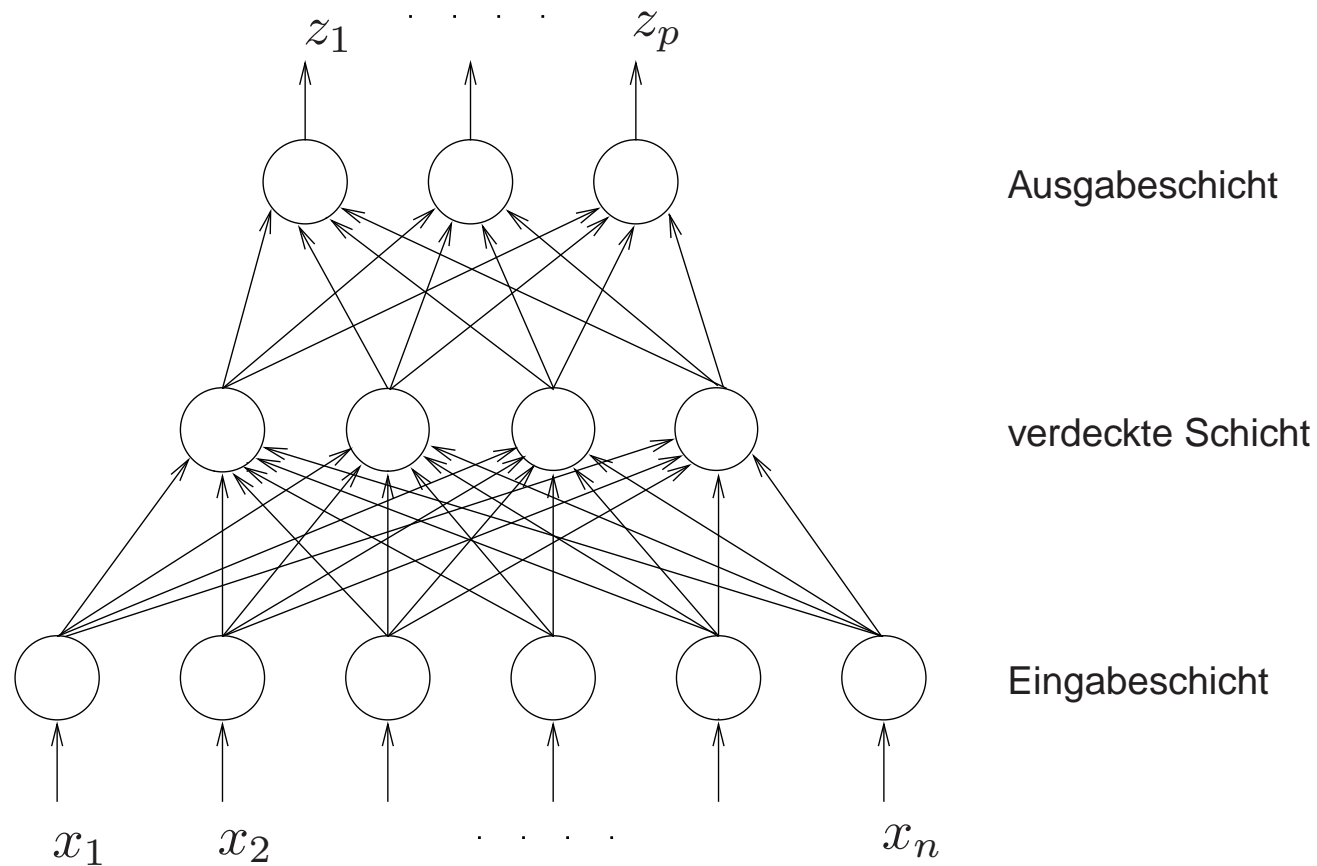
**Satz 2** *Bei Eingabelänge  $n$  können Formeln in disjunktiver Normalform mit höchstens  $k$  Literalen pro Monom nach höchstens  $O(sk \log_2 n)$  Gegenbeispielen gelernt werden. Die Zielformel lasse sich dabei als Disjunktion von  $s$  Monomen schreiben.*

