Osservazioni su **Find-S**

Find-S può essere adattato ad altri Spazi delle Istanze ed Ipotesi.

L'idea base dell'algoritmo consiste nel calcolare una *generalizzazione minima* della ipotesi corrente quando questa non è consistente con l'esempio corrente.

Si noti che, ogni qualvolta che l'ipotesi corrente h è *generalizzata* portando ad una nuova ipotesi h' ($h' \geq_g h$), tutti gli esempi positivi presentati in precedenza sono soddisfatti dalla nuova ipotesi h' (infatti, poiché $h' \geq_g h$, si ha che
$\forall x \in X, \, (h(x) = 1) \rightarrow (h'(x) = 1))$

Infine, se il concetto da apprendere è contenuto in $\mathcal{H}$, tutti gli esempi negativi (per cui, $c(x) = 0$) sono soddisfatti automaticamente dalla ipotesi restituita da **Find-S** poiché tale ipotesi è la più specifica fra quelle consistenti, cioè quella che assegna il numero più piccolo di "1" alle istanze in X.

Esiste un motivo valido per preferire l'ipotesi consistente più specifica?

Uso degli esempi negativi...

Esiste un motivo valido per preferire l'ipotesi consistente più specifica ? NO!

Pertanto cerchiamo di capire come trovare l'insieme di TUTTE le ipotesi che sono consistenti con l'insieme di apprendimento (detto **Version Space**).

(Per semplificare l'esposizione, assumiamo che il Version Space abbia un' ipotesi più specifica di tutte, cioè quella restituita da **Find-S**).

Una ipotesi nel Version Space sarà più generale od equivalente a quella restituita da **Find-S**; in aggiunta, deve essere consistente con tutti gli esempi negativi.

Pertanto l'intuizione è di partire con un Version Space candidato inizialmente equivalente all'intero Spazio delle Ipotesi (nessun esempio ancora presentato), e poi usare gli esempi positivi per rimuovere ipotesi che sono troppo specifiche, e gli esempi negativi per rimuovere le ipotesi che sono troppo generali (algoritmo **Candidate-Elimination**).

Algoritmo **Candidate-Elimination**

Definisce implicitamente il Version Space tramite

- **Confine più Specifico (Specific Boundary)**

$$S \equiv \{s \in \mathcal{H} | consistente(s, Tr) \wedge (\neg \exists s' \in \mathcal{H})[s >_g s') \wedge consistente(s', Tr)])\}$$

- **Confine più Generale (General Boundary)**

$$G \equiv \{g \in \mathcal{H} | consistente(g, Tr) \wedge (\neg \exists g' \in \mathcal{H})[g' >_g g) \wedge consistente(g', Tr)])\}$$

Il Version Space è definito come

$$VS_{\mathcal{H},Tr} = \{h \in \mathcal{H} | (\exists s \in S)(\exists g \in G)(g \geq_g h \geq_g s)\}$$

Provate ad immaginare come funziona l'algoritmo...

Candidate-Elimination

inizializza G all'insieme delle ipotesi più generale e S all'insieme delle ipotesi più specifiche

for each $d \equiv (x, c(x)) \in Tr$ **do**

if $c(x) = 1$ /* esempio positivo */

rimuovi da G ogni ipotesi inconsistente con d

$\forall$ ipotesi $s \in S$ inconsistente con d

rimuovi s da S

aggiungi ad S tutte le generalizzazioni minime h di s t.c.

$consistente(h, d)$ ed $\exists g \in G$ t.c. $g \geq_g h$

rimuovi da S tutte le ipotesi s per cui $\exists s' \in S$ t.c. $s \geq_g s'$

if $c(x) = 0$ /* esempio negativo */

rimuovi da S ogni ipotesi inconsistente con d

$\forall$ ipotesi $g \in G$ inconsistente con d

rimuovi g da G

aggiungi a G tutte le specializzazioni minime h di s t.c.

$consistente(h, d)$ ed $\exists s \in S$ t.c. $h \geq_g s$

rimuovi da G tutte le ipotesi g per cui $\exists g' \in S$ t.c. $g' \geq_g g$

Version Space

Notare che la cardinalità del Version Space:

- può essere infi nita (se $\mathcal{H}$ è infi nito), ma sia S che G possono avere una rappresentazione fi nita
- in generale la sua cardinalità decresce con il crescere della cardinalità di Tr (pi`u vincoli per le ipotesi)

Assumendo che $c \in \mathcal{H}$, pi`uè piccola la cardinalità del Version Space, pi`u altaè la probabilità che selezionando un' ipotesi a caso $h \in VS_{\mathcal{H},Tr}$ si ottenga il concetto desiderato c, cioè $h \equiv c$.

Apprendimento PAC

PAC Learning (Probably Approximately Correct Learning)

Assume che: input ed output sono binari, non c'è rumore, le istanze sono estratte da X concordemente ad una distribuzione di probabilità $\mathcal{D}$ *arbitraria* ma *stazionaria*.

