JOHANNES KEPLER
UNIVERSITY LINZ

Neural Logic Machines

Formal Methods and Automated Reasoning Seminar

Acile Chahboun

JOHANNES KEPLER
UNIVERSITY LINZ
Altenberger Straße 69
4040 Linz, Austria
jku.at

Introduction

Overview

Neural Logic Machines

A Neurosymbolic Architecture combining Neural Networks with Symbolic Logic to enable both inductive learning and logical reasoning.

References

Dong, H., Mao, J., Lin, T., Wang, C., Li, L., & Zhou, D. (2019).
Neural Logic Machines.
arXiv:1904.11694.
https://arxiv.org/abs/1904.11694

Key Terminology

- **Machine Learning:** Uses statistical models and optimization to learn patterns from data and make predictions or classifications.
- **Symbolic Reasoning:** Manipulates structured knowledge representations using predefined logical rules to perform explicit, rule-based inference.
- **Multilayer Perceptron:** Multi-Layer Perceptron (MLP) it's a feedforward neural network that consists fully connected dense layers that transform input data from one dimension to another. It is called multi-layer because it contains an input layer, one or more hidden layers and an output layer. The purpose of an MLP is to model complex relationships between inputs and outputs.
- **Neuro-symbolic AI:** Combines machine learning with symbolic representations and logical reasoning to support both data-driven learning and structured inference.

The core problem: Learning vs. Reasoning

Learning (Neural)	Reasoning (Symbolic)
Acquires representations from data	Derives conclusions from known facts
Adjusts parameters to reduce error	Applies explicit rules and constraints
Statistical generalization	Logical inference
Bottom-up	Top-down
Approximate	Exact
Implicit knowledge	Explicit knowledge
Answers: *What is likely?*	Answers: *What must be true?*
Struggles with explicit logic, long inference chains, guarantees, and truthfulness (hallucinations)	Struggles with learning from data, perception, noise, and scalability

Why Neuro-Symbolic AI?

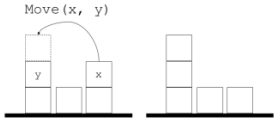
Neuro-symbolic approaches address the limitations of purely neural and purely symbolic methods by integrating learned patterns with explicit reasoning, allowing a system to both learn from data and reason about that data and the broader structure in which it exists.
This combination more closely reflects human intelligence, which relies on both learning from experience and reasoning over abstract concepts.

The Architecture

Objective/thesis

This architecture shows that neural networks can approximate forward-chaining relational reasoning with strong size generalization, without explicit symbolic rules or proofs.

Block World Reasoning Task



$\texttt{On}(x, y)$	True if x is on y
$\texttt{IsGround}(x)$	True if x is the ground
$\texttt{Clear}(x)$	True if there is no block on x
$\texttt{Moveable}(x)$	$\neg\texttt{IsGround}(x) \wedge \texttt{Clear}(x)$
$\texttt{Placeable}(x)$	$\texttt{IsGround}(x) \vee \texttt{Clear}(x)$

Figure 1: (Left) A graphical illustration of the blocks world. Given an initial and a target worlds, the agent is required to move blocks to transform the initial configuration to the target one. (Right) A set of sentences used throughout the paper to define the blocks world.

Figure: Block world reasoning task (Dong et al., 2019).

- Objects are blocks described by logical predicates
- Input: relations such as `On(x,y)`, `Clear(x)`, `IsGround(x)`
- Goal: infer new predicates (e.g. `Move(x,y)`) to reach target world
- Reasoning is symbolic and relational, not perceptual

Architecture Overview

Stage	Intuition / Meaning
Input state	A spreadsheet of facts about the world
Arity separation	Facts about one object vs object pairs
Reduce	“Is there something related to *x*?” (∃-like summary)
Expand	Broadcast a fact across an additional variable
Concatenation	Collect all relevant facts
MLP (rule)	A learned numeric rule applied everywhere
Training	Numbers adjusted to fit examples
Interpretation	Humans assign logical meaning after training

Architecture Overview

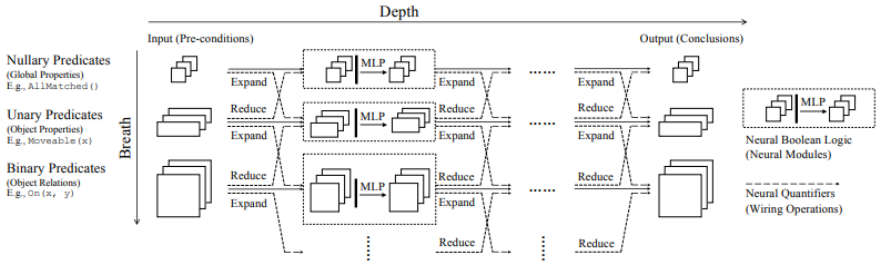


Figure 2: An illustration of Neural Logic Machines (NLM). During forward propagation, NLM takes object properties and relations as input, performs sequential logic deduction, and outputs conclusive properties or relations of the objects. Implementation details can be found in Section 2.3.

