Introduction & Motivation: Overview

► Introduction & Motivation
► Propositional Logic
  ▶ Existing Approaches
  ▶ Propositional Logic Programs and the Core Method
► First-Order Logic
  ▶ Existing Approaches
  ▶ First-Order Logic Programs and the Core Method
► The Neural-Symbolic Learning Cycle
► Challenge Problems
Introduction & Motivation: Connectionist Systems

► Well-suited to learn, to adapt to new environments, to degrade gracefully etc.
► Many successful applications.
► Approximate functions.
  ▶ Hardly any knowledge about the functions is needed.
  ▶ Trained using incomplete data.
► Declarative semantics is not available.
► Recursive networks are hardly understood.
► McCarthy 1988: We still observe a propositional fixation.
► Structured objects are difficult to represent.
  ▶ Smolensky 1987: Can we instantiate the power of symbolic computation within fully connectionist systems?
Introduction & Motivation: Logic Systems

- Well-suited to represent and reason about structured objects and structure-sensitive processes.
- Many successful applications.
- Direct implementation of relations and functions.
- Explicit expert knowledge is required.
- Highly recursive structures.
- Well understood declarative semantics.
- Logic systems are brittle.
- Expert knowledge may not be available.

▷ Can we instantiate the power of connectionist computation within a logic system?
Introduction & Motivation: Objective

► Seek the best of both paradigms!
► Understanding the relation between connectionist and logic systems.
► Contribute to the open research problems of both areas.
► Well developed for propositional case.
► Hard problem: going beyond.
► In this lecture:
  ► Overview on existing approaches.
  ► Logic programs and recurrent networks.
  ► Semantic operators for logic programs can be computed by connectionist systems.

Introduction to the core method.
Connectionist Networks

- A connectionist network consists of
  - a set $U$ of units and
  - a set $W \subseteq U \times U$ of connections.

- Each connection is labeled by a weight $w \in \mathbb{R}$.
- If there is a connection from unit $u_j$ to $u_k$, then $w_{kj}$ is its associated weight.
- A unit is specified by
  - an input vector $\vec{i} = (i_1, \ldots, i_m)$, $i_j \in \mathbb{R}, 1 \leq j \leq m$,
  - an activation function $\Phi$ mapping $\vec{i}$ to a potential $p \in \mathbb{R}$,
  - an output function $\Psi$ mapping $p$ to an (output) value $v \in \mathbb{R}$.

- If there is a connection from $u_j$ to $u_k$ then $w_{kj}v_j$ is the input received by $u_k$ along this connection.
- The potential and value of a unit are synchronously recomputed (or updated).
- Often a linear time $t$ is added as parameter to input, potential and value.
- The state of a network with units $u_1, \ldots, u_n$ at time $t$ is $(v_1(t), \ldots, v_n(t))$. 
Propositional Logic

▶ Existing Approaches

▷ Finite Automata and McCulloch-Pitts Networks
▷ Weighted Automata and Semiring Artificial Neural Networks
▷ Propositional Reasoning and Symmetric/Stochastic Networks
▷ Other Approaches

▶ Propositional Logic Programs and the Core Method

▷ The Very Idea
▷ Logic Programs
▷ Propositional Core Method
▷ Backpropagation
▷ Knowledge-Based Artificial Neural Networks
▷ Propositional Core Method using Sigmoidal Units
▷ Further Extensions
Finite Automata and McCulloch-Pitts Networks

- **McCulloch, Pitts 1943:**
  Can the activities of nervous systems be modelled by a logical calculus?

- A **McCulloch-Pitts network** consists of a set $U$ of binary threshold units and a set $W \subseteq U \times U$ of weighted connections.

- The set $U_I$ of **input units** is defined as $U_I = \{ u_k \in U \mid (\forall u_j \in U) w_{kj} = 0 \}$.

- The set $U_O$ of **output units** is defined as $U_O = \{ u_j \in U \mid (\forall u_k \in U) w_{kj} = 0 \}$.

- **Theorem** McCulloch-Pitts networks are finite automata and vice versa.
Binary Threshold Units

- $u_k$ is a binary threshold unit if

$$
\Phi(\vec{v}_k) = p_k = \sum_{j=1}^{m} w_{kj} v_j
$$

$$
\Psi(p_k) = v_k = \begin{cases} 
1 & \text{if } p_k \geq \theta_k \\
0 & \text{otherwise}
\end{cases}
$$

where $\theta_k \in \mathbb{R}$ is a threshold.

- Three binary threshold units:

1. $v_1 \xleftarrow{w_{21}} -1 \quad \theta_2 = -0.5 \quad v_2 = \neg v_1$

2. $v_1 \xleftarrow{w_{31}} 1 \quad \theta_3 = 0.5 \quad v_3 = v_1 \lor v_2$

3. $v_1 \xleftarrow{w_{31}} 1 \quad \theta_3 = 1.5 \quad v_3 = v_1 \land v_2$
Weighted Automata and Semiring Artificial Neural Networks

- **Bader, Hölldobler, Scalzitti 2004:**
  Can the result by McCulloch and Pitts be extended to weighted automata?

- **Let** \((K, \oplus, \odot, 0_K, 1_K)\) **be a semiring.**

- **\(u_k\) is a \(\oplus\)-unit if**
  \[
  \Phi(\vec{i}_k) = p_k = \bigoplus_{j=1}^{m} w_{kj} \odot v_j \\
  \Psi(p_k) = v_k = p_k
  \]

- **\(u_k\) is a \(\odot\)-unit if**
  \[
  \Phi(\vec{i}_k) = p_k = \bigodot_{j=1}^{m} w_{kj} \odot v_j \\
  \Psi(p_k) = v_k = p_k
  \]

- **A semiring artificial neural network consists of a set \(U\) of \(\oplus\)- and \(\odot\)-units and a set \(W \subseteq U \times U\) of \(K\)-weighted connections.**

- **Theorem** Weighted automata are semiring artificial neural networks.
Symmetric Networks

- **Hopfield 1982**: Can statistical models for magnetic materials explain the behavior of certain classes of networks?

