

Neuro-Symbolic Computing: Advancements and Challenges in Hardware-Software Co-Design

Xiaoxuan Yang, Zhangyang Wang, X. Sharon Hu, Chris H. Kim, Shimeng Yu, Miroslav Pajic, Rajit Manohar, Yiran Chen, and Hai Helen Li

Abstract—The rapid progress of artificial intelligence (AI) has led to the emergence of a highly promising field known as neuro-symbolic (NeSy) computing. This approach combines the strengths of neural networks, which excel at data-driven learning, with the reasoning capabilities of symbolic AI. Neuro-symbolic models have the potential to overcome the limitations of each approach individually, resulting in interpretable and explainable AI systems that can reason over complex knowledge bases, learn from limited and/or noisy data, and be generalizable. However, the exploration of NeSy AI from a system perspective remains limited. This paper provides an in-depth analysis of the state-of-the-art hardware-software co-design techniques for NeSy AI and discusses the associated challenges in improving system efficiency for heterogeneous computing. By examining the intersection of NeSy computing and system design, we aim to bridge the gap and foster advancements in this domain.

Index Terms—Neuro-symbolic, Hardware-Software Co-Design, Compositionality, Reasoning, Generalization

I. INTRODUCTION

THE advancement of artificial intelligence (AI) has revolutionized virtually every aspect of human existence. AI applications span from personal assistants and health monitoring to the development of self-driving cars. A comparative analysis of AI's capabilities reveals that the evolution of AI models has led to exceptional achievements in tackling complex tasks such as speech recognition, image classification, and object detection [1]. AI models have transitioned from systems with thousands of parameters [2] to billions of parameters in large language models (LLMs) [3], and this dramatic increase in model complexity has led to systems that can surpass human performance in a growing list of tasks.

However, the current exploration of AI is primarily oriented towards seeking the optimal and powerful models in terms of performance, often prioritizing accuracy on specific

tasks, while neglecting more general reasoning capabilities. For instance, DNN models lack the capacity for reasoning and generalization when faced with out-of-distribution data or novel tasks [4]. Moreover, LLMs still exhibit limitations for accurately quantifying uncertainty and effectively solving mathematical problems [5]. These limitations hamper models' overall adaptability and utility. Thus, it is important to incorporate reasoning ability with computing performance, leading to the study of neuro-symbolic (NeSy) computing [6], [7].

Neural networks require end-to-end training with global optimization. This interdependence couples all parameters, preventing arbitrary combinations from different layers to form a high-performance neural network. The absence of compositional structure complicates the understanding and explanation of the functions within neural network building blocks. Consequently, researchers focus on achieving compositional and interpretable aspects in NeSy computing [8]–[10].

For neural network deployment, hardware accelerators are often specifically designed for one type of task or model. For instance, the systolic array design for convolutional neural networks [11] and crossbar-based in-memory computing for vector-matrix multiplication [12]. Moreover, due to the specific requirements of neural building blocks, hardware designs must align with the software algorithm and lack universal applicability. More importantly, optimizing hardware configurations for one scenario does not necessarily provide guidance for future deployments [13]. Therefore, solving the compositionality challenge could enable highly efficient and reusable hardware system optimization across various scenarios.

NeSy algorithms, which involve symbolic reasoning and dense neural computations, require significant computational demands. Hardware acceleration can significantly speed up these computations, emphasizing the necessity and potential of hardware-software co-design in NeSy. This paper comprehensively analyzes state-of-the-art hardware-software co-design techniques for NeSy AI while addressing the challenges related to enhancing system efficiency in the context of heterogeneous computing. We delve into the trends and challenges of hardware-software co-design for NeSy system development. Key areas of focus include improving reasoning and generalization capabilities (in particular, enabling compositional unseen generalization), addressing uncertainty, enhancing system efficiency, and demonstrating full-stack NeSy systems.

The organization of this paper is as follows: Section II briefly introduces the background. Section III further covers the important trends and challenges in NeSy architectures, learning mechanisms, and hardware system design. Section IV discusses the application and future topics of NeSy, and Section V concludes this paper.

X. Yang is with the Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904 USA (e-mail: xiaoxuan@virginia.edu).

Z. Wang is with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712 USA (e-mail: atlaswang@utexas.edu).

X. S. Hu is with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556 USA (e-mail: shu@nd.edu).

C. H. Kim is with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: chriskim@umn.edu).

S. Yu is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: shimeng.yu@ece.gatech.edu).

R. Manohar is with the Computer Systems Laboratory, Yale University, New Haven, CT 06520 USA (e-mail: rajit.manohar@yale.edu).

M. Pajic, Y. Chen, and H. H. Li are with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA (e-mail: miroslav.pajic@duke.edu; yiran.chen@duke.edu; hai.li@duke.edu).

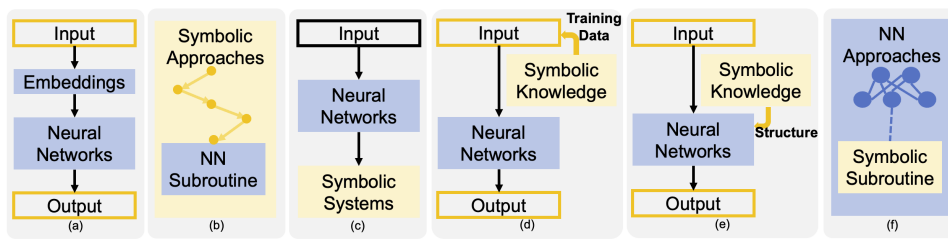


Fig. 1. Six representative Neuro-Symbolic designs.

