EinChip: A 4.6/1.5 TOPS/W 6/24b Log-compute Einsum-based Accelerator for Neuro Symbolic AI

Lingyun Yao*, Shirui Zhao†, Marian Verhelst†, Martin Andraud*§
*ELEC, Aalto University, Otakaari 1B, 02150, Espoo, Finland †MICAS, KU Leuven, Oude Markt 13, 3000 Leuven, Belgium
§ICTEAM, UC Louvain, Pl. de l'Université 1, 1348, Louvain-la-Neuve, Belgium
shirui.zhao@kuleuven.be

2026 IEEE Custom Integrated Circuits Conference (CICC) | 979-8-3315-7210-5/26/$31.00 ©2026 IEEE | DOI: 10.1109/CICC65509.2026.11509460

***Abstract*—Neuro Symbolic (NeSy) AI combines deep neural networks (DNNs) with symbolic reasoning, but existing DNN accelerators are ill-suited for NeSy workloads. We present EinChip, a unified NeSy accelerator that expresses both neuro and symbolic models in Einstein Summation Notation (Einsum) and executes them on an approximate 6/24-bit logarithmic processing unit. A 4mm^2 16nm prototype achieves 4.6 TOPS/W on neuro tasks and 1.5 TOPS/W on symbolic tasks with only 0.1% accuracy loss.**

***Index Terms*—Einsum, log-compute, Neuro Symbolic, AI accelerator**

I. Introduction

Although deep neural networks (DNNs) and Large Language Models (LLMs) have become a de facto standard for the development of Artificial Intelligence (AI), they are often criticized for limited interpretability, overconfidence, and trustworthiness, as well as for their ever-growing computational footprint [1]. For trustworthy and efficient embedded AI applications, it is thus crucial to find alternatives. Neuro symbolic AI (NeSy AI), as illustrated in Fig. 1 (top), is one of them [2]. NeSy AI combines the complex perception capabilities of neuro (Ne) models with the reasoning abilities of symbolic (Sy) models, and is increasingly applied to "human-like" reasoning, vision, and language [2]. Among symbolic models, probabilistic approaches are particularly prominent, offering explicit, explainable reasoning via probability computations [3].

In an ideal scenario, a NeSy AI accelerator would efficiently span across varying sparsity levels (dense Ne and sparse Sy tasks) and computation resolutions (INT4/INT8 for Ne and FP32/FP64 for Sy). In reality, Fig. 1 (middle) shows that symbolic models become the computational bottleneck of NeSy tasks despite having significantly fewer operations [4]. As shown in Fig. 1 (bottom), DNNs benefit massively from hardware acceleration on GPUs, NPUs, or Compute-in-memory (CIM) ASICs [5]; however, these accelerators perform poorly on symbolic benchmarks [4]. Although dedicated accelerators exist for symbolic tasks, e.g. DPU [6], supporting high-resolution computation and computational irregularity, their peak throughput (18.4 GOPS) is orders of magnitude below DNN accelerators operating at TOPS scale [5], heavily limiting their performance on Ne tasks. Moreover, developing an AI acceleration system with multiple accelerators may add significant complexity and cost.

In this context, we present EinChip, a 4mm^2 16nm SoC for NeSy AI tasks. It allows computing symbolic (Sy) tasks with significant throughput and efficiency, while staying competitive on neuro (Ne) tasks. This is achieved through (1) adapting the Einsum notation to capture basic Einsum operations and reconstructing both Ne and Sy models such that they can be accelerated by a unified computing architecture; (2) an efficient mixed-precision logarithmic processing unit for both Ne and Sy tasks; and (3) a flexible Chip architecture to accelerate both Ne and Sy tasks. EinChip attains 156.6 GOPS on Sy workloads (8.5× higher than 18.4 GOPS DPU baseline) while also achieving a peak throughput of 979.2 GOPS for Ne tasks.

