Integrating Abduction and Induction in the Learning from Interpretation Setting

Evelina Lamma¹, Paola Mello², Fabrizio Riguzzi¹

¹University of Ferrara, Italy
²University 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

- \( C = h_1 \lor h_2 \lor \ldots \lor h_n \leftarrow b_1, b_2, \ldots, b_m \)
- \( \text{head}(C) = h_1 \lor h_2 \lor \ldots \lor h_n \)
- \( \text{body}(C) = b_1 \land b_2 \land \ldots \land b_m \)
- **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 \( P: M(P) \)
A clause $C$ is true in an interpretation $i$ if for all grounding substitutions $\theta$ of $C$:
\[
\text{body}(C)\theta \subseteq i \rightarrow \text{head}(C)\theta \cap i \neq \emptyset
\]

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 $\text{?} - \text{body}(C'), \text{not head}(C')$ 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 $\text{?} - \text{body}(C'), \text{not head}(C')$ against the logic program $P \cup i$. 
Learning from Interpretations

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 \( H = \{C_1, \ldots, C_n\} \)
- for \( j := 1 \) to \( n \)
  - if \( C_j \) is false on \( B \cup i \) then return \textit{negative}
- return \textit{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.
A correct theory for distinguishing positive from negative configurations is

\[
\begin{align*}
\text{pin3at1:} & \neg \text{pin1at0}, \text{pin2at0}. \\
\text{pin4at1:} & \neg \text{pin1at0}, \text{pin2at1}. \\
\text{pin5at1:} & \neg \text{pin1at1}, \text{pin2at0}. \\
\text{pin6at1:} & \neg \text{pin1at1}, \text{pin2at1}.
\end{align*}
\]

All the clauses are true for the multiplexer 010110

\[
\begin{align*}
\text{pin1at0. pin2at1. pin3at0. pin4at1. pin5at1. pin6at0.}
\end{align*}
\]

Test of 1st clause: query:

\[
?- \text{pin1at0, pin2at0, not pin3at1}
\]

Test of 2nd clause: query:

\[
?- \text{pin1at0, pin2at1, not pin4at1}
\]
**ICL Covering Algorithm**

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

- \( HV(C) = p(\ominus|\bar{C}) = \frac{n^{\ominus}(\bar{C}) + 1}{n(C) + 2} \)

- \( LR(C) = 2n(C) \times \left( p(\oplus|C) \log \frac{p(\oplus|C)}{p_a(\oplus)} + p(\ominus|C) \log \frac{p(\ominus|C)}{p_a(\ominus)} \right) \)

- \( LR(C) \) is distributed approximately as \( \chi^2 \) 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, $HV_{best}(C) = \frac{n^\oplus(\bar{C})+1}{n^\oplus(C)+2}$), 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: $h_1 \lor h_2 \lor \ldots \lor h_n \leftarrow b_1, b_2, \ldots, b_m$
- Test query: $? - b_1, b_2, \ldots, b_m, \text{not } h_1, \text{not } h_2, \ldots, \text{not } h_n$
- 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

\[ C = \text{pin3at1} : \neg \text{pin1at0}, \text{pin2at0} \]

\[ p = \{ \text{pin1at0}, \text{pin2at0}, \text{pin4at1}, \text{pin5at1}, \text{pin6at0} \} \]

(00?110)

\[ B = \{ :- \text{pin3at0}, \text{pin3at1} \} \]

Test query \( ?- \text{pin1at0}, \text{pin2at0}, \neg \text{pin3at1} \)

The query fails by abducing \text{pin3at1} (compatible with the integrity constraints)
Coverage Test of Negative Interpretations

- $n$ negative incomplete interpretation
- Test query: $? \leftarrow b_1, b_2, \ldots, b_m, \text{not } h_1, \text{not } h_2, \ldots, \text{not } h_n$
- 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

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

**Test query**

\[ ?- \text{pin1at0, pin2at0, not pin3at1 succeeds} \]

\[ n' = \{\text{pin1at0, pin3at0, pin4at1, pin5at1, pin6at0}\} (0?0110) \]

The query fails, but it succeeds by abducing \text{pin2at0} (compatible with the integrity constraints)
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 function Learn: addition of the facts abduced during the test to the corresponding interpretation
Test of $C$ on a Positive Interpretation $p$

find the set $\Theta$ of all the answer substitutions for $\text{-body}(C)$

$\Delta := \emptyset$

$\text{covered} := \text{true}$

while $\Theta$ is not empty and $\text{covered}$

remove the first element $\theta$ from $\Theta$

$\text{Head} := \text{head}(C)\theta, \text{found} := \text{false}$

while there are literals in $\text{Head}$ and not $\text{found}$

remove the first literal $L$ in $\text{Head}$

if $\text{AbdDer}(L, p \cup B, \Delta)$ succeeds returning $(\theta, \Delta_{out})$ then

$\text{found} := \text{true}$

$\Delta := \Delta_{out}$

if $\text{found} = \text{false}$ then

$\text{covered} := \text{false}$
Example

\[ C = \text{pin3at1} :- \text{pin1at0}, \text{pin2at0} \]

\( p = \{\text{pin1at0, pin2at0, pin4at1, pin5at1, pin6at0} \} \) (00?110)

\( B = \{\text{:- pin3at0, pin3at1} \)
\[ ... \} \]

\( \Theta = \{\emptyset\}, \text{covered} = \text{true} \)

\( \text{Head} = \text{pin3at1}, \text{found} = \text{false} \)

\( \text{AbdDer}(\text{pin3at1}, p \cup B, \emptyset) \) returns (\( \emptyset, \{\text{pin3at1}\} \))

\( \text{found} = \text{true}, \Delta = \{\text{pin3at1}\} \)

\( \text{covered remains at true} \)
find the set $E$ of all the couples $(\theta, \Delta)$ such that 
$\text{AbdDer}(\text{body}(C'), n \cup B, \emptyset)$ succeeds returning $(\theta, \Delta)$

$\text{covered} := \text{true}$

while $E$ is not empty and $\text{covered}$

remove the first element $(\theta, \Delta)$ from $E$

$\text{Head} := \text{head}(C')\theta$

call Der($\text{(not Head), n \cup B \cup \Delta}$)

if the derivation succeeds then

$\text{covered} := \text{false}$
Example

$C = \text{pin3at1} : - \text{pin1at0}, \text{pin2at0}$

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

$B = \{ : - \text{pin1at0}, \text{pin1at1}.
\quad \ldots \}$

$\text{AbdDer}((\text{pin1at0}, \text{pin2at0}), n \cup B, \emptyset) \text{ returns } (\emptyset, \{ \text{pin2at0}\})$

$E = \{(\emptyset, \{ \text{pin2at0}\})\}, \text{covered} = \text{true}$

$\text{Head} = \text{pin3at1},$

$\text{Der}((\text{not pin3at1}), n \cup B \cup \{\text{pin2at0}\}) \text{ returns } \{\emptyset\}$

$\text{covered} = \text{false}$
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

Accuracy

ICL
AICL
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