Continuity of Semantic Operators and Their Approximation by Artificial Neural Networks

Pascal Hitzler and Anthony Karel Seda

ISDT01, Chengdu, October 2001

We want to represent operators associated with Logic Programs by Artificial Neural Networks.

P. Hitzler, Artificial Intelligence Institute, Department of Computer Science, Dresden University of Technology, Germany phitzler@inf.tu-dresden.de
http://www.wv.inf.tu-dresden.de/~pascal/

A.K. Seda, Department of Mathematics, National University of Irland, University College Cork
a.seda@ucc.ie http://maths.ucc.ie/staff/seda/

Idea

- Logic Programs and Neural Networks are very different paradigms.
- Neural Networks can uniformly approximate continuous real operators.
- We study this kind of continuity for Logic Programs
- and use it for obtaining approximation results.

The approach builds on work by Hölldobler, Kalinke and Störr 199x.

We thank Howard A. **Blair** (Syracuse), Steffen **Hölldobler**, and Hans-Peter **Störr** (Dresden) for helpful discussions on the subject matter.

Logic Programs

A logic program P is a finite set of clauses

$$\forall(A \leftarrow L_1 \wedge \cdots \wedge L_n)$$

from first order logic, usually written as

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

where A an atom, L_i a literal, $n \geq 0$.
B_P: Herbrand base (all ground atoms).
$I_P = 2^{B_P}$: set of all Herbrand interpretations.
ground(P): set of all ground instances
of clauses of P.

Define (non-monotonic) operator $T_P : I_P \to I_P$ by
$T_P(I)$ is set of all $A \in B_P$
for which there is a clause $A \leftarrow L_1 \wedge \cdots \wedge L_n$
in ground(P) s.t. $I \models L_1 \wedge \cdots \wedge L_n$.

I is a *supported model* iff $T_P(I) = I$.

T_P operator in 2-valued logic.
Many-valued logic has also been studied.

Many-valued Interpretations

Truth values $\mathcal{T} = \{t_1, \ldots, t_n\}$.

Interpretations are functions $I : B_P \to \mathcal{T}$.

$I_{P,n} (= I_P)$ set of all interpretations.

$A \in B_P$ then $\mathcal{B}_A$ set of all atoms in clauses of ground(P) with head A.

$T : I_P \to I_P$ *consequence operator* for P
if for all $I \in I_P$ and for all $A \leftarrow \texttt{body}$ in P,
where $T(I)(A) = t_i$ and $I(\texttt{body}) = t_j$, say,
$t_i \leftarrow t_j$ is true via truth table.

Consequence operator T is *local* if for all $A \in B_P$ and all $I, K \in I_P$ which agree on all atoms in $\mathcal{B}_A$, we have $T(I)(A) = T(K)(A)$.

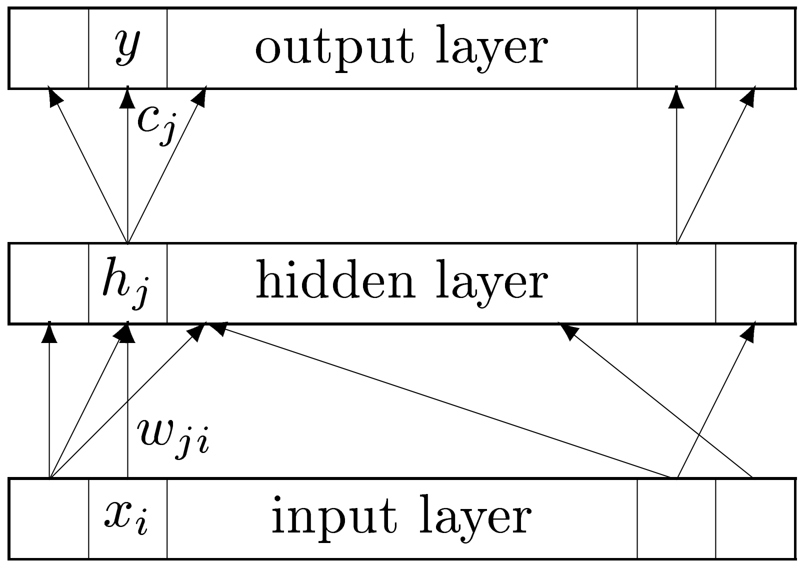
T_P is a local consequence operator.