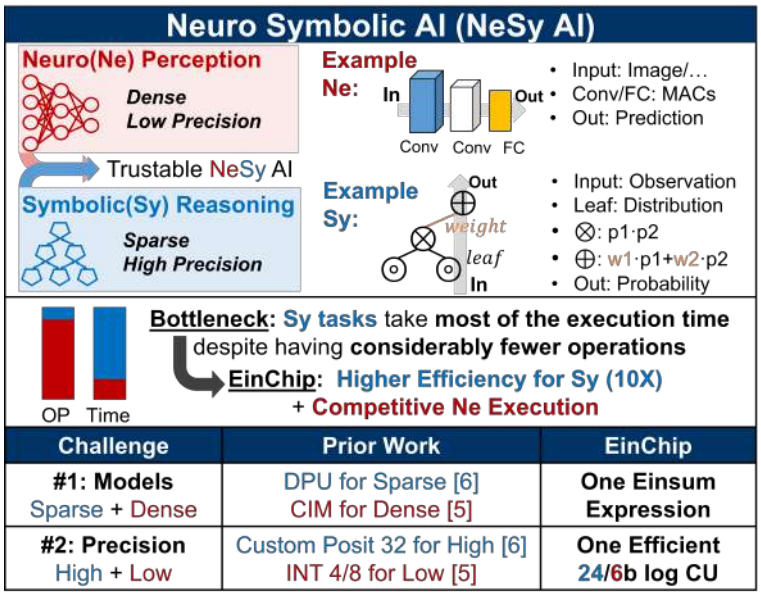

Fig. 1. Top: Neuro Symbolic AI; Middle: Symbolic Computing Bottleneck due to Sparsity and High Precision; Bottom: Two Challenges Addressed by EinChip.

II. EinChip Design

In this section, we detail the three contributions made by EinChip: *A:* representing Ne and Sy tasks with Einsums, *B:* presenting a log computing block (LSE) to increase power and area efficiency, and *C:* developing a flexible Chip architecture for computational efficiency.

A. Using Einsums to Capture Various Operations

The Einsum notation provides a unified framework for representing operations existing in different models, as shown in Fig. 2 (top). Here, the symbolic model vectorising approach [7] is used to identify the basic Einsum operations to reconstruct neuro and symbolic models, such that they can be accelerated by one unified computing architecture.

For symbolic models, this work focuses on probabilistic methods. Such models are built upon two core principles: Joint Probability

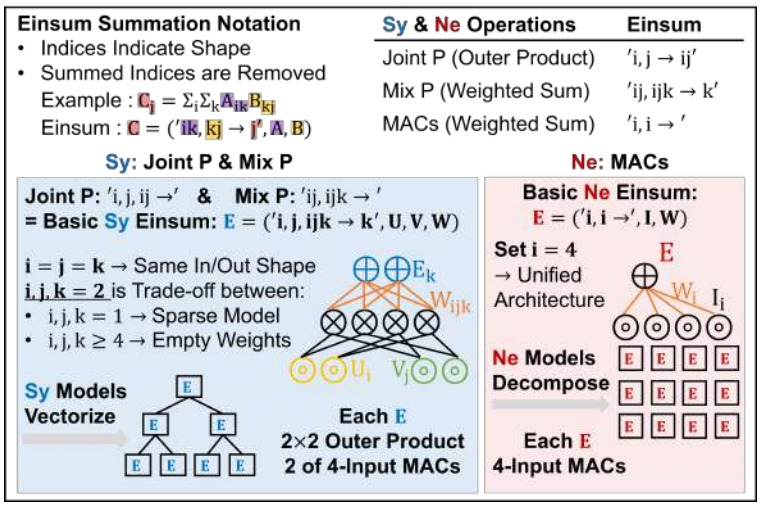

Fig. 2. Top: Einsum representation of operations in NeSy AI; Bottom: Basic Einsum operations, E and E, are identified to reconstruct Sy and Ne models for acceleration within a unified computing architecture in EinChip.

(Joint P) and Mixture Probability (Mix P) calculations. The Joint P, $'i, j \rightarrow ij'$, computes the outer product of probability distributions. The Mix P, $'ij, ijk \rightarrow k'$, performs a weighted summation over probabilities. By aligning the output dimension of Joint P with the input dimension of Mix P, they can be merged into one basic Sy Einsum operation: $E = ('i, j, ijk \rightarrow k', U, V, W)$, where U and V denote input probability vectors, W represents the weight tensor, and the vector E denotes the output probabilities vector. We employ E as the fundamental computational unit during training to construct the symbolic model. To ensure consistency between input and output dimensions, indices i, j, k are all set as 2. This configuration results in each Sy Einsum containing eight weights for a balance between maintaining a vectorised structure and preserving sufficient flexibility in constructing the model. Computing each E begins with an outer product layer comprising four multipliers, whose results are shared by two parallel 4-input multiplication accumulating (MAC) units.

