

Theorie des Algorithmischen Lernens
Sommersemester 2006

Teil 2.4: Lernen formaler Sprachen:
Inkrementelles Lernen

Version 1.1

Gliederung der LV

Teil 1: Motivation

1. Was ist Lernen
2. Das Szenario der Induktiven Inferenz
3. Natürlichkeitsanforderungen

Teil 2: Lernen formaler Sprachen

1. Grundlegende Begriffe und Erkennungstypen
2. Die Rolle des Hypothesenraums
3. Lernen von Patternsprachen
4. Inkrementelles Lernen

Teil 3: Lernen endlicher Automaten

Teil 4: Lernen berechenbarer Funktionen

1. Grundlegende Begriffe und Erkennungstypen
2. Reflexion

Teil 5: Informationsextraktion

1. Island Wrappers
2. Query Scenarios

Incremental Learning

Basic idea:

Modify previous hypothesis instead of recomputing it from scratch

Iterative Learning

First approach to incremental learning:

- extend IIM to two arguments:
 - *previous hypothesis*
 - *current example*
- need ***initial hypothesis***
 - need some convention, lets set it to **-1**

Properties

- new hypothesis only depends on previous hypothesis and new example
- no *per se* information about the number of examples already seen

Iterative Learning

First approach to to definition:

An IIM M *ItTxt* \mathcal{H} -identifies L iff, for every text $t = (x_n)_{n \in \mathbb{N}}$ for L , the following conditions are fulfilled:

- (1) $h_0 = M(-1, x_0)$
 $h_{n+1} = M(h_n, x_{n+1})$
- (2) the sequence $(h_n)_{n \in \mathbb{N}}$ converges to a number j with $h_j = L$.

Iterative Learning

Definition 2.4.1:

Let \mathcal{L} be an indexable class, let $L \in \mathcal{L}$ be a language, and let $\mathcal{H} = (h_j)_{j \in \mathbb{N}}$ be a hypothesis space.

An IIM M **ItTxt \mathcal{H}** -identifies L iff, for every text $t = (x_n)_{n \in \mathbb{N}}$ for L , the following conditions are fulfilled:

(1) for all $n \in \mathbb{N}$, $M_n(t)$ is defined, where

(i) $M_0(t) = M(-1, x_0)$,

(ii) $M_{n+1}(t) = M(M_n(t), x_{n+1})$.

(2) the sequence $(M_n(t))_{n \in \mathbb{N}}$ converges to a number j with $h_j = c$.

Surprise?

We could also use our old (unary) concept of IIM:

An IIM M works **iteratively** iff $M(t_x) = M(t'_{x'})$ implies $M(t_x \circ y) = M(t'_{x'} \circ y)$

Encoding Ideas

Example 1:

Consider the set of all finite languages

$$M(-1, w) = \{w\}$$

$$M(h, w) = h \cup \{w\}$$

Hypothesis *encodes* information about the previous examples

- but it can only contain finite amount of information
 - Otherwise no convergence could be achieved

Counterexample:

Consider the set of all co-finite languages

Bounded Example Memory

Another approach to incremental learning:

- Why only use the last example?
 - Use the last 17 examples...
- Extension: the learning IIM decides which examples to store
 - has an *internal example memory*
 - hypotheses are computed as usual in dependence on
 - * the previous hypothesis
 - * the current example
 - * the stored examples
 - example memory needs to be bounded
 - * otherwise we would be in the *Lim* setting

-
- approach results in *2 sequences*:
 - sequence of hypotheses
 - sequence of content of example memory

Bounded Example Memory

Definition 2.4.2:

Let \mathcal{L} be an indexable class, let $L \in \mathcal{L}$ be a language, and let $\mathcal{H} = (h_j)_{j \in \mathbb{N}}$ be a hypothesis space.

Moreover, let $k \in \mathbb{N}$. An IIM M ***Bem_kTxt_H***-identifies L iff, for every text $t = (x_n)_{n \in \mathbb{N}}$ for L , the following conditions are fulfilled:

(1) for all $n \in \mathbb{N}$, $M_n(t)$ is defined, where

(i) $M_0(t) = M(\langle -1, \emptyset \rangle, x_0) = \langle j_0, S_0 \rangle$

- with $S_0 \subseteq \{x_0\}$ and $\text{card}(S_0) \leq k$

(ii) $M_{n+1}(t) = M(M_n(t), x_{n+1}) = \langle j_{n+1}, S_{n+1} \rangle$

- with $S_{n+1} \subseteq S_n \cup \{x_{n+1}\}$ and $\text{card}(S_{n+1}) \leq k$.

(2) the j_n in the sequence $(\langle j_n, S_n \rangle)_{n \in \mathbb{N}}$ of M 's guesses converge to a number j with $h_j = c$.

Remark: $It\text{Txt} = \text{Bem}_0\text{Txt}$.

Incremental vs. Standard Learning

Theorem 2.4.1:

$FinTxt \subset ItTxt$

Proof: [Exercise](#)

Theorem 2.4.2:

For all $k \in \mathbb{N}$: $Bem_k Txt \subset ConsvTxt$.

