Neural=Symbolic

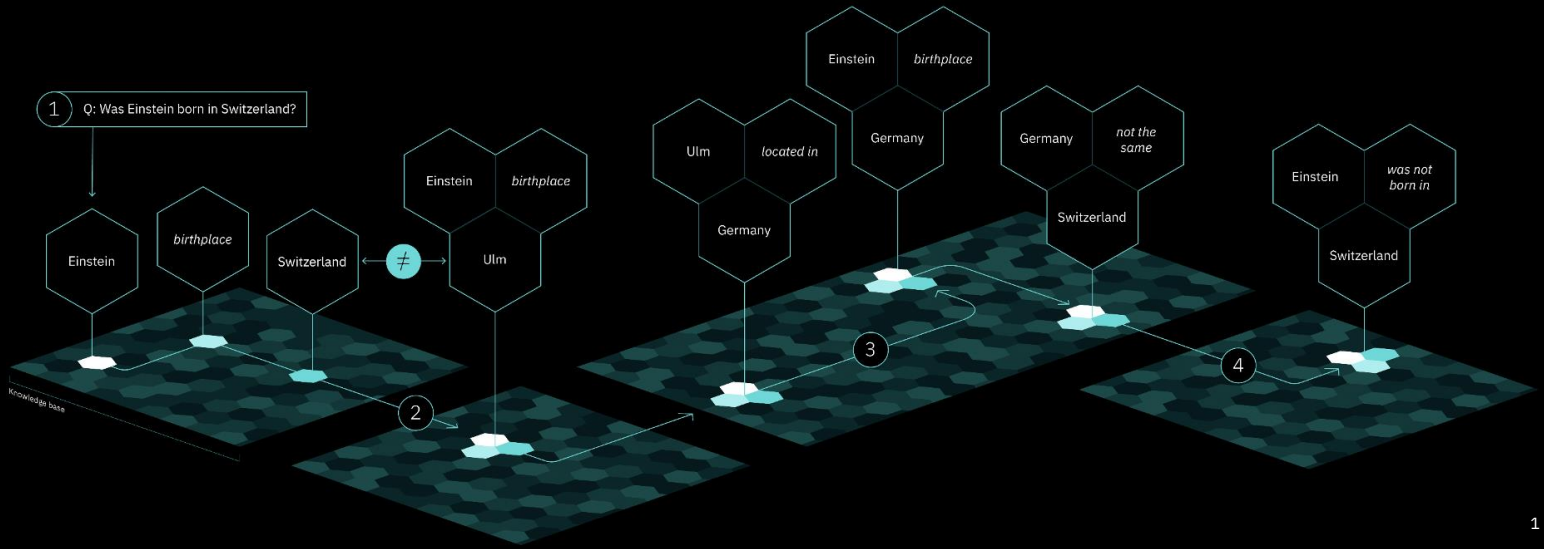
A New Paradigm of Logical Neural Networks

Ryan Riegel

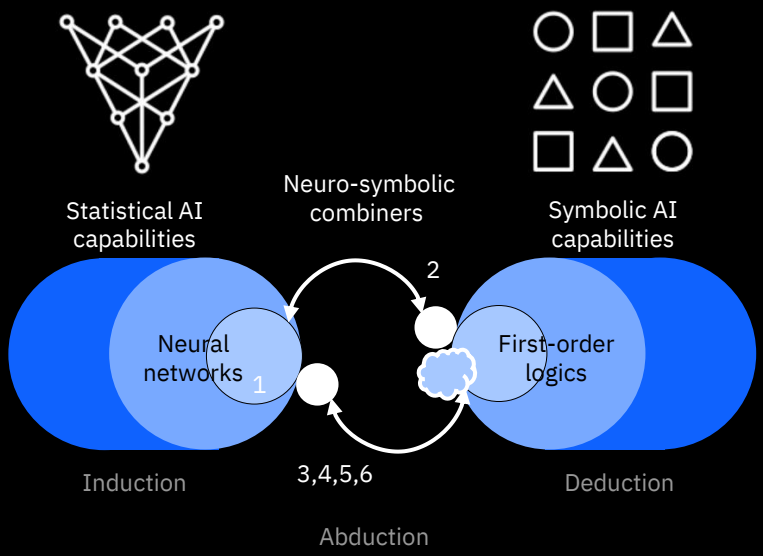
AI Reasoning group

IBM Research

Neuro-symbolic AI has recently gained significant interest amid growing industrial requirements for high-performance models that are nonetheless interpretable, verifiable, and adaptable to new problem domains with a minimum of reconfiguration. Numerous distinct categories of such methods have emerged, often characterized either as neural nets somehow informed by symbolic logic or as symbolic logic somehow extracted from neural nets. In contrast, we introduce a new paradigm to the mix, Neural=Symbolic, in which the underlying neural model exactly corresponds to a system of logical formulae in any of various real-valued logics (with classical logic as a special case). Evaluation of such a Logical Neural Network (LNN) performs deductive inference in the associated logical system and can answer complex, zero-shot queries rather than focusing exclusively on predefined outputs. LNNs can easily incorporate existing domain knowledge, but can also learn weights on (sub)formulae so as to minimize logical contradiction, thereby yielding resilience to inconsistency. Additionally, LNNs are careful to distinguish true, false, intermediate, and unknown truth values according to the open-world assumption by working in terms of bounds rather than individual values, thereby yielding resilience to incomplete knowledge. Lastly, LNNs are beginning to generate state-of-the-art results with respect to both theory and application.



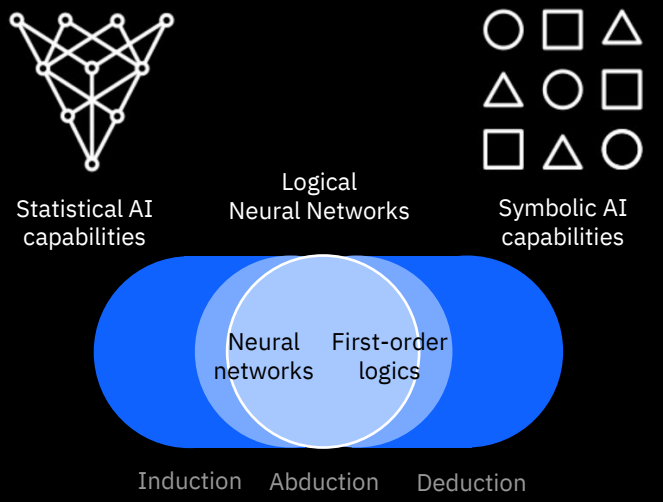
Neuro-symbolic methods so far



- Neuro-symbolic combination patterns:
  1. symbolic Neural symbolic — standard DL, 2011+
  2. Symbolic[Neural] — AlphaGo, 2016
  3. Neural ; Symbolic — NS Concept Learner, 2019
  4. Neural: Symbolic → Neural — MLN, 2006; ProbLog, 2007
  5. $\text{Neural}_{\text{Symbolic}}$ — LTN, 2016; NTP, 2017
  6. Neural[Symbolic] — NTM, 2014; TRAIL, 2019
- Most common goals:
  1. Understandability (via human-readable symbolic form)
     - *But: Maintain two representations, including black box*
  2. Better task generalizability (via reusable knowledge)
     - *But: Non-compositional models are not reusable*
  3. More complex problems (via adding reasoning ability)
     - *But: Non-rigorous reasoning, simpler logics*

Garcez, 2019; Belle, 2020 (surveys)
Kautz, 2020 https://www.cs.rochester.edu/u/kautz/talks/index.html
Gray, 2020 http://ibm.biz/neuro-symbolic-ai

Neuro-symbolic methods: another category



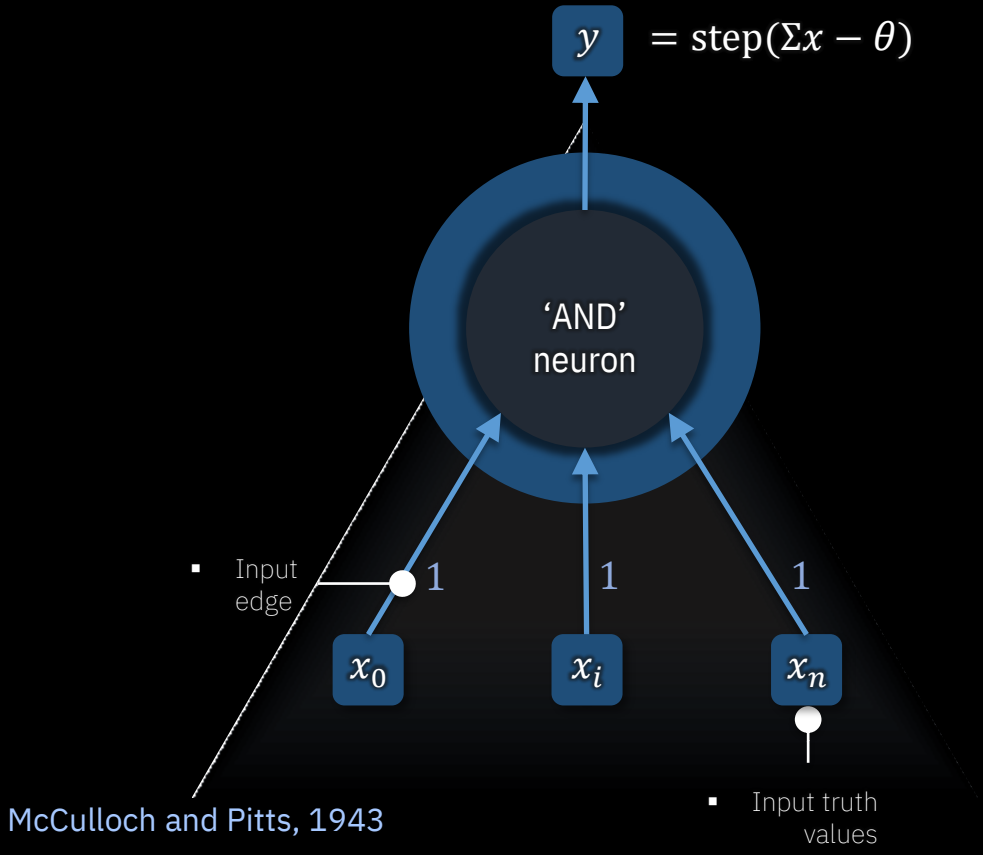
- Added neuro-symbolic combination pattern:

1.	symbolic Neural symbolic	standard DL, 2011+
2.	Symbolic[Neural]	AlphaGo, 2016
3.	Neural ; Symbolic	NS Concept Learner, 2019
4.	Neural: Symbolic → Neural	MLN, 2006; ProbLog, 2007
5.	$\text{Neural}_{\text{Symbolic}}$	LTN, 2016; NTP, 2017
6.	Neural[Symbolic]	NTM, 2014; TRAIL, 2019
7.	Neural=Symbolic	Logical Neural Networks, 2020

