Advances in Logic Gate Networks: From Differentiable Relaxations to Convolutional Designs

Taegun An
Department of CSE
Korea University
Seoul, South Korea
antaegun20@korea.ac.kr

Dohun Kim
Department of CSE
Korea University
Seoul, South Korea
dohunkim@korea.ac.kr

Changhee Joo*
Department of CSE
Korea University
Seoul, South Korea
changhee@korea.ac.kr

***Abstract*—Logic Gate Networks (LGNs) are emerging as efficient alternatives to traditional neural networks by leveraging hardware-native Boolean operations. This paper reviews recent advancements that address the challenges posed by the discrete, non-differentiable nature of these systems. Beginning with differentiable LGNs that enable gradient-based training, stochastic and convolutional LGNs achieve enhanced generalization and strong performance on vision tasks. This paper concludes by discussing open research directions, including training cost optimization, hardware acceleration, and the design of hybrid architectures.**

***Index Terms*—Logic gate networks, Differentiable logic gates, Gumbel-Softmax, Convolutional logic networks, Boolean operations, Hardware-Efficient inference, architecture search**

I. Introduction

Logic-based neural models originated with threshold logic units capable of implementing arbitrary Boolean functions [1], laying the foundation for binary neuron paradigms and computations such as binary-state dynamics [2]. Their hardware-friendly characteristics enabled efficient Boolean computations [3], particularly through FPGA implementations and threshold logic circuits. This line of research has been continuously studied and is now called *Logic Gate Networks* (LGNs), where it marks a shift toward highly efficient neural architectures that operate directly on logic-level primitives, from costly floating-point operations. LGNs utilize basic Boolean functions—AND, OR, XOR, NAND—natively implemented in digital hardware. This enables extreme inference efficiency, achieving over one million images per second on a single CPU core while preserving competitive accuracy [4].

Despite their efficiency advantages, logic-based binary neural models were largely under-explored due to their limited expressive capacity and the difficulty of optimization arising from their inherently discrete and non-differentiable nature [5]. Moreover, applying LGNs to complex modern AI tasks demands the construction of large-scale logic circuits with carefully designed connectivity and gate selection, which further complicates their design. Recently, the growing demand for fast and energy-efficient inference has revived interest in LGNs, spurring the development of differentiable architecture search techniques suited to this discrete setting. Breakthroughs in this area have demonstrated the feasibility of fully connected LGNs on MNIST [4], [6] and convolutional LGNs on CIFAR-10 [7], leveraging differentiable relaxations [4] for trainability, stochastic sampling to enhance generalization [6], and architectural innovations to improve performance on vision benchmarks [7].

In this paper, we review recent advances in LGNs that demonstrate competitive performance on benchmark datasets such as MNIST and CIFAR-10. We highlight methods that structure LGNs analogously to deep neural networks (DNNs) and convolutional neural networks (CNNs), and conclude by outlining key open challenges in the field.

This work was supported in part by NRF grant funded by the Korea government (MSIT). (RS-2022-NR070834)

C. Joo is the corresponding author.

II. Logic Gate Networks

We provide a brief introduction to three important developments of logic gate networks. First, we explain *Differentiable LGNs* that relax discrete binary logic gates to be differentiable, enabling efficient gradient-based methods for learning LGNs. Second, we describe *Stochastic LGNs*, which exploit Gumbel noise to relaxed gates for better generalization capacity. Finally, we explain *Convolutional LGNs*, which construct the convolutional kernel using a logic gate binary tree, achieving better performance for complex vision tasks.

A. Differentiable Logic Gate Networks

This cornerstone innovation allows gradient-based training of logic gate networks through *continuous relaxation* of discrete Boolean operations [4]. This technique transforms non-differentiable binary logic gates into differentiable real-valued functions using probabilistic logic interpretations.

The relaxation process operates in two aspects: activations and operations. First, binary activations $\{0, 1\}$ are relaxed to probabilistic activations in the range $[0, 1]$. Second, each logic gate is replaced by computing the expected value of the activation given the probabilities of independent inputs. For example, the logical AND operation ($a_1 \wedge a_2$) becomes the probabilistic multiplication $a_1 \cdot a_2$, while the exclusive OR ($a_1 \oplus a_2$) transforms to $a_1 + a_2 - 2 \cdot a_1 \cdot a_2$.

Applying these relaxations, it can parameterize the logic gate selection through a learnable probability distribution. Each neuron maintains this probability distribution over all N possible binary logic functions $g_1, ..., g_N$, encoded as the

softmax of N trainable parameters $\{w_i\}_{i=1}^N$. During training, the network evaluates all possible logic gates and computes their weighted average according to the learned probability distribution. The activation of a differentiable logic gate neuron is computed as

$$f_{\mathbf{w}}(a_1, a_2) = \sum_{i=1}^{N} \frac{\exp(w_i)}{\sum_{j=1}^{N} \exp(w_j)} \cdot g_i(a_1, a_2).$$

After training, continuous networks are discretized for hardware deployment by selecting the logic gate with the highest probability at each neuron. This discretization process introduces trivial accuracy loss (typically less than 0.1% for MNIST) while enabling extremely fast inference on Boolean hardware. Unlike traditional fully connected networks, the resulting networks achieve a computational complexity proportional to only the number of gates due to the sparsity nature of logic operations.