**Beweis** Benutze statt der Eingabe  $x$  das Ergebnis von  $x$  auf allen möglichen Monomen mit  $k$  Literalen. Beachte

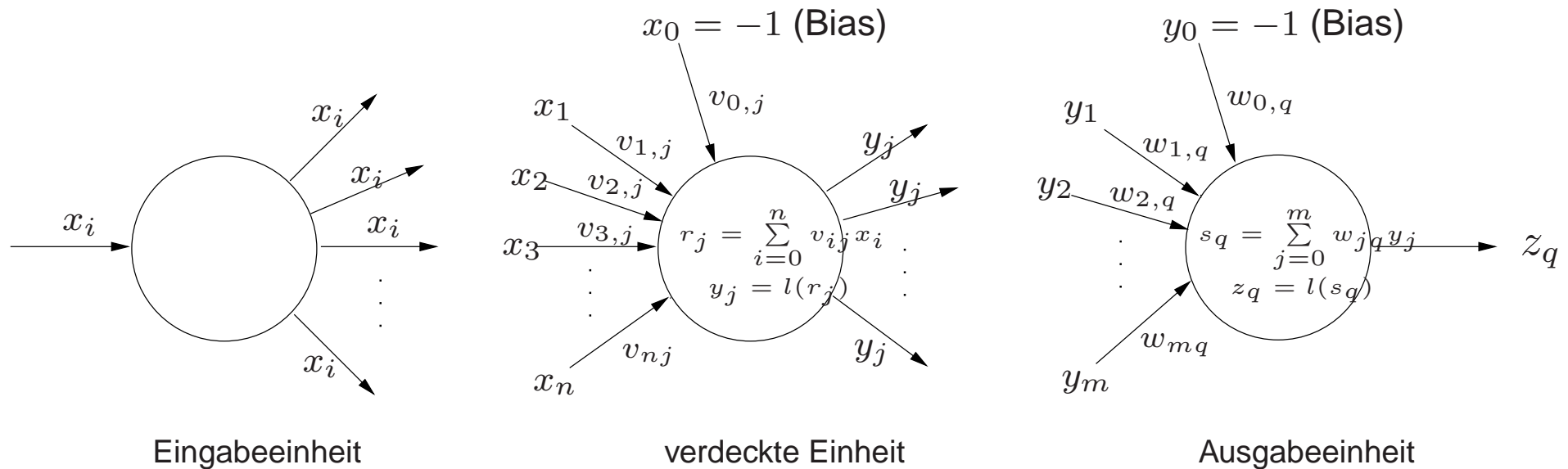
$$O(s \log_2((2n)^k)) = O(sk \log_2 n).$$

□

## 2.2 Vorwärtsgerichtete neuronale Netze



Vollständig verbundenes feed-forward Netz mit einer verdeckten Schicht



Die Einheiten der verschiedenen Schichten.

Ankommenden Werte bei der verdeckten Einheit und beim Ausgabeneuron werden mit  $l$  transformiert.

Der Wert des verdeckten Neurons bleibt für den Beobachter verborgen.

Jedes Neuron  $i$  der Eingabeschicht leitet ein bestimmtes Merkmal eines Objekts als binären (0/1 oder  $\pm 1$ ) oder reellen Wert  $x_i$  an jede Einheit der verdeckten Schicht mit unterschiedlichen Gewichten  $v_{ij}$  weiter.

Bias-Neuronen Nr. 0 haben Gewicht 1.

Jedes Neuron  $j > 0$  der verdeckten Schicht erhält als Eingabe die gewichtete Summe der Ausgabe der vorangehenden Schicht

