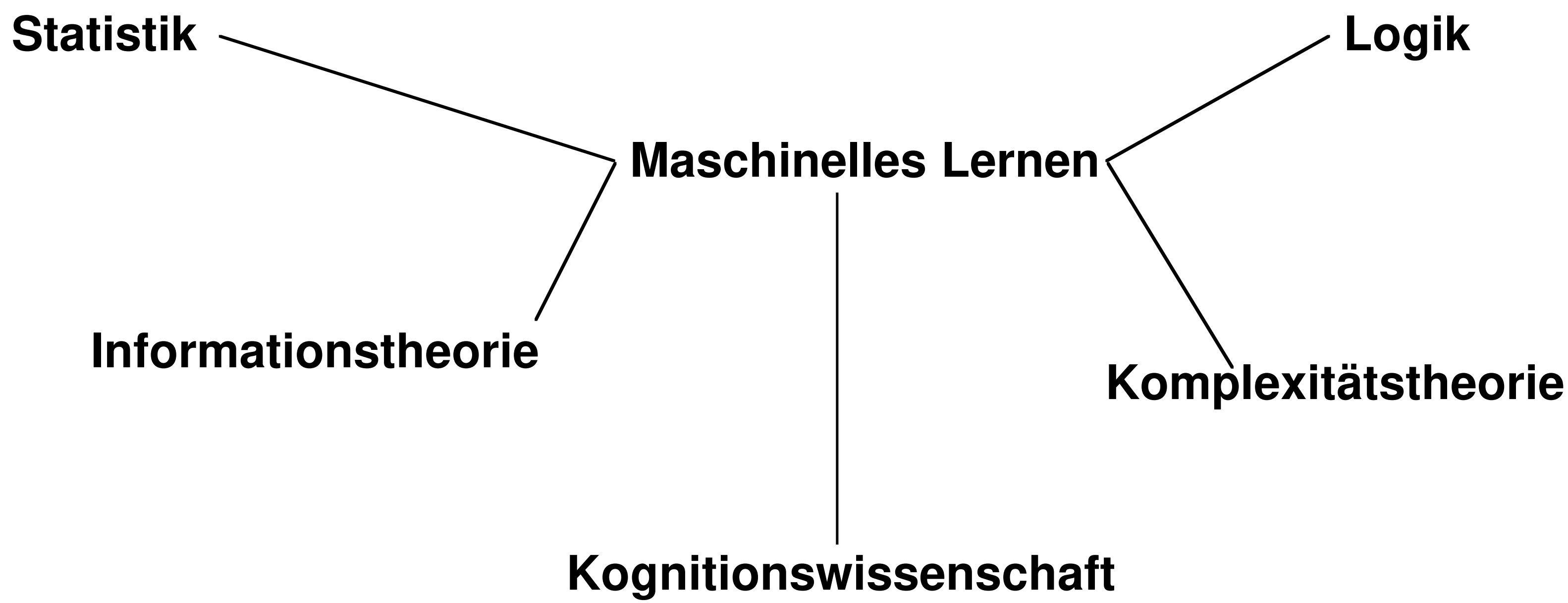
Maschinelles Lernen

Vorlesung

Übung



Anwendungen

Entdecken von Mißbrauchmustern in Kreditkarten-Transaktionen

Entdecken von Merkmalen kreditwürdiger Kunden

Anpassen an an einen Sprecher beim Verstehen gesprochener Sprache

Anpassen an einen Benutzer bei der Auswahl von interessanten Web-Seiten

Klassifikation von Objekten in Bilddaten, z.B. Himmelskörper

Erkennen von handgeschriebenen Buchstaben, Zahlen

Spiele lernen, z.B. Backgammon durch 1 Mio. Spiele eines Systems gegen sich selbst

Lernen, einen Stab auf einem Schienenfahrzeug zu balancieren

Lernen einer Grammatik für eine Sprache

Lernen von Programm-Code aus Beispielen der gewünschten Ein-/Ausgabe

...

Lernen ist...

Funktionsapproximation

Suche

Wissenserwerb

Kompression

der induktive Schluß

Leistungssteigerung

Fragen

Wieviele Beispiele muß ich mindestens haben, bis ich ein ausreichend korrektes und vollständiges Lernergebnis erzielen kann?

Wie mächtig muß mein Repräsentationsformalismus sein, damit ich ein annähernd korrektes und vollständiges Lernergebnis ausdrücken kann?

Wie schwach darf er sein?

Unter welchen Umständen wird der Lernalgorithmus zu einem Ergebnis kommen und anhalten?

Sind auch die Zwischenergebnisse interpretierbar?

Welche Zusicherungen kann der Algorithmus über sein Ergebnis garantieren?

z.B.: dies ist die speziellste Generalisierung über allen Daten -- wenn sie falsch ist, sind auch die Daten falsch!

z.B.: Dies sind alle Regeln, die in den Daten verborgen sind -- wenn eine fehlt, fehlen auch die entsprechenden Daten!

Ausschnitt der Vorlesung

Algorithmen	Lernaufgaben	Theorie	Anwendungen
backprop	Begriffslernen	Ordnung nach	KDD
lgg	Regellernen	Allgemeinheit	BLearn
Revision	Theorieerwerb		Kognition
		Lernbarkeit	...

Literatur

Lehrbücher:

Tom Mitchell "Machine Learning" McGraw-Hill Companies, 1997

Nada Lavrac, Saso Dzeroski "Inductive Logic Programming" Ellis Horwood, 1994

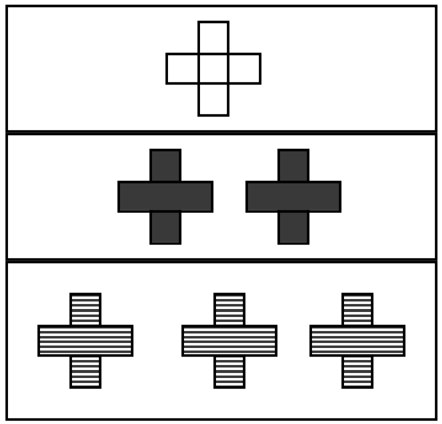
Originalarbeit (deutsch)

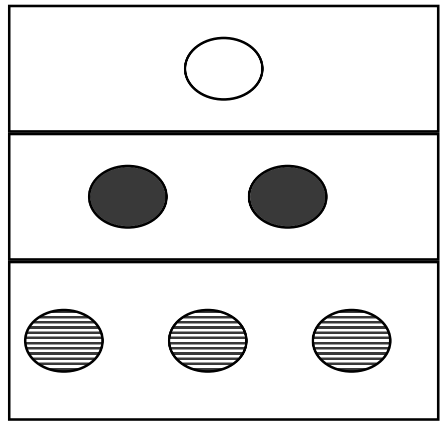
Jörg-Uwe Kietz "Induktive Analyse relationaler Daten" Diss. TU Berlin 1996

zahlreiche Artikel und Bücher!

Begriffsbildung Kategorisierung

- alles zusammenfassen zu einer Klasse, was gemeinsame Merkmale hat
- was sind Merkmale?





Bedarf für Kategorisierung:

Es gibt schon ein Wort dafür

Vorhersage ist nötig

Begriff erleichtert Definition anderer Begriffe

Begriffsbildung
Charakterisierung

Kategorien abgrenzend beschreiben

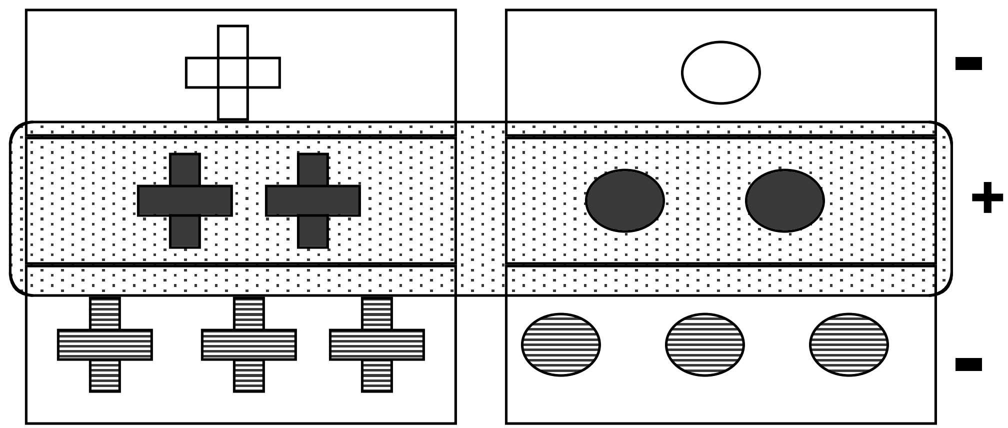
für Gegensätze dieselben Merkmale verwenden

so wenig Merkmale wie möglich

Vererbbarkeit der Merkmale

Operationalität der Merkmale

- sind Begriffe und Merkmale wirklich verschieden?



a) ist schwarz,

b) besteht aus 2 Teilobjekten,

c) ist schwarz und besteht aus 2 Teilobjekten

Probleme der Charakterisierung

Herkunft der Merkmale:

- **verankert in der Wahrnehmung**
- **vermittelt über die Sprache/Kultur**

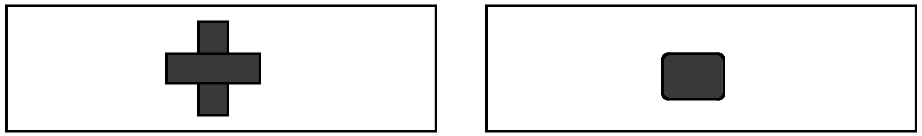
Auswahl von Merkmalen:

- **nur solche Merkmale, die auf alle Begriffe in Gegensatz-Relation anwendbar sind (Konsistenz)**
- **Vererbung definitorischer Merkmale**
- **beschreibende vs. erklärende Merkmale je nach Wissensstand**

Begriffsrepräsentation:

- **Merkmale und Begriffe sind eigentlich nicht verschieden!**

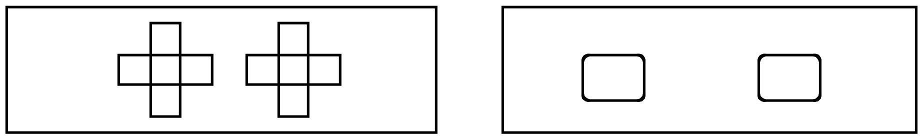
Anwendung des Begriffs Klassifikation



a) ist schwarz: +

b) besteht aus 2 Teilobjekten: -

c) ist schwarz und besteht aus 2 Teilobjekten: -



a) ist schwarz: -

b) besteht aus 2 Teilobjekten: +

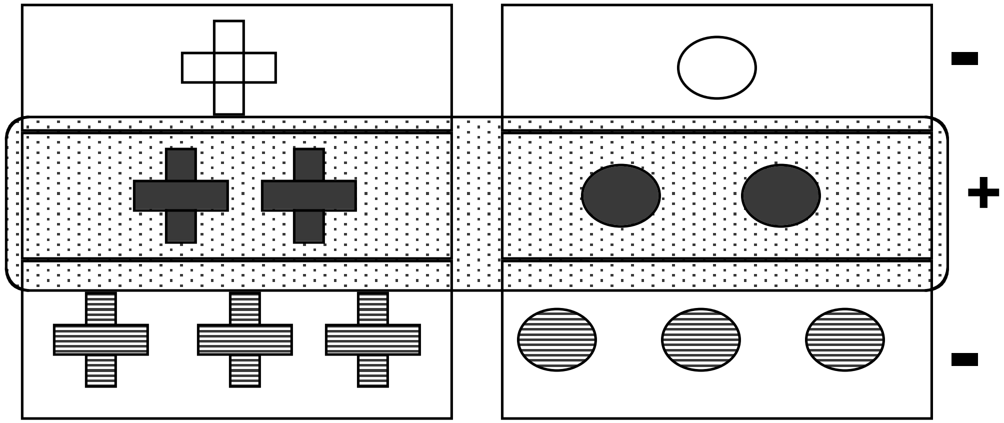
c) ist schwarz und besteht aus 2 Teilobjekten: -

Lernmenge/Testmenge

Lernmenge:

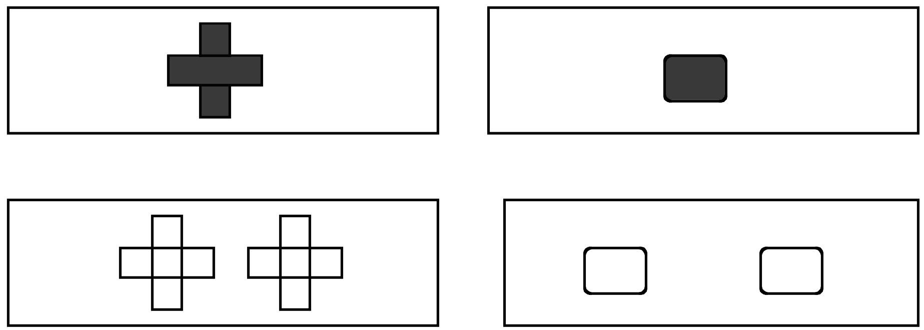
Menge von Beispielen = klassifizierte Beobachtungen

Lernen einer Definition



Testmenge:

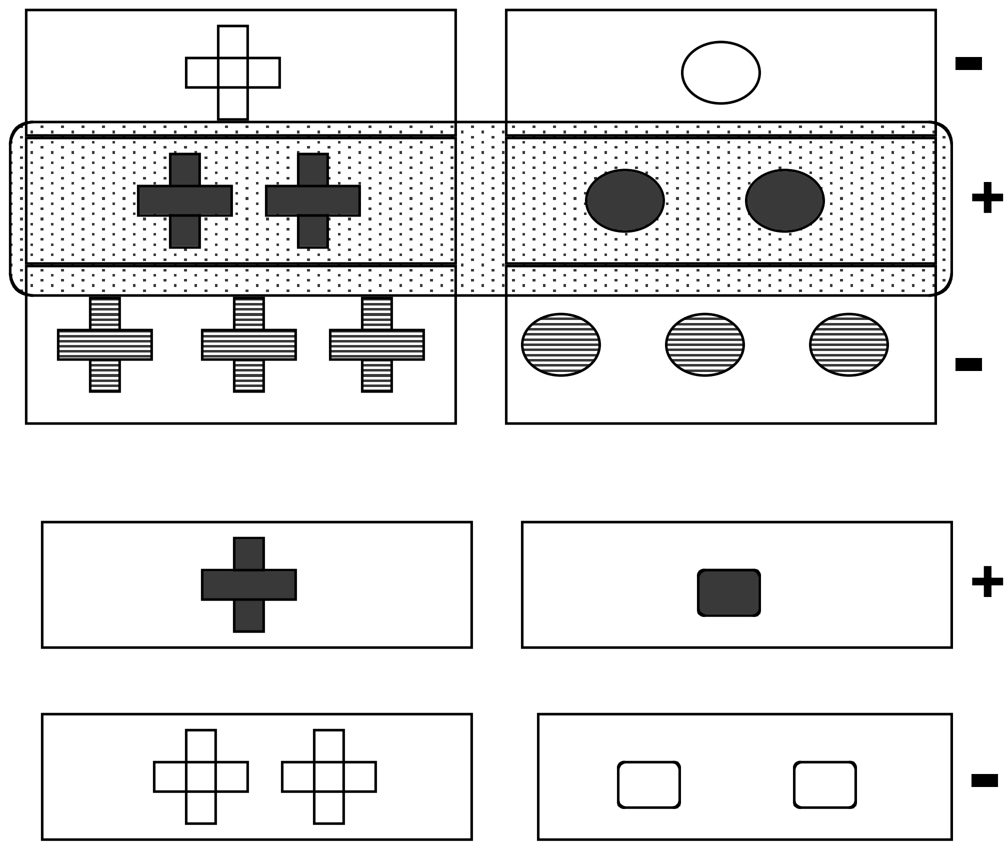
Menge von Beispielen, bei denen die tatsächliche Klassifikation mit der von der Definition vorhergesagten verglichen wird



korrekt, vollständig

korrekt ist eine Definition, wenn sie kein negatives Beispiel abdeckt;

vollständig ist eine Definition, wenn sie alle positiven Beispiele abdeckt



a) ist schwarz: korrekt und vollständig

b) besteht aus 2 Teilobjekten:

nicht korrekt

nicht vollständig

c) ist schwarz und besteht aus 2 Teilobjekten:

korrekt

unvollständig

Sprachen

LE für die Beobachtungen oder Beispiele

zulässig!

LH für die Begriffsdefinitionen bzw. Hypothesen

Allgemeinheitsordnung

ALLES

ab) ist schwarz oder besteht aus 2 Teilobjekten

a) ist schwarz, **b) besteht aus 2 Teilobjekten,**

c) ist schwarz und besteht aus 2 Teilobjekten

NICHTS

Cross Validation

Man teile alle verfügbaren Beispiele in n Mengen auf, z.B. n= 10.

Für i=1 bis i=n:
Wähle die i-te Menge als Testmenge und
die restlichen n-1 Mengen als Lernmenge.
Messe Korrektheit und Vollständigkeit auf der Testmenge.

Bilde das Mittel der Korrektheit und Vollständigkeit über allen n Lernläufen.
Das Ergebnis gibt die Qualität des Lernergebnisses an.

Suchverfahren

Top-down:
Beginne mit der allgemeinsten Hypothese;
solange noch negative Beispiele abgedeckt werden -- spezialisieren!
sonst -- anhalten.

Bottom-up:
Beginne mit den speziellsten Hypothesen;
solange noch positive Beispiele nicht abgedeckt werden -- generalisieren!
sonst -- anhalten.

Bi-direktionale Suche
Versionenraum

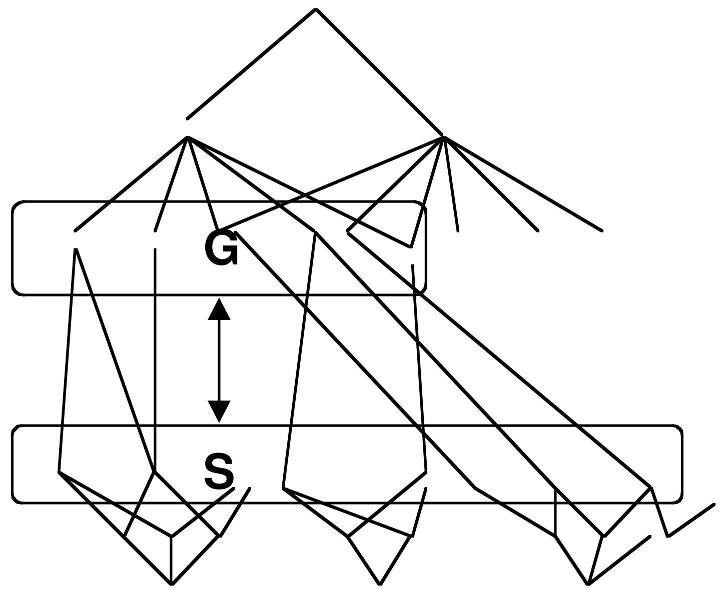
G: { g |$\forall$ p $\in$ P: covers (g, p),
$\forall$ n $\in$ N: $\neg$ covers (g, n),

es gibt kein g', das generelller ist als g und konsistent mit P und N ,

es gibt ein s $\in$ S, das spezieller ist als g }

S: {s |$\forall$ p $\in$ P: covers (s, p),
$\forall$ n $\in$ N: $\neg$ covers (s, n),

es gibt kein s', das spezieller ist als s und konsistent mit P und N
es gibt ein g $\in$ G, das genereller ist als s }



Verfahren

Initialisiere G mit den generellsten, S mit den speziellsten Begriffen.

