

# Theorie des maschinellen Lernens

Hans U. Simon

1. Juni 2016

## 10 Boosting

In diesem Kapitel gehen wir der Frage nach: „Kann man aus den Ratschlägen eines Narren Weisheit schöpfen?“ Die Rolle des „Narren“ übernimmt ein Algorithmus, den wir als „schwachen Lerner“ bezeichnen. Seine Voraussagen sind nur etwas besser als zufälliges Raten, d.h., er sagt ein binäres Label mit einer etwas unter  $1/2$  liegenden Fehlerrate voraus. Den Part der „Weisheit“ übernimmt eine Voraussagefunktion, die ein gewichtetes Majoritätstotum über eine Kollektion von schwachen Hypothesen durchführt. Kern dieses Kapitels ist eine auf Freund und Schapire zurückgehende Methode namens AdaBoost, welche einen schwachen Lerner in einen (starken) PAC-Lerner (mit einer Fehlerrate nahe bei 0) transformiert.

In Abschnitt 10.1 beschäftigen wir uns mit dem Konzept eines „schwachen Lerners“. Abschnitt 10.2 liefert eine obere Schranke für die VC-Dimension der Klasse der Linearkombinationen von Basishypothesen. Dies wird uns später helfen, den Schätzfehler von AdaBoost zu kontrollieren. Der das Kapitel abschließende Abschnitt 10.3 ist AdaBoost und seiner Analyse gewidmet.

### 10.1 Schwache Lernbarkeit

**Definition 10.1** *Es sei  $0 < \gamma < 1/2$ .  $\mathcal{H}$  und  $\mathcal{B}$  seien Hypothesenklassen mit Grundbereich  $\mathcal{X}$ .*

- *Wir sagen ein Lernalgorithmus  $A$  lernt  $\mathcal{H}$   $\gamma$ -schwach mit Hypothesen aus  $\mathcal{B}$ , wenn eine Funktion  $m_{\mathcal{H}} : (0, 1) \rightarrow \mathbb{N}$  existiert, so dass für jedes  $\delta \in (0, 1)$ , jede Verteilung  $D$  über  $\mathcal{X}$  und jede Funktion  $f : \mathcal{X} \rightarrow \{-1, 1\}$  unter der „Realizability Assumption (RA)“ für  $\mathcal{H}, D, f$  folgendes gilt. Wenn  $A$  auf einer Trainingsmenge der Größe  $m \geq m_{\mathcal{H}}(\delta)$  gestartet wird, deren Instanzen bezüglich  $D^m$  generiert und von  $f$  gelabelt werden, dann ist mit einer Erfolgswahrscheinlichkeit von mindestens  $1 - \delta$  die Ausgabe von  $A$  eine Hypothese  $h \in \mathcal{B}$  mit  $L_{D,f}(h) \leq 1/2 - \gamma$ .*
- *$\mathcal{H}$  heißt  $\gamma$ -schwach lernbar mit Hypothesen aus  $\mathcal{B}$ , wenn ein entsprechender  $\gamma$ -schwacher Lerner existiert.*