Other examples: Operators due to Fitting (1985, 199x), in 3- and 4-valued logic.

Artificial Neural Networks I

A *3-layer feedforward network* (3ffn) consists of

- finitely many computational units
- organized in three layers:
  - input layer, hidden layer, output layer
- weighted connections between units
  - from input to hidden layer and
  - from hidden to output layer.



x_i	inputs
w_{ji}, c_j	connection weights
y	output

Artificial Neural Networks II

The input-output function $f : \mathbb{R}^n \to \mathbb{R}$ is

$$y = f(x_1, \ldots, x_r) = \sum_j c_j \phi \left(\sum_i w_{ji} x_i - \theta_j \right)$$

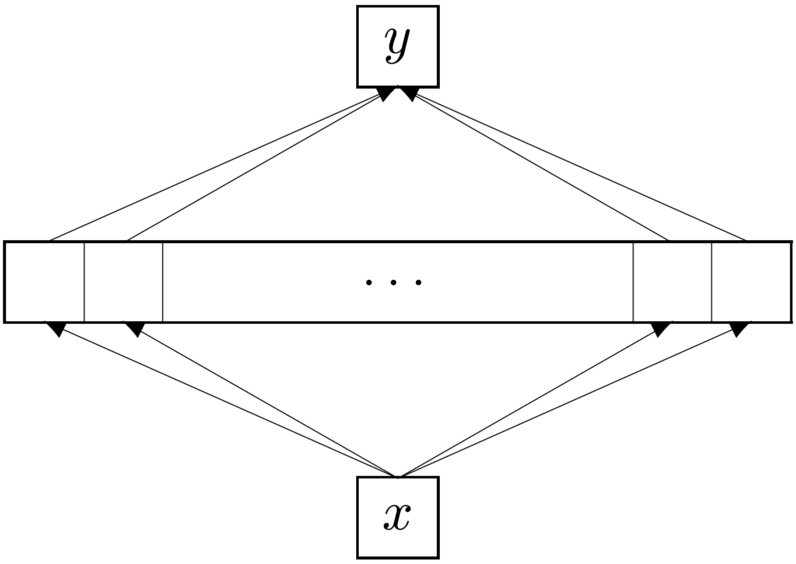
with *thresholds* $\theta_j \in \mathbb{R}$ and *squashing function* ϕ.

$\phi : \mathbb{R} \to \mathbb{R}$ is the same for each unit and usually

- non-constant, bounded, monotonic increasing,
- sometimes continuous.

The following architecture will suffice:

- one input unit x
- one output unit y



LPs versus ANNs

Neural Networks:

- approximates ("interpolates") functions
- hardly any knowledge about the fct^n needed
- trained using incomplete data
- declarative semantics not available
- recursive networks hardly understood
- symbolic data difficult to represent

Logic Programming:

- direct implementation of relations
- explicit expert knowledge required
- highly recursive structure
- well understood declarative semantics
- symbolic data easy to represent

Seek the best of both paradigms!

Approximation of LPs by ANNs

Theorem (Funahashi 1989, simplified version):

ϕ non-constant, bounded, monotone increasing, continuous.

$K \subseteq \mathbb{R}$ compact,

$f : K \to \mathbb{R}$ continuous,

$\varepsilon > 0$.

Then exists a 3ffn with squashing fct$^{\mathrm{n}}$ ϕ and input-output function $\bar{f} : K \to \mathbb{R}$ with

$$\max_{x \in K} \left\{ d\left(f(x), \bar{f}(x)\right) \right\} < \varepsilon;$$

d metric which induces natural top. on $\mathbb{R}$.

- "Each continuous function $f : K \to \mathbb{R}$ can be uniformly approximated by input-output functions of 3ffns."

The approach in [HKS]

Let P be a logic program which is *acyclic*, i.e. there exists a *level mapping* $l : B_P \to \mathbb{N}$ such that for each $A \leftarrow L_1, \dots, L_n$ in ground(P) we have $l(A) > l(L_i)$ for all $i = 1, \dots, n$.