Sketch of proof.

$Bem_k Txt \subseteq ConsvTxt$:

- search for a stabilizing sequence similar to the constructions in the last proofs

Incremental vs. Standard Learning

$ConsvTxt \setminus Bem_k Txt \neq \emptyset$:

Consider $\mathcal{L} = (L_j)_{j \in \mathbb{N}}$ with $L_j = \{a\}^* \setminus \{a^j\}$.

Exercise: Show $\mathcal{L} \in ConsvTxt$.

$\mathcal{L} \notin Bem_k Txt$ for any $k \in \mathbb{N}$:

- there exists a stabilizing sequence σ for L_1

- let m be the maximal length of strings in σ

- now consider sequences of the following form:

$$\sigma \circ \underbrace{a^{m+1}, a^{m+2}, \dots, a^{m+n}, a^{m+n+2}, a^{m+n+3}, \dots}_{\tau} \circ a \circ \dots,$$

- which form texts for languages L_{m+n+1}

- but:

- after seeing σ , M cannot encode any further information in its hypothesis until a appears

- hence, all information must be stored in the example memory

- but: this memory is limited, hence for long τ , M cannot distinguish it

Influence of the Size of the Example Memory

Theorem 2.4.3:

For all $k \in \mathbb{N}$: $Bem_k Txt \subset Bem_{k+1} Txt$.

Separating class \mathcal{L}_{bem_k} :

$$L_0 = \{a\}^*$$

$$L_{j,l_0,\dots,l_k} = \{a^m \mid 1 \leq m \leq j\} \cup \{b^{j+1}, a^{l_0}, \dots, a^{l_k}\}$$

It holds $\mathcal{L}_{bem_{k+1}} \in Bem_{k+1} Txt \setminus Bem_k Txt$

Incremental Learning from Informant

Definition for informant analogously.

Theorem 2.4.4:

$$FinInf \subset ItInf \subset LimInf$$

Proof: [Exercise](#)

Incremental Learning from Informant

Surprise:

Theorem 2.4.5:

$$\text{Bem}_1 \text{Inf} = \text{LimInf}$$

Proof.

Idea:

- the 1-bounded example-memory learner M outputs as hypothesis a triple (F, m, j) along with a singleton set containing the one data element stored
 - the triple (F, m, j) consists of a finite set F and two numbers m and j .
 - it is used to describe a finite variant of the language L_j , namely the language $F \cup L_j^{\vec{m}}$.
 - intuitively, $L_j^{\vec{m}}$ is the part of the language L_j that definitely does not contradict the data seen so far, while F is used to handle exceptions.

Incremental Learning from Informant

Let $L \in \mathcal{L}$ and let $i = ((x_n, b_n))_{n \in \mathbb{N}}$ be any informant for L .

Let $(w_j)_{j \in \mathbb{N}}$ denote the lexicographically ordered enumeration of all elements in Σ^* .

For all $m \in \mathbb{N}$ and all $L \subseteq \Sigma^*$, we set $L^m = \{w_z \mid z \leq m, w_z \in L\}$ and $L^{\vec{m}} = \{w_z \mid z > m, w_z \in L\}$.

Incremental Learning from Informant

Stage 0. On input (x_0, b_0) do the following:

Fix $m \in \mathbb{N}$ with $w_m = x_0$. Determine the least j such that L_j is consistent with (x_0, b_0) . Set $F = L_j^m$ and $S = \{(x_0, b_0)\}$. Output $\langle (F, m, j), S \rangle$ and goto Stage 1.

Stage n , $n \geq 1$. On input $\langle (F, m, j), S \rangle$ and (x_n, b_n) proceed as follows:

Let $S = \{(x, b)\}$. Fix $z, z' \in \mathbb{N}$ such that $w_z = x$ and $w_{z'} = x_n$. If $z' > z$, set $S' = \{(x_n, b_n)\}$. Otherwise, set $S' = S$. Test whether $h_{(F, m, j)} = F \cup L_j^m$ is consistent with (x_n, b_n) . In case it is, goto (A). Otherwise, goto (B).

(A) Output $\langle (F, m, j), S' \rangle$ and goto Stage $n + 1$.

(B) If $z' \leq m$, goto ($\beta 1$). If $z' > m$, goto ($\beta 2$).

($\beta 1$) If $b_n = +$, set $F' = F \cup \{x_n\}$. If $b_n = -$, set $F' = F \setminus \{x_n\}$. Output $\langle (F', m, j), S' \rangle$ and goto Stage $n + 1$.

($\beta 2$) Determine $l = \max\{z, z'\}$ and $F' = \{w_r \mid r \leq l, w_r \in h_{(F, m, j)}\}$. If $b_n = +$, set $F'' = F' \cup \{x_n\}$. If $b_n = -$, set $F'' = F' \setminus \{x_n\}$. Search for the least index $k > j$ such that L_k is consistent with (x_n, b_n) . Then, output $\langle (F'', l, k), S' \rangle$ and goto Stage $n + 1$.

Summary