**Beispiel 10.2** Es sei  $\mathcal{X} = \mathbb{R}$ . Weiter sei  $\mathcal{G}_T$  die Klasse aller Funktionen von  $\mathbb{R}$  nach  $\{-1, 1\}$  mit höchstens  $T$  Vorzeichenwechseln, wenn wir die reelle Zahlengerade von links nach rechts durchlaufen (= Klasse der  $\pm 1$ -wertigen  $(T + 1)$ -stückweise konstanten Funktionen). Zum Beispiel ist jede Funktion aus  $\mathcal{G}_2$  gegeben durch eine Zerlegung der reellen Zahlengeraden in drei (oder weniger) Intervalle  $I_1, I_2, I_3$ , geordnet von links nach rechts, auf denen entweder das Binärmuster  $1, -1, 1$  oder das Binärmuster  $-1, 1, -1$  angenommen wird. Eine Schwellenfunktion (mit Vorzeichen) ist eine Funktion der Form  $x \mapsto b \cdot \text{sign}(x - \theta)$  mit  $\theta \in \mathbb{R}$  und  $b \in \{-1, 1\}$ . Sie ordnet  $x$  den Wert  $b$  zu, falls  $x \geq \theta$  und den Wert  $-b$  andernfalls. Es sei  $\text{DS}_1$  die Klasse aller Funktionen dieser Form. Wir zeigen, dass ein ERM-Algorithmus  $A$  für die Klasse  $\text{DS}_1$  zu einem  $1/12$ -schwachen Lerner für die Klasse  $\mathcal{G}_2$  aufgebaut werden kann. Es sei  $S$  eine Trainingsmenge, die oBdA so gelabelt ist, dass Punkte auf den Intervallen  $I_1, I_2, I_3$  jeweils die Werte  $-1, 1, -1$  annehmen. (Der Fall mit vertauschten  $\pm 1$ -Labels oder mit weniger als drei Intervallen ist analog zu behandeln.) Es ist leicht zu sehen, dass es zu jedem Intervall  $I \in \{I_1, I_2, I_3\}$  eine Hypothese aus  $\text{DS}_1$  gibt, die nur die Trainingspunkte in  $I$  falsch labeled. Daher gilt für eine ERM-Hypothese  $A(S) \in \text{DS}_1$ :  $L_S(A(S)) \leq 1/3 = 1/2 - 1/6$ . Die VC-Dimension von  $\text{DS}_1$  ist gleich 2 (einfach zu sehen). Daher hat  $\text{DS}_1$  die „Uniform Convergence Property (UCP)“. Es sei  $D$  die auf  $\mathbb{R}$  zugrunde liegende Verteilung. Wir können die Größe der Trainingsmenge  $S$  so bemessen, dass mit einer Wahrscheinlichkeit von mindestens  $1 - \delta$  die absolute Differenz von  $L_D(h)$  und  $L_S(h)$  durch  $1/12$  beschränkt ist. Dann gilt:  $L_D(A(S)) \leq 1/2 - 1/6 + 1/12 = 1/2 - 1/12$ .

Die Funktionen der Klasse  $\text{DS}_1$  werden auch *Entscheidungsstümpfe* (*decision stumps*) über  $\mathbb{R}$  genannt (hauptsächlich weil sie als rudimentäre Entscheidungsbäume gesehen werden können).

Es sei

$$S = [(x_1, y_1), \dots, (x_m, y_m)] \in (\mathbb{R} \times (\mathbb{R} \setminus \{0\}))^m \quad \text{mit} \quad x_1 < x_2 < \dots < x_m \quad (1)$$

eine „sortierte gewichtete Trainingssequenz“. Die Gewichte können wahlweise Mehrfachvorkommen einer Instanz oder auch Wahrscheinlichkeiten repräsentieren. Zum Beispiel könnte  $(x_i, -3)$  das 3-fache Vorkommen einer mit  $-1$  markierten Instanz  $x_i$  repräsentieren.

Wir wollen im Folgenden überlegen, welcher Entscheidungsstumpf auf  $S$  die kleinste (gewichtete) Anzahl von Fehlern macht. Es ist leicht zu sehen, dass wir nur Hypothesen aus  $\text{DS}_1$  mit  $\theta \in \{x_1, \dots, x_m\}$  in Betracht ziehen müssen. Für  $j = 1, \dots, m$  sei

$$W^+(j) = \sum_{i \in [j]: y_i > 0} y_i \quad \text{und} \quad W^-(j) = \sum_{i \in [j]: y_i < 0} |y_i| . \quad (2)$$

