Logical Neural Networks

Kinshuk Vasisht[1]

[1] Student of M.Sc. Computer Science, Department of Computer Science, University of Delhi

Outline

Motivation[1]

- Modern Deep Learning structures are able to achieve accuracies surpassing human-level performance over narrowly-defined tasks.
- Such structures encode knowledge in complex networks of interconnected neurons. This complexity enforces researchers to think abstractly about the behaviour of the network. Further, lack of per-unit interpretability prevents generalization to broader tasks.
- Grounding the behaviour of a neuron in an understandable manner allows enhancing interpretability and explainability / justification of decisions.
- By integration of logic with neurons, such grounding can be obtained.

[1] Khan et al. 2020.

Introduction[2]

- Logical Neural Networks (LNNs) are a neuro-symbolic framework designed to simultaneously provide key properties of both neural networks (learning) and symbolic logic (reasoning).
- LNNs implement the Neuro=Symbolic framework, providing direct interpretability, utilization of rich domain knowledge realistically, and general problem-solving ability of a full theorem prover.
- The network is equivalent to a set of logical statements, whilst retaining the learning capability of a neural network.
- Equivalence is achieved by creation of a 1-1 correspondence between neurons and elements of logical formulae. Logical reasoning is modeled via recurrent neural computation of truth values in a bidirectional manner.

[2] Riegel et al. 2020.

Features[3]

- **Per-Neuron interpretability, via full logical expressivity**: LNNs support function-free First-Order Logic with real values, which is expressible and allows for interpretable representation of knowledge and modeling uncertainity. Network structure is compositional, modular and disentangled.
- **Tolerant to incomplete knowledge, via truth bounds**: By use of truth bounds for each element, LNNs are able to model the open-world assumption, accomodating incomplete knowledge in a robust manner.
- **Many-task generality, via omnidirectional inference**: Neurons express bidirectional relationships with each neighbor, allowing omnidirectional inference useful for full-fledged theorem proving.

[3] Riegel et al. 2020.

Innovations[4]

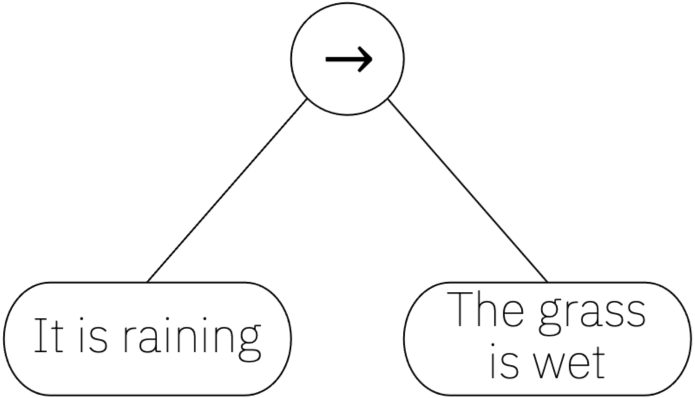
- Neural activation functions at every node representing a logical operation are constrained to implement the behaviour of the logical function. This is achieved by learning weights and biases that produce the desired truth table behaviour for given data.

- Results of neurons are expressed in terms of a pair of lower and upper bounds over truth values, so as to distinguish across known, approximately known, unknown and contradictory states.

- Bidirectional inference allowing for induction of consequents given antecedents, as well as deduction of antecedents from consequents (via rules like Modus Ponens).

[4] Khan et al. 2020.

Innovations[4]



[4] Khan et al. 2020.

Differences from Traditional Neural Networks[5]

- **Standard Neural Networks**: Dense representations with poor interpretability and explainability. Neurons perform feed-forward inference, with unidirectional flow of information. Further, absence of encoding or representation for uncertainities render inability to use incomplete knowledge.

- **Logical Neural Networks**: Interpretable, symbolic meaning associated with every individual neuron, identifying relative importance of each input fact. Able to perform sound logical reasoning using bi-directional inference. Further, using truth bounds allows expressing uncertainities and straight-forward use of domain knowledge, even when incomplete.