We can define a complete metric d_l on I_P by $d_l(J, K) = 2^{-n}$, where $n \in \mathbb{N}$ is least s.t. J and K differ on an atom of level n.

For P acyclic, T_P is a contraction wrt. d_l.

- Banach contraction mapping theorem applies.
- T_P has unique fixed point M.
- M can be obtained as limit (in d_l) of the sequence $(T_P^n(K))_{n\in\mathbb{N}}$ for any $K \in I_P$.
- T_P is continuous wrt. d_l.

For injective l and acyclic P, [HKS] gave imbedding $\iota : I_P \to \mathbb{R}$ s.t. $\iota(T_P)$ was contraction on $\mathbb{R}$.

Generalized Atomic Topology $\mathcal{Q}$

Extends atomic topology (Seda 1995) and query topology (Batharek and Subrahmanian 1989).

Given P we define $\mathcal{Q}$ on I_P to be the product topology on $\mathcal{T}^{B_P}$, where $\mathcal{T} = \{t_1, \ldots, t_n\}$ is endowed with the discrete topology.

$\mathcal{Q}$ second countable totally disconnected compact Hausdorff topology which is dense in itself.

$\mathcal{Q}$ is metrizable and homeomorphic to the Cantor topology on the unit interval of the real line. (Note B_P is countable.)

Cantor Space $\mathcal{C}$ with subspace topology from $\mathbb{R}$ carries Cantor topology.

$\mathcal{C}$ compact subset of $\mathbb{R}$.

Continuity in $\mathcal{Q}$

Consequence operator T on I_P is *finitely local* if for all $A \in B_P$ and all $I \in I_P$ there exists a finite subset $S \subseteq \mathcal{B}_A$ such that $T(J)(A) = T(I)(A)$ for all $J \in I_P$ which agree with I on S.

Theorem

> A local consequence operator is finitely local if and only if it is continuous in $\mathcal{Q}$.

Conditions which imply that T is finitely local:

- P has no local variables.
- There exists injective *level mapping* $l : B_P \to \mathbb{N}$ such that for each $A \in B_P$ there exists $n_A \in \mathbb{N}$ such that $l(B) < n_A$ for all $B \in \mathcal{B}_A$. (Communicated by H.A. Blair.)

Main Theorem

Theorem

Let P be a logic program and T a consequence operator which is finitely local, and let ι be a homeomorphism from $(I_{P,n}, \mathcal{Q})$ to $\mathcal{C}$.
Then T (more precisely $\iota(T)$) can be uniformly approximated by input-output mappings of 3-layer feedforward networks.

Measurability I

Theorem (Hornik, Stinchcombe, White 1989, simplified version)

$\phi : \mathbb{R} \to (0,1)$ monotone increasing, surjective.
$f : \mathbb{R} \to \mathbb{R}$ Borel-measurable,
μ probability Borel-measure on $\mathbb{R}$,
$\varepsilon > 0$.

Then exists 3ffn with squashing fct$^{\mathrm{n}}$ ϕ and input-output function $\bar{f} : \mathbb{R} \to \mathbb{R}$ with
$\varrho_\mu(f, \bar{f}) =$

$$\inf \left\{ \delta > 0 : \mu \left\{ x : \left| f(x) - \bar{f}(x) \right| > \delta \right\} < \delta \right\} < \varepsilon.$$

- "The class of functions computed by 3ffns is dense in the set of all Borel measurable functions $f : \mathbb{R} \to \mathbb{R}$ rel. to the metric ϱ_μ."

Measurability II

Theorem

Local consequence operators are always measurable with respect to $\sigma(\mathcal{Q})$.

Approximation is only *almost everywhere*
i.e. except a set of measure 0.