Weiter sei  $W^+ = W^+(m)$  bzw.  $W^- = W^-(m)$  die Summe aller positiven Gewichte bzw. aller Absolutbeträge von negativen Gewichten. Die folgende Funktion wird sich als nützlich erweisen:

$$F(j) = W^+(j) - W^-(j) = \sum_{i \in [j]} y_i , \quad (3)$$

d.h.,  $F(j)$  ist die Summe der ersten  $j$  Gewichte, wobei  $F(0) = 0$ . Es bezeichne  $\text{DS}_1^+$  (bzw.  $\text{DS}_1^-$ ) die Menge der Hypothesen aus  $\text{DS}_1$  mit  $b = 1$  (bzw.  $b = -1$ ).

**Lemma 10.3** *Es sei  $j \in [m]$ . Dann gilt folgendes. Die Hypothese  $h_j^+(x) = \text{sign}(x - x_j)$  macht auf  $S$  insgesamt  $W^- + F(j - 1)$  Fehler und die Hypothese  $h_j^-(x) = -\text{sign}(x - x_j)$  macht auf  $S$  insgesamt  $W^+ - F(j - 1)$  Fehler.*

**Beweis** Wir beweisen die erste Aussage. (Der Beweis für die zweite Aussage ist analog zu führen.) Die Hypothese  $h_j^+$  macht  $W^+(j - 1)$  Fehler auf  $(x_1, y_1), \dots, (x_{j-1}, y_{j-1})$  und  $W^- - W^-(j - 1)$  Fehler auf  $(x_j, y_j), \dots, (x_m, y_m)$ . Die gesamte Fehlerzahl von  $h_j^+$  beträgt demnach  $W^+(j - 1) + (W^- - W^-(j - 1)) = W^- + F(j - 1)$ , wie behauptet. **qed.**

Dies führt zu dem folgenden Algorithmus  $A_{DS}$ , der die ERM-Lernregel für die Klasse  $DS_1$  implementiert:

1. Transformiere eine Trainingssequenz in die zugehörige sortierte ganzzahlig-gewichtete Trainingssequenz  $S$  der Form (1).
2. Berechne die Größen  $W^+$ ,  $W^-$  und  $F(j)$  für  $j = 0, 1, \dots, m$ .
3. Bestimme  $j^+ = \text{argmin}_{j \in [m]} F(j - 1)$  und  $j^- = \text{argmax}_{j \in [m]} F(j - 1)$ .
4. Falls  $W^- + F(j^+ - 1) \leq W^+ - F(j^- - 1)$ , dann gib die Hypothese  $\text{sign}(x - x_{j^+})$  aus, andernfalls die Hypothese  $-\text{sign}(x - x_{j^-})$ .

Die Laufzeit für den ersten Schritt wird vom Sortieren dominiert und beträgt somit  $O(|S| \log |S|)$ . Die weiteren Schritte lassen sich in Linearzeit ausführen. Wir fassen diese Diskussionen in folgendem Resultat zusammen:

**Lemma 10.4** *Die ERM-Lernregel für die Hypothesenklasse  $DS_1$  ist mit Laufzeitschranke  $O(|S| \log |S|)$  implementierbar (bzw. sogar Linearzeit, falls die Trainingssequenz bereits sortiert gegeben ist).*

Wir kommen nun zu einer  $d$ -dimensionalen Verallgemeinerung der Klasse  $DS_1$ . Die Klasse der Entscheidungsstümpfe über  $\mathbb{R}^d$  ist gegeben durch

$$DS_d = \{x \mapsto b \cdot \text{sign}(x_i - \theta) : b \in \{-1, 1\}, i \in [d], \theta \in \mathbb{R}\} .$$

Wir können also eine Hypothese aus  $DS_d$  wählen, indem wir uns für eine Dimension  $i \in [d]$  entscheiden und dann einen 1-dimensionalen Entscheidungsstumpf auswählen, welcher auf  $x_i$  angewendet wird.