[5] Roukos, Gray, and Kapanipathi 2020.

Outline

Model Structure[6]

- Structure: DAG, composed of syntax trees of represented formulae, connected via neurons added for each proposition.
- Every neuron represents an element in a clause: concept/constant or a logical connective/operation. Connections between neurons are weighted, representing strengths of relationships between logical elements.
- Neurons return pair of values in the range [0,1], representing lower and upper bounds on truth values of the corresponding subformulae and propositions.
- Bounds are interpreted by use of a threshold $\alpha, 1/2 < \alpha <= 1$:

Bounds	Unknown	True	False	Contradiction
Upper	$[\alpha, 1]$	$[\alpha, 1]$	$[0, 1-\alpha]$	Lower > Upper
Lower	$[0, 1-\alpha]$	$[\alpha, 1]$	$[0, 1-\alpha]$	

[6] Riegel et al. 2020.

Model Structure[6]

- Logical Connective neurons accept outputs of operand neurons as input and have activation functions to model the connectives' truth function. Proposition neurons accept outputs of neurons established as proofs of bounds on truth values and have activation functions to aggregate the tightest bounds.
- Choice of activation functions determine the model of real-valued logic that is implemented. LNNs utilize weighted non-linear logic for activation, which can model most real-valued logic:
  - LNN-And:

$$^{\beta}\left(\bigotimes_{i\in I} x_i^{\otimes w_i}\right) = f\left(\beta - \sum_{i\in I} w_i(1 - x_i)\right)$$

  - LNN-Or:

$$^{\beta}\left(\bigoplus_{i\in I} x_i^{\oplus w_i}\right) = f\left(1 - \beta + \sum_{i\in I} w_i x_i\right)$$

[6]Riegel et al. 2020.

Inference[7]

- Inference involves computing truth value bounds for formulae, subformulae and atoms based on initial knowledge, resulting in predictions at neurons corresponding to queried formulae or other results of interest such as learned parameters.
- LNN achieves inference via repeated passes over the representing formulae, propogating tightened bounds across neurons until convergence.
- Bounds tightening is monotonic, computation cannot oscillate and necessarily converges for modeled logic.
- Each step of inference composed of upward and downward passes.

[7] Riegel et al. 2020.

Upward Pass[8]

- Connectives compute the truth value bounds based on available bounds for subformulae.
- Bounds computed according to evaluation of connectives based on operands.
- For monotonic activation functions implementing logical connectives' behaviour, convergence is achieved in finite time.
- Upwards bounds computation for disjunction:

$$L_{\bigoplus_i x_i} \geq {}^{\beta}(\textstyle\bigoplus_{i\in I} L_{x_i}^{\oplus w_i}), \qquad U_{\bigoplus_i x_i} \leq {}^{\beta}(\textstyle\bigoplus_{i\in I} U_{x_i}^{\oplus w_i})$$

- Inference for other connectives can be defined in terms of disjunction.

[8] Riegel et al. 2020.

Downward Pass[9]

- Neurons representing subformulae tighten their truth value bounds using bounds known for formulae and other sibling subformulae according to inference rules.
- Allows informing bound for propositions/predicates based on prior belief in truth or falsity of a formula.
- Such computations correspond to the inference rules of classical logic, whose precise nature is determined by the choice of activation functions.
- Downward bounds computation from disjunctions:

$$\begin{aligned} L_{x_i} &\geq {}^{\beta/w_i}((\textstyle\bigotimes_{j\neq i}(1-U_{x_j})^{\otimes w_j/w_i}) \otimes L_{\bigoplus_i x_i}^{\otimes 1/w_i}) && \text{if } L_{\bigoplus_i x_i} > 1-\alpha, && \text{else } 0 \\ U_{x_i} &\leq {}^{\beta/w_i}((\textstyle\bigotimes_{j\neq i}(1-L_{x_j})^{\otimes w_j/w_i}) \otimes U_{\bigoplus_i x_i}^{\otimes 1/w_i}) && \text{if } U_{\bigoplus_i x_i} < \alpha, && \text{else } 1 \end{aligned}$$

- Using inference rules, proofs for atoms can be generated.