- **A symmetric network** consists of a set $U$ of binary threshold units and a set $W \subseteq U \times U$ of weighted connections such that $w_{kj} = w_{jk}$ for all $k, j$ with $k \neq j$.

- **Asynchronous update procedure**: while state $\vec{v}$ is unstable: update an arbitrary unit.

- **Minimizes the energy function** $E(\vec{v}) = - \sum_{k<j} w_{kj} v_j v_k + \sum_k \theta_k v_k$.

Propositional Logic: Existing Approaches
Stochastic Networks or Boltzmann Machines

▶ Hinton, Sejnowski 1983: Can we escape local minima?
▶ A stochastic network is a symmetric network, but the values are computed probabilistically

\[ P(v_k = 1) = \frac{1}{1 + e^{(\theta_k - p_k)/T}} \]

where \( T \) is called pseudo temperature.
▶ In equilibrium stochastic networks are more likely to be in a state with low energy.
▶ Kirkpatrick et al. 1983: Can we compute a global minima?
▶ Simulated annealing: decrease \( T \) gradually.
▶ Theorem (Geman, Geman 1984)
  A global minima is reached if \( T \) is decreased in infinitesimal small steps.
Propositional Reasoning and Energy Minimization

- Pinkas 1991:
  Is there a link between propositional logic and symmetric networks?
- Let $D = \langle C_1, \ldots, C_m \rangle$ be a propositional formula in clause form.
- We define

\[
\tau(C) = \begin{cases} 
0 & \text{if } C = [], \\
p & \text{if } C = [p], \\
1 - p & \text{if } C = [\neg p], \\
\tau(C_1) + \tau(C_2) - \tau(C_1)\tau(C_2) & \text{if } C = (C_1 \lor C_2). 
\end{cases}
\]

\[
\tau(D) = \sum_{i=1}^{m} (1 - \tau(C_i))
\]

- Example

\[
\tau(\langle [\neg o, m], [\neg s, \neg m], [\neg c, m], [\neg c, s], [\neg v, \neg m] \rangle)
= vm - cm - cs + sm - om + 2c + o.
\]
Propositional Reasoning and Symmetric Networks

- **Theorem** \( \vec{v} \models D \) iff \( \tau(D) \) has a global minima at \( \vec{v} \).
- **Compare** \( \tau(D) = \) \( vm - cm - cs + sm - om + 2c + o \) with \( E(\vec{v}) = - \sum_{k<j} w_{kj} v_j v_k + \sum_k \theta_k v_k \).

\[
\begin{align*}
  u_1 &= o \\
  u_2 &= m \\
  u_3 &= s \\
  u_4 &= c \\
  u_5 &= v
\end{align*}
\]
Propositional Non-Monotonic Reasoning

- **Pinkas 1991a:**
  Can the above mentioned approach be extended to non-monotonic reasoning?

- Consider $D = \langle (C_1, k_1), \ldots, (C_m, k_m) \rangle$, where $C_i$ are clauses and $k_i \in \mathbb{R}^+$. 

- **The penalty** of $\vec{v}$ for $(C, k)$ is $k$ if $\vec{v} \not\models C$ and 0 otherwise.

- **The penalty** of $\vec{v}$ for $D$ is the sum of the penalties for $(C_i, k_i)$.

- $\vec{v}$ is preferred over $\vec{w}$ wrt $D$
  if the penalty of $\vec{v}$ for $D$ is smaller than the penalty of $\vec{w}$ for $D$.

- **Modify** $\tau$ to become $\tau(D) = \sum_{i=1}^{m} k_i (1 - \tau(C_i))$, e.g.,

  \[
  \tau(\langle (\neg o, m], 1), (\neg s, \neg m], 2), (\neg c, m], 4), (\neg c, s], 4), (\neg v, \neg m], 4) \rangle
  = 4vm - 4cm - 4cs + 2sm - om + 8c + o.
  \]

- **The corresponding stochastic network** computes most preferred interpretations.
Propositional Logic Programs and the Core Method

- The Very Idea
- Logic Programs
- Propositional Core Method
- Backpropagation
- Knowledge-Based Artificial Neural Networks
- Propositional Core Method using Sigmoidal Units
- Further Extensions
The Very Idea

- Various semantics for logic programs coincide with fixed points of associated immediate consequence operators (e.g., Apt, vanEmden 1982).

- **Banach Contraction Mapping Theorem** A contraction mapping $f$ defined on a complete metric space $(X, d)$ has a unique fixed point. The sequence $y, f(y), f(f(y)), \ldots$ converges to this fixed point for any $y \in X$.

  - *Fitting 1994*: Consider logic programs, whose immediate consequence operator is a contraction.

- **Funahashi 1989**: Every continuous function on the reals can be uniformly approximated by feedforward connectionist networks.

  - *Hölldobler, Kalinke, Störr 1999*: Consider logic programs, whose immediate consequence operator is continuous on the reals.
Metrics

A metric on a space $M$ is a mapping $d : M \times M \to \mathbb{R}$ such that

- $d(x, y) = 0$ iff $x = y$,
- $d(x, y) = d(y, x)$, and
- $d(x, y) \leq d(x, z) + d(z, y)$.

Let $(M, d)$ be a metric space and $S = (s_i \mid s_i \in M)$ a sequence.

- $S$ converges if $(\exists s \in M)(\forall \epsilon > 0)(\exists N)(\forall n \geq N) d(s_n, s) \leq \epsilon$.
- $S$ is Cauchy if $(\forall \epsilon > 0)(\exists N)(\forall n, m \geq N) d(s_n, s_m) \leq \epsilon$.
- $(M, d)$ is complete if every Cauchy sequence converges.

