Apprendimento di Concetti

Corso di AA, anno 2017/18, Padova

Fabio Aiolli

18 Ottobre 2017

Concept Learning

- Definizione di una categoria (concetto) basata su esempi positivi e negativi della categoria
- È una istanza di Supervised Learning, ovvero può essere formulato come il problema di cercare in uno spazio di ipotesi potenziali quella che meglio si adatta ("fitta") gli esempi di training
- Nel caso specifico vedremo che possiamo avvantaggiarci della struttura dello spazio delle ipotesi per rendere la ricerca piú efficiente

Esempio del libro

- Determinare i giorni in cui l'amico Aldo deciderà di praticare il suo sport d'acqua preferito
- Ogni esempio (giorno) viene rappresentato come un insieme di attributi discreti
- Il task è quello di predire il valore dell'attributo `EnjoySport` sulla base degli altri attributi

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

Esempio del libro

- **Given:**
  - Instances X: Possible days, each described by the attributes
    - *Sky* (with possible values *Sunny*, *Cloudy*, and *Rainy*),
    - *AirTemp* (with values *Warm* and *Cold*),
    - *Humidity* (with values *Normal* and *High*),
    - *Wind* (with values *Strong* and *Weak*),
    - *Water* (with values *Warm* and *Cool*), and
    - *Forecast* (with values *Same* and *Change*).
  - Hypotheses H: Each hypothesis is described by a conjunction of constraints on the attributes *Sky*, *AirTemp*, *Humidity*, *Wind*, *Water*, and *Forecast*. The constraints may be "?" (any value is acceptable), "$\emptyset$" (no value is acceptable), or a specific value.
  - Target concept c: $EnjoySport : X \rightarrow \{0, 1\}$
  - Training examples D: Positive and negative examples of the target function (see Table 2.1).
- **Determine:**
  - A hypothesis h in H such that $h(x) = c(x)$ for all x in X.

Concept Learning: alcune definizioni

- Un concetto in uno spazio delle istanze (instance space) $\mathcal{X}$ è definito come una funzione booleana su $\mathcal{X}$, $c : \mathcal{X} \to \{0, 1\}$
- Un esempio (di un concetto c) su uno spazio delle istanze $\mathcal{X}$ è definito come una coppia $(x, c(x))$, dove $x \in \mathcal{X}$ e $c()$ è una funzione booleana su $\mathcal{X}$
- Sia $h : \mathcal{X} \to \{0, 1\}$ una funzione booleana su $\mathcal{X}$, diciamo che h soddisfa $x \in \mathcal{X}$ se $h(x) = 1$
- Sia $h : \mathcal{X} \to \{0, 1\}$ una funzione booleana su $\mathcal{X}$ e $(x, c(x))$ un esempio di c. Diciamo che h è consistente con l'esempio se $h(x) = c(x)$. Similmente, una ipotesi h è consistente con un insieme di esempi D (e si indica $h \triangleright D$) se e solo se $h(x) = c(x)$ per ogni esempio $\langle x, c(x) \rangle$.

Struttura dello spazio delle ipotesi

Definizione: Siano h_i e h_j funzioni booleane definite su uno spazio delle istanze $\mathcal{X}$. Diciamo che h_i è più generale o equivalente di h_j ($h_i \geq_g h_j$) se e solo se

$$\forall x \in \mathcal{X}, \quad [(h_j(x) = 1) \Rightarrow (h_i(x) = 1)]$$

Esempi:

- $l_1 \geq_g (l_1 \wedge l_2)$ e $l_2 \geq_g (l_1 \wedge l_2)$
- $l_1 \not\geq_g l_2$ e $l_2 \not\geq_g l_1$ (non comparabili)

Algoritmo FIND-S

Consideriamo lo spazio delle ipotesi delle congiunzioni di m letterali (positivi e negativi)

/* Find-S trova l'ipotesi più specifica che è consistente con gli esempi di apprendimento */

- Input: Insieme di apprendimento Tr
- Inizializza h con l'ipotesi più specifica: $h \equiv l_1 \wedge \neg l_1 \wedge \cdots \wedge l_m \wedge \neg l_m$
- Per ogni istanza di apprendimento positiva $(x, True) \in Tr$, rimuovi da h ogni letterale non soddisfatto in x
- Restituisci h

Esempio di applicazione di FIND-S con $m = 5$

Esempio (+)	Ipotesi corrente
	$h_0 = l_1 \wedge \neg l_1 \wedge l_2 \wedge \neg l_2 \wedge l_3 \wedge \neg l_3 \wedge l_4 \wedge \neg l_4 \wedge l_5 \wedge \neg l_5$
11010	$h_1 = l_1 \wedge l_2 \wedge \neg l_3 \wedge l_4 \wedge \neg l_5$
10010	$h_2 = l_1 \wedge \neg l_3 \wedge l_4 \wedge \neg l_5$
10110	$h_3 = l_1 \wedge l_4 \wedge \neg l_5$
10100	$h_4 = l_1 \wedge \neg l_5$
00100	$h_5 = \neg l_5$

Notare che:

- $h_0 \leq_g h_1 \leq_g h_2 \leq_g h_3 \leq_g h_4 \leq_g h_5$
- Ad ogni passo, l'ipotesi corrente h_i è sostituita con una sua generalizzazione minima h_{i+1} consistente con l'esempio corrente