B. Stochastic Logic Gate Networks

Based on the differentiable framework, stochastic logic gate networks [6] incorporate a sampling method to improve generalization through regularized learning. The key technique lies in applying the Gumbel-Softmax reparameterization for categorical distribution in logic gate selection.

The Gumbel-Softmax trick enables differentiable sampling from categorical distributions by adding Gumbel noise to unnormalized logits before applying softmax [8]. For logic gate networks, this translates to the following:

$$s_i = \frac{\exp((w_i + G_i)/\lambda)}{\sum_{j=1}^{N} \exp((w_j + G_j)/\lambda)},$$

where Gumbel noise G_i is sampled from $\mathrm{Gumbel}(0, 1)$, and w_i represents the logits equal to the above learnable parameters, and λ is the temperature parameter.

C. Convolutional Logic Gate Networks

Convolutional logic gate networks [7] introduce three techniques to improve the performance for complex vision tasks: (1) Adopt logic gate tree function to capture fixed spatial patterns and correlations, analogous with convolutional neural networks, (2) logical *Or* pooling, and (3) residual initialization method to alleviate the vanishing gradient problem for deep networks.

Typically, the connections of differentiable logic gates are initialized at random, and LGNs learn logic operations over the random connectivity. Convolutional logic gate networks take a different approach. They employ a differentiable logic gate tree kernel to generalize edge, texture, and shapes in different locations as with CNNs. Although this tree kernel is wired at random in the receptive field, convolving activations with the tree kernel processes the equivariant features more effectively than the completely random connections.

Convolutional logic gate networks utilize logical *Or* pooling for traditional max-pooling. This operation selects the maximum activation over a receptive field using $\max(a, b)$ instead of probabilistic relaxation of logical *Or*, $a + b - ab$. The technique provides computational advantages: faster computation than probabilistic methods, reduced memory requirements (storing only maximum activation and index), and backpropagation only through maximum activations. An important emergent property is that training automatically prevents activation saturation, maintaining balanced activation levels without explicit regularization

Convolutional logic gate networks employ "residual initializations" to address the vanishing gradient problem in deep logic gate networks. Each logic gate is initialized to be primarily a feedforward gate (typically $a' = f(a_1, a_2) = a_1$ with 90% probability). This initialization prevents information loss and gradient vanishing in deeper networks, enabling effective training beyond the previous 6-layer limitation of LGNs. The technique acts as a differentiable form of residual connections without requiring additional logic gates, and the bias toward feedforward operations reduces transistor count in hardware implementations.

The combination of tree structures and *Or* pooling enables substantial computational improvements. Fused CUDA kernels process entire logic gate trees and pooling operations in single kernel calls, reducing memory accesses by 68% and memory footprint by 90% during training. The convolutional implementation achieves up to 200× speed improvement per logic gate compared to randomly connected implementations.

III. Conclusion

Logic Gate Networks (LGNs) offer a hardware-friendly alternative to traditional neural networks by using continuously relaxed logic gates, enabling ultra-fast and energy-efficient inference. Innovations like Gumbel-Softmax sampling, tree-based convolutions, and residual initialization have evolved LGNs into practical solutions for edge and embedded systems.

Future work should focus on improving training scalability, as current methods are still costlier than conventional networks due to multiple operator evaluations per neuron. Enhancing input connectivity, designing hardware accelerators, and integrating LGNs with standard neural components also present promising research directions.

References

[1] W. Mcculloch and W. Pitts, "A logical calculus of ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 127–147, 1943.

[2] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities." *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[3] R. Sayed, H. Azmi, H. Shawkey, A. H. Khalil, and M. Refky, "A systematic literature review on binary neural networks," *IEEE Access*, vol. 11, pp. 27 546–27 578, 2023.

[4] F. Petersen, C. Borgelt, H. Kuehne, and O. Deussen, "Deep differentiable logic gate networks," in *NeruIPS*, 2022, pp. 2006–2018.

[5] H. Qin, R. Gong, X. Liu, X. Bai, J. Song, and N. Sebe, "Binary neural networks: A survey," *Pattern Recognition*, vol. 105, 2020.

[6] Y. Kim, "Deep stochastic logic gate networks," *IEEE Access*, vol. 11, pp. 122 488–122 501, 2023.

[7] F. Petersen, H. Kuehne, C. Borgelt, J. Welzel, and S. Ermon, "Convolutional differentiable logic gate networks," in *NeruIPS*, vol. 37, 2024, pp. 121 185–121 203.

[8] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," 2017. [Online]. Available: https://arxiv.org/abs/1611.01144