Computational Learning Theory

Wintersemester 2011/12

Herzlich willkommen!

Worum gehts?

Das Lernen von Konzepten

Gegeben ist ein unbekanntes Konzept c und eine Menge $S = \{(x_1, b_1), \ldots, (x_s, b_s)\}$ klassifizierter Beispiele, die gemäß c klassiziert wurden:

- $b_i = 1$ genau dann, wenn $x_i \in c$ und
- $b_i = 0$ genau dann, wenn $x_i \notin c$.

Bestimme eine Hypothese h, die das Konzept c möglichst gut approximiert.

(1) Wieviele Beispiele werden für ein erfolgreiches Lernen benötigt?
(2) Kann eine gute Hypothese in vertretbarer Zeit gefunden werden, wenn genügend viele Beispiele vorhanden sind?
(3) Was bedeutet es überhaupt, dass Hypothese h das Konzept c gut approximiert? Wann können wir behaupten, dass unsere Hypothese vermutlich gut approximiert?

Anwendungsszenarien I

(1) Handschriftenerkennung: Für jeden Anwender x und jede Ziffer $z \in \{0, 1, \ldots, 9\}$ besteht das Konzept $c_{x,z}$ aus allen Schriftproben des Anwenders x, die der Ziffer z entsprechen.

(2) Textinterpretation: Nachrichten sind zum Beispiel automatisch einer der Kategorien „Politik, Wirtschaft, Kultur, Sport ...“ zuzuordnen. Jede dieser Kategorien entspricht einem Konzept.

(3) Bilderkennung: Eine Pixelmatrix ist einer Kategorie (wie etwa „Sonnenaufgang, Wüste, Meer, Stadt, ...“) zuzuordnen.

(4) Bestimmung der Funktion von Proteinen: ein Ähnlichkeitsbegriff ist zu erlernen, so dass die Funktion eines Proteins aufgrund der Ähnlichkeit (zum Beispiel in der Primärstruktur) zu bereits untersuchten Proteinen vorausgesagt werden kann. Proteinfamilien mit ähnlicher Funktion entsprechen Konzepten.

Anwendungszenarien II

(5) Expertenauswahl: Wir müssen eine Folge von Entscheidungen treffen.

- Zu jedem Zeitpunkt geben Experten Empfehlungen ab.
- Nachdem wir, basierend auf allen Empfehlungen, unsere Entscheidung getroffen haben, wird uns die richtige Entscheidung mitgeteilt.

Können wir, sogar ohne den die Entscheidungen betreffenden Sachverhalt zu kennen, eine Strategie entwickeln, deren Entscheidungen fast so gut sind wie die Empfehlungen des bisher besten Experten?

Themen

(1) Grundlagen

(1a) PAC Algorithmen:
Lernalgorithmen, die vermutlich gute Hypothesen liefern.
(PAC = *probably approximately correct*)

(1b) Online Lernen:
Was tun, wenn sofort Hypothesen gebildet werden müssen, ohne dass alle klassifizierten Beispiele bekannt sind (Expertenauswahl)?

(2) Wichtige Lernalgorithmen:
Support Vektor Maschinen, Neuronale Netzwerke, Bayes Lernverfahren, ...

(3) Boosting:
Wie lässt sich das Lernergebnis verbessern?

(4) Aktives Lernen:
Wie stark verbessert sich die Lernleistung, wenn wir eigenständig Fragen stellen dürfen?

Vorwissen

(1) Die Veranstaltung „Algorithmentheorie“ führt die für uns wichtigen Konzepte wie Asymptotik und NP-Vollständigkeit ein.

(2) Die Veranstaltung „Lineare Algebra und Analysis“ ist für die Behandlung von Support Vektor Maschinen von zentraler Bedeutung.

(3) Eine Veranstaltung in elementarer Stochastik ist sehr hilfreich, da wir häufig mit Wahrscheinlichkeiten arbeiten: Zum Beispiel um festzustellen, mit welcher Wahrscheinlichkeit unsere Hypothese eine gute Approximation des Zielkonzepts ist.

Das Kapitel 1 im Skript stellt die wichtigsten Grundlagen vor allem aus der elementaren Stochastik zusammen.

Literatur

(1) Ein Skript wird zur Verfügung gestellt.

(2) M. J. Kearns und U. V. Vazirani, An Introduction to Computational Learning Theory, *MIT Press*, 1994.

(3) N. Cristianini und J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, *Cambridge University Press*, 2000.

(4) T. M. Mitchell, Machine Learning, *McGraw-Hill*, 1997.

Weitere Informationen auf der Webseite der Veranstaltung

Achtung: Raum- und Terminänderung

- Die Vorlesung findet statt
  - Montags um 12:30 im Hörsaal H 10,
  - Dienstags um 16:00 im Hörsaal H 12.
- Übungen finden Mittwochs um 12:15 in der neuen Mensa NM 117 statt: Das erste Treffen ist in zwei Wochen.
  - Übungsblätter werden stets am Montag ausgegeben: Das erste Blatt erscheint am kommenden Montag.
  - Besprechung und Rückgabe stets am darauffolgenden Mittwoch.

BITTE UNBEDINGT AN DEN ÜBUNGEN TEILNEHMEN!

Administratives

Die Veranstaltung wird für die Studiengänge

- Master Informatik,
- Diplomhauptstudium Bioinformatik und Informatik

angeboten.

Prüfungs- und Studienleistungen:

(1) Master Informatik:
Mündliche Prüfung nach Absprache. Das Ergebnis der Prüfung kann durch Übungspunkte um bis zu eine Note verbessert werden.

(2) Ein Leistungsnachweis für das Diplom Informatik/Bioinformatik: bei regelmäßiger, aktiver Teilnahme, wenn mindestens 60% aller Übungspunkte erzielt wurden.

Konzepte, Hypothesen und Konzeptklassen

Sei Σ eine (nicht notwendigerweise endliche) Menge.

- Beispiele und Gegenbeispiele werden mit Worten aus Σ^* bezeichnet.
- Ein Konzept c über Σ ist eine Teilmenge von Σ^*, also eine Menge von Beispielen.
- Eine Konzeptklasse $\mathcal{C}$ über Σ ist eine Menge von Konzepten.
- Eine Hypothese h über Σ ist ein Konzept über Σ und eine Hypothesenklasse $\mathcal{H}$ ist eine Menge von Hypothesen.

Die Lernsituation

- Ein Lernalgorithmus $\mathcal{L}$ versucht ein unbekanntes Zielkonzept c aus einer bekannten Konzeptklasse $\mathcal{C}$ zu erlernen.
- $\mathcal{L}$ erhält klassifizierte Beispiele und bestimmt eine Hypothese $h \in \mathcal{H}$, die c hoffentlich gut approximiert.

Die Konzeptklasse der monotonen Monome

Jedem Beispiel b wird ein Vektor $b = (b_1, \ldots, b_n)$ von n Eigenschaften (mit $b_i \in \{0, 1\}$) zugeordnet.

- Im unbekannten Zielkonzept c werden alle Beispiele gesammelt, die eine unbekannte Teilmenge $I \subseteq \{1, \ldots, n\}$ der n Eigenschaften erfüllen.
- Wir müssen die Menge I dieser kritischen Eigenschaften lernen, also das Konzept

$$c = \{b \in \{0,1\}^n : b_i = 1 \text{ für alle } i \in I\}.$$

- Die Konzeptklasse **MONOTON-MONOM**$_n$ besteht aus der Menge aller monotonen Monome

$$x_{i_1} \wedge x_{i_2} \wedge \cdots \wedge x_{i_k}$$

für Teilmengen $I = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$.

Weitere boolesche Konzeptklassen

- Und wenn wir nur wissen, dass Eigenschaften aus einer unbekannten Teilmenge $I \subseteq \{1, \ldots, n\}$ aller Eigenschaften entweder jeweils erfüllt oder nicht erfüllt werden?

  Jetzt entsprechen die unbekannten Zielkonzepte der Konzeptklasse **MONOM**$_n$ aller Monome mit den Literalen $x_1, \neg x_1, \ldots, x_n, \neg x_n$.

- Darf es ein wenig mehr sein?

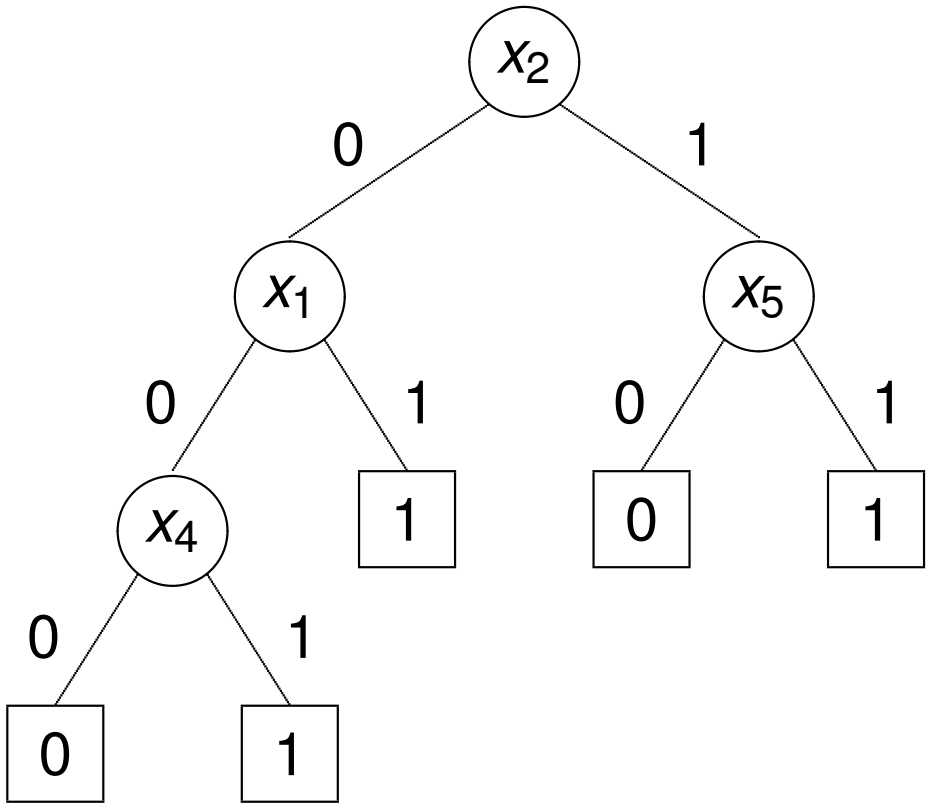
  k-**TERM DNF**$_n$ ist die Konzeptklasse aller DNF-Formeln mit den Literalen $x_1, \neg x_1, \ldots, x_n, \neg x_n$ und höchstens k Monomen.

- Und vielleicht noch ein wenig(?) mehr:

  BOOLEAN$_n$ ist die Konzeptklasse aller booleschen Funktionen $f : \{0,1\}^n \to \{0,1\}$.

Entscheidungsbäume

Die Konzeptklasse **DECISION**$_{n,s}$ besitzt ein Konzept für jeden Entscheidungsbaum (über den Boole'schen Variablen $x_1, \ldots, x_n$) mit höchstens s Knoten.



Automaten / Halbräume

AUTOMAT$_{n,\Sigma}$ besitzt für jeden deterministischen, endlichen Automaten A mit höchstens n Zuständen und dem Eingabealphabet Σ ein Konzept, nämlich die von A akzeptierte Sprache $L(A)$.

HALBRAUM$_n$ besitzt für alle Gewichte $w_1, \ldots, w_n \in \mathbb{R}$ und jeden Schwellenwert (bzw. Threshold) $t \in \mathbb{R}$ den Halbraum

$$\{ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} w_i \cdot x_i \geq t \}$$

als Konzept.

HALBRAUM ist unsere erste reellwertige Kozeptklasse.

Halbräume und neuronale Netzwerke

- Die Konzeptklasse der neuronalen Netzwerke besitzt **HALBRAUM**$_n$ als eine Teilklasse: **Neuronale Netzwerke** haben „neuronale Gatter“ der Form $g(\sum_{i=1}^{n} w_i \cdot y_i - t)$.

  Zum Beispiel das Standard-Sigmoid

$$g(x) = \frac{1}{1 + e^{-x}}.$$

- In der Methode der **Support Vektor Maschinen** wird eine Feature-Funktion ϕ auf die (klassifizierten) Beispiele $b_1, \ldots, b_n$ angewandt. Dann wird nach einer Hyperebene gesucht, die, wenn möglich die positiven Beispiele $\phi(b_i)$ von den negativen Beispielen $\phi(b_j)$ trennt.

Wichtige Fragestellungen

(1) Wieviele Beispiele werden benötigt, um eine Chance zu haben die einzelnen Konzeptklassen erfolgreich zu erlernen?
   - Für $\mathbf{MONOM}_n$ sollten sehr viel weniger Beispiele ausreichen als für $\mathbf{BOOLEAN}_n$.
   - Die notwendige Beispielzahl sollte von der Anzahl der „Freiheitsgrade“ der Konzeptklasse abhängen: Was misst man Freiheitsgrade?

(2) Können wir tatsächlich lernen, wenn genügend viele Beispiele vorhanden sind?

Wann können wir sagen, dass wir hochwahrscheinlich erfolgreich lernen werden?