A mapping $f : M \to M$ is a contraction on $(M, d)$ if $(\exists 0 < k < 1)(\forall x, y \in M) d(f(x), f(y)) \leq k \cdot d(x, y)$.
Propositional Logic Programs

- A propositional logic program $\mathcal{P}$ over a propositional language $\mathcal{L}$ is a finite set of clauses

$$A \leftarrow L_1 \land \ldots \land L_n,$$

where $A$ is an atom, $L_i$ are literals and $n \geq 0$. $\mathcal{P}$ is definite if all $L_i$, $1 \leq i \leq n$ are atoms.

- Let $\mathcal{V}$ be the set of all propositional variables occurring in $\mathcal{L}$.

- An interpretation $I$ is a mapping $\mathcal{V} \rightarrow \{\top, \bot\}$.

- $I$ can be represented by the set of atoms which are mapped to $\top$ under $I$.

- $2^\mathcal{V}$ is the set of all interpretations.

- Immediate consequence operator $T_\mathcal{P} : 2^\mathcal{V} \rightarrow 2^\mathcal{V}$:

$$T_\mathcal{P}(I) = \{A \mid \text{there is a clause } A \leftarrow L_1 \land \ldots \land L_n \in \mathcal{P} \text{ such that } I \models L_1 \land \ldots \land L_n\}.$$

- $I$ is a supported model iff $T_\mathcal{P}(I) = I$. 
The Core Method

▶ Let \( \mathcal{L} \) be a logic language.
▶ Given a logic program \( \mathcal{P} \) together with immediate consequence operator \( T_\mathcal{P} \).
▶ Let \( \mathcal{I} \) be the set of interpretations for \( \mathcal{P} \).
▶ Find a mapping \( R : \mathcal{I} \rightarrow \mathbb{R}^n \).
▶ Construct a feed-forward network computing \( f_\mathcal{P} : \mathbb{R}^n \rightarrow \mathbb{R}^n \), called the core, such that the following holds:
  ▶ If \( T_\mathcal{P}(I) = J \) then \( f_\mathcal{P}(R(I)) = R(J) \), where \( I, J \in \mathcal{I} \).
  ▶ If \( f_\mathcal{P}(\vec{s}) = \vec{t} \) then \( T_\mathcal{P}(R^{-1}(\vec{s})) = R^{-1}(\vec{t}) \), where \( \vec{s}, \vec{t} \in \mathbb{R}^n \).
▶ Connect the units in the output layer recursively to the units in the input layer.
▶ Show that the following holds
  ▶ \( I = \text{lfp}(T_\mathcal{P}) \) iff the recurrent network converges to or approximates \( R(I) \).

Connectionist model generation using recurrent networks with feed forward core.
3-Layer Recurrent Networks

At each point in time all units do:

- apply activation function to obtain potential,
- apply output function to obtain output.
Propositional Core Method using Binary Threshold Units

Let $\mathcal{L}$ be the language of propositional logic over a set $\mathcal{V}$ of variables.

Let $\mathcal{P}$ be a propositional logic program, e.g.,

$$\mathcal{P} = \{ A, \ C \leftarrow A \land \neg \ B, \ C \leftarrow \neg A \land B \}.$$ 

$\mathcal{I} = 2^\mathcal{V}$ is the set of interpretations for $\mathcal{P}$.

$T_\mathcal{P}(I) = \{ A \mid A \leftarrow L_1 \land \ldots \land L_m \in \mathcal{P} \text{ such that } I \models L_1 \land \ldots \land L_m \}$.

- $T_\mathcal{P}(\emptyset) = \{ A \}$
- $T_\mathcal{P}(\{A\}) = \{ A, C \}$
- $T_\mathcal{P}(\{A, C\}) = \{ A, C \} = lfp(T_\mathcal{P})$
Representing Interpretations

- \( I = 2^\mathcal{V} \)

- Let \( n = |\mathcal{V}| \) and identify \( \mathcal{V} \) with \( \{1, \ldots, n\} \).

- Define \( R : I \to \mathbb{R}^n \) such that for all \( 1 \leq j \leq n \) we find:

\[
R(I)[j] = \begin{cases} 
1 & \text{if } j \in I, \\
0 & \text{if } j \not\in I.
\end{cases}
\]

E.g., if \( \mathcal{V} = \{A, B, C\} = \{1, 2, 3\} \) and \( I = \{A, C\} \) then \( R(I) = (1, 0, 1) \).

- Other encodings are possible, e.g.,

\[
R(I)[j] = \begin{cases} 
1 & \text{if } j \in I, \\
-1 & \text{if } j \not\in I.
\end{cases}
\]
Computing the Core

Consider again $\mathcal{P} = \{A, C \leftarrow A \land \neg B, C \leftarrow \neg A \land B\}$.

A translation algorithm translates $\mathcal{P}$ into a core of binary threshold units:
Some Results

- **Proposition** 2-layer networks cannot compute $T_P$ for definite $\mathcal{P}$.
- **Theorem** For each program $\mathcal{P}$, there exists a core computing $T_P$.
- **Recall** $\mathcal{P} = \{A, C \leftarrow A \land \neg B, C \leftarrow \neg A \land B\}$.
- **Adding recurrent connections:**

![Diagram of recurrent connections](image)
More Results

- A logic program $\mathcal{P}$ is said to be strongly determined if there exists a metric $d$ on the set of all Herbrand interpretations for $\mathcal{P}$ such that $T_{\mathcal{P}}$ is a contraction wrt $d$.

- Corollary: Let $\mathcal{P}$ be a strongly determined program. Then there exists a core with recurrent connections such that the computation with an arbitrary initial input converges and yields the unique fixed point of $T_{\mathcal{P}}$.

- Let $n$ be the number of clauses and $m$ be the number of propositional variables occurring in $\mathcal{P}$.
  - $2m + n$ units, $2mn$ connections in the core.
  - $T_{\mathcal{P}}(I)$ is computed in 2 steps.
  - The parallel computational model to compute $T_{\mathcal{P}}(I)$ is optimal.
  - The recurrent network settles down in $3n$ steps in the worst case.