In deep neural networks, the various layers can be decomposed into MAC operations, which can also be expressed as Einsums of the form $'i, i \rightarrow'$. To align symbolic and neuro operations within a unified hardware architecture for EinChip, neuro computations are further decomposed with $E = ('i, i, \rightarrow', I, W)$, with $i = 4$, where each E corresponds to a 4-input MAC unit.

B. Logarithmic Computing Unit using Approximate LSE

Computation format. To develop an area and power-efficient computing block *for both Sy and Ne kernels*, EinChip employs a customised approximate logarithmic processing element, based on the Log-Sum-Exp (LSE) operation [8]. As shown in Fig. 3 (bottom right), a 24-bit FxP logarithmic format (14 integer + 10 fractional bits) is chosen to obtain a higher range than double-precision FP, necessary for Sy workloads. As Ne workloads use significantly lower resolution, the same unit can be reconfigured as 4x 6b FxP log computation blocks, using Single Instruction Multiple Data (SIMD) execution. An additional sign bit handles negative values, with 4 integer bits

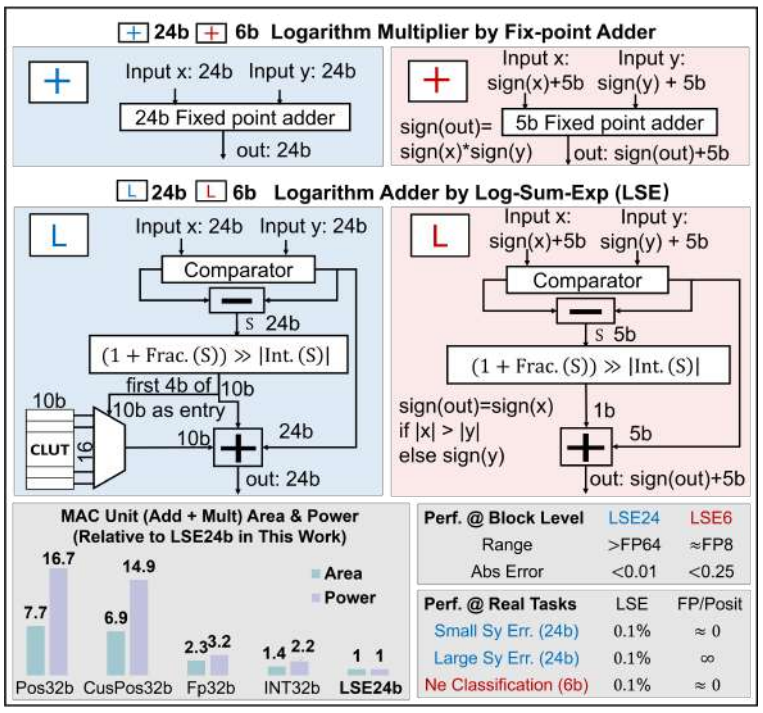

Fig. 3. Top: Custom Logarithmic Computing Unit using Approximate LSE in EinChip: + (24-bit Sy) and + (6-bit Ne) for Multiplication, L (24-bit Sy) and L (6-bit Ne) for Addition; Bottom left: MAC unit (Add + Mult) Area, Power Comparison; Bottom right: LSE Range, Accuracy Performance (∞ due to underflow).

providing a dynamic range comparable to a 4-bit exponent FP formats and 1 fractional bit for precision.