II. BACKGROUND

A. Neuro-Symbolic Artificial intelligence

According to Henry Kautz’s taxonomy [14], there are six types of neuro-symbolic designs that offer different integrations and interactions between neural and symbolic counterparts. The first type is known as *Symbolic Neuro Symbolic*, where the neural interface connects the symbolic input and output. As an illustration, consider the Transformer model [15], which begins with the input of a document. This input passes through the embedding block before connecting to the multi-head self-attention and feedforward block. Finally, a softmax block is designed to transform the output from the last layer to a symbolic category or information. The advantage of this type of NeSy design is that the hardware accelerators for different neural networks are well studied, and the area and energy efficiency of the integrated system can be boosted with dedicated hardware architecture [16], [17] and hardware-software co-designs [18], [19]. However, in such kind of integration, neural networks play a significant role, potentially resulting in reduced interpretability and compositionality.

The second type is to utilize the neural part as a subroutine for the symbolic reasoning process, denoted as *Symbolic[Neuro]*, among which the AlphaGo program is a notable example [20]. To achieve a latency of under 5 seconds, the distributed AlphaGo program utilizes 1,202 CPUs and 176 GPUs to conduct inference for policy and value networks with Monte Carlo tree search. Besides, the neural part can work as a coroutine for the symbolic process, leading to the third type *Neuro;Symbolic*. This type of NeSy extends the input range to non-symbolic input, and the neural part can transform the input into symbolic results. In summary, for the second and third types of NeSy, the task mapping and scheduling between the neural part and the symbolic part should be investigated in the NeSy architecture design.

The fourth NeSy system *Neuro;Symbolic*→*Neuro* complies symbolic knowledge into training data. For instance, expression tree [21], logic program [22], and graph knowledge [23] can be compiled to augment the reasoning ability. The fifth type *Neuro_Symbolic* includes approaches in which the first-order logic language is tensorized and neural techniques are applied to reason over the tensorized first-order logic representation [24]. It is worth investigating combinatorial reasoning based on these two types of NeSy [14]. Furthermore, the system design can be heterogeneous and flexible to efficiently support the various data processing and movement [25].

The sixth type of NeSy is denoted as *Neuro[Symbolic]* and can be considered as a transpose version of *Symbolic[Neuro]*. *Neuro[Symbolic]* embed a symbolic reasoning engine inside a neural engine in order to achieve the computing performance of the neural engine while maintaining the reasoning ability at the symbolic level. NeSy system employs the neural network interface to recognize the problem and utilizes the symbolic reasoning engine to perform the combinatorial search. The process concludes with neural network interpretation/translation of symbolic behavior to instructions. The symbolic reasoning engine can also be considered a subroutine for the whole process. Furthermore, independent symbolic reasoning engines can be combined to perform complex reasoning tasks, thus achieving compositionality. Therefore, the last type has great potential in building cognitive and interpretable NeSy systems.

In summary, NeSy AI exhibits two key features. Firstly, it embraces compositionality. Note that this is not a unique feature for NeSy architectures but is considered a generic property shared by other compositional systems or models, such as compositional neural networks [26]–[28]. More discussions on the hardware-software co-design for compositionality can be found in Section III-B. Secondly, NeSy systems incorporate the stacking or interleaving of neural and symbolic modules, as shown in Fig. 1. We will discuss the related potential hardware benefits in Section III-D.

B. Emerging Hardware Computing Platforms

In this paper, we aim to explore the different platform candidates for NeSy systems, including traditional platforms and emerging hardware platforms. Due to space limit, this background mainly covers examples of hardware computing platforms based on emerging device technologies, namely resistive random-access memory (ReRAM), ferroelectric field effect transistor (FeFET), and coupled oscillator. We also analyze the potential of implementing NeSy tasks on these platforms.

Resistive Random-Access Memory (ReRAM): ReRAM cell is made up of a metal-oxide material layer inserted between two metal electrode layers. Moreover, ReRAM crossbars can realize vector-matrix multiplication (VMM), which significantly mitigates the communication cost of data between the memory and computing unit [29]. Hence, leveraging ReRAM-based in-memory computing presents a promising solution for achieving energy-efficient solutions to VMM, a common bottleneck in various neural network computations. The potential utilization of the ReRAM platform for NeSy applications will be elaborated in Section III-A and III-D.

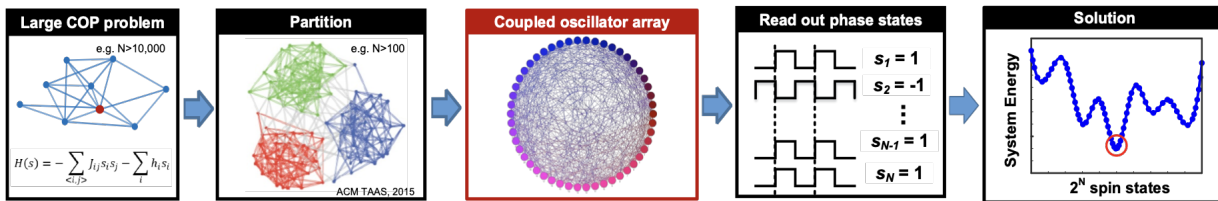


Fig. 2. General workflow of Ising computers, including applications, problem partition, and obtaining solutions [32].