[9]Riegel et al. 2020.

Learning[10]

- Using weighted non-linear logic, the model retains its differentiability, allowing for optimization via back-propogation of parameters: operand importance weights and truth value bounds.
- Loss functions may exploit logical interpretability, by penalizing contradiction states, to enforce complex logical requirements. This is modeled by addition of contradiction loss:

$$\begin{aligned} \min_{B,W} \quad & E(B,W) + \textstyle\sum_{k \in N} \max\{0, L_{B,W,k} - U_{B,W,k}\} \\ \text{s.t.} \quad & \forall k \in N,\ i \in I_k, \qquad \alpha \cdot w_{ik} - \beta_k + 1 \geq \alpha, \qquad w_{ik} \geq 0 \\ & \forall k \in N, \qquad \textstyle\sum_{i \in I_k} (1-\alpha) \cdot w_{ik} - \beta_k + 1 \leq 1 - \alpha, \qquad \beta_k \geq 0 \end{aligned}$$

- To allow weights to drop to 0 during optimization and permit non-classical behaviour, utilize slack variables:

$$\begin{aligned} \min_{B,W,S} \quad & E(B,W) + \textstyle\sum_{k \in N} \max\{0, L_{B,W,k} - U_{B,W,k}\} + \textstyle\sum_{k \in N} \mathbf{s_k} \cdot \mathbf{w_k} \\ \text{s.t.} \quad & \forall k \in N,\ i \in I_k, \qquad \alpha \cdot w_{ik} - s_{ik} - \beta_k + 1 \geq \alpha, \qquad w_{ik}, s_{ik} \geq 0 \\ & \forall k \in N, \qquad \textstyle\sum_{i \in I_k} (1-\alpha) \cdot w_{ik} - \beta_k + 1 \leq 1 - \alpha, \qquad \beta_k \geq 0 \end{aligned}$$

[10] Riegel et al. 2020.

Uses[11]

- LNN Inputs:
  - Training: Feature-value pairs, Loss function modeling constraints
  - Initial truth bounds for input nodes, inferrable from a Propositional / First-order Logic (FOL) Knowledge Base (KB)
  - Injected formulae representing queries or specific inference problems
- LNN Tasks:
  - Infer given formulae or determine truth values for specific nodes, using final truth value bounds of one or more output neurons
  - Determine relevancy of predicates in a connective, deduce rules for reasoning, or examine inconsistency of a knowledge base, using values of neural parameters — serving as a form of Inductive Logic Programming (ILP) — after learning using a specified loss function and training dataset.

[11] Riegel et al. 2020.

Outline

NSQA: Neuro-Symbolic Knowlegde-Base Question Answering[12]

- This work proposes a modular system for Knowledge-Base Question Answering (KBQA) using Neuro-Symbolic Techniques using Abstract Meaning Representation (AMR) parsers and LNN reasoners.
- Proposed system comprises of following components:
  1. AMR parsers using a stack-Transformer transition-based model, for semantic parsing of queries in a task-independent manner
  2. Transformation of AMR Graph to a query graph aligned with the underlying KG, using a path-based transformation approach coupled with entity and relationship linking to avoid problems of arity, structure or granular mismatch between AMR and KG.
  3. Encoding query graphs into an intermediate FOL representation to enable the use of independent FOL reasoners for spatial and temporal reasoning and handling incompleteness of the KGs.

[12] Kapanipathi et al. 2021.

