Inverse Entailment in Nonmonotonic Logic Programs

Chiaki Sakama

Wakayama University, Japan

Inverse Entailment

B : background theory (Horn program)

E : positive example (Horn clause)

$$B \wedge H \models E \Leftrightarrow B \models (H \rightarrow E)$$

$$\Leftrightarrow B \models (\neg E \rightarrow \neg H)$$

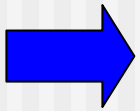
$$\Leftrightarrow B \wedge \neg E \models \neg H$$

A possible hypothesis H (Horn clause) is obtained by B and E.

Problems in Nonmonotonic Programs

1. Deduction theorem does not hold.
2. Contraposition is undefined.

The present IE technique cannot be used in nonmonotonic programs.



Reconstruction of the theory is necessary in nonmonotonic ILP.

Normal Logic Programs (NLP)

- An NLP is a set of rules of the form:

  $A_0 \leftarrow A_1, \ldots, A_m, \mathit{not}\ A_{m+1}, \ldots, \mathit{not}\ A_n$

  (A_i : atom, *not*: *Negation as failure*)
- A *Nested Rule* is a rule of the form:

  $A \leftarrow R$ where R is a rule
- An *LP-literal* is $L \in HB^+$ where

  $HB^+ = HB \cup \{\text{not } A \mid A \in HB\}$

Stable Model Semantics
[Gelfond& Lifschitz,88]

- An NLP may have none, one, or multiple stable models in general.
- An NLP having exactly one stable model is called *categorical*.
- An NLP is *consistent* if it has a stable model.
- A stable model coincides with the least model in Horn LPs.

Problems of Deduction Theorem in NML [Shoham87]

- In classical logic, $T \models F$ holds if F is satisfied in every model of T.
- In NML, $T \models_{NML} F$ holds if F is satisfied in every *preferred* model of T.

Proposition 3.1 *Let P be an NLP. For any rule R, $P \models R$ implies $P \not\models_s R$, but not vice-versa.*

Entailment Theorem (1)

Theorem 3.2 Let P be an NLP and R a rule s.t. $P \cup \{R\}$ is consistent. If $P \cup \{R\} \not\models_s A$, then $P \not\models_s A \leftarrow R$ for any ground atom A.

The converse does not hold in general.

ex) $P = \{ a \leftarrow \text{not } b \}$ has the stable model $\{a\}$, then $P \models_s a \leftarrow b$. $P \cup \{b\}$ has the stable model $\{b\}$, so $P \cup \{b\} \not\models_s a$.

Entailment Theorem (2)

Theorem 3.2 *Let P be an NLP and R a rule s.t. $P \cup \{R\}$ is consistent. If $P \not\models_s A \leftarrow R$ and $P \not\models_s R$, then $P \cup \{R\} \not\models_s A$ for any ground atom A.*

Contraposition in NLP

- In NLPs a rule is not a clause by the presence of NAF.
- A rule and its contraposition wrt $\leftarrow$ and *not* are semantically different.

ex) {a$\leftarrow$} has the stable model {a}, while {$\leftarrow$ not a} has no stable model.

Contrapositive Rules

Given a rule R of the form:

$A_0 \leftarrow A_1, \ldots, A_m, \mathit{not}\ A_{m+1}, \ldots, \mathit{not}\ A_n$

its *contrapositive rule* ${}^cR$ is defined as

$\mathit{not}\ A_1\ ;\ \ldots\ ;\ \mathit{not}\ A_m\ ;$

$\mathit{not\ not}\ A_{m+1}\ ;\ \ldots\ ;\ \mathit{not\ not}\ A_n \leftarrow \mathit{not}\ A_0$

where ; is disjunction and

not not A is *nested NAF.*

Properties of Nested NAF
[Lifschitz et al, 99]

$not\ A_1\ ;\ \ldots\ ;\ not\ A_m\ ;$

$not\ not\ A_{m+1}\ ;\ \ldots\ ;\ not\ not\ A_n \leftarrow\ not\ A_0$

is semantically equivalent to

$\leftarrow A_1, \ldots, A_m, not\ A_{m+1}, \ldots, not\ A_n\ , not\ A_0$

In particular,

$not\ A \leftarrow$ is equivalent to $\leftarrow A$

$not\ not\ A \leftarrow$ is equivalent to $\leftarrow not\ A$

Contrapositive Theorem

Theorem 4.1

Let P be an NLP, R a (nested) rule, and ${}^{c}R$ its contrapositive rule. Then, $P \not\models_s R$ iff $P \not\models_s {}^{c}R$

Inverse Entailment in NLP

Theorem 5.3 *Let P be an NLP and R a rule s.t. $P \cup \{R\}$ is consistent. For any ground LP-literal L, if $P \cup \{R\} \not\models_s L$ and $P \not\models_s$ not L , then $P \cup \{$not L$\} \not\models_s$ not R where $P \cup \{$not L$\}$ is consistent.*

Induction Problem

Given:

- a background KB as a categorical NLP
- a ground LP-literal L s.t. P $|=_s$ not L,

  L=A represents a positive example;
  L=not A represents a negative example
- a target predicate to be learned

Induction Problem

Find:

a hypothetical rule R s.t.