Rule Extraction (1)

► Proposition

For each core $C$ there exists a program $\mathcal{P}$ such that $C$ computes $T_\mathcal{P}$.

<table>
<thead>
<tr>
<th>$u_1$</th>
<th>$u_2$</th>
<th>$u_3$</th>
<th>$u_4$</th>
<th>$u_5$</th>
<th>$u_6$</th>
<th>$u_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$p_3$</td>
<td>$v_3$</td>
<td>$p_4$</td>
<td>$v_4$</td>
<td>$p_5$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1.5</td>
<td>1</td>
<td>.3</td>
<td>1</td>
<td>.8</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>0</td>
<td>-.5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2.5</td>
<td>1</td>
<td>-.7</td>
<td>0</td>
<td>.3</td>
</tr>
</tbody>
</table>
Rule Extraction (2)

- Extracted program:

\[ P = \{ \begin{align*}
  A_1 &\leftarrow \neg A_1 \land \neg A_2, \\
  A_1 &\leftarrow \neg A_1 \land A_2, \\
  A_1 &\leftarrow A_1 \land \neg A_2, \\
  A_1 &\leftarrow A_1 \land A_2,
\end{align*} \}
\]

- Simplified form:

\[ P = \{ A_1, A_2 \leftarrow A_1, A_2 \leftarrow \neg A_1 \land A_2 \}. \]
3-Layer Feed-Forward Networks Revisited

▶ Theorem (Funahashi 1989) Suppose that $\Psi : \mathbb{R} \rightarrow \mathbb{R}$ is non-constant, bounded, monotone increasing and continuous. Let $K \subseteq \mathbb{R}^n$ be compact, let $f : K \rightarrow \mathbb{R}$ be continuous, and let $\varepsilon > 0$. Then there exists a 3-layer feed-forward network with output function $\Psi$ for the hidden layer and linear output function for the input and output layer whose input-output mapping $\overline{f} : K \rightarrow \mathbb{R}$ satisfies

$$\max_{x \in K} |f(x) - \overline{f}(x)| < \varepsilon.$$  

▶ Every continuous function $f : K \rightarrow \mathbb{R}$ can be uniformly approximated by input-output functions of 3-layer feed-forward networks.

▶ $u_k$ is a sigmoidal unit if

$$\Phi(\vec{i}_k) = p_k = \sum_{j=1}^{m} w_{kj} v_j,$$

$$\Psi(p_k) = v_k = \frac{1}{1 + e^{\beta(\theta_k - p_k)}}$$

where $\theta_k \in \mathbb{R}$ is a threshold (or bias) and $\beta > 0$ a steepness parameter.
Backpropagation

- Training set of input-output pairs \( \{(\vec{i}^l, \vec{o}^l) \mid 1 \leq l \leq n\} \).
- Minimize \( E = \sum_l E^l \) where \( E^l = \frac{1}{2} \sum_k (o_k^l - v_k^l)^2 \).
- Gradient descent algorithm to learn appropriate weights.

Backpropagation

1. Initialize weights arbitrarily.
2. Present input pattern \( \vec{i}^l \) at time \( t \).
3. Compute output pattern \( \vec{v}^l \) at time \( t + 2 \).
4. Change weights according to \( \Delta w_{ij}^l = \eta \delta_i^l v_j^l \), where

- \( \delta_i^l = \begin{cases} \Psi'_i(p_i^l) \times (o_i^l - v_i^l) & \text{if } i \text{ is output unit}, \\ \Psi'_i(p_i^l) \times \sum_k \delta_k^l w_{ki} & \text{if } i \text{ is hidden unit}, \end{cases} \)
- \( \eta > 0 \) is called learning rate.
Knowledge Based Artificial Neural Networks

▶ Towell, Shavlik 1994: Can we do better than empirical learning?
▶ Sets of hierarchical logic programs, e.g.,

\[ \mathcal{P} = \{A \leftarrow B \land C \land \neg D, \ A \leftarrow D \land \neg E, \ H \leftarrow F \land G, \ K \leftarrow A, \neg H\}. \]
Knowledge Based Artificial Neural Networks – Learning

- Given hierarchical sets of propositional rules as background knowledge.
- Map rules into multi-layer feed forward networks with sigmoidal units.
- Add hidden units (optional).
- Add units for known input features that are not referenced in the rules.
- Fully connect layers.
- Add near-zero random numbers to all links and thresholds.
- Apply backpropagation.

Empirical evaluation: system performs better than purely empirical and purely hand-built classifiers.
Knowledge Based Artificial Neural Networks – A Problem

- Works if rules have few conditions and there are few rules with the same head.

\[ p_A = p_B = 9\omega \text{ and } v_A = v_B = \frac{1}{1+e^{\beta(9.5\omega-9\omega)}} \approx 0.46 \text{ with } \beta = 1. \]

\[ p_C = 0.92\omega \text{ and } v_c = \frac{1}{1+e^{\beta(0.5\omega-0.92\omega)}} \approx 0.6 \text{ with } \beta = 1. \]
Propositional Core Method using Bipolar Sigmoidal Units

- d’Avila Garcez, Zaverucha, Carvalho 1997:
  Can we combine the ideas in Hölldobler, Kalinke 1994 and Towell, Shavlik 1994 while avoiding the above mentioned problem?

- Consider propositional logic language.