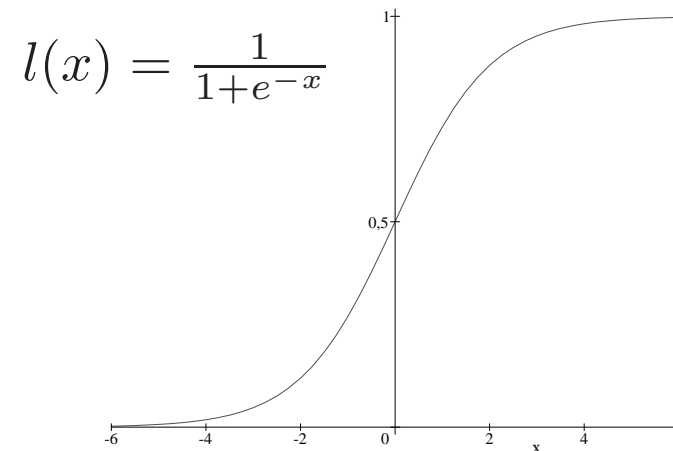
$$r_j = \sum_{i=0}^n v_{ij} x_i$$

und berechnet seine Ausgabe

$$y_j = a(r_j),$$

mit der *Aktivierungsfunktion*  $a(\cdot)$ ,  
z.B. die logistische Funktion  $l(\cdot)$ , d.h.

$$y_j = l(r_j) = \frac{1}{1 + e^{-r_j}}$$



$w_{jq}$  : Gewicht an der Kante zwischen verdecktem Neuron  $j$  und Ausgabeneuron  $q$ .

Ausgabeneuron  $q$  erhält als Eingabe die gewichtete Summe

$$s_q = \sum_{j=0}^m w_{jq} y_j$$

und berechnet seine Ausgabe  $z_q$  mit

$$z_q = l(s_q).$$

Jedes Ausgabeneuron repräsentiert dann eine Klasse.

Ein Muster, dessen Daten in das Netz eingegangen sind, wird der Klasse zuzuordnen, deren Ausgabeneuron den höchsten Wert liefert.

Zu den Eingaben  $x_i$  erhalten wir im zweischichtigen Netz mit

$$z_q = l\left(\sum_j w_{jq} l\left(\sum_i v_{ij} x_i\right)\right) =: \phi_{q,v,w}(x)$$

die zur Klasse  $q$  gehörige Ausgabefunktion zu festen Gewichten  $v$  und  $w$ .

## 2.3 Das Lernverfahren Backpropagation

Daten seien von der Form  $O = (X, T) \in \mathcal{X} \times \mathcal{T}$  mit Merkmalraum  $\mathcal{X}$  und

$$\mathcal{T} = \left\{ (t_0, \dots, t_{p-1}) \mid t_i \in \{0, 1\}, \sum t_i = 1 \right\}.$$

$o = (x, t)$  mit Muster  $x$  gehört der Klasse  $q$  an, wenn  $t_q = 1$  gilt.

$\phi : \mathcal{X} \rightarrow [0, 1]^p$  sei die Ausgabefunktion des Netzes

Die Entscheidungsregel:

$$\hat{T}(x) = (0, \dots, 0, t_q = 1, 0, \dots, 0) \quad \Leftrightarrow \quad \phi_q \geq \phi_r \quad \forall r \neq q,$$

Definiere  $\phi$  durch Minimieren der Fehlerfunktion

$$F(v, w) = \mathbf{E}\{\|\phi(X) - T\|^2\} = \sum_{q=0}^{p-1} \mathbf{E}\{|\phi_q(X) - T_q|^2\}$$

Minimiere  $F$  durch Gradientenabstieg: Verändere  $v_{ij}$  und  $w_{ij}$  im Schritt  $k$  um

$$\Delta_k v_{ij} = -\eta \frac{\partial}{\partial v_{ij}} F_k(v, w) \quad \text{und} \quad \Delta_k w_{jq} = -\eta \frac{\partial}{\partial w_{jq}} F_k(v, w)$$

Dabei ist

$$F_k(v, w) = \frac{1}{2} \sum_{q=0}^{p-1} (z_q^{(k)} - t_q^{(k)})^2$$

der beim  $k$ -ten Objekt beobachtete Fehler.