Der obige Algorithmus zur Implementierung der ERM-Lernregel für die Klasse  $DS_1$  lässt sich auf die offensichtliche Weise benutzen, um die ERM-Lernregel auf  $DS_d$  zu implementieren:

- Wende  $A_{DS}$   $d$ -mal an (eine Anwendung pro Dimension) und erhalte die jeweils besten Hypothesen, sagen wir  $h_1, \dots, h_d$  sowie die Anzahl der Fehler, die sie jeweils auf der Trainingssequenz machen.

- Wähle unter  $h_1, \dots, h_d$  die Hypothese mit der kleinsten Fehleranzahl aus.

Diese Vorgehensweise führt zu folgendem Ergebnis:

**Theorem 10.5** Die ERM-Lernregel für die Hypothesenklasse  $DS_d$  ist mit Laufzeitschranke  $O(d \cdot |S| \log |S|)$  implementierbar.

Wie man sieht wird die Laufzeit dominiert durch das  $d$ -malige Sortieren der Trainingssequenz (eine Sortierung für jede der  $d$  Dimensionen).

## 10.2 Linearkombination von Basishypothesen

Wir werden sehen, dass geeignete Linearkombinationen von Hypothesen, welche von einem schwachen Lerner produziert wurden, eine sehr kleine empirische Fehlerrate auf den Trainingsdaten haben. Mit anderen Worten: der Approximationsfehler wird sich als kontrollierbar erweisen. Dies wirft die Frage nach der Kontrollierbarkeit des Schätzfehlers und somit die Frage nach der VC-Dimension auf.

Es sei nun  $\mathcal{B}$  eine Klasse von „Basishypothesen“ über einem Grundbereich  $\mathcal{X}$  und  $T \in \mathbb{N}$ . Wir definieren

$$\text{LIN}(\mathcal{B}, T) = \left\{ x \mapsto \text{sign} \left( \sum_{t=1}^T w_t h_t(x) \right) : w \in \mathbb{R}^T, h_1, \dots, h_T \in \mathcal{B} \right\} .$$

Der Einfachheit halber nennen wir die Elemente von  $\text{LIN}(\mathcal{B}, T)$  „Linearkombinationen“ von  $T$  Basisfunktionen aus  $\mathcal{B}$  (obwohl es sich, streng genommen, um die sign-Funktion angewendet auf eine solche Linearkombination handelt).

Wie mächtig kann die Klasse  $\text{LIN}(\mathcal{B}, T)$  im Vergleich zu  $\mathcal{B}$  werden? Wir betrachten zunächst ein

**Beispiel 10.6** Es sei  $\mathcal{G}_T$  die bereits in Beispiel 10.2 eingeführte Hypothesenklasse. Offensichtlich gilt  $VCdim(\mathcal{G}_T) = T + 1$ . Es ist nicht schwer zu zeigen, dass  $\mathcal{G}_T$  eine Teilklasse von  $\text{LIN}(DS_1, T)$  ist. S. Übung. Es folgt, dass  $VCdim(\text{LIN}(DS_1, T)) \geq VCdim(\mathcal{G}_T) = T + 1$ , wohingegen  $VCdim(DS_1) = 2$ .

Allgemeinere Einsichten ergeben sich aus folgendem

**Theorem 10.7** Es sei  $\mathcal{B}$  eine Klasse von Basishypothesen der VC-Dimension  $d \geq 3$  und  $3 \leq T \in \mathbb{N}$ . Dann gilt:

$$VCdim(\text{LIN}(\mathcal{B}, T)) \leq \underbrace{T(d+1)(2+3\log(d+1))}_{=D(T,d)} .$$