Solange G und S disjunkt sind, lies ein neues Beispiel e ein.

Falls $e \in N$:

- **entferne alle s aus S, die e abdecken;**
- **für alle $g \in G$, die e abdecken:**
  - **entferne g aus G,**
  - **füge alle schrittweisen Spezialisierungen h von g hinzu, so daß gilt:**
    - **h deckt nicht e ab,**
    - **es gibt ein $s \in S$, das spezieller ist als h;**
  - **entferne alle $g \in G$, die echt spezieller sind als ein anderes $g' \in G$**

Verfahren (cont'd)

Falls e $\in$ P:

- **entferne alle g aus G, die e nicht abdecken;**
- **für alle s $\in$ S, die e nicht abdecken:**
  - **entferne s aus S,**
  - **füge alle schrittweisen Generalisierungen h von s hinzu, so daß gilt:**
    - **h deckt e ab,**
    - **es gibt ein g $\in$ G, das genereller ist als h;**
  - **entferne alle s $\in$ S, die echt genereller sind als ein anderes s' $\in$ S**

Sobald S = G , so ist das Ergebnis gefunden:

- **gib die Hypothese aus und halte an!**

Sonst: Mißerfolg (zu wenig Beispiele)

Beispiel

?- learn.
    POSITIVES BEISPIEL?
[2Räder, Benzin]
    G= {[-mittel, -art]}   S= {[2Räder, Benzin]} BEISPIEL?
[3Düsen, Hand] n
    G= { [Boden-, -art], [-mittel, Motor] } S= { [2Räder, Benzin]} BEISPIEL?
[4Räder, Diesel] p
    G= { [Boden-, -art], [-mittel, Motor] } S= { [Boden-, Motor] } BEISPIEL?
[2Räder, Fuß]   p
    G= { [Boden-, -art] } S={ [Boden-, -art] }
    LÖSUNG: [Boden-, -art]

Eigenschaften des Verfahrens

Der Versionenraum enthält alle Hypothesen S und G und dazwischen.

Die Mengen S und G enthalten alternative Hypothesen.

Gelernt wird eine (die richtige) Hypothese, wenn

- **die Beispiele keine Fehler enthalten,**
- **die richtige Hypothese in LH ausdrückbar ist,**
- **der Hypothesenraum sich (halb-)ordnen läßt**
- **und die Notwendigkeit zu generalisieren und zu spezialisieren gegeben ist.**

LH könnte Disjunktionen und Negation von Merkmalen erlauben.

dann wäre z.B. S: {[p1 v p2 v p3]}

und G: {[¬ (n1 v n2)] }

Die einzige Lösung wäre die Disjunktion aller Beispiele und die Negation aller Gegenbeispiele, falls sie sich alle aufzählen lassen:

S = G { [(p1 v p2 v .... pn) & ¬ (n1 v n2 v ... nm)]}

Funktionsapproximation

Gegeben:

Beispiele, die anhand einer Wahrscheinlichkeitsverteilung P (X) erzeugt wurden und

mit einem Funktionswert Y = t(X) versehen sind. t(X) kann mit einer bestimmten Wahrscheinlichkeit falsch sein (verrauschte Daten).

Hypothesensprache LH

Ziel:

Eine Hypothese H = h(X) aus LH, die das Fehlerrisiko minimiert

Risiko:

$$R(H) = \sum_{i=1}^{n} Q(X_i, H)\ P(X)$$

Fehler Q(X, H)

P(X) Wahrscheinlichkeit, daß das Beispiel X aus der Menge aller Beobachtungen gezogen wird

Q(X, H) Fehlerfunktion, die angibt, wie genau H die Funktion approximiert

Klassifikation:

t(X) ist jeweils eine Klassenbezeichnung

Q(X, H) ist 1, falls $t(X) \neq h(X)$,
0, falls $t(X) = h(X)$

Regression:

Annäherung an eine reellwertige Funktion

quadratischer Fehler

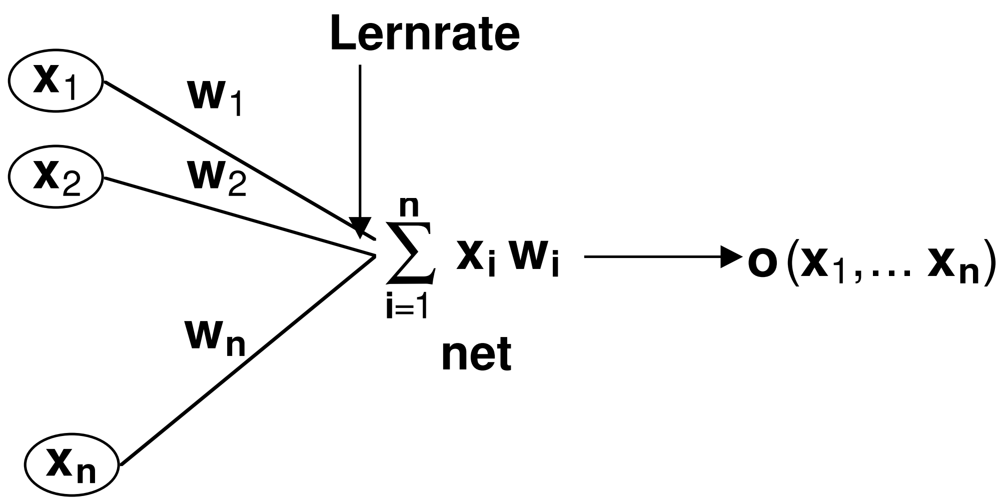
$$Q(X, H) = (t(X) - h(X))^2$$

Minimierung des beobachteten Fehlers

Da wir die tatsächliche Funktion t(X) nicht kennen, können wir nur eine hinreichend große Lernmenge nehmen und für diese den Fehler minimieren.

empirical risk minimization

Perzeptron



linear: keine Funktion o -- die Summe der Produkte wird direkt ausgegeben

Schwellwertfunktion: o liefert 1, falls die Summe größer s, sonst -1

sigmoide Funktion: o liefert einen Wert zwischen 0 und 1

Perzeptron-Lernregel

1) Beginne mit irgendwelchen beliebigen Gewichten.
2) Gehe alle Beispiele durch und vergleiche den vom Neuron errechneten Ausgabewert mit t(X) im Beispiel.
3) Korrigiere jedes Gewicht nach der Regel:

$$w_i := w_i + \Delta w_i$$

Dabei ist $\Delta w_i = n\,(t-o)\,x_i$

n ist die Lernrate , z.B. 0.1

t ist der Wert im Beispiel, o der Wert, den das Neuron ausgibt

x ist der Eingabewert -- muß als Zahl dargestellt werden.

4) Sind alle Beispiele richtig klassifiziert, dann STOP, sonst 2).

Vorhersage

Wann arbeitet der Nachbar im Garten?

Gegeben:

Wahrnehmung des Wetters

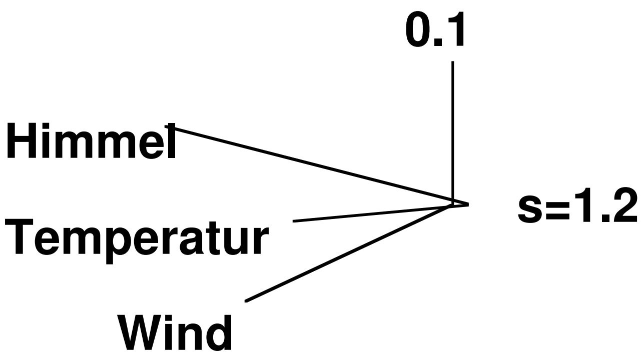
- **Himmel ist wolkenlos oder bewölkt.**
- **Temperatur ist warm, mittel oder kalt.**
- **Es ist windig oder nicht.**

Beobachtung des Nachbarn

- **Er arbeitet im Garten oder nicht.**

Tatsächlich arbeitet er nur im Garten, wenn es warm ist.

Beispiel



e1: 1, 2, 1 t(e1) = 1

e2: 1, 1, 1 t(e2) = -1

e3: 0.1, 1, 0.1 t(e3) = -1

mit Himmel: 1 = wolkenlos, 0.1 = bewölkt

Temperatur: 2 = warm, 1 = mittel, 0.1 = kalt

Wind: 1 = windstill, 0.1 = windig

Wenn das Perzeptron für e1 ausgibt o = -1,
dann sind die Gewichte offensichtlich zu niedrig, z.B. bei allen Eingaben 0.1.
Sie werden dann auf im ersten Schritt erhöht, weil

$\Delta w_1 = 0.1\ (1 - (-1))\ 1 = 0.2$ und $\Delta w_2 = 0.1\ (1 - (-1))\ 2 = 0.4$

Wenn das Perzeptron für e2 o = 1 ausgibt,
dann sind die Gewichte offensichtlich zu hoch, z.B. bei allen Eingaben 0.8.
Sie werden dann im ersten Schritt auf 0.6 erniedrigt, weil

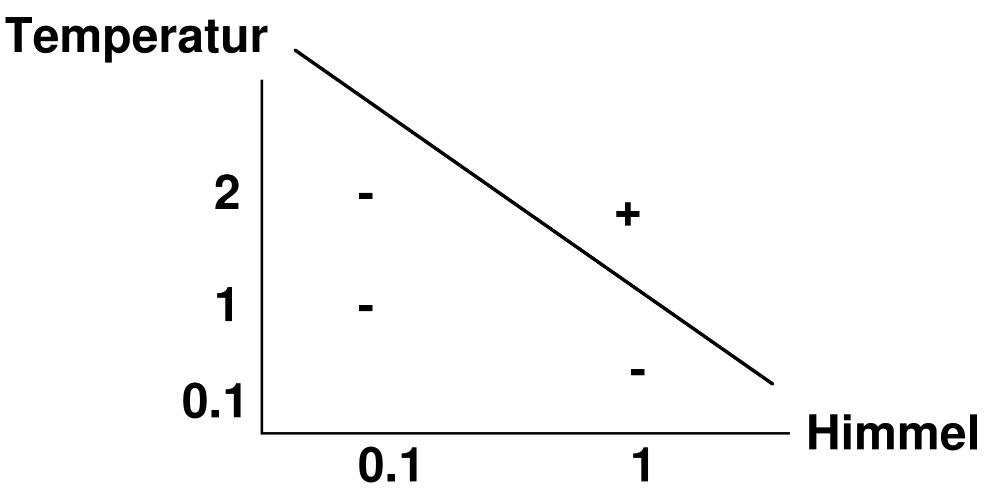
$\Delta w = 0.1\ ((-1) - 1)\ 1 = -0.2 \qquad -0.2 + 0.8 = 0.6$

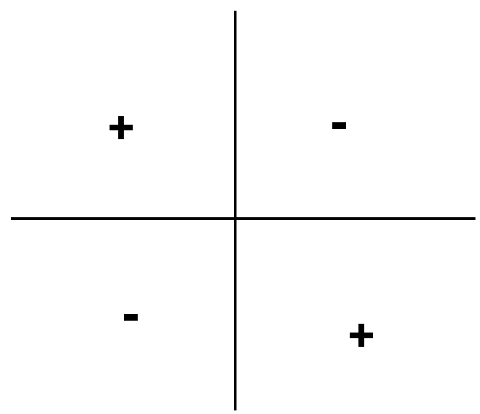
linear separierbare Beispiele

Die Lernregel findet geeignete Gewichte, so daß die lokalen Minima mit den globalen überein stimmen, wenn das Perzeptron die Funktion darstellen kann. (Rosenblatt 1959)

Das ist z.B. bei den Boolschen Funktionen AND, OR, NAND, NOR der Fall.

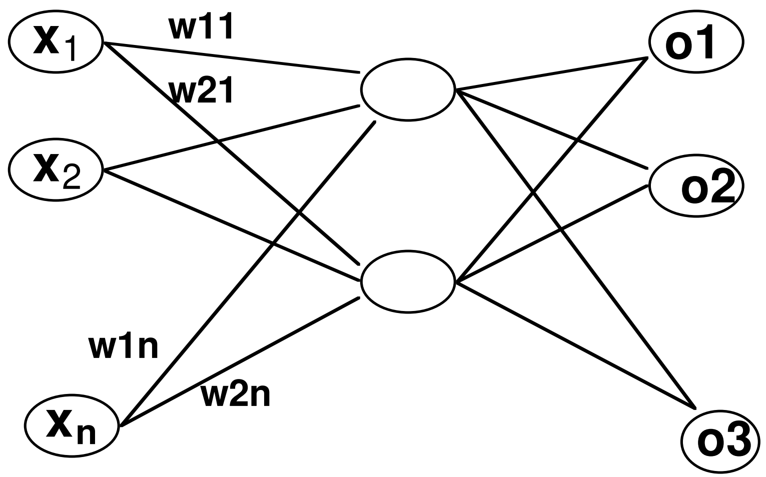
Bei XOR klappt es nicht. (Minsky, Papert 1969)





XOR

Feedforward-Neuronale Netze



$$net_j = \sum_i w_{ji} x_{ji}$$

Vorwärtsdurchlauf:

Berechnung der Gewichte der Ausgabeschicht aus der Mittelschicht, die wiederum aus der Eingabeschicht berechnet wurde

Rückwärtsdurchlauf:

Anpassen der Gewichte aufgrund bekannter Werte der Ausgabeschicht

Backpropagation

Sigmoide Funktion statt einfachen Schwellwertes.

Minimierung des quadratischen Fehlers:

$$F = \sum_e (t_e - o_e)^2$$

Da wir mehrere Ausgaben haben, wird die Summe über den Fehlern aller Ausgabeneuronen gebildet.

Der Fehler bei Neuronen k der Ausgabeschicht ist fast so wie gehabt, nur daß t-o noch mit o (1 - o) multipliziert wird -- wegen der sigmoiden Funktion.

$$\delta_k = o_k \, (1 - o_k) \, (t_k - o_k)$$

Bei Neuron j wird für jeden Eingabewert x_{ji} das entsprechende Gewicht angepaßt:

$$w_{ji} := w_{ji} + \Delta w_{ji}$$

$$\Delta w_{ji} = n \, \delta_j \, x_{ji}$$

Mittelschicht

Wir dürfen aber nicht nur die Gewichte, die zur Ausgabe führen, verändern, sondern müssen auch die Gewichte zur Mittelschicht anpassen.

Wir haben aber keinen Zielwert t.

Wir wissen nur, zu welchem Fehler in der Ausgabeschicht das Neuron beigetragen hat: δ_k ist der Fehler des Ausgabeneurons k.

Und wir wissen, in welchem Maße es beigetragen hat -- sein Gewicht der Verbindung zu dem Ausgabeneuron: w_{kh} ist das Gewicht der Verbindung von h nach k.

Der Fehler eines Mittelschicht-Neurons ist die Summe der gewichteten Fehler, zu denen es beigetragen hat:

$$\delta_h = o(e)\ (1 - o(e))\ \Sigma\ w_{kh}\ \delta_k$$

Wir passen die Gewichte wieder entsprechend an.

Herleitung der Delta-Regel

Wir wollten den quadratischen Fehler der Summe aller Ausgabeneuronen minimieren durch ein Gradientenverfahren.

$$\Delta w_{ji} = -n \frac{\partial F_e}{\partial w_{ki}}$$

$$F_e = \frac{1}{2} \sum_k (t_k - o_k)^2$$

Wir haben für die Ausgabeneuronen den bisherigen Gewichten hinzugefügt:

$$w_{ki} := w_{ki} + \Delta w_{ki}$$

$$\Delta w_{ki} = n\, \delta_k\, x_{ki}$$

$$\delta_K = o_k\,(1 - o_k)\,(t_k - o_k)$$

Ist wirklich beides gleich?

Ableitung

$o(x) = 1/(1+e^{-x})$

$-(1/(1+e^{-x})^2)\,(1+e^{-x})'$

$-(1/(1+e^{-x})^2)\quad -e^{-x}$

$1/(1+e^{-x})\;\;(e^{-x})/(1+e^{-x})$

$1 - o(x) = 1 - 1/(1+e^{-x})$

$= (e^{-x})/(1+e^{-x})$

Kettenregel $1/y\;\; y'$

$(1/y)' = -(1/(y^2))$

$(a+b)' = a' + b'$

$1' = 0$ und $e^{-x}{}' = -e^{-x}$

Eigenschaften

- **Backpropagation findet nicht notwendigerweise globale Minima.**

  Es kommen aber selten lokale Minima bei allen Gewichten auf einmal vor. Deshalb bieten meistens die anderen Gewichte einen "Ausweg".

- **Neuronale Netz sind abhängig von ihrer Topologie.**

Alle Boolschen Funktionen können durch ein Netzwerk mit einer Mittelschicht dargestellt werden.

Je mehr Eingabeneuronen das Netz hat, desto mehr Neuronen der Mittelschicht können nötig werden:

für jede Kombination von Eingabewerten konstruiert man ein Neuron der Mittelschicht, das genau bei dieser Eingabe aktiviert wird;

die Ausgabe wird als ODER realisiert, das zu jeder Eingabe die gewünschte Ausgabe liefert.

Bei weniger als exponentiell vielen Neuronen in der Mittelschicht kann die Zielfunktion vielleicht nicht gelernt werden.

Anwendung

- **Ein-/ Ausgabe sind als Vektor von Zahlen darstellbar.**
- **Die Lernmenge darf Fehler enthalten.**
- **Lange Lernzeiten sind akzeptabel.**
- **Die Anwendung des Lernergebnisses soll schnell sein.**
- **Das Lernergebnis muß nicht verständlich sein.**

==> Erwerb von Fertigkeiten

Beispiel

Aus 624 Fotos von Gesichtern soll gelernt werden, wohin das Gesicht schaut.

Jedes Bild besteht aus 120 x 128 Bildpunkten in 256 Graustufen.