- Let \( I \) be an interpretation and \( a \in [0, 1] \).

\[
R(I)[j] = \begin{cases} 
\nu \in [a, 1] & \text{if } j \in I, \\
\omega \in [-1, -a] & \text{if } j \not\in I.
\end{cases}
\]

- Replace threshold and sigmoidal units by bipolar sigmoidal ones, i.e., units with

\[
\Phi(\vec{i}_k) = p_k = \sum_{j=1}^{m} w_{kj} \nu_j, \\
\Psi(p_k) = \nu_k = \frac{2}{1 + e^{\beta(\theta_k - p_k)}} - 1,
\]

where \( \theta_k \in \mathbb{R} \) is a threshold (or bias) and \( \beta > 0 \) a steepness parameter.
The Task

- How should $a$, $\omega$ and $\theta_i$ be selected such that:
  - $v_i \in [a, 1]$ or $v_i \in [-1, -a]$ and
  - the core computes the immediate consequence operator?
Hidden Layer Units

- Consider $A \leftarrow L_1 \land \ldots \land L_n$.
- Let $u$ be the hidden layer unit for this rule.

  - Suppose $I \models L_1 \land \ldots \land L_n$.
    - $u$ receives input $\geq \omega a$ from unit representing $L_i$.
    - $p_u \geq n\omega a = p^+_u$.
  
  - Suppose $I \not\models L_1 \land \ldots \land L_n$.
    - $u$ receives input $\leq -\omega a$ from at least one unit representing $L_i$.
    - $p_u \leq (n - 1)\omega 1 - \omega a = p^-_u$.

- $\theta_u = \frac{n\omega a + (n - 1)\omega - \omega a}{2} = (na + n - 1 - a)\frac{\omega}{2} = (n - 1)(a + 1)\frac{\omega}{2}$. 

Propositional Logic Programs and the Core Method
Output Layer Units

- Let \( \mu \) be the number of clause with head \( A \).
- Consider \( A \leftarrow L_1 \land \ldots \land L_n \).
- Suppose \( I \models L_1 \land \ldots \land L_n \).
  \[ p_A \geq \omega a + (\mu - 1)\omega(-1) = \omega a - (\mu - 1)\omega = p_A^+ \]
- Suppose for all rules of the form \( A \leftarrow L_1 \land \ldots \land L_n \) we find \( I \not\models L_1 \land \ldots \land L_n \).
  \[ p_A \leq -\mu \omega a = p_A^- \]
- \( \theta_A = \frac{\omega a - (\mu - 1)\omega - \mu \omega a}{2} = (a - \mu + 1 - \mu a)\frac{\omega}{2} = (1 - \mu)(a + 1)\frac{\omega}{2} \)
Computing a Value for $a$

- $p_u^+ > p_u^-:$
  - $n\omega a > (n - 1)\omega - \omega a.$
  - $n\omega a + \omega a > (n - 1)\omega.$
  - $a(n + 1)\omega > (n - 1)\omega.$
  - $a > \frac{n-1}{n+1}.$

- $p_A^+ > p_A^-:$
  - $\omega a - (\mu - 1)\omega > -\mu a \omega.$
  - $\omega a + \mu a \omega > (\mu - 1)\omega.$
  - $a(1 + \mu)\omega > (\mu - 1)\omega.$
  - $a > \frac{\mu-1}{\mu+1}.$

- Consider all rules $\iff$ minimum value for $a.$
Computing a Value for $\omega$

- $\Psi(p) = \frac{2}{1+e^{\beta(\theta-p)}} - 1 \geq a.$
- $\frac{2}{1+e^{\beta(\theta-p)}} \geq 1 + a.$
- $\frac{2}{1+a} \geq 1 + e^{\beta(\theta-p)}.$
- $\frac{2}{1+a} - 1 = \frac{2}{1+a} - \frac{1+a}{1+a} = \frac{1-a}{1+a} \geq e^{\beta(\theta-p)}.$
- $\ln\left(\frac{1-a}{1+a}\right) \geq \beta(\theta - p).$
- $\frac{1}{\beta} \ln\left(\frac{1-a}{1+a}\right) \geq \theta - p.$

Consider a hidden layer unit:

- $\frac{1}{\beta} \ln\left(\frac{1-a}{1+a}\right) \geq (n-1)(a+1)\frac{\omega}{2} - n\omega a = \frac{na+n-a-1-2na}{2} = \frac{n-1-a(n+1)}{2} \omega.$
- $\omega \geq \frac{2}{(n-1-a(n+1))\beta} \ln\left(\frac{1-a}{1+a}\right)$ because $a \geq \frac{n-1}{n+1}.$

Consider all hidden and output layer units as well as the case that $\Psi(p) \leq -a$:

minimum value for $\omega.$
Results

- Relation to logic programs is preserved.
- The core is trainable by backpropagation.
- Many interesting applications, e.g.:
  - DNA sequence analysis.
  - Power system fault diagnosis.
- Empirical evaluation:
  system performs better than well-known machine learning systems.
- See d’Avila Garcez, Broda, Gabbay 2002 for details.
Further Extensions

- Many-valued logic programs
- Modal logic programs
- Answer set programming
- Metalevel priorities
- Rule extraction
Propositional Core Method – Three-Valued Logic Programs

- **Kalinke 1994**: Consider truth values $\top$, $\bot$, $u$.
- Interpretations are pairs $I = \langle I^+, I^- \rangle$.
- Immediate consequence operator $\Phi_P(I) = \langle J^+, J^- \rangle$, where
  
  \[ J^+ = \{ A \mid A \leftarrow L_1 \land \ldots \land L_m \in \mathcal{P} \text{ and } I(L_1 \land \ldots \land L_m) = \top \}, \]
  
  \[ J^- = \{ A \mid \text{for all } A \leftarrow L_1 \land \ldots \land L_m \in \mathcal{P} : I(L_1 \land \ldots \land L_m) = \bot \}. \]

- Let $n = |\mathcal{V}|$ and identify $\mathcal{V}$ with $\{1, \ldots, n\}$.
- Define $R : \mathcal{I} \to \mathbb{R}^{2n}$ as follows:
  
  \[ R(I)[2j - 1] = \begin{cases} 
    1 & \text{if } j \in I^+ \\
    0 & \text{if } j \notin I^+ 
  \end{cases} \quad \text{and} \quad R(I)[2j] = \begin{cases} 
    1 & \text{if } j \in I^- \\
    0 & \text{if } j \notin I^- 
  \end{cases} \]
Propositional Core Method – Multi-Valued Logic Programs