Fehlermessung

Das Szenario

- Die (klassifizierten) Beispiele werden zufällig, gemäß einer Verteilung D_1, gezogen.
- Der Lernalgorithmus $\mathcal{L}$ berechnet eine Hypothese h.
- Wir messen den Fehler zwischen dem unbekannten Zielkonzept c und der Hypothese h gemäß einer Verteilung D_2:

$$\text{fehler}_{D_2}(c, h) = \sum_{x \in c \oplus h} D_2(x).$$

Fairness-Bedingung: Wenn Beispiele nach der Verteilung D_1 erzeugt werden, dann sollte der Fehler auch gemäß D_1 gemessen werden. Wir fordern deshalb

$$D_1 = D_2.$$

Hochwahrscheinlich approximatives Lernen

Der Anwender gibt einen Fehlerparameter ε vor.

- Der Fehler, also der Unterschied zwischen Zielkonzept und Hypothese, soll höchstens ε betragen.
- Aber wenn der Lernalgorithmus *zufälligerweise* wenig informative Beispiele erhält, dann wird die Hypothese im Allgemeinen nicht gut sein.
- Der Anwender gibt deshalb einen Vertrauensparameter δ vor und fordert:

  Der Fehler darf, mit ***Wahrscheinlichkeit*** *mindestens* $1 - \delta$ *den Fehlerparameter* ε *nicht übersteigen.*

PAC-Algorithmen

PAC Lernen = probably (mit Wahrscheinlichkeit mindestens $1 - \delta$) approximately (mit Fehler höchstens ε) correct.

- Wir möchten also hochwahrscheinlich approximativ korrekt lernen.
- Für welche Verteilungen D? Für alle!

Ein Lernalgorithmus $\mathcal{L}$ ist genau dann ein PAC-Algorithmus für eine Konzeptklasse $\mathcal{C}$, wenn **für alle** $\varepsilon \in (0, 1]$, **für alle** $\delta \in (0, 1]$, **für alle Konzepte** $c \in \mathcal{C}$ und **für alle Verteilungen** D

$$\text{prob}_D[\ \text{fehler}_D(c, h_c) \leq \varepsilon\ ] \geq 1 - \delta$$

gilt, wobei h_c die von $\mathcal{L}$ bestimmte Hypothese ist.

Die Form eines PAC-Algorithmus

Die Grundstruktur eines PAC-Algorithmus

(0) Die Eingabe besteht aus dem Vertrauensparameter δ und dem Fehlerparameter ε.

(1) Bestimme $s = s(\delta, \varepsilon)$, die Anzahl der anzufordernden Beispiele.

(2) Wiederhole s mal:

Fordere ein klassifiziertes Beispiel (x, b) an.
/* Es ist stets $b \in \{0, 1\}$ und weiterhin gilt $b = 1$ genau dann, wenn x zum zu lernenden Konzept c gehört. */

(3) Bestimme eine Hypothese $h_c \in \mathcal{H}$.

Die zentralen Fragen I

- Fordern wir nicht zuviel? Unser Algorithmus muss die Vorgaben an Fehler und Verläßlichkeit für alle ε, δ, D und alle Konzepte einhalten!

  In Anwendungen achtet man darauf, dass die Trainingsmenge aus möglichst informativen Beispielen besteht: Die Beispielverteilung D_1 ist dann sogar unterstützend.

- Andererseits wäre die Existenz von PAC-Algorithmen $\mathcal{L}$ klasse, denn wir können $\mathcal{L}$ mit einem Gütesiegel versehen:

  Wie auch immer die uns nicht bekannte Verteilung D aussieht, falls $\mathcal{L}$ mindestens $s(\varepsilon, \delta)$ Beispiele erhält, ist approximativ korrektes Lernen hochwahrscheinlich ***garantiert****.*

Die zentralen Fragen II

- Wie groß ist die Beispielzahl $s(\varepsilon, \delta)$ in Abhängigkeit vom erlaubten Fehler ε und vom Vertrauensparameter δ?
- Angenommen, $\mathcal{L}$ ist ein PAC-Algorithmus mit Fehler ε and Vertrauensparameter δ.
  - Um wieviel steigt die Beispielzahl und Laufzeit an, wenn δ durch δ' mit $0 < \delta' \leq \delta$ ersetzt wird?
  - Um wieviel steigt die Beispielzahl und Laufzeit an, wenn ε durch ε' mit $0 < \varepsilon' \leq \varepsilon$ ersetzt wird?

Ein Lernalgorithmus für $\mathbf{MONOM}_n$

(1) Bestimme $s = s_n(\varepsilon, \delta)$ und fordere s Beispiele an.

(2) Die erste vorläufige Hypothese ist das Monom

$$h \equiv x_1 \wedge \neg x_1 \wedge x_2 \wedge \neg x_2 \wedge \cdots \wedge x_n \wedge \neg x_n.$$

(3) Streiche alle den positiven Beispielen widersprechende Literale.

(4) Gib die resultierende Hypothese aus.

h ist mit allen Beispielen konsistent

(1) Das Literal x_j (bzw. $\neg x_j$) wird entfernt, falls ein positives Beispiel diesem Literal widerspricht.

(2) Wir bestimmen somit das längste mit allen positiven Beispielen konsistente Monom h.

(3) Ist y ein negatives Beispiel und ist M das unbekannte Zielmonom, dann
   - gibt es i, so dass das Literal x_i (bzw. $\neg x_i$) in M vorkommt,
   - aber x_i (bzw. $\neg x_i$) verwirft das negative Beispiel y.

   Da h das längste mit allen positiven Beispielen konsistente Monom ist, kommt x_i (bzw. $\neg x_i$) in h vor und h verwirft y.

Wir zeigen jetzt, dass unser Algorithmus ein PAC-Algorithmus für die Konzeptklasse MONOM_n ist, falls

$$s_n(\varepsilon, \delta) = \lceil \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right) + \frac{\ln(3^n)}{\varepsilon} \rceil.$$

PAC-Algorithmen für endliche Konzeptklassen

Die Eingabe besteht aus dem Vertrauensparameter δ und dem Fehlerparameter ε.

(1) Setze $s = \lceil \frac{1}{\varepsilon} \cdot \left(\ln |\mathcal{C}| + \ln\left(\frac{1}{\delta}\right)\right) \rceil$.

(2) Wähle *irgendein* Konzept $h \in \mathcal{C}$ als Hypothese, das mit allen s Beispielen konsistent ist.
/* Eine Hypothese h ist genau dann **konsistent**, wenn h alle positiven Beispiele akzeptiert und alle negativen Beispiele verwirft.

- Für MONOM_n genügen, wie versprochen, $\lceil \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right) + \frac{\ln(3^n)}{\varepsilon} \rceil$ Beispiele.
- Beachte, dass wir irgendeine konsistente Hypothese auswählen! Kann das zu einem Problem werden?

Die Analyse I

Behauptung

Wir erhalten einen PAC-Algorithmus, der $s = \lceil \frac{1}{\varepsilon} \cdot \left(\ln |\mathcal{C}| + \ln\left(\frac{1}{\delta}\right)\right) \rceil$ Beispiele anfordert.

- Wir müssen für jedes Konzept $c \in \mathcal{C}$ und jede Verteilung D

$$\text{prob}_D[\ \text{fehler}_D(c, h_c) \leq \varepsilon\ ] \geq 1 - \delta$$

nachweisen.

- Sei $p = \text{prob}_D[\ \text{fehler}_D(c, h_c) > \varepsilon\ ]$. Dann ist

$$p \leq \text{prob}[\ \text{Es gibt eine konsistente Hypothese } h \text{ mit } \text{fehler}_D(c, h) > \varepsilon\ ]$$

$$\leq \sum_{h \in \mathcal{C} \text{ mit } \text{fehler}_D(c,h) > \varepsilon} \text{prob}[\ h \text{ ist konsistente Hypothese }].$$

Die Analyse II

- Wenn $h \in \mathcal{C}$ eine Hypothese mit großem Fehler ist (also $\text{fehler}_D(c, h) > \varepsilon$), dann wird h ein zufällig gezogenes Beispiel mit Wahrscheinlichkeit höchstens $1 - \varepsilon$ richtig klassifizieren.
- Also ist

$$\begin{aligned} p &\leq \sum_{h \in \mathcal{C} \text{ mit } \text{fehler}_D(c,h) > \varepsilon} \text{prob}[h \text{ ist eine konsistente Hypothese }] \\ &\leq \sum_{h \in \mathcal{C} \text{ mit } \text{fehler}_D(c,h) > \varepsilon} (1-\varepsilon)^s \leq \sum_{h \in \mathcal{C}} (1-\varepsilon)^s \\ &= |\mathcal{C}| \cdot (1-\varepsilon)^s. \end{aligned}$$

- Es ist $p = \text{prob}_D[\ \text{fehler}_D(c, h_c) > \varepsilon\ ]$ und wir müssen $p \leq \delta$ fordern!

Die Analyse III

- Wir müssen

$$|\mathcal{C}| \cdot (1-\varepsilon)^s \leq \delta$$

  erfüllen.
- Logarithmiere beide Seiten der Ungleichung:

$$\ln|\mathcal{C}| + s \cdot \underbrace{\ln(1-\varepsilon)}_{\leq -\varepsilon} \leq \ln\delta.$$

  (Es ist $\ln(1-\varepsilon) \leq -\varepsilon$. Warum?
- Somit reicht der Nachweis von $\ln|\mathcal{C}| - s \cdot \varepsilon \leq \ln\delta$. Diese Ungleichung ist äquivalent zu

$$\frac{1}{\varepsilon}(\ln|\mathcal{C}| - \ln\delta) \leq s$$

  und das war zu zeigen.

Diskussion

Die zentrale „Idee“: Wähle eine **konsistente** Hypothese!

Kann eine konsistente Hypothese aber auch effizient bestimmt werden?

Die Beispielzahl $s = \lceil \frac{1}{\varepsilon} \cdot \left(\ln |\mathcal{C}| + \ln\left(\frac{1}{\delta}\right)\right) \rceil$:

- Eine große Verläßlichkeit, also ein kleines δ, kann billig erzielt werden, denn die Beispielanzahl wächst nur logarithmisch mit $1/\delta$.
- Ein kleiner Fehler, also ein kleines ε, ist teuer, denn $1/\varepsilon$ geht linear in der Beispielanzahl ein.

Wir haben $\ln |\mathcal{C}|$ als „Definition“ der Anzahl der Freiheitsgrade gewählt. Ist das die richtige Interpretation?

Weitere Anwendungen

(a) Beispiel(MONOTON-MONOM$_n$) $= O(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$,

Beispiel(MONOM$_n$) $= O(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(b) Beispiel(k-KNF$_n$) $= O(\frac{n^k}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$ für festes k,

Beispiel(k-DNF$_n$) $= O(\frac{n^k}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$ für festes k.

(c) Beispiel(k-KLAUSEL-KNF$_n$) $= O(\frac{k \cdot n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$,

Beispiel(k-TERM-DNF$_n$) $= O(\frac{k \cdot n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(d) Beispiel(KNF$_n$) $= O(\frac{1}{\varepsilon} \cdot 2^n + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$,

Beispiel(DNF$_n$) $= O(\frac{1}{\varepsilon} \cdot 2^n + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$,

Beispiel(BOOLEAN$_n$) $= O(\frac{1}{\varepsilon} \cdot 2^n + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(e) Beispiel(SYM$_n$) $= O(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(f) Beispiel(AUTOMAT$_{n,\Sigma}$) $= O(\frac{n}{\varepsilon} \cdot |\,\Sigma\,| \cdot \log_2 n + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

Kritik

- Für praktische Anwendungen können wir nicht davon ausgehen, dass jedes Zielkonzept auch in der Hypothesenklasse liegt:

  Die Schriftproben einer bestimmten Person werden wir nur approximativ, aber nicht exakt erkennen können!

- Stattdessen nehmen wir an, dass es zu jedem Zielkonzept $c \in \mathcal{C}$ eine Hypothese $h \in \mathcal{H}$ mit kleinem Fehler gibt, d.h. es gelte

$$\text{fehler}_D(c, h) \leq \varepsilon.$$

Der Lernalgorithmus:

(1) Bestimme die Beispielzahl s in Abhängigkeit von ε und δ.

(3) Wähle *irgendeine* Hypothese $h \in \mathcal{H}$, die mit mindestens $(1 - 2\varepsilon) \cdot s$ Beispielen konsistent ist. Gib eine Fehlermeldung aus, wenn eine solche Hypothese nicht existiert.

Das Ziel

Bestimme die Beispielzahl s so, dass

$$\text{prob}_D[\ \begin{array}{l}\text{Algorithmus findet eine Hypo–}\\ \text{these } h_c \text{ mit fehler}_D(c, h_c) \leq 4\varepsilon\end{array}\ ] \geq 1 - \delta$$

gilt.