Eingabe (LE):

Projektion auf 30 x 32 Bildpunkte	Reduktion der Komplexität
960 Neuronen für die 960 Bildpunkte	feste Vektorlänge
Projektion der Grauwerte auf [0, 1] der Mittelschicht und Ausgabe	gleiches Intervall wie bei Neuronen

LH

Mittelschicht:

Alle Einheiten mit allen der folgenden Schicht verbunden

1 Mittelschicht

3 Neuronen -- 90% korrekt, 5 Min. Lernzeit (260 Bilder)

30 Neuronen - 92% korrekt, 1 Std. Lernzeit (260 Bilder)

Ausgabe:

4 Neuronen für die Blickrichtungen links, rechts, geradeaus, nach oben

0.9, 0.1, 0.1, 0.1 statt 1, 0, 0, 0 (sigmoide Funktionen)

Ergebnis ist das Neuron mit dem höchsten Gewicht.

Abstand des höchsten zum nächst höchsten Gewicht zeigt Konfidenz an!

Alternativ:

1 Neuron mit verschiedenen Werten für die Blickrichtungen

Nur ein Viertel der Gewichte zur Approximation der Funktion

Dynamische Topologie

Lernen mit einem Netzwerk ohne Mittelschicht

Einfügen von Neuronen der Mittelschicht
bis akzeptable Fehlerrate erreicht ist.

Cascade-correlation (Fahlmann, Lebiere 1990)

Verhindern von Overfitting

1) Gewichtreduktion bei jeder Iteration -- Strafe für die Anpassung an die Daten

2) Netzwerk bereits bei jeder Iteration auf Testmenge anwenden,

Gewichte speicher, die bisher bestes Ergebnis über der Testmenge ergaben,

Lernen beenden, falls der Fehler über der Testmenge größer wird.

Lernbarkeitstheorie

LH

- **Größe des Hypothesenraums, Schwierigkeit der Suche**

Zusicherungen an das Lernergebnis

- **Korrektheit, speziellste Generalisierung, generellste Diskriminierung**

Konvergenz

- **Wann, unter welchen Umständen, mit welcher Wahrscheinlichkeit wird ein angemessenes Lernergebnis ausgegeben?**

Wieviele Beispiele werden für das Erreichen eines angemessenen Lernergebnisses gebraucht?

Wahrscheinlich annähernd korrektes Lernen (PAC)

X in LE: alle möglichen Beobachtungen, repräsentiert in LE

- Menschen, beschrieben durch Name, Alter, Größe, Beruf, Sportart, ...

x in X: ein Beispiel

- Uta, 36, 175, Informatikerin, Ski, ...

C in LH: mögliche Begriffe, die zu lernen sind

c in C: Zielbegriff

- Ski-fahrende InformatikerInnen
- als Funktion: X --> {0,1}

D: Distribution der Beispiele -- hier: nicht bekannt, aber fest

Beispiele werden gemäß D zufällig gezogen

- wenn es viele Ski-fahrende Informatikerinnen gibt, werden auch viele als Beispiel vorkommen

PAC-learning

Eine Begriffsklasse C mit Bezug auf Beispiele X ist PAC-lernbar durch einen Lernalgorithmus L und einen Hypothesenraum H, wenn für alle c in C, Verteilungen D über X, $0 < \varepsilon < 1/2$ und $0 < \delta < 1/2$

- **L mit einer Wahrscheinlichkeit von mindestens $1 - \delta$**
- **eine Hypothese h aus H ausgibt, deren Fehler über der Verteilung D nicht größer ist als ε**
- **in einer Zeit, die durch ein Polynom über $1/\varepsilon$, $1/\delta$, $|c|$ und $|x|$ begrenzt ist.**

L liefert also
wahrscheinlich
ein annähernd korrektes Ergebnis
in poynomieller Zeit ab.

L muß aus nur polynomiell vielen Beispielen lernen,
wobei die Verarbeitung jeden Beispiels nur polynomielle Zeit benötigt.

Wieviele Beispiele braucht man im Versionenraum?

- Der Hypothesenraum ist endlich. $|H|$
- Alle Hypothesen in H sind konsistent mit den Beispielen.

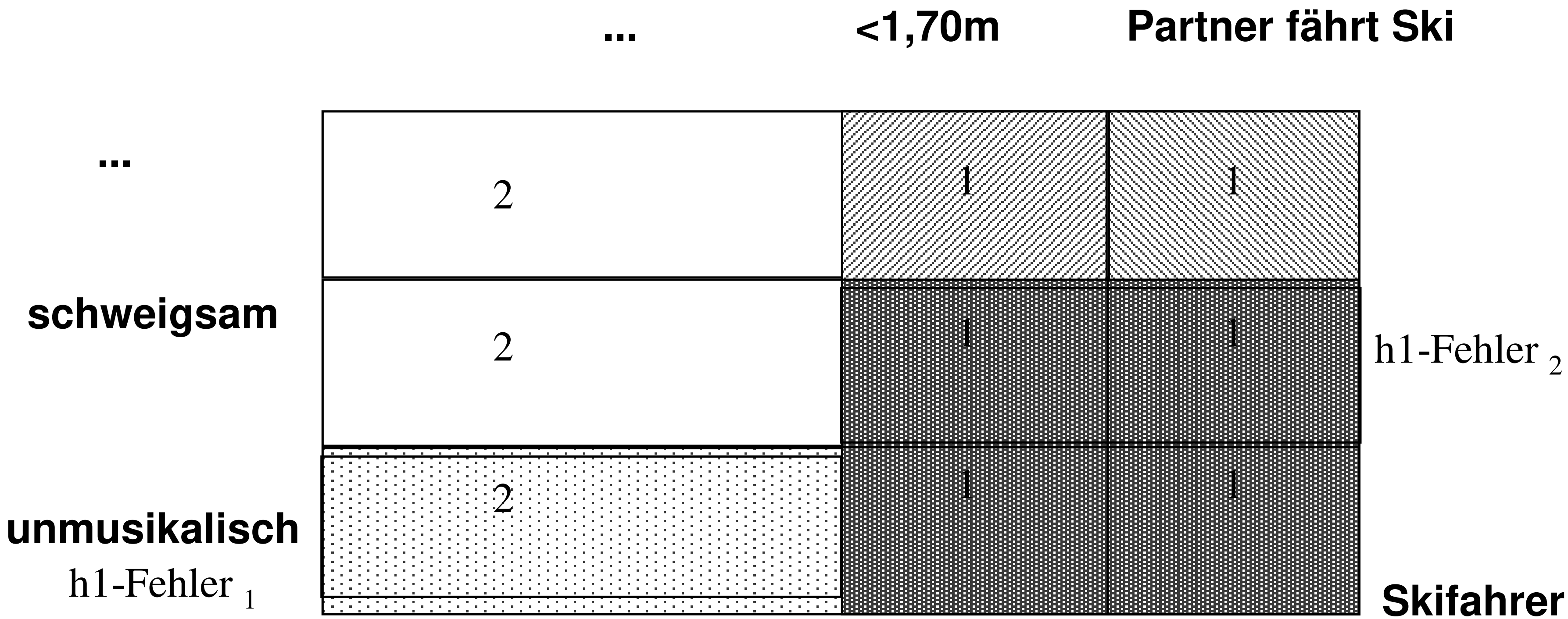
Fehler einer Hypothese: Wahrscheinlichkeit, daß ein Beispiel gezogen wird, das eine andere Klassifikation hat als die Hypothese angibt.

Nach wievielen Beispielen gibt das Verfahren mit der Wahrscheinlichkeit $1 - \delta$ eine Hypothese aus, deren Fehler höchstens ε beträgt?

Wann brauchen wir noch Beispiele, um falsche Hypothesen auszuschließen?

Falsche Hypothesen sind korrekt bzgl. der Beispiele, aber in Wirklichkeit nicht korrekt, werden also möglicherweise durch weitere Beispiele ausgeschlossen.

Beispiel



h1: unmusikalisch --> ski Fehler:4/12

h2: <1,70m --> ski Fehler:3/12

h3: unmusikalisch & <1,70m --> ski Fehler:3/12

h4: (unmusikalisch & (<1,70m v pfs)) v (schweigsam & (<1,70m v pfs)) --> ski

Fehler: 0

Obere Schranke

H: $\{h_1, \ldots, h_k\}$

Die Wahrscheinlichkeit, daß eine konsistente Hypothese mit Fehler $> \varepsilon$

konsistent mit einem neuen Beispiel ist, ist höchstens $(1 - \varepsilon)$.

Die Wahrscheinlichkeit, daß die Hypothese mit m Beispielen konsistent ist, beträgt also höchstens

$$(1 - \varepsilon)^m$$

Die Wahrscheinlichkeit, daß mindestens eine der Hypothesen mit m Beispielen konsistent ist (obwohl sie tatsächlich einen Fehler $> \varepsilon$ hat), beträgt höchstens

$$|H|\,(1 - \varepsilon)^m$$

Da $0 \leq \varepsilon \leq 1$ ist $(1 - \varepsilon) \leq e^{-\varepsilon}$ und wir begrenzen die Wahrscheinlichkeit, daß m Beispiele nicht die schlechten Hypothesen entfernen, durch

$$|H|\,e^{-\varepsilon m}$$

Für das PAC-Lernen muß dies kleiner sein als δ.

Anzahl der Beispiele

Wahrscheinlichkeit, daß m Beispiele nicht die schlechten Hypothesen entfernen, kann nicht größer sein als

$$|H|\, e^{-\varepsilon m}$$

Für das PAC- Lernen:

$$|H|\, e^{-\varepsilon m} \leq \delta$$

Also brauchen wir mindestens m Beispiele:

$$m \geq 1/\varepsilon \; (\ln |H| + \ln (1/\delta))$$

Beispiel -- cont'd

$m \geq 45$ bei $\varepsilon = 0.2$ und $\delta = 0.1$

¬ unmusikalisch, ¬ schweigsam, >1,70m, ¬ pfs, ¬ ski
¬ unmusikalisch, ¬ schweigsam, >1,70m, ¬ pfs, ¬ ski
¬ unmusikalisch, ¬ schweigsam,<1,70m, ¬ pfs, ¬ ski
¬ unmusikalisch, ¬ schweigsam, >1,70m, pfs, ¬ ski
¬ unmusikalisch, schweigsam, >1,70m, ¬ pfs, ¬ ski
¬ unmusikalisch, schweigsam, >1,70m, ¬ pfs, ¬ ski
¬ unmusikalisch, schweigsam, <1,70m, ¬ pfs, ski
¬ unmusikalisch, schweigsam, >1,70m, pfs, ski
unmusikalisch, ¬ schweigsam, >1,70m, ¬ pfs, ¬ ski
unmusikalisch, ¬ schweigsam, >1,70m, ¬ pfs, ¬ ski
unmusikalisch, ¬ schweigsam, < 1,70m, ¬ pfs, ski
unmusikalisch, ¬ schweigsam, >1,70m, pfs, ski

$|H| = 3^6 = 729$
bei 6 Attributen, die die Werte 0,1,egal annehmen

$m \geq 1/0{,}2\ (\ln 729 + \ln (1/0{,}1))$
$m \geq 5\ (6{,}59 + 2{,}3) = 44{,}46$

mehrmals ziehen, um jedes Beispiel bei beliebigen Verteilungen zu sehen

Vapnik-Chervonenkis Dimension

H **Hypothesenraum über X**

S **Beispiele -- m-elementige Teilmenge von X**

Wenn H für alle Teilmengen von S eine Hypothese hat, die nur diese Teilmenge abdeckt, dann wird S von H zerschmettert.

Die VCdim(H) ist die größte Anzahl von Elementen in S, die H zerschmettern kann.

VCdim (H) = max {m: ∃ S ⊆ X, |S| = m, H zerschmettert S}

Wenn es keine maximale Kardinalität von S gibt, ist VCdim unendlich.

Wenn H endlich ist, gilt:

VCdim (H) ≤ $\log_2$ (|H|)

|S| = m erfordert 2^m Hypothesen, bzw.

m = log2 (2^m)

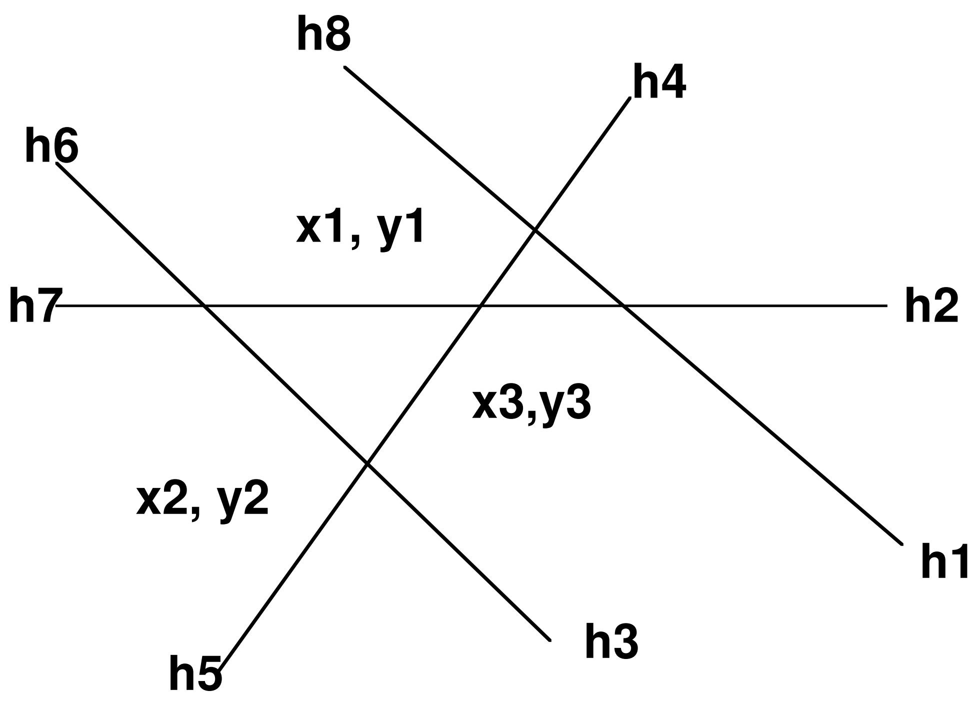
Beispiel

X **Punkte in der Ebene**

H **trennende Linien (entspricht einem Perzeptron)**

|S| = 3 erfordert 8 Hypothesen

VCdim ist also mindestens 3.



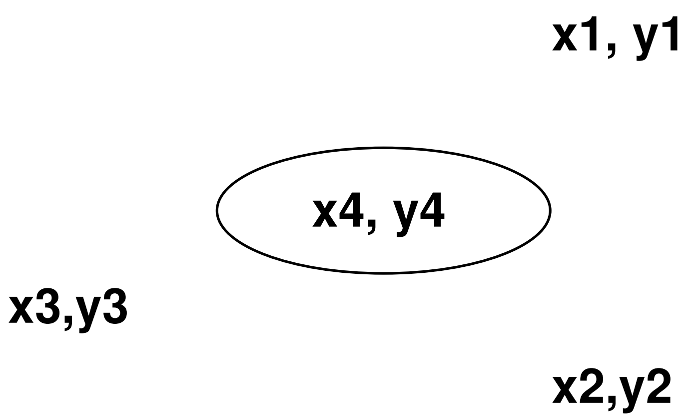
S	h1	h2	h3	h4	h5	h6	h7	h8
(x1,y1)	-	+	-	+	-	+	-	+
(x2,y2)	-	-	+	+	-	-	+	+
(x3,y3)	-	-	-	-	+	+	+	+

Beispiel -- cont'd

X Punkte in der Ebene

|S| = 4 kann nicht zerschmettert werden.

VCdim ist also höchstens 3.



Mit dem Hypothesenraum der
separierenden Linien
kann (x4,y4) nicht
von allen anderen Punkten getrennt werden.

Stichprobengröße bei unendlichem Hypothesenraum

PAC-Lernen und VCdim

obere Schranke

hinreichende Anzahl Beispiele m

$$m \geq 1/\varepsilon \; (8 \; VCdim(H) \; \log_2 (13/\varepsilon) + 4 \log_2 (2/\delta))$$

untere Schranke

nötige Anzahl Beispiele m

$0 < \varepsilon < 1/8$, $0 < \delta < 1/100$ bei weniger als

$$\max [\; 1/\varepsilon \; \log(1/\delta), \; (VCdim(H) - 1) / 32 \varepsilon \;]$$

wird mit mindestens δ Wahrscheinlichkeit eine Hypothese h ausgegeben, deren Fehler $> \varepsilon$.

Zerschmettern von Mengen

C Klasse von Begriffen über X

S Stichprobe in X (Beispiele)

$\Pi_C(S) = \{ c \cap S : c \in C \}$

Menge von Teilmengen von S

$S = \{x1, x2, ..., xm\}$

$\{0,1\}^m \supseteq \Pi_C(S)$ mit $m = |S|$

$\Pi_C(S) = \{(c(x1), ..., c(xm)) : c \in C\}$

Menge von binären Vektoren

Zerschmettern der Beispielmenge

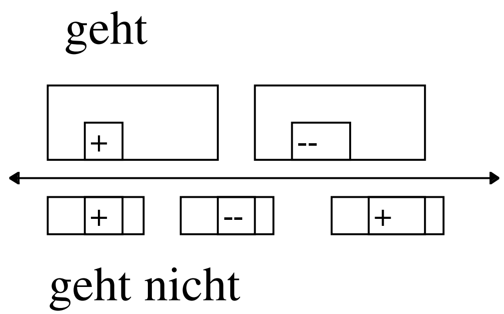
Def.: Wenn $\Pi_C(S) = \{0,1\}^m$ mit $m = |S|$,
dann wird S von C zerschmettert.

$|S| = 3$

0 0 0	0 0 1	0 1 0	0 1 1
1 0 0	1 0 1	1 1 1	1 1 0

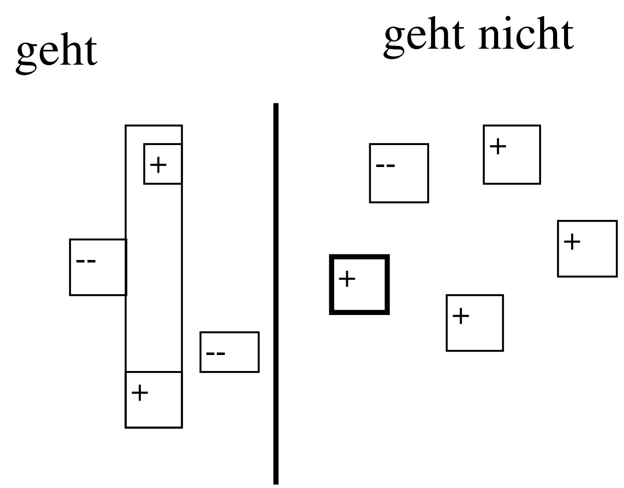
2^m unterschiedliche Verhaltensweisen in S

Beispiele für VCdim



- Intervalle eines Zahlenstrangs können nur in 2 Mengen zerschmettert werden.

