Nonmonotonic Inductive Logic Programming (NMILP)

Chongbing Liu

October 24, 2005
Outlines

■ Why NMILP?

■ SLDNF Based Approaches

■ Moving from ILP to NMILP

■ Stable Models Based Approaches
Why NMILP?
Nonmonotonic Logic Programming (NMLP)

• normal logic programs (CAW by NAF)
  \[ A_0 \leftarrow A_1, \ldots, A_m, \neg A_{m+1}, \ldots, \neg A_n \]

• mainly, stable model semantics (beliefs)

• default reasoning on incomplete knowledge (defaults + observation \( \Rightarrow \) conclusion)

• rules acts as contraints or derivation rules (not as definitions)

• nonmonotonicity (addition of new info may contradict previous conclusions)

• No learning mechanisms are provided
Inductive Logic Programming (ILP)

- Given:
  - Background Knowledge $B$ and
  - Examples $E = E^+ \cup E^-$ ($B \not\models E$)

Find a theory $H$ such that
- $B \cup H \models e$ for every $e \in E^+$
- $B \cup H \not\models f$ for every $f \in E^-$

- Present ILP uses Horn clauses for $B$ and $H$
  - less expressive language
  - monotonic reasoning

- armed with various learning mechanisms
  - incremental learning (non-monotonic learning)
  - batch learning (monotonic learning)
  - top-down search and bottom-up search
  - inverse resolution
  - inverse entailment
Nonmonotonic Inductive Logic Programming (NMILP)

NMLP: expressive language, human commonsense reasoning, but no learning mechanisms

ILP: language with limited expressiveness, armed with learning mechanisms, but does not simulate human commonsense reasoning

NMILP: hopefully takes advantages of both paradigms

\[ \text{NMILP} = \text{NMLP} + \text{ILP} \]
SLDNF Based Approaches
Representive Efforts

- Non-monotonic learning, M. Bain, S. Muggleton, 1992

- Learning Logic Programs with negation as failure, 1996

- Learning nonmonotonic logic program: learning exceptions, 1995

- Normal programs and multiple predicate learning, 1998

- Learning extended logic programs, 1997

- A three-valued framework for the induction of general programs, 1996
Incremental Learning

Initialize $\Sigma$ to $\{\Box\}$

repeat

read the next (positive or negative) example

while $\Sigma$ is not correct w.r.t. the examples read so far

if $\exists e^-$ s.t. $\Sigma \models e^-$

specialize $\Sigma$ by identifying a false clause

and delete it from $\Sigma$

if $\exists e^+$ s.t. $\Sigma \not\models e^+$

generalize $\Sigma$ by constructing a clause $C \models e$

and add it to $\Sigma$

until no examples left to read.

=================================

An Example

$B = \{\text{bird(swan). bird(eagle). bird(penguin). bird(pigeon).}\}$

$E = \{\oplus \text{flies(swan).} \oplus \text{flies(eagle).} \ominus \text{flies(penguin).}\}$

$\Sigma_0 = \Box$

$\oplus: \text{flies(swan)}$

$\Sigma_1 = \{\text{flies}(X) \leftarrow \text{bird}(X).\}$

$\oplus: \text{flies(eagle)}$

$\Sigma_2 = \{\text{flies}(X) \leftarrow \text{bird}(X).\}$

$\ominus: \text{flies(penguin)}$

$\Sigma_3 = \{\text{flies(swan). flies(eagle).}\}$
An Example

\[ B = \{ \text{bird(swan). bird(eagle). bird(penguin). bird(pigeon).} \} \]

\[ E = \{ \oplus \text{flies(swan).} \oplus \text{flies(eagle).} \ominus \text{flies(penguin).} \} \]

\[ \Sigma_0 = \Box \]

\[ \oplus: \text{flies(swan)} \]

\[ \Sigma_1 = \{ \text{flies}(X) \leftarrow \text{bird}(X). \} \]

\[ \oplus: \text{flies(eagle)} \]

\[ \Sigma_2 = \{ \text{flies}(X) \leftarrow \text{bird}(X). \} \]

\[ \ominus: \text{flies(penguin)} \]

\[ \Sigma_3 = \{ \text{flies(swan). flies(eagle).} \} \]

-------------------------------

comments

1. monotonic reasoning (Horn clauses based)

2. non-monotonic learning (correct info not preserved, e.g., both \( \Sigma_1 \) and \( \Sigma_2 \) imply \( \text{flies(pigeon)} \), but \( \Sigma_3 \) does not.)