NSQA: Neuro-Symbolic Knowlegde-Base Question Answering[12]

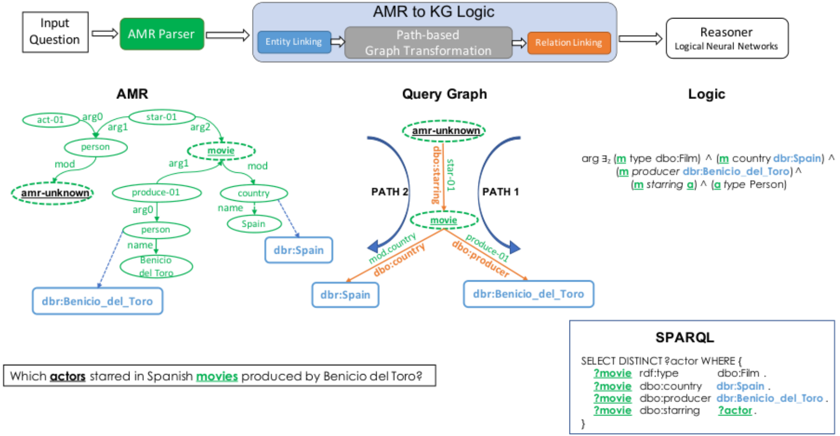


Figure: Example NSQA prediction for given query

[12] Kapanipathi et al. 2021

NSQA: Neuro-Symbolic Knowlegde-Base Question Answering[12]

Use of LNNs

- Use of LNNs as FOL reasoners over the intermediate FOL representation of the query graph.
- Currently, LNNs support type-based and geographic reasoning.
- Combined with heuristics to determine query type, target variables, and requirement of sorting and counting, reasoners can aid in simplification of the query.

[12] Kapanipathi et al. 2021.

LNN-EL: Short-Text Entity Linking[13]

- This work proposes a neuro-symbolic method utilizing LNNs for short-text Entity Linking, combining advantages of use of interpretable rules with performance of neural learning.
- Given a labeled dataset of mention-entity tuples and rule templates in FOL combining multiple features and/or previous EL approaches, the proposed approach can learn a suitable weighting of rules and features within rules, allowing for improved performance and interpretability.
- Rules are modeled as a set of predicates connected via logical connectives. Predicates are of form $f_k > \theta$, with f_k being a feature function. Feature functions may include both non-embedding and embedding based functions.

[13] Jiang et al. 2021.

LNN-EL: Short-Text Entity Linking[13]

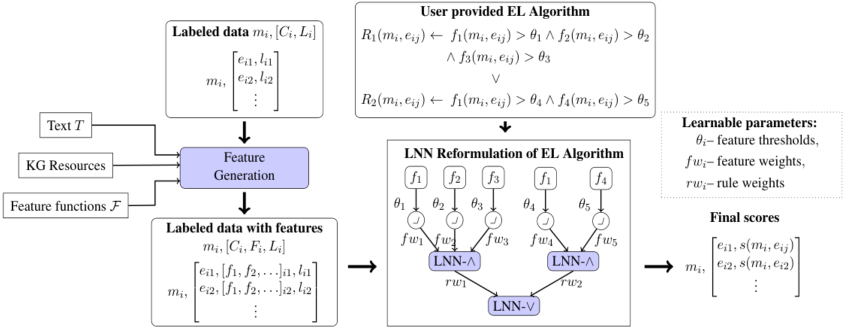


Figure: Overview of Entity-Linking using LNN-EL

[13] Jiang et al. 2021.

LNN-EL: Short-Text Entity Linking[13]

Use of LNNs

- Disjunction of EL rules form a larger EL algorithm. By assignment of weights to rules and feature predicates within rules, scoring of entity-mention mappings can be determined.
- To facilitate learning of weights and thresholds, rules may be mapped to an LNN formalism, utilizing disjunctions over rules and conjunction over feature predicates to represent rules.
- Feature predicates approximated using a differentiable function:

$$score(f > 0) = f \cdot \sigma(f - \theta)$$

- LNN formulated EL rules then trained using margin-ranking triplet loss:

$$\Sigma_{e_{ini} - \{e_{ip}\}} max(0, -(s(m_i, e_{ip}) - s(m_i, e_{in})) + \mu)$$

[13] Jiang et al. 2021.