Ferroelectric Field Effect Transistor (FeFET): These transistors are built by stacking a ferroelectric (FE) layer on the gate of a MOSFET. FeFET-based ternary content addressable memory (TCAM) performs parallel and efficient searches over the stored memory blocks per input query and has been widely used in data-intensive applications, such as associative memory, database, and in-memory data processing [30]. Furthermore, multi-bit FeFET can be utilized to perform hyperdimensional computing [31]. This will benefit the area-efficient deployment of the NeSy approach with promising compositional capability, as explained in Section III-C.

Coupled Oscillator: In this approach, the state information is encoded using the phase of an oscillator. When a non-linearity is introduced, an oscillator resonating at one specific frequency can synchronize its phase with a pump frequency that doubles the resonating frequency, resulting in two stable phase states, 0 or π , which represent digital information. Furthermore, this approach allows for more parallel and natural computing methods [33]. For example, coupled oscillators have the potential to address combinatorial optimization problems (COPs) effectively [32], [34]. As illustrated in Fig. 2, a large COP can be decomposed to several sub-COPs and solved by the computing unit of all-to-all connected coupled oscillator array [35]. This prospective hardware design for COP has two advantages for the NeSy System. Firstly, certain reasoning tasks, such as the design of robotic architectures [36], demand an efficient solver for COP. Secondly, the NeSy system itself is intricately complex and necessitates the optimization of design configurations, as elaborated in Section III-E.

III. TRENDS AND CHALLENGES

NeSy models have already demonstrated promising results in specific tasks, rendering them exceptionally attractive for future explorations. For instance, symbolic regression offers efficient analytic solutions and can outperform a three-layer neural network in terms of latency by $13\times$ on a physics benchmark [37]. Another example is that the symbolic branching algorithm converts a reinforcement learning model to a symbolic tree, thus enabling $23\times$ speedup [38]. These efforts target relatively simple NeSy tasks and mark the beginning of the exploration into the performance-resource tradeoff for NeSy systems. Accordingly, this section analyzes examples of more complicated NeSy architectures, learning mechanisms, and system designs. Moreover, we highlight the current trends and challenges for NeSy hardware-software co-designs.

A. Enhance Reasoning and Generalization

Tremendous efforts have been made to integrate the computing performance from the data-driven approach with the reasoning ability from the symbolic part and develop state-of-the-art neural-symbolic architectures for complex tasks. For the visual question-answering tasks, incorporating symbolic structures as prior knowledge offers a transparent reasoning process and delivers a memory-efficient solution for memory-intensive applications [39]. Furthermore, the visual concept-metaconcept learner proposes to jointly learn the concept (related to the vector embedding) and the metaconcept (related to the neural predictor for concept relations), which enables learning from limited data and improves the generalization ability to unseen data [40]. Additionally, for abstract reasoning, the innovative approach of neuro-symbolic probabilistic abduction and execution learning emulates the cognitive manipulation of objects through a probabilistic scene representation. This strategy yields enhanced performance in cross-configuration generalization and empowers the system to not only select the most likely answer from a candidate pool but also propose answers independently [41].

To execute the NeSy architectures with reasoning and generalization capability, we need to configure and implement the submodels in the NeSy architectures. Take the neuro-symbolic dynamic reasoning model [42] as an example; In terms of computing latency, the dynamic predictor within the NeSy model is the bottleneck of the overall performance [43]. The dynamic predictor, based on PropNet [44], functions as a learned physics engine tailored for modeling and forecasting object collisions. It comprises multiple compact neural networks with fully connected and convolutional layers. To gather force propagation information for the symbolic part, executing the neural models multiple times for one frame is necessary. However, this iterative process requires redundant data transfers between the symbolic and neural parts. In general, the NeSy system consists of both the symbolic executor and the neural network block, harnessing domain knowledge and rule constraints to attain reasoning and generalization capabilities [39], [45], [46]. At the same time, the data transfer overhead arises from the separate execution of neural and symbolic components on GPUs and CPUs, posing hardware design challenges in terms of efficiency. Therefore, it is essential to explore emerging hardware systems to reduce the data transfer overhead.

B. Enable Compositionality

Vector symbolic architectures (VSAs) or hyperdimensional computing (HDC) are computational models that harness

high-dimensional distributed vectors and leverage algebraic properties of their operations [47]. Additionally, the operations involved in VSA/HDC can help achieve compositionality. For instance, the attribute vectors in the reasoning task can be combined to formulate one object vector, and multiple object vectors can be bundled to generate a composite vector to represent the combination of objects. Even in the composite vector, the attribute can be recognized through distance comparison.

Therefore, this kind of high-dimensional representation and related operations benefits the compositional reasoning. This leads to the integration of hyperdimensional computing with neural networks to build the neuro-vector-symbolic architecture (NVSA) to perform the NeSy with outstanding probabilistic reasoning ability and remarkable low latency performance [48]. Storing and processing the substantial volume of hyper-dimensional vectors in the VSA/HDC component poses challenges. Previous approaches have developed accelerators for this VSA/HDC segment [31], [49], and can be merged with the NeSy acceleration design to achieve enhanced performance.