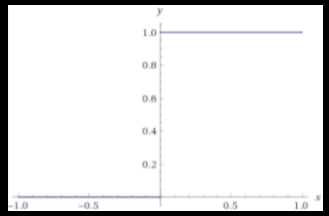
- Most common goals:
  1. Understandability (via human-readable symbolic form)
     - Single human-readable representation, not two
  2. Less data (generalize over tasks via reusable knowledge)
     - Sub-models are composable/modular/reusable
  3. More complex problems (via adding reasoning ability)
     - Rigorous foundation 1) making both NNs and classic logic special cases, 2) (bonus) formalizing abduction

Garcez, 2019; Belle, 2020 (surveys)
Kautz, 2020 https://www.cs.rochester.edu/u/kautz/talks/index.html
Gray, 2020 http://ibm.biz/neuro-symbolic-ai

1. Original (McCulloch and Pitts 1943) *neuron as logic gate*



McCulloch and Pitts, 1943

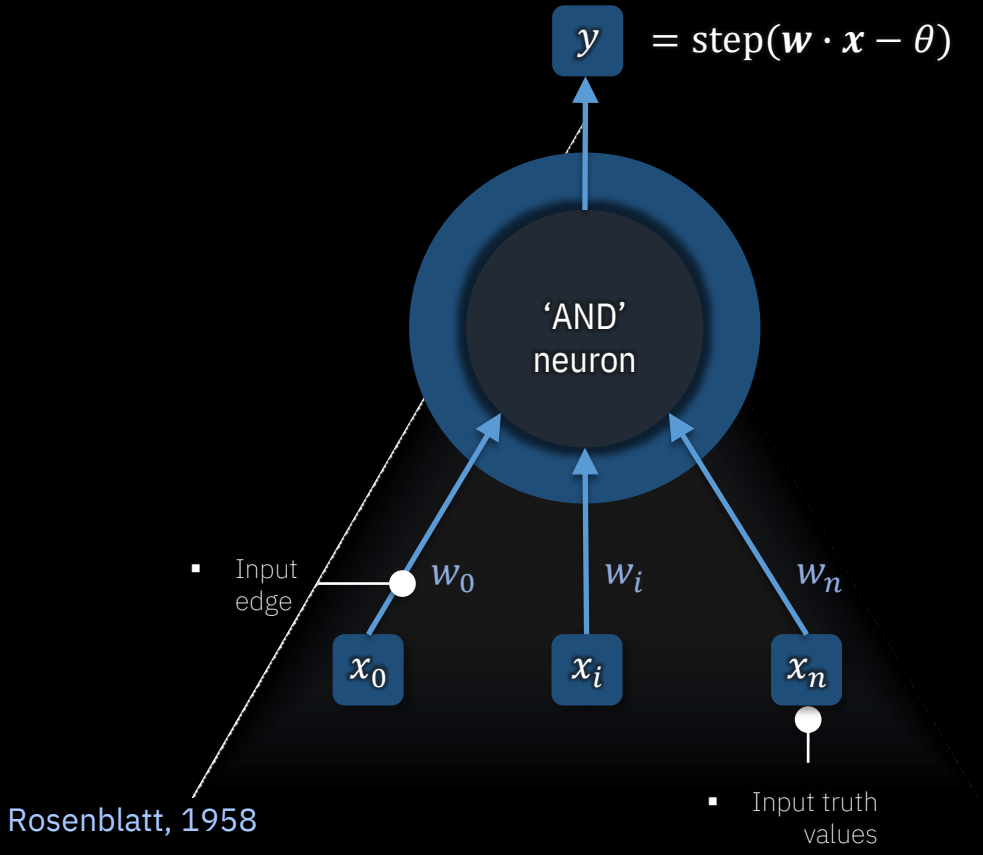


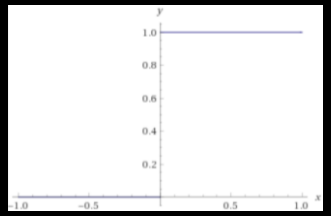
- *Literally the first artificial neuron model was intended to model logical gates*
- 0/1 inputs and outputs, variable number of inputs
- This precisely achieves (and generalizes) classical 'AND' behavior:

$$p \wedge q = \quad [\![p + q > 1.5]\!]$$
$$p \vee q = \quad [\![p + q > 0.5]\!]$$
$$p \rightarrow q = [\![1 - p + q > 0.5]\!]$$

p	q	$p \wedge q$
0	0	0
1	0	0
0	1	0
1	1	1

2. *Weighted* neuron (perceptron, 1958) as logic gate





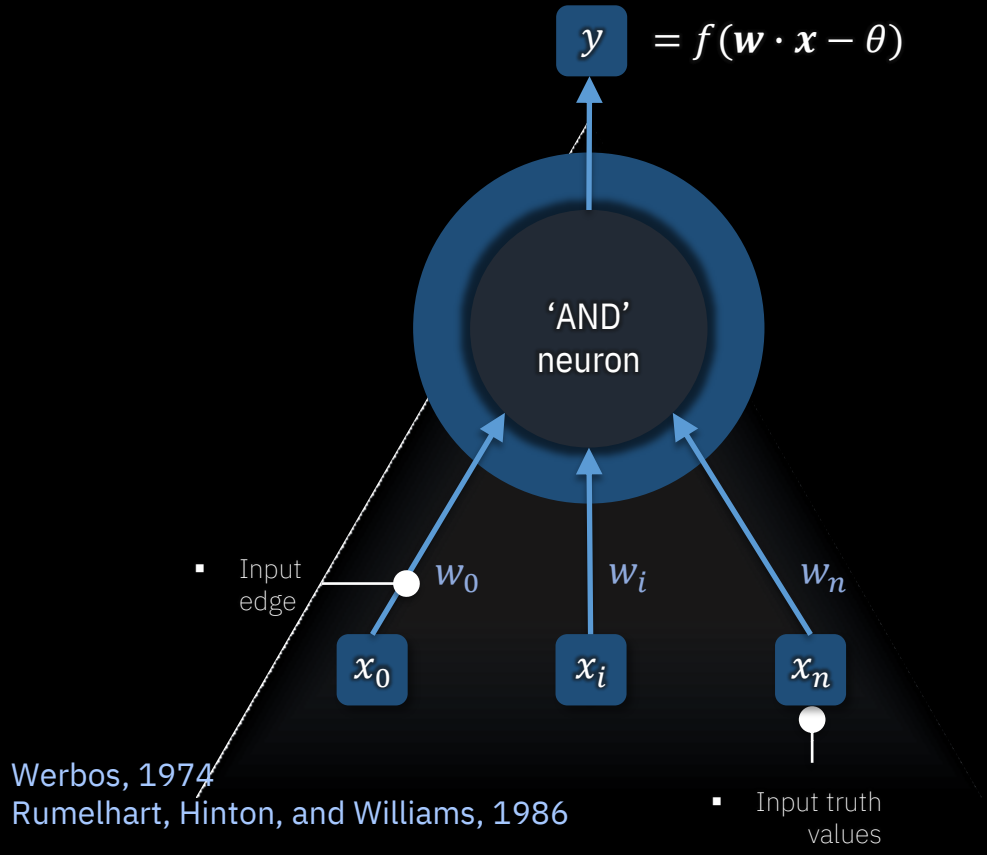
- Now add weights (and way to learn them)
- *Observe that 'AND' behavior is achieved in a constrained region of the weight space:*

$$\sum_i w_i - \theta > 0 \quad \text{Conditions for true output}$$

$$\forall i, \sum_j w_j - w_i - \theta \leq 0 \quad \text{Conditions for false output}$$

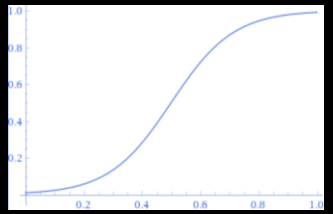
- Intuition: Even one false input to 'AND' must result in false, but all inputs true must result in true

3. *Differentiable* neuron (MLPs, deep learning) as logic gate



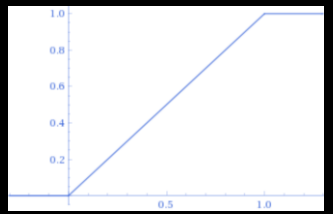
Werbos, 1974
Rumelhart, Hinton, and Williams, 1986

Sigmoid



Wilson and Cowan, 1972

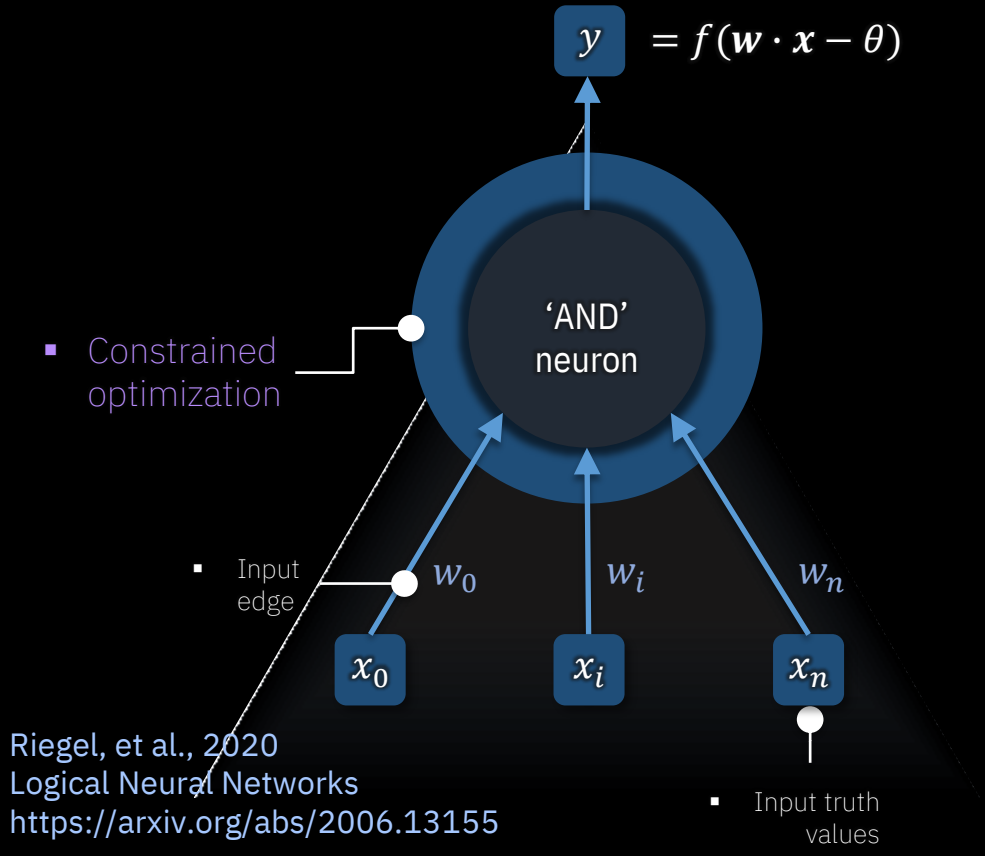
ReLU



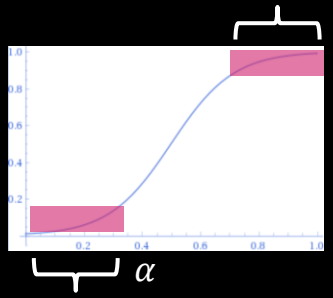
Hahnloser, et al., 2000
Goodfellow, et al., 2015