Die Schrittgröße  $\eta$  wird im Laufe des Verfahrens verkleinert.

$$\frac{\partial F_k}{\partial w_{jq}} = ?$$

$$F_k(v, w) = \frac{1}{2} \sum_{q=0}^{p-1} \left( z_q^{(k)} - t_q^{(k)} \right)^2$$

$$z_q^{(k)} = l(s_q), \quad s_q = \sum_{j=0}^m w_{jq} y_j$$

$$\frac{\partial F_k}{\partial w_{jq}} = \frac{\partial F_k}{\partial s_q^{(k)}} \frac{\partial s_q^{(k)}}{\partial w_{jq}} = \frac{\partial F_k}{\partial z_q^{(k)}} \frac{\partial z_q^{(k)}}{\partial s_q} \frac{\partial s_q}{\partial w_{jq}}$$

$$\frac{\partial F_k}{\partial z_q^{(k)}} = z_q^{(k)} - t_q^{(k)}, \quad \frac{\partial s_q}{\partial w_{jq}} = y_j$$

$$\frac{\partial z_q^{(k)}}{\partial s_q} = ?, \quad z_q^{(k)} = l(s_q) = \frac{1}{1 + e^{-s_q}}$$

$$l'(x) = \frac{-1}{(1 + e^{-x})^2} \cdot e^{-x} \cdot (-1) = \frac{1}{1 + e^{-x}} \cdot \frac{e^{-x}}{1 + e^{-x}} = l(x) \cdot (1 - l(x))$$

Also:

$$\frac{\partial z_q^{(k)}}{\partial s_q} = z_q^{(k)} \cdot (1 - z_q^{(k)})$$

$$\alpha_q^{(k)} = \frac{\partial F_k}{\partial s_q^{(k)}} = \frac{\partial F_k}{\partial z_q^{(k)}} \frac{\partial z_q^{(k)}}{\partial s_q^{(k)}} = (z_q^{(k)} - t_q^{(k)}) z_q^{(k)} (1 - z_q^{(k)}).$$

heißt *Fehlersignal* am  $q$ -ten Ausgabeneuron im  $k$ -ten Lernschritt.

Wir erhalten damit

$$\Delta_k w_{jq} = -\eta \frac{\partial F_k}{\partial w_{jq}} = -\eta \frac{\partial F_k}{\partial s_q^{(k)}} \cdot \frac{\partial s_q^{(k)}}{\partial w_{jq}} = -\eta \alpha_q^{(k)} y_j^{(k)}$$

Nun zu

$$\frac{\partial F_k}{\partial v_{ij}} = \frac{\partial F_k}{\partial r_j^{(k)}} \frac{\partial r_j^{(k)}}{\partial v_{ij}}.$$

Wir erinnern uns:

$$F_k(v, w) = \frac{1}{2} \sum_{q=0}^{p-1} \left( z_q^{(k)} - t_q^{(k)} \right)^2$$

$$z_q = l \left( \sum_j w_{jq} y_j \right), \quad y_i = l(r_j), \quad r_j = \sum_i v_{ij} x_i$$

Also

$$\frac{\partial r_j^{(k)}}{\partial v_{ij}} = x_i^{(k)}$$

In das Fehlersignal am  $j$ -ten Verdeckten Neuron ist

$$\begin{aligned} \delta_j^{(k)} &:= \frac{\partial F_k}{\partial r_j^{(k)}} = \frac{\partial y_j^{(k)}}{\partial r_j^{(k)}} \frac{\partial F_k}{\partial y_j^{(k)}} = y_j^{(k)} (1 - y_j^{(k)}) \sum_q w_{jq} z_q^{(k)} (1 - z_q^{(k)}) (z_q^{(k)} - t_q^{(k)}) \\ &= y_j^{(k)} (1 - y_j^{(k)}) \sum_q w_{jq} \alpha_q^{(k)}. \end{aligned}$$

Wir erhalten also:

$$\Delta_k v_{ij} = -\eta x_i^{(k)} \delta_j^{(k)} \quad \text{für } j > 0,$$

mit

$$\delta_j^{(k)} = y_j^{(k)} (1 - y_j^{(k)}) \sum_q w_{jq} \alpha_q^{(k)}.$$

Auch bei mehr als einer verdeckten Schicht lassen sich die Fehlersignale aus denen der jeweils höheren Schicht berechnen. Daher der Name “Backpropagation”.