- Wir haben jetzt zwei Fehlerquellen: Entweder wir finden keine „fast konsistente“ Hypothese h (Fehlermeldung) oder der Fehler ist zu groß.
- Bestimme s, so dass

$$\begin{aligned}\text{prob}_D[\ \text{Algorithmus gibt eine Fehlermeldung}\ ] &\leq \frac{\delta}{2} \text{ und}\\ \text{prob}_D[\ \text{fehler}_D(c, h_c) \geq 4\varepsilon\ ] &\leq \frac{\delta}{2}\end{aligned}$$

Die Wahrscheinlichkeit einer Fehlermeldung I

Die Situation

Es gibt eine Hypothese $h \in \mathcal{H}$ mit $\text{fehler}_D(c, h) \leq \varepsilon$, aber es gibt keine Hypothese, die mit mindestens $(1 - 2\varepsilon) \cdot s$ Beispielen konsistent ist.

- Insbesondere ist h also auch mit mehr als $2\varepsilon \cdot s$ Beispielen inkonsistent.
- Wenn wir aber s Beispiele zufällig ziehen, dann ist die erwartete Anzahl durch h falsch klassifizierter Beispiele höchstens

$$\varepsilon \cdot s.$$

- Im Falle einer Fehlermeldung verstoßen wir signifikant gegen den Erwartungswert: Statt Inkonsistenz mit höchstens $\varepsilon \cdot s$ Beispielen sogar Inkonsistenz mit mehr als $2\varepsilon \cdot s$ Beispielen.
- Ein heftiger Verstoß gegen den Erwartungswert sollte unwahrscheinlich sein!

Die Wahrscheinlichkeit einer Fehlermeldung II

Wir liegen mit $2\varepsilon s$ zweifach über dem Erwartungswert und die Chernoff-Schranke für $\beta = 1$ liefert

$$e^{-\frac{\varepsilon s}{3}}.$$

als Wahrscheinlichkeit einer Fehlermeldung.

- Wir müssen die Wahrscheinlichkeit einer Fehlermeldung durch $\delta/2$ nach oben beschränken.
- Wir fordern

$$s \geq \frac{3}{\varepsilon} \cdot \ln\left(\frac{2}{\delta}\right).$$

Zu große Fehler I

- Sei $h \in \mathcal{H}$ eine beliebige Hypothese mit zu großem Fehler, also $\text{fehler}_D(c, h) \geq 4 \cdot \varepsilon$.
- Ist es nicht dann unwahrscheinlich, dass h mit höchstens $2\varepsilon \cdot s$ Beispielen inkonsistent ist?

  Mehr als $4\varepsilon \cdot s$ inkonsistente Beispiele sind zu erwarten.
- Chernoff muss ran und zwar mit $\beta = 1/2$.

$$p = \text{prob}_D[\ \begin{array}{l} h \text{ ist mit höchstens } 2\varepsilon s \\ \text{Beispielen nicht konsistent} \end{array}\ ] \leq e^{-\frac{4\varepsilon \cdot s}{4 \cdot 2}} = e^{-\frac{\varepsilon \cdot s}{2}}$$

Zu große Fehler II

Für die Wahrscheinlichkeit q eines zu großen Fehlers gilt:

$$q \leq \sum_{h \in \mathcal{H} \text{ mit } \text{fehler}_D(c,h) \geq 4\varepsilon} \text{prob}_D[\ \begin{array}{l} h \text{ ist mit} \leq 2\varepsilon s \text{ Beispielen} \\ \text{nicht konsistent} \end{array}\ ]$$

- Also

$$q \leq \sum_{h \in \mathcal{H}} e^{-\frac{\varepsilon s}{2}} = \mid \mathcal{H} \mid \cdot e^{-\frac{\varepsilon s}{2}}.$$

- Um $q \leq \delta/2$ zu erhalten, fordere $\mid \mathcal{H} \mid \cdot e^{-\frac{\varepsilon s}{2}} \leq \delta/2$.
- Nach logarithmieren folgt $\ln \mid \mathcal{H} \mid - \frac{\varepsilon s}{2} \leq \ln\left(\frac{\delta}{2}\right)$ und äquivalent

$$\frac{2}{\varepsilon} \cdot \left[\ln \mid \mathcal{H} \mid + \ln\left(\frac{2}{\delta}\right)\right] \leq s.$$

Zusammenfassung

Wir erreichen $\text{fehler}_D(c, h) \leq 4\varepsilon$ mit Wahrscheinlichkeit mindestens $1 - \delta$, falls es eine Hypothese h' mit $\text{fehler}_d(c, h') \leq \varepsilon$ gibt und falls

$$s(\delta, \varepsilon) \geq \frac{3}{\varepsilon} \cdot \left[\ln | \mathcal{H} | + \ln \left(\frac{2}{\delta} \right) \right].$$

Die Beispielzahl steigt, im Vergleich zur garantierten Existenz konsistenter Lösungen nur moderat an!

Welche Fragen bleiben?

- Sind die Schranken für die Beispielzahl scharf?

  Sind $\ln(|\mathcal{C}|)$, bzw. $\ln(|\mathcal{H}|)$ gute Messungen der Anzahl der Freiheitsgrade von Konzept- bzw. Hypothesenklasse?
- Wie bestimmt man (fast-)konsistente Hypothesen?

Occam's Razor

„Non sunt multiplicanda entia praeter necessitatem“.
In freier Übersetzung:

„Wähle stets eine möglichst einfache Erklärung“.

Was ist eine **einfache** Erklärung (bzw. Hypothese)?

- Wir weisen jeder Hypothese eine Beschreibung zu.
- Eine möglichst einfache Hypothese ist eine Hypothese mit möglichst kurzer Beschreibung.

Die Konzeptklasse MONOM_n

Sei m ein Monom mit

$$m = l_1 \wedge l_2 \wedge \cdots \wedge l_k.$$

$l_1, \ldots, l_k$ sind Literale mit $l_j \in \{x_{i_j}, \neg x_{i_j}\}$.

- Beschreibe m durch die Binärdarstellung

$$b_1 \cdot \text{code}(i_1) \cdot b_2 \cdot \text{code}(i_2) \cdots b_k \cdot \text{code}(i_k).$$

  - $\text{code}(i)$ ist die Binärdarstellung der Zahl $i \in \{0, \ldots, n-1\}$ mit führenden Nullen auf $\lceil \log_2 n \rceil$ Bits aufgefüllt.
  - $b_j \cdot \text{code}(i_j)$ gibt an, ob $l_j = x_{i_j}$ ($b_j = 0$) bzw. $l_j = \neg x_{i_j}$ ($b_j = 1$).
- Die Beschreibungslänge ist $k\left(\lceil \log_2 n \rceil + 1\right)$.

Unser PAC-Algorithmus hat ein längstes konsistentes Monom bestimmt. War das keine gute Idee?

Occam Algorithmen I

Der Setup

- Für $h \in \mathcal{H}$ sei länge(h) die Länge des Namens von h.
- $\mathcal{H}_r$: alle Hypothesen in $\mathcal{H}$ mit Namenslänge kleiner als r.
- Für welche Beispielzahl s können wir ein Gütesiegel vergeben, wenn wir eine konsistente Hypothese in $\mathcal{H}_r$ finden?

- Es ist $| \mathcal{H}_r | \leq \sum_{i=0}^{r-1} 2^i = 2^r - 1$.
- Sei $p_{s,r}$ die Wahrscheinlichkeit, dass es eine Hypothese $h \in \mathcal{H}_r$ gibt, die mit s Beispielen konsistent ist und einen Fehler größer als ε besitzt.

Dann ist

$$p_{s,r} \leq (1-\varepsilon)^s \cdot | \mathcal{H}_r | \leq (1-\varepsilon)^s \cdot 2^r.$$

Occam Algorithmen II

(1) Wir möchten $p_{s,r} \leq \delta$ erreichen und müssen fordern

$$(1-\varepsilon)^s \cdot 2^r \leq \delta.$$

(2) Nach Logarithmieren ist dies äquivalent zu:

$$s \cdot \ln(1-\varepsilon) + r \cdot \ln 2 \leq \ln \delta.$$

(3) Aber $\ln(1-\varepsilon) \leq -\varepsilon$ mit der Mutter aller Ungleichungen. Fordere

$$-s \cdot \varepsilon + r \cdot \ln(2) \leq \ln(\delta).$$

Wenn für

$$s = \frac{1}{\varepsilon} \cdot r \cdot \ln 2 + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right)$$

Beispiele eine Hypothese h der Länge $< r$ gefunden wird, dann hat h mit Wahrscheinlichkeit $\geq 1-\delta$ einen Fehler $\leq \varepsilon$.

Fazit

Erfolgreiches Lernen für Hypothesenlänge $< r$ und

$$s = \frac{1}{\varepsilon} \cdot r \cdot \ln 2 + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right) \quad \text{Beispiele.}$$

- Selbst sehr große Hypothesenklassen erlauben erfolgreiches Lernen, **wenn** eine Hypothese gefunden wird, die wesentlich kürzer als die Anzahl der Beispiele ist.
- Kurze Hypothesen komprimieren die Beispiele!

Kurze Hypothesen lernen nicht nur die Trainingsmenge auswendig, sondern sie verallgemeinern erfolgreich.

Ein Occam Algorithmus für MONOM_n

Sei S die Menge der klassifizierten Beispiele.

- Wir haben das längste mit S konsistente Monom gefunden.
- Wir möchten ein kurzes mit S konsistentes Monom bestimmen.

- Sei S_- (bzw. S_+) die Menge aller negativen (bzw. positiven) Beispiele.
- Sei L die Menge aller Literale, die von allen positiven Beispielen erfüllt werden. Dann ist

$$\wedge_{l \in L} l$$

  das längste mit S konsistente Monom.
- Die Idee: Versuche alle negativen Beispiele mit möglichst wenigen Literalen aus L auszuschalten.
  Diese möglichst wenigen Literale bilden dann unsere Hypothese.

Das Überdeckungsproblem I

- Für jedes I in L ist $\text{negativ}(I) = \{ y \in S_- \mid I(y) = 0 \}$ die Menge der von I ausgeschalteten negativen Beispiele in S_-.
- Es ist $S_- = \bigcup_{I \in L} \text{negativ}(I)$ und wir suchen eine möglichst kleine Menge $L' \subseteq L$ mit

$$S_- = \bigcup_{I \in L'} \text{negativ}(I).$$

Das Überdeckungsproblem

- Ein Universum U und Teilmengen $A_1, \ldots, A_n \subseteq U$ mit $A_1 \cup A_2 \cup \cdots \cup A_n = U$ sind gegeben.
- Wir suchen eine Menge $I \subseteq \{1, \ldots, n\}$ kleinster Größe, so dass

$$\bigcup_{i \in I} A_i = U.$$

Das Überdeckungsproblem II

- Die schlechten Nachrichten: Das Überdeckungsproblem führt auf ein $\mathcal{NP}$-vollständiges Problem.
- Die guten Nachrichten: Es gibt einen effizienten Greedy-Algorithmus:

(1) $I = \emptyset$.

(2) WHILE $U \neq \emptyset$ DO

  (2a) Wähle eine größte Menge A_i und setze $I = I \cup \{i\}$ sowie $U = U - A_i$.

  (2b) Entferne alle Elemente von A_i aus den Mengen $A_1, \ldots, A_n$, d.h. A_j wird durch $A_j - A_i$ ersetzt.

(3) Gib die Indexmenge I aus.

Das Überdeckungsproblem III

Wenn eine kleinste Überdeckung opt Mengen benötigt, dann benötigt der gierige Überdeckungsalgorithmus höchstens

$$\text{opt} \cdot (1 + \ln(|U|)) \text{ Mengen.}$$

(1) Wenn wir s Beispiele anfordern und wenn das Zielmonom aus höchstens k Literalen besteht, dann finden wir eine konsistente Hypothese mit höchstens $k' = k \cdot (1 + \ln(s))$ Literalen.

(2) Wir erhalten für die Beschreibungslänge

$$\begin{aligned} r &\leq k \cdot (\lceil \log_2 n \rceil + 1) \cdot (1 + \ln(s)) \\ &= \Theta(k \cdot \log_2 n \cdot \log_2 s). \end{aligned}$$

Das Ergebnis

Wir wissen:

Erfolgreiches Lernen für Hypothesenlänge $< r$ und

$$s = \frac{1}{\varepsilon} \cdot r \cdot \ln 2 + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right) \quad \text{Beispiele.}$$

Weiterhin ist $r = \Theta(k \cdot \log_2 n \cdot \log_2 s)$.

Wenn das Zielmonom höchstens k Literale besitzt, dann genügen

$$\Theta(\frac{1}{\varepsilon} \cdot k \cdot \log_2 n \cdot \log_2(\frac{1}{\varepsilon} \cdot n) + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right)) \quad \text{Beispiele,}$$

während unser PAC-Algorithmus mit $\Theta(\frac{1}{\varepsilon} \cdot n + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$ Beispielen arbeitet.

Zwischenfazit

Was haben wir erreicht?

- Wir können ein Gütesiegel

  (hochwahrscheinlich ein kleiner Fehler)

  an (fast-)konsistente Hypothesen vergeben, wenn $\Omega(\frac{1}{\varepsilon} \cdot (\ln(|\mathcal{H}|) + \ln(\frac{1}{\delta})))$ Beispiele angefordert werden.
- Occam-Algorithmen erlauben eine Vergabe des Gütesiegels mgl. bei sehr kleiner Beispielmenge.