- Soften the step function to have derivatives
- Train multiple connected neurons via backpropagation
- *However, since inputs/outputs are now not just 0 or 1, we no longer have a connection to classical logic as we did in previous neuron models*

4a. *Constrained* differentiable neuron (LNN) as logic gate



Riegel, et al., 2020
Logical Neural Networks
https://arxiv.org/abs/2006.13155



"Classical region"

Can use any odd, monotonic activation function f with range [0,1] scaled such that $f(\alpha) = \alpha$ and $f(1-\alpha) = 1-\alpha$ (including sigmoid, [0,1]-ReLU)

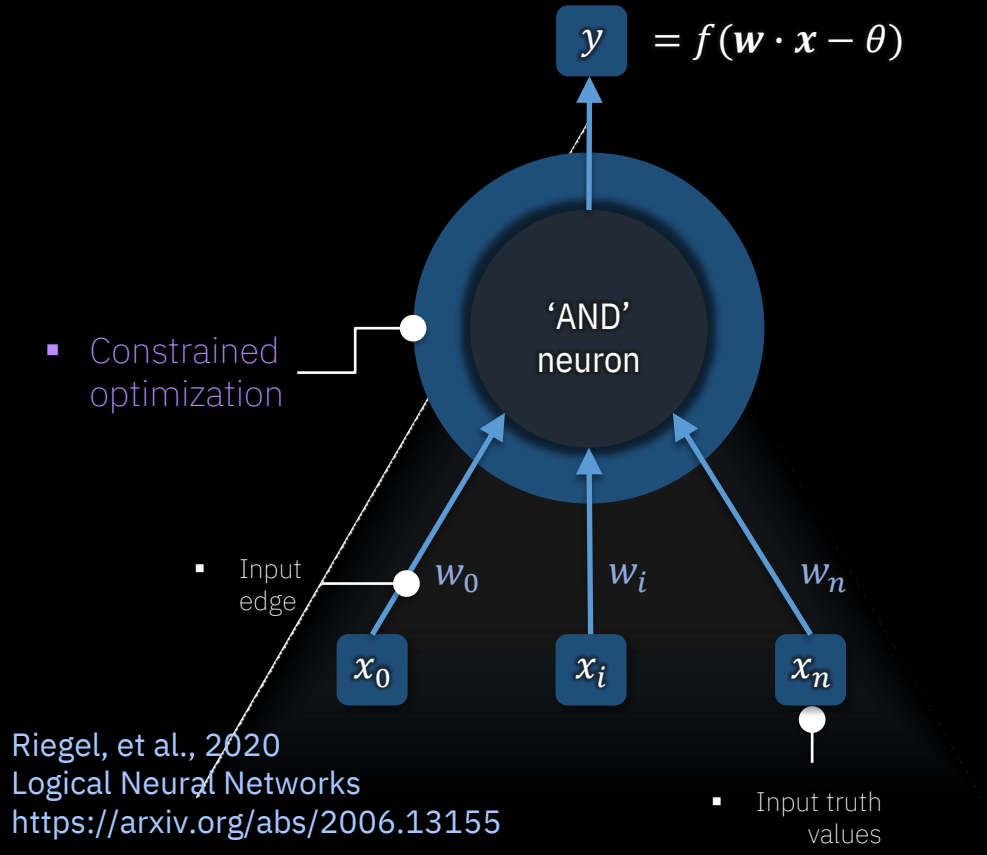
- Threshold of truth parameter $0.5 < \alpha \leq 1$:
  - Any $p \geq \alpha$ is "true," $p \leq 1-\alpha$ is "false"
- Now the weight region for classical 'AND' is:

$$\sum_i w_i \alpha - \theta \geq \alpha$$

$$\forall i, \sum_j w_j - w_i \alpha - \theta \leq 1 - \alpha$$

- *Activation functions obeying LNN's constraints behave as classical logic gates for classical inputs (theorem)*

4b. *Constrained* differentiable neuron (LNN) as logic gate



Riegel, et al., 2020
Logical Neural Networks
https://arxiv.org/abs/2006.13155

- Provide *slack* parameters $s \geq 0$ that govern the degree of adherence to classical behavior
  - Normal (unconstrained) neural networks are a special case where the slacks are large
  - Allows the idea of subsymbolic sub-network where, say, only the output node acts as a truth value
  - Slacks s_i allow w_i to shrink, thus can provide pruning of unnecessary inputs: Penalty on $s_i \cdot w_i$ encourages either to equal 0
- Now the weight region for classical 'AND' is:

$$\sum_i w_i \alpha - \theta \geq \alpha$$

$$\forall i, \sum_j w_j - w_i \alpha - \theta \leq 1 - \alpha + s_i$$

- But this does not yet address semantics of (non-classical) values between α and $1 - \alpha$

5a. Neuron (LNN) as *real-valued logic* gate

The most common real-valued logics:

Logic	T-norm (AND) $a \otimes b$	T-conorm (OR) $a \oplus b$	Residuum (IMPLIES) $a \to b$
Gödel	$\min\{a, b\}$	$\max\{a, b\}$	b if $a < b$ else 1
Product	$a \cdot b$	$a + b - a \cdot b$	$\frac{b}{a}$ if $a < b$ else 1
Łukasiewicz	$\max\{0, a + b - 1\}$	$\min\{1, a + b\}$	$\min\{1, 1 - a + b\}$

Łukasiewicz, 1920
Zadeh, 1965
Hájek, 1998

- Since 1920, multiple rigorous real-valued logics (where truth values $0 \le x, y \le 1$) have been studied mathematically and used
  - A.k.a. *many-valued*, *infinite-valued*, or *fuzzy* logics
  - R-logics (Hájek): IMPLIES/NOT via the residuum
  - S-logics (Zadeh): IMPLIES/NOT via $(1 - a) \oplus b$
- All behave as classical logic for the special case of 0/1 extremes, but differ for in-between values
- Can capture probabilities (more on this later)

5b. Neuron (LNN) as *real-valued logic* gate

Example: Łukasiewicz logic

Conjunction

$$p \otimes q = \max\{0, p + q - 1\}$$

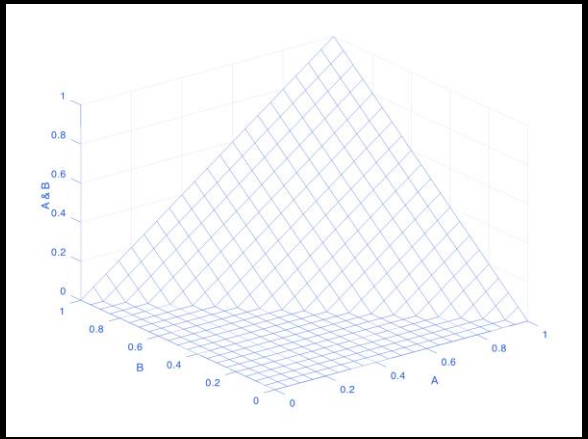
Disjunction

$$p \oplus q = 1 - \big((1 - \mathrm{p}) \otimes (1 - q)\big) = \min\{1, p + q\}$$

Implication

$$p \to q = (1 - p) \otimes q = \min\{1, 1 - p + q\}$$

- Implication actually defined according to the residuum, specifically: $p \to q = \arg\max_x \{q \geq p \otimes x\}$
  i.e. such that *modus ponens* is just AND



- *Note that this happens to be the same as the ReLU activation function!*
- But it doesn't allow the use of *weighted inputs*

Łukasiewicz, 1920
Hajek, 1998

5c. Neuron (LNN) as *real-valued logic* gate

New logic: Weighted Łukasiewicz logic

Conjunction

$$
\begin{aligned}
{}^{\beta}(p^{\otimes w_p} \otimes q^{\otimes w_q}) &= \max\{0, \min\{1, \beta - w_p(1-p) - w_q(1-q)\}\} \\
&= f(\boldsymbol{w} \cdot \boldsymbol{x} - \theta) \quad \text{for} \quad \theta = \textstyle\sum \boldsymbol{w} - \beta
\end{aligned}
$$

Disjunction

$$
\begin{aligned}
{}^{\beta}(p^{\oplus w_p} \oplus q^{\oplus w_q}) &= 1 - {}^{\beta}((1-p)^{\otimes w_p} \otimes (1-q)^{\otimes w_q}) \\
&= \max\{0, \min\{1, 1 - \beta + w_p p + w_q q\}\} \\
&= f(\boldsymbol{w} \cdot \boldsymbol{x} - \theta) \quad \text{for} \quad \theta = \beta - 1
\end{aligned}
$$

Implication

$$
\begin{aligned}
{}^{\beta}(p^{\otimes w_p} \to q^{\oplus w_q}) &= {}^{\beta}((1-p)^{\oplus w_p} \oplus q^{\oplus w_q}) \\
&= \max\{0, \min\{1, 1 - \beta + w_p(1-p) + w_q q\}\}
\end{aligned}
$$

Amato, di Nola, and Gerla, 2013
Riegel, et al., 2020

- Properties:
  - Weights w express importance
  - Bias β establishes the operation
  - All $w_i = \beta = 1$ gives unweighted case
  - All operations are continuous
- Upholds many classical tautologies:
  - Associativity (when $\beta \leq \min\{1, w_i\}$)
  - $\neg p = 1 - p, \qquad \neg\neg p = p$
  - $p \to q = \neg p \oplus q, \qquad$ De Morgan laws
  - *Modus ponens* is ${}^{\beta/w_q}(p^{\otimes w_p/w_q} \otimes (p \to q)^{\otimes 1/w_q})$
- *Now we have rigorous logical semantics for all input/output values*
  - Note that LNN can use similarly weighted versions of any of the aforementioned real-valued logics