Moreover, compositionality provides a generic hardware efficiency potential for NeSy systems through the use of a “divide and conquer” strategy with reusable modules as building blocks. For instance, the tree-like symbolic model can be efficiently implemented via re-use of the parts of the models [50]. This approach can further facilitate higher-level abstraction, customized design, and packaging for hardware integration and system design.

C. Handle Uncertainty

Many real-world reasoning and control tasks encounter the challenging hurdle of uncertainty arising from either information gaps and/or the environment’s inherent stochastic characteristics. For safety-critical missions under uncertain conditions, a probabilistic approach is essential. Here, relying on probabilistic inference offers a systematic solution [51]. To enable probabilistic inference and handle uncertainty, researchers aim to extend the deterministic primitives to probabilistic versions and find efficient inference algorithms.

Specifically, probabilistic primitives generalize the functions in reasoning primitives to probabilistic functions. This is achieved by replacing a function relation with a conditional probability distribution. Conceptually, for every reasoning or control primitive, the probabilistic counterpart can be developed and calculated. As a result, a large network of complex relations can be specified by very few atomic functions and combinators under the connectionist probabilistic program (CPP) framework [52], as illustrated in Fig. 3. Furthermore, the reasoning task is formally equivalent to computing the value of conditional probability distribution of any output-function-input triplets. For atomic functions, the inference is as trivial as a database query. For composite functions, exact inference requires expensive summation, which grows exponentially with respect to the number of involved atomic functions. At this stage, an efficient approximation of probabilistic function calculation should be incorporated [53].

Handling uncertainty poses challenges for hardware designs due to the stochastic calculation involved. For the conditional

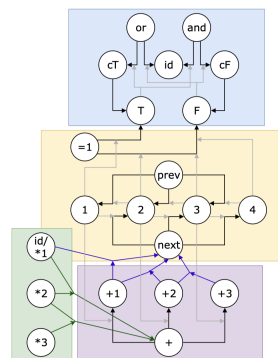


Fig. 3. A connection probabilistic program example of Boolean logic and basic arithmetic [52]. Blue region: Boolean logic; Yellow region: Counting logic; Purple region: Integer addition; Green region: Integer multiplication.

probabilistic calculation, true random number generator and pseudo-random number generator designs are considered as potential candidates [54], [55]. Furthermore, the intrinsic randomness within the hardware system can be leveraged to generate data points that follow a specific probabilistic distribution. For instance, using ReRAM cells with distinct set current values can stimulate a targeted distribution [56]. This approach takes advantage of intrinsic “non-ideal” programming noise to reconstruct an “ideal” distribution. Likewise, the probability distribution of the signal-amplified spontaneous emission (signal-ASE) beat noise can be controlled and harnessed to sample the weights in Bayesian Neural Networks [57]. Additionally, this approach efficiently supports the uncertainty generalization and stochastic inference.

D. Improve System Efficiency

In the context of the six types of NeSy designs discussed in Section II, both the neural and symbolic components play crucial roles and are integrated into the system. In general, the quantization and pruning with software-based methods [58], [59] can help reduce the memory and data communication overhead and contribute to improved system efficiency. Furthermore, we need to build hardware-software co-designs after analyzing the specific requirements of each workload or application. For instance, recent studies show that the symbolic part is comparatively lightweight compared with the neural part (considered as the direct benefit of the symbolic part). Consequently, symbolic reasoning design can require limited memory and computation resources [39], opening up the possibility of efficiency gain. Moreover, NeSy systems leverage symbolic-encoded knowledge, enabling the creation of smaller neural networks that are as generalizable as large models (considered as the indirect benefit of the symbolic part). This approach has also garnered significant attention within the machine learning community [60]–[65]. Besides, smaller NNs are highly advantageous for enhancing hardware efficiency. Therefore, the existing neural network efficiency improvement approaches can be leveraged to improve the NeSy system efficiency. While using reusable modules as building units is a generic property, researchers should prioritize exploring integrated heterogeneous systems to fully

TABLE I
DESIGN SPACE OF NeSy SYSTEMS.

Attribute	Details
NeSy Type	Type I - VI as described in Section II
Computation	GEMM, CONV, FC, Sparse MM
Technology	SRAM, DRAM, Flash, ReRAM, FeFET
Specifics	Technology Node, Integration, Interconnect
Evaluation	Area, Power
Target	Latency, Throughput, Energy, Capability

harness this property’s potential, particularly considering the inherent lightweight symbolic modules. In summary, combining NeSy algorithms with hardware-software co-designs holds significant promise for boosting system efficiency.

NeSy architectures are data-efficient, meaning they achieve promising performance with a limited amount of data [40], [66], [67]. However, the NeSy system efficiency suffers from the heavyweight neural computation part rather than the symbolic part [43]. Therefore, improving the efficiency of General Matrix Multiplication (GEMM), Convolution (CONV), and attention-based computation plays a vital role in improving NeSy system’s overall efficiency. For reasoning tasks, the GEMM calculation requires irregular and sparse matrix, and thus, flexible and scalable accelerators should be developed to enable a high utilization rate of processing element [68]. Moreover, for CONV computation, in-memory computing based on emerging technologies also requires increased parallelism. For instance, inter-layer parallelism and intra-layer parallelism are investigated to enhance the energy efficiency for in-memory CONV computing [69]. The sparsity pattern in the neural network can be investigated, and dedicated hardware design for sparse computation can benefit the system efficiency [70]. Additionally, recent advancements in near-memory and in-memory computing [71]–[76] provide valuable insights on leveraging data movement and improving system efficiency.

