Finite Identifiability and Knowledge Update

Dick de Jongh[1]

Institute for Logic, Language and Computation, University of Amsterdam

Social Dynamics of Information Change - LogiCIC workshop 2013
Amsterdam, December 4, 2013

[1]in cooperation with Nina Gierasimczuk

Outline

Preliminaries

Preset Learning

Eliminative Power and Complexity

Knowledge Update

Fastest Learning

Outline

Basis

N. Gierasimczuk and D. de Jongh, On the Complexity of Conclusive Update. *The Computer Journal*, 56(3): 365-377, 2013.

Approach

So, we will first introduce finite identifiability.

Then connect it to knowledge update in this context.

Preliminaries

In Angluin-style we will restrict to uniformly recursive families of languages.

Definition

We call any $L \subseteq \mathbb{N}$ a language. An indexed family of recursive languages is a class $\mathcal{L} = (L_i)_{i \in \mathbb{N}}$ for which a computable function $f : \mathbb{N} \times \mathbb{N} \to \{0, 1\}$ exists that uniformly decides $\mathcal{L}$, i.e.,

$$f(i, w) = \begin{cases} 1 & \text{if } w \in L_i, \\ 0 & \text{if } w \notin L_i. \end{cases}$$

.

Notation

Definition

A text t of L is an infinite sequence of all the elements from L.

- t_n is the n-th element of t;
- $t[n]$ is the sequence $(t_0, t_1, \ldots, t_{n-1})$;
- $\mathrm{set}(t)$ is the set of elements that occur in t;
- M is a learning function — a recursive map from finite data sequences to indices of hypotheses, $M : \mathbb{N}^* \to \mathbb{N} \cup \{\uparrow\}$. The function can be undefined, written as $\uparrow$.

Identification in the limit

Definition (Gold 1967)

A learning function M:

1. identifies $L_i \in \mathcal{L}$ in the limit on t iff, when inductively given t, at some point M gives an output j such that $L_j = L_i$ and continues giving the same output ever after;
2. identifies $L_i \in \mathcal{L}$ in the limit iff it identifies L_i in the limit on every t for L_i;
3. identifies $\mathcal{L}$ in the limit iff it identifies every $L_i \in \mathcal{L}$ in the limit.

A class $\mathcal{L}$ is identifiable in the limit iff there is an M that identifies $\mathcal{L}$ in the limit.

Finite identifiability from positive data

Definition

A learning function M:

1. finitely identifies $L_i \in \mathcal{L}$ on t iff, when inductively given t, at some point M gives a single output j such that $L_j = L_i$;
2. finitely identifies $L_i \in \mathcal{L}$ iff it finitely identifies L_i on every t for L_i;
3. finitely identifies $\mathcal{L}$ iff it finitely identifies every $L_i \in \mathcal{L}$.

A class $\mathcal{L}$ is finitely identifiable iff there is an M that finitely identifies $\mathcal{L}$.

Epistemic Spaces I

One can transform the above definitions to an epistemic setting.

- The set $\mathbb{N}$ is replaced by the set of propositional variables $P = \{p_0, p_1, \ldots\}$,
- Languages are replaced by possible worlds, which are given by sets of propositional variables.
- An indexed family of languages $\mathcal{L}$ is replaced by an indexed family of possible worlds.

Since the truth value of a propositional variable is in epistemic logic commonly considered to be decidable this does not seem to be restrictive with respective to the traditional setting.

One can see learning as an infinite iteration of updating.

Epistemic Spaces II

The introduction of computable learning functions is of course an uncommon appearance in that setting.

The definitions of learning (classes of) languages can now be transformed to the epistemic setting. In general (i.e. in the identifiability in the limit case) it will then be more natural to let the hypotheses not be single worlds but propositions, sets of worlds. In this lecture we will keep the language style way of speaking as basic.

For knowledge update it would be much more natural to at least allow negative information: p_i is not in the language (possible world), of course in the form $\neg p_i$. We don't do that here, but it makes no essential difference to the results.

Characterization of finite identifiability

Definition

A set D_i is a definite finite tell-tale set (DFTT) for $L_i \in \mathcal{L}$ if

1. $D_i \subseteq L_i$,
2. D_i is finite, and
3. for any index j, if $D_i \subseteq L_j$ then $L_i = L_j$.

Theorem (Mukouchi 1992, Lange and Zeugmann 1992)

A class $\mathcal{L}$ is finitely identifiable from positive data iff there is an effective procedure $\mathcal{D} : \mathbb{N} \to \mathcal{P}^{<\omega}(\mathbb{N})$, given by $n \mapsto \mathcal{D}_n$, that on input i produces a definite finite tell-tale for L_i.