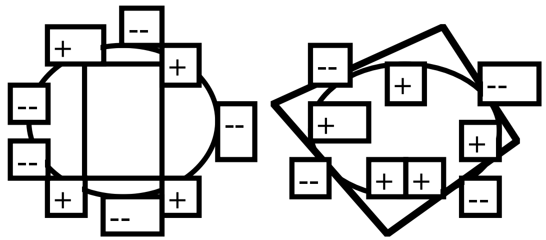
Achsenparallele Rechtecke



- Achsenparallele Rechtecke können in einer Ebene manchmal 4 Punkte zerschmettern, nicht aber 5.

Konvexe Polygone

mehr negative Beispiele mehr positive Beispiele



- Für d-gone ist die VCdim 2d+1. Wenn es mehr negative als positive Beispiele gibt, werden die positiven durch Kanten verbunden. Wenn es mehr positive als negative gibt, formen die Tangenten der negativen Beispiele das Polygon.

Übersicht

- VCdim ist eine Eigenschaft der Hypothesensprache: wieviele Elemente kann sie in <u>alle</u> Teilmengen zerlegen.
- $\Pi_C(S)$ ist eine Eigenschaft der Beispiele: in wieviele Teilmengen können wir sie mit einer Hypothesensprache zerlegen.
- $\Pi_C(m)$ gibt die Eigenschaft für die höchste Anzahl von m Beispielen an.

Beschränkung der Verhaltensweisen in S

- Wir wollen $\Pi_C(S)$ in der Größe beschränken.
- Die maximale Größe einer Menge S, die von C zerschmettert wird, notieren wir
  $\Pi_C(m) = \max \{ \mid \Pi_C(S) \mid : |S|=m \}$
- Die Funktion $\Pi_C(m)$ liefert uns die Anzahl der Verhaltensweisen über m Beispielen.
- Wenn $VCdim(C) = \infty$ dann $\Pi_C(m) = 2^m$

Polynomielle Beschränkung

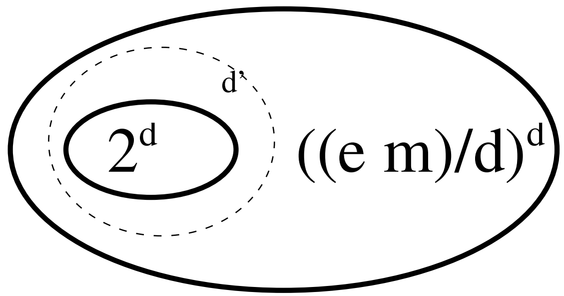
- Wenn VCdim(C) $\neq \infty$ dann $\Pi_C(m) \leq 2^m$.
- $\Pi_C(m)$ wird durch eine Funktion beschränkt und
- die ist beschränkt durch $\sum_{i=0}^{d} \binom{m}{i}$
- Diese Schranke verwenden wir, um die Stichprobenkomplexität für C mit endlicher VCdim zu zeigen.

Zusammenhang VCdim -- Beispiele

VCdim(C) = d, also können bei d Elementen alle 2^d Teilmengen gebildet werden.

Bei m > d Elementen werden nur $((e\,m)/d)^d$ Teilmengen $\Pi_C(m)$ gebildet.

$2^d \leq ((e\,m)/d)^d < 2^m$



Schranke für Π_C (m)

Lemma:

Wenn VCdim(C)=d, dann $\Pi_C(m) \leq \Phi_d(m)$

$\Phi_d(m) = \Phi_d(m-1) + \Phi_{d-1}(m-1)$, wobei

$\Phi_d(0) = \Phi_0(m) = 1$

Beweisskizze

- Ein Beispiel weniger:

  $|\Pi_C(S - \{x\})| \leq \Phi_d(m-1)$

  Hier fallen einige Unterschiede zusammen und zwar gerade $|\Pi_{C'}(S - \{x\})|$ viele.

Also: $|\Pi_C(S)| = |\Pi_C(S - \{x\})| + |\Pi_{C'}(S - \{x\})|$

- Eine VCdim weniger:

  $VCdim(C') \leq d-1$

  $S - \{x\} \supseteq S'$ und S' wird von C' zerschmettert.

Also: $|S'| \leq d-1$, d.h. $|\Pi_{C'}(S - \{x\})| \leq \Phi_{d-1}(m-1)$

Können wir Φ ausrechnen?

JA!

$\Phi_d(m) = \Sigma^d_{i=0} \binom{m}{i}$

mit der binomischen Formel erhalten wir für $m \leq d$:

$\Phi_d(m) = 2^m$

$m > d$: wenn $m > d$, dann $0 \leq d/m \leq 1$

$(d/m)^d \, \Sigma^d_{i=0} \binom{m}{i} \leq (1 + d/m)^m$

jetzt beide Seiten durch $(d/m)^d$ teilen.

Zur Erinnerung

Binomische Formel:

$$(a+b)^d = \Sigma_{i=0} \binom{d}{i} a^i b^{d-i}$$

Natürlicher Logarithmus:

$$e^d = \lim_{m \to \infty} (1 + d/m)^m$$

Vereinfachen

$$\sum_{i=0}^{d} \binom{m}{i} \leq \left(\frac{em}{d} \right)^{d}$$

Was haben wir jetzt?

- Die Menge der Verhaltensweisen (Unterschiede) wächst NICHT exponentiell mit der Menge der Beispiele, wenn VCdim nicht unendlich ist.
- $\Pi_C(m) \leq ((e\, m)/d)^d$

Was wollen wir noch?

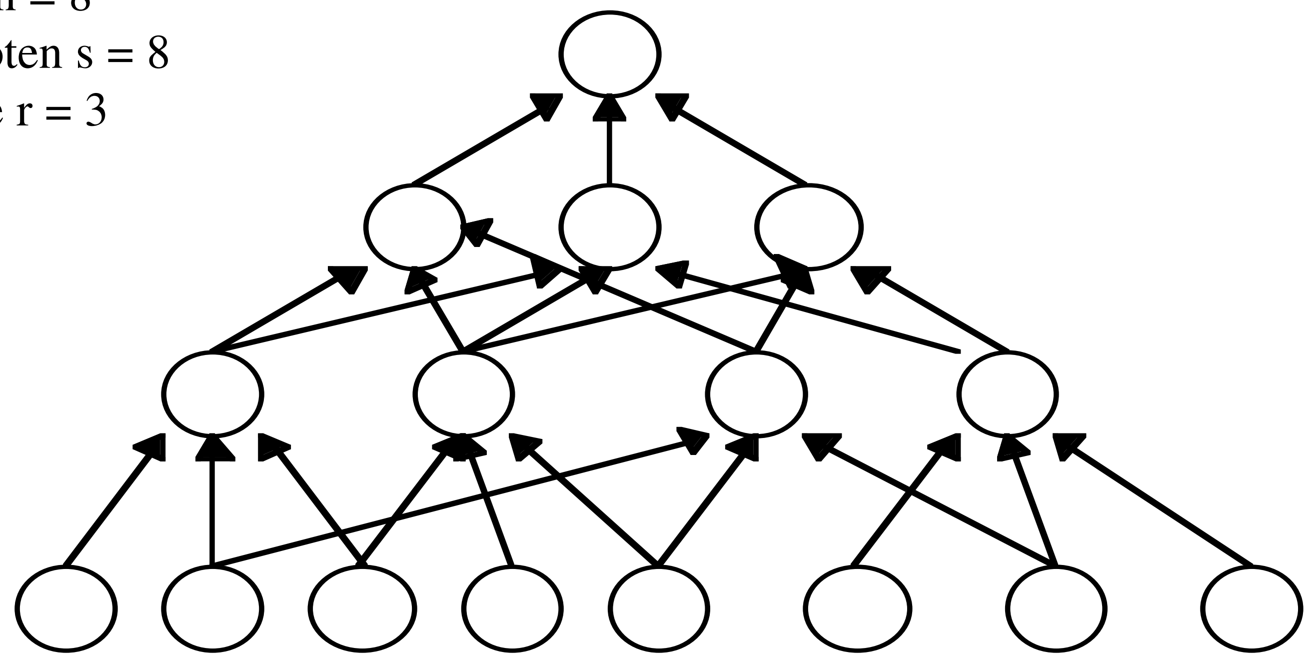
- Können wir diesen Gedankengang umdrehen und herausfinden, wieviele Beispiele wir brauchen, um noch Unterschiede zu finden?

Stichprobenkomplexität neuronaler Netze

- Wir betrachten neuronale Netze als geschichtete Graphen,
- wobei jeder Knoten des Graphen
  - nur r Eingänge hat und
  - ein Schwellwertautomat ist, dessen VCdim r+1 ist.

Geschichteter Graph

Eingabe n = 8
Innenknoten s = 8
Eingänge r = 3

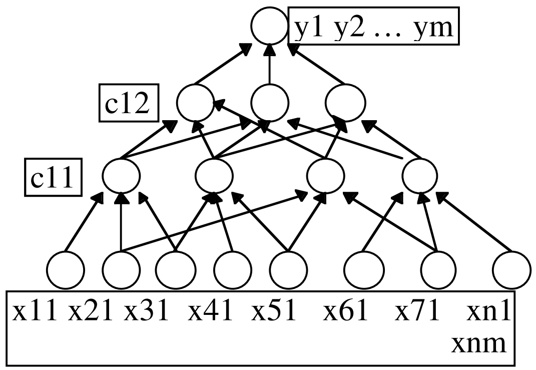


G-Komposition von C

Jeder Knoten hat eine Funktion c_i aus C.

C berechnet r-stellige Funktionen. Jeder Eingabeknoten gibt seinen Wert an die Funktionen der übergeordneten Knoten. Ausgabe ist ein Boolescher Wert.

Alle Begriffe, die durch Beschriftung des Graphen G mit Werten von c_i aus C gebildet werden können, heißen die G-Komposition von C, notiert GC.



CG für m Beispiele.

Theorem 3.7
Kearns & Vazirani 1994

- Sei G ein geschichteter azyklischer Graph mit n Eingabeknoten und $s \geq 2$ internen Knoten, jeder mit r Eingängen.
- Sei C eine Begriffsklasse über R^r mit VCdim(C)=d und GC die G-Komposition von C.
- Dann ist $VCdim(C) \leq 2\, d\, s\, \log(e\, s)$, mit e der Basis des natürlichen Logarithmus'.

Beweisidee

- Wir führen $\Pi_{GC}(S)$ auf $\Pi_{GC}(m)$ zurück und
- wir wissen, daß $\Pi_{GC}(m)$ durch $((e\ m)/d)^d$ beschränkt ist.
- Wenn $((e\ m)/d)^d < 2^m$, dann ist m eine obere Schranke für die VCdim (GC).
- Wir berechnen m, so daß die Ungleichung gilt.

Alle Berechnungen von C über G

Eine Berechnung von C über G:

S: m Eingabevektoren, jeder n lang

c_i an N_i: ein Vektor der Länge m mit von c_i berechneten Werte für m Beispiele

$T_{CG}(S)$: Anzahl verschiedener Berechnungen von C über G,

d.h. solcher, die an irgendeinem Knoten für ein Beispiel einen anderen Wert berechnen.

$$|\Pi_{GC}(S)| \leq T_{CG}(S)$$

- Die Menge positiver Beispiele ist sicherlich in der Menge aller Berechnungen enthalten.

$T_{CG}(S) \leq \Pi_{GC}(m)^s$

- Für alle inneren Knoten:

G’: G ohne Ausgabeknoten N_o

$T_{CG'}(S)$: alle möglichen Berechnungen von G’

Jetzt gibt es nur $\Pi_{GC}(m)$ Vervollständigungen durch verschiedene c_o, so daß wir verschiedene Werte $y_1, ..., y_m$ als Ausgabe haben.

$$T_{CG}(S) \leq T_{CG'}(S) \times \Pi_{GC}(m)$$

Alle Berechnungen von G sind weniger als alle ihre Vervollständigungen bei allen Knoten:

$$|\Pi_{GC}(S)| \leq T_{CG}(S) \leq \Pi_{GC}(m)^s$$

$$\Pi_{GC}(m)^s \leq ((e\, m)/d)^{ds}$$

$((e\ m)/d)^{ds} < 2^m$

- Wenn die Ungleichung gilt, muß $VCdim(GC) \leq m$ sein.
- Wann gilt die Ungleichung?
- $((e\ m)/d)^{ds} < 2^m$ gilt für $m=2\ d\ s\ \log(e\ s)$

Wie man leicht sieht...

$$((e\, m)/d)^{ds} < 2^m$$

$$m = 2\, d\, s\, \log(e\, s)$$

$(2\, es \log(e\, s))^{ds} < 2^{2ds \log(es)}$	einsetzen
$2^{2\, ds\, 1/\log 2\, e\, \log 2(es)}$	$\log_a x = 1/\log_b a \log_b x$
$(e\, s)^{2\, ds/\, \log 2\, e}$	$2^{\log 2(x)} = x$
$((2\, es \log(e\, s))^{((\log 2\, e)/2)})^{ds} < es^{ds}$	$(\log_2 e)/2 > 0$
$(2\, es \log(e\, s))^{((\log 2\, e)/2)} < es$	^{ds} auf beiden Seiten
$e^{1/2} es^{(\log 2\, e)/2} (\log es)^{(\log 2\, e)/2} < es$	$(\log_2 e)/2$ anwenden
$e^{1/2} es^{(\log 2\, e)/2} < es^{1-\, 1/2 \log 2\, e}$	
$\log(e\, s)^{0,2} < es^{0,9}$	oder so ähnlich...

Stichprobenkomplexität Neuronaler Netze

- Die Funktionen in C an jedem Knoten im Graph G seien Schwellwertfunktionen.
- Deren Eingabe sei der r-stellige Gewichtsvektor.
- VCdim(Schwellwertfunktionen) = r +1=d.

$$m \geq 1/\varepsilon \, (\, 4 \log(2/\delta) + 8 \; 2ds \log(es) \log(13/\varepsilon))$$

Stützvektormethode
support vector machines

Gegeben: eine Menge von n Beispielen (x, y), wobei x ein numerischer Vektor ist und $y=+1$ oder $y=-1$

Finde: eine optimale Hyperebene, die die positiven von den negativen Beispielen trennt.

Optimal bedeutet dabei nicht nur kleinen Fehler, sondern auch Einfachheit.

Klassifikation

LH: Regeln der Form

$$h(\vec{x}) = sign\left(b + \sum_{i=1}^{n} w_i\, x_i\right) = sign(\vec{w} \cdot \vec{x} + b)$$

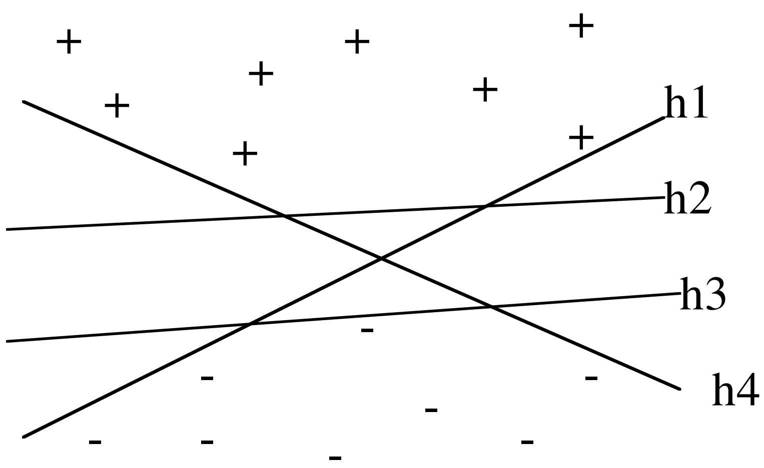
im n-dimensionalen Raum von LE.

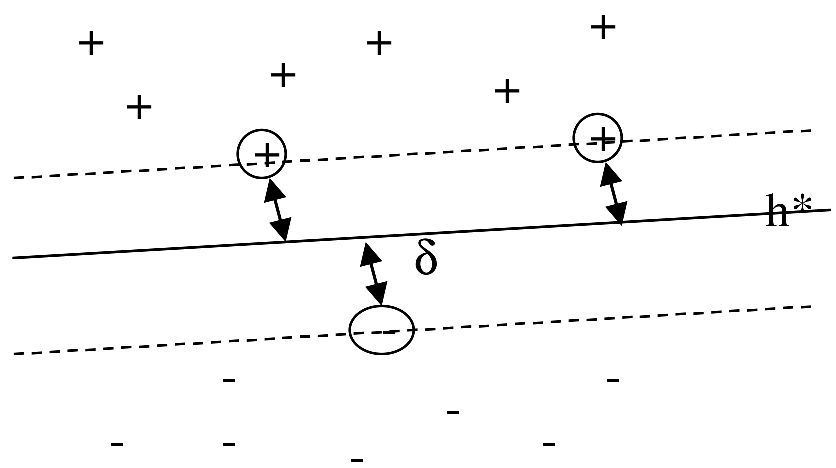
$sign(\vec{w}\,\vec{x} + b) = 1$ positive Beispiele

$sign(\vec{w}\,\vec{x} + b) = -1$ negative Beispiele

Die Gewichte der Merkmale w und den Schwellwert b gilt es zu finden.

Optimale Hyperebene





Optimierungsproblem

$$\textit{finde} \quad \vec{w}, b$$

$$\textit{so dass } \delta \text{ max}$$

$$\textit{und} \quad y_1 \frac{1}{\|\vec{w}\|}\left[\vec{w} \cdot \vec{x}_1 + b\right] \geq \delta, \cdots, y_n \frac{1}{\|\vec{w}\|}\left[\vec{w} \cdot \vec{x}_n + b\right] \geq \delta$$

$$\|\vec{w}\| = \sqrt{\vec{w} \cdot \vec{w}}$$

Bei Stützvektoren ist $y_j \frac{1}{\|\vec{w}\|}\left[\vec{w} \cdot \vec{x}_j + b\right] = \delta$

Lösung (Skizze)

Die Länge des Gewichtsvektors beeinflusst nicht die Lage der optimalen Hyperebene. Daher kann sie fixiert werden:

$$\delta = \frac{1}{\|\vec{w}\|}$$

Dann ist die Maximierung von δ gleich der Minimierung von $\vec{w} \cdot \vec{w}$

$$\textit{finde} \quad \vec{w}, b$$

$$\textit{so dass } \vec{w} \cdot \vec{w} \text{ min}$$

$$\textit{und} \quad y_1[\vec{w} \cdot \vec{x}_1 + b] \geq 1, \cdots, y_n[\vec{w} \cdot \vec{x}_n + b] \geq 1$$

Quadratisches Programm in Normalform

$$\textit{finde} \quad \alpha_1, \cdots, \alpha_n$$

$$\textit{so dass} - \sum_{i=1}^{n} \alpha_i + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \left(\vec{x}_i \cdot \vec{x}_j \right) \text{ min}$$

$$\textit{und} \quad \sum_{i=1}^{n} y_i \alpha_i = 0 \quad \textit{und} \ \forall i : 0 \le \alpha_i$$

Klassifikationsregel:

$$h(\vec{x}) = sign\left(b + \sum_{i=0}^{n} \alpha_i y_i \vec{x}_i \cdot \vec{x} \right)$$

$b = y_i - \vec{w} \cdot \vec{x}_i$ wobei x_i ein Stützvektor ist

$\alpha_i > 0$ **nur bei Stützvektoren**

Anwendung in der Intensivmedizin

Gegeben Patientendaten mit Messwerten und Intervention des Arztes

finde die Regel, die klassifiziert in Dosis erhöhen oder nicht

Für jedes Medikament wird die Lernaufgabe gelöst.