Eine Backpropagation-Iteration für ein  $N$ -schichtiges Netz:

1. Dem Netz wird ein Muster  $x^{(k)}$  eines zufällig ausgewählten Trainingsobjekts  $o^{(k)} = (x^{(k)}, t^{(k)})$  präsentiert:  $v_i^0 = x_i^{(k)}$
2. Es verarbeitet die eingegangene Information *vorwärts* von Schicht zu Schicht weiter gemäß  $v_j^n = l\left(\sum_i w_{ij}^n v_i^{n-1}\right)$ ,
3. Es erfährt den wahren Typ  $t^{(k)}$  und berechnet die Fehlersignale  $\delta^N = \alpha$  der Ausgabeneuronen:  $\delta_j^N = v_j^N (1 - v_j^N)(v_j^N - t_j^{(k)})$
4. Es berechnet die Fehlersignale *rückwärts* von Schicht zu Schicht gemäß  $\delta_i^{n-1} = v_i^{n-1} (1 - v_i^{n-1}) \sum_j w_{ij}^n \delta_j^n$  für  $n = N, N - 1, \dots, 2$ .
5. Es ändert seine Gewichte um den Zuwachs  $\Delta w_{ij}^n = -\eta \delta_j^n v_i^{n-1}$  für  $n = 1, \dots, N$ .

## 2.4 PAC-Lernen

PAC = Probably Approximately Correct

Ziel: Lerne aus Beispielen, so dass mit einer Wahrscheinlichkeit von mindestens  $1 - \delta$  der Fehler beim Klassifizieren höchstens  $\epsilon$  ist.

Ist  $D$  eine W'keitsverteilung auf  $\Omega$  und  $C \in \mathcal{C}$  das zu lernende Konzept, so können wir den **Fehler** der Hypothese  $H \in \mathcal{H}$  durch  $F_D(C, H) := D((C \cup H) \setminus (C \cap H))$  definieren.

**Fairness-Bedingung:** Bewerte den Fehler mit derselben Verteilung  $D$ , gemäß der die Lernbeispiele erzeugt wurden.

# PAC-Algorithmus für eine Konzeptklasse $\mathcal{C}$

## Ablauf:

- Eingabe: Vertrauensparameter  $\delta$ , Fehlerparameter  $\epsilon$ , Länge  $n$  der Beispiele
- Bestimme Anzahl  $s = s(\delta, \epsilon, n)$  der anzufordernden Beispiele.
- Lies  $s$  Beispiele an
- Gib ein  $H \in \mathcal{H}$  aus.

**Kriterium:** Für alle  $\epsilon, \delta \in (0, 1]$ ,  $C \in \mathcal{C}$  und alle Verteilungen  $D$  auf  $\Omega$  gilt:

$$W_{sD}(F_D(C, H) \leq \epsilon) \geq 1 - \delta$$

( $W_{sD}(\cdot)$  := W'keit unter der Voraussetzung, dass die Lerndaten aus  $D$  kommen)

Falls  $\mathcal{H} = \mathcal{C}$  und es leicht ist, für jede mit  $C \in \mathcal{C}$  verträgliche Folge von Beispielen  $\{(x_1, b_1), \dots, (x_n, b_n)\} \subset \Omega \times \{0, 1\}$  (d.h.  $b_i = 1 \Leftrightarrow x_i \in C$ ) eine konsistente Hypothese  $H \in \mathcal{H}$  zu finden (d.h.  $b_i = 1 \Leftrightarrow x_i \in H$ ), dann

- setze  $s := \lceil (\ln(|\mathcal{C}|) - \ln(\delta)) / \varepsilon \rceil$
- lies  $s$  Beispiele ein
- Gib dazu konsistente Hypothese  $H$  aus

**Satz 3** *Damit erhalten wir einen PAC-Algorithmus.*

**Beweis:**

$$\begin{aligned} \mathbb{W}_{s_D}(F_D(C, H) > \varepsilon) &\leq \mathbb{W}_{s_D}(\exists \text{ konsistentes } H \in \mathcal{C} : F_D(C, H) > \varepsilon) \\ &\leq \sum_{H \in \mathcal{C} : F_D(C, H) > \varepsilon} \mathbb{W}_{s_D}(H \text{ konsitent mit allen } (x_i, b_i)) \\ &\leq |\mathcal{C}|(1 - \varepsilon)^s \end{aligned}$$

Wähle also  $s$  so groß, dass  $|\mathcal{C}|(1 - \varepsilon)^s \leq \delta$  gilt.