3. may result in poor learning quality

4. due to problem of “overly(drastic)-specialization”

5. we desire to preserve correct info

6. can not be achieved by any forms of “incremental-specialization” within classical logic framework

7. SOLUTION: introducing negation !
Closed World Specialization

Input:

set of clauses $T$ (possibly with negation) and ground atom $A$ s.t. $T \models A$ and $A$ is incorrect

Operations:

Generate proof of $T \models A$ using SLDNF-resolution
Assume $C \in T$ resolved with $\leftarrow A$
Let $C = \text{Hd} : \neg \text{Bd}$
Let $\theta$ be the substitution for variables in $C$
If literal $\neg B \in \text{Bd}$
    Let $T' = T \cup \{ B\theta \}$
else
    Let $\{ V_1, \ldots, V_n \}$ be the domain of $\theta$
    Let $q$ be a predicate symbol not found in $T$
    Let $B = q(V_1, \ldots, V_n)$
    Let $T' = T - \{ C \} \cup \{ \text{Hd} : \neg (B \cup \neg B) \} \cup \{ B\theta \}$

Output: $T'$

Note: $T'$ specializes $T$, but not in traditional sense, since $T'$ has a new predicate symbol.

In our example, the following theory will be learned

$\{ \text{flies}(X) \leftarrow \text{bird}(X), \neg \text{flightless}(X). \text{flightless}(\text{penguin}). \}$

Now since $\text{flightless(\text{penguin})}$ is true, $\neg \text{flightless(\text{penguin})}$ is false. Therefore, the newly learned theory does not derive $\text{flies(\text{penguin})}$ any more.
Moving from ILP to NMILP
Inverse Resolution is not directly applicable in NMILP!

**Inverse resolution:** (absorption)

\[
\begin{align*}
C_1 &: q \leftarrow A \\
C_2 &: p \leftarrow q, B \\
C_3 &: p \leftarrow A, B
\end{align*}
\]

- \( \Sigma_1 \) generalizes \( \Sigma_2 \) if \( \Sigma_2 \models a \) implies \( \Sigma_1 \models a \)
- Denote \( \Sigma = \{C_1, C_3\}, A(\Sigma) = \{C_1, C_2\} \).
- \( A(\Sigma) \) generalizes \( \Sigma \) in Horn clausal logic

In NMLP, however

- \( A(\Sigma) \) does not necessarily generalizes \( \Sigma \)
  \( \Sigma = \{p \leftarrow \neg q, q \leftarrow r, s \leftarrow r, s \leftarrow\} \) (V:3,2,2)
  \( A(\Sigma) = \{p \leftarrow \neg q, q \leftarrow s, s \leftarrow r, s \leftarrow\} \)
  Then, \( \Sigma \models p \) but \( A(\Sigma) \models\not p \).

- It may be the case that \( \Sigma \) is consistent, but \( A(\Sigma) \) is not.
  \( \Sigma = \{p \leftarrow q, \neg p, q \leftarrow r, s \leftarrow r, s \leftarrow\} \) (V:3,2,2)
  \( A(\Sigma) = \{p \leftarrow q, \neg q, q \leftarrow s, s \leftarrow r, s \leftarrow\} \)
  Then, \( \Sigma \) is consistent, but \( A(\Sigma) \) is not.

- ...
Inverse Entailment is not directly applicable to NMLP!

Deduction Theorem (Horn clausal logic)

For any formula $A$, we have

$$P \cup \{R\} \models A \iff P \models R \to A$$

**Inverse entailment:**

Given Horn program $B$ and an example $E$, deduction theorem gives:

$$B \cup \{H\} \models E \iff B \models (H \to E) \iff B \models (\neg E \to \neg H) \iff B \cup \{\neg E\} \models \neg H$$

$B \land \neg E \models \neg H$ serves as a necessary condition for constructing $H$.

In NMLP, however

- Deduction theorem in Eq. (1) and (3) does not hold in general
- Contrapositive implication in Eq. (2) is undefined
Stable Model Based Approaches
Main Results (by Chiaka Sakama)

Deduction Theorem (Horn clausal logic)
For any formula \( A \), we have

\[
P \cup \{ R \} \models A \iff P \models R \rightarrow A
\]

Entailment Theorem (NMLP)
For any ground literal \( A \), we have

\[
P \cup \{ R \} \models S A \implies P \models S R \rightarrow A \quad \text{(i)}
\]

\[
P \cup \{ R \} \models S A \iff P \models S R \rightarrow A \text{ and } P \models S R \quad \text{(ii)}
\]