▶ For each program $\mathcal{P}$, there exists a core computing $\Phi_\mathcal{P}$, e.g.,

$$\mathcal{P} = \{ C \leftarrow A \land \neg B, \ D \leftarrow C \land E, \ D \leftarrow \neg C \}. $$

▶ Lane, Seda 2004: Extension to finitely determined sets of truth values.
Propositional Core Method – Modal Logic Programs

- Let $\mathcal{L}$ be a propositional logic language plus
  - the modalities $\Box$ and $\Diamond$, and
  - a finite set of labels $w_1, \ldots, w_k$ denoting worlds.
- Let $B$ be an atom, then $\Box B$ and $\Diamond B$ are modal atoms.
- A modal definite logic program $\mathcal{P}$ is a set of clauses of the form
  \[ w_i : A \leftarrow A_1 \land \ldots \land A_m \]
  together with a finite set of relations $w_i \rhd w_j$, where $w_i, 1 \leq i, j \leq k$, are labels and $A, A_1, \ldots, A_m$ are atoms or modal atoms.
- $\mathcal{P} = \bigcup_{i=1}^{k} \mathcal{P}_i$, where $\mathcal{P}_i$ consists of all clauses labelled with $w_i$. 
Modal Logic Programs – Semantics

Example: \[ \mathcal{P} = \{ w_1 : A, \ w_1 : \Diamond C \leftarrow A \} \]
\[ \cup \{ w_2 : B \} \]
\[ \cup \{ w_3 : B \} \]
\[ \cup \{ w_4 : B \} \]
\[ \cup \{ w_1 \rightarrow w_2, \ w_1 \rightarrow w_3, \ w_1 \rightarrow w_4, \ w_2 \rightarrow w_4, \} \]

Kripke semantics:

\[ f_C(w_1) = w_4 \]
Modal Immediate Consequence Operator

- Interpretations are tuples $I = \langle I_1, \ldots, I_k \rangle$
- Immediate consequence operator $MT_\mathcal{P}(I) = \langle J_1, \ldots, J_k \rangle$, where

$$J_i = \left\{ A \mid \text{there exists } A \leftarrow A_1 \land \ldots \land A_m \in \mathcal{P}_i \right. \wedge \left. \text{such that } \{A_1, \ldots, A_m\} \subseteq I_i \right\} \cup \left\{ \Diamond A \mid \text{there exists } w_i \rightarrow w_j \in \mathcal{P} \text{ and } A \in I_j \right\} \cup \left\{ \Box A \mid \text{for all } w_i \rightarrow w_j \in \mathcal{P} \text{ we find } A \in I_j \right\} \cup \left\{ A \mid \text{there exists } w_j \rightarrow w_i \in \mathcal{P} \text{ and } \Box A \in I_j \right\} \cup \left\{ A \mid \text{there exists } w_j \rightarrow w_i \in \mathcal{P}, \Diamond A \in I_j \text{ and } f_A(w_j) = w_i \right\}$$
Modal Logic Programs – The Translation Algorithm

- Let $n = |\mathcal{V}|$ and identify $\mathcal{V}$ with $\{1, \ldots, n\}$.
- Let $a \in [0, 1]$.
- Define $R : \mathcal{I} \rightarrow \mathbb{R}^{3n}$ as follows:

\[
R(I)[3j - 2] = \begin{cases} 
    v \in [a, 1] & \text{if } j \in I_j \\
    w \in [-1, -a] & \text{if } j \not\in I_j 
\end{cases}
\]

\[
R(I)[3j - 1] = \begin{cases} 
    v \in [a, 1] & \text{if } \Box j \in I_j \\
    w \in [-1, -a] & \text{if } \Box j \not\in I_j 
\end{cases}
\]

\[
R(I)[3j] = \begin{cases} 
    v \in [a, 1] & \text{if } \Diamond j \in I_j \\
    w \in [-1, -a] & \text{if } \Diamond j \not\in I_j 
\end{cases}
\]

- Translation algorithm such that
  - for each world the “local” part of $MT_P$ is computed by a core,
  - the cores are turned into recurrent networks, and
  - the cores are connected with respect to the given set of relations.
The Example Network

Propositional Logic Programs and the Core Method
First-Order Logic

- Existing Approaches
  - Reflexive Reasoning and SHRUTI
  - Connectionist Term Representations
    - Holographic Reduced Representations Plate 1991
    - Recursive Auto-Associative Memory Pollack 1988
  - Horn logic and CHCL Hölldobler 1990, Hölldobler, Kurfess 1992
  - Other Approaches

- First-Order Logic Programs and the Core Method
  - Initial Approach
  - Construction of Approximating Networks
  - Topological Analysis and Generalisations
  - Employing Iterated Function Systems
Reflexive Reasoning

- Humans are capable of performing a wide variety of cognitive tasks with extreme ease and efficiency.
- For traditional AI systems, the same problems turn out to be intractable.
- Human consensus knowledge: about $10^8$ rules and facts.
- Wanted: “Reflexive” decisions within sublinear time.
- Shastri, Ajjanagadde 1993: SHRUTI.
Finite set of constants $C$, finite set of variables $\mathcal{V}$.

Rules:
1. $(\forall X_1 \ldots X_m) (p_1(\ldots) \land \ldots \land p_n(\ldots) \rightarrow (\exists Y_1 \ldots Y_k p(\ldots)))$.
2. $p$, $p_i$, $1 \leq i \leq n$, are multi-place predicate symbols.
3. Arguments of the $p_i$: variables from $\{X_1, \ldots, X_m\} \subseteq \mathcal{V}$.
4. Arguments of $p$ are from $\{X_1, \ldots, X_m\} \cup \{Y_1, \ldots, Y_k\} \cup C$.
5. $\{Y_1, \ldots, Y_k\} \subseteq \mathcal{V}$.
6. $\{X_1, \ldots, X_m\} \cap \{Y_1, \ldots, Y_k\} = \emptyset$.