Regel für Glyceroltrinitrat erhöhen

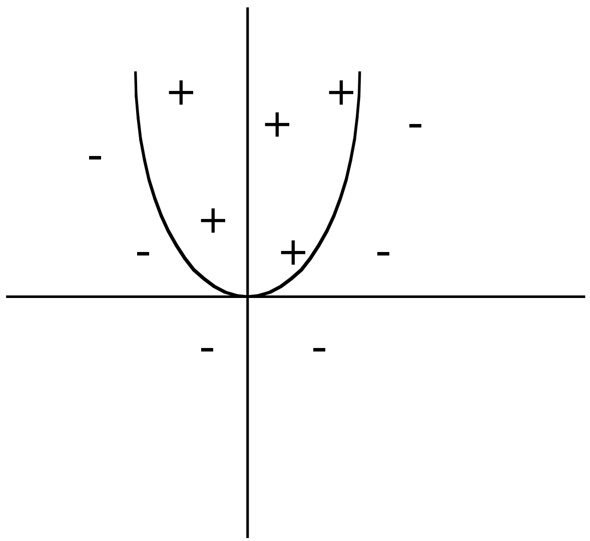
$$h(x) = \left[\begin{pmatrix} 0.014 \\ 0.019 \\ -0.001 \\ -0.015 \\ -0.016 \\ 0.026 \\ 0.134 \\ -0.177 \\ \ldots \end{pmatrix} \begin{pmatrix} Artsys\ 174.00 \\ Artdia\ 86.00 \\ Artmn\ 121.00 \\ Cvp\ 8.00 \\ Hr\ 79.00 \\ Papsys\ 26.00 \\ Papdia\ 13.00 \\ Papmn\ 15.00 \\ \ldots \end{pmatrix} - 4.368 \right]$$

Kernfunktionen

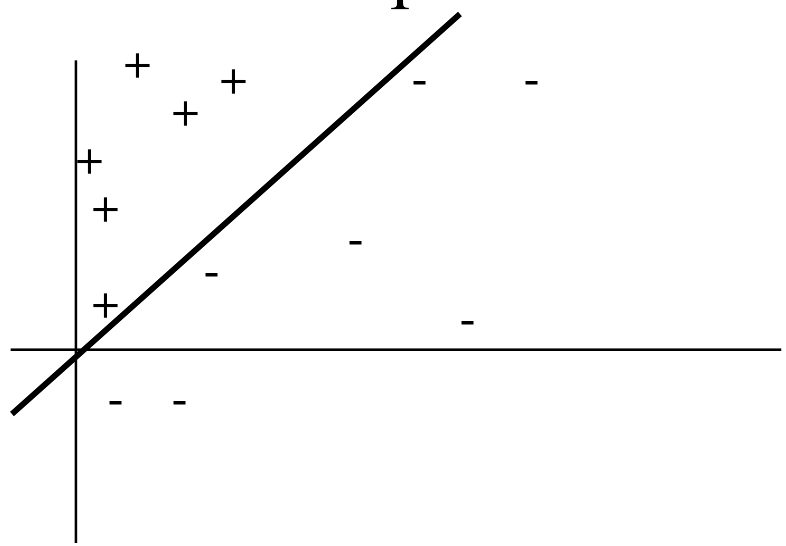
- Bei nicht linear separierbaren Klassen, werden die Beispiele in einen Raum transformiert, in dem sie linear separierbar sind. $\Phi(x)$
- Zum Glück muss man nicht wirklich alle Beispiele umrechnen! Man braucht ja beim Lernen wie bei der Klassifikation nur das Skalarprodukt zweier Beispielvektoren. $\Phi(x_i) \cdot \Phi(x_j)$

Beispiel

- Eingabedaten -- nicht linear separierbar



- Transformierte Beispiele -- linear separierbar



Kernfunktion: $K_{polynom}(\vec{x}_i, \vec{x}_j) = \left(\vec{x}_i \cdot \vec{x}_j + 1\right)^2$

Begriffslernen aus Beispielen

- gegeben: Hypothesensprache LH für den Begriff
- Hintergrundwissen B in einer Sprache LB
- Menge E = P ∪ N in einer Sprache LE
- wobei B, E |≠ , B |≠ E
- Ziel:
- H ∈ LH mit
- B, H, E |-/- (Konsistenz)
- B, H |-- P (Vollständigkeit)
- ∀e ∈ N: B, H |-/- e (Korrektheit)
- Präferenzkriterium:
- z.B.: speziellste Generalisierung, generellste Diskriminierung
- u.a.: triviale Lösung (Aufzählung von P) ausschließen!

Zuverlässige Lernoperatoren

LH wird durch eine Generalisierungsbeziehung geordnet.

Diese soll eindeutig sein:

für zwei Klauseln existiert genau eine Generalisierung (Spezialisierung).

Dann ist Lernen die Suche entlang der Generalisierungsbeziehung im Hypothesenraum.

Der Hypothesenraum kann sicher beschnitten werden:

- Wenn beim Generalisieren bereits die Hypothese C1 ein negatives Beispiel abdeckt, so wird dies auch jede Generalisierung von C1 tun. Von C1 ausgehend braucht nicht generalisiert zu werden.
- Wenn beim Spezialisieren bereits die Hypothese C1 ein positives Beispiel nicht abdeckt, so wird dies auch jede Spezialisierung nicht tun. Von C1 ausgehend braucht nicht spezialisiert zu werden.

Zuverlässige Lernoperatoren liefern die speziellste Generalisierung oder generellste Spezialisierung.

Heuristische Verfahren liefern irgendeine konsistente Hypothese.

Generalisierung: Implikation

Eine Hornklausel C1 ist genereller als eine andere, C2, gdw.

C1 --> C2

C1 ist genereller als C2 bezüglich einer Theorie T , gdw.

T, C1 --> C2

Äquivalenz:

Sei T eine Konjunktion von Hornklauseln, dann sind die Klauseln C1 und C2 logisch äquivalent bzgl. T gdw.

T, C1 --> C2 und T, C2 --> C1

Redundanz :

Ein Literal L ist redundant in einer Klausel C bzgl. T gdw.

C und C \ {L} sind äquivalent bzgl. T.

Beispiel Generalisierung

T: append ([ ], C, C).

C2: append ([1,2], [3], [1,2,3]).

C1: append([A | B], C, [A | E]) :- append (B, C, E).

C1, T --> C2

¬ append ([1,2], [3], [1,2,3]).	append ([A | B], C, [A | E]) , ¬ append (B, C, E).
	{A/1, B/[2], C/[3], E/[2,3]}
¬ append ([2 | []], [3], [2,3]).	append ([A | B], C, [A | E]) , ¬ append (B, C, E).
	{A/[2], B/ [], C/ [3], E / [3]}
¬ append ([ ], [3], [3]).	append ([ ], C, C).
	{C / [3]}

Beweis von C2 mithilfe von T und C1

Beispiel Redundanz

T: member(X, [X| Y]).

C: member (X, [Y|Z]) :- member (X, Z), member(Y, (Y |Z]).

C': C \ { member(Y, (Y | Z])} ist äquivalent zu C.

T, C --> C' und T, C' --> C

T beschreibt den Fall, daß das Element am Anfang der Liste steht.

C' beschreibt den Fall, daß das Element im Rest der Liste steht.

C beschreibt beide Fälle.

Nachteile

Um für eine Menge von Beispielen die Generalisierung zu finden, müssen wir die Klausel(n) finden, die alle Beispiele impliziert! Dies ist zu aufwendig! (Semi-Entscheidbarkeit der logischen Folgerung in der Prädikatenlogik.)

Der Hypothesenraum ist nicht so strukturiert, daß bei jedem Generalisierungsschritt der Ausschnitt der erreichbaren Hypothesen kleiner wird.

Also ist die Implikation als Generalisierungsbeziehung nicht geeignet.

Generalisierung: Subsumtion

Eine Hornklausel C1 ist genereller als eine andere, C2, gdw.

C1 subsumiert C2.

Ein Literal L1 subsumiert ein Literal L2 gdw.

$L1\,\sigma = L2$.

Eine Klausel C1 subsumiert eine andere, C2, gdw.

$C1\theta \subseteq C2$

Eine Hornklausel subsumiert eine andere, gdw.

$C1kopf\theta \subseteq C2kopf$ und

$C1körper\,\theta = C2körper$

Die Subsumtion ist eine korrekte, aber nicht vollständige Ableitung, d.h.

C1 subsumiert C2 ==> C1 impliziert C2, aber nicht umgekehrt.

Beispiel Generalisierung

C1: member (X, [Y| Z]):- member(X,Z).

C2: member (3, [1,2,3]):- member(3, [2,3]), member(3, [3]).

θ: { X/3, Y/1, Z/ [2,3] }

C1kopf θ = member (3, [1 | [2,3]) = C2kopf

C1körper θ = member (3, [2, 3]) ⊆ C2körper

Nicht alles, was unter der Implikation eine Generalisierung ist,

ist es auch unter der Subsumtion!

T: append ([ ], C, C).

C2: append ([1,2], [3], [1,2,3]).

C1: append([A | B], C, [A | E]) :- append (B, C, E).

kein Hintergrundwissen!

Generalisierung kann nicht länger sein als Generalisiertes!

echt genereller

Zwei Klauseln C1 und C2 sind äquivalent, wenn gilt:
C1 subsumiert C2 und C2 subsumiert C1.

Eine Klausel C1 ist echt genereller als eine andere, C2, gdw.
C1 subsumiert C2 und C1 ist nicht äquivalent C2.

Redundanz

Ein Literal L in der Klausel C ist redundant, wenn gilt:

C subsumiert $C \setminus \{L\}$.

Eine Klausel heißt reduziert, wenn sie keine redundanten Literale enthält.

Algorithmus, der eine Klausel C reduziert (Plotkin):

1. Initialisiere D mit C.

2. Finde ein Literal in D und eine Substitution θ, so daß

$D\theta \subseteq D \setminus \{L\}$.

Gelingt dies nicht, STOP.

3. $D := D\theta$, gehe zu 2.

Die reduzierte Form einer Klausel ist eindeutig.

Beispiel Redundanz

C1: member(X, [Y | Z]):- member(X, Z), member (X, U).

θ ={U/Z}

C1 $\theta \subseteq$ C1' \ { member (X, U) }

C1 θ: member(X, [Y | Z]):- member(X, Z) =

C1 \ { member (X, U)}: member(X, [Y | Z]):- member(X, Z)

Vor-/ Nachteile

Vorteil:

Der Hypothesenraum wird schrittweise eingeschränkt.

Nachteil:

Hintergrundwissen wird nicht berücksichtigt.

Least General Generalization Plotkin

LGG (C1, C2):

Für alle Paare von Literalen $L1_i \in C1$, $L2_i \in C2$, suche die mit gleichem Prädikatensymbol und gleichem Vorzeichen heraus --

bilde LGG ($L1_i$, $L2_i$)

Die Generalisierung von C1 und C2 ist die Vereinigung aller generalisierten Literale.

Aus dieser Generalisierung werden alle redundanten Literale entfernt.

Generalisierung von Literalen
Anti-Unifikation

zwei Literale mit demselben Prädikatsymbol und Vorzeichen
p(s1, ..., sn)   p (t1, ..., tn) von links nach rechts durchgehen

LGG (si, ti) = X, falls si, ti konstante Terme oder Variablen ungleich X sind;

LGG (f(s1, ..., sn), f(t1, ..., tn)) = f (LGG(s1, t1), ..., LGG(sn, tn))

LGG (f(s1, ..., sn), g(t1, ..., tm)) = X

Beispiel

L1: unterhalt (ulf, maria, alimente(ulf, 1000))

L2: unterhalt(udo, marion, alimente(udo,300))

LGG(L1, L2): unterhalt (X, Y, alimente(X, V))

wobei es für jede andere Generalisierung G(L1, L2) eine Substitution gibt, so daß G(L1, L2) σ = LGG(L1, L2)

LGG (C1, C2)

C1: member(2, [1,2]) :- member (2, [2]).

C2: member (c, [a, b, c]):- member(c, [b,c]), member (c, [c]).

alle Paare:

[¬ m(2, [2]),   ¬ m(2, [2]), ~~¬m(2, [2]),~~   ~~m(2, [1,2]),~~   ~~m(2, [1,2]),~~ m(2, [1,2])]

[¬ m(c, [b,c]), ¬ m((c, [c]), ~~m(c, [a, b, c]),~~ ~~–m(c, [b,c]),~~ ~~–m((c, [c]),~~ m(c, [a, b, c])]

LGG (¬ m(2, [2]), –m(c, [b,c])) = ¬m(A, [C|D])

LGG (¬ m(2,[2]), ¬ m(c, [c])) = ¬ m(A, [A])

LGG (m(2, [1,2]), m(c, [a, b, c])) = m (A, [B, C|D])

LGG (C1, C2) = m (A, [B, C|D]) :- m (A, [C|D]), m (A, [A]).

Bei jedem Literal probieren, ob es weggelassen werden kann, ohne zu generalisieren. Dieser Schritt ist leider NP-schwierig, weil der Subsumtionstest NP-schwierig ist.

Eigenschaften des LGG

kommutativ

$$lgg(e1, e2) = lgg(e2, e1)$$

assoziativ

$$lgg(lgg(e1, e2), e3) = lgg(e1, lgg(e2, e3))$$

idempotent

$$lgg(e, e) = e$$

Das bedeutet auch: reihenfolgeunabhängig

$$lgg(e1, e2, ..., en) = lgg(\ldots lgg(lgg(e1, e2), e3), \ldots en)$$

und eindeutig.

In einem Verband von Äquivalenzklassen von Klauseln ist das Supremum zweier Klauseln ihr LGG.

Aufwand

Die gute Nachricht:

Die Länge des LGG ist linear in der Anzahl der Selektionen.

Der Aufwand der Generalisierung ist linear in der Anzahl der Selektionen.

Die schlechte Nachricht:

Hat die längste Klausel in den Beispielen m Literale und

gibt es n Klauseln als positive Beispiele,

dann gibt es höchstens m^n Selektionen.

Es werden also exponentiell viele Selektionen in linearer Zeit bearbeitet.

Und dann kommt die Reduktion, die für jedes Literal noch einmal den aufwendigen Subsumtionstest braucht...

Hintergrundwissen

T: haustier (X) :- tier(X), im_haus (X)

C1: kuscheltier (Z) :- flauschig (Z), tier (Z), im_haus (Z).

C2: kuscheltier (Y) :- flauschig (Y), haustier (Y).

C1 ist genereller als C2, denn T, C2 |- C1

T: haustier, ¬ tier, ¬ im_haus

C2: kuscheltier, ¬ flauschig, ¬ haustier

C1: kuscheltier, ¬ flauschig, ¬ tier, ¬ im_haus

Semantik

C1 ist genereller als C2 bzgl. T, wenn in jeder Interpretation I, die T wahr macht, für alle Atome A gilt:

wann immer C2 auf A zutrifft, trifft auch C1 auf A zu.

Subsumtion bzgl. Hintergrundwissen Plotkin

C1 ist genereller als C2 bezüglich einer Theorie T, gdw.
T, C1 |-- C2
wobei C1 in der Ableitung höchstens einmal vorkommt.

T: kuscheltier (X) :- flauschig (X), klein (X), haustier (X).
haustier (X) :- katze (X).
C1: klein (X) :- katze (X).
C2: kuscheltier (X) :- flauschig (X), katze (X).

C1 handelt von anderem als C2.

Ableitung

T

{kuscheltier (X), ¬ flauschig (X), ¬ klein (X), ¬ haustier (X)}

T

{haustier (X), ¬ katze (X)}

{kuscheltier (X), ¬ flauschig (X), ¬ klein (X), ¬ katze (X)}

C1

{klein (X), ¬ katze (X)}

C2

{kuscheltier (X), ¬ flauschig (X), ¬ katze (X)}

generalisierte Subsumtion Buntine

C1 ist genereller als C2 bzgl. T, gdw.

es gibt eine Substitution θ, so daß $C1_{kopf}\,\theta = C2_{kopf}$

es gibt eine Skolemsubstitution σ, die alle Variablen in C2 durch neue Konstanten ersetzt

es gibt einen Klauselkörper C1 mit den Substitutionen, so daß

T, $\neg$ C2 körper σ |-- $\neg$ C1 körper $\sigma\,\theta$

Beispiel 1

T: haustier (X) :- tier (X), im_haus (X).

C1: kuscheltier (Z) :- flauschig (Z), tier (Z), im_haus (Z).

C2: kuscheltier (Y):- flauschig (Y), haustier (Y).

$C2 \geq C1$ bzgl. T ?

$C2_{kopf}\,\sigma = C1_{kopf}$

kuscheltier (Y) σ = kuscheltier (Z) mit $\sigma = \{Y/Z\}$

T, $\neg$ C1 körper θ |-- $\neg$ C2 körper $\sigma\theta$

T

{haustier(X), $\neg$ tier (X), $\neg$ im_haus (X)}

{ flauschig (b), tier (b), im_haus(b)} θ:{Z/b} **$\sigma 1$: {X/b}**

{haustier (b), flauschig (b)} = $\neg$ C2 körper $\sigma\theta$

Beispiel 1 cont'd

T: haustier (X) :- tier (X), im_haus (X).

C1: kuscheltier (Z) :- flauschig (Z), tier (Z), im_haus (Z).

C2: kuscheltier (Y):- flauschig (Y), haustier (Y).

C3: kuscheltier (Y):- flauschig (Y), tier (Y), im_haus (Y), zahm (Y)

C1 $\geq$ C3 bzgl. T ?

(T), – C3 körper θ |-- ¬ C1 körper σ θ

Widerspruchsbeweis:

¬ C3 körper θ wird ¬ ¬ C1 körper σ θ hinzugefügt.

¬ C3 körper θ mit θ = {Y/b}	**¬ ¬ C1 körper σ θ mit σ = {Z/Y}**
flauschig (b),	**{¬ flauschig (b), ¬ tier (b), ¬ im_haus (b) }**
tier (b),	**{¬ tier (b), ¬ im_haus (b) }**
im_haus (b),	**{¬ im_haus (b) }**
zahm (b)	$\square$

Beispiel 2

T: member (A, [A | B]).