NeSy system has adequate space for exploration and optimization due to the available design types and flexible components. We believe that the neuro-symbolic architecture search, following the philosophy of neural architecture search, can be introduced to jointly optimize reasoning ability, generalization performance, and system efficiency. Furthermore, the existing NeSy model can be expanded to tackle more complex challenges, and thus, an automatic layer growing strategy should be incorporated to enable powerful NeSy systems [77].

E. Demonstrate Full-Stack NeSy Systems

As presented in Section II, the NeSy AI covers six types of integration. Hence, the system design should account for the diverse scenarios represented by the six available types, offering the necessary support for various data processing requirements. In Table I, we include the design configurations for full-stack NeSy systems. In the computing platform part, we include the SRAM/ReRAM designs for digital/analog in-memory computing, FeFETs for in-memory search and hyperdimensional computing, systolic arrays for GEMM, coupled oscillator and logic for reasoning tasks. This complicated NeSy system should also have an effective modeling

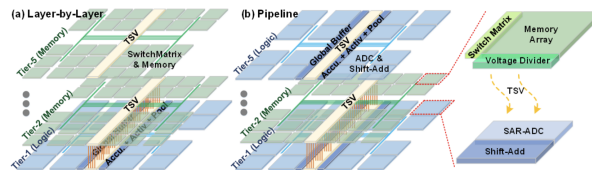


Fig. 4. Floorplan of heterogeneous 3D multitier CIM accelerator [80].

framework as prior work for full-stack DNN system [78], [79]. This cognitive NeSy system covers various computing components for different tasks and supports efficient resource disaggregation, and thus should take advantage of advanced packaging, such as heterogeneous 3D integration with multiple chiplets that are specialized for certain primitives in the NeSy system [80] (illustrated in Fig. 4). The attributes in the above table create a broad and diverse range of design possibilities. To find the optimal design, researchers should investigate the multi-objective and combinational optimization problems. The coupled-oscillator-based platform can also be leveraged to provide a quick solver for design optimization and facilitate the agile design of full-stack cognitive NeSy systems.

IV. APPLICATIONS AND FUTURE TOPICS

One trending NeSy application is natural language processing (NLP) for context understanding, extractive summarization, and semantic reasoning [81], [82]. NeSy’s ability to combine statistical patterns with structured knowledge enhances its performance in understanding and generating responses. NeSy has the potential to further extend to the spoken dialog system and improve the service quality of digital assistants.

In the field of robotics, NeSy approaches play an important role in enhancing perception, planning, and decision-making capabilities. These approaches empower robots to comprehend and navigate intricate environments by synergizing neural pattern recognition with logical reasoning. Furthermore, robot systems are expected to demonstrate generalization abilities under unseen environments [83]. This fusion of capabilities equips robots with the ability to work in real-world scenarios, contributing to their autonomy and adaptability.

Leveraging NeSy approaches in autonomous driving has the potential to enhance the creation of self-driving systems that are more dependable, contextually aware, adversarially robust, and secure. In this domain, providing high assurance when operating in contested and adversarial environments is critical. Thus, the NeSy approach can be explored to obtain such security guarantees and build trust between the NeSy system and the users, as discussed in a recent NeSy survey [84].

V. CONCLUSIONS

This paper extensively explores the limitations inherent in neural networks and emphasizes the crucial role of neuro-symbolic artificial intelligence. Additionally, it presents the diverse NeSy architecture and offers valuable insights into ongoing research trends, with a focus on the hardware-software co-design aspect. In our following work, we plan to build a heterogeneous system, incorporating both CPUs and GPUs as computing units and aligning with the task requirements, to fully exploit the potential of compositional NeSy systems.