LSE computation. Log computing, especially LSE, is widely adopted as a numerically stable method for training probabilistic models on conventional computing platforms [7]. Given two logarithmic inputs x and y, log-domain multiplication can be efficiently realised through a fixed-point addition: $x + y$, as indicated by the symbol + (24-bit Sy) and + (6-bit Ne) in Fig. 3. The exact log-domain addition is transformed into: $\log_2(2^x + 2^y) = x + \log_2(1 + 2^{(y-x)})$, for $x > y$, and is implemented using a customized approximate logsumexp operation, denoted by the symbol L (24-bit Sy) and L (6-bit Ne) in Fig. 3. Both L and L begin with a comparator and a subtraction, where the result is stored as S. Two approximations are then applied to replace the exponential and logarithmic term: $2^z \approx z + 1, \log_2(1 + z) \approx z$. The resulting expression $(1 + \text{Frac.}(S)) \gg |\text{Int.}(S)|$, where $\text{Int.}(S)$ and $\text{Frac.}(S)$ denote the integer and fractional parts of S, respectively, approximates $\log_2(1 + 2^{(y-x)})$. This value remains within $[0, 1)$, allowing L for Sy computing to use a compact 4-bit correction lookup table (CLUT) for error compensation [8]. As seen in Fig. 3 (bottom left), the 24-bit LSE MAC hardware achieves up to 7.7× area and 16.7× power reduction compared to Posit arithmetic.

LSE performance. On Sy workloads, the CLUT correction maintains the absolute error below 0.01. The 24-bit LSE unit achieves $\approx 0.1\%$ relative error versus software references, outperforming prior 32-bit FP and Posit designs that underflow in large-scale tasks. For Ne workloards, the 4x6-bit log SIMD execution, achieves 4× higher

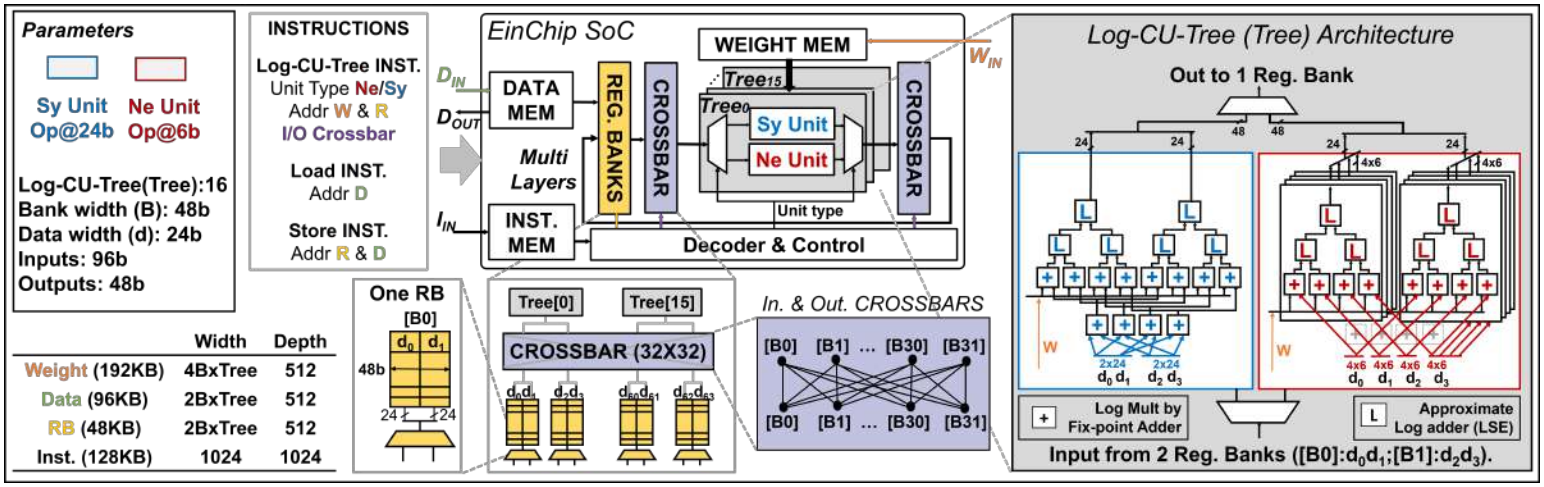

Fig. 4. EinChip Architecture and Single Log-CU-Tree architecture

computational density under the same input bandwidth. The 6-bit LSE error is bounded by one fractional bit, corresponding to a maximum absolute error of 0.25, thereby eliminating the need for a CLUT. Experiments with 6-bit LSE on the MNIST classification task demonstrate $\approx 98\%$ accuracy using subsets of test points, with the software baseline (6-bit log-quantised weights) of 98.12% accuracy.