SLATE - NeSy Text-Based Policy Learning[14]

- This work proposes a neuro-symbolic reinforcement learning model that aims to learn interpretable action-policy rules from symbolic abstractions of textual observations.
- Employs gradient-based logical rule learning over extracted symbolic fact abstractions coupled with negations to relevantly weight symbolic predicates towards every action verb. Actions then sampled based on probabilities of possible action commands for each grounding at time step t:

$$\{f_\theta(S_t(x_1, y_1)), f_\theta(S_t(x_1, y_2)), \cdots\}$$

- By use of end-to-end differential rule learning method, improved generalization based on evaluation over environments found in text-based games is observed.

[14] Chaudhury et al. 2021.

SLATE - NeSy Text-Based Policy Learning[14]

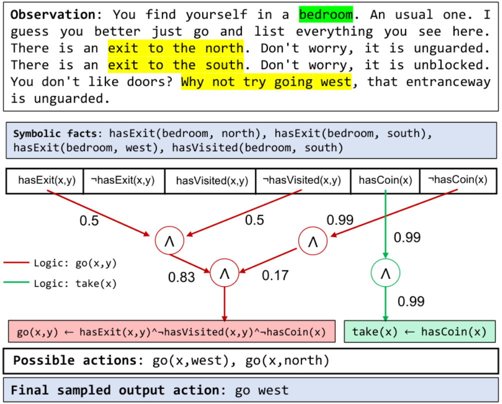


Figure: Overview of Rule Learning using SLATE

[14] Chaudhury et al. 2021.

- Differential Rule Learning achieved using two approaches: Symbolic MLP and LNNs, in an action specific manner.
- LNNs employed for action probability generation, by learning trainable parameters to constrain the network nodes to simulate the assigned logical connective, with the forward function modeled using an equivalent function from a suitable weighted real-valued logic.
- Model parameters trained using maximum likelihood training with cross-entropy loss with a frequency of updation after every 10 episodes for 100 episodes. Learning enhanced by rollout and teacher-imitation techniques.
- Following learning, rule extraction performed by selecting leaf level predicates whose weights satisfy a threshold (in the paper, $1/N_{input}$) as constituents.

[14] Chaudhury et al. 2021.

Bibliography I

- Khan, Naweed Aghmad et al. (2020). *Logical Neural Networks — Resources — IBM Research*. https://ibm.github.io/LNN/.
- Riegel, Ryan et al. (2020). *Logical Neural Networks*. DOI: 10.48550/ARXIV.2006.13155. URL: https://arxiv.org/abs/2006.13155.
- Roukos, Salim, Alexander Gray, and Pavan Kapanipathi (2020). *Using symbolic AI for knowledge-based question answering — IBM Research Blog*. https://research.ibm.com/blog/ai-neurosymbolic-common-sense.
- Chaudhury, Subhajit et al. (Nov. 2021). "Neuro-Symbolic Approaches for Text-Based Policy Learning". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 3073–3078. DOI: 10.18653/v1/2021.emnlp-main.245. URL: https://aclanthology.org/2021.emnlp-main.245.

Bibliography II

Jiang, Hang et al. (Aug. 2021). "LNN-EL: A Neuro-Symbolic Approach to Short-text Entity Linking". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, pp. 775–787. DOI: 10.18653/v1/2021.acl-long.64. URL: https://aclanthology.org/2021.acl-long.64.

Kapanipathi, Pavan et al. (Aug. 2021). "Leveraging Abstract Meaning Representation for Knowledge Base Question Answering". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Online: Association for Computational Linguistics, pp. 3884–3894. DOI: 10.18653/v1/2021.findings-acl.339. URL: https://aclanthology.org/2021.findings-acl.339.