Ja, aber

- werden auch so viele Beispiele tatsächlich benötigt ?
- Was machen wir mit reellwertigen Hypothesenklassen wie HALBRAUM$_n$
- und was sind unsere Chancen, (fast)-konsistente Hypothesen effizient berechnen zu können?

Wie misst man Freiheitsgrade?

Sei $\mathcal{C}$ eine Konzeptklasse

(1) $\mathcal{C}$ *zertrümmert* eine Menge S von Beispielen, falls

$$\mathcal{P}(S) = \{S \cap c \mid c \in \mathcal{C}\}.$$

($\mathcal{P}(S) = \{T \mid T \subseteq S\}$ ist die Potenzmenge von S.)

(2) Die VC-Dimension $\mathrm{VC}(\mathcal{C})$ ist die Mächtigkeit der größten Menge S, die von $\mathcal{C}$ zertrümmert wird.

Wenn $\mathcal{C}$ die Menge S zertrümmert, dann erhalten wir jede Teilmenge von S als den Durchschnitt von S mit einem Konzept in $\mathcal{C}$:

Wenn $\mathcal{C}$ die Menge S zertrümmert, dann müssen wir die Klassifikation eines jeden Beispiels in S verlangen, um eine vernünftige Vermutung über das Zielkonzept formulieren zu können.

Die Anzahl der Freiheitsgrade

Wir definieren $VC(\mathcal{C})$ als die Anzahl der Freiheitsgrade von $\mathcal{C}$.
Ist diese Definition vernünftig?

BOOLEAN$_n$ besitzt ein Konzept c_f für jede Boolesche Funktion $f : \{0,1\}^n \to \{0,1\}$:

$$c_f = \{x \in \{0,1\}^n \mid f(x) = 1\}.$$

- **BOOLEAN**$_n$ „sollte“ 2^n Freiheitsgrade besitzen, da der Wert von f auf jedem Argument $x \in \{0,1\}^n$ bestimmt werden muss.
- **BOOLEAN**$_n$ zertrümmert $\{0,1\}^n$, denn es gibt für jede Teilmenge $T \subseteq \{0,1\}^n$ ein Konzept $c \in$ **BOOLEAN**$_n$ mit $c = T$.
- Also ist VC(**BOOLEAN**$_n$) $= 2^n$.

Die Konzeptklasse RECHTECK$_n$

RECHTECK$_n$ besitzt für jedes achsenprallele Rechteck $R \subseteq \mathbb{R}^n$ ein Konzept, das aus allen Punkten in R besteht.

(1) Die Anzahl der Freiheitsgrade von **RECHTECK$_2$** „sollte“ vier sein, denn jedes Rechteck wird durch die vier Koordinaten diagonal gegenüber liegender Eckpunke beschrieben.

(2) Die vier Punkte

$$e_1 = (-1, 0), \quad e_2 = (0, 1), \quad e_3 = (0, -1), \quad e_4 = (1, 0)$$

werden von RECHTECK$_2$ zertrümmert und VC(RECHTECK$_2$) ≥ 4 folgt.

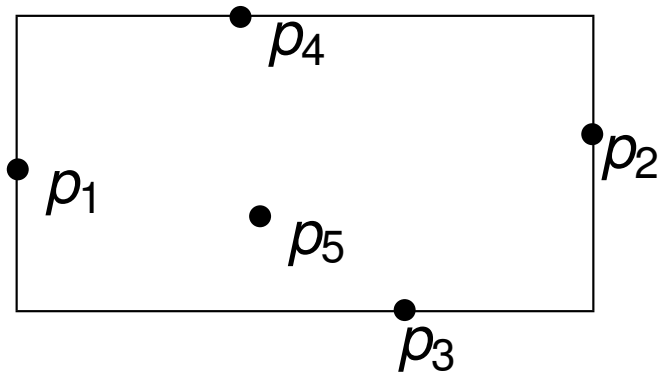
$\mathrm{VC}(\mathrm{RECHTECK}_2) = 4$

Wir zeigen $\mathrm{VC}(\mathbf{RECHTECK}_2) \leq 4$.

(1) Angenommen, $\mathbf{RECHTECK}_2$ zertrümmert die fünf Punkte p_1, p_2, p_3, p_4, p_5.

(2) p_1 und p_2 seien die Punkte mit minimaler (bzw. maximaler) x-Koordinate. p_3 und p_4 seinen die Punkte mit minimaler (bzw. maximaler) y-Koordinate.

(3) Dann kann die Teilmenge $T = \{p_1, p_2, p_3, p_4\}$ aber nicht von p_5 getrennt werden!



HALBRAUM$_n$

Jedes Konzept in **HALBRAUM**$_n$ besteht aus allen Punkten $x \in \mathbb{R}$, die die Ungleichung $\sum_{i=1}^{n} w_i \cdot x_i \geq t$ erfüllen.

HALBRAUM$_n$ „sollte“ $n+1$ Freiheitsgrade besitzen: Jedes Konzept wird durch die n Gewichte und den Schwellenwert beschrieben.

- **HALBRAUM**$_n$ zertrümmert die $n+1$ Punkte $0, e_1, \ldots, e_n$, wobei e_i der $i$te Einheitsvektor ist. Warum?
  - Sei $T \subseteq \{0, e_1, \ldots, e_n\}$ eine beliebige Teilmenge.
  - Wenn $0 \notin T$, wähle c als den Halbraum $\sum_{i=1}^{n} w_i \cdot x_i \geq 1$ mit $w_i = 1$, wenn $e_i \in T$, und $w_i = -1$, wenn $e_i \notin T$.
  - Wenn $0 \in T$, dann arbeite mit den gleichen Gewichten, setze aber den Schwellenwert auf 0.
- Also ist VC(HALBRAUM$_n$) $\geq n+1$.

$\mathrm{VC}(\mathrm{HALBRAUM}_n) \leq n+1$

- Sei S eine Menge von $n+2$ Punkten.
- Wie beschafft man sich eine Teilmenge $T \subseteq S$, die nicht von ihrem Komplement $S \setminus T$ getrennt werden kann?

Der Satz von Radon

Für jede Menge $S \subseteq \mathbb{R}^n$ mit $|S| \geq n+2$ gibt es eine nicht-leere Teilmenge $T \subset S$, so dass

$$\text{konvexe Hülle}(T) \cap \text{konvexe Hülle}(S \setminus T) \neq \emptyset.$$

Zum Beispiel: Vier Punkte im $\mathbb{R}^2$ besitzen stets eine Teilmenge T, die nicht von ihrem Komplement weggetrennt werden kann.

$\mathrm{VC}(\mathrm{HALBRAUM}_n) \leq n + 1$

Angenommen, $\sum_{i=1}^{n} w_i \cdot x_i \geq t$ für alle Punkte in T **und** $\sum_{i=1}^{n} w_i \cdot x_i < t$ für alle Punkte in $S \setminus T$.

(1) Setze

$$H_1 = \{ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} w_i x_i \geq t \}, \quad H_2 = \{ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} w_i x_i < t \}.$$

(2) Dann $H_1 \cap H_2 = \emptyset$.

(3) Aber $T \subseteq H_1$ und $S \setminus T \subset H_2$ nach Annahme.

(4) Wir erhalten einen Widerspruch zum Satz von Radon, denn H_1 und H_2 sind konvex.

$\mathrm{VC}(\mathrm{HALBRAUM}_n) \leq n + 1$.

Kreise und konvexe Polygone

(a) Die Konzeptklasse **KREIS** besitzt für jeden Kreis K in $\mathbb{R}^2$ ein Konzept. Für den Kreis mit Mittelpunkt x und Radius r besteht das entsprechende Konzept aus allen Punkten $y \in \mathbb{R}^2$ mit $||x - y|| \leq r$. Es ist $\mathrm{VC}(\textbf{KREIS}) = 3$.

(b) Die Konzeptklasse **k-GONS** ist die Klasse der konvexen Polygone in $\mathbb{R}^2$, die *höchstens k* Ecken besitzen. Das Konzept besteht aus allen Punkten der Ebene, die innerhalb des Polygons liegen. Es ist $\mathrm{VC}(\textbf{k-GONS}) = 2k + 1$.

- Der Beweis wird in den Übungen behandelt.
- Das Ergebnis für **k-GONS** ist auf den ersten Blick überraschend, da ein k-gon durch k Punkte mit jeweils zwei Koordinaten festgelegt wird: Aber es werden nur höchstens k Ecken gefordert.

Einfache Eigenschaften der VC-Dimension

(a) Wenn $\mathcal{C}_1 \subseteq \mathcal{C}_2$, dann ist $\mathrm{VC}(\mathcal{C}_1) \leq \mathrm{VC}(\mathcal{C}_2)$.

(b) $\overline{\mathcal{C}} = \{ \overline{c} \mid c \in \mathcal{C} \}$ ist die Komplement-Konzeptklasse von $\mathcal{C}$. Dann gilt $\mathrm{VC}(\mathcal{C}) = \mathrm{VC}\left(\overline{\mathcal{C}}\right)$.

(c) Für jede endliche Konzeptklasse $\mathcal{C}$ ist $\mathrm{VC}(\mathcal{C}) \leq \lfloor \log_2 |\mathcal{C}| \rfloor$.

Zu Teil (c):

- Angenommen, $\mathcal{C}$ zertrümmert eine Beispielmenge S.
- Dann gibt es zu jeder Teilmenge $T \subseteq S$ ein Konzept $c_T \in \mathcal{C}$ mit $T = S \cap c_T$.
- Also ist die Anzahl der Teilmengen von S höchstens so groß wie die Anzahl der Konzepte in $\mathcal{C}$ und $2^{|S|} \leq |\mathcal{C}|$ folgt.
- Aber dann $\mathrm{VC}(\mathcal{C}) \leq \lfloor \log_2 |\mathcal{C}| \rfloor$.

Die VC-Dimension wichtiger Konzeptklassen

(a) $\mathrm{VC}(\text{MONOTON-MONOM}_n) = n$.

(b) $\mathrm{VC}(\text{MONOM}_1) = 2$ und $\mathrm{VC}(\text{MONOM}_n) = n$ für $n \geq 2$.

(c) $\binom{n}{k} \leq \mathrm{VC}(k\text{-KNF}_n) = \mathrm{VC}(k\text{-DNF}_n) \leq \binom{n}{k} \cdot 2^k$.

(d) $\mathrm{VC}(k\text{-KLAUSEL-KNF}_n) = \mathrm{VC}(k\text{-Term-DNF}_n) = \Theta(k \cdot n)$ für $k \leq 2^{n/2}$.

(e) $\mathrm{VC}(\text{BOOLEAN}_n) = 2^n$.

(f) $\mathrm{VC}(\text{SYM}_n) = n + 1$.

(g) $\mathrm{VC}(\text{AUTOMAT}_{n,\Sigma}) = \Theta(n \cdot |\Sigma| \cdot \log_2 n)$.

(h) $\mathrm{VC}(\text{RECHTECK}_n) = 2n$.

(i) $\mathrm{VC}(\text{HALBRAUM}_n) = n + 1$.

Die VC-Dimension von SYM_n

- Eine boolesche Funktion $f : \{0,1\}^n \to \{0,1\}$ heißt symmetrisch, wenn der Wert $f(x)$ nur von der Anzahl der Einsen in x abhängt.
- $\mathbf{SYM}_n$ besitzt für jede symmetrische Funktion f das Konzept $c_f = \{x \mid f(x) = 1\}$.
- Behauptung: $\text{VC}(\text{SYM}_n) = n + 1$.

- Die Anzahl der symmetrischen Funktionen f stimmt mit der Anzahl der Funktionen $g : \{0, \ldots, n\} \to \{0,1\}$ überein.
- Also ist $|\text{SYM}_n| = 2^{n+1}$ und $\text{VC}(\text{SYM}_n) \leq n + 1$ folgt.
- Der Vektor e_i habe Einsen in den Positionen $1, \ldots, i$ und sonst nur Nullen. Wir zertrümmern $S = \{e_0, \ldots, e_n\}$ mit $e_0 = 0$.
  - Für $T \subseteq S$ definiere die symmetrische Funktion f mit

$$f(e_i) = \begin{cases} 1 & e_i \in T \\ 0 & \text{sonst,} \end{cases}$$

  - Dann ist $S \cap c_f = T$.

Wieviele Beispiele sind notwendig: Intervalle I

Die Konzeptklasse **INTERVALL** besteht aus allen Intervallen $I \subseteq [0, 1]$. Wieviele Beispiele sind notwendig, um Intervalle mit Fehler höchstens ε zu lernen?

- **INTERVALL** besteht aus überabzählbar unendlich vielen Konzepten. Unsere $\ln |\mathcal{C}|$-Schranke versagt völlig.
- Angenommen, unser Lernalgorithmus gibt das kleinste Intervall, das alle positiven Beispiele enthält, als Hypothese aus.

Was passiert für das Zielkonzept $I = [0, 1]$?