**Beweis** Es seien  $x_1, \dots, x_m \in \mathcal{X}$ . Wir sagen eine Matrix  $B \in \{-1, 1\}^{T \times m}$  ist *realisierbar* durch  $\mathcal{B}$ , falls  $h_1, \dots, h_T \in \mathcal{B}$  existieren mit  $h_t(x_i) = B[t, i]$  für alle  $t \in [T]$  und für alle  $i \in [m]$ . Mit Sauer's Lemma folgt, dass in jeder Zeile  $[h_t(x_1), \dots, h_t(x_m)]$  von  $B$  höchstens  $(em/d)^d$  Binärmuster durch  $\mathcal{B}$  realisierbar sind. Somit sind höchstens  $(em/d)^{dT}$  verschiedene Matrizen  $B \in \{-1, 1\}^{T \times m}$  durch  $\mathcal{B}$  realisierbar. Die Realisierung eines Binärmusters auf  $x_1, \dots, x_m$  durch die Klasse  $\text{LIN}(\mathcal{B}, T)$  können wir uns zweistufig vorstellen wie folgt:

1. Wir wählen  $h_1, \dots, h_T$  und bilden  $x_i$  auf den Spaltenvektor  $B_i = (h_1(x_i), \dots, h_T(x_i))^\top$  ab (für alle  $i \in [m]$ ).
2. Zu jeder festen (durch  $\mathcal{B}$  realisierbaren) Wahl von  $B = [B_1 \dots B_m]$  wählen wir einen Gewichtsvektor  $w \in \mathbb{R}^T$  und bilden  $B_i$  auf  $\text{sign}(\langle w, B_i \rangle)$  ab (für alle  $i \in [m]$ ).

Wie bereits oben festgestellt gibt es höchstens  $(em/d)^{dT}$  mögliche Wahlen von  $B$ . Da die VC-Dimension von  $\text{sign} \circ \text{LIN}_T^0$  gleich  $T$  ist, gibt es für jede feste Wahl von  $B$  höchstens  $(em/T)^T$  Binärmuster, die auf  $B_1, \dots, B_m$  durch Wahl eines Gewichtsvektors  $w$  realisierbar sind. Insgesamt sind in diesem zweistufigen Prozess also maximal

$$(em/d)^{dT} \cdot (em/T)^T < m^{(d+1)T}$$

Binärmuster realisierbar. Falls  $x_1, \dots, x_m$  durch  $\text{LIN}(\mathcal{B}, T)$  aufspaltbar wären, dann müsste

$$2^m \leq m^{(d+1)T} \quad (4)$$

gelten. Dies ist für „große“ Werte von  $m$  sicherlich falsch, da die linke Seite von (4) exponentiell, die rechte jedoch nur polynomiell, in Abhängigkeit von  $m$  wächst. Eine genauere Rechnung (s. Lemma A.1 im Appendix A des Lehrbuches) zeigt, dass (4) für alle  $m > T(d+1)(2+3\log(d+1))$  falsch ist. Daher muss die VC-Dimension von  $\text{LIN}(\mathcal{B}, T)$  durch  $T(d+1)(2+3\log(d+1))$  nach oben beschränkt sein. **qed.**

Wir merken kurz an, dass wir die Voraussetzung  $T, d \geq 3$  in Theorem 10.7 eliminieren können, wenn wir in der oberen Schranke des Theorems  $T$  bzw.  $d$  durch  $\max\{3, T\}$  bzw.  $\max\{3, d\}$  ersetzen.