Il framework di apprendimento PAC cerca di rispondere alle seguenti domande:

- Sotto quali condizioni apprendere con successo è possibile o impossibile ?
- Sotto quali condizioni si può assicurare che un particolare algoritmo di apprendimento apprenda con successo ?

Apprendimento PAC

Consideriamo una classe C di possibili concetti target (funzioni che vogliamo apprendere) definita su uno Spazio delle Istanze X (con istanze di dimensione m), ed un algoritmo di apprendimento L che utilizza uno Spazio delle Ipotesi $\mathcal{H}$.

Def.: C è PAC-apprendibile da L usando $\mathcal{H}$ se per ogni

- $c \in C$,
- distribuzione $\mathcal{D}$ su X,
- ϵ tale che $0 < \epsilon < 1/2$,
- δ tale che $0 < \delta < 1/2$,

l'algoritmo di apprendimento L con probabilità almeno $(1 - \delta)$ restituisce una ipotesi $h \in \mathcal{H}$ tale che $error_{\mathcal{D}}(h) \leq \epsilon$, in tempo che è **polinomiale** in $1/\epsilon$, $1/\delta$, m, e $size(c)$ (spazio di memoria necessario per rappresentare c).

Apprendimento PAC

Si può dimostrare che assumendo:

- $c \in \mathcal{H}$
- L consistente, cioè che restituisce una ipotesi h consistente con Tr, o equivalentemente $h \in VS_{\mathcal{H},Tr}$ (ad esempio, **Find-S** è consistente)

allora, con probabilità almeno $(1-\delta)$, L restituisce una ipotesi $h \in \mathcal{H}$ tale che $error_{\mathcal{D}}(h) < \epsilon$ se il numero di esempi di apprendimento n soddisfa la seguente disuguaglianza:

$$n \geq \frac{1}{\epsilon}(ln(|\mathcal{H}|) + ln(\frac{1}{\delta}))$$

dove $|\mathcal{H}|$ è la cardinalità dello Spazio delle Ipotesi e $ln()$ è il logaritmo naturale

Prova...

Idea base: bisogna dare un limite al numero di esempi necessario ad assicurare che il Version Space (ricordiamo che L è consistente) non contiene ipotesi non "accettabili":

$$(\forall h \in VS_{\mathcal{H},Tr})\ error_{\mathcal{D}}(h) < \epsilon$$

Prova...

$$(\forall h \in VS_{\mathcal{H},Tr})\ error_{\mathcal{D}}(h) < \epsilon$$

Primo risultato: se $|\mathcal{H}| < \infty$, $|Tr| = n > 0$ e Tr è costituito da esempi di un concetto target c estratti indipendentemente ed a caso, allora per ogni $0 \leq \epsilon \leq 1$, la probabilità che la disuguaglianza **NON** sia soddisfatta è minore od uguale a $|\mathcal{H}|e^{-\epsilon n}$:

- siano $h_1, \ldots, h_k$ tutte le ipotesi con $error_{\mathcal{D}}(h) \geq \epsilon$; la disuguaglianza NON è soddisfatta se e solo se ALMENO una di tali ipotesi è consistente con Tr
- la probabilità che una di tali ipotesi sia consistente con un singolo esempio è $(1-\epsilon)$, e quindi per n esempi indipendenti la probabilità è $(1-\epsilon)^n$
- poiché esistono k di tali ipotesi, la probabilità che ci interessa è a più $k(1-\epsilon)^n \leq |\mathcal{H}|(1-\epsilon)^n$ (poiché $k \leq |\mathcal{H}|$)
- infine, poiché $(1-\epsilon) \leq e^{-\epsilon}$ se $0 \leq \epsilon \leq 1$, abbiamo $|\mathcal{H}|(1-\epsilon)^n \leq |\mathcal{H}|e^{-\epsilon n}$

Prova...

Cosa abbiamo ottenuto: abbiamo un limite superiore alla probabilità che usando un Tr con n esempi, il Version Space contenga qualche ipotesi CATTIVA, cioè che L restituisca h, una delle ipotesi cattive, per cui $error_{\mathcal{D}}(h) \geq \epsilon$!