Logarithmieren ergibt  $\ln |\mathcal{C}| + s \ln(1 - \varepsilon) \leq \ln \delta$  und wegen  $\ln(1 - \varepsilon) \leq -\varepsilon$  genügt  $s \geq (\ln |\mathcal{C}| - \ln \delta) / \varepsilon$ . □

**Satz 4** *Aus jedem Online-Algorithmus für  $\mathcal{C}$ , der stets weniger als  $g$  Gegenbeispiele anfordert, erhält man einen PAC-Algorithmus, der nur  $O(\frac{1}{\varepsilon}(g + \ln \frac{g}{\delta}))$  Beispiele anfordert.*

**Satz 5** Seien  $A_1 \dots, A_n$  unabhängige Ereignisse, die jeweils mit Wahrscheinlichkeit  $p$  eintreten und sei  $K$  die Anzahl der ‘Erfolge’.

Dann gilt für jedes  $\gamma \in [0, 1]$ :

- Chernoff-Schranke:

$$\text{Ws}(K > (1 + \gamma) \cdot pn) \leq e^{-np\gamma^2/3}$$

$$\text{Ws}(K < (1 - \gamma) \cdot pn) \leq e^{-np\gamma^2/2}$$

- Hoeffding-Ungleichungen (auch: “additive Chernoff-Schranke”):

$$\text{Ws}(K > (p + \gamma) \cdot n) \leq e^{-2n\gamma^2}$$

$$\text{Ws}(K < (p - \gamma) \cdot n) \leq e^{-2n\gamma^2}$$

**Beweis:** Siehe Stochastik-Literatur

## Beweis von Satz 4:

### Algorithmus:

- Phase 1: Suche mit dem Online-Algo und den ersten  $\frac{4}{\varepsilon}(g + 4 \ln \frac{2}{\delta})$  Beispielen nach Hypothesen
- Phase 2: Überprüfe **alle in Phase 1 betrachteten Hypothesen** an weiteren  $\frac{32}{\varepsilon} \ln \frac{2g}{\delta}$  Daten und gib die beste aus.

### Beweisschritte:

1. Falls alle betrachteten Hypothesen Fehler  $\geq \varepsilon/2$  haben, werden mit  $W'$ keit  $\geq 1 - \delta/2$  mindestens  $g$  Gegenbeispiele gefunden, und damit auch das Zielkonzept mit Fehler 0.
2. Jede Hypothese mit Fehler  $\leq \varepsilon/2$  macht in Phase 2 mit  $W'$ keit  $\geq 1 - \delta/(2g)$  höchstens und jede Hypothese mit Fehler  $\geq \varepsilon$  mit  $W'$ keit  $\geq 1 - \delta/(2g)$  mindestens  $24 \cdot \ln \frac{2g}{\delta}$  Fehler.

Schritt 1 folgt aus der ersten Chernoff-Schranke mit  $\gamma = 1/2$ ,  $p = \varepsilon/2$  und  $n = \frac{4}{\varepsilon}(g + 4 \ln \frac{2}{\delta})$  und  $K$  =Anzahl gefundener Gegenbeispiele:

$$\begin{aligned} \text{Ws}(K < g) &\leq \text{Ws} \left( K < \left( 1 - \frac{1}{2} \right) \cdot \frac{\varepsilon}{2} \cdot \frac{4}{\varepsilon} \left( g + 4 \ln \frac{2}{\delta} \right) \right) \\ &\leq e^{-(2g+8 \ln \frac{2}{\delta})/8} \leq e^{-\frac{g}{4}} \cdot \frac{\delta}{2} \leq \frac{\delta}{2} \end{aligned}$$