6a. Neural network *inference as logical reasoning*

Inference rules for classical logic:

$$
\begin{array}{rrcll}
p, & p \to q & \vdash & q & \textit{modus ponens} \\
\neg q, & p \to q & \vdash & \neg p & \textit{modus tollens} \\
\neg(p \to q) & & \vdash & p & \\
\neg(p \to q) & & \vdash & \neg q & \\
p \wedge q & & \vdash & p & \textit{conjunction elimination} \\
p, & \neg(p \wedge q) & \vdash & \neg q & \textit{modus ponendo tollens} \\
\neg p, & p \vee q & \vdash & q & \textit{disjunctive syllogism} \\
\neg(p \vee q) & & \vdash & \neg p &
\end{array}
$$

Boole, 1854 (mathematical logic)
Gödel, 1929 (FOL soundness and completeness)
Hájek, 1998 (t-norm fuzzy logics)

- Steps that allow for the correct determination (*entailment*) of a truth value given other truth values
  - Exact form is dependent on logic
- There are sound and complete deductive systems for classical first-order logic (1929)
  - A logical system is sound if and only if the inference rules of the system admit only valid formulas
  - A logical system is complete if and only if all valid formula can be derived from the axioms and the inference rules
- Variants of this formalism have also been shown for some real-valued logics
  - Could provide a rigorous formalization of abduction

6b. Neural network *inference as logical reasoning*

Inference rules for real-valued logic:

Upward

$$L_{p\oplus q} = {}^{\beta}(L_p^{\oplus w_p} \oplus L_q^{\oplus w_q}) \qquad U_{p\oplus q} = {}^{\beta}(U_p^{\oplus w_p} \oplus U_q^{\oplus w_q})$$

Downward upper bounds

$$U_q \leq \begin{cases} {}^{\beta/w_q}((1 - L_p)^{\otimes w_p/w_q} \otimes U_{p\oplus q}^{\otimes 1/w_q}), & U_{p\oplus q} < 1 \\ 1 & \text{otherwise} \end{cases}$$

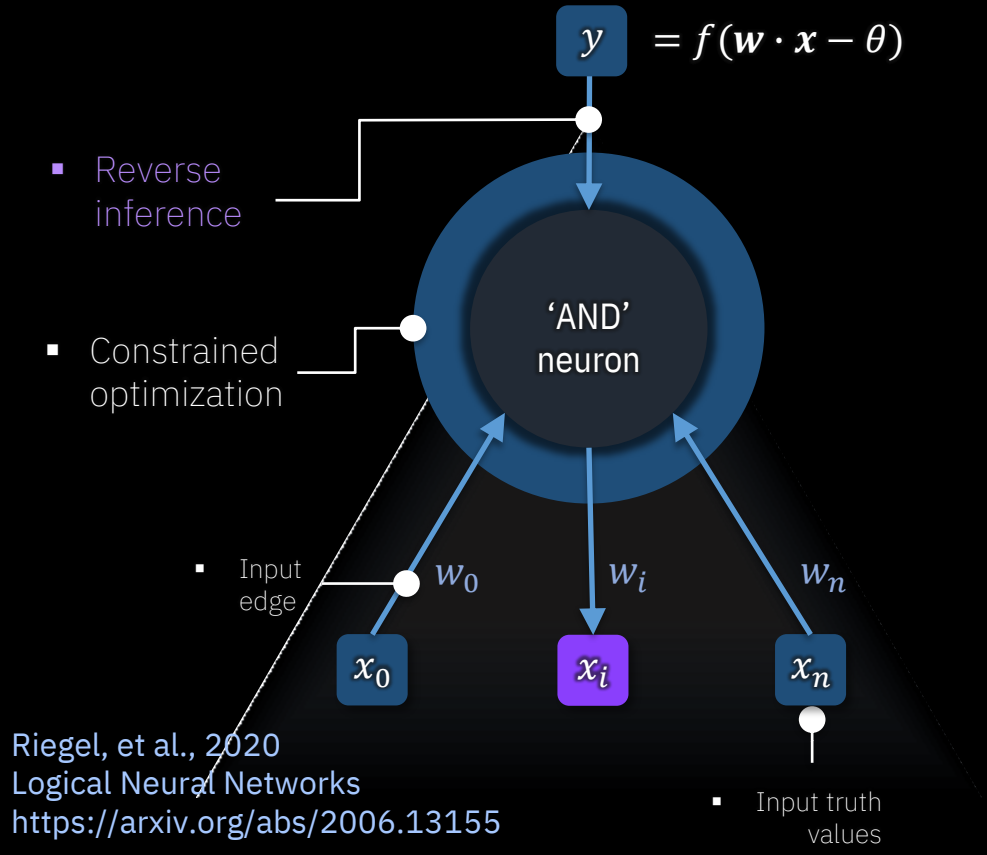
Downward lower bounds

$$L_q \geq \begin{cases} {}^{\beta/w_q}((1 - U_p)^{\otimes w_p/w_q} \otimes L_{p\oplus q}^{\otimes 1/w_q}), & L_{p\oplus q} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Fagin, Riegel, and Gray, 2020
Foundations of Reasoning with Uncertainty via Real-valued Logic
https://arxiv.org/abs/2008.02429

- We showed for the first time that inference in *all* real-valued logics (including weighted versions) can be sound and (strongly) complete
  - There exists a sound/complete axiomatization that works for *any* choice of connective functions
  - For Łukasiewicz and Gödel logic, showed an MILP-based decision procedure to check if $\gamma_1, \ldots, \gamma_n \vdash \phi$ when ϕ and each γ_i are associated with a disjoint union of intervals of candidate truth values
- But: we would like a cheaper message-passing procedure that can use current infrastructure, e.g. Pytorch
  - Note that when viewed as neural network propagations, the necessary inference rules cannot be done using only forward ("upward") inference

6c. Neural network *inference as logical reasoning*



Riegel, et al., 2020
Logical Neural Networks
https://arxiv.org/abs/2006.13155

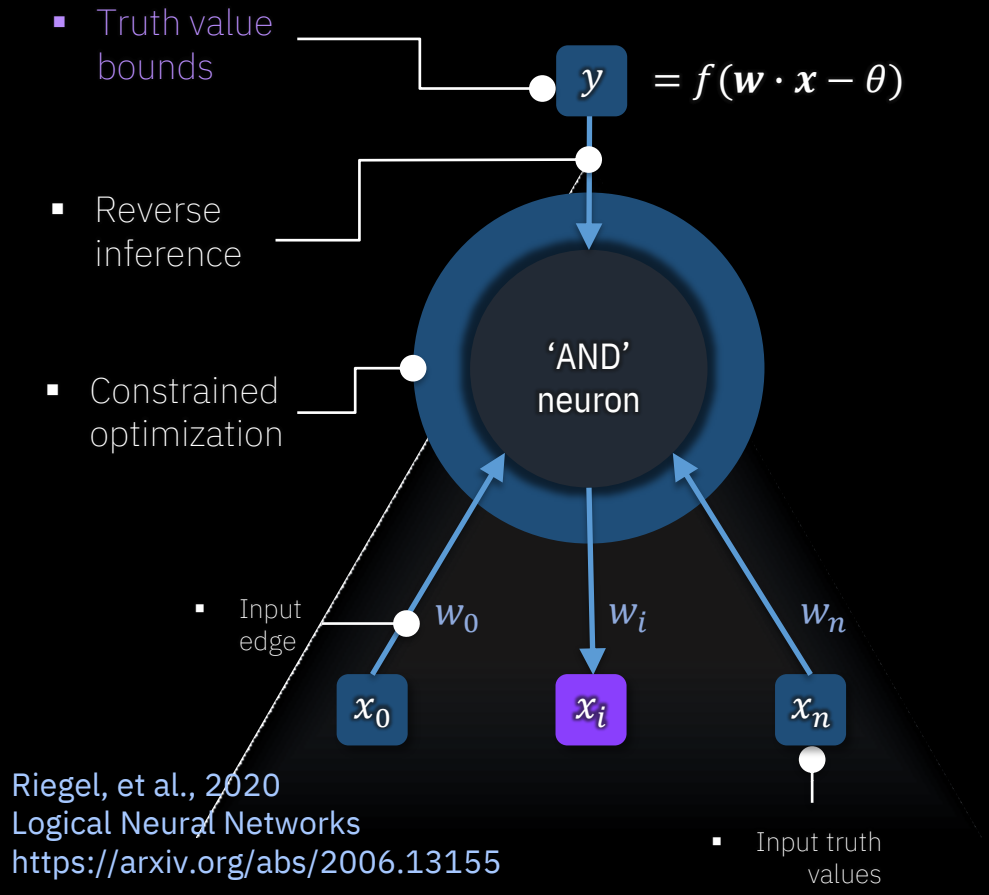
- *Allow reverse ("downward") inference to compute inferences such as modus ponens*

$$x_i = \frac{f^{-1}(y) + \theta - \boldsymbol{w}_{\setminus i} \cdot \boldsymbol{x}_{\setminus i}}{\boldsymbol{w}_i}$$

- Message-passing style inference via Upward–Downward algorithm:
  - Provably converges in finite time
  - Can be shown to be sound but ~~not~~ complete because dependencies between truth values ~~are not tracked~~ can be modeled with an extension to the NN

1. Initialize neurons with observed truth values
2. While not converged:
   a. Upward pass
   b. Downward pass
   c. Aggregate truth values at propositions/predicates via (optionally smooth) min/max
3. Inspect neurons representing predictions/queries

