Integrating Abduction and Induction in the Learning from Interpretation Setting

Evelina Lamma[1], Paola Mello[2], **Fabrizio Riguzzi**[1]
[1]University of Ferrara, Italy
[2]University of Bologna, Italy

Summary

- Aim
- Preliminaries
- Learning from interpretations
- Running example
- ICL
- AICL
- Experiments
- Results
- Conclusions

Aim of the Work

- Integrating abduction and induction
- In learning from entailment: Progol 5.0 [Muggleton, Bryant 2000], SOLDR [Yamamoto 2000], CF-Induction [Inoue 2001], HAIL [Ray et al., 2003], ACL [Kakas, Riguzzi 2000], LAP [Lamma et al. 1999]
- In learning from interpretation: not investigated
- Aim: integration for learning from incomplete interpretations
- Comparison of approaches: ICL and ICL+abduction (AICL)

Notation

- $C = h_1 \vee h_2 \vee \ldots \vee h_n \leftarrow b_1, b_2, \ldots, b_m$
- $head(C) = h_1 \vee h_2 \vee \ldots \vee h_n$
- $body(C) = b_1 \wedge b_2 \wedge \ldots \wedge b_m$
- *Herbrand base* of a clausal theory P: $H_B(P)$
- *(Herbrand) interpretation* i: a subset of $H_B(P)$
- Herbrand Model of definite clause theory P: $M(P)$

Truth in an Interpretation

- A clause C *is true* in an interpretation i if for all grounding substitutions θ of C:
  $body(C)\theta \subset i \rightarrow head(C)\theta \cap i \neq \emptyset$
- A clausal theory T *is true* in i iff all clauses in T are true in i

Test of Truth of a Clause

- Clause C, finite interpretation i: run the query $?-body(C), not\ head(C)$ against a logic program containing i
- If the query succeeds, C is false in i. If the query finitely fails, C is true in i
- Clause C, interpretation $M(P \cup i)$: run the query $?-body(C), not\ head(C)$ against the logic program $P \cup i$.

Learning from Intepretations

Given

- a set P of interpretations
- a set N of interpretations
- a definite clause background theory B

Find: a clausal theory H such that

- for all $p \in P$, H is true in $M(B \cup p)$
- for all $n \in N$, H is false in $M(B \cup n)$

i.e. **Find**: a clausal theory H such that

- for all $p \in P$, for all $C \in H$, C is true in $M(B \cup p)$
- for all $n \in N$, there exists a $C \in H$ such that C is false in $M(B \cup n)$

Classifying an Unseen Interpretation

- Let i be an unseen interpretation to be classified
- Let H be $\{C_1, \ldots, C_n\}$
- for $j := 1$ to n
  - if C_j is false on $B \cup i$ then return $negative$
- return $positive$

Running Example

Two bit multiplexer:

- two input pins and four output pins
- the output pin whose number is represented by the input pins is at 1
- the other output pins may assume either 0 or 1

Example: 010110, working multiplexer
Aim: distinguishing a working multiplexer configuration from a faulty one

- 64 possible examples: 32 positive, 32 negative

Representation

- 12 nullary predicates
- For 010110 we have:

```
pin1at0.  pin2at1.  pin3at0.
pin4at1.  pin5at1.  pin6at0.
```

Correct theory

- A correct theory for distinguishing positive from negative configurations is

```
pin3at1:-pin1at0,pin2at0.
pin4at1:-pin1at0,pin2at1.
pin5at1:-pin1at1,pin2at0.
pin6at1:-pin1at1,pin2at1.
```

- All the clauses are true for the multiplexer 010110

```
pin1at0.  pin2at1.  pin3at0.
pin4at1.  pin5at1.  pin6at0.
```

- Test of 1st clause: query:
  `?- pin1at0,pin2at0,not pin3at1`
- Test of 2nd clause: query:
  `?- pin1at0,pin2at1,not pin4at1`