Replacing sets by characteristic functions

Definite tell-tale sets can be presented by a decision procedure deciding for each finite set whether to count it as a tell-tale for one of the languages.

Definition

A dftt-function for $\mathcal{L}$ is a recursive function $f_{dftt} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0, 1\}$, s.t.:

1. if $f_{dftt}(S, i) = 1$, then S is a DFTT of L_i;
2. for every $i \in \mathbb{N}$ there is a finite $S \subseteq \mathbb{N}$, s.t. $f_{dftt}(S, i) = 1$.

Theorem

$\mathcal{L}$ is finitely identifiable from positive data iff there is a dftt-function for $\mathcal{L}$.

f_{dftt} may not 'know' all DFTTs but accesses at least one.

Set-drivenness

Definition (Wexler and Cullicover 1980)

A learning function M is said to be set-driven w.r.t. $\mathcal{L}$ iff

for any two texts t_1 and t_2 for some languages in $\mathcal{L}$ and any two $n, k \in \mathbb{N}$,

if $\mathrm{set}(t_1[n]) = \mathrm{set}(t_2[k])$, $M(t_1[n]) \neq \uparrow$ and $M(t_2[k]) \neq \uparrow$ (i.e., they both have a natural number value),

then $M(t_1[n]) = L(t_2[k])$.

Theorem (Lange and Zeugmann 1996)

Set-drivenness does not restrict the power of finite identification.

Outline

Preset Learning

A preset learner is a learner that explicitly uses DFTT's to make its conjectures.

Definition

Let t be a text for some $L_i \in \mathcal{L}$, $f : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$.
For the preset learner *M based on f*, $M(t[n])$ is:

$$\begin{cases} \mu j\,(\mathrm{set}(t[n]) \subseteq L_j) & \text{if for that } j, \exists S \subseteq \mathrm{set}(t[n])(f(S,j)=1) \\ & \ \&\ \forall k < n\ L(t[k]) = \uparrow; \\ \uparrow & \text{otherwise.} \end{cases}$$

Proposition

Any finitely identifiable $\mathcal{L}$ is finitely identified by a preset learner.

Subset-drivenness

Definition

A learning function M is subset-driven w.r.t. a class $\mathcal{L}$ iff for any two texts t_1 and t_2 for some languages in $\mathcal{L}$, and any $n, k \in \mathbb{N}$:

If $M(t_1[n]) \downarrow$ (i.e., $M(t_1[n])$ gives a natural number value) and $\text{set}(t_1[n]) \subseteq \text{set}(t_2[k])$ and for all $\ell < k$, $M(t_2[\ell]) = \uparrow$,

then $M(t_1[n]) = L(t_2[k])$.

This means that, a learner is subset-driven, if whenever in some situation it gives a certain value to an input σ, and it sees that input as a subset of another input τ and has not given a value yet to an initial segment of τ, then it should give the same value to τ as to σ.

Subset-drivenness and Preset Learning

Theorem

Let f_{dftt} be a dftt-function for a class $\mathcal{L}$. If M is a preset learning function based on f_{dftt}, then M is subset-driven w.r.t. $\mathcal{L}$.

Theorem

Assume that $\mathcal{L}$ is finitely identified by a subset-driven learning function M. Then M is a preset learner (w.r.t. some f).

Outline

Eliminative Power

Eliminative power of x w.r.t. $\mathcal{L}$ is the set of possibilities in $\mathcal{L}$ that x excludes.

Definition

Take $\mathcal{L}$ and $x \in \bigcup \mathcal{L}$. The eliminative power of x w.r.t. $\mathcal{L}$ is given by a function $El_{\mathcal{L}} : \bigcup \mathcal{L} \to \mathcal{P}(\mathbb{N})$, s.t.:
$El_{\mathcal{L}}(x) = \{i \mid x \notin L_i \;\&\; L_i \in \mathcal{L}\}$. Additionally,
$El_{\mathcal{L}}(X) = \bigcup_{x \in X} El_{\mathcal{L}}(x)$.

Redefining DFTTs via eliminative power gives the following characterization.

Theorem

A class $\mathcal{L}$ is finitely identifiable from positive data iff there is an effective procedure $\mathcal{D} : \mathbb{N} \to \mathcal{P}^{<\omega}(\mathbb{N})$, such that $\mathcal{D}(i) \subseteq L_i$ and $El_{\mathcal{L}}(\mathcal{D}(i)) = \mathbb{N} - \{j \mid L_j = L_i\}$.