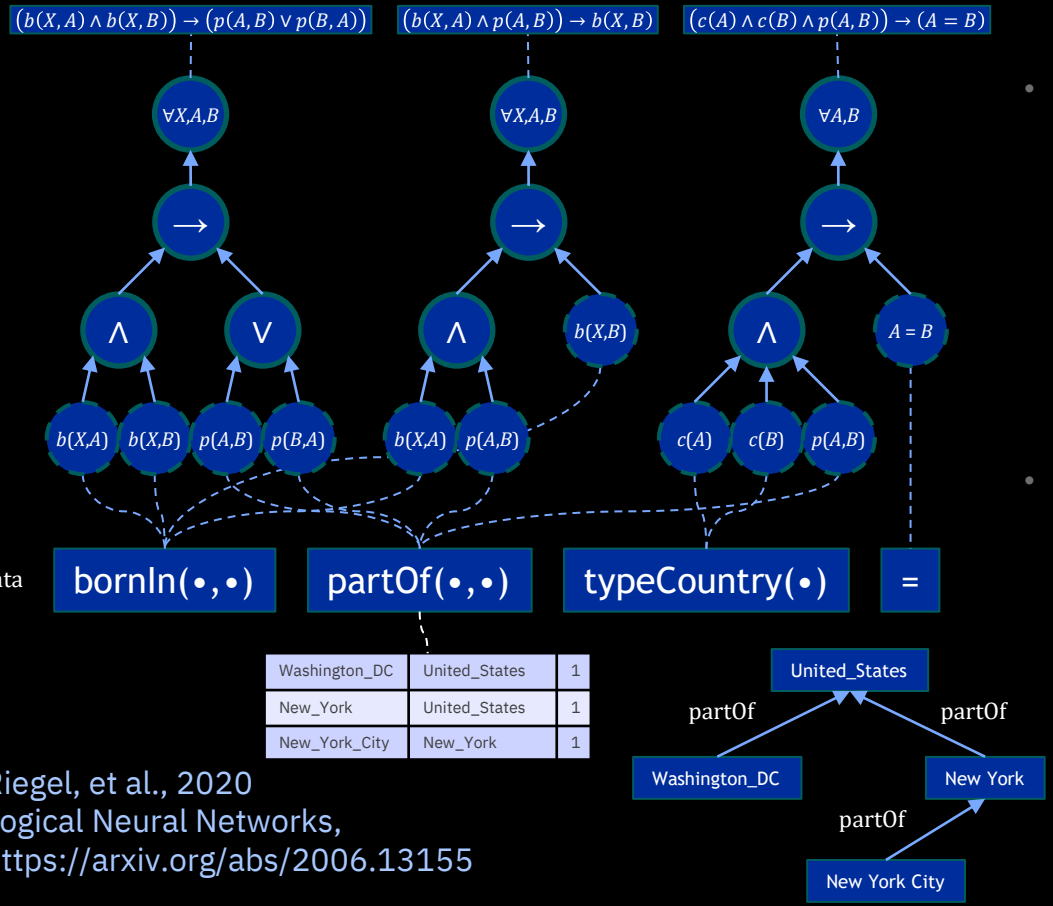
6d. Neural network *inference as logical reasoning*



Riegel, et al., 2020
Logical Neural Networks
https://arxiv.org/abs/2006.13155

- Note that for some activation functions, this value may not be unique
  - e.g. due to flat regions of ReLU
  - But we can maintain lower and upper uncertainty bounds l_i and $u_i \in [0,1]$ on the truth value of x_i
- This allows for the explicit representation of ignorance ("don't know"), thus permitting the open-world assumption
  - In addition, it serves as an explicit representation of contradiction whenever $l_i > u_i$
- For a certain choice of activation function, truth value bounds are probability bounds
  - Uses hybrid Łukasiewicz/Gödel activation function implementing the Fréchet inequalities
  - Bounds make no assumptions about independence and are tight for acyclic formula graphs

7a. Data and learning



Riegel, et al., 2020
Logical Neural Networks,
https://arxiv.org/abs/2006.13155

- Data
  - Grounded representation; natively relational
  - Predicates embodied as tables or, equivalently, tensors or replicated neurons for each grounding
  - Knowledge graph triples = cells in usual example/feature dataset table
  - Operators perform joins; quantifiers reductions
- Inputs and outputs
  - Any-task learning: generalization of supervised learning: predict any variable(s) given settings of any other variables(s)
  - Training examples: worlds 1…M: values $\{X_i\}$, $\{Y_j\}$; each world may have different variables set
  - Ignorance/unobservability: generalization of missing data handling: values are of the general form $\{l, u\}$

7b. Data and learning

Learning problem:

$$\min_{B,W} \sum_{j \in M} E(X_j, Y_j | B, W) + \overbrace{\sum_{k \in N} \sum_{r \in P_k} \ell\left(l_r(X_j|B,W), u_r(X_j|B,W)\right)}^{\text{Contradiction loss}}$$

$$\text{s.t.} \quad \forall k \in N, \forall i \in I_k, \quad s_{ik} + \alpha \cdot w_{ik} - \beta_k + 1 \geq \alpha, \quad w_{ik} \geq 0$$

$$\forall k \in N, \quad \sum_{i \in I_k} (1-\alpha) \cdot w_{ik} - \beta_k + 1 \leq 1 - \alpha, \quad \beta_k \geq 0$$

Example error function *E* with ℓ = hinge loss:

$$E(X, Y | B, W) = \sum_{(r, l^*, u^*) \in Y} \left(\ell\left(l^*, l_r(X|B,W)\right) + \ell\left(u_r(X|B,W), u^*\right)\right)$$

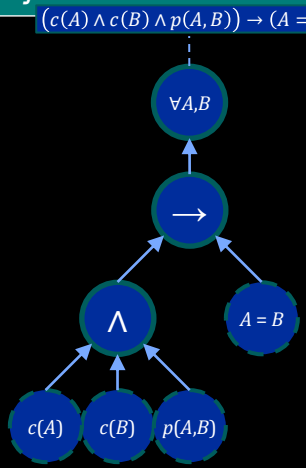
$$\ell(l, u) = (\max\{0, l - u\})^2$$

Riegel, et al., 2020: Frank-Wolfe for LNN
Sen, et al., 2020: Double description method for LNN *(and ILP, i.e. adding neurons)*
Lu, et al., 2020: Inexact ADMM for LNN *(also distributed)*

- Versatile general loss function
  - Prediction error *E*: sum error on *Y* variables over all worlds 1…*M*
    - E.g. hinge loss: try to make predicted truth value bounds l_r and u_r for each grounding *r* in *Y* at least as tight as target truth value bounds *l** and *u**
  - Contradiction loss ℓ: sum amount of contradiction (degree to which lower bounds cross upper bounds) over all neurons 1…*N*: maintain logical consistency of all knowledge
    - P_k is the predicate at neuron *k*, *r* is every (known) grounding in P_k, and I_k is the *k*th neuron's inputs
- Gradient-based optimization
  - All operations are continuous and, with smoothing, differentiable; implemented in PyTorch
  - Constrained optimization; alternatively, activation functions may be tailored to avoid the need for constraints (see paper)

8. *Equivalence* between neural networks and symbolic logic

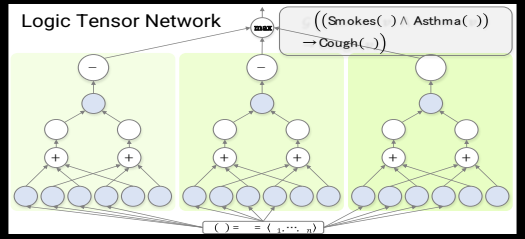
standard (deep, recurrent) neural net + *constraints*	=	set of (*weighted real-valued logic*) logic statements
standard NN forward inference + *reverse inference*	=	(*weighted real-valued logic*) logical inference

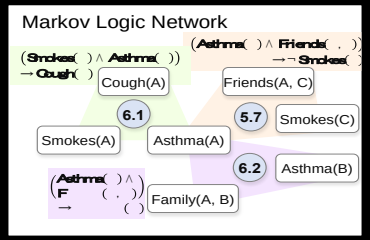


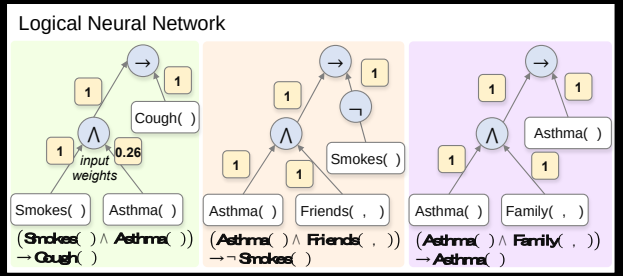
Riegel, et al., 2020
Logical Neural Networks,
https://arxiv.org/abs/2006.13155

- Neural net and logic statements are just two renderings of the same model ("particle-wave duality"), not two models that communicate
  - Classical logic is precisely a special case: precise deduction, e.g. math, code; planning
    - Not: Approximation of logical behavior in the limit of infinite training data/samples, etc.
  - Standard neural networks are precisely a special case: SotA prediction, object detection, etc.
    - Not: Simpler NN/ML models not used in practice
- Allows full spectrum in-between
  - Can have 'upper' symbolic network and 'lower' subsymbolic network from raw inputs to first symbols; or freely mix symbolic and subsymbolic neurons
  - Can freely mix precise and imprecise logic statements (noisy rules, partial/inconsistent domain knowledge)

Comparison to other common neuro-symbolic ideas







Interpretability and generalizabilty

Logic statements → points in embedding

- Distributed/entangled: no node has a stand-alone meaning; numbers (weights in high-d space) have non-obvious semantics; structure (layers, width, connectivity) has non-obvious interpretation
- Inference (neural net inference) has no obvious step-by-step explanation

Logic statements → cliques of terms in MRF

- Disentangled, but not compositional (e.g. no re-use of Smokes(A) ^ Asthma(A)); no representation of logic connectives in MRF; numbers (potentials between 0 and ∞) hard to interpret (e.g. 6.2)
- Inference (sampling) has no obvious step-by-step explanation

Logic statements = syntax trees of neurons

- Disentangled, and 1-1 with each piece of logic statement: each neuron has a meaning: either predicate or logical connective; compositional/modular (i.e. language-like): sub-expressions are reused, rather than repeated; numbers have clear semantics: activations = real-valued truth values, can represent probabilities if desired, weights = relative importance in logical connectives
- Inference is deterministically repeatable and has step-by-step explanation: sequence of logical inferences

Problem-solving power

Learning: approximate satisfiability via gradient-based training; Inference: NN

- Precise logical inference is not a special case, except in the limit of infinite training samples
- Standard NN does not appear to be a special case, but combinable with standard NN

Learning: approximate satisfiability via MCMC; Inference: MRF

- Precise logical inference is not a special case, except in the limit of infinite weights (but then you're not learning)
- Standard NN is not a special case of MRF in general, but perhaps combinable with standard NN

Learning: standard loss + contradiction term, gradient-based; Inference: logical inference

- Precise logical inference is a special case; standard NN (deep, recurrent) is a special case; most common type of benchmark: link pred w/ imperfect domain knowledge:

Method	Precision	Recall	Accuracy
Hidden cough model: $(\texttt{Smokes}(x) \vee \neg\texttt{Exercises}(x)) \wedge (\texttt{Asthma}(x) \vee \texttt{Allergic}(x)) \rightarrow \texttt{Cough}(x)$			
LTN	34.6 ±20	39.8 ±26	66.7 ±11
MLN	81.0 ±11	73.2 ±30	90.6 ±3.5
LNN	**85.1** ±11	**76.3** ±23	**91** ±3.7

Use case: Knowledge base question answering (KBQA)

DBpedia

http://depedia.org/resource/John_McCarthy

DBpedia · Browse using · Formats · Faceted Browser · Sparql Endpoint

About: John McCarthy (computer scientist)

An Entity of Type : scientist, from Named Graph : http://dbpedia.org, within Data Space : dbpedia.org

Property	Value
dbo:almaMater	• dbr:California_Institute_of_Technology • dbr:Princeton_University
dbo:award	• dbr:Turing_Award
dbo:birthDate	• 1927-09-04 (xsd:date) • 1927-9-4
dbo:birthPlace	• dbr:Boston,_Massachusetts • dbr:Boston
dbo:deathDate	• 2011-10-24 (xsd:date)
dbo:deathPlace	• dbr:Stanford,_California
dbo:knownFor	• dbr:Artificial_intelligence • dbr:Lisp_(programming_language) • dbr:Circumscription_(logic) • dbr:Situation_calculus

Knowledge base triples

Wikipedia

https://en.wikipedia.org/wiki/John_McCarthy

John McCarthy initially completed graduate studies at **California Institute of Technology** before moving to **Princeton University**.