C. Architecture

Fig. 4 illustrates the overall architecture of EinChip. It comprises 16 parallel Logarithm Computation Units Trees: Log-CU-Tree (Tree), 32 register banks (RBs), in which each bank contains two 24-bit elements (48-bit). The 16 Log-CU-Tree and 32 register banks are connected through 32x32 input and output crossbars, similar to [6], to execute multiple Sy layers. Weight and data memories store the weights (W) and input/output data, where the weight memory is directly connected with the Log-CU-Tree. The data memory is connected to the Log-CU-Tree through the RBs and the crossbar.

Each Log-CU-Tree contains a Sy unit and a Ne Unit. Each Sy unit reads 48-bit U, V from 2 different RBs and executes one basic Sy Einsum E (see II-A), utilizing 4 $\boxed{+}$ for the outer product layer, and 8 $\boxed{\mathrm{L}}$, 6 $\boxed{+}$ for 2 parallel 4-input MACs. Each E uses eight weights directly from the weight memory and generates two 24-bit outputs, which are transferred to one 48-bit RB. Each Ne unit skips the outer product layer, and 2 parallel 4-input MACs share the input data. Each 4-input MACs further partitions each 24-bit input data and 24-bit weights into four 6-bit logarithmic computing units. Consequently, each Ne unit performs eight Ne Einsum operations E (see II-A), using 6-bit logarithmic computations. The Ne Unit maintains the same input and output bandwidth as the Sy unit, which allows multiplexing their memory interfaces. With 16 Log-CU-Trees, the system achieves a total of 512 parallel MAC operations per cycle.

The processor is programmed through a custom compiler, with SIMD instructions stored in the instruction memory. The input data is loaded from data memory through LOAD instructions with a data memory address. The Log-CU-Tree computation is controlled by the selected task type, weight address, RB addresses, and crossbars.

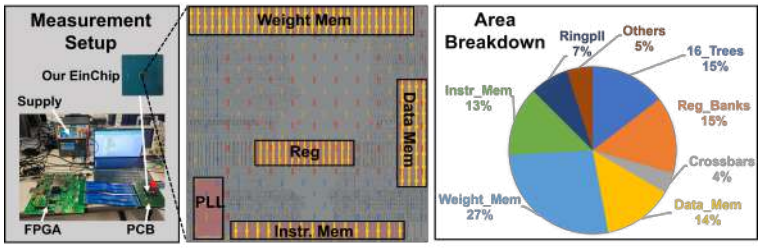

Fig. 5. Measurement Setup, Post Layout Photo, Post Synthesis Area Breakdown.

Results are written back through the STORE instructions with the data-memory address and the RB addresses.

III. Measurement Results

EinChip is implemented in 16nm CMOS technology, occupying $4\mathrm{mm}^2$. Fig. 5 shows the measurement setup, post-layout photo, post-synthesis area breakdown of each element (without IO) on the chip.

A. Peak Performance

Fig. 6 presents the chip's Shmoo plot obtained from successfully executing multiple neuro and symbolic workloads. EinChip achieves a peak Neuro throughput 979.2 GOPS at $\mathrm{F}_{\max} = 956.25$ MHz; peak Symbolic throughput 156.6 GOPS at $\mathrm{F}_{\max} = 543.75$ MHz at VDD $= 1.05$ V. The highest efficiency is 4.57 GOPS/W for Neuro and 1.50 TOPS/W for Symbolic at VDD $= 0.55$ V.

B. Performance Comparison

EinChip is compared with prior accelerators in Table I. As there were no direct SoC comparisons for NeSy tasks, EinChip is compared to reference chips for mixed-precision Sy workloads [6], Ne acceleration with intensive-CIM and sparse-digital FP cores [5], and mixed-precision DSP tasks [9]. For low-resolution Ne tasks, EinChip achieves comparable throughput with CIM cores and up to $13.3\times$ higher throughput and $8.5\times$ higher efficiency compared to other accelerators. For high-resolution Sy tasks, EinChip achieves up to $24.5\times$ higher throughput and $13.8\times$ higher efficiency, exceeding FP64