Figure: Overview of the Neural Logic Machine architecture (Dong et al., 2019; arXiv:1904.11694).

Architecture Details

Stage	What happens	Why this exists
Input state	Objects and relations stored as predicate tables	Logic operates on relations, not vectors
Arity separation	Predicates grouped by arity (unary, binary, etc.)	Rules operate at specific arities
Reduce	Aggregate higher-arity predicates (max / min)	Summarize many relations into one fact
Expand	Broadcast lower-arity predicates to higher arity	Align variables across predicates
Concatenation	Aligned predicates stacked together	Combine all available information
MLP (rule)	Shared neural network computes new predicates	Learn how facts combine

Logic Predicates as Tensors

- Neural Logic Machines represent logical predicates as tensors
- Predicate arity corresponds to tensor order
  - Unary predicates → vectors
  - Binary predicates → matrices
  - Higher-arity predicates → higher-order tensors
- Each tensor entry corresponds to a grounded object tuple
- Values in $[0, 1]$ represent (probabilistic) truth

Neural Logic Machines: Example

- **World:** Objects $\{A, B, C\}$ with facts: IsGround(A) = 1; On(B, A) = 1, On(C, B) = 1.
- **Unary tensor:**

$$X^{(1)} = [1,\ 0,\ 0]$$

- **Binary tensor (On(y, x)):**

	A	B	C
B	1	0	0
C	0	1	0

Logic Rules as Neural Operators: Boolean Logic

- Meta-rule (1): Boolean logic over predicates

$$\hat{p}(x_1, \dots, x_r) \leftarrow \text{expression}(x_1, \dots, x_r)$$

- Expression consists of logical operators (AND, OR, NOT)
- All predicates have the same arity

Neural encoding (high-level):

- Predicate tensors are concatenated
- An MLP learns the boolean combination
- Output is a new predicate tensor (same arity)

Example (Block World):

$$\texttt{Moveable}(x) \leftarrow \neg \texttt{IsGround}(x) \wedge \texttt{Clear}(x)$$

Logic Rules as Neural Operators: Quantification

- Meta-rule (3): **Expansion** (adds a variable)

$$p(x_1, \dots, x_r) \rightarrow p(x_1, \dots, x_r, x_{r+1})$$

- Meta-rule (4): **Reduction** (removes a variable)

$$\forall x_{r+1}\ p(x_1, \dots, x_r, x_{r+1})$$

Neural encoding (high-level):

- Expansion: repeat predicate tensor along a new dimension
- Reduction: aggregate over a dimension (e.g. max / min)
- Enables interaction between predicates of different arity

Example (Block World):

$$\texttt{Clear}(x) \leftarrow \neg \texttt{On}(y, x)$$

Neural Logic Machines: Example

- **Reduce:**

$$\text{HasTop}(x) = \max_y \text{On}(y, x) = [1,\ 1,\ 0]$$

- **Concat (unary):**

$$Z(x) = [\text{IsGround}(x),\ \text{HasTop}(x)]$$

- **MLP inputs:** $A : [1, 1]$, $B : [0, 1]$, $C : [0, 0]$
- **MLP outputs:**

$$[0,\ 0,\ 1]$$

- **Interpretation:**

$$\text{Clear}(x) = [0,\ 0,\ 1]$$

The shared MLP learned a soft rule behaving like ¬IsGround(x) AND ¬HasTop(x).

Neural Logic Machines

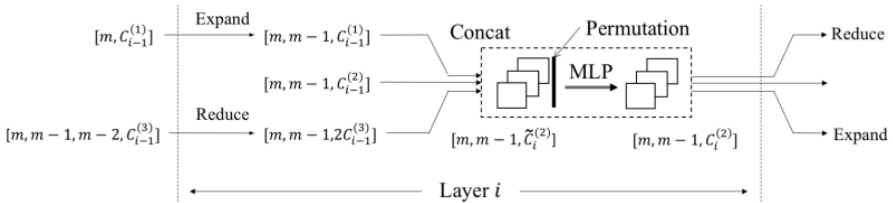


Figure 3: An illustration of the computational block inside NLM for binary predicates at layer i. $C_i^{(j)}$ denotes the number of output predicates of group j at layer i. $[\cdot]$ denotes the shape of the tensor.

Figure: Computational block of a Neural Logic Machine layer (Dong et al., 2019).

Layer Computation in Neural Logic Machines

Notation: $O_i^{(r)}$ denotes predicates of arity r at layer i, one layer corresponds to one reasoning step.