ICL Covering Algorithm

Learn(P, N, B)
$H := \emptyset$
repeat until best clause C not found or N is empty
    find best clause C
    if best clause C found then
        add C to H
        remove from N all interpretations that are false for C
return H

ICL Beam Search Algorithm

FindBestClause(P, N, B)
$Beam := \{false \leftarrow true\}$, $BestC := \emptyset$
while $Beam$ is not empty do
    $NewBeam := \emptyset$
    for each clause C in $Beam$ do
        for each refinement Ref of C do
            if Ref is better than $BestC$ and Ref
                is statistically significant then $BestC := Ref$
            if Ref is not to be pruned then
                add Ref to $NewBeam$
                if size of $NewBeam > MaxBeamSize$ then
                    remove worst clause from NewBeam
    $Beam := NewBeam$
return $BestClause$

ICL Heuristics

- $HV(C) = p(\ominus|\bar{C}) = \frac{n^{\ominus}(\bar{C})+1}{n(\bar{C})+2}$
- $LR(C) = 2n(C) \times \left(p(\oplus|C) \log \frac{p(\oplus|C)}{p_a(\oplus)} + p(\ominus|C) \log \frac{p(\ominus|C)}{p_a(\ominus)} \right)$
- $LR(C)$ is distributed approximately as χ^2 with one degree of freedom
- C is significant if $LR(C) > T$ with T a significance threshold (e.g. 6.64 for 99% significance)

ICL Pruning

A clause C is pruned if

- no refinements of it can become better than the best clause at the moment (the best refinement would be false for the same negative interpretations and true for all positive interpretations, $HV_{best}(C) = \frac{n^{\ominus}(\bar{C})+1}{n^{\ominus}(\bar{C})+2}$), or
- no refinements of it can become statistically significant

Abductive ICL

- Abduction is used in order to complete incomplete interpretations
- An abductive proof procedure in (partial) substitution of the Prolog procedure in the test of a clause

Coverage Test of Positive Interpretations

- p positive incomplete interpretation
- Clause to be tested: $h_1 \vee h_2 \vee \ldots \vee h_n \leftarrow b_1, b_2, \ldots, b_m$
- Test query: $? - b_1, b_2, \ldots, b_m, not\ h_1, not\ h_2, \ldots, not\ h_n$
- suppose h_i is false because p is incomplete
- By using an abductive proof procedure, we complete p so that h_i is true
- The abduction of facts can be performed only if the facts are consistent with the integrity constraints

Example

C=`pin3at1 :- pin1at0,pin2at0`
$p = \{$`pin1at0,pin2at0,pin4at1,pin5at1,pin6at0`$\}$
(OO?110)
$B = \{$`:- pin3at0,pin3at1`$\}$

Test query `?- pin1at0,pin2at0,not pin3at1`

The query fails by abducing `pint3at1` (compatible with the integrity constraints)

Coverage Test of Negative Interpretations

- n negative incomplete interpretation
- Test query: $? - b_1, b_2, \ldots, b_m, not\ h_1, not\ h_2, \ldots, not\ h_n$
- Test over interpretation n: suppose b_j is false because n is incomplete
- By using an abductive proof procedure, we complete n so that b_j is true
- The abduction of facts can be performed only if the facts are consistent with the integrity constraints

Example

C=`pin3at1 :- pin1at0,pin2at0`
$n = \{$`pin1at0,pin2at0,pin3at0,pin4at1,`
`pin5at1,pin6at0`$\}$ (000110)
$B = \{$`:- pin2at0,pin2at1.`$\}$

Test query `?- pin1at0,pin2at0,not pin3at1` succeeds

$n' = \{$`pin1at0,pin3at0,pin4at1,`
`pin5at1,pin6at0`$\}$ (0?0110)

The query fails, but

it succeeds by abducing `pint2at0` (compatible with the integrity constraints)

AICL

ICL is modified in two points:

- in function FindBestClause: test of the refinement so that the heuristic and the likelihood ratio can be computed
- in funtion Learn: addition of the facts abduced during the test to the corresponding interpretation

Test of C on a Positive Interpretation p

find the set Θ of all the answer substitutions for :-$body(C)$
$\Delta := \emptyset$
$covered := true$
while Θ is not empty and $covered$
    remove the first element θ from Θ
    $Head := head(C)\theta$, $found := false$
    while there are literals in $Head$ and $not\ found$
        remove the first literal L in $Head$
        if AbdDer$(L, p \cup B, \Delta)$ succeeds returning (θ, Δ_{out}) then
            $found := true$
            $\Delta := \Delta_{out}$
    if $found = false$ then
        $covered := false$

Example

C=`pin3at1 :- pin1at0,pin2at0`
$p = \{$`pin1at0,pin2at0, pin4at1,pin5at1,pin6at0`
$\}$ (OO?110)
$B = \{$`:- pin3at0,pin3at1`
`...` $\}$

$\Theta = \{\emptyset\}$, $covered = true$
$Head =$`pin3at1`, $found = false$
AbdDer(`pin3at1`,$p \cup B, \emptyset$) returns $(\emptyset, \{$`pin3at1`$\})$
$found = true$, $\Delta = \{$ `pin3at1`$\}$
$covered$ remains at $true$

Test of C on a Negative Interpretation n

find the set E of all the couples (θ, Δ) such that
    AbdDer$(body(C), n \cup B, \emptyset)$ succeeds returning (θ, Δ)
$covered := true$
while E is not empty and $covered$
    remove the first element (θ, Δ) from E
    $Head := head(C)\theta$
    call Der$((not\ Head), n \cup B \cup \Delta)$
    if the derivation succeeds then
        $covered := false$

Example

C=`pin3at1 :- pin1at0,pin2at0`
$n = \{$`pin1at0,pin3at0,`
`pin4at1,pin5at1,pin6at0`$\}$ (0?0110)
$B = \{$`:- pin1at0,pin1at1.`
`...` $\}$

AbdDer((`pin1at0,pin2at0`),$n \cup B, \emptyset$) returns $(\emptyset, \{$
`pin2at0`$\})$
$E = \{(\emptyset, \{$ `pin2at0`$\})\}$, $covered = true$
$Head =$`pin3at1`,
Der((`not pin3at1`),$n \cup B \cup \{$`pin2at0`$\}$) returns $\{\emptyset\}$
$covered = false$

Addition of Abduced Facts

- The function FindBestClause returns also the literals abduced for each interpretation during the test of the best clause
- The function Learn
  - adds the best clause to the current theory H
  - adds to each interpretation the facts abduced during the test of the coverage of the clause on that interpretation