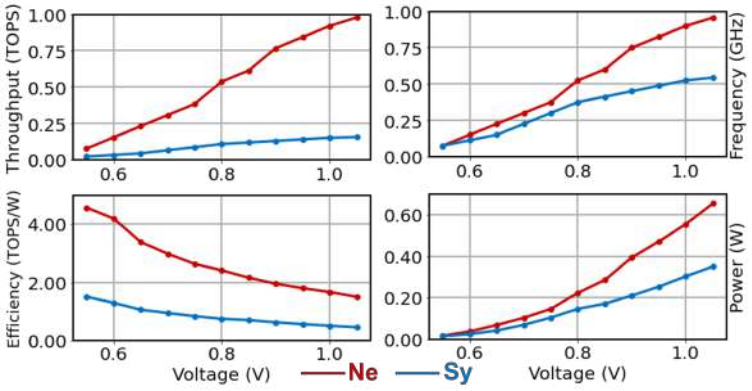

Fig. 6. Top left: Peak throughput (TOPS); bottom left: Peak energy efficiency (TOPS/W); top right: Frequency (GHz); and bottom right: Power (W) versus voltage.

TABLE I
PERFORMANCE COMPARISON OF REPRESENTATIVE ACCELERATORS

Specification	DPU [6]	CIM+FP [5]	INT8+FP32 [9]	EinChip
Tech. (nm)	28	28	16	**16**
Area (mm^2)	3.8	4.54	16.0	**4.0**
Core (Low)	Sy Posit8/16	CIM INT4/8/16	INT8	**Ne LSE6**
Throu. (GOPS)	73.8 (Posit8)	1640 (INT4); 820 (INT8)	274 (INT8)	**979.2 (LSE6)**
Effi. (GOPS/W)	538 (Posit8)	51k (INT4); 12.8k (INT8)	414 (INT8)	**4571.4 (LSE6)**
Acc. Loss	≈15% ↓ (Posit8)	4.2% ↓ (INT4 Cifar10)	–	0.1% ↓ **(LSE6)**
Core (High)	Sy Posit32	Digital BF16/FP16	FP32	**Sy LSE24**
Throu. (GOPS)	18.4 (Posit32)	6.4 (FP16)	69.6 (FP32)	**156.6 (LSE24)**
Effi. (GOPS/W)	134.5 (Posit32)	–	109 (FP32)	**1500 (LSE24)**
Acc. Loss	≈ 0/∞ (Posit32)	2.6% ↓ (FP16 Cifar10)	–	0.1% ↓ **(LSE24)**

range with only a 0.1% numerical accuracy drop while ∞ error exists in SotA accelerator due to underflow.

C. Performance on (Neuro) Symbolic Tasks

On Sy tasks from the DPUV2 dataset repository [10], EinChip is evaluated against DPU [6], DPUV2, and their reported CPU/GPU baselines. As shown in Fig. 7, EinChip achieves 2.2 to 8.9x higher throughput than DPU, with the 24-bit LSE providing stable 99.99% numerical accuracy. Its vectorised Einsum representation requires fewer operations than the scalar probabilistic circuits used in prior works while maintaining comparable model performance (i.e., similar likelihood) [7], thereby further reducing Sy runtime.

EinChip's performance is further evaluated on a NeSy AI task, as shown in Fig. 8, compared against a hypothetical state-of-the-art (SotA) platform comprising the DPU for Sy [6] and a CIM accelerator for Ne workloads using the highest throughput INT4 configuration [5]. On the baseline, the Sy component accounts for 95% of the total runtime (2950 ns) despite representing only 8% of overall operations. In contrast, EinChip achieves a 4.7× reduction in total inference time, driven by its Sy performance (7.3× faster than SotA) while maintaining 98% Ne accuracy and 99.99% Sy numerical accuracy.

IV. CONCLUSION

EinChip is the first chip accelerating Neuro Symbolic AI tasks in one integrated architecture. Implemented in 16nm using 4mm^2, it combines Einsum representation and LSE computing units, addresses both the structural and the computational challenges of NeSy AI, demonstrating 4.6/1.5 TOPS/W for Ne/Sy computing.