Für Schritt 2 setzen wir  $n = \frac{32}{\varepsilon} \ln \frac{2g}{\delta}$  und wenden die erste Chernoff-Schranke mit  $p = \frac{\varepsilon}{2}$  und  $\gamma = \frac{1}{2}$  bzw. die zweite mit  $p = \varepsilon$  und  $\gamma = \frac{1}{4}$  an:

$$\begin{aligned}
 \text{Ws} \left( K_1 > 24 \cdot \ln \frac{2g}{\delta} \right) &= \text{Ws} \left( K_1 > \left( 1 + \frac{1}{2} \right) \cdot \frac{\varepsilon}{2} \cdot \frac{32}{\varepsilon} \ln \frac{2g}{\delta} \right) \\
 &\leq e^{-16 \cdot \ln \left( \frac{2g}{\delta} \right) / 12} = \left( \frac{\delta}{2g} \right)^{\frac{4}{3}} \leq \frac{\delta}{2g} \\
 \text{Ws} \left( K_2 < 24 \cdot \ln \frac{2g}{\delta} \right) &\leq \text{Ws} \left( K_2 < \left( 1 - \frac{1}{4} \right) \cdot \varepsilon \cdot \frac{32}{\varepsilon} \ln \frac{2g}{\delta} \right) \\
 &\leq e^{-32 \cdot \ln \left( \frac{2g}{\delta} \right) \cdot \left( \frac{1}{4} \right)^2 / 2} = \frac{\delta}{2g}
 \end{aligned}$$

□

## Was ist wenn $\mathcal{C} \not\subseteq \mathcal{H}$ ?

Annahme:  $\exists H^+ \in \mathcal{H} : F_D(C, H^+) \leq \varepsilon$

Ziel: finde  $H \in \mathcal{H}$  mit  $F_D(C, H) \leq 4 \cdot \varepsilon$

Algorithmus:

- bestimme  $s$  für gegebene  $\delta$  und  $\varepsilon$
- lies  $s$  klassifizierte Beispiele ein
- Wenn Hypothese  $H \in \mathcal{H}$  existiert, die  $(1 - 2\varepsilon) \cdot s$  Beispiele korrekt klassifiziert, gib eine solche aus. Sonst Fehlermeldung.

Wie bestimmen wir  $s$ ?

Es soll gelten:

$Ws_D(\text{Fehlermeldung oder Ausgabe } H \text{ mit } F_D(C, H) \geq 4\varepsilon) \leq \delta.$

Dafür genügt:

$$Ws_D(\text{Fehlermeldung}) \leq \delta/2$$

$$Ws_D(\text{Ausgabe } H \text{ mit } F_D(C, H) \geq 4\varepsilon) \leq \delta/2$$

Aus der ersten Chernoff-Schranke mit  $\gamma = 1$  folgt:

$Ws_D(\text{Fehlermeldung}) \leq Ws(H^+ \text{ klassifiziert } \geq 2\varepsilon s \text{ Beispiele falsch}) \leq e^{-\varepsilon s/3}.$

Für  $Ws_D(\text{Fehlermeldung}) \leq \delta/2$  genügt also  $s \geq \frac{3}{\varepsilon} \ln \frac{2}{\delta}.$

Sei  $H \in \mathcal{H}$  mit  $F_D(H, C) \geq 4\varepsilon$ .

$\text{Ws}_D(H \text{ klassifiziert } \leq 2\varepsilon s \text{ Beispiele falsch}) \leq e^{-4\varepsilon \cdot s/8}$

(zweite Chernoff mit  $\gamma = \frac{1}{2}$ ).

W'keit, dass mindestens eines der  $H \in \mathcal{H}$  mit  $F_D(H, C) \geq 4\varepsilon$  mindestens  $(1 - 2\varepsilon)s$  Bsp. richtig klassifiziert, ist also  $\leq |\mathcal{H}| \cdot e^{-\varepsilon \cdot s/2}$ .

Damit das kleiner als  $\delta/2$  wird, genügt

$$s \geq \frac{2}{\varepsilon} \left( \ln |\mathcal{H}| + \ln \frac{2}{\delta} \right)$$

Wir haben also bewiesen:

**Satz 6** *Falls eine Hypothese  $H^+ \in \mathcal{H}$  mit  $F_D(H^+, C) \leq \varepsilon$  existiert, kann man mit  $\frac{3}{\varepsilon} (\ln |\mathcal{H}| + \ln \frac{2}{\delta})$  Beispielen mit W'keit  $\geq 1 - \delta$  eine Hypothese  $H \in \mathcal{H}$  mit  $F_D(H, C) \leq 4\varepsilon$  finden.*