REFERENCES

- [1] D. Kiela *et al.*, “Dynabench: Rethinking benchmarking in nlp,” in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021*, pp. 4110–4124, ACL, 2021.
- [2] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, “Benchmark analysis of representative deep neural network architectures,” *IEEE access*, vol. 6, pp. 64270–64277, 2018.
- [3] T. Brown *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
- [4] J. Yang, K. Zhou, Y. Li, and Z. Liu, “Generalized out-of-distribution detection: A survey,” *arXiv preprint arXiv:2110.11334*, 2021.
- [5] L. Floridi and M. Chiriatti, “Gpt-3: Its nature, scope, limits, and consequences,” *Minds and Machines*, vol. 30, pp. 681–694, 2020.
- [6] S. Bader and P. Hitzler, “Dimensions of neural-symbolic integration—a structured survey,” in *We Will Show Them! Essays in Honour of Dov Gabbay, Volume One*, pp. 167–194, College Publications, 2005.
- [7] A. d. Garcez, M. Gori, L. C. Lamb, L. Serafini, M. Spranger, and S. N. Tran, “Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning,” *FLAP*, vol. 6, no. 4, pp. 611–632, 2019.
- [8] E. Tsamoura, T. Hospedales, and L. Michael, “Neural-symbolic integration: A compositional perspective,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, pp. 5051–5060, 2021.
- [9] X. Chen, C. Liang, A. W. Yu, D. Song, and D. Zhou, “Compositional generalization via neural-symbolic stack machines,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 1690–1701, 2020.
- [10] L. C. Lamb, A. Garcez, M. Gori, M. Prates, P. Avelar, and M. Vardi, “Graph neural networks meet neural-symbolic computing: A survey and perspective,” in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pp. 4877–4884, ijcai.org, 2020.
- [11] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, “Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,” *IEEE journal of solid-state circuits*, vol. 52, no. 1, pp. 127–138, 2016.
- [12] M. Hu, H. Li, Q. Wu, G. S. Rose, and Y. Chen, “Memristor crossbar based hardware realization of bsb recall function,” in *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, IEEE, 2012.
- [13] X. Yang *et al.*, “Multi-objective optimization of reram crossbars for robust dnn inferencing under stochastic noise,” in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pp. 1–9, IEEE, 2021.
- [14] H. Kautz, “The third ai summer: Aai robert s. engelmore memorial lecture,” *AI Magazine*, vol. 43, no. 1, pp. 105–125, 2022.
- [15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [16] X. Yang, B. Yan, H. Li, and Y. Chen, “Retransformer: Reram-based processing-in-memory architecture for transformer acceleration,” in *Proceedings of the 39th International Conference on Computer-Aided Design*, pp. 1–9, 2020.
- [17] M. Kang, H. Shin, J. Kim, and L.-S. Kim, “Mgen: A framework for energy-efficient in-ram acceleration of multi-task bert,” *IEEE Transactions on Computers*, 2023.
- [18] P. Qi and p. y. o. others, booktitle=2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD), “Accelerating framework of transformer by hardware design and model compression co-optimization.”
- [19] S. Tuli and N. K. Jha, “Transcode: Co-design of transformers and accelerators for efficient training and inference,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023.
- [20] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, “Mastering the game of go with deep neural networks and tree search,” *nature*, vol. 529, no. 7587, pp. 484–489, 2016.
- [21] F. Arabshahi, Z. Lu, S. Singh, and A. Anandkumar, “Memory augmented recursive neural networks,” *arXiv*, 2019.
- [22] A. d. Garcez *et al.*, “Neural-symbolic learning and reasoning: contributions and challenges,” in *2015 AAAI Spring Symposium Series*, 2015.
- [23] J. Zhang, “Graph-toolformer: To empower llms with graph reasoning ability via prompt augmented by chatgpt,” *arXiv preprint arXiv:2304.11116*, 2023.
- [24] L. Serafini and A. d. Garcez, “Logic tensor networks: Deep learning and logical reasoning from data and knowledge,” in *Proceedings of the 11th International Workshop on Neural-Symbolic Learning and Reasoning (NeSy’16)*, vol. 1768, CEUR-WS.org, 2016.
- [25] I. Karageorgos *et al.*, “Hardware-software co-design for brain-computer interfaces,” in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pp. 391–404, IEEE, 2020.
- [26] A. Kortylewski, J. He, Q. Liu, and A. L. Yuille, “Compositional convolutional neural networks: A deep architecture with innate robustness to partial occlusion,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8940–8949, 2020.
- [27] W. Shen *et al.*, “Interpretable compositional convolutional neural networks,” in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021*, pp. 2971–2978, ijcai.org, 2021.
- [28] W. Zhong *et al.*, “Disentangling reasoning capabilities from language models with compositional reasoning transformers,” in *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 7587–7600, ACL, 2023.
- [29] X. Yang, B. Taylor, A. Wu, Y. Chen, and L. O. Chua, “Research progress on memristor: From synapses to computing systems,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 5, pp. 1845–1857, 2022.
- [30] X. Yin, M. Niemier, and X. S. Hu, “Design and benchmarking of ferroelectric fet based team,” in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pp. 1444–1449, IEEE, 2017.
- [31] A. Kazemi, M. M. Sharifi, Z. Zou, M. Niemier, X. S. Hu, and M. Imani, “Mimhd: Accurate and efficient hyperdimensional inference using multi-bit in-memory computing,” in *2021 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pp. 1–6, IEEE, 2021.
- [32] I. Ahmed, P.-W. Chiu, W. Moy, and C. H. Kim, “A probabilistic compute fabric based on coupled ring oscillators for solving combinatorial optimization problems,” *IEEE Journal of Solid-State Circuits*, vol. 56, no. 9, pp. 2870–2880, 2021.
- [33] G. Csaba and W. Porod, “Coupled oscillators for computing: A review and perspective,” *Applied physics reviews*, vol. 7, no. 1, 2020.
- [34] W. Moy, I. Ahmed, P.-w. Chiu, J. Moy, S. S. Sapatnekar, and C. H. Kim, “A 1,968-node coupled ring oscillator circuit for combinatorial optimization problem solving,” *Nature Electronics*, vol. 5, no. 5, pp. 310–317, 2022.
- [35] H. Lo, W. Moy, H. Yu, S. Sapatnekar, and C. H. Kim, “An ising solver chip based on coupled ring oscillators with a 48-node all-to-all connected array architecture,” *Nature Electronics*, pp. 1–8, 2023.
- [36] N. Funk, G. Chalvatzaki, B. Belousov, and J. Peters, “Learn2assemble with structured representations and search for robotic architectural construction,” in *Conference on Robot Learning*, pp. 1401–1411, PMLR, 2022.
- [37] H. F. Tsoi *et al.*, “Symbolic regression on fpgas for fast machine learning inference,” *arXiv preprint arXiv:2305.04099*, 2023.
- [38] S. Sharan, W. Zheng, K.-F. Hsu, J. King, A. Chen, and Z. Wang, “Symbolic distillation for learned tcp congestion control,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 10684–10695, 2022.
- [39] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. Tenenbaum, “Neural-symbolic vqa: Disentangling reasoning from vision and language understanding,” *Advances in neural information processing systems*, vol. 31, 2018.
- [40] C. Han, J. Mao, C. Gan, J. Tenenbaum, and J. Wu, “Visual concept-metaconcept learning,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [41] C. Zhang, B. Jia, S.-C. Zhu, and Y. Zhu, “Abstract spatial-temporal reasoning via probabilistic abduction and execution,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9736–9746, 2021.
- [42] K. Yi *et al.*, “Clevrer: Collision events for video representation and reasoning,” in *8th International Conference on Learning Representations, ICLR 2020*, OpenReview.net, 2020.
- [43] Z. Susskind, B. Arden, L. K. John, P. Stockton, and E. B. John, “Neuro-symbolic ai: An emerging class of ai workloads and their characterization,” *arXiv preprint arXiv:2109.06133*, 2021.
- [44] Y. Li, J. Wu, J.-Y. Zhu, J. B. Tenenbaum, A. Torralba, and R. Tedrake, “Propagation networks for model-based control under partial observation,” in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 1205–1211, IEEE, 2019.
- [45] X. Chen, C. Liang, A. W. Yu, D. Zhou, D. Song, and Q. V. Le, “Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension,” in *International Conference on Learning Representations*, 2019.