- $P \cup \{R\} \models_s L$
- $P \cup \{R\}$ is consistent
- R has the target predicate in its head

Computing Hypotheses

When $P \cup \{R\} \models_s L$ and $P \models_s$not L ,

$P \cup \{\text{not } L\} \models_s \text{not } R$ (by Th5.3)

By $P \models_s$not L, it is simplified as

$P \models_s \text{not } R$

Put $M^+ = \{ \ell \mid \ell \in HB^+ \text{ and } P \models_s \ell \}$.Then,

$M^+ \models \text{not } R$

Let $r_0 = \leftarrow \Gamma$ where $\Gamma \subseteq M^+$. Then,

$M^+ \models \text{not } r_0$

Relevance

Given two ground LP-literals L_1 and L_2, define $L_1 \sim L_2$ if L_1 and L_2 have the same predicate and contain the same constants.

L_1 in a ground rule R is *relevant* to L_2 if (i) $L_1 \sim L_2$ or (ii) L_1 shares a constant with L_3 in R s.t. L_3 is relevant to L_2. Otherwise, L_1 is *irrelevant* to L_2.

Transformations

1. When $r_0 = \leftarrow \Gamma_0$ contains any LP-literal irrelevant to the example, drop them and put $r_1 = \leftarrow \Gamma_1$
2. When $r_1 = \leftarrow \Gamma_1$ contains both an atom A and a conjunction B s.t. $A \leftarrow B \in P$, put $r_2 = \leftarrow \Gamma_1 - \{A\}$.
3. When $r_2 = \leftarrow \Gamma_2$ contains not A with the target pred., put $r_3 = A \leftarrow \Gamma_2 - \{not\ A\}$
4. Construct r_4 s.t. $r_4\theta = r_3$.

Inverse Entailment Rule

Theorem 5.5 When R_{ie} is a rule obtained by the transformation sequence, $M^+ \not\models$ not R_{ie} holds.

When $P \cup R_{ie}$ is consistent, we say that R_{ie} is computed by the *inverse entailment rule.*

Example(1)

P: bird(x)← penguin(x),
bird(tweety)←, penguin(polly)←.

L: flies(tweety).

target: flies

M^+ = {bird(t),bird(p),penguin(p), not penguin(t),not flies(t),not flies(p)}

Example(1)

r_0: $\leftarrow$ *bird(t),bird(p),penguin(p),*
not penguin(t),not flies(t),not flies(p).

Dropping irrelevant LP-literals:

r_1: $\leftarrow$ *bird(t),not penguin(t),not flies(t).*

Shifting the target to the head:

r_3: *flies(t)* $\leftarrow$ *bird(t),not penguin(t).*

Replacing tweety by x:

r_4: *flies(x)* $\leftarrow$ *bird(x),not penguin(x).*

Example(2)

P: flies(x)← bird(x), not ab(x),

bird(x)← penguin(x),

bird(tweety)←, penguin(polly)←.

L: not flies(polly).

target: ab

M^+ = {bird(t),bird(p),penguin(p),
not penguin(t),flies(t),flies(p),
not ab(t),not ab(p)}.

Example(2)

r_0: $\leftarrow$ *bird(t),bird(p),penguin(p),*
not penguin(t),not ab(t),not ab(p).

Dropping irrelevant LP-literals:

r_1: $\leftarrow$ *bird(p),penguin(p),not ab(p).*

Dropping implied atoms in P:

r_2: $\leftarrow$ *penguin(p),not ab(p).*

Shifting the target and generalize:

r_4: *ab(x)* $\leftarrow$ *penguin(x).*

Properties of the IE rule

- It is *not correct* for the computation of rules satisfying $P \cup \{R\} \models_s L$.

  $\rightarrow$ The meaning of a rule changes by its representation form.

- It is *not complete* for the computation of rules satisfying $P \cup \{R\} \models_s L$.

  $\rightarrow$ The transformation sequence filters out useless hypotheses.

Correctness Issue

$$a \leftarrow b,\ c = \neg b \leftarrow \neg a,\ c$$
$$= \neg c \leftarrow \neg a,\ b$$
$$= \leftarrow \neg a,\ b,\ c$$

★ *Independence of its written form*

$$a \leftarrow b,\ c \neq \text{not } b \leftarrow \text{not } a,\ c$$
$$\neq \text{not } c \leftarrow \text{not } a,\ b$$
$$\neq \leftarrow \text{not } a,\ b,\ c$$

★ *Dependence on its written form*

Completeness Issue

B: $q(a) \leftarrow$, $r(b) \leftarrow$

$r(f(x)) \leftarrow r(x)$

E: $p(a)$

H: $p(x) \leftarrow q(x)$

$p(x) \leftarrow q(x), r(b)$

$p(x) \leftarrow q(x), r(f(b))$, ...

! *Infinite number of hypotheses exist in general (and many of them are useless)*

Summary

- New inverse entailment rule in nonmonotonic ILP is introduced.
- The effects in ILP problems are demonstrated by some examples.

Future research includes:

- Extension to non-categorical programs (having multiple stable models).
- Exploiting practical applications.