Eliminative Power and Complexity

For finite classes of finite sets, eliminative power allows studying computational complexity of finite identification.

Theorem

Checking a finite class of finite sets to be finitely identifiable is polynomial in the cardinality of the class and the maximal cardinality of its sets.

Two Kinds of Minimality of DFTT's I

Definition

A minimal DFTT of L_i in $\mathcal{L}$ is a $D_i \subseteq L_i$, such that

1. D_i is a DFTT for L_i in $\mathcal{L}$, and
2. $\forall X \subset D_i \; El_{\mathcal{L}}(X) \neq I_{\mathcal{L}} - \{j \mid L_j = L_i\}$.

Theorem

Let $\mathcal{L}$ be a finitely identifiable finite class of finite sets. Finding a minimal DFTT of $L_i \in \mathcal{L}$ can be done in polynomial time.

Two Kinds of Minimality of DFTT's II

Definition

A minimal-size DFTT of L_i in $\mathcal{L}$ is a minimal DFTT of smallest cardinality.

Definition (Minimal-size DFTT Problem)

Instance	A finite class of finite sets $\mathcal{L}$, a set $L_i \in \mathcal{L}$, and $k \in \mathbb{N}$ such that $k \leq \lvert L_i \rvert$.
Question	Is there a minimal DFTT $X_i \subseteq L_i$ of size $\leq k$?

Theorem

The Minimal-size DFTT *Problem is NP-complete.*

Outline

Restricting to 1-1 enumerations

For the remainder we restrict to indexed families $\mathcal{L}$ with enumerations without repetitions (if $L_i = L_j$, then $i = j$). This is hardly a real restriction, since any indexed $\mathcal{L}$ can be indexed in such a way.
The first part of the definition of finite identifiability then changes from

A learning function M:

1. finitely identifies $L_i \in \mathcal{L}$ on t iff, when inductively given t, at some point M gives a single output j such that $L_j = L_i$;

to

1. finitely identifies $L_i \in \mathcal{L}$ on t iff, when inductively given t, at some point M gives the single output i;

DEL

Knowledge update in DEL (Dynamic Epistemic Logic) also proceeds by way of the eliminative power we introduced. (For DEL the updates here are of a very simple kind.)

Instead of learner M we have an agent A. Learning agent A starts with the set $\{L_i \mid i \in \mathbb{N}\}$.

$$A(t[1]) = \{L_i \mid L_i \models t_0\}$$

$$A(t[n]) = \{L_i \in A(t[n-1]) \mid L_i \models t_n\}$$

The updates are obtained simply by using the eliminative power of the t_i.

DEL 2

There is an obvious way to define a learner M_A which applies more or less the DEL-method, namely simply take the first L_i that still satisfies the part of the text given. This learner always exists and is recursive. In general this learner may have the problem of overgeneralization. If the language chosen at any stage properly contains the one of the text, this method will fail in the limit, but of course there is no such problem in the present case where the class is finitely identifiable.

DEL 3

$$M_A(t[1]) = \mu i\,(|L_i \models t_0)$$

$$M_A(t[n]) = \mu i\,(L_i \models t_0, \ldots, t_{n-1})$$

Of course, $M_A(t[n]) = \mu i\,(L_i \in A(t[n]))$. This is a learner who identifies $\mathcal{L}$ in the limit. A finite learner to identify $\mathcal{L}$ finitely would be

$$M_A^F(t[n]) = \uparrow,$$

unless n is the least number such that $A(t[n]){=}\{i\}$ for some i, then $M_A^F(t[n]) = i$.

But is this learner M_A^F recursive?

Outline

Preliminaries

Preset Learning

Eliminative Power and Complexity

Knowledge Update

Fastest Learning

Fastest Learning

The fastest learner finitely identifies a language L_i as soon as any DFTT for it has been enumerated.
One can say that the procedure defined by DEL for update gives the fastest learner (by defining it as M_A^F).

Definition

$\mathcal{L}$ is finitely identifiable in the fastest way if and only if there is a learning function M such that, for each t for some $L \in \mathcal{L}$ and n, there exists i such that

$$\begin{aligned} M(t[n]) = i \quad \text{iff} \quad & \mathrm{set}(t[n]) \subseteq L_i \wedge \neg \exists j \neq i (\mathrm{set}(t[n]) \subseteq L_j) \\ & \wedge \exists j \neq i (\mathrm{set}(t[n-1]) \subseteq L_j). \end{aligned}$$

We will call such M a fastest learning function.