- [46] N. Díaz-Rodríguez *et al.*, “Explainable neural-symbolic learning (x-nesyl) methodology to fuse deep learning representations with expert knowledge graphs: The monumai cultural heritage use case,” *Information Fusion*, vol. 79, pp. 58–83, 2022.
- [47] D. Kleyko *et al.*, “Vector symbolic architectures as a computing framework for emerging hardware,” *Proceedings of the IEEE*, vol. 110, no. 10, pp. 1538–1571, 2022.
- [48] M. Hersche, M. Zeqiri, L. Benini, A. Sebastian, and A. Rahimi, “A neuro-vector-symbolic architecture for solving raven’s progressive matrices,” *Nature Machine Intelligence*, vol. 5, no. 4, pp. 363–375, 2023.
- [49] Y. Kim, M. Imani, N. Moshiri, and T. Rosing, “Geniehd: Efficient dna pattern matching accelerator using hyperdimensional computing,” in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 115–120, IEEE, 2020.
- [50] A. Kortylewski *et al.*, “Greedy structure learning of hierarchical compositional models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11612–11621, 2019.
- [51] X. Yang, H. Zhang, G. Qi, and J. Cai, “Causal attention for vision-language tasks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9847–9857, 2021.
- [52] X. Qiao and H. Li, “On a new type of neural computation for probabilistic symbolic reasoning,” in *International Joint Conference on Neural Networks, IJCNN 2023*, pp. 1–9, IEEE, 2023.
- [53] F. Wood, J. W. Meent, and V. Mansinghka, “A new approach to probabilistic programming inference,” in *Artificial intelligence and statistics*, pp. 1024–1032, PMLR, 2014.
- [54] Y. Wang, W. Wen, H. Li, and M. Hu, “A novel true random number generator design leveraging emerging memristor technology,” in *Proceedings of the 25th edition on Great Lakes Symposium on VLSI*, pp. 271–276, 2015.
- [55] X. Yang, H. Yang, J. R. Doppa, P. P. Pande, K. Chakrabarty, and H. Li, “Essence: Exploiting structured stochastic gradient pruning for endurance-aware rram-based in-memory training systems,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022.
- [56] T. Dalgaty, E. Vianello, and D. Querlioz, “Harnessing intrinsic memristor randomness with bayesian neural networks,” in *2021 International Conference on IC Design and Technology (ICIDT)*, pp. 1–4, IEEE, 2021.
- [57] C. Wu, X. Yang, Y. Chen, and M. Li, “Photonic bayesian neural network using programmed optical noises,” *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 29, no. 2: Optical Computing, pp. 1–6, 2022.
- [58] H. Yang, X. Yang, N. Z. Gong, and Y. Chen, “Hero: Hessian-enhanced robust optimization for unifying and improving generalization and quantization performance,” in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pp. 25–30, 2022.
- [59] H. Yang, L. Duan, Y. Chen, and H. Li, “Bsq: Exploring bit-level sparsity for mixed-precision neural network quantization,” *arXiv preprint arXiv:2102.10462*, 2021.
- [60] P. West *et al.*, “Symbolic knowledge distillation: from general language models to commonsense models,” in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022*, pp. 4602–4625, ACL, 2022.
- [61] C. Bhagavatula *et al.*, “I2d2: Inductive knowledge distillation with neurologic and self-imitation,” in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9614–9630, ACL, 2023.
- [62] P. Howard, J. Wang, V. Lal, G. Singer, Y. Choi, and S. Swamydiptra, “Neurocomparatives: Neuro-symbolic distillation of comparative knowledge,” *arXiv preprint arXiv:2305.04978*, 2023.
- [63] J. Jung *et al.*, “Impossible distillation: from low-quality model to high-quality dataset & model for summarization and paraphrasing,” *arXiv preprint arXiv:2305.16635*, 2023.
- [64] W. Zhou, R. L. Bras, and Y. Choi, “Commonsense knowledge transfer for pre-trained language models,” in *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 5946–5960, ACL, 2023.
- [65] L. H. Li, J. Hessel, Y. Yu, X. Ren, K.-W. Chang, and Y. Choi, “Symbolic chain-of-thought distillation: Small models can also” think” step-by-step,” in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023*, pp. 2665–2679, ACL, 2023.