### 10.3 AdaBoost

„AdaBoost“ steht für „Adaptive Boosting“ und bezeichnet ein von Freund und Schapire entworfenes Verfahren, einen schwachen Lerner WL für eine Hypothesenklasse  $\mathcal{H}$  in einen (starken) PAC-Lerner für  $\mathcal{H}$  zu transformieren. Falls WL Hypothesen aus einer Basisklasse  $\mathcal{B}$  verwendet, dann verwendet AdaBoost Hypothesen aus  $\cup_{T \geq 1} \text{LIN}(\mathcal{B}, T)$ . AdaBoost wird WL als Hilfsprozedur einsetzen. Ein Aufruf von WL hat die allgemeine Form  $\text{WL}(P, S, \delta')$ . Hierbei bezeichnet  $S \in (\mathcal{X} \times \{-1, 1\})^m$  eine Trainingssequenz,  $P$  eine Verteilung auf  $[m]$  (und damit auch auf  $S_{\mathcal{X}}$ ) und  $0 < \delta' < 1$  den Zuverlässigkeitsparameter. Die Parameter  $P, S$  repräsentieren eine auf  $S_{\mathcal{X}}$  konzentrierte Verteilung auf  $\mathcal{X}$  und WL wird mit einer Erfolgswahrscheinlichkeit von mindestens  $1 - \delta'$  eine Hypothese  $h \in \mathcal{B}$  mit  $L_P(h) \leq 1/2 - \gamma$  abliefern.

**Bemerkung 10.8** Da wir WL über die Parameter  $P, S$  die Verteilung auf  $\mathcal{X}$  mitteilen, kann (eine entsprechende Erweiterung von) WL sich die von ihm benötigte Trainingssequenz selber generieren. Die markierten Beispiele werden dabei  $P$ -zufällig aus  $S$  gezogen.

Falls WL die ERM-Lernregel für  $\mathcal{B}$  auf gewichteten Trainingssequenzen implementiert (wie es zum Beispiel für  $\mathcal{B} = \text{DS}_d$  auf effiziente Weise möglich ist) so vereinfacht sich die Vorgehensweise: die Prozedur WL kann dann unmittelbar auf die mit  $P$  gewichtete Trainingssequenz  $S$  angewendet werden.

AdaBoost wird rundenweise vorgehen, in jeder Runde  $t$  folgende Objekte generieren:

- eine Basishypothese  $h_t$  (mit Hilfe von WL)
- die Fehlerrate  $\varepsilon_t$  von  $h_t$  bezüglich der in dieser Runde gültigen Verteilung  $D^{(t)}$  auf  $S$
- einen Gewichtsparameter  $w_t$
- die für die nächste Runde gültige Verteilung  $D^{(t+1)}$

Am Ende wird eine Art gewichtetes Majoritätvotum der Basishypothesen  $h_t$  ausgegeben.

Der Schlüssel zum Verständnis von AdaBoost liegt in der Wahl der Verteilungen  $D^{(t)}$ . Sie legen ein vergleichsweise hohes Gewicht auf Beispiele, die bei einem gewichteten Majoritätvotum (zum gegenwärtigen Zeitpunkt) falsch behandelt würden. Dies zwingt den schwachen Lerner, den „problematischen“ Beispielen verstärkte Aufmerksamkeit zu schenken.

Es folgt eine kompakte Beschreibung des Algorithmus' AdaBoost.

**Eingabedaten:** eine zufällige und zu einer Hypothese aus  $\mathcal{H}$  konsistente Trainingssequenz  $S = [(x_1, y_1), \dots, (x_m, y_m)] \in (\mathcal{X} \times \{-1, 1\})^m$ , ein Parameter  $T \in \mathbb{N}$  für die Anzahl der Runden sowie ein Zuverlässigkeitsparameter  $\delta$

**Hilfsprozedur:** ein schwacher Lerner WL für  $\mathcal{H}$ , welcher Hypothesen aus der Basisklasse  $\mathcal{B}$  produziert

**Initialisierung:**  $D^{(1)} = (1/m, \dots, 1/m)$ .

**Hauptschleife:** Für  $t = 1, \dots, T$  mache folgendes:

$$h_t = \text{WL} \left( D^{(t)}, S, \frac{\delta}{2T} \right), \quad (5)$$

$$\varepsilon_t = \sum_{i=1}^m D_i^{(t)} \mathbb{1}_{[h_t(x_i) \neq y_i]}, \quad (6)$$

$$w_t = \frac{1}{2} \log \left( \frac{1}{\varepsilon_t} - 1 \right), \quad (7)$$

$$D_i^{(t+1)} = \frac{D_i^{(t)} \exp(-w_t y_i h_t(x_i))}{\sum_{j=1}^m D_j^{(t)} \exp(-w_t y_j h_t(x_j))} \text{ für } i = 1, \dots, m \quad (8)$$