Wieviele Beispiele sind notwendig: Intervalle II

- Unter der Annahme der Gleichverteilung: Wenn wir ein Beispiel x zufällig ziehen, dann ist $x \in [0, \varepsilon]$ mit Wahrscheinlichkeit ε.
- Ein Fehler kleiner als ε kann nur dann gewährleistet werden, wenn der Algorithmus zumindest ein Beispiel aus $[0, \varepsilon]$ sieht.
- Mit Wahrscheinlichkeit $(1-\varepsilon)^s = e^{-\Theta(s\cdot\varepsilon)}$ wird keines von s Beispielen in $[0, \varepsilon]$ liegen. Um mit Wahrscheinlichkeit mindestens $1-\delta$ einen Fehler von höchstens ε zu erlauben, muss

  $$e^{-\Theta(s\cdot\varepsilon)} \leq \delta, \text{ bzw. } 1/\delta \leq e^{\Theta(s\cdot\varepsilon)}$$

  gelten.

$s = \Omega(\frac{\ln(1/\delta)}{\varepsilon})$ Beispiele sind notwendig!

Eine weitreichende Verallgemeinerung I

Nenne eine Konzeptklasse $\mathcal{C}$ trivial, wenn $\mathcal{C}$ nur aus einem Konzept oder aus disjunkten Konzepten besteht.

(1) Wenn $\mathcal{C}$ nicht-trivial ist, dann gibt es zwei Beispiele $x \neq y$ und zwei Konzepte $c_1, c_2 \in \mathcal{C}$ mit

$$\{x, y\} \subseteq c_1 \text{ sowie } x \in c_2, y \notin c_2.$$

(2) Mache x hochwahrscheinlich (Wahrscheinlichkeit $1 - 2\varepsilon$) und y niedrigwahrscheinlich (Wahrscheinlichkeit 2ε).

(3) Ein deterministischer Lernalgorithmus irrt für entweder c_1 oder c_2, wenn Beispiel y nicht gesehen wird.

Wenn der Lernalgorithmus y nicht sieht, dann resultiert der unerlaubt große Fehler 2ε.

Eine weitreichende Verallgemeinerung II

(1) Es muss gewährleistet sein, dass der Algorithmus Beispiel y mit Wahrscheinlichkeit höchstens δ nicht sieht.

(2) Aber y wird mit Wahrscheinlichkeit $(1-2\varepsilon)^s = e^{-\Theta(\varepsilon \cdot s)}$ nicht gesehen.

(3) Wir müssen $e^{-\Theta(\varepsilon \cdot s)} \leq \delta$, bzw. $1/\delta \leq e^{\Theta(\varepsilon \cdot s)}$ fordern.

Für jede nicht-triviale Konzeptklasse $\mathcal{C}$ werden mindestens

$$\Omega(\frac{1}{\varepsilon} \ln(\frac{1}{\delta}))$$

Beispiele benötigt.

Wieviele Beispiele sind notwendig: Der allgemeine Fall

Es gelte $\mathrm{VC}(\mathcal{C}) = d + 1$ und $\mathcal{C}$ zertrümmere die Beispielmenge

$$S = \{x_0, \ldots, x_d\}$$

(1) S wird zertrümmert: Jede Teilmenge von S entspricht einem Konzept.

(2) Wir definieren eine schwierige Verteilung D auf den Beispielen:
- x_0 wird hochwahrscheinlich, und zwar mit Wahrscheinlichkeit $1 - 4\varepsilon$ gewählt.
- Jedes andere x_i erhält die Wahrscheinlichkeit $4\varepsilon/d$.
- Beispiele außerhalb S erhalten die Wahrscheinlichkeit 0.

(3) x_0 wird sehr häufig gewählt und damit sieht ein Lernalgorithmus nur dann genügend viele verschiedene klassifizierte Beispiele in S, wenn die Beispielanzahl genügend groß ist. Wie groß?

Wieviele Beispiele sind notwendig: Der allgemeine Fall

Wenn s Beispiele angefordert werden und wenn $d = \mathrm{VC}(\mathcal{C})$:

(1) Die erwartete Häufigkeit von x_0 ist $s \cdot (1 - 4\varepsilon)$.

(2) Der Algorithmus sieht nur $s \cdot 4\varepsilon$ Beispiele aus $\{x_1, \ldots, x_d\}$.

(3) Angenommen $s \leq \frac{d}{8\varepsilon}$.

- Der Algorithmus sieht nur $d/2$ verschiedene Beispiele aus S.
- Die Klassifikation der nicht gesehenen Beispiele ist beliebig (denn S wird zertrümmert) und damit nicht prognostizierbar.
- Der Algorithmus muss die Klassifikation der nicht gesehenen $d/2$ Beispiele raten und wird sich für ein Zielkonzept in den nicht gesehenen Beispielen irren.
- Die erwartete Fehlerwahrscheinlichkeit ist mindestens $(d/2) \cdot (4\varepsilon/d) = 2\varepsilon > \varepsilon$.

Mindestens

$$\Omega\left(\frac{\mathrm{VC}(\mathcal{C})}{\varepsilon}\right)$$

Beispiele müssen angefordert werden.

Zusammenfassung

Wir kombinieren die untere Schranke bei großer VC-Dimension mit der unteren Schranke für kleines δ:

Für jede nicht-triviale Konzeptklasse $\mathcal{C}$ sind mindestens

$$\Omega(\frac{1}{\varepsilon} \cdot \left[\mathsf{VC}(\mathcal{C}) + \ln\left(\frac{1}{\delta}\right)\right])$$

Beispiele notwendig.

Diese untere Schranke ist in vielen Fällen asymptotisch exakt!

Konsequenzen der unteren Schranke

Da $\Omega(\frac{1}{\varepsilon} \cdot (\mathrm{VC}(\mathcal{C}_n) + \ln(\frac{1}{\delta}))) = \text{Beispiel}(\mathcal{C}_n) = O(\frac{1}{\varepsilon} \cdot (\ln(|\mathcal{C}_n|) + \ln(\frac{1}{\delta})))$, kann die Beispielzahl exakt vorausgesagt werden, wenn

$$\mathrm{VC}(\mathcal{C}_n) = \Theta(\ln(|\mathcal{C}_n|)).$$

(a) $\text{Beispiel}(\text{MONOTON-MONOM}_n) = \Theta(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(b) $\text{Beispiel}(\text{MONOM}_n) = \Theta(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(c) $\text{Beispiel}(k\text{-KNF}_n) = \text{Beispiel}(k\text{-DNF}_n) = \Theta(\frac{n^k}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(e) $\text{Beispiel}(k\text{-KLAUSEL-KNF}_n) = \Theta(\frac{k \cdot n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(f) $\text{Beispiel}(k\text{-TERM-DNF}_n) = \Theta(\frac{k \cdot n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(g) $\text{Beispiel}(\text{BOOLEAN}_n) = \Theta(\frac{1}{\varepsilon} \cdot 2^n + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(h) $\text{Beispiel}(\text{SYM}_n) = \Theta(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$.

(i) $\text{Beispiel}(\text{AUTOMAT}_{n,\Sigma}) = \Theta(\frac{1}{\varepsilon} \cdot n \cdot \log_2 n + \frac{1}{\varepsilon} \cdot \ln\left(\frac{1}{\delta}\right))$. für $|\Sigma| = 2$.

Diskussion

Das Ziel

Die Konzeptklasse $\mathcal{C}$ sei gegeben mit $\mathrm{VC}(\mathcal{C}) = d$. Dann gibt es einen PAC-Algorithmus für $\mathcal{C}$, der höchstens

$$\left\lceil \frac{4}{\varepsilon} \cdot \left(d \cdot \ln\left(\frac{12}{\varepsilon}\right) + \ln\left(\frac{2}{\delta}\right) \right) \right\rceil$$ Beispiele anfordert.

- Vollständig zertrümmerte Beispielmengen S sind chaotisch, denn jede Teilmenge kann als Konzept auftreten.
- Bei genügend vielen Beispielen werden aber fast alle Beispiele in S aufgedeckt, und wir haben kein Problem.

Was passiert, wenn der Algorithmus eine Beispielmenge S mit $s >> \mathrm{VC}(\mathcal{C})$ Beispielen sieht? Wie chaotisch ist S?

Das partielle Zertrümmern großer Beispielmengen

$\mathcal{C}$ zertrümmert eine Beispielmenge S mit $|S| = \mathrm{VC}(\mathcal{C})$. Für größere Beispielmengen: Wieviele Teilmengen „produziert“ $\mathcal{C}$ höchstens?

- Für eine Beispielmenge S setze

$$\Pi_{\mathcal{C}}(S) = \{\, c \cap S \mid c \in \mathcal{C} \,\}.$$

- Für $s \in \mathbb{N}$ ist

$$\Pi_{\mathcal{C}}(s) = \max\{\, |\Pi_{\mathcal{C}}(S)| \mid |S| = s \,\}.$$

Sauer's Lemma

Für $d = \mathrm{VC}(\mathcal{C})$ und jedes $s \in \mathbb{N}$ ist

$$\Pi_{\mathcal{C}}(s) \leq \sum_{i=0}^{d} \binom{s}{i}.$$

Für $s \geq d$ ist $\Pi_{\mathcal{C}}(s) \leq \left(\frac{e \cdot s}{d}\right)^d$.

Der (einfache) Beweis wird im Skript beschrieben.

Sauer's Lemma: Eine Diskussion

Die Beispielmenge S sei gegeben. Wieviele Konzepte der Konzeptklasse $\mathcal{C}$ unterscheiden sich auf der Menge S?

Genau $|\Pi_{\mathcal{C}}(S)|$ Konzepte, denn $\Pi_{\mathcal{C}}(S) = \{ c \cap S \mid c \in \mathcal{C} \}$.

Wenn $|S| = s$ und $d = \mathrm{VC}(\mathcal{C})$, dann gibt es höchstens $\sum_{i=0}^{d} \binom{s}{i}$ Konzepte aus $\mathcal{C}$, die sich auf der Beispielmenge S voneinander unterscheiden.

Wie scharf ist Sauer's Lemma?

- Für $s \leq d = \mathrm{VC}(\mathcal{C})$ ist $\Pi_{\mathcal{C}}(s) = 2^s$, denn es gibt Beispielmengen der Größe s, die vollständig zertrümmert werden können.
- Sauer's Lemma besagt für $s \leq d$:

$$\Pi_{\mathcal{C}}(s) \leq \sum_{i=0}^{d} \binom{s}{i} = \sum_{i=0}^{s} \binom{s}{i} = \sum_{i=0}^{s} \binom{s}{i} 1^i \cdot 1^{s-i} = 2^s.$$

- Sauer's Lemma ist exakt für $s \leq \mathrm{VC}(\mathcal{C})$.

ε-Netze

Die Situation

- Eine Beispielmenge S mit s Beispielen wurde zufällig gewählt.
- Das (unbekannte) Zielkonzept sei $c \in \mathcal{C}$ und D sei die (unbekannte) Verteilung.
- Die Gefahr: Es gibt ein konsistentes Konzept $c' \in \mathcal{C}$ mit zu großem Fehler.
  - c' ist genau dann mit S konsistent, wenn $S \cap (c \oplus c') = \emptyset$.
  - Der Fehler ist zu groß, falls $\text{fehler}_D(c, c') > \varepsilon$.

Wir nennen

$$\Delta_\varepsilon(c) = \{ c \oplus c' \mid c' \in \mathcal{C} \text{ und } \text{fehler}_D(c, c') > \varepsilon \}$$

die Menge der ε-Fehlerregionen.

ε-Netze

Wir nennen eine Beispielmenge S ein ε-**Netz**, wenn $S \cap c^* \neq \emptyset$ für jede ε-Fehlerregion $c^* \in \Delta_\varepsilon(c)$.

Angenommen, S ist ein ε-Netz und c' ist eine konsistente Hypothese mit zu großem Fehler:

- Dann ist $c \oplus c'$ eine ε-Fehlerregion und $c \oplus c' \in \Delta_\varepsilon(c)$ folgt.
- Aber S ist ein ε-Netz und $S \cap (c \oplus c') \neq \emptyset$ folgt.
- Es gibt also ein $x \in S$ mit $x \in c \oplus c'$.
- Aber dann ist c' nicht mit c auf x konsistent.

Ein konsistenter Lernalgorithmus ist ein PAC-Algorithmus, falls

$$\text{prob}[\text{ S ist ein } \varepsilon\text{-Netz }] \geq 1 - \delta$$

.

Die Wahrscheinlichkeit eines ε-Netzes I

Sei A_s das Ereignis, dass eine Menge von s Beispielen **kein** ε-Netz ist. Was ist die Wahrscheinlichkeit von A_s?

$$\begin{aligned}
\text{prob}[\ A_s\ ] &= \text{prob}[\ \text{S ist } \textit{kein}\ \varepsilon\text{-Netz}\ ] \\
&= \text{prob}[\ \text{Es gibt ein } c^* = c \oplus c' \in \Delta_\varepsilon(c) \text{ und } S \cap c^* = \emptyset\ ] \\
&\leq \sum_{c^* \in \Delta_\varepsilon(c)} \text{prob}[\ S \cap c^* = \emptyset\ ]
\end{aligned}$$

Aber die Anzahl der Fehlerregionen in $\Delta_\varepsilon(c)$ kann sogar unendlich groß sein. So geht es nicht.