Inter-group computation (Eq. 5)

$$I_i^{(r)} = \text{Concat}\left(\text{Expand}\left(O_{i-1}^{(r-1)}\right), O_{i-1}^{(r)}, \text{Reduce}\left(O_{i-1}^{(r+1)}\right)\right)$$

$I_i^{(r)}$ combines information across arities, Expand increases arity, Reduce decreases arity.

Intra-group computation (Eq. 6)

$$O_i^{(r)} = \sigma\left(\text{MLP}\left(\text{Permute}\left(I_i^{(r)}\right); \theta_i^{(r)}\right)\right)$$

Permute enforces variable-order invariance, $\theta_i^{(r)}$ are trainable MLP parameters, σ maps outputs to $[0, 1]$.

Neural Logic Machines: Example

- **Target:**

$$\text{Moveable}(x) = [0,\ 0,\ 1]$$

- **Learning:**
  - Layer 1 forms Clear(x)
  - Layer 2 uses Clear(x) and IsGround(x)
- **Backprop:** Gradients update MLP weights so patterns like $[0, 0] \rightarrow 1$ reduce loss.
- **Why it generalizes:** Shared weights + permutation invariance suppress shortcuts.

Intermediate predicates survive only if they consistently reduce final error.

Evaluation

Baselines

- **Memory Networks (MemNN):** Neural models with explicit memory and attention for multi-hop fact retrieval.
- **Differentiable ILP (∂ILP):** Softened ILP that enumerates clause templates and learns continuous weights via gradient descent.

Experimental Tasks

- **Family Tree**: infer kinship relations from base facts.
- **Graph Reasoning**: infer connectivity, degree, and adjacency.
- **Blocks World**: plan moves under logical constraints.
- **Sorting**: learn a swap-based sorting algorithm.
- **Shortest Path**: choose actions to reach a target node.

Neural Logic Machines: Experiments+results

Task	Learning	Train	Test	Comparison to Baselines
Family Tree	Supervised	20	20, 100	NLM: **100% accuracy**; MemNN degrades with size; ∂ILP accurate but less scalable
Graph Reasoning	Supervised	10	10, 50	NLM: **100% accuracy**; MemNN drops sharply; ∂ILP fails on higher-arity rules
Blocks World	Reinforcement Learning	≤12	10, 50	NLM: **100% success**; MemNN: **0% success**; ∂ILP not scalable

Task	Learning	Train	Test	Comparison to Baselines
Sorting	Reinforcement Learning	≤12	10, 50	NLM: **100% success**, few swaps; MemNN inefficient and fails to scale
Shortest Path	Reinforcement Learning	≤12	Larger	NLM: **100% success**, optimal paths; MemNN/DNC low success rates

Neural Logic Machines: Strengths

- Learns *lifted, size-invariant rules* that generalize beyond training instances.
- Combines symbolic structure (variables, arity, quantifiers) with neural learning.
- Strong *systematic generalization*, outperforming standard neural baselines.
- Supports both relational reasoning and decision-making / algorithmic tasks.
- Fully differentiable and trainable end-to-end.
- Architecture enforces interpretable reasoning stages (Expand, Reduce, Boolean combination).

Neural Logic Machines: Weaknesses

- Logic is *soft and approximate*, lacking strict logical guarantees.
- Learned rules are implicit in weights, not explicit symbolic clauses.
- No recursion or cyclic reasoning; depth and arity must be fixed in advance.
- Computational cost increases rapidly with predicate arity.
- Training can be fragile for RL tasks; relies on curriculum learning.
- Practical scalability limited to moderate numbers of objects.
- Each task requires a separately trained model.
- Training instability (RL): Results often reported on graduated runs; success depends on random seed, curriculum, and multiple restarts (non-trivial failure rate despite perfect performance when training succeeds).

Open Questions in Neuro-Symbolic AI

- **Rule extraction:** How to reliably extract explicit, human-readable rules from neural models.
- **Logical guarantees:** How to combine differentiability with exact, sound logical reasoning.
- **Recursion and depth:** How to support recursive and unbounded reasoning without fixed depth.
- **Scalability:** How to scale reasoning to large numbers of objects and knowledge bases.
- **Data efficiency:** How to learn rules from sparse, weak, or noisy supervision.
- **Task transfer:** How to reuse learned rules across tasks and domains.
- **Perception–reasoning integration:** How to connect raw sensory inputs with symbolic reasoning robustly.
- **Theory:** What formal guarantees (soundness, completeness, expressiveness) are possible.

Conclusion

Neural Logic Machines demonstrate that neural networks can learn structured, symbolic-style reasoning by fixing the reasoning structure and learning how predicates are combined. This enables strong systematic generalization compared to standard neural models.

However, the learned logic is soft and implicit in the weights, reasoning depth is fixed, and explicit logical guarantees are not provided. This reflects a broader tension in neuro-symbolic AI between learnability and semantic control.

This naturally motivates Logic Tensor Networks, which take a complementary approach by enforcing explicit logical constraints through the learning objective and will be covered in the next presentation.