- [66] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, “The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision,” in *7th International Conference on Learning Representations, ICLR 2019, OpenReview.net*, 2019.
- [67] Q. Li, S. Huang, Y. Hong, Y. Chen, Y. N. Wu, and S.-C. Zhu, “Closed loop neural-symbolic learning via integrating neural perception, grammar parsing, and symbolic reasoning,” in *International Conference on Machine Learning*, pp. 5884–5894, PMLR, 2020.
- [68] E. Qin *et al.*, “Sigma: A sparse and irregular gemm accelerator with flexible interconnects for dnn training,” in *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 58–70, IEEE, 2020.
- [69] L. Song, X. Qian, H. Li, and Y. Chen, “Pipelayer: A pipelined rram-based accelerator for deep learning,” in *2017 IEEE international symposium on high performance computer architecture (HPCA)*, pp. 541–552, IEEE, 2017.
- [70] S. Li, E. Hanson, X. Qian, H. H. Li, and Y. Chen, “Escalate: Boosting the efficiency of sparse cnn accelerator with kernel decomposition,” in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 992–1004, 2021.
- [71] O. Mutlu, “Processing data where it makes sense in modern computing systems: Enabling in-memory computation,” in *Proceedings of the 2019 on Great Lakes Symposium on VLSI*, pp. 5–6, 2019.
- [72] S. Yu, H. Jiang, S. Huang, X. Peng, and A. Lu, “Compute-in-memory chips for deep learning: Recent trends and prospects,” *IEEE circuits and systems magazine*, vol. 21, no. 3, pp. 31–56, 2021.
- [73] A. Kazemi *et al.*, “Achieving software-equivalent accuracy for hyperdimensional computing with ferroelectric-based in-memory computing,” *Scientific reports*, vol. 12, no. 1, p. 19201, 2022.
- [74] B. Yan *et al.*, “Resistive memory-based in-memory computing: from device and large-scale integration system perspectives,” *Advanced Intelligent Systems*, vol. 1, no. 7, p. 1900068, 2019.
- [75] F. Chen, L. Song, and Y. Chen, “Regan: A pipelined rram-based accelerator for generative adversarial networks,” in *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 178–183, IEEE, 2018.
- [76] Q. Zheng *et al.*, “Mobilattice: a depth-wise dcnn accelerator with hybrid digital/analog nonvolatile processing-in-memory block,” in *Proceedings of the 39th International Conference on Computer-Aided Design*, pp. 1–9, 2020.
- [77] W. Wen, F. Yan, Y. Chen, and H. Li, “Autogrow: Automatic layer growing in deep convolutional networks,” in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 833–841, 2020.
- [78] X. Peng, S. Huang, Y. Luo, X. Sun, and S. Yu, “Dnn+ neurosim: An end-to-end benchmarking framework for compute-in-memory accelerators with versatile device technologies,” in *2019 IEEE international electron devices meeting (IEDM)*, pp. 32–5, IEEE, 2019.
- [79] X. Peng *et al.*, “Benchmarking monolithic 3d integration for compute-in-memory accelerators: overcoming adc bottlenecks and maintaining scalability to 7nm or beyond,” in *2020 IEEE International Electron Devices Meeting (IEDM)*, pp. 30–4, IEEE, 2020.
- [80] X. Peng, A. Kaul, M. S. Bakir, and S. Yu, “Heterogeneous 3-d integration of multitier compute-in-memory accelerators: An electrical-thermal co-design,” *IEEE Transactions on Electron Devices*, vol. 68, no. 11, pp. 5598–5605, 2021.
- [81] H. Zhang, X. Liu, and J. Zhang, “Extractive summarization via chatgpt for faithful summary generation,” *arXiv preprint arXiv:2304.04193*, 2023.
- [82] A. Rajasekharan, Y. Zeng, P. Padalkar, and G. Gupta, “Reliable natural language understanding with large language models and answer set programming,” *arXiv preprint arXiv:2302.03780*, 2023.
- [83] A. K. Bozkurt, Y. Wang, and M. Pajic, “Secure planning against stealthy attacks via model-free reinforcement learning,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10656–10662, IEEE, 2021.
- [84] A. Sheth, K. Roy, and M. Gaur, “Neurosymbolic artificial intelligence (why, what, and how),” *IEEE