Die Wahrscheinlichkeit eines ε-Netzes II

Es ist

$$\text{prob}[\ A_s\ ] = \text{prob}[\ \text{Es gibt ein } c^* = c \oplus c' \in \Delta_\varepsilon(c) \text{ und } S \cap c^* = \emptyset\ ].$$

Wir definieren das Ereignis

$$B_s \equiv \exists c^* \in \Delta_\varepsilon(c) \left[c^* \cap S_1 = \emptyset \wedge |c^* \cap S_2| \geq \frac{\varepsilon \cdot s}{2} \right]$$

für Beispielmengen S_1, S_2 von jeweils s Beispielen.

Das Ereigniss B_s tritt ein, wenn eine schlechte Hypothese c' mit $c^* = c \oplus c'$ auf den ersten s Beispielen nicht auffällt, wohl aber auf den zweiten s Beispielen.

Das Ereignis B_s

(1) Wenn das Ereignis B_s eintritt, dann ist eine schlechte Hypothese c' auf der Menge S_1 der ersten s Beispielen konsistent und S_1 ist kein ε-Netz. Also tritt dann auch das Ereignis A_s auf.

(2) Also folgt $\text{prob}[\ B_s\ ] = \text{prob}[\ B_s \mid A_s\ ] \cdot \text{prob}[\ A_s\ ]$.

(3) Wenn c' aber schlecht ist, also einen großen Fehler hat, dann ist die Wahrscheinlichkeit groß, dass c' auf der Menge S_2 der zweiten s Beispiele inkonsistent ist.

(4) $\text{prob}[\ B_s \mid A_s\ ]$ sollte groß sein.

Wenn $s \geq \frac{8}{\varepsilon}$, dann ist

$$\text{prob}[\ A_s\ ] \leq 2 \cdot \text{prob}[\ B_s\ ].$$

Der Beweis, eine Anwendung der Chernoff-Schranke, wird im Skript beschrieben.

Zwischenfazit

(1) Unser Ziel ist der Nachweis, dass eine Menge von s Beispielen hochwahrscheinlich ein ε-Netz ist, falls s hinreichend groß ist.

(2) A_s ist das Ereignis, dass eine Menge von s Beispielen **kein** ε-Netz ist. Wir wissen, dass $\text{prob}[\, A_s \,] \leq 2 \cdot \text{prob}[\, B_s \,]$ mit

$$B_s \equiv \exists c^* \in \Delta_\varepsilon(c) \left[c^* \cap S_1 = \emptyset \wedge |c^* \cap S_2| \geq \frac{\varepsilon \cdot s}{2} \right].$$

(3) Es genügt zu zeigen, dass B_s unwahrscheinlich ist.

(4) Für B_s ziehe eine Beispielmenge S von $2s$ Beispielen: Zuerst die s Beispiele aus S_1 und dann die s Beispiele aus S_2.

- Beachte, dass $(c^* \cap S) \cap S_1 = c^* \cap (S \cap S_1) = c^* \cap S_1$ und $(c^* \cap S) \cap S_2 = c^* \cap (S \cap S_2) = c^* \cap S_2$.
- Wir können also c^* durch $c^* \cap S$ ersetzen.

Die Reduktion auf Sauer's Lemma I

(1) Wir fassen die Einzelexperimente (ziehe zuerst S_1, dann S_2) zu Großexperimenten S zusammen:
- Ziehe zuerst S,
- dann zerlege S zufällig in S_1 und S_2.

(2) Fixiere ab jetzt das Großexperiment S.

Wie groß ist die Wahrscheinlichkeit prob[$B_s \mid S$] des Ereignisses B_s, wenn wir das Großexperiment S durchführen?

$$
\begin{aligned}
& \text{prob}[\, B_s \mid S \,] \\
= \;& \text{prob}[\exists c^* \in \Delta_\varepsilon(c) : (c^* \cap S) \cap S_1 = \emptyset, \mid (c^* \cap S) \cap S_2 \mid \geq \frac{\varepsilon \cdot s}{2}\,] \\
\leq \;& \sum_{c^* \cap S, c^* \in \Delta_\varepsilon(c)} \text{prob}[\, (c^* \cap S) \cap S_1 = \emptyset, (c^* \cap S) \cap S_2 \mid \geq \frac{\varepsilon \cdot s}{2}\,].
\end{aligned}
$$

Die Reduktion auf Sauer's Lemma II

Die erste Frage

Wie groß ist die Menge $\{c^* \cap S \mid c^* \in \Delta_\varepsilon(c)\}$?

(1) Fasse die Menge $\Delta_\varepsilon(c)$ der Fehlerregionen als eine Konzeptklasse auf.
- Dann ist $\Delta_\varepsilon(c) \subseteq c \oplus \mathcal{C}$
- und $\mathrm{VC}(\Delta_\varepsilon(c)) \leq \mathrm{VC}(c \oplus \mathcal{C}) = \mathrm{VC}(\mathcal{C})$ folgt.

(2) Die Menge $\{c^* \cap S \mid c^* \in \Delta_\varepsilon(c)\}$ ist die Einschränkung der Konzeptklasse $\Delta_\varepsilon(c)$ auf die Beispielmenge S.

Wende Sauer's Lemma an: Für $d = \mathrm{VC}(\mathcal{C})$ ist

$$|\{c^* \cap S \mid c^* \in \Delta_\varepsilon(c)\}| \leq \left(\frac{e \cdot s}{d}\right)^d.$$

Abschluss der Analyse I

Die zweite Frage

Wie groß ist

$$\text{prob}[\ (c^* \cap S) \cap S_1 = \emptyset, (c^* \cap S) \cap S_2 \mid \geq \frac{\varepsilon \cdot s}{2}\ ]$$

für eine beliebige Fehlerregion c^*?

(1) Angenommen, $|c^* \cap S| = r$.

(2) Wie groß ist die Wahrscheinlichkeit, dass die Auswahl von S_1 diese r Elemente auslässt?

Genau 2^{-r}, denn jedes der r Beispiele in $c^* \cap S$ erscheint mit Wahrscheinlichkeit genau $\frac{1}{2}$ in S_2.

(3) Für uns ist aber nur der Fall $r \geq \frac{\varepsilon \cdot s}{2}$ interessant.

Es ist $\text{prob}[\ (c^* \cap S) \cap S_1 = \emptyset, (c^* \cap S) \cap S_2 \mid \geq \frac{\varepsilon \cdot s}{2}\ ] \leq 2^{-\frac{\varepsilon \cdot s}{2}}$.

Abschluss der Analyse II

Die Konsequenz der ersten und zweiten Frage

$$
\begin{aligned}
& \text{prob}[\, B_s \mid S \,] \\
\leq & \sum_{c^* \cap S, c^* \in \Delta_\varepsilon(c)} \text{prob}[\, (c^* \cap S) \cap S_1 = \emptyset, (c^* \cap S) \cap S_2 \mid \geq \frac{\varepsilon \cdot s}{2} \,] \\
\leq & \left(\frac{e \cdot s}{d}\right)^d \cdot 2^{-\frac{\varepsilon \cdot s}{2}}.
\end{aligned}
$$

A_s war das Ereignis, dass eine Menge von s Beispielen **kein** ε-Netz ist.

Also folgt $\text{prob}[A_s] \leq 2 \cdot \text{prob}[B_s] \leq 2 \cdot \left(\frac{e \cdot s}{d}\right)^d \cdot 2^{-\frac{\varepsilon \cdot s}{2}}$.

Abschluss der Analyse III

Die Forderung $\text{prob}[A_s] \leq \delta$ führt auf

$$2 \cdot \left(\frac{e \cdot 2s}{d}\right)^d \cdot e^{-\frac{\varepsilon \cdot s}{2}} \leq \delta.$$

Jetzt logarithmiere

$$d \cdot \ln\left(\frac{2e \cdot s}{d}\right) - \frac{\varepsilon \cdot s}{2} \leq \ln\frac{\delta}{2}$$

und vertausche Terme

$$d \cdot \ln\left(\frac{2e \cdot s}{d}\right) + \ln\frac{2}{\delta} \leq \frac{\varepsilon \cdot s}{2}$$

Die Forderung $s = \alpha\frac{1}{\varepsilon} \cdot (d\ln(\frac{1}{\varepsilon}) + \ln(\frac{1}{\delta}))$ für eine hinreichend große Konstante α genügt.

Zusammenfassung

Sei $\mathcal{C}$ eine Konzeptklasse mit $d = \mathrm{VC}(\mathcal{C})$.

(1) $s = \frac{\alpha}{\varepsilon} \cdot (d \ln(\frac{1}{\varepsilon}) + \ln(\frac{1}{\delta}))$ Beispiele genügen für ein erfolgreiches PAC-Lernen von $\mathcal{C}$, solange α hinreichend groß ist,

(2) aber $s = \frac{\beta}{\varepsilon} \cdot (d + \ln(\frac{1}{\delta}))$ Beispiele sind erforderlich, solange β hinreichend klein.

Konsequenzen:

- Die VC-Dimension ist eine beweisbar fast-exakte Formalisierung des Begriffs „Freiheitsgrad“.
- Als Konsequenz der unteren Schranke in (2):
  Wenn $\mathrm{VC}(\mathcal{C}_n) = \Theta(\ln(|\mathcal{C}_n|))$, dann sind genau

$$\Theta(\frac{1}{\varepsilon} \cdot (\ln |\mathcal{C}_n| + \ln(\frac{1}{\delta})))$$

  Beispiele für ein erfolgreiches PAC-Lernen von $\mathcal{C}_n$ hinreichend und notwendig.

Das Konsistenzproblem

(a) Eine *parametrisierte Konzeptklasse* $\mathcal{C} = (\mathcal{C}_n)_{n\in\mathbb{N}}$ besteht aus Konzeptklassen $\mathcal{C}_n$, wobei ein Konzept in $\mathcal{C}_n$ nur aus Beispielen der Länge höchstens n besteht.

(b) Wir sagen, dass ein PAC-Algorithmus L *effizient* für $\mathcal{C}$ ist, wenn L für das Erlernen von $\mathcal{C}_n$ höchstens $s = \mathrm{poly}(n, \frac{1}{\varepsilon}, \log_2(\frac{1}{\delta}))$ Beispiele anfordert und in Zeit $\mathrm{poly}(s)$ rechnet.

(1) Die wichtigsten Konzeptklassen besitzen eine in der Beispiellänge n polynomielle VC-Dimension: Bis auf $\mathbf{BOOLEAN}_n$ genügen stets $s = \mathrm{poly}(n, \frac{1}{\varepsilon}, \log_2(\frac{1}{\delta}))$ Beispiele.

(2) Um eine gute Hypothese in Zeit $\mathrm{poly}(s)$ zu bestimmen, sollten wir insbesondere das **Konsistenzproblem**

> Gibt es eine konsistente Hypothese für eine Menge klassifizierter Beispiele?

in Zeit $\mathrm{poly}(s)$ lösen können.

Einfache Konzeptklassen

Die folgenden Konzeptklassen besitzen effiziente PAC-Algorithmen:

(a) MONOTON-MONOM,

MONOM,

k-DNF für fixiertes k,

k-KNF für fixiertes k,

k-TERM-DNF für fixiertes k (mit Hypothesenklasse $\mathcal{H} = k$-KNF),

k-KLAUSEL-KNF für fixiertes k (mit Hypothesenklasse $\mathcal{H} = k$-DNF),

(b) SYM,

(c) RECHTECK und

(c) HALBRAUM.

Boolesche Konzeptklassen I

(a1) **MONOTON-MONOM$_n$**:

- Das Monom

$$x_1 \wedge x_2 \wedge \cdots \wedge x_n$$

  ist die erste Hyothese.
- Streiche alle Variablen, die einem positiven Beispiel widersprechen.

(a2) **MONOM$_n$**:

- Das Monom

$$x_1 \wedge \neg x_1 \wedge x_2 \wedge \neg x_2 \wedge \cdots \wedge x_n \wedge \neg x_n$$

  ist die erste Hyothese.
- Streiche alle Literale, die einem positiven Beispiel widersprechen.

Boolesche Konzeptklassen II

(a3) **k-KNF**: In den Übungen.

(a4) **k-DNF**: In den Übungen

(a5) **k-TERM DNF**:

- Übungsaufgabe: Zu jeder **k-Term-DNF**-Formel gibt es eine äquivalente **k-KNF**-Formel.
- ACHTUNG: Wenn **k-TERM DNF** als Hypothesenklasse benutzt wird, dann ist das Konsistenzproblem $\mathcal{NP}$-vollständig und effizientes Lernen ist unmöglich!

(a6) **k-KLAUSEL KNF**: Wie **k-TERM DNF**.

SYM und RECHTECK

(b) **SYM**$_n$:

- Wenn eine Funktion $f : \{0,1\}^n \to \{0,1\}$ symmetrisch ist und die Eingabe x genau i Einsen hat, dann ist $f(x) = f(1^i 0^{n-i})$.
- Die Hypothese akzeptiert Eingaben mit i Einsen genau dann, wenn es mindestens ein positives Beispiel mit i Einsen gibt.

(c) **RECHTECK**$_n$:

- Bestimme das kleinste Rechteck, das alle positiven Beispiele enthält.
- Wie? Für Beispiele $x_1, \ldots, x_s \in \mathbb{R}^n$ und jede Dimension j bestimme das Interval $I_j = [\min_{i=1}^{s} x_{i,j}, \max_{i=1}^{s} x_{i,j}]$.
- Setze $R = I_1 \times \cdots \times I_n$.