Contrapositive rule in NMLP

\[
R : A_0 \leftarrow A_1, \ldots, A_m, \neg A_{m+1}, \ldots, \neg A_n
\]

\[
R^c : \neg A_1; \ldots, \neg A_m; \neg \neg A_{m+1}, \ldots, \neg \neg A_n \leftarrow \neg A_0
\]

\[
R^c : \leftarrow A_1, \ldots, A_m, \neg A_{m+1}, \ldots, \neg A_n, \neg A_0
\]

We can prove that \( P \models S R \iff P \models S R_C \quad \text{(iii)} \)

Inverse Entailment in NMLP

Given normal program \( B \) and a positive example \( E \) such that

\[
B \models S \neg E \quad \text{(iv)}
\]

Then

\[
B \cup \{ H \} \models S E \iff \text{by (i) } B \models S (H \rightarrow E)
\]

\[
\iff \text{by (iii) } B \models S (\neg E \rightarrow \neg H)
\]

consider \( H = p(x_1, \ldots, x_k) \) where \( p \) is a new atom

\[
\text{by (ii) and (iv) } B \cup \{ \neg E \} \models S \neg H
\]

So \( B \cup \{ \neg E \} \models S \neg H \) serves as a necessary condition for \( H \).

This necessary condition can be simplified as \( B \models S \neg H \).
Learning from a single positive example

Classical Inverse Entailment (IE):

- necessary condition for $H$: $B \land \neg E \models \neg H$ \hspace{1cm} (*)

- let $Bot$ be the conjunction of ground literals which are true in every model of $B \land \neg E$.

- we consider $Bot \models \neg H$ (but note: this IE is not complete since condition (*) does not imply $Bot \models \neg H$).

- $H_0 = \neg Bot$ is a trivial valid (ground) candidate of $H$.

- organize $H_0$ s.t. target predicate atom $A$ is left to “←”.

- generalizing $H_0$ by replacing constants with variables, we get a most specific hypothesis with variables.

NMLP Inverse Entailment (NMLP IE):

- necessary condition for $H$: $B \models_s \neg not H$ \hspace{1cm} (**) (same as $B \cup \{not\ E\} \models_s \neg not H$)

- let $M^+ = M \cup \{not\ l \mid l \notin M \text{ and } l \in \mathcal{HB}\}$, where $M$ is the stable model of $B$ and $\mathcal{HB}$ is the Herbrand model of $B$.

- condition (**) implies $M^+ \models \neg not H$.

- let $\Gamma = \{K \in M^+ \mid K \text{ is relevant to } L \text{ and is involved in } B \cup \{E\}\}$.

- since $M^+ \models \Gamma$, we have $M^+ \models not\ r_0$ where $r_0 = \leftarrow \Gamma$.

- integrity constraint $r_0 = \leftarrow \Gamma$ is a trivial valid (ground) candidate of $H$.

- shift the target predicate atom to the left of “←” in $r_0$.

- generalizing $r_0$ by replacing constants with variables, we get a most specific hypothesis with variables.
Illustration

Given:
\[ \mathcal{B} = \{ \text{bird}(X) \leftarrow \text{penguin}(X). \text{bird}(\text{tweety}). \text{penguin}(\text{polly}). \} \]
\[ \mathcal{E} = \{ \oplus \text{flies}(\text{tweety}). \} \]
Note: \( \mathcal{B} \models \text{not flies}(\text{tweety}) \).

Steps:

stable model of \( \mathcal{B} \) (same as that of \( \mathcal{B} \mid \mathcal{E} \)):
\[ \mathcal{M} = \{ \text{bird}(\text{tweety}). \text{bird}(\text{polly}). \text{penguin}(\text{polly}). \} \]

expansion of \( \mathcal{M} \):
\[ \mathcal{M}^+ = \{ \text{bird}(\text{tweety}). \text{bird}(\text{polly}). \text{penguin}(\text{polly}). \text{not penguin}(\text{tweety}). \text{not flies}(\text{tweety}). \text{not flies}(\text{polly}). \} \]

integrity constraint:
\[ r_0 = \leftarrow \text{bird}(\text{tweety}), \text{not penguin}(\text{tweety}), \text{not flies}(\text{tweety}). \]

shift the atom with target predicate to the left side:
\[ r_0 = \text{flies}(\text{tweety}) \leftarrow \text{bird}(\text{tweety}), \text{not penguin}(\text{tweety}). \]

generalize \( r_0 \) by replacing constant with variables:
\[ H = \text{flies}(X) \leftarrow \text{bird}(X), \text{not penguin}(X). \]