McCarthy received many accolades and honors, such as the 1971 **Turing Award** for his contributions to the topic of AI.

John McCarthy, who coined the term "**artificial intelligence**", was born in **Boston** on **September 4, 1927**.

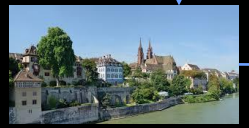
- If TODs are present, they are utilized for ranking in sentence selection.
- Any set of documents (e.g., Common Crawl, Web Search) can be used as the document corpus.

Was Roger Federer born in United States?

Roger Federer

Birthplace

Basel



Part Of

Switzerland

Type

USA

Type

COUNTRY

KBQA: Why it challenges default AI (end-to-end deep learning)

QALD (2016-10)
- 408 questions train and 150 test

LC-QuAD (2016-04)
- 4000 train and 1000 test
- Template based questions

Question Type/Reasoning	Example	Supported
Simple	Who is the mayor of Paris	✓
Multi-relational	Give me all actors starring in movies directed by William Shatner.	✓
Count-based	How many theories did Albert Einstein come up with?	✓
Superlative	What is the highest mountain in Italy?	✓
Comparative	Does Breaking Bad have more episodes than Game of Thrones?	
Geographic	Was Natalie Portman born in the United States?	✓
Temporal	When will start [*sic*] the final match of the football world cup 2018?	

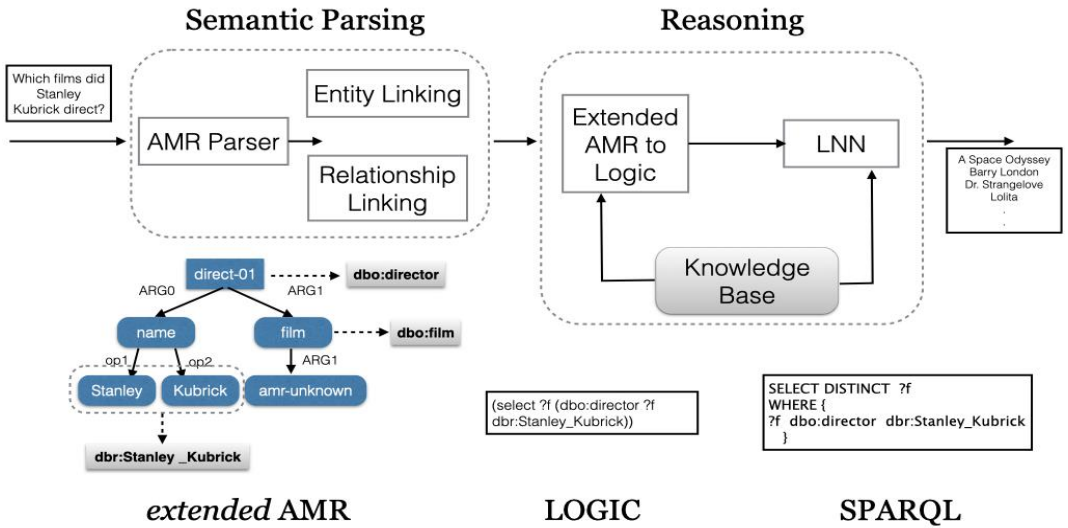
Table 2: Question types supported by DTQA , with examples from QALD

- Going beyond canned answers
  - End-to-end deep learning (DL) selects from pre-canned existing sentences: can't extrapolate to answers that don't appear in training data at all
  - Existing systems generally are demonstrated on a single dataset
  - No reasoning or understanding: can't answer questions that require non-trivial reasoning beyond surface patterns
- Small training sets
  - Space of all possible sentences is combinatorial
  - Unclear whether even end-to-end DL training on all of the sentences on the Internet is enough for 'understanding' to emerge
- No ability to explain answer
  - End-to-end DL would rely on ability to explain pattern matching

KBQA: an approach via *understanding*

Instead of trying to map input (question) words to output (answer) words: **first map question words to abstract concepts (logic), then use reasoning** to answer question

- Intermediate representations: AMR, SPARQL
- Reusable, plug-and-play SotA/near-SotA components



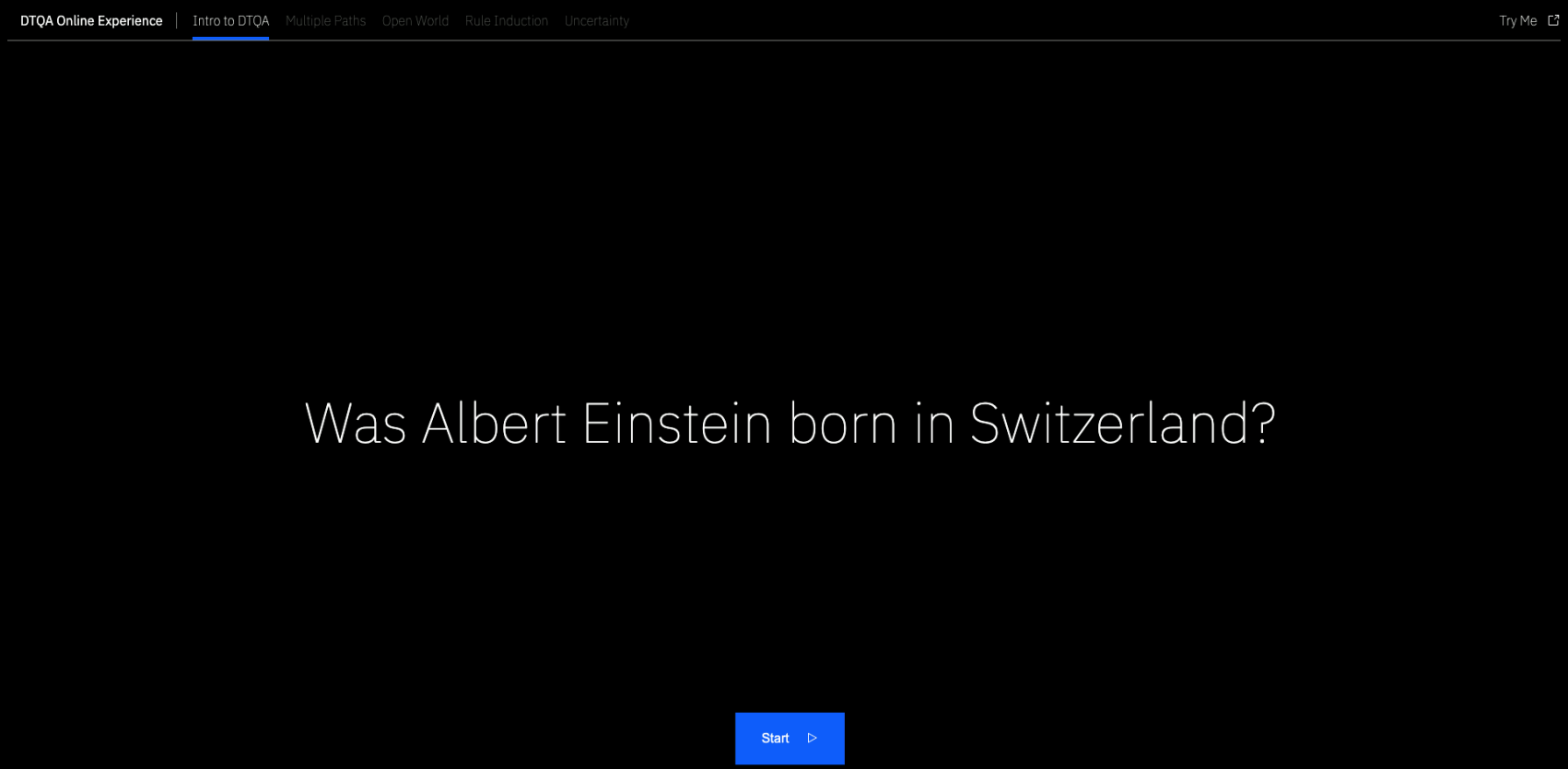
Kapanipathi, et al., 2020 (NSQA system: SotA KBQA)
Asudillo, et al., 2020 (SotA AMR parsing)
Abdelaziz, et al., 2020 (SotA relation linking)

	Dataset	P	R	F	F1 QALD
WDAqua*	QALD-9	26.09	26.7	24.99	28.87
gAnswer*	QALD-9	29.34	32.68	29.81	42.96
NSQA	QALD-9	31.41	32.16	**30.88**	**45.33**
WDAqua*	LC-QuAD	22.00	38.00	28.00	–
QAMP*	LC-QuAD	25.00	50.00	33.00	–
NSQA	LC-QuAD	38.19	40.39	**38.29**	–

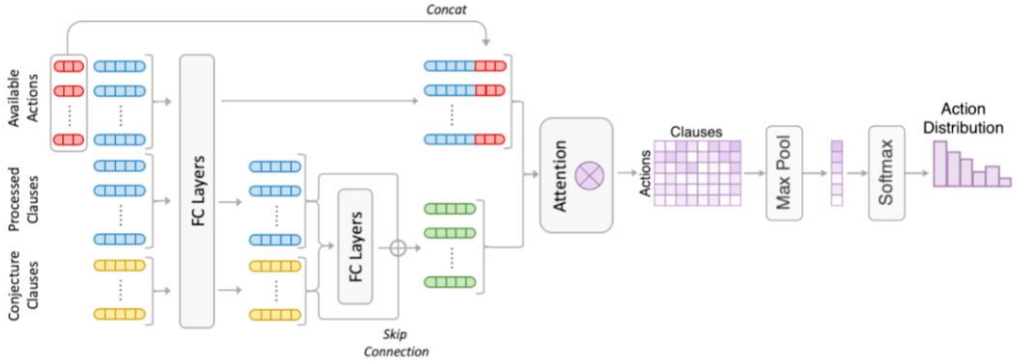
Table 1: NSQA performance on QALD-9 and LC-QuAD *Numbers from respective papers