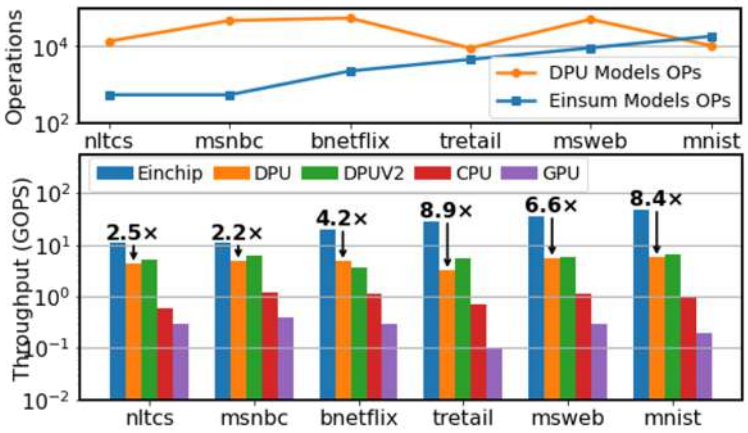

Fig. 7. Top: Sy Task Operations; Bottom: Throughput (GOPS) compared to DPU, DPUV2, CPU and GPU data from [10].

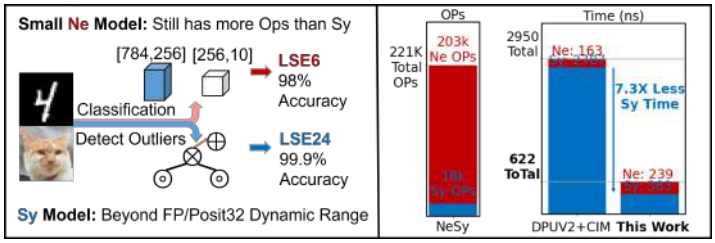

Fig. 8. Neuro Sy Task Performance, a probabilistic model (EiNet) is used as an outlier detection to prevent a DNN from making predictions on unknown input data.

REFERENCES

[1] G. Marcus, "The next decade in ai: four steps towards robust artificial intelligence," *arXiv preprint arXiv:2002.06177*, 2020.
[2] A. d. Garcez and L. C. Lamb, "Neurosymbolic ai: The 3 rd wave," *Artificial Intelligence Review*, vol. 56, no. 11, pp. 12 387–12 406, 2023.
[3] Z. Ghahramani, "Probabilistic machine learning and artificial intelligence," *Nature*, vol. 521, no. 7553, pp. 452–459, May 2015.
[4] Z. Wan, H. Yang, R. Raj, C.-K. Liu, A. Samajdar, A. Raychowdhury, and T. Krishna, "Cogsys: Efficient and scalable neurosymbolic cognition system via algorithm-hardware co-design," in *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2025, pp. 775–789.
[5] S. Yan, J. Yue, C. He, Z. Wang, Z. Cong, Y. He, M. Zhou, W. Sun, X. Li, C. Dou *et al.*, "A 28-nm floating-point computing-in-memory processor using intensive-cim sparse-digital architecture," *IEEE Journal of Solid-State Circuits*, vol. 59, no. 8, pp. 2630–2643, 2024.
[6] N. Shah, L. I. G. Olascoaga, S. Zhao, W. Meert, and M. Verhelst, "Dpu: Dag processing unit for irregular graphs with precision-scalable posit arithmetic in 28 nm," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 8, pp. 2586–2596, 2021.
[7] R. Peharz, S. Lang, A. Vergari, K. Stelzner, A. Molina, M. Trapp, G. Van den Broeck, K. Kersting, and Z. Ghahramani, "Einsum networks: Fast and scalable learning of tractable probabilistic circuits," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7563–7574.
[8] L. Yao, S. Zhao, M. Trapp, J. Leslin, M. Verhelst, and M. Andraud, "Logsumexp: Efficient approximate logarithm acceleration for embedded tractable probabilistic reasoning," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2025.
[9] V. Jain, D. Grubb, J. Zhao, K. Anderson, K. T. Ho, Y. Chi, E. Schwarz, K. Asanović, Y. S. Shao, and B. Nikolić, "Cygnus: A 1 ghz heterogeneous octa-core risc-v vector processor for dsp," in *2025 Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, 2025, pp. 1–3.
[10] N. Shah, W. Meert, and M. Verhelst, "Dpu-v2: Energy-efficient execution of irregular directed acyclic graphs," in *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2022, pp. 1288–1307.