Quindi, se vogliamo che tale probabilità sia inferiore a livello desiderato δ

$$|\mathcal{H}|e^{-\epsilon n} \leq \delta$$

bisogna usare un valore per n per cui

$$n \geq \frac{1}{\epsilon}(ln(|\mathcal{H}|) + ln(\frac{1}{\delta}))$$

o, alternativamente, se si usa un tale valore per n, con probabilità $1 - \delta$ l'ipotesi h restituita da L avrà $error_{\mathcal{D}}(h) < \epsilon$

Apprendimento PAC: esempio $\mathcal{H}_4$

Congiunzione di m letterali

- Spazio delle Istanze $\rightarrow$ stringhe di m bit: $X = \{s | s \in \{0,1\}^m\}$
- Spazio delle Ipotesi $\rightarrow$ tutte le sentenze logiche che riguardano i letterali $l_1, \ldots, l_m$ (anche in forma negata, $\neg l_i$) e che contengono solo l'operatore $\wedge$ (**and**):

$$\mathcal{H} = \{f_{\{i_1,\ldots,i_j\}}(s) | f_{\{i_1,\ldots,i_j\}}(s) \equiv L_{i_1} \wedge L_{i_2} \wedge \cdots \wedge L_{i_j},$$
$$\text{dove } L_{i_k} = l_{i_k} \text{ oppure } \neg l_{i_k}, \{i_1, \ldots, i_j\} \subseteq \{1, \ldots, 2m\}\}$$

Notare che se in una formula un letterale compare sia affermato che negato, allora la formula ha sempre valore di verità $false$ (formula non soddisfacibile)

Quindi, tutte le formule che contengono almeno un letterale sia affermato che negato sono equivalenti alla funzione che vale sempre $false$

Apprendimento PAC: esempio $\mathcal{H}_4$

E' la congiunzione di m letterali PAC-apprendibile usando **Find-S** con $\mathcal{H}_4$?

Apprendimento PAC: esempio $\mathcal{H}_4$

E' la congiunzione di m letterali PAC-apprendibile usando **Find-S** con $\mathcal{H}_4$? Si !

Infatti:

- ogni congiunzione di m letterali è inclusa in $\mathcal{H}_4$

Apprendimento PAC: esempio $\mathcal{H}_4$

E' la congiunzione di m letterali PAC-apprendibile usando **Find-S** con $\mathcal{H}_4$? Si !

Infatti:

- ogni congiunzione di m letterali è inclusa in $\mathcal{H}_4$
- **Find-S** è consistente

Apprendimento PAC: esempio $\mathcal{H}_4$

E' la congiunzione di m letterali PAC-apprendibile usando **Find-S** con $\mathcal{H}_4$? Si !

Infatti:

- ogni congiunzione di m letterali è inclusa in $\mathcal{H}_4$
- **Find-S** è consistente
- $|\mathcal{H}_4| = 3^m + 1$ e poiché $\frac{1}{\epsilon}(ln(3^{m+1}) + ln(\frac{1}{\delta})) > \frac{1}{\epsilon}(ln(3^m + 1) + ln(\frac{1}{\delta}))$

$$n \geq \frac{1}{\epsilon}((m+1)ln(3) + ln(\frac{1}{\delta}))$$

quindi n è polinomiale in $1/\epsilon, 1/\delta, m$, e $size(c)$ (che non compare)

Apprendimento PAC: esempio $\mathcal{H}_4$

E' la congiunzione di m letterali PAC-apprendibile usando **Find-S** con $\mathcal{H}_4$? Si !

Infatti:

- ogni congiunzione di m letterali è inclusa in $\mathcal{H}_4$
- **Find-S** è consistente
- $|\mathcal{H}_4| = 3^m + 1$ e poiché $\frac{1}{\epsilon}(ln(3^{m+1}) + ln(\frac{1}{\delta})) > \frac{1}{\epsilon}(ln(3^m + 1) + ln(\frac{1}{\delta}))$

$$n \geq \frac{1}{\epsilon}((m+1)ln(3) + ln(\frac{1}{\delta}))$$

quindi n è polinomiale in $1/\epsilon$, $1/\delta$, m, e $size(c)$ (che non compare)

- per ogni esempio di apprendimento **Find-S** impiega tempo lineare nella dimensione della ipotesi corrente (e tale dimensione è $\geq size(c)$) e nella dimensione dell'input (m), quindi di nuovo polinomiale, e globalmente è polinomiale per Tr

Apprendimento PAC: esempio $\mathcal{H}_4$

E' la congiunzione di m letterali PAC-apprendibile usando **Find-S** con $\mathcal{H}_4$? Si !

Infatti:

- ogni congiunzione di m letterali è inclusa in $\mathcal{H}_4$
- **Find-S** è consistente
- $|\mathcal{H}_4| = 3^m + 1$ e poiché $\frac{1}{\epsilon}(ln(3^{m+1}) + ln(\frac{1}{\delta})) > \frac{1}{\epsilon}(ln(3^m + 1) + ln(\frac{1}{\delta}))$

$$n \geq \frac{1}{\epsilon}((m+1)ln(3) + ln(\frac{1}{\delta}))$$

  quindi n è polinomiale in $1/\epsilon$, $1/\delta$, m, e $size(c)$ (che non compare)
- per ogni esempio di apprendimento **Find-S** impiega tempo lineare nella dimensione della ipotesi corrente (e tale dimensione è $\geq size(c)$) e nella dimensione dell'input (m), quindi di nuovo polinomiale, e globalmente è polinomiale per Tr

Quindi tutte le condizioni per la PAC-apprendibilità sono soddisfatte !