- More generalizability
  - SotA on more than one QA dataset
  - Can extrapolate to unseen situations via transferable knowledge summarizing many examples; doesn't rely exclusively on training set
- More explainability
  - Provides which knowledge and reasoning steps relied on; can say "don't know" via truth bounds
- First neuro-symbolic win?
  - Over current default AI on competed benchmark

Making the model & inference process human-understandable



Learning to reason



- Problem Setting:

  Given a set of axioms

  Given a theorem or a conjecture to prove

  Search for a proof of the theorem/conjecture

- Approach:

  Deep reinforcement learning approach to learn proof guidance strategies from scratch

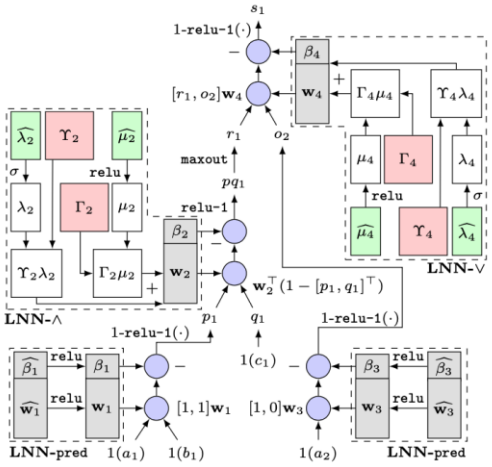
  Novel neural representation of the state of a theorem-prover (**logic embedding**)

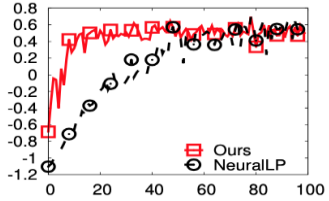
  Novel attention-based policy

- Use learning to tame worst-case complexity
  - Reasoning in FOL or HOL is very hard in worst case (undecidable)
  - Infinite number of actions (i.e., inferred facts)
- SotA theorem-proving performance
  - Outperformed existing all learning-based approaches (15% more theorems) and some traditional heuristics-based reasoners
  - Recently surpassed the mature E-prover on the hard Mizar-MPTP2078 subset by 2%

Abdelaziz, et al., A Deep Reinforcement Learning Approach to First-Order Logic Theorem Proving, AAAI 2021

Logical rule induction (ILP)





Gridworld: Rewards vs. Training Grids

		NTP-λ	NeuralLP	Ours
Kinship	Hits@10	0.878	0.912	**0.932**
	MRR	0.793	0.619	**0.926**
UMLS	Hits@10	**1.000**	0.962	0.983
	MRR	0.912	0.778	**0.962**
WN18RR	Hits@10	-	0.657	**0.731**
	MRR	-	**0.463**	0.337
FB15K-237	Hits@10	-	0.348	**0.604**
	MRR	-	0.227	**0.327**
NELL-995	Hits@10	-	-	**0.752**
	MRR	-	-	**0.715**

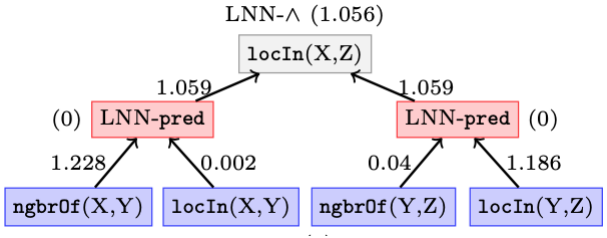
KBC Results

$$\texttt{locIn}(X,Z) \leftarrow \texttt{ngbrOf}(X,Y) \wedge \texttt{locIn}(Y,Z)$$

Ground truth rule (Countries-S2)

$$0.001 : \texttt{locIn}(X,Z) \leftarrow \texttt{inv\_ngbrOf}(Y,Z) \wedge \texttt{locIn}(X,Y)$$
$$0.002 : \texttt{locIn}(X,Z) \leftarrow \texttt{ngbrOf}(Y,Z) \wedge \texttt{locIn}(X,Y)$$
$$0.006 : \texttt{locIn}(X,Z) \leftarrow \texttt{inv\_locIn}(Y,Z) \wedge \texttt{locIn}(X,Y)$$
$$0.050 : \texttt{locIn}(X,Z) \leftarrow \texttt{locIn}(X,Z)$$
$$0.922 : \texttt{locIn}(X,Z) \leftarrow \texttt{locIn}(Y,Z) \wedge \texttt{locIn}(X,Y)$$

Rules from other neuro-symbolic baseline methods (dILP, NTP, NeuralILP, NLM)



Learned LNN rule

- Joint end-to-end learning of rules and operators (adds neurons)
  - Flexible rule templates, backprop + double description
- High-quality rules learned from small, noisy data
  - Weights allow higher accuracy than typical representations; qualitatively closer to ground truth, simpler

Sen, et al., Neuro-Symbolic Inductive Logic Programming with Logical Neural Networks, under submission, 2020

Optimization/learning

$$\mathbb{P}1: \min_{\alpha,\beta,W} \left(\underbrace{\sum_k \max(0, L_k(\alpha,\beta,w) - U_k(\alpha,\beta,w))}_{\text{contradication}} \right) + \underbrace{f(\alpha,\beta,w)}_{\text{loss/mismatch}}$$

$$\text{s.t.} \quad \frac{1}{2} \le \alpha \le 1, \quad w_{ik} \ge 0, i \in \mathcal{I}_k, \forall k \in N$$

$$w_{ik}\alpha - \beta_k + 1 \ge \alpha, \text{(Conjunction)}$$

$$\sum_{i \in \mathcal{I}_k} w_{ik}(1-\alpha) - \beta_k + 1 \le 1 - \alpha, \forall k \in N$$

Training target mismatch

$$f(\alpha,\beta,w) = \sum_i \tau_i \left(\left(L_i^{(g)} - L_i(\alpha,\beta,w) \right)^2 + \left(U_i^{(g)} - U_i(\alpha,\beta,w) \right)^2 \right)$$

w, α, β are optimziation variables
i, k: indices of neuron
w: weights; α: threshold; β: bias

- Non-convex objective
- *L* and *U* non-smooth
- Constraints contain nonlinear coupling: *α* now learnable (optionally per-neuron)

Optimization/learning

Problem $\mathbb{P}1$ can be reformulated as:

$$\min_{\mathbf{x}\in\mathbb{R}^{d_1},\mathbf{y}\in\mathbb{R}^{d_2}} \quad f(\mathbf{x},\mathbf{y})+h(\mathbf{x},\mathbf{y}) \quad \text{s.t.} \quad \mathbf{c}(\mathbf{x},\mathbf{y})\le 0$$

$\mathbf{x}=[\beta,w]^T$, $\mathbf{y}=\alpha$
$h(\cdot)$: nonsmooth (contradiction) and convex,
$\mathbf{c}(\cdot):\mathbb{R}^{d_1+d_2}\to\mathbb{R}^m$: nonconvex and smooth (variable coupling between w_{ik} and α)
$f(\cdot)$ is nonconvex (loss function)

(improved) Inexact alternating direction method of multipliers (iADMM):
Augmented Lagrangian:

$$\mathcal{L}_\mu(\mathbf{x},\mathbf{y},\mathbf{z}) := f(\mathbf{x},\mathbf{y})+\frac{\mu}{2}\left\|\left[c(\mathbf{x},\mathbf{y})+\frac{\mathbf{z}}{\mu}\right]_+\right\|^2-\frac{1}{2\mu}\|\lambda\|^2 \qquad (1)$$

Find $\mathbf{x}^{(t+1)}$ be an approximate solution of $\min_{\mathbf{x}}\mathcal{L}_\mu(\mathbf{x},\mathbf{y}^{(t)},\mathbf{z}^{(t)})+\rho^{(t)}\|\mathbf{x}-\mathbf{x}^{(t)}\|$ by accelerated proximal point method

Find $\mathbf{y}^{(t+1)}$ be an approximate solution of $\min_{\mathbf{y}}\mathcal{L}_\mu(\mathbf{x}^{(t+1)},\mathbf{y},\mathbf{z}^{(t)})+\rho^{(t)}\|\mathbf{y}-\mathbf{y}^{(t)}\|$ by accelerated proximal point method

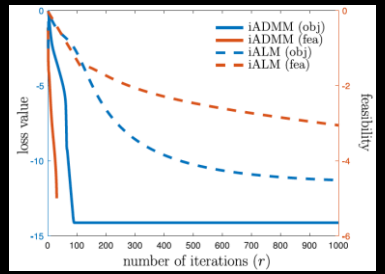
Update $\mathbf{z}^{(t+1)}=[\mathbf{z}^{(t)}+\omega^{(t)}\mathbf{c}(\mathbf{x}^{(t+1)},\mathbf{y}^{(t+1)})]_+$

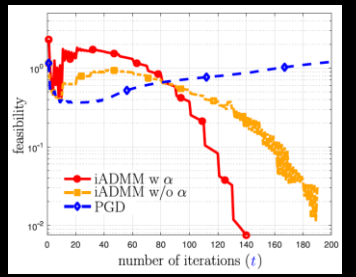
Theorem 1. When $\omega^{(t)}=\frac{\|\mathbf{c}(\mathbf{x}^1)\|}{(t+1)\log^2(t+2)}$, iADMM converges to the KKT points in a rate of $\mathcal{O}(\epsilon^{-3})$ for nonconvex objective and nonconvex constraints problems.