**Ausgabe:**  $h_S(x) = \text{sign} \left( \sum_{t=1}^T w_t h_t(x) \right)$ .

Der folgende Satz beschreibt, wie der Approximationsfehler von AdaBoost kontrolliert werden kann.

**Theorem 10.9** *Mit den im Algorithmus AdaBoost verwendeten Notationen gilt folgendes. Falls WL in jeder Runde  $t \in [T]$  eine Hypothese  $h_t$  mit*

$$\varepsilon_t \leq 1/2 - \gamma \quad (9)$$

*abliefern, dann gilt für die von AdaBoost ausgegebene Hypothese*

$$L_S(h_S) = \frac{1}{m} \sum_{i=1}^m \mathbb{1}_{[h_S(x_i) \neq y_i]} \leq e^{-2\gamma^2 T} .$$

**Beweis** Für  $t = 0, 1, \dots, T$  definieren wir

$$f_t = \sum_{p=1}^t w_p h_p \quad \text{und} \quad Z_t = \frac{1}{m} \sum_{i=1}^m \exp(-y_i f_t(x_i)) . \quad (10)$$

Es gilt dann  $f_0 = 0$  und  $Z_0 = 1$ . Expandieren der rekursiven Gleichung (8) für  $D_i^{(t+1)}$  liefert

$$D_i^{(t+1)} = \frac{e^{-y_i f_t(x_i)}}{\sum_{j=1}^m e^{-y_j f_t(x_j)}} . \quad (11)$$

Die von AdaBoost ausgegebene Hypothese  $h_S$  hat die Form  $H_S = \text{sign} \circ f_T$ . Für eine beliebige reellwertige Funktion  $f$  und  $(x, y) \in \mathcal{X} \times \{-1, 1\}$  gilt  $\mathbb{1}_{[\text{sign}(f(x)) \neq y]} \leq e^{-yf(x)}$ . Folglich ist  $Z_T$  eine obere Schranke für  $L_S(h_S) = L_S(\text{sign} \circ f_T)$ . Es genügt daher  $Z_T \leq e^{-2\gamma^2 T}$  nachzuweisen. Wegen  $Z_0 = 1$  kann  $Z_T$  als Teleskopprodukt geschrieben werden:

$$Z_T = \frac{Z_T}{Z_0} = \prod_{t=0}^{T-1} \frac{Z_{t+1}}{Z_t} .$$

Also brauchen wir für jedes  $t$  lediglich die Ungleichung  $Z_{t+1}/Z_t \leq e^{-2\gamma^2}$  nachzuweisen. Dies

leistet die folgende Rechnung:

$$\begin{aligned}
\frac{Z_{t+1}}{Z_t} &\stackrel{(10)}{=} \frac{\sum_{i=1}^m \exp(-y_i f_{t+1}(x_i))}{\sum_{j=1}^m \exp(-y_j f_t(x_j))} \\
&\stackrel{(10)}{=} \frac{\sum_{i=1}^m \exp(-y_i f_t(x_i)) \exp(-y_i w_{t+1} h_{t+1}(x_i))}{\sum_{j=1}^m \exp(-y_j f_t(x_j))} \\
&\stackrel{(11)}{=} \sum_{i=1}^m D_i^{(t+1)} \exp(-y_i w_{t+1} h_{t+1}(x_i)) \\
&= e^{-w_{t+1}} \sum_{i: y_i h_{t+1}(x_i)=1} D_i^{(t+1)} + e^{w_{t+1}} \sum_{i: y_i h_{t+1}(x_i)=-1} D_i^{(t+1)} \\
&= e^{-w_{t+1}} (1 - \varepsilon_{t+1}) + e^{w_{t+1}} \varepsilon_{t+1} \\
&\stackrel{(7)}{=} \frac{1}{\sqrt{1/\varepsilon_{t+1} - 1}} (1 - \varepsilon_{t+1}) + \sqrt{1/\varepsilon_{t+1} - 1} \varepsilon_{t+1} \\
&= \sqrt{\frac{\varepsilon_{t+1}}{1 - \varepsilon_{t+1}}} (1 - \varepsilon_{t+1}) + \sqrt{\frac{1 - \varepsilon_{t+1}}{\varepsilon_{t+1}}} \varepsilon_{t+1} \\
&= 2\sqrt{\varepsilon_{t+1}(1 - \varepsilon_{t+1})} \\
&\stackrel{(9)}{\leq} 2\sqrt{\left(\frac{1}{2} - \gamma\right) \left(\frac{1}{2} + \gamma\right)} = (1 - 4\gamma^2)^{1/2} \leq e^{-2\gamma^2}
\end{aligned}$$

Im letzten Schritt haben wir die bekannte Ungleichung  $1 + a \leq e^a$  verwendet. **qed.**

Wir merken an, dass die Voraussetzung (9) in Theorem 10.9 mit einer Wahrscheinlichkeit von mindestens  $1 - \delta/2$  für alle  $t \in [T]$  erfüllt ist, da AdaBoost die Prozedur WL mit dem Parameter  $\delta/(2T)$  aufruft.

**Zusammensetzen der Puzzleteile.** Es bezeichne

$$m_D(\varepsilon, \delta) = O\left(\frac{1}{\varepsilon^2} \left(D + \log \frac{1}{\delta}\right)\right)$$

eine Anzahl von Trainingsbeispielen, die den Schätzfehler für eine Hypothesenklasse der VC-Dimension  $D$  mit einer Wahrscheinlichkeit von mindestens  $1 - \delta$  auf  $\varepsilon$  begrenzt. Die Basisklasse  $\mathcal{B}$  habe die VC-Dimension  $d$ . Es sei  $D(T, d)$  die aus Theorem 10.7 bekannte obere Schranke von  $\text{VCdim}(\text{LIN}(\mathcal{B}, T))$ .

Wir wollen klären, wie die Parameter  $T$  und  $m$  gewählt werden müssen, damit AdaBoost mit einer Wahrscheinlichkeit von mindestens  $1 - \delta$  eine Hypothese mit einer Fehlerrate von höchstens  $\varepsilon$  abliefert. Es genügt zu diesem Zweck folgende Ziele durchzusetzen:

- Der Approximationsfehler von AdaBoost soll (unter der Voraussetzung (9)) durch  $\varepsilon/2$  beschränkt sein. Hierzu genügt es,

$$T = \left\lceil \frac{\log(2/\varepsilon)}{2\gamma^2} \right\rceil$$



zu setzen, da dann  $e^{-2\gamma^2 T} \leq \varepsilon/2$ .

- Der Schätzfehler von AdaBoost soll mit einer Wahrscheinlichkeit von  $1 - \delta/2$  durch  $\varepsilon/2$  beschränkt sein. Hierzu genügt es,

$$m = m_{D(T,d)}(\varepsilon/2, \delta/2)$$

zu setzen.

**Theorem 10.10** *Es sei WL ein Algorithmus, der  $\mathcal{H}$  mit Hypothesen aus  $\mathcal{B}$   $\gamma$ -schwach lernt. Dann ist AdaBoost (mit der Hilfsprozedur WL und mit den obigen Wahlen der Parameter  $T, m$ ) ein PAC-Lerner für  $\mathcal{H}$  mit Hypothesen aus  $\cup_{T \geq 1} \text{LIN}(\mathcal{B}, T)$ .*