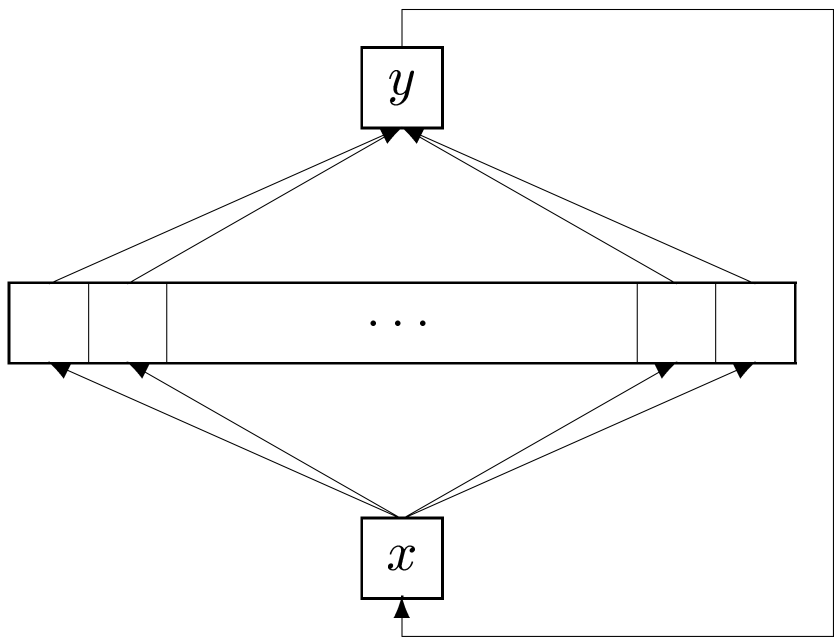
$\iota(I_P) = \mathcal{C}$ is set of measure 0.

Possible approach: "Blowing up" of Cantor set in order to give it positive measure.

Recurrent Architecture I

- Approximation results give no way of constructing the network.
- Is the obtained approximation sufficient?

[HKS] use the following recurrent architecture:



- Network iterates consequence operator.

Recurrent Architecture II

T locally finite local consequence operator.
f I/O function of approximating network.

For any $I \in I_P$ and any $n \in \mathbb{N}$ we have

$$|f^n(\iota(I)) - \iota(T^n(I))| \leq \varepsilon \frac{1-\lambda^n}{1-\lambda}.$$

λ Lipschitz constant of F which is the extension of $\iota(T)$ onto $[0,1]$.
ε error of the network.

If F is a contraction on $[0,1]$,
then $\left(F^k(\iota(I))\right)$ converges for every I to the unique fixed point x of F and there exists $m \in \mathbb{N}$ such that for all $n \geq m$ we have

$$|f^n(\iota(I)) - x| \leq \varepsilon \frac{1}{1-\lambda}.$$

If F is a contraction on $[0,1]$, then T is a contraction on the complete subspace $\mathcal{C}$, and also has a fixed point M, and $\iota(M) = x$.

Recurrent Architecture III

If for some $I \in I_P$, $T^n(I)$ converges in $\mathcal{Q}$ to a fixed point M of T, then for every $\delta > 0$ there exists a network with input-output function f, and some $n \in \mathbb{N}$ such that $|f^n(\iota(I)) - \iota(M)| < \delta$.

A logic program P is called *acyclic* if there exists a mapping $l : B_P \to \mathbb{N}$, called a *level mapping*, such that for each clause $A \leftarrow L_1, \ldots, L_n$ in ground(P) we have $l(A) > l(L_i)$ for all $i = 1, \ldots, n$.

Define $d : I_P \times I_P \to \mathbb{R}$ by $d(I, J) = 2^{-n}$, where n is least such that I and J differ on some atom A with $l(A) = n$.
d is a complete metric on I_P.

T is a contraction with respect to d if P is acyclic. Banach contraction theorem applies.

Let P be an acyclic program and T be a local consequence operator for P. Then for any $I \in I_P$ we have that $T^n(I)$ converges in $\mathcal{Q}$ to the unique fixed point M of T.

Conclusions

Proposed the study of nonmonotonic semantic operators in multi-valued logic. This extends the tools available for studying declarative semantics of logic programs.

Obtained theoretical results concerning the representation of first order logic programs by neural networks.

Questions:

Constructing the networks?

How to circumvent the measurability problem?