Osservazioni su FIND-S

- Find-S può essere adattato ad altri spazi delle istanze e delle ipotesi
- Idea base: calcolare una generalizzazione minima della ipotesi corrente quando essa non è consistente con l'esempio corrente
- Ogni volta che l'ipotesi corrente h_i viene generalizzata con una ipotesi h_{i+1} ($h_{i+1} \geq_g h_i$), tutti gli esempi positivi presentati in precedenza continuano ad essere soddisfatti. Infatti, poiché $h_{i+1} \geq_g h_i$, avremo $\forall x \in \mathcal{X}, (h_i(x) = 1) \Rightarrow (h_{i+1}(x) = 1)$
- Se il concetto target $c()$ è contenuto nello spazio delle ipotesi, allora tutti gli esempi negativi rimarranno consistenti con l'ipotesi corrente durante tutta l'evoluzione dell'algoritmo. Questo perché ad ogni passo l'ipotesi corrente è l'ipotesi piú specifica (ovvero quella che assegna il minor numero di 1 alle istanze in $\mathcal{X}$) consistente con gli esempi (positivi) visti fino a quel momento.

Limiti di FIND-S

- Convergenza al target concept? Non c'è modo di determinare se la regola trovata da FIND-S è l'unica in H consistente con gli esempi (ovvero il concetto target corretto), oppure se ci sono altre ipotesi consistenti con gli esempi di training.
- Perché preferire l'ipotesi più specifica? e non quella più generale o un'altra di generalità intermedia.
- Gli esempi sono consistenti? Nelle applicazioni pratiche gli esempi di training potrebbero contenere rumore (noise). In questo caso FIND-S potrebbe addirittura convergere ad una ipotesi inconsistente con gli esempi negativi.
- Cosa succede se ci sono più ipotesi massimamente specifiche? In questo caso FIND-S dovrebbe essere esteso per prevedere una sorta di backtracking sulle scelte di generalizzazione delle ipotesi, rendendo possibile percorrere un ramo diverso nell'ordine parziale.

Uso degli esempi negativi

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

NO

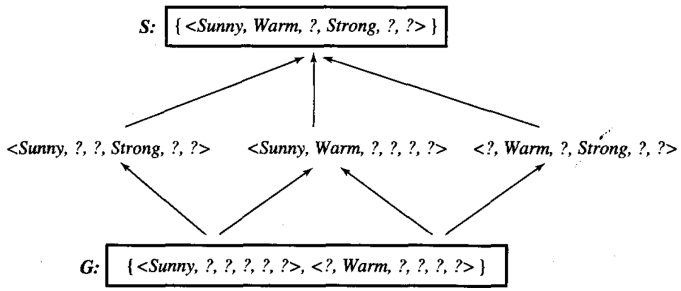
Vediamo invece come sia possibile trovare TUTTE le ipotesi consistenti con un insieme di apprendimento (detto Version Space).

IDEA: Algoritmo Candidate-Elimination

- Partire con un version space candidato uguale all'intero spazio delle ipotesi.
- Usare gli esempi positivi per rimuovere le ipotesi troppo specifiche dal version space candidato
- Usare gli esempi negativi per rimuovere le ipotesi troppo generali dal version space candidato

Candidate-Elimination (definizioni)

- Il version space è il sottoinsieme di ipotesi in H consistente con gli esempi di training.
- Il general boundary di uno spazio di ipotesi H e dati di training D è l'insieme dei membri di H di massima generalità consistenti con D, $G \equiv \{g \in H | g \triangleright D \wedge (\nexists g' \in H)[(g' >_g g) \wedge g' \triangleright D]\}$.
- Lo specific boundary di uno spazio di iptesi H e dati di training D è l'insieme dei membri di H di massima specificità consistenti con D, $S \equiv \{s \in H | s \triangleright D \wedge (\nexists s' \in H)[(s >_g s') \wedge s' \triangleright D]\}$.
- Dat G e S, si può dimostrare che una ipotesi h appartiene al version space se e solo se: $(\exists s \in S)(\exists g \in G)(g \geq_g h \geq_g s)$.



Algoritmo Candidate-Elimination

Initialize G to the set of maximally general hypotheses in H
Initialize S to the set of maximally specific hypotheses in H
For each training example d, do

- If d is a positive example
  - Remove from G any hypothesis inconsistent with d
  - For each hypothesis s in S that is not consistent with d
    - Remove s from S
    - Add to S all minimal generalizations h of s such that
      - h is consistent with d, and some member of G is more general than h
    - Remove from S any hypothesis that is more general than another hypothesis in S
- If d is a negative example
  - Remove from S any hypothesis inconsistent with d
  - For each hypothesis g in G that is not consistent with d
    - Remove g from G
    - Add to G all minimal specializations h of g such that
      - h is consistent with d, and some member of S is more specific than h
    - Remove from G any hypothesis that is less general than another hypothesis in G

Recap

Nozioni

- Consistenza e soddisfacimento per le ipotesi
- Version space
- Gerarchia su spazi delle ipotesi: Generalità e specificità
- Algoritmo Find-S e suoi limiti
- Caratterizzazione del version space mediante boundaries
- Algoritmo Candidate-Elimination

Esercizi

- Più facile: Implementazione in Python dell'algoritmo Find-S
- Più difficile: Implementazione in Python dell'algoritmo Candidate-Elimination