Mit Ausnahme der einfachen Konzeptklassen **SYM**$_n$ und **RECHTECK**$_n$ wurde das Konsistenzproblem mit Varianten des Algorithmus für **MONOM**$_n$ gelöst.

(d) **HALBRAUM**$_n$:

- $x_1, \ldots, x_r$ seien die positiven und $y_1, \ldots, y_t$ seien die negativen Beispiele.
- Stelle das folgende lineare Programm auf

$$\max \varepsilon \text{ so dass } \begin{array}{l} w^T \cdot x_i \geq t + \varepsilon \text{ für } i = 1, \ldots, r \\ w^T \cdot y_j \leq t - \varepsilon \text{ für } j = 1, \ldots, t. \end{array}$$

  Die Variablen des linearen Programms sind ε, der Schwellenwert t und die Komponenten des Vektors w.
- Das lineare Programm kann mit Interior Point Verfahren (oder mit dem Simplex Verfahren) effizient gelöst werden.

(e) $\wedge$-**HALBRAUM**$_n$ besteht aus allen Konzepten, die durch ein neuronales Netz mit zwei Threshold-Gattern und einem abschließenden UND-Gatter berechnet werden.

In den Übungen: Das Konsistenzproblem ist $\mathcal{NP}$-vollständig,

Wie entscheidend ist das Konsistenzproblem?

Und wenn das Konsistenzproblem für die Konzeptklasse $\mathcal{C}$ nicht effizient lösbar ist?
Kann es trotzdem effiziente PAC-Algorithmen für $\mathcal{C}$ geben?

(1) Ja: wir zeigen später, dass das Konsistenzproblem für **k-TERM DNF** $\mathcal{NP}$-vollständig ist, aber **k-TERM DNF** kann effizient mit Hypothesenklasse **k-KNF** gelernt werden.

(2) Angenommen, das Konsistenzproblem für $\mathcal{C}$ ist $\mathcal{NP}$-vollständig. Kann es dann einen effizienten PAC-Algorithmus $\mathcal{L}$ für $\mathcal{C}$ mit $\mathcal{C}$ als Hypothesenklasse geben?

Wahrscheinlich nicht: Wir konstruieren aus $\mathcal{L}$ einen effizienten, randomisierten Algorithmus für das Konsistenzproblem.

Eine effiziente Lösung des Konsistenzproblems für $\mathcal{C}_n$

(1) Die klassifizierte Beispielmenge S sei vorgegeben. Wir müssen entscheiden, ob es ein mit S konsistentes Konzept in $\mathcal{C}_n$ gibt.

Setze $\varepsilon = \frac{1}{|S|+1}$ und $\delta = \frac{1}{2}$.

(2) Der PAC-Algorithmus $\mathcal{L}$ möge $s = s(\varepsilon, \delta) = \text{poly}(n)$ Beispiele anfordern. Erzeuge s Beispiele aus S gemäß der Verteilung

$$D(x) = \begin{cases} \frac{1}{|S|} & x \in S \\ 0 & \text{sonst.} \end{cases}$$

(3) Sei h_c die von L ausgegebene Hypothese.
/* h_c ist eine effiziente Lösung des Konsistenzproblem für S mit Wahrscheinlichkeit mindestens $\frac{1}{2}$. */

(4) Antworte „S konsistent mit einem Konzept in $\mathcal{C}$“, wenn h_c konsistent ist.

PAC-Lernen von $\mathcal{C}$ mit Hypothesenklasse $\mathcal{C}$

Warum ist h_c eine effiziente Lösung des Konsistenzproblem für S mit Wahrscheinlichkeit mindestens $\frac{1}{2}$?

(1) $\mathcal{L}$ erfüllt als PAC-Algorithmus

$$\text{prob}[\ \text{fehler}_D(c, h_c) \leq \frac{1}{|\ S\ | + 1}\ ] \geq \frac{1}{2},$$

(2) Wenn aber nur ein einziges Beispiel aus S missklassifiziert wird, dann ist $\text{fehler}_D(c, h_c) \geq \frac{1}{|S|} > \frac{1}{|\ S\ |+1}$.

(3) Unser Algorithmus ist fehlerfrei bei positiver Antwort. Eine negative Antwort ist mit Wahrscheinlichkeit höchstens $\frac{1}{2}$ fehlerhaft.

Wenn das Konsistenzproblem $\mathcal{NP}$-vollständig ist, dann gibt es vermutlich keine effizienten Lernalgorithmen.

Das Konsistenzproblem für 3-KLAUSEL-KNF

Das 3-Färbbarkeitsproblem

- Gegeben sei ein ungerichteter Graph G mit Knotenmenge V und Kantenmenge E.
- Ist es möglich die Knoten mit drei Farben zu färben, so dass jede Kante nur verschieden gefärbte Knoten verbindet?

(1) Das 2-Färbbarkeitsproblem ist effizient lösbar,

(2) das 3-Färbbarkeitsproblem ist hingegen $\mathcal{NP}$-vollständig.

Unser Ziel:

Konstruiere die polynomielle Reduktion

3-Färbbarkeit $\leq_p$ Konsistenz Problem für **3-KLAUSEL-KNF**.

Die Reduktion

Der ungerichtete Graph $G = (V, E)$ mit $V = \{1, \ldots, n\}$ sei die Eingabe für 3-Färbbarkeit.

Die transformierte Eingabe für das Konsistenzproblem:

(1) Die Beispiele sind n-Bit Worte, wobei die Bitposition v dem Knoten v „entspricht“.

(2) Die Menge S_- der negativen Beispiele besteht aus allen Worten mit genau einer Eins.

(3) Die Menge S_+ der positiven Beispiele besteht aus allen Worten mit genau zwei Einsen, die einer Kante von G entsprechen.

(4) Die Beispielmenge ist $S = S_- \cup S_+$.

Wir müssen nachweisen, dass G genau dann 3-färbbar ist, wenn es eine mit S konsistente 3-KLAUSEL-KNF gibt.

Verifikation der Reduktion I

Angenommen, der Graph G ist 3-färbbar mit der Färbung $\text{Farbe} : V \to \{1, 2, 3\}$.

Wir betrachten die Formel $c = k_1 \wedge k_2 \wedge k_3$ mit

$$k_i \equiv \bigvee_{v \in V,\ \text{Farbe}(v) \neq i} x_v.$$

(1) c verwirft jedes Wort x mit genau einer Eins (in Position v) mit der Klausel, die der Farbe von v entspricht.

Wenn z.B. $\text{Farbe}(v) = 1$, dann ist $k_1(e_v) = 0$ und somit $c(e_v) = 0$.

(2) c akzeptiert jedes Wort x mit genau zwei Einsen, das einer Kante $\{u, v\}$ entspricht.

- Da $\{u, v\}$ eine Kante ist, besitzen u und v verschiedene Farben.
- Jede Klausel k_i wird von x erfüllt.

Verifikation der Reduktion II

Angenommen die 3-KLAUSEL-KNF$_n$-Formel $c \equiv k_1 \wedge k_2 \wedge k_3$ ist konsistent mit der Beispielmenge S.

(1) c verwirft alle Worte x mit genau einer Eins (in Position v):
- Es gibt mindestens eine Klausel $k_{f(v)}$ mit $k_{f(v)}(x) = 0$.
- Wähle $f(v) \in \{1, 2, 3\}$ als Farbe für Knoten v.

(2) f definiert eine 3-Färbung. Warum verbinden Kanten nur verschieden gefärbte Knoten?
- Sei $\{u, v\}$ eine Kante von G.
- Wenn $f(u) = f(v)$, dann enthält die Klausel $k_{f(u)}$ weder das Literal x_u noch das Literal x_v und c verwirft das der Kante $\{u, v\}$ entsprechende Wort.

Das Konsistenzproblem für **3-KLAUSEL-KNF** ist $\mathcal{NP}$-vollständig.

Sehr schwierige Konzeptklassen

- Die Konzeptklasse **3-KLAUSEL-KNF** ist nur schwierig, wenn **3-KLAUSEL-KNF** auch als Hypothesenklasse verwandt wird.
- Gibt es Konzeptklassen, die effizientes Lernen für alle möglichen Hypothesenklassen verhindern?

Erfahrungsgemäß sehr schwierige Konzeptklassen sind

- Schaltkreise: sogar wenn wir nur logarithmische Tiefe erlauben.
- Neuronale Netzwerke: sogar wenn die Tiefe durch eine Konstante beschränkt ist
- und überraschenderweise endliche Automaten, sogar wenn wir die Zustandszahl beschränken.

Wir werden zeigen, dass wir das RSA Kryptosystem brechen können, wenn wir eine dieser Konzeptklassen effizient lernen können.

Public-Key Kryptographie

- Zwei Personen, Alice und Bob, möchten verschlüsselte Nachrichten über einen öffentlichen Kanal austauschen.
- Eine dritte Person, Eve, möchte die mitgehörte Nachricht entschlüsseln.

Zur Verschlüsselung werden Trapdoor-Funktionen f_w eingesetzt:
$f_w(x)$ ist einfach zu berechnen, aber $f_w^{-1}(y)$ kann nur dann effizient berechnet werden, wenn die geheime Information w bekannt ist.

Die Kommunikation

(1) Alice stellt für sich eine Trapdoor-Funktion f_w auf und veröffentlicht ein Programm zur Berechnung von f_w.

(2) Bob kodiert seine Nachricht x mit dem veröffentlichten Programm und veröffentlicht seinerseits $f_w(x)$.

(3) Alice kann $f_w(x)$ effizient dekodieren, da sie w kennt. Eve kennt w nicht und kann deshalb keine effiziente Dekodierung erreichen.

Das RSA-Kryptosystem

Rivest, Shamir und Adleman haben 1978 das nach ihnen benannte RSA-Kryptosystem vorgestellt:

(1) Alice wählt zwei große Primzahlen p und q mit $p \neq q$, sowie den Kodierungsexponenten e. Sie veröffentlicht $N = p \cdot q$ und e.

(2) Bob möchte eine Nachricht x mit $1 \leq x < N$ und $\mathrm{ggT}(x, N) = 1$ kodieren.

- Bob berechnet zuerst $x \bmod N, x^2 \bmod N, x^4 \bmod N, \ldots, x^{2^i} \bmod N, \ldots, x^{2^{\lfloor \log_2 N \rfloor}} \bmod N$ durch fortgesetztes Quadrieren.
- Danach berechnet er $x^e \bmod N$ durch Multiplikation der geeigneten Potenzen
- und veröffentlicht $f_{p,q,e}(x) = x^e \bmod N$.

Alice nutzt ihr Wissen der Primfaktoren p und q, um $f_{p,q,e}(x)$ effizient zu dekodieren. Wie?

Die Eulersche ϕ-Funktion

$\phi(N)$ ist die Anzahl der primen Restklassen modulo N, also

$$\phi(N) = |\{ x \in \{1, \ldots, N\} \mid \text{ggT}(x, N) = 1 \}|$$

- Welche Zahlen sind nicht teilerfremd zu N?
  - die p Vielfachen von q, d.h. $\{q, 2q, \ldots, (p-1)q, p \cdot q\}$
  - und die q Vielfachen von p, d.h. $\{p, 2p, \ldots, (q-1) \cdot p, q \cdot p\}$, sind nicht teilerfremd zu N.
- $\phi(N) = p \cdot q - p - q + 1 = (p-1)(q-1)$.
- Alice kennt p und q und kann deshalb $\phi(N)$ leicht bestimmen.

Wie dekodiert Alice?

Der Satz von Euler

Für jede endliche Gruppe G mit Gruppenordnung m gilt $g^m \equiv 1$ für alle $g \in G$.

(1) Bob sendet die Nachricht $y = x^e \mod N$.

(2) Alice bestimmt den Dekodierungsexponenten d so, dass

$$e \cdot d \equiv 1 \mod \phi(N).$$

(3) Alice berechnet

$$y^d = (x^e)^d \equiv x^{ed} \equiv x^{1+k\cdot\phi(N)} \equiv x^1 \equiv x \mod N.$$

Die RSA Hypothese

Es gibt keinen probabilistischen Algorithmus A mit den folgenden beiden Eigenschaften.

(a) Für jede Zahl $N = p \cdot q$ und jeden Exponenten $e > 1$ mit $\text{ggT}(e, \phi(N)) = 1$ gilt

$$\text{prob}\Big[\begin{array}{l} A \text{ berechnet zu gegebenen } N, e, x^e \bmod N \\ \text{das Urbild } x, \text{ sofern } x^e \in \mathbb{Z}_N^* \text{ und 0 sonst} \end{array} \Big] \geq \frac{3}{4},$$

wobei die Wahrscheinlichkeit über die internen Münzwürfe von A und die zufällige Wahl von $x \in \{1, \ldots, N-1\}$ zu bilden ist.

(b) Algorithmus A läuft in Zeit $\text{poly}(\log_2 N)$.

Die RSA-Hypothese konnte seit über 30 Jahren nicht widerlegt werden.

Die Konzeptklasse **RSA**

(a) N sei das Produkt zweier verschiedener Primzahlen p, q. $e \in \mathbb{Z}^*_{\phi(N)}$ sei der Kodierungsexponent und i sei eine Bitposition mit $1 \leq i \leq \lfloor \log_2 N \rfloor + 1$.