C2: member (2, [1, 2]).

$C1_1$:member (A , [B, A]). Generalisierung unter θ Subsumtion

$C1_2$: member (2, [1,2]) :- member (2, [2]). unter generalisierter

$C1_3$: member (A, [B|C]) :- member (A, [C]). θ Subsumtion

Leider aber auch:

$C1_4$: member (2, [1,2]):- member (2, [2]), member (2, [2,2,2]).

$C1_5$: member (2, [1,2]) :- member (2, [2]), member (2, [2,3,4]).

...

Dies gilt, ohne ¬ C2 körper überhaupt zu benutzen!

T:	**¬ $C1_4$ körper:**
member(A, [A|B]).	**member (2, [2]).**
member(A, [A|B]).	**member (2, [2,2,2]).**

Denn schließlich gilt eine Konjunktion, wenn jedes Literal darin gilt.

Notwendigkeit von θ

T: haustier (X):- tier (a), im_haus (X).
C1: kuscheltier (Y):- flauschig (Y), tier(a), im_haus(Y).
C2: kuscheltier (Z):- flauschig(Z), haustier (a).

C1 soll nicht genereller sein als C2

$C1_{kopf}\ \theta = C2_{kopf}$

kuscheltier (Y) θ = kuscheltier (Z) mit θ = {Y/Z}

T, C2 körper σ |-- C1 körper θ σ

T
{haustier (X), ¬ tier (a), ¬ im_haus (X)}

{¬flauschig (b), ¬ haustier (a)} σ: {Z/b} σ1: {X/a}

{¬ flauschig (b), ¬ tier(a), ¬ im_haus(a)} ≠ {¬ flauschig(b), ¬ tier(a), im_haus(b)}

Nachteile der generalisierten Subsumtion

Es gibt unendlich viele Generalisierungen zu einer Klausel bezüglich des Hintergrundwissens.

Es gibt auch viele für die θ-Subsumtion. Dort hat man aber die Reduktion, die alle diese Generalisierungen auf eine Klausel zurückführt. Ein entsprechender Schritt fehlt bei der generalisierten θ-Subsumtion.

Wenn wirklich die logische Folgerung zwischen den Klauselkörpern geprüft werden muß, ist dies nur semi-entscheidbar.

Wenn man Rekursion verbietet und die Länge der generalisierten Klausel beschränkt, können nicht mehr unendlich viele Generalisierungen erzeugt werden.

Hintergrundwissen in Beispiele hineinrechnen und dann LGG bilden

LE: Grundfakten

LB: Grundfakten

Beispiele werden saturiert:

e_{neu} = e :- K wobei K die Konjunktion aller Fakten aus dem Hintergrundwissen ist.

ggf. werden die neuen Beispiele auf verbundene Klauseln beschränkt.

Eine Klausel heißt verbunden, wenn alle ihre Literale verbunden sind. Ein Literal heißt verbunden, wenn mindestens einer seiner Terme verbunden ist. Ein Term heißt verbunden mit der Tiefe 0, wenn er im Kopf der Klausel vorkommt. Ein Term heißt verbunden mit der Tiefe d+1, wenn ein anderer Term desselben Literals mit der Länge d verbunden ist.

oma (X, Z) :- mutter (X,Y), vater (Y, Z). X, Z haben die Tiefe 0, Y die Tiefe 1.

LH: funktionsfreie, nicht rekursive Klauseln

Dann ist die θ-Subsumtion eine korrekte und vollständige Ableitung!

Beispiel

Beispiele:

oma(anna, christof).

oma(anita, cecilie).

Hintergrundwissen:

mutter(anna, bernd). vater (bernd, christof).

mutter (anita, bruno). vater (bruno, cecilie).

Beispiele neu:

oma(anna, christof):- mutter (anna, bernd), vater(bernd,christof).

oma(anita, cecilie):- mutter (anita, bruno), vater (bruno, cecilie).

LGG:

oma (A, C) :- mutter (A, B), vater (B, C).

generalisierte Subsumtion?

Ist

C1: oma (A, C) :- mutter (A, B), vater (B, C).

allgemeiner als

C2: oma(anna, christof).

bzgl.

T: mutter(anna, bernd). vater (bernd, christof).

? Trivialerweise: JA.

C1kopfσ = C2kopf

σ ={A/anna, C/christof}

T: mutter(anna, bernd). vater (bernd, christof).

|=

$\neg$ C1körper σ: mutter(anna, bernd). vater (bernd, christof).

Und wenn LB Klauseln sind?

funktionsfreie Klauseln:

Jedes Argument eines Prädikats ist entweder einer Variable oder eine Konstante -- Funktionen sind ausgeschlossen.

generative Klauseln (bereichsbeschränkt):

Jede Variable im Klauselkopf kommt auch im Körper vor.

oma (X,Z) :- mutter (X,Y), vater (Y, Z).

Wenn man alle Variablen im Hintergrundwissen durch die in den Beispielen vorkommenden Konstanten ersetzt, so wird das Hintergrundwissen variablenfrei.

Wenn LB auf funktionsfreie, generative Klauseln beschränkt ist, so kann man durch einen (tiefenbeschränkten) Ableitungsprozeß ebenfalls variablenfreies Hintergrundwissen herstellen.

Und dann kann man das Hintergrundwissen in die Beispiele hineinrechnen und den LGG bilden.

Beispiel

Beispiele:	Hintergrundwissen:
	vater(Y,Z) :- kind (Z, Y), mann (Y).
oma(anna, christof).	kind(christof, bernd). mann(bernd).
oma(anita, cecilie).	kind(cecilie, bruno). mann (bruno).

Beispiele neu:

oma(anna, christof):- mutter (anna, bernd), vater(bernd,christof).

oma(anita, cecilie) :- mutter (anita, bruno), vater (bruno, cecilie).

Saturierung

Rouveirol

Sei C1 die Klausel H1 :- B1 und

C2 die Klausel H2:- B2, wobei B2 θ-subsumiert B1.

Dann ist die elementare Saturierung von C1 durch C2

D: H1:- B1, H2θ

C1: oma(anna, christof):- mutter(anna, bernd), kind (christof, bernd), mann (bernd).

C2: vater(Y,Z) :- kind (Z, Y), mann (Y).

D: oma(anna, christof):- mutter (anna, bernd), kind (christof,bernd), mann(bernd), vater(bernd,christof).

Sprachbeschränkungen

deterministische Klauseln (bzgl. des Hintergrundwissens):

jede Variable in jedem Literal hat eine eindeutige Substitution durch das Hintergrundwissen.

Klausel: oma(anna, Z):- mutter(anna, X), vater(X, Z).

Hintergrundwissen:

mutter(anna, bernd). vater (bernd, christof).

hier: nur {X/ bernd, Z/christof}

Aber wenn Hintergundwissen:

mutter(anna, bernd). vater(bernd, christof). vater (bernd, christiane).

dann ist die Klausel indeterministisch, weil es zwei Substitutionen für Z gibt.

ij-deterministisch

Ein Term aus dem Klauselkopf K ist mit einer Kette 0 deterministisch verbunden.

Für den Klauselkörper nehmen wir an, daß die Literale nach Verbundenheit mit dem Klauselkopf geordnet sind:

$\{ \neg L_1, \ldots, \neg L_m, \neg L_{m+1}, \ldots, \neg L_n \}$

Ein Term t aus L_{m+1} ist genau dann durch eine deterministische Kette der Länge d+1 verbunden, wenn

alle Terme im Klauselkopf und in $\{\neg L_1, \ldots, \neg L_m \}$ verbunden sind durch deterministische Ketten, die höchstens d lang sind,

es für jede Substitution θ, die K mit einem Beispiel und die ersten Literale mit dem Hintergrundwissen unifiziert (d.h. $K\theta \in E^+$ und $\{ \{L_1\}, \ldots, \{\neg L_m\}\}\theta \subseteq B$) genau eine eindeutige Substitution σ gibt, so daß $L_{m+1}\,\theta\sigma \in B$.

Die minimale Länge der deterministisch verbindenden Ketten ist die deterministische Tiefe eines Terms.

Eine Klausel mit maximaler deterministischer Tiefe i und maximaler Stelligkeit j heißt ij-deterministisch.

Beispiel

oma(X, Z) :- mutter (Y, Z) , mutter (X, Y) oma(X, Z) :- vater (Y, Z), mutter (X, Y)	12-deterministisch
oma(X, Z) :- mutter (X, Y), elternteil (Y, Z)	indeterministisch, insofern eine Mutter mehrere Kinder haben kann und ein Kind 2 Elternteile hat.
tante(X1, Z) :- geschwister (X1, Liste), member (X2, Liste), elternteil (X2, Z).	indeterministisch, insofern als die Liste mehr als ein Element enthalten kann und X2 nicht im Kopf gebunden ist.
tante(X, Z) :- mutter (Y, Z), schwester (X, Y) tante(X, Z) :- vater (Y, Z), schwester (X, Y)	12-deterministisch

Lernbarkeit

Sei LE Grundfakten mit höchstens t Termen,

LB Grundfakten mit m verschiedenen Prädikaten, die höchstens f Terme enthalten,

LH ij-deterministische Klauseln, wobei i und j festgelegt sind,

so werden Hypothesen mit höchstens

$O((t\ f\ m)^{i^j})$ Literalen gelernt.

Wegen der Tiefenbeschränkung ist die Länge der Klauseln also nicht mehr exponentiell.

Indeterministische Klauseln sind auch bei Tiefenbeschränkung nicht polynomiell lernbar.

beschränkte Klauseln

Eine Klausel ist beschränkt, wenn alle Variablen aus dem Klauselkörper auch im Klauselkopf vorkommen.

Spezialfall der ij-deterministischen Klauseln mit i = 0.
Polynomiell lernbar, PAC-lernbar.
Nicht ausdrucksstärker als Aussagenlogik.

k- lokale Klauseln

Bestehe eine Klausel aus einem deterministischen Teil D_{DET} und einem indeterministischen Teil D_{NONDET}.

$D0$:- D_{DET} , D_{NONDET}

Sei vars eine Funktion, die alle Variablen einer Klausel findet.

Als lokalen Teil LOC einer Klausel bezeichnen wir die Literale aus D_{NONDET}, für die gilt:

$(vars(LOC) \setminus vars(\{D0, D_{DET}\})) \cap vars(D_{NONDET} \setminus LOC) = \{\}$

Minimaler lokaler Teil für eine Konstante k

k-vlokal gdw. $k \geq | vars(LOC) \setminus vars(\{D0, D_{DET}\}) |$ ---- nicht lernbar

k-llokal gdw. $k \geq | LOC |$ -- lernbar

Beispiel

Geschwister, deren Mutter Oma ist:

geschwister(X, Z) :- mutter (Y1, X), mutter (Y1, Z),
mutter (Y1, Y2),
elter (Y2, Y3).

vars ({$D0$, D_{DET} }): X, Y1, Z

vars (D_{NONDET}): Y1, Y2, Y3

LOC 1: mutter (Y1, Y2)

LOC 2: elter (Y2, Y3)

LOC 3: mutter (Y1, Y2), elter (Y2, Y3)

(vars (LOC) \ vars ({$D0$, D_{DET} })) ∩ vars (D_{NONDET} \LOC) = {}

LOC1:

({Y1, Y2 } \ {X, Y1, Z}) ∩ vars({mutter(Y1,Y2),elter(Y2,Y3)}\{mutter(Y1,Y2}) =
{Y2} ∩ {Y2,Y3} ≠ {}

LOC2:

({Y2, Y3} \ {X, Y1, Z}) ∩ ({Y1, Y2}) = {Y2, Y3} ∩ {Y1,Y2} ≠ {}

LOC3:

({Y1, Y2, Y3} \ {X, Y1, Z}) ∩ ({ }) = {Y2, Y3} ∩{} = { }

2-vlokal, 2-llokal

Regellernen

Gegeben:

Hypothesensprache für Mengen von Regeln LH

Beobachtungen E in einer Sprache LE

Hintergrundwissen Tin einer Sprache LT

T, E |-/–   (Konsistenz)

Ziel:

C ∈ LH, so daß $\mathcal{M}$+ (T, E) ⊆ $\mathcal{M}$ (C)   (Gültigkeit)

∀c ∈ C, ∃ e ∈ E: T, E-{e} |-/- e,

aber T, E-{e}, c |-- e   (Notwendigkeit)

Wenn c gültig und notwendig ist, dann C |-- c   (Vollständigkeit)

Es gibt keine echte Teilmenge von C,

die gültig und vollständig ist   (Minimalität)

(minimales) Modell

Gegeben eine Interpretation I für eine Menge von Formeln F.

I ist ein Modell von F, $\mathcal{M}$ (F), wenn alle Formeln von F in I wahr sind.

Wenn es keine Interpretation I' gibt, mit I' $\subseteq$ I und I' ist ein Modell von F, ist I ein minimales Modell für F, $\mathcal{M}^+$.

Lernende Roboter

Bisherige Ansätze:

Neuronale Netze zum Lernen eines kollisionsfreien Pfades oder zur automatischen Synthese von Fuzzy Controllern.

Dies sind sinnvolle Lernläufe auf der untersten Ebene eines Roboters .

Erklärungsbasiertes Lernen zur Optimierung der Planung.

Dies sind sinnvolle Lernläufe auf der höchsten Ebene eines Roboters.

Wie aber kann die Reflex-Ebene mit höheren Ebenen kombiniert werden?

Können Wahrnehmung und Handlung in Merkmalen (Begriffen) aller Ebenen integriert werden?

Beispiel

Zeitbehandlung durch Relationen zwischen Zeitpunkten.

Abstrakte Wahrnehmungsmerkmale, die Sensordaten komprimieren.

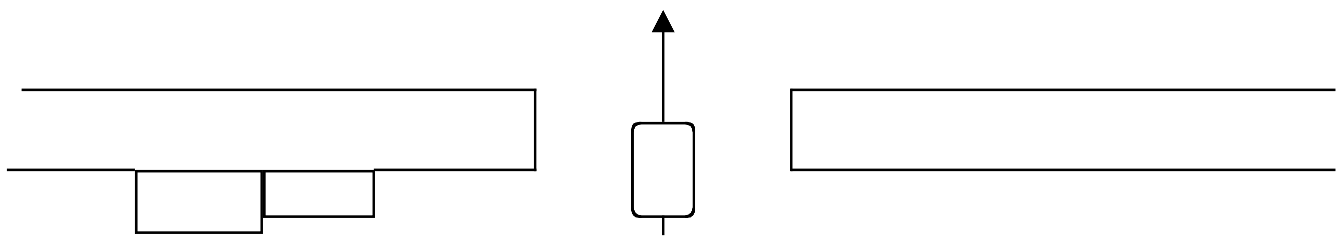
Anwendbarkeit, — **standing (Trc, T1, T2, *in_front_of_door* , PDir, small_side, PrevP) &**

Handlung mit Wahrnehmung, — **parallel_moving(Trc, T2, T3, Speed, PDir, *through_door*, right_and left) &**

Verifikation — **standing (Trc, T3, T4, *in_front_of_door* , back, small_side, *through_door*)**

--> move_through_door (Trc, T1, T4)

through_door (...) --> parallel_moving (...)
Basismuster --> through_door (...)
Sensor, Bewegung --> incr_peak (...)



KDD mit ILP und direktem Datenbankzugriff

Rule Discovery Tool (Kietz, Wrobel 1992)

Regelschemata in Allgemeinheitsordnung

Benutzer geben an, welche Form von Regeln sie interessant finden

m1(C,P, Q): P(X,C) → Q(X)

m2(P1, P2, Q): P1(X,Y) & P2(X) → Q(Y)

m3(C, P1, P2, Q): P1(X, Y) & P2(X, C) → Q(Y)

Regelschemata werden mit allen Prädikaten instantiiert, c mit allen vorhandenen Werten (Intervallen).

Jede Instanz ist eine Regel. Ihre Gültigkeit wird mithilfe von SQL-Anfragen über der Datenbank getestet.

Akzeptanzkriterien

Benutzer geben Kriterium für die Akzeptanz einer Hypothese an.

Beispiele:

$$\frac{pos(H)}{concl(H)} - \frac{neg(H)}{concl(H)} \geq 0.8$$

a posteriori **a priori** **bei 2 Klassen**

$$\frac{pos(H)}{pos(H) + neg(H)} \geq \frac{concl(H)}{concl(H) + negconcl(H)}$$

pos(H): Prämisse und Konklusion kommen gemeinsam vor neg(H): Prämisse und Negation der Konklusion kommen gemeinsam vor	concl(H): Konklusion kommt vor negconcl(H): Konklusions-prädikat ist anwendbar, kommt aber nicht vor

Abbildung von Datenbank auf Prädikate

Jede Datenbanktabelle ist ein Prädikat, ihre Attribute sind die Argumente.

customer (Person, Income, Customer)

Auswahl interessanter Attribute:

Vorauswahl bei sehr vielen Datenbankattributen

Jedes Datenbank-Attribut wird ein Prädikat, der Schlüssel und der Attributwert die Argumente.

income (Person, Income) .., wife (Husband, Wife)

Jeder Wert eines Datenbankattributs wird Prädikat, Schlüssel das Argument

customer (Person), inc_10_20 (Person)

Sinnvolle Intervalle in Attributwerten vorher finden!

Größe des Hypothesenraums

$$r\,(p\, i^{c})^{k}$$

- r **Anzahl der Regelschemata**
- p **Anzahl der Prädikate**
- k **max. Anzahl von Literalen in einem Regelschema**

Bei Konstantenlernen:

- c **Anzahl der zu lernenden Konstanten**
- i **max. Anzahl von Werten für eine Konstante**

Je nach gewählter Abbildung ist die Größe des Hypothesenraums sehr unterschiedlich.