Method	Type	Problem	Constraint	Complexity
Frank-Wolfe	penalty	$\min_{\mathbf{x}} f(\mathbf{x}), \text{s.t.} c(\mathbf{x})\le 0$	convex	$\mathcal{O}(\epsilon^{-2})$
QP-AIPP	penalty	$\min_{\mathbf{x}} f(\mathbf{x})+h(\mathbf{x}), \text{s.t.}\mathbf{Ax}=\mathbf{b}$	convex	$\mathcal{O}(\epsilon^{-3})$
HiAPeM	hybrid	$\min_{\mathbf{x}} f(\mathbf{x})+h(\mathbf{x}), \text{s.t.}\mathbf{Ax}=\mathbf{b}, c(\mathbf{x})\le 0$	convex	$\mathcal{O}(\epsilon^{-5/2})$
iPPP	penlty	$\min_{\mathbf{x}} f(\mathbf{x})+h(\mathbf{x}), \text{s.t.}\mathbf{Ax}=\mathbf{b}, c(\mathbf{x})\le 0$	nonconvex	$\mathcal{O}(\epsilon^{-3})$
iALM	AL	$\min_{\mathbf{x}} f(\mathbf{x})+h(\mathbf{x}), \text{s.t.} c(\mathbf{x})=0$	nonconvex	$\mathcal{O}(\epsilon^{-3})$
iADMM (Our work)	AL	$\min_{\mathbf{x},\mathbf{y}} f(\mathbf{x},\mathbf{y})+h(\mathbf{x},\mathbf{y}), \text{s.t.} c(\mathbf{x},\mathbf{y})\le 0$	nonconvex	$\mathcal{O}(\epsilon^{-3})$

The major advantage of AL v.s. penalty: finite szie of dual variable in AL can guarantee the convergence (avoid ill-conditioning)

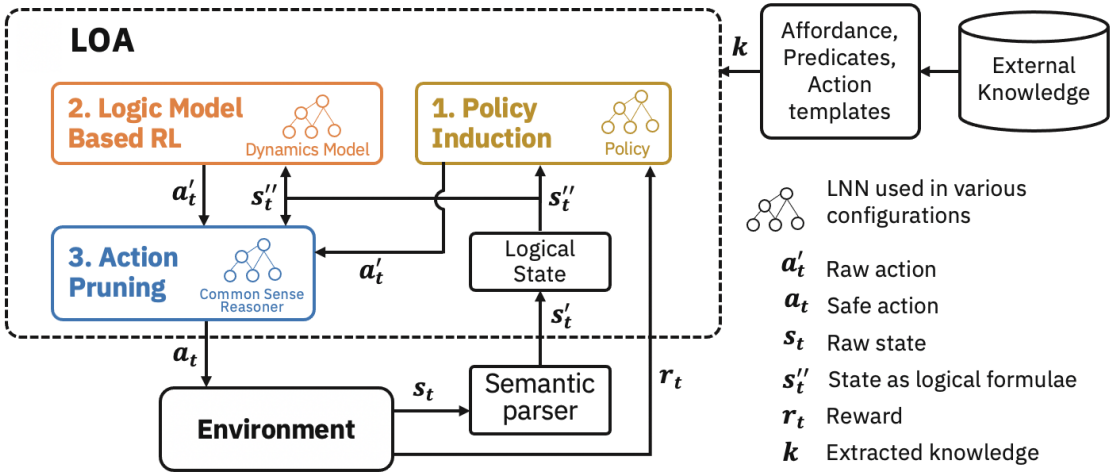
- SotA convergence rate
- Scalable with number of constraints
- Better empirical performance
- Can be made distributed





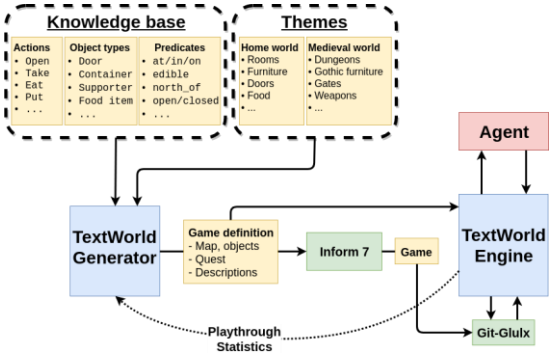
Lu, et al., "Training logical neural networks by primal-dual methods for neuro-symbolic reasoning", submitted 2020

Reinforcement learning



LNN used in various configurations

a'_t	Raw action
a_t	Safe action
s_t	Raw state
s''_t	State as logical formulae
r_t	Reward
k	Extracted knowledge

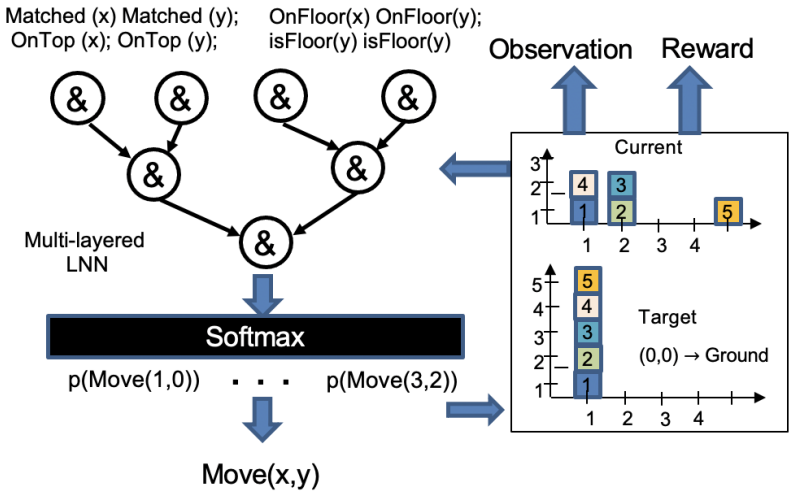


- Reinforcement learning
  - Generally massive number of trials needed
  - Generally uses no knowledge (‘model-free’)
- Goal: use knowledge to dramatically reduce number of trials needed

```
West of House
You are standing in an open field west of
 a white house, with a boarded front door.

There is a small mailbox here.

> open mailbox

Opening the small mailbox reveals a leaflet.
```

Policy induction via rule learning

$$G^* = argmax_G\ [E_\tau(R(s_t, \pi_G(a_t|s_t)))], s.t.\ \pi_G\ is\ LNN.$$



Learn interpretable rules for logical actions:

Move(x,y)
= OnTop (x) & OnTop(y) & Matched(y) & ¬Matched(x)

Moveable(x,y) IsMoveRequired(x,y)

& OnFloor(x) & ¬isFloor(x) & ¬isFloor(y) & ¬onFloor(y)

SourceFloorQuery(x) TargetFloorQuery(y)

- Learning rule-based policies
  - RL (expected reward maximization) with LNN constraints for interpretable policy
  - Currently working on small problems like Blocks Stacking with Double-Description optimization

Method (50 games)	Success Ratio	#Steps
Blocks (PILNN, N=3)	1.0	3.45
Zero-shot transfer (train N=3->4)	1.0	12.06
Zero-shot transfer (train N=3->5)	0.98	28.7
Zero-shot transfer (train N=3->6)	0.9	51.42

Kimura, et al., Reinforcement Learning with External Knowledge by using Logical Neural Networks, KbRL workshop at IJCAI 2020

AGI: Bengio-Marcus Desiderata

Desideratum	Symbolic AI (best of)	Statistical AI (best of)	MRF-based	Embedding-based	LNN
Neural nets can be a universal solvent (incl learning)		✓		✓	✓
Allows specialized sub-networks and specialized neurons		✓	✓	✓	✓
Meta-learning/multi-task		✓		✓	✓
Modular design	✓	✓	✓		✓
Can use prior/innate knowledge	✓		✓		✓
Capable of true reasoning	✓		✓	✓	✓
Variables	✓		✓	✓	✓
Symbol manipulation	✓				coming soon
Can use a generic kind of model	✓	✓	✓	✓	✓
Causality	✓	✓			coming soon-ish
'Agent view' / formulating a plan over multiple time scales	✓	✓			✓
Seamlessly blends system 1 (perception) and system 2 (reasoning), with learning throughout			✓	✓	✓
Can perform true natural language understanding, with ability to generate novel interpretations					✓
Can acquire knowledge via natural language					coming soon-ish
Can learn with less data & generalize to new domains easily					working on it!

Ongoing directions

Applied

- **Scaling** to massive KBs – MILP, HPC, typing, graph DB
- **QA/NLP –** incomplete KBs, temporal, narratives

Representation

- **Probabilities** – extend to handle enriched prob knowledge as in Bayes nets
- **Embeddings** – sub-symbolic emergence, imprecise concepts, intuition

Knowledge

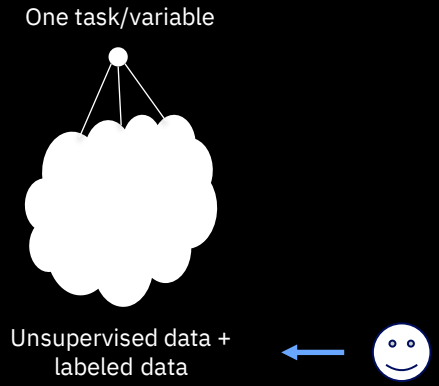
- **Logic** – lifting, higher-order logic, including temporal and spatial logic
- **Knowledge acquisition** – via semantic parsing

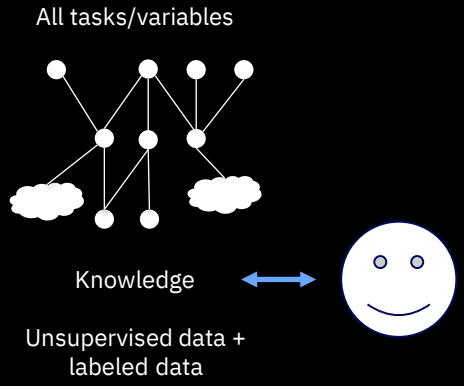
Learning

- **Reinforcement learning –** action pruning, RL+planning, causal RL
- **Compositional & multi-task learning** – take advantage of known structure

Seeking collaborators!

Philosophical shift: Humans+AI





Input/human role: Relies on largest number of labels possible

- One-time human input, relatively thought-free
- Try to be knowledge-free, i.e. always start from scratch/no assumptions (blank slate)

Output/what model does: 1 task (predict 1 variable)

- For new task, get new labels and train separate model

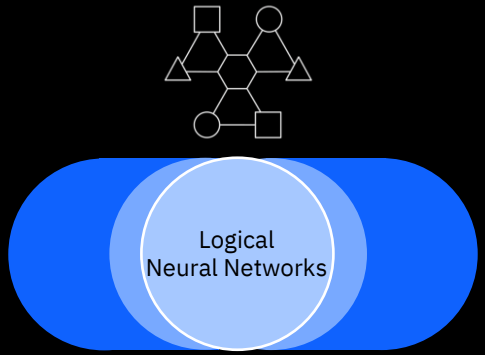
Input/human role: Augments data with domain/innate/common sense knowledge

- Humans oversee/adjust/control knowledge/model; reduces pure reliance on massive data
- Don’t need to start tabula rasa every time, keep building up knowledge model (lifelong)

Output/what model does: all possible tasks (predict any variable)

- Add to loss function: make all tasks work together
- Sub-models (areas of knowledge) are modular, shareable, reusable

Summary



Logical Neural Networks

A framework for neural nets with a 1-to-1 correspondence with a system of logical formulae, in which propagation is equivalent to logical inference

Key ideas:

- Learning: 1) constraints, 2) contradiction loss
- Inference: 3) bidirectional, 4) truth bounds

1. Single representation capable of all 3 kinds of reasoning: induction, deduction, abduction
   - Full power of classical logic as special case
   - Subsymbolic (standard) NN as special case/module
   - Reasoning w/ uncertainty, probabilities as special case
2. Entire model is human-readable, each step in decision-making has an explanation, sub-models are reusable
3. Rigorous theoretical foundation: semantics of real-valued logic

For more on this research program:

alexander.gray@ibm.com
http://ibm.biz/neuro-symbolic-ai