(b) Das Konzept $c_{N,e,i}$ besteht aus allen Vektoren

$$\left(N, e, x^e \bmod N, x^{e \cdot 2^1} \bmod N, \ldots, x^{e \cdot 2^{\lfloor \log_2 N \rfloor}} \bmod N \right),$$

so dass das $i$te Bit von x mit 1 übereinstimmt.

(c) Die Konzeptklasse $\mathbf{RSA}_n$ umfasst alle Konzepte $c_{N,e,i}$ mit RSA-Modulen N der Bitlänge höchstens n. Es ist

$$\mathbf{RSA}_n = \left\{ c_{N,e,i} \;\middle|\; \begin{array}{l} N \text{ ist RSA-Modul mit } 1 \leq N < 2^n, \\ e \in \mathbb{Z}^*_{\phi(N)} \text{ und } 1 \leq i \leq \lfloor \log_2 N \rfloor + 1 \end{array} \right\}.$$

Es ist $\mathbf{RSA} = (\mathrm{RSA}_n)_{n \in \mathbb{N}}$.

PAC Algorithmen für RSA

Angenommen, es gibt einen effizienten PAC-Algorithmus für **RSA**.

(1) Alice veröffentlicht ihre Kodierung (N, e).

(2) Eve knackt das RSA-Kryptosystem wie folgt:

- Eve bestimmt zufällig Zahlen $x_1, \ldots, x_s \in \{0, \ldots, N-1\}$ gemäß der Gleichverteilung und berechnet jeweils $y_i = x_i^e \bmod N$.
- Sie klassifiziert genau die Beispiele y_i als positiv für die x_j an der niedrigstwertigsten Position eine 1 hat und $\mathrm{ggT}(x_j, N) = 1$ gilt.
- Die klassifizierten Beispiele werden einem PAC-Algorithmus (mit Fehler $\varepsilon = \frac{1}{n^2}$ und $\delta = \frac{1}{n^2}$) übergeben.

(3) Jetzt hat Eve das niedrigstwertige Bit der Nachricht entschlüsselt und wiederholt ihr Vorgehen für die verbleibenden Bits.

Wenn die RSA-Hypothese gilt, dann besitzt **RSA** keine effizienten PAC-Algorithmen.

Konsequenzen

Die RSA-Hypothese möge gelten. $\mathcal{C} = (\mathcal{C}_n \mid n \in \mathbb{N})$ sei eine parametrisierte Konzeptklasse, wobei $\mathcal{C}_n$ nur aus Konzepten mit Beispielen der Länge höchstens n bestehe.

Wenn $\text{RSA}_n \subseteq \mathcal{C}_{\text{poly}(n)}$ gilt, dann besitzt $\mathcal{C}$ keine effizienten PAC-Algorithmen.

Welche Konzeptklassen sind mächtig genug, um **RSA** zu enthalten?

LOG-SCHALTKREIS$_n$:

- Sei S ein $\{\wedge, \vee, \neg\}$-Schaltkreis mit Fanin höchstens zwei, Eingaben $x \in \{0,1\}^n$, Tiefe höchstens $\log_2 n$ und einem Ausgabegatter.
- **LOG-SCHALTKREIS**$_n$ enthält die von S berechnete boolesche Funktion als Konzept.

Wie einschneidend ist die Einschränkung auf logarithmische Tiefe?

LOG-SCHALTKREIS$_n$

RSA$_n \subseteq$ LOG-SCHALTKREIS$_{\mathrm{poly}(n)}$.

(1) Schaltkreise können n Binärzahlen $x_1, \ldots, x_n \in \{0,1\}^n$ in logarithmischer Tiefe $O(\log_2 n)$ miteinander multiplizieren.

(2) Wenn N, p, q und der Dekodierungsexponent d bekannt sind, dann kann in logarithmischer Tiefe

überprüft werden, ob die Eingabe die Form

$$\left(N, e, y \bmod N, y^2 \bmod N, y^{2^2} \bmod N, \ldots, y^{2^{\lfloor \log_2 N \rfloor}} \bmod N\right)$$

hat und die Nachricht x kann rekonstruiert werden:

$$x \equiv y^d \equiv y^{\sum_{i=0}^{k} d_i \cdot 2^i} \equiv \prod_{i,\ d_i = 1} y^{2^i} \bmod N.$$

THRESHOLD-SCHALTKREIS$_{t,n}$:

- Ein Schaltkreis S ist ein Threshold-Schaltkreis, wenn seine Gatter Thresholdfunktionen der Form

$$f(y_1, \ldots, y_m) = \begin{cases} 0 & \sum_{i=1}^{m} w_i \cdot y_i < t \\ 1 & \text{sonst} \end{cases}$$

  berechnen.
- Wenn S Eingaben $x \in \{0,1\}^n$ erhält, aus n Gattern besteht, die Tiefe höchstens t besitzt und nur ein Ausgabegatter hat, dann enthält **THRESHOLD-SCHALTKREIS**$_{t,n}$ die von S berechnete boolesche Funktion als Konzept.
- Threshold-Schaltkreise können n Binärzahlen $x_1, \ldots, x_n \in \{0,1\}^n$ in Tiefe 5 mit poly(n) Gattern miteinander multiplizieren.

$\text{RSA}_n \subseteq$ **THRESHOLD-SCHALTKREIS**$_{7,\text{poly}(n)}$.

Sind neuronale Netze effizient lernbar?

Die RSA-Hypothese gelte. Dann hat weder **LOG-SCHALTKREIS**$_n$ noch **THRESHOLD-SCHALTKREIS**$_{7,n}$ effiziente PAC-Algorithmen.

- Insbesondere sind Perzeptrons mit sieben Schichten nicht effizient lernbar, egal mit welcher Hypothesenklasse.
- Neuronale (feedforward-)Netzwerke besitzen statt Threshold-Gattern reellwertige Gatter wie z. B. Sigmoid-Gatter.

  feedforward Netze der Tiefe sieben sind mindestens so mächtig wie Perzeptrons mit sieben Schichten.

Ein Grundgesetz

Ausdrucksstarke Konzeptklassen wie etwa neuronale Netzwerke können viele Phänomene modellieren, sind deshalb aber auch schwer zu erlernen:

Lernverfahren für neuronale Netze müssen inhärent langsam sein!

Sind endliche Automaten effizient lernbar?

Endliche Automaten haben, wenn die Eingabe häufig genug wiederholt wird, die Berechnungskraft von Schaltkreisen:

Sei S ein Schaltkreis der Tiefe t. Wir konstruieren einen endlichen Automaten A_S mit $\text{poly}(2^t)$ Zuständen, der $x^{2^{t+1}-1}$ genau dann akzeptiert, wenn S die Eingabe x akzeptiert.

(1) Wir wissen: Schaltkreise logarithmischer Tiefe besitzen keine effizienten PAC-Algorithmen, selbst wenn Beispiele gemäß der Gleichverteilung generiert werden.

(2) Sei $t = \log_2 n$ und sei $D_n(r)$ die Gleichverteilung auf allen Worten der Form $x^{2^{r+1}-1}$ für $x \in \{0,1\}^n$.

Wenn $\textbf{AUTOMAT}_{\text{poly}(2^t),\{0,1\}}$ effiziente PAC-Algorithmen für $D_n(t)$ hat, dann ist $\textbf{SCHALTKREIS}_n$ effizient PAC-lernbar für $D_n(0)$.

AUTOMAT_n hat keine effizienten PAC-Algorithmen.

Das Pebble Spiel

Eine Schaltkreisauswertung mit möglichst wenigen Registern:

- Ein Pebble darf auf eine Quelle des Schaltkreises gesetzt werden (das entsprechende Eingabebit wird in ein Register geladen).
- Wenn alle Vorgänger eines Bausteins Pebbles besitzen, dann darf der Baustein selbst gepebbelt werden (das Ergebnis des Bausteins darf berechnet und in ein Register geladen werden). Im gleichen Zug dürfen alle „Vorgänger-Pebbles“ gelöscht werden.

(1) Übungsaufgabe: Sei S ein $\{\wedge, \vee, \neg\}$-Schaltkreis mit Fan-in höchstens 2 und genau einer Ausgabe. Die Tiefe von S sei t.

S kann mit $\leq t + 1$ Pebbles in Zeit $\leq 2^{t+1} - 1$ gepebbelt werden.

(2) Ein endlicher Automat kann das Pebble Spiel mit $\text{poly}(2^t)$ Zuständen spielen.

Das PAC-Modell: Eine Zusammenfassung I

(1) VC-Dimension einer Konzeptklasse $\mathcal{C}$ formalisiert die Anzahl der Freiheitsgrade von $\mathcal{C}$.

(2) Das PAC-Modell:
- Fairness Bedingung: Werte die Hypothese auf derselben Verteilung aus auf der Beispiele erzeugt wurden.
- Ein kleiner Fehler muss hochwahrscheinlich garantiert werden.

(3) Wenn Fehler höchstens ε mit Wahrscheinlichkeit mindestens $1-\delta$ erreicht werden soll:
- mindestens $\Omega(\frac{1}{\varepsilon} \cdot \left[\mathrm{VC}\left(\mathcal{C}\right) + \ln\left(\frac{1}{\delta}\right)\right])$ Beispiele sind erforderlich
- und höchstens $O(\frac{1}{\varepsilon} \cdot \left[\mathrm{VC}\left(\mathcal{C}\right) \cdot \ln\left(\frac{1}{\varepsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right])$ Beispiele sind ausreichend.

(4) Gute Lernalgorithmen:
- bestimmen eine (möglichst) konsistente Hypothese, lernen die klassifizierten Beispiele also auswendig.
- Lernerfolg, wenn Anzahl der Beispiele wesentlich größer ist als die VC-Dimension der Hypothesenklasse.
- Occam Algorithmen suchen eine möglichst kurze, mit vielen Beispielen konsistente Hypothese.

Das PAC-Modell: Eine Zusammenfassung II

(5) Schwierige Konzeptklassen.

- Repräsentationsabhängige Ergebnisse:
  - **k-TERM DNF** kann nicht mit **k-TERM DNF** effizient PAC-gelernt werden, wohl aber mit Hypothesenklasse **k-KNF**.
  - Wenn die Konzeptklasse $\mathcal{C}$ aus dem Durchschnitt von zwei „booleschen“ Halbräumen besteht, dann ist $\mathcal{C}$ nicht effizient mit $\mathcal{C}$ PAC-lernbar.
- Repräsentationsunabhängige Ergebnisse:
  - Weder SCHALTKREIS$_n$, noch
  - neuronale feedforward Netze aus mindestens sieben Schichten noch
  - endliche Automaten sind mit *irgendeiner* Hypothesenklasse effizient PAC-lernbar.

Kritik am PAC-Modell

- In den meisten Anwendungen (Handschriftenerkennung, Textklassifizierung etc.) können wir die Konzeptklasse nicht exakt beschreiben.
- Erfolgreiches Lernen wird für *jede* Beispielverteilung gefordert.
- Wir müssen stattdessen ausnutzen, dass die Beispielverteilung das Lernen oft unterstützt:

  Informative Beispiele werden sorgfältig ausgewählt und aufbereitet.

Später: Support-Vektor Maschinen bieten dem Anwender die Möglichkeit, über Feature Vektoren zusätzliches Wissen über das Konzept „in die Beispiele einzuarbeiten“.

Die Mutter aller Ungleichungen

Für jedes $x > -1$ gilt

$$e^{x/(1+x)} \leq 1 + x \leq e^x.$$

Darüberhinaus gilt $1 + x \leq e^x$ für alle $x \in \mathbb{R}$.

- Der natürliche Logarithmus ist eine konkave Funktion:
  Die Steigung der Sekante in den Punkten 1 und $1 + x$ ist durch die Tangentensteigungen in den Punkten 1 nach oben und $1 + x$ nach unten beschränkt.
- Als Konsequenz

$$\frac{1}{1+x} \leq \frac{\ln(1+x) - \ln(1)}{(1+x) - 1} = \frac{\ln(1+x)}{x} \leq 1.$$

- Jetzt exponentieren.

Die Chernoff-Schranke

Die Situation

- $X_1, \ldots, X_n$ seien **unabhängige**, binäre Zufallsvariablen: Jedes X_i nimmt also nur die Werte 0 oder 1 an.
- X_i habe die „Erfolgswahrscheinlichkeit“ $p_i = \text{prob}[X_i = 1]$.
- Dann ist $N = \sum_{i=1}^{n} p_i$ die erwartete Anzahl der Erfolge.

Die Chernoff-Schranken:

$$\text{prob}[\sum_{i=1}^{n} X_i > (1+\beta)\cdot N] \quad \leq \quad \left(\frac{e^{\beta}}{(1+\beta)^{1+\beta}}\right)^{N} \leq e^{-N\cdot\beta^2/3}$$

$$\text{prob}[\sum_{i=1}^{n} X_i < (1-\beta)\cdot N] \quad \leq \quad \left(\frac{e^{-\beta}}{(1-\beta)^{1-\beta}}\right)^{N} \leq e^{-N\cdot\beta^2/2}$$

für jedes $\beta > 0$ (bzw. $0 < \beta \leq 1$ im zweiten Fall).