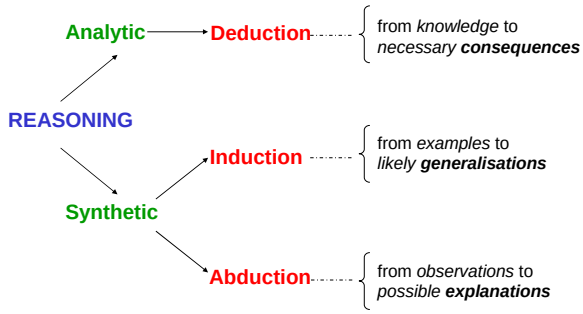
Towards the integration of abduction and induction in artificial neural networks

Oliver Ray and Artur Garcez

NeSy'06 (August 29th, 2006)

Intelligent Reasoning [Prc1900]



Neural Network Abduction

- Existing Work
  - **Diagnostic Problem Solving:** inferring diagnoses (faults) from manifestations (symptoms) with Hopfield [GR96] and competition-based [RPT93] neural networks
  - **Cost-Based Abduction:** specify cost of assuming an abducible with higher-order recurrent networks [AEA05]
- This Work
  - **aim:** to provide a massively parallel abductive method with no representational restrictions and the ability to handle multiple solutions; and allow the network to be revised by standard connectionist learning methods
  - **approach:** generalise neuro-symbolic approaches from logic programs to abductive logic programs…

Abductive Logic Programming [KKT92]

- Given

T	Theory – set of normal clauses
G	Goal – set of literals
IC	Integrity Constraints – set of negative clauses
A	Abducibles – set of ground atoms

- Find

$\Delta \subseteq A$	Hypothesis – set of Horn clauses

- Such that

$T \cup \Delta \models \exists G$	i.e. explanation
$T \cup \Delta \cup IC \not\models \bot$	i.e. consistent

Example

$$T = \left\{ \begin{array}{l} wont\_start \leftarrow battery\_flat \\ wont\_start \leftarrow fuel\_empty \\ battery\_flat \leftarrow wet\_day \\ overheat \leftarrow fan\_broke \\ lights\_on \end{array} \right\}$$

$$G = \{ wont\_start \}$$

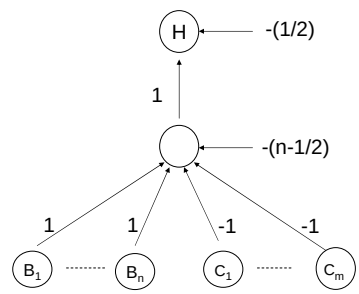
$$IC = \{ \bot \leftarrow battery\_flat, lights\_on \}$$

$$A = \{ fan\_broke, fuel\_empty, wet\_day \}$$

$$\Delta_1 = \{fuel\_empty\}$$
$$\Delta_2 = \{fuel\_empty, fan\_broke\}$$

Neuro-Symbolic Translation

$$H \leftarrow B_1, \ldots, B_n, \neg C_1, \ldots, \neg C_n$$



Representation of Abductive Context

$$T' = \left\{ \begin{array}{l} wont\_start \leftarrow battery\_flat \\ wont\_start \leftarrow fuel\_empty \\ battery\_flat \leftarrow wet\_day \\ overheat \leftarrow fan\_broke \\ lights\_on \end{array} \right\}$$

$$G' = \{goal \leftarrow wont\_start\}$$

$$IC' = \{ic \leftarrow battery\_flat, lights\_on\}$$

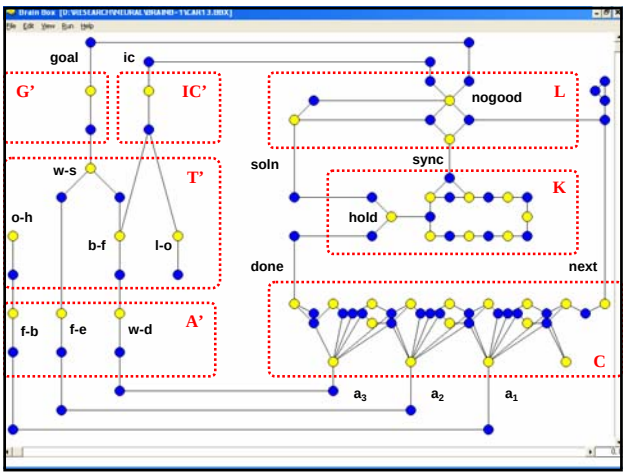
$$A' = \left\{ \begin{array}{l} fan\_broke \leftarrow a_1 \\ fuel\_empty \leftarrow a_2 \\ wet\_day \leftarrow a_3 \end{array} \right\}$$

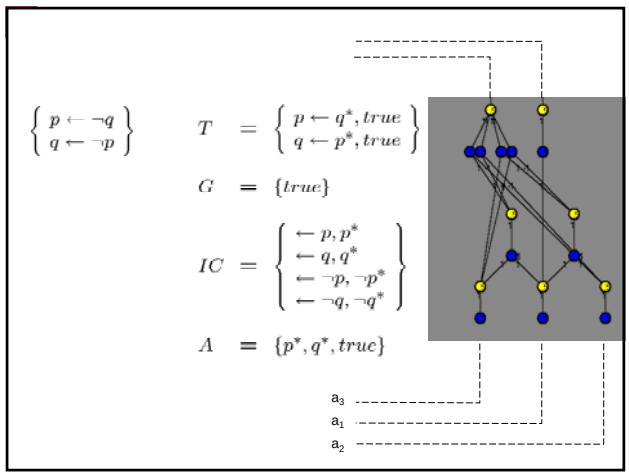
Extension with Abductive Machinery

$$Counter(C) = \bigcup_{i=1}^{3} \left\{ \begin{array}{l} a_i \leftarrow a_i, \neg c_i \\ a_i \leftarrow d_i \\ b_i \leftarrow a_i \\ c_i \leftarrow b_{i-1}, \neg a_{i-1}, a_i \\ d_i \leftarrow b_{i-1}, \neg a_{i-1}, \neg a_i \end{array} \right\} \cup \left\{ \begin{array}{l} b_0 \leftarrow next \\ done \leftarrow b_3, \neg a_3 \end{array} \right\}$$

$$clock(K) = \bigcup_{i=1}^{6} \left\{ k_i \leftarrow k_{i-1} \right\} \cup \left\{ \begin{array}{l} k_0 \leftarrow \neg hold, \neg k_M \\ sync \leftarrow k_0, \neg k_1 \end{array} \right\}$$

$$logic(L) = \left\{ \begin{array}{l} nogood \leftarrow ic \\ nogood \leftarrow \neg goal \\ soln \leftarrow sync, \neg nogood \\ soln \leftarrow soln, \neg nogood \\ hold \leftarrow soln \\ hold \leftarrow done \\ next \leftarrow sync, nogood \end{array} \right\}$$





Conclusion

- We have presented a neural network method for abduction which
  - generalises existing translations from logic programs to **abductive logic programs**
  - provides a neuro-symbolic method for (i) handling **non-acceptable** programs, (ii) answering **queries**, and (iii) expressing **integrity constraints**
- But variables must be grounded and it remains to see how this method can be integrated with neural network learning techniques