Characterization of Fastest Learning

Definition

The complete dftt-function for a class $\mathcal{L}$ is a recursive function $f_{\text{c-dftt}} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0, 1\}$, such that:

1. $f_{\text{c-dftt}}(S, i) = 1$ iff S is a DFTT of L_i;
2. for every $i \in \mathbb{N}$ there is a finite $S \subseteq \mathbb{N}$, s.t. $f_{\text{c-dftt}}(S, i) = 1$.

Characterization of Fastest Learning II

Theorem

A class $\mathcal{L}$ is finitely identifiable in the fastest way iff there exists a complete dftt-function for $\mathcal{L}$.

In fact, recognizing all minimal DFTTs is good enough.
If we define min-dftt-function by replacing DFTT by minimal DFTT we obtain:

Theorem

$\mathcal{L}$ is finitely identifiable in the fastest way iff $\mathcal{L}$ has a min-dftt-function.

Finite Identifiability and Fastest Learning

Not every finitely identifiable class is identified by a fastest learner.

Theorem

$\mathcal{L}$ exists that is finitely identifiable, but not in the fastest way.

A witness:

$$\mathcal{L} = \{L_i \mid i \in \mathbb{N}\} \text{ given by } L_i = \{2i, 2i{+}1\} \cup \{2j \mid Rji\} \cup \{2j{+}1 \mid Sji\},$$

where the r.e. predicates $\exists y Rxy$ and $\exists y Sxy$ are recursively inseparable.

Proof, part 1

The idea is that $L_i = \{2i, 2i+1\}$ except that, additionally, for some m, Rim or Sim may be true, and then $2i \in L_m$ or $2i+1 \in L_m$, respectively.

Note that:

- There can be at most one such m, and for that m only one of Rim or Sim can be true.
  (We assume that Riy can be made true for at most one m, same for Siy.)
- Since $A = \{x \mid \exists y Rxy\}$ and $B = \{x \mid \exists y Sxy\}$ are computably inseparable there is no computable f that makes the choice for each i.
- Except for such intruders the languages are disjoint.

Proof, part 2

The argument:

- $\{2i, 2i+1\}$ is a DFTT for L_i.
- But, $\{2i+1\}$ is a DFTT for S_i if $i \notin B$, and $\{2i\}$ is a DFTT for S_i if $i \notin A$.
- So, a computable function that would give the minimal DFTTs of L_i gives a computable separating set of A and B.
- And this is impossible, since A and B are computably inseparable.

So there cannot be a computable fastest learner.

FASTEST LEARNING AND LEARNING IN THE LIMIT

How can it be that in case the fastest learner is not recursive the above defined learner M_A is recursive whereas its corresponding finite learner M_A^F is not?

It must be that the code of the singleton $\{i\}$ cannot be given as its canonical code by M_A. The code given is a code of the set $\{i\}$ but cannot be deciphered as such. The learner M_A does not "know" yet that it has reached the final conclusion.

Strict Preset Learning

Learners may have at their direct disposal all minimal DFTTs of languages from the given class, especially in the case of classes of finite languages.

Definition

$\mathcal{L}$ is strict preset finitely identifiable iff there is a recursive F such that $F(i)$ outputs the set min-$\mathbb{D}_i$ of all minimal DFTTs of L_i.

Proposition

Finding min-$\mathbb{D}_i$ of $L_i \in \mathcal{L}$ is NP-hard.

Strict Preset Learning II

Theorem

If a class $\mathcal{L}$ is strict preset finitely identifiable, then $\mathcal{L}$ is learnable in the fastest way.

Theorem

A finitely identifiable $\mathcal{L}$ of finite sets exists for which there is no recursive F s.t. for each i, $F(i) =$min-$\mathbb{D}_i$.

A witness:

$$\mathcal{L} = \{L_i \mid i \in \mathbb{N}\} \text{ is given by } L_i = \{2i, 2(\mu y T(i,i,y)) + 1\},$$

where T is Kleene's T-predicate.

Strict Preset Learning and Fastest Learning

Corollary

There exist a class $\mathcal{L}$ that can be learnt in the fastest way but is not strict preset learnable.

Proposition

If a class $\mathcal{L}$ of finite languages is learnable in the fastest way and $\mathcal{L}$ is bounded by a recursive function, then $\mathcal{L}$ is strict preset learnable.

Of course, it is sufficient to bound the minimal DFTTs , even for infinite languages.

Thank you!