Experimente

Daimler Benz AG

2,6 Gigabyte Datenbank: alle Fahrzeuge und ihre Garantiefälle

40 Tabellen mit je bis zu 40 Attributen

1. **Unterschiedlich großer Hypothesenraum:**
   4913 ≤ Größe ≥ 2,8 E 41
2. **Unterschiedlich großer Datenbankauszug:**
   max. 23 Tabellen
   max. 750 000 Tupel
3. **Verwendung von Hintergrundwissen**
4. **Vergleich mit anderen ILP-Verfahren**

Verwendung von Hintergrundwissen

Gegeben:

elektronisch verfügbares Werkstattbuch für PKW mit allen Fahrzeugteilen

Finde:

Gruppen von Teilen, die räumlich, funktional oder bzgl. ihrer Schadensart zusammenhängen

Umformen der Datei in einstellige Fakten, wobei das Fahrzeugteil als Argument, der Zusammenhang als Prädikat ausgedrückt ist

Verwendung von STT (Kietz 1989) zum Finden einer Subsumtionshierachie

Klassen von Fahrzeugteilen der Datenbank hinzufügen

Lernergebnisse

rel_niveauregulierung (FIN) → beanstandet (FIN)

motor_e-typ (FIN, Typ) & mobr_cyl (Typ, 6) → beanstandet (FIN)

rel_garantie (X1, X2, RB, X4, X5, X6, X7, Konfig, Teil) &
rel_garantie_benzman (X1, X2, RB, X4, X5, X6, X7, FIN) &
rel_motor_typ (FIN, 206) & italien(RB) → class_419 (Konfig, Teil)

rel_garantie (X1, X2, X3, X4, X5, X6, X7, Konfig, Teil) & class_35 (Konfig, Teil)
→ kostbean_0_500 (X1, X2, X3, X4, X5, X6, X7)

a priori Wahrscheinlichkeit: 0,84 a posteriori Wahrscheinlichkeit: 0,89

Weitere Anwendungen

Daten über Filialen, Produkte, Verkauf

- **6 Gigabyte Daten**
- **Fragestellung nach "Ausreißern"**

Daten aus der Intensivmedizin

- **beliebig viele**
- **Monitoring**

Identifikation im Grenzwert

Es gibt eine Wahrheit, die einen Strom von Daten schafft.

Lernen ist die Beobachtung der Daten und das Raten der Wahrheit.

Der Lerner erzeugt eine operationale Beschreibung der Daten:

- die Beschreibung erzeugt alle bisher gesehenen Daten und
- kann neue Daten ergänzen.

Irgendwann muß durch neue Daten die Beschreibung nicht mehr geändert werden, weil die Wahrheit identifiziert wurde.

Man weiß nie, ob man gerade die Wahrheit identifiziert hat.

Funktionen identifizieren

Problemspezifikation:

Definitions- und Wertebereich,

Menge von Funktionen, Repräsentation der Funktionen,

Präsentation der Beispiele

Gegeben: eine Maschine, die die Werte einer Funktion $p(x)$ für Werte von $x=0,1,2,3...$ auf Anforderung ausgibt

Vorgehen: Raten, was $p(x)$ ist und den nächsten Wert anfordern

Wenn der Wert nicht übereinstimmt mit dem aus der geratenen Funktion berechneten - neues Raten und neuen Wert anfordern;

wenn der Wert übereinstimmt, bei der geratenen Funktion bleiben und neuen Wert anfordern.

Identifikation: die Funktion $p(x)$ wird richtig geraten und es wird nicht weiter geraten (aber immer weiter ein neuer Wert angefordert)

Beispiel

die ersten fünf Werte der unbekannten Funktion p(x): (0,1),(1,1),(2,1),(3,1),(4,1)

geratene Funktion: .... konstantes Polynom 1

nächster Wert: .... (5, 121)

Identifikation: .... 1 + x (x-1) (x-2) (x-3) (x-4)

Definitionsbereich: N

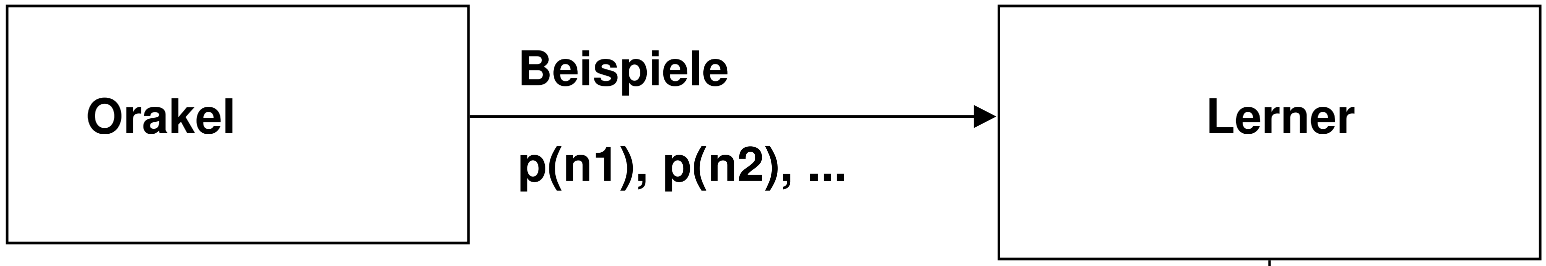
Wertebereich: N

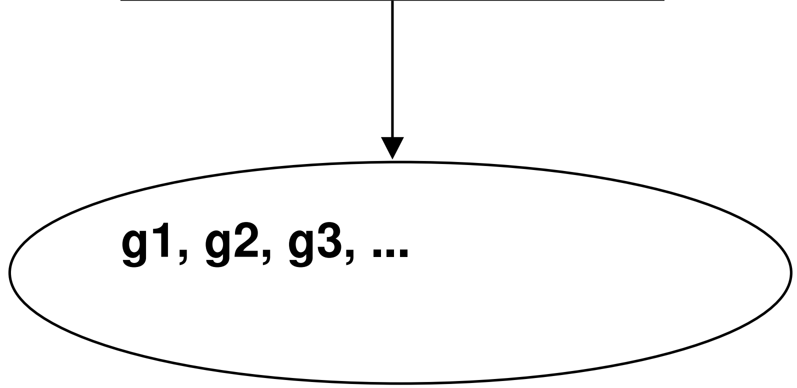
Menge von Funktionen von Definitions- auf Wertebereich:
alle Polynome mit nur einer Variablen

Repräsentation der Funktionen: Polynom zur Berechnung der Funktion

Präsentation der Beispiele von Funktionswerten:
Paare (n, p(n)) in aufsteigender Reihenfolge

induktive Inferenzmethode





geratene Funktionen
(LISP-Programme zu ihrer Berechnung)

Identifikationsqualität:

Ein Lerner *erklärt* eine Funktion, wenn er irgendwann nur noch zu ihr syntaktisch äquivalente Berechnungen ausgibt.

Ein Lerner *beschreibt* eine Funtkion, wenn er irgendwann nur noch Berechnungen ausgibt, die richtige Funktionswerte liefern.

Ergebnisse

Natürlich sind alle erklärbaren Funktionen auch beschreibbar.

Leider sind nicht alle vollständig berechenbaren Funktionen beschreibbar.

Es gibt keine induktive Inferenzmethode, die alle vollständig berechenbaren Funktionen beschreibt oder gar erklärt.

Es gibt nur eine Reihe von Spezialverfahren.

Sprachklassen identifizieren

Problemspezifikation:

Alphabet

Klasse von Sprachen über dem Alphabet

Menge zulässiger Sequenzen von Beispielen der Sprache (Wörter)

erlaubte Methoden der induktiven Inferenz

Erfolgskriterium

Alphabet: 0, 1

rekursiv aufzählbare Sprachen

vollständige Präsentation der Sprache durch positive und negative Beispiele:

- **(w,1) w ist Wort der Sprache (positives Beispiel)**
- **(w,0) w ist nicht Wort der Sprache (negatives Beispiel)**

Methoden induktiver Inferenz:

- **Raten der Funktion fL(w), die 1 ausgibt gdw. w ein Wort in L ist; Berechnung durch einen Parser.**

erfolgreich, wenn für kein negatives Beispiel fL(w)=1, für kein positives fL(w)=0

Ergebnisse

Aus vollständiger Präsentation von Beispielen können reguläre und kontextfreie Sprachen identifiziert werden.

Kinder lernen Sprache überwiegend aus positiven Beispielen!

Aus positiven Beispielen allein können nicht einmal alle regulären Sprachen identifiziert werden.

**Wenn die Beispiele von einem rekursiven Programm erzeugt werden, können alle rekursiv aufzählbaren Sprachen aus positiven Beispielen identifiziert werden.
(Genau wie Induktion von Funktionen, wobei diese hier Grammatiken sind, die alle Wörter der Sprache aufzählen.)**

Den kürzesten regulären Ausdruck zu finden, der mit gegebenen positiven und negativen Beispielen vereinbar ist, ist ein NP-schwieriges Problem.

Modell-Inferenz System

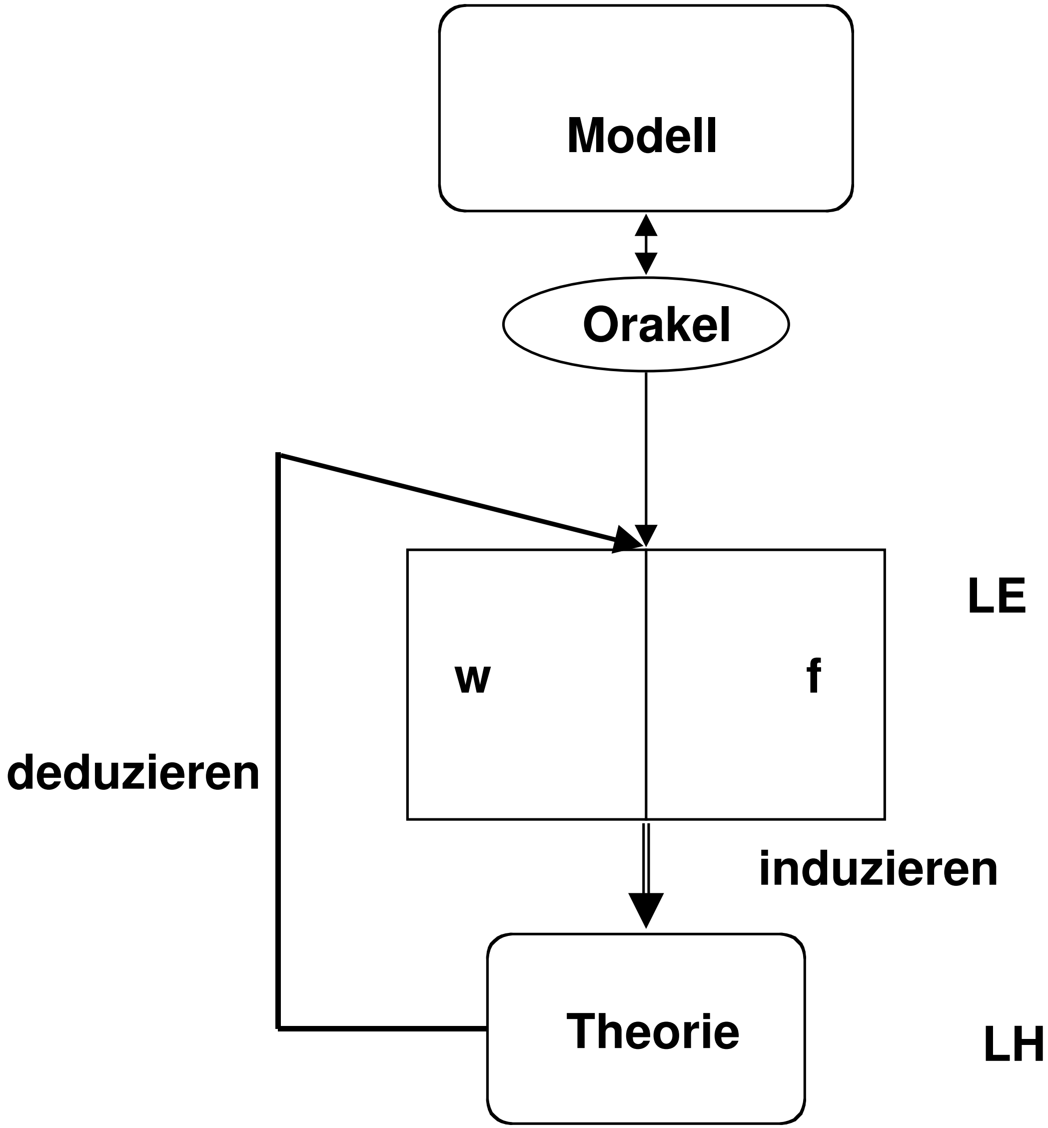
Gegeben: eine Beschreibungssprache für Beobachtungen LE
eine Hypothesensprache LH
ein Orakel, das jede Beobachtung bewertet als zum Modell gehörig oder nicht
einige Fakten (Beobachtungen mit Bewertung des Orakels)

Ziel: eine Theorie, die alle wahren Fakten ableitet und keine falschen.

Eine Theorie ist zu schwach, wenn es ein wahres Faktum gibt, das nicht abgeleitet wird.

Eine Theorie ist zu stark, wenn es ein falsches Faktum gibt, das abgeleitet wird.

Ehud Shapiro 1983


Modell
Orakel
w
f
LE
deduzieren
induzieren
Theorie
LH

Angemessene LE und LH

LE und LH sind angemessen, d.h.

es sind Sprachen derselben Signatur;

für jedes Modell der Signatur und für jede Theorie in LH gilt:

wenn die wahren Fakten in LE aus der Theorie in LH ableitbar sind,

dann gilt die Theorie im Modell.

Bei unangemessenen Sprachen LE und LH könnte es passieren, daß die Theorie in LH alle wahren Fakten ableitet, aber trotzdem falsch ist!

Eine Theorie soll durch Fakten falsifizierbar sein!

Beispiel

LE: 0	**interpretiert als 0**
X'	**interpretiert als Nachfolger von X**
X ≤ Y	**interpretiert als Xist kleiner oder gleich Y**
plus(X, Y, Z)	**interpretiert als X+Y=Z**

LH: Hornformeln

Fakten:

0 ≤ 0 ist wahr

plus(0, 0', 0) ist falsch

plus(0', 0'', 0''') ist wahr

LE und LH sind angemessen.

Allgemein

Wenn L eine Signatur der Prädikatenlogik ist ,

LE variablenfreie Fakten von L und

LH Hornklauseln von L,

dann sind LE und LH angemessen.

Nehmen wir an, $T \sqsubseteq LH$ wäre ungültig im Modell.

Dann existiert eine Aussage P in T, die ungültig ist im Modell.

Dann gibt es eine Substitution σ, so daß $P\sigma$ variablenfrei ist und $P\sigma$ ist ungültig im Modell.

$P\sigma$ ist in LE.

Also sind LE und LH angemessen.

h-easy Modelle

Alle wahren Fakten sollen abgeleitet werden von der gelernten Theorie.

Die Ableitung soll nun beschränkt werden.

Dazu definiert man eine Funktion h,
die für das i-te Faktum h(i) Ableitungsschritte erlaubt.

Ein Modell, für das es eine Theorie gibt,
bei der das i-te Faktum mit h(i) Schritten abgeleitet wird, heißt h-easy.

Gelernt werden nur h-easy-Modelle.

Vorgehen

MIS:

1. **wähle irgendeine Theorie (z.B. die allgemeinste) als Ausgangspunkt.**
2. **lies den nächsten Fakt ein**

   solange die Theorie zu stark ist, wende *contradiction backtracing* an

   solange die Theorie zu schwach ist, wende den *Verfeinerungsoperator* auf die soeben verworfenen Hypothesen an.
3. **Wenn die Theorie weder zu stark noch zu schwach ist bezüglich aller bisher gesehenen Fakten, drucke die Theorie aus.**
4. **gehe zu 2.**

forever

contradiction backtracing

Was in der Theorie hat zur Ableitung eines falschen Faktes geführt?

Gegeben:

- **Ableitungsbaum, der aus der Theorie ein falsches Faktum hergeleitet hat.**
- **Der Baum verzweigt sich so, daß links die negierten, rechts die nicht negierten Atome stehen.**

Finde die Formel, die falsch ist, und lösche sie aus der Theorie.

Dabei wird zu jedem Resolutionsschritt das Orakel nach Fakten befragt.

Ist ein Fakt falsch, so wird in der rechten Seite des Baums gesucht.

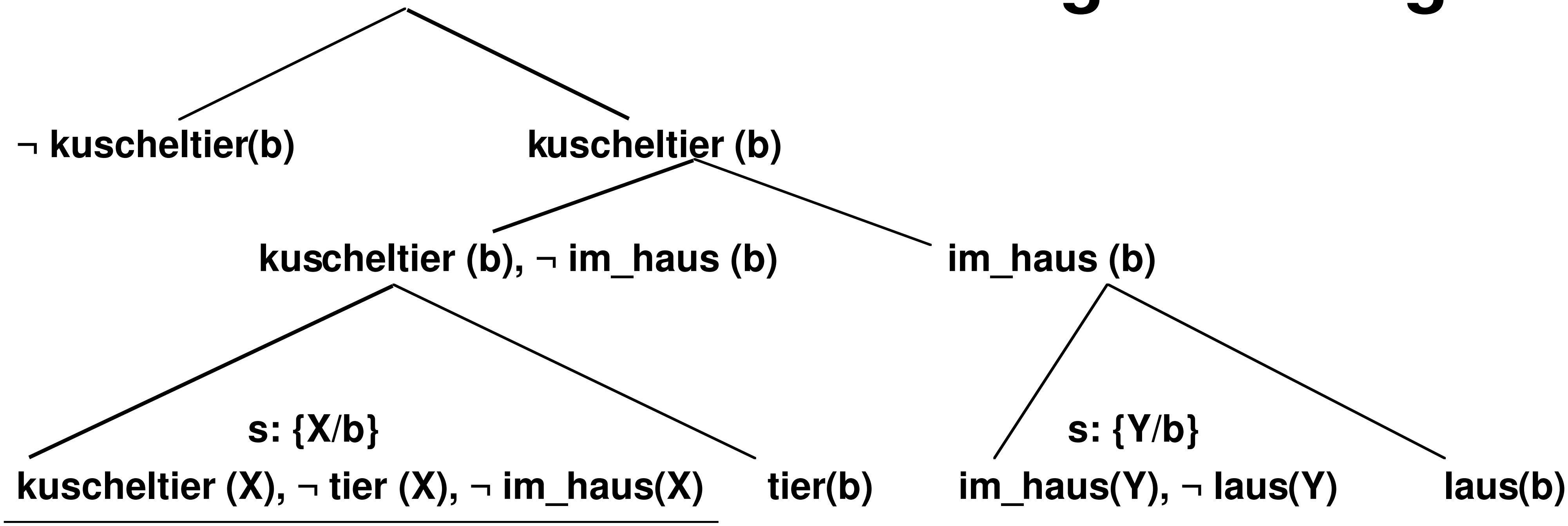
Ist ein Fakt wahr, so wird in der linken Seite des Baumes gesucht.

Beispiel

Beobachtungen bisher:	Theorie bisher:
tier (a), im_haus (a), kuscheltier(a) tier (b), laus (b)	kuscheltier (X), ¬ tier (X), ¬ im_haus (X) im_haus (X), ¬ laus (X)

neue Beobachtung:	**leitet ab:**
¬ kuscheltier(b)	**kuscheltier (b)**

Ableitung und Fragen



kuscheltier(b)? - f	also nach rechts!
im_haus (b)? - w	also nach links!
tier(b)? - w	also nach rechts!

Die falsche Klausel ist gefunden und wird aus der Theorie gelöscht.