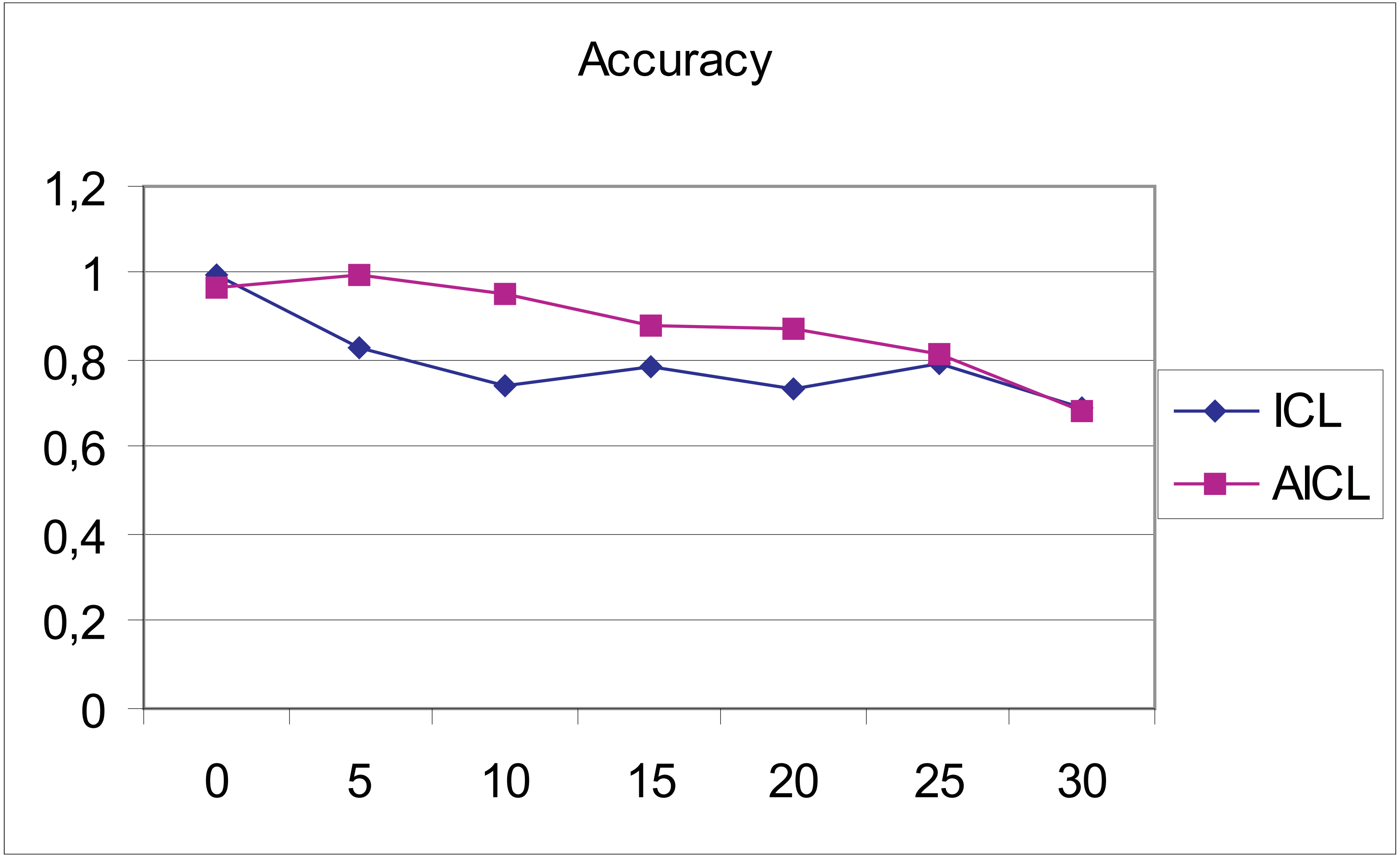
Implementation

- Kakas Mancarella abductive proof procedure
- Sicstus Prolog: use of the module system:
  - each interpretation is loaded into a different module
  - addition of abduced facts to interpretations by asserts in the module

Experiments

- Comparison with ICL
- Multiplexer dataset
- Ten folds
- For each fold, we have removed 5%, 10%, 15%, 20%, 25% and 30% of facts
- ICL background: empty
- AICL background: all the 12 predicates abducible, constraints on the predicates
- Learning settings: significance level = 0, defaults for the others
- For each level of incompleteness, compute average accuracy over the ten folds.

Results



Conclusions

- Integration of abduction and induction in learning from interpretations
- Aim: learning from incomplete interpretations
- Abductive proof procedure (partially) used in place of the Prolog proof procedure
- Abduction for covering positive interpretations and not covering negative ones
- Comparison of AICL with ICL on the multiplexer dataset: better performances up to 20%

Future Works

- Experiment with different uses of abduction
- More experiments on larger, non-propositional domains
- Learning the specification of protocols of interaction among agents from traces of their execution
- Use of proof procedures that handle non ground abducibles: IFF, SCIFF, A-system