Facts and queries (goals):
1. $(\exists Z_1 \ldots Z_l) q(\ldots)$.
2. Multi-place predicate symbol $q$.
3. Arguments of $q$ are from $\{Z_1, \ldots, Z_l\} \cup C$.
4. $\{Z_1, \ldots, Z_l\} \subseteq \mathcal{V}$.
Further Restrictions

- Restrictions to rules, facts, and goals:
  - No function symbols except constants.
  - Only universally bound variables may occur as arguments in the conditions of a rule.
  - All variables occurring in a fact or goal occur only once and are existentially bound.
  - An existentially quantified variable is only unified with variables.
  - A variable which occurs more than once in the conditions of a rule must occur in the conclusion of the rule and must be bound when the conclusion is unified with a goal.
  - A rule is used only a fixed number of times.

⇔ Incompleteness.
SHRUTI – Example

▶ Rules
\[ p = \{ \text{owns}(Y, Z) \leftarrow \text{gives}(X, Y, Z), \]
\[ \text{owns}(X, Y) \leftarrow \text{buys}(X, Y), \]
\[ \text{can-sell}(X, Y) \leftarrow \text{owns}(X, Y), \]
\[ \text{gives}(\text{john}, \text{josephine}, \text{book}), \]
\[ (\exists X) \text{buys}(\text{john}, X), \]
\[ \text{owns}(\text{josephine}, \text{ball}) \} , \]

▶ Queries:
\[ \text{can-sell}(\text{josephine}, \text{book}) \rightsquigarrow \text{yes} \]
\[ (\exists X) \text{owns}(\text{josephine}, X) \rightsquigarrow \text{yes} \{X \mapsto \text{book}\} \]
\[ \{X \mapsto \text{ball}\} \]
SHRUTI : The Network

can-sell

owns

gives

from john
from jos.
from book

buys

from john

book
john
ball
josephine
Solving the Variable Binding Problem

First-Order Logic
SHRUTI – Remarks

- Answers are derived in time proportional to depth of search space.
- Number of units as well as of connections is linear in the size of the knowledge base.
- Extensions:
  - compute answer substitutions
  - allow a fixed number of copies of rules
  - allow multiple literals in the body of a rule
  - built in a taxonomy
- Biological plausibility.
- Trading expressiveness for time and size.
- Logical reconstruction by Beringer, Hölldobler 1993:
  - Reflexive reasoning is reasoning by reduction.
First-Order Logic Programs and the Core Method

- Initial Approach
- Construction of Approximating Networks
- Topological Analysis and Generalisations
- Employing Iterated Function Systems
Logic Programs

- A logic program $\mathcal{P}$ over a first-order language $\mathcal{L}$ is a finite set of clauses

$$A \leftarrow L_1 \land \ldots \land L_n,$$

where $A$ is an atom, $L_i$ are literals and $n \geq 0$.

- $B_\mathcal{L}$ is the set of all ground atoms over $\mathcal{L}$ called Herbrand base.

- A Herbrand interpretation $I$ is a mapping $B_\mathcal{L} \rightarrow \{\top, \bot\}$.

- $2^{B_\mathcal{L}}$ is the set of all Herbrand interpretations.

- $\text{ground}(\mathcal{P})$ is the set of all ground instances of clauses in $\mathcal{P}$.

- Immediate consequence operator $T_\mathcal{P} : 2^{B_\mathcal{L}} \rightarrow 2^{B_\mathcal{L}}$:

$$T_\mathcal{P}(I) = \{ A \mid \text{there is a clause } A \leftarrow L_1 \land \ldots \land L_n \in \text{ground}(\mathcal{P}) \text{ such that } I \models L_1 \land \ldots \land L_n \}.$$ 

- $I$ is a supported model iff $T_\mathcal{P}(I) = I$. 

First-Order Logic Programs and the Core Method
The Initial Approach

▶ Hölldobler, Kalinke, Störr 1999:
Can the core method be extended to first-order logic programs?

▶ Problem

➢ Given a logic program $\mathcal{P}$ over a first order language $\mathcal{L}$ together with $T_{\mathcal{P}} : 2^{B_{\mathcal{L}}} \rightarrow 2^{B_{\mathcal{L}}}$.

➢ $B_{\mathcal{L}}$ is countably infinite.

➢ The method used to relate propositional logic and connectionist systems is not applicable.

➢ How can the gap between the discrete, symbolic setting of logic, and the continuous, real valued setting of connectionist networks be closed?
The Goal

- Find \( R : 2^{\mathcal{L}} \rightarrow \mathbb{R} \) and \( f_\mathcal{P} : \mathbb{R} \rightarrow \mathbb{R} \) such that the following conditions hold.
  - \( T_\mathcal{P}(I) = I' \) implies \( f_\mathcal{P}(R(I)) = R(I') \).
  - \( f_\mathcal{P}(x) = x' \) implies \( T_\mathcal{P}(R^{-1}(x)) = R^{-1}(x') \).

  \( f_\mathcal{P} \) is a sound and complete encoding of \( T_\mathcal{P} \).

- \( T_\mathcal{P} \) is a contraction on \( 2^{\mathcal{L}} \) iff \( f_\mathcal{P} \) is a contraction on \( \mathbb{R} \).

  The contraction property and fixed points are preserved.

- \( f_\mathcal{P} \) is continuous on \( \mathbb{R} \).

  A connectionist network approximating \( f_\mathcal{P} \) is known to exist.
Acyclic Logic Programs

Let \( \mathcal{P} \) be a program over a first order language \( \mathcal{L} \).

A level mapping for \( \mathcal{P} \) is a function \( l : B_\mathcal{L} \rightarrow \mathbb{N} \).

We define \( l(\neg A) = l(A) \).

We can associate a metric \( d_\mathcal{L} \) with \( \mathcal{L} \) and \( l \). Let \( I, J \in 2^{B_\mathcal{L}} \):

\[
d_\mathcal{L}(I, J) = \begin{cases} 
0 & \text{if } I = J \\
2^{-n} & \text{if } n \text{ is the smallest level on which } I \text{ and } J \text{ differ.}
\end{cases}
\]

Proposition (Fitting 1994) \((2^{B_\mathcal{L}}, d_\mathcal{L})\) is a complete metric space.

\( \mathcal{P} \) is said to be acyclic wrt a level mapping \( l \), if for every \( A \leftarrow L_1 \land \ldots \land L_n \in \text{ground}(\mathcal{P}) \) we find \( l(A) > l(L_i) \) for all \( i \).

Proposition Let \( \mathcal{P} \) be an acyclic logic program wrt \( l \) and \( d_\mathcal{L} \) the metric associated with \( \mathcal{L} \) and \( l \), then \( T_\mathcal{P} \) is a contraction on \((2^{B_\mathcal{L}}, d_\mathcal{L})\).
Mapping Interpretations to Real Numbers

- Let $\mathcal{D} = \{ r \in \mathbb{R} \mid r = \sum_{i=1}^{\infty} a_i 4^{-i}, \text{ where } a_i \in \{0, 1\} \text{ for all } i \}$.  
- Let $l$ be a bijective level mapping.  
- $\{\top, \bot\}$ can be identified with $\{0, 1\}$.  
- The set of all mappings $B_L \to \{\top, \bot\}$ can be identified with the set of all mappings $\mathbb{N} \to \{0, 1\}$.  
- Let $I_L$ be the set of all mappings from $B_L$ to $\{0, 1\}$.  
- Let $R : I_L \to \mathcal{D}$ be defined as  
  $$R(I) = \sum_{i=1}^{\infty} I(l^{-1}(i)) 4^{-i}.$$  
- Proposition $R$ is a bijection.

We have a sound and complete encoding of interpretations.
We define \( f_P : D \to D : r \mapsto R(T_P(R^{-1}(r))) \).

We have a sound and complete encoding of \( T_P \).

**Proposition** Let \( P \) be an acyclic program wrt a bijective level mapping. \( f_P \) is a contraction on \( D \).

Contraction property and fixed points are preserved.
Approximating Continuous Functions

► Corollary $f_P$ is continuous.

► Recall Funahashi’s theorem:

▶ Every continuous function $f : K \rightarrow \mathbb{R}$ can be uniformly approximated by input-output functions of 3-layer feed forward networks.

► Theorem $f_P$ can be uniformly approximated by input-output functions of 3-layer feed forward networks.

▶ $T_P$ can be approximated as well by applying $R^{-1}$. 

Connectionist network approximating immediate consequence operator exists.
An Example

Consider \( P = \{ q(0), q(s(X)) \leftarrow q(X) \} \) and let \( l(q(s^n(0))) = n + 1 \).

\( P \) is acyclic wrt \( l \), \( l \) is bijective, \( R(B_L) = \frac{1}{3} \).

\[
\begin{align*}
\mathcal{f}_P(R(I)) &= 4^{-l(q(0))} + \sum_{q(X) \in I} 4^{-l(q(s(X)))} \\
&= 4^{-l(q(0))} + \sum_{q(X) \in I} 4^{-(l(q(X))+1)} = \frac{1+R(I)}{4}.
\end{align*}
\]

Approximation of \( \mathcal{f}_P \) to accuracy \( \varepsilon \) yields

\[
\overline{f}(x) \in \left[ \frac{1 + x}{4} - \varepsilon, \frac{1 + x}{4} + \varepsilon \right].
\]

Starting with some \( x \) and iterating \( \overline{f} \) yields in the limit a value

\[
r \in \left[ \frac{1 - 4\varepsilon}{3}, \frac{1 + 4\varepsilon}{3} \right].
\]

Applying \( R^{-1} \) to \( r \) we find

\[
q(s^n(0)) \in R^{-1}(r) \text{ if } n < -\log_4 \varepsilon - 1.
\]
Approximation of Interpretations

- Let $\mathcal{P}$ be a logic program over a first order language $\mathcal{L}$ and $l$ a level mapping.
- An interpretation $I$ approximates an interpretation $J$ to a degree $n \in \mathbb{N}$ if for all atoms $A \in B_\mathcal{L}$ with $l(A) < n$ we find $I(A) = \top$ iff $J(A) = \top$.

$I$ approximates $J$ to a degree $n$ iff $d_\mathcal{L}(I, J) \leq 2^{-n}$. 
Approximation of Supported Models

- Given an acyclic logic program $\mathcal{P}$ with bijective level mapping.
- Let $T_{\mathcal{P}}$ be the immediate consequence operator associated with $\mathcal{P}$ and $M_{\mathcal{P}}$ the least supported model of $\mathcal{P}$.
- We can approximate $T_{\mathcal{P}}$ by a 3-layer feed forward network.
- We can turn this network into a recurrent one.

Does the recurrent network approximate the supported model of $\mathcal{P}$?

Theorem For an arbitrary $m \in \mathbb{N}$ there exists a recursive network with sigmoidal activation function for the hidden layer units and linear activation functions for the input and output layer units computing a function $\overline{f}_{\mathcal{P}}$ such that there exists an $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$ and for all $x \in [-1, 1]$ we find

$$d_{\mathcal{L}}(R^{-1}(\overline{f}_{\mathcal{P}}^n(x)), M_{\mathcal{P}}) \leq 2^{-m}.$$
Literature


Lane, Seda 2004: Some Aspects of the Integration of Connectionist and Logic-Based Systems. *University College Cork*.