Abschwächung der Theorie

Die neue, abgeschwächte Theorie ist:

im_haus (X), ¬ laus (X)

Sie leitet nicht mehr kuscheltier (b) ab.

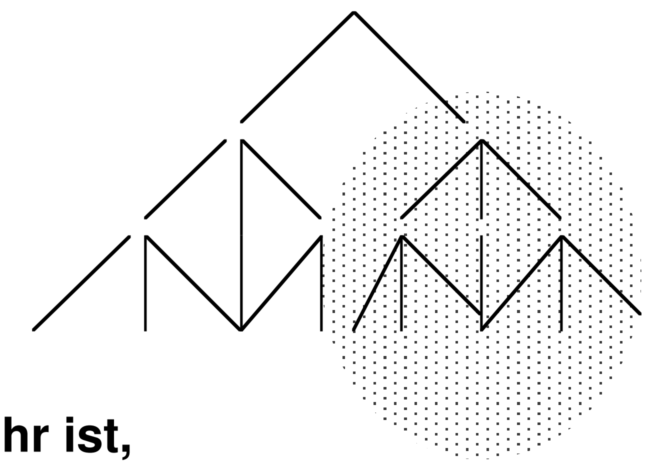
Verfeinerungsoperator

Spezialisierung zurückgewiesener Hypothesen

Gegeben:

eine allgemeine (falsche) Formel P

Finde eine Spezialisierung Q, so daß P --> Q



Verfeinerungsrelation sr

Wenn eine Hypothese wahr ist,

so sind auch alle ihre

Verfeinerungen wahr.

Verfeinerungsrelation sr

sr (P, Q) gilt für P, Q in LH, gdw.

1. **P ist ein Atom p (... U...), dann ist**

   Q dasselbe Atom mit einer anderen Variablen p(... V...) und U=V wird angehängt; oder

2. **P ist ein Atom p (...U...), dann ist**

   Q dasselbe Atom mit einer n-stelligen Funktion über Variablen X_i, die in P nicht vorkommen - Substitution $\{U / f(X_1, ..., X_n)\}$; oder

3. **Q ist P $\cup$ {L}, wobei L ein generellstes Literal bzgl. P ist, so daß Q reduziert ist.**

Q soll umfangreicher sein als P. Q soll nur einen Schritt feiner als P sein.

sr ist <u>vollständig</u>, wenn aus der leeren Aussage

alle in LH bildbaren Formeln erzeugt werden können.

Shapiros Trick

Shapiros Trick besteht darin, den Raum aller Formeln auf reduzierte Formeln zu beschränken. Dazu verwendet er Plotkins Reduktion.

Eine Formel ist reduziert, gdw. sie nicht äquivalent zu einer Teilmenge von sich selbst ist.

Von allen Formeln nimmt er nur in seine Verfeinerung die äquivalenten reduzierten Formeln auf.

Leider ist damit seine Verfeinerung nicht vollständig...

Redundanz

Ein Literal L in der Klausel C ist redundant, wenn gilt:

C subsumiert $C \setminus \{L\}$.

Eine Klausel heißt <u>reduziert</u>, wenn sie keine redundanten Literale enthält.

Algorithmus, der eine Klausel C reduziert (Plotkin):

1. **Initialisiere D mit C.**
2. **Finde ein Literal in D und eine Substitution θ, so daß**

   $D\theta \subseteq D \setminus \{L\}$.

   Gelingt dies nicht, STOP.
3. **D:= $D\theta$, gehe zu 2.**

Die reduzierte Form einer Klausel ist eindeutig.

generellstes Literal L...

bezüglich einer Klausel P, so daß $P \cup \{L\}$ reduziert sind:

$P \cup \{L\}$ ist reduziert und

für jedes Literal A, für das es eine Substitution θ gibt, so daß

$A\theta = L$ und $P\theta = P$ (genereller bezüglich P), gilt

$P \cup \{A\}$ ist nicht reduziert.

Beispiel:

P: p(X) :- q(X,Y)

L: ¬ q(U,X)

A: ¬ q(U,V)

L ist ein generellstes Literal bezüglich P, so daß $P \cup \{L\}$ reduziert ist.

A ist genereller als L bezüglich P, aber $P \cup \{A\}$ ist nicht reduziert.

$A\theta = L$ mit θ:{V/X} und $P\theta = P$

$P \cup \{A\}$: p(X):- q(X, Y), q(U, V)

mit σ:{U / X, V/Y} wird

$P \cup \{A\}\sigma \subseteq P \cup \{A\} \setminus \{A\} = P$

So ein σ finden wir für L nicht!

Beispiel

Beobachtungen bisher:	Theorie bisher:
tier (a), im_haus (a), kuscheltier(a)	im_haus (X), ¬ laus (X)
tier (b), laus (b), ¬ kuscheltier(b)	abgelehnte Hypothese:
flauschig (a), ¬ flauschig (b)	kuscheltier (X), ¬ tier (X), ¬ im_haus(X)

P: kuscheltier (X), ¬ tier (X), ¬ im_haus(X)

Q: P ∪ { flauschig (X)}

sr(P,Q) gilt nach 3.

kuscheltier (X), ¬ tier (X), ¬ im_haus(X), ¬ flauschig (X) ist reduziert.

Es gibt keine Verallgemeinerung A von flauschig (X), so daß P ∪ { A } reduziert ist.

Eigenschaften von MIS

- **Wenn das Modell M passend zur Signatur L h-easy ist, dann wird MIS irgendwann eine beobachtungsvollständige Axiomatisierung des Modells liefern, ohne allerdings zu merken, wann die Axiomatisierung erreicht ist.**
- **Eine beobachtungsvollständige Axiomatisierung eines Modells ist eine Menge von Formeln T gdw., $T \subseteq LH$, T ist im Modell gültig und T leitet alle wahren Beobachtungen ab.**
- **Eine Axiomatisierung ist ausreichend, wenn sie alle bisher eingelesenen wahren Fakten ableitet.**
- **Wenn es einen Algorithmus gibt, der eine ausreichende Axiomatisierung findet, dann gibt es auch einen, der eine beobachtungsvollständige findet.**

Ordnung ist das halbe Lernen...

Lernen als Suche bedeutet:

alle Hypothesen werden entlang ihrer Allgemeinheit angeordnet;
von den Beispielen ausgehend wird generalisiert und/oder
von der allgemeinsten Hypothese ausgehend wird spezialisiert,
bis die Bedingungen der Lernaufgabe erfüllt sind.

Man muß also

- **den Hypothesenraum ordnen**
- **aufgrund der Beispiele von einer Hypothese zur nächsten kommen**

Also:

Wahl der richtigen Sprache LH:

der Hypothesenraum muß die richtige Hypothese enthalten,

soll dabei aber so klein wie möglich sein!

Erreichbarkeit der Hypothesen:

Generalisierungs-/Spezialisierungsschritte müssen minimal sein, um nicht an der richtigen Hypothese "vorbeizulaufen";

vom Ausgangspunkt aus muß die richtige Hypothese durch eine Kette von minimalen Schritten erreichbar sein.

θ-Subsumtion als Ordnung

θ-Subsumtion induziert eine Halbordnung, d.h.

reflexiv: $C \geq_\theta C$

transitiv: $C \geq_\theta D, D \geq_\theta E \ \models \ C \geq_\theta E$ aber

nicht antisymmetrisch: $C \geq_\theta D, D \geq_\theta C \ \not\models C = D$, sondern nur

C äquivalent D!

Shapiros Struktur des Hypothesenraums

Hypothesenraum nach θ -Subsumtion geordnet,

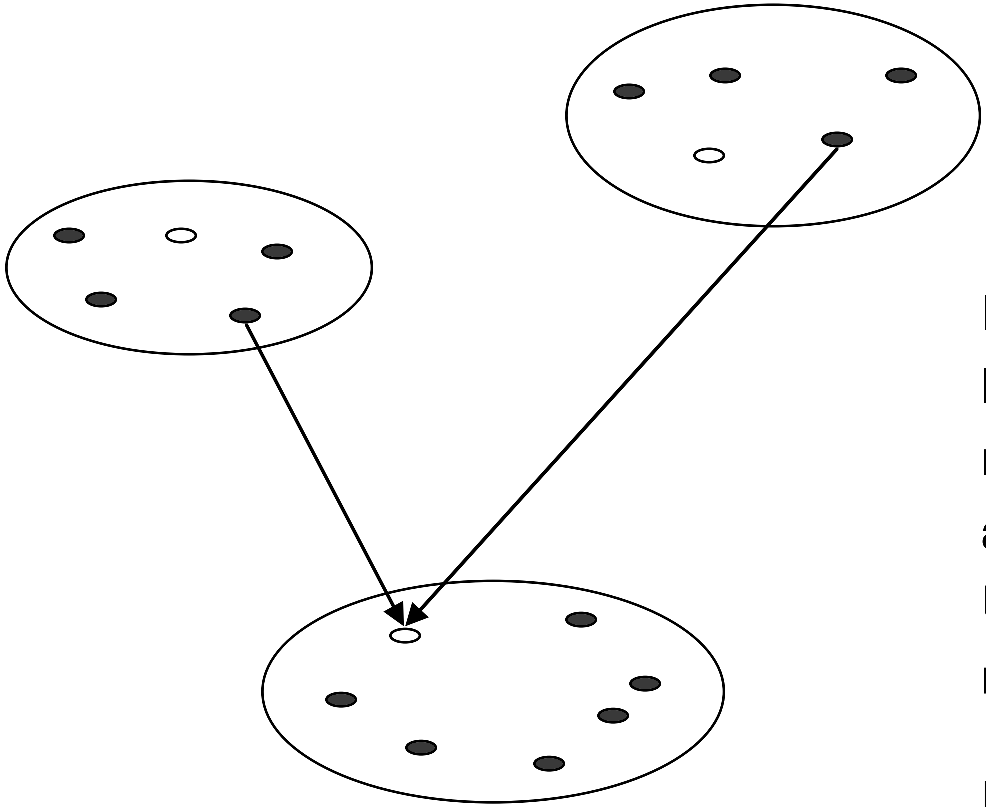
wobei nur reduzierte Formeln enthalten sind, d.h.:

zu jeder Äquivalenzklasse von Formeln ist nur die reduzierte Form im Hypothesenraum.

Beabsichtigter Effekt:

- die Anzahl der Literale steigt von der allgemeinsten (der leeren) Klausel zu den immer spezielleren Klauseln;
- Überanpassung an die Beispiele kann durch ein Längenmaß verhindert werden (Klauseln dürfen nicht zu lang werden).

Unvollständigkeit der Verfeinerung



Eine reduzierte Klausel

kann manchmal nur von

nicht-reduzierten Klauseln

aus erreicht werden!

Um alle Formeln zu erreichen,

müssen nicht-reduzierte Klauseln vorkommen dürfen, oder

Klauseln bei der Verfeinerung kürzer werden dürfen.

Patrick van der Laag 1995

Problemfall 1

P: { p (a, W), p (X, b), p (c, Y), p (Z, d) } gegebene, zu allgemeine Hypothese

P1: { p (a, b), p (X, b), p (c, Y), p (Z, d) }
P2: { p (a, b), p (c, b), p (c, Y), p (Z, d) } nicht-reduzierte Klauseln!
P3: { p (a, b), p (c, b), p (c, d), p (Z, d) }

Q: { p (a, b), p (c, b), p (c, d), p (a, d) } gewünschte Spezialisierung

reduzierte Form von P1:

P1': { p (a, b), p (c, Y), p (Z, d) } denn $P1\theta \subseteq P1 \setminus \{ p(X, b) \}$ mit $\theta = \{X / a\}$

Dieser Zwischenschritt ist aber kürzer als P und darf deshalb bei Shapiro nicht in dem Spezialisierungspfad vorkommen.

Problemfall 2

P: { p (X), ¬ q (X, a) } zu allgemeine Hypothese

R: { p(X), ¬ q (X, a), ¬ q (Y, Z) } nicht-reduzierte Klausel

Q: { p (X), ¬ q (X, a), ¬ q (Y, Z), ¬ q (Z, Y) } gewünschte Spezialisierung

Rθ ⊆ R \ ¬ q (Y, Z), wobei θ: { Y / X, Z / a}

R': { p (X), ¬ q (X, a) } reduzierte Form von R ist gerade P!

Vervollständigung der Spezialisierung

Hypothesenraum mit äquivalenten, nicht-reduzierten Klauseln der Länge m

eq (P) liefert alle zu P äquivalenten und um höchstens k Literale erweiterte Klauseln

- **an die reduzierte Klausel C werden 1...k Literale angehängt,**
- **angehängte Literale haben ein Prädikatensymbol und Vorzeichen, das in C vorkommt,**
- **einige Variablen in angehängten Literalen kommen nicht in C vor.**

Reine Umbenennungen sind in eq (P) nicht enthalten.

Beispiel für inverse Reduktion

C: { p (X, X) } k= 1	
eq (P): { { p (X, X), p (X, Y) },	statt Y und Z könnten auch
{ p (X, X), p (Y, X) },	beliebige andere Variablennamen
{ p (X, X), p (Y, Z) },	gewählt werden, die aber dann nur
{ p (X, X), p (Y, Y) })	Varianten darstellen.

Neue, vollständige Verfeinerung

sr_τ (P, Q) gilt für P, Q in LH, gdw.

1. **P θ-subsumiert echt Q und es gibt Klauseln P' in eq (P) und Q' in eq (Q), so daß Q' = P' θ, wobei θ = {X / Y } und X, Y in P'; oder**
2. **P θ-subsumiert echt Q und es gibt Klauseln P' in eq (P) und Q' in eq (Q), so daß θ: {X / f(Y_1, ..., Y_n)} und X kommt in P' vor und die Y_i sind verschiedene, nicht in P' vorkommende Variablen; oder**
3. **Q ist P ∪ {L}, wobei L ein generellstes Literal bzgl. P ist, das bzgl. Prädikatensymbol oder Vorzeichen von jedem Literal in P verschieden ist.**

sr_τ ist <u>vollständig</u>.

Die Beschränkung auf höchstens k anzuhängende Literale kann bei zu kleinem k natürlich zur Unvollständigkeit führen.

Verfeinerungsgraph

Der Verfeinerungsgraph, der durch die Relation sr_τ gebildet wird, ist unendlich.

Jeder einzelne Pfad in dem Verfeinerungsgraphen ist endlich.

Während es beim lgg zu zwei Klauseln genau eine Generalisierung gab,

gibt es zu zwei Klauseln sehr viele Spezialisierungen.

Der Verfeinerungsgraph von sr_τ ist breiter als der von sr, weil jetzt alle äquivalenten Klauseln mit untersucht werden müssen.

Das Verfahren von Nienhuys-Cheng und van der Laag zeigt die Schwierigkeit der Spezialisierung in der Prädikatenlogik.

Es ist praktisch nicht anwendbar.

Spezialisierung als Revision

Gegeben eine Theorie B und eine Menge von Fakten F = { f1, ..., fn} finde eine revidierte Theorie B — F, so daß Cn (B) ∩ F = 0.

Eine Theorie besteht aus dem, was tatsächlich aufgeschrieben ist (Basis B) und dem, was daraus logisch folgt (Cn (B)).

Eine Theorie ist <u>abgeschlossen</u>, wenn Cn (B) = B.

B: { p (a), q (X) :- p (X) }
Cn (B): { p (a), q (X) :- p (X), p (a) v p (b), p (a) v ¬ p (b), ..., q (a) }

Gärdenfors Postulate

1. **B — f sei eine abgeschlossene Theorie.**
2. **B — f $\subseteq$ B**
3. **Wenn f $\notin$ B, dann B — f = B.**
4. **Wenn f $\notin$ Cn (0), dann f $\notin$ B — f** **Erfolg**
5. **Wenn Cn (f) = Cn (g), dann B — f = B — g** **Erhalt**
6. **B $\subseteq$ Cn (B — f $\cup$ {f})** **Wiederherstellung**

Maximale korrekte Teilmengen

Sei B eine abgeschlossene Theorie und f ein auszuschließendes Fakt, dann ist

$$B \downarrow f : \{ B' \subseteq B \mid f \notin Cn(B') \text{ und für alle } B'' \text{ gilt: wenn } B' \subseteq B'' \subseteq B, \text{ dann } f \in Cn(B'') \}$$

$$B \downarrow q(a) : \{ \{ p(a), p(a) \vee p(b), p(a) \vee \neg p(b), \ldots \},$$
$$\{ q(X) \text{ :- } p(X), p(a) \vee p(b), \ldots \},$$
$$\{ q(X) \text{ :- } p(X), p(a) \vee \neg p(b), \ldots \}, \ldots \}$$

Revision nach Gärdenfors

— ist eine Revisionsoperation, die die Gärdenfors-Postulate befolgt, dann und nur dann, wenn

es eine Selektionsfunktion γ gibt mit $\gamma(B \downarrow f) \subseteq B \downarrow f$, so daß

$$B \text{—} f = \begin{cases} \bigcap \gamma(B \downarrow f), & \text{wenn } f \notin Cn(0) \\ B & \text{sonst.} \end{cases}$$

Je nach γ gibt es verschiedene Revisionsoperatoren.

Wählt γ nur eine Menge aus, so ist — eine *maxi-choice contraction*.

Nachteile der maxi-choice contraction

- **liefert abgeschlossene Theorie zurück -- also nur für Spielbeispiele geeignet!**
- **Unterschied zwischen expliziter Basis und abgeleiteten Aussagen verschwindet.**

  z.B. $B \downarrow p(a)$ könnte $q(a)$ enthalten.
- **Besserwisser-Effekt: eine abgeschlossene Theorie enthält für jede Aussage das Fakt oder seine Negation, d.h. $g \in Cn(B — f)$ oder $\neg g \in Cn(B — f)$**

Praktisches Szenario

Die Basis B wurde von Entwicklern sorgfältig konstruiert.

Eine Menge von Fakten aus Cn (B)

- **passen den Benutzern nicht oder**
- **gelten nicht mehr oder**
- **stellen sich als falsch heraus.**

Gerade nur diese Menge von Fakten soll nicht mehr gelten und so viel als möglich soll von der Basis erhalten bleiben.

Wir wollen die Basis revidieren, und zwar minimal!

Wrobels Postulate

1. $B - f \subseteq B \cup \{g' \mid \exists\, g \in B$ so daß $g \geq g'\}$ minimaler syntaktischer Abstand
2. $Cn(B - f) \subseteq Cn(B)$
3. Wenn $f \notin Cn(B)$, dann $B - f = B$.
4. Wenn $f \notin Cn(0)$, dann $f \notin Cn(B - f)$ Erfolg
5. Wenn $Cn(f) = Cn(g)$, dann $Cn(B - f) = Cn(B - g)$ Erhalt
6. $Cn(B) \subseteq Cn(B - f \cup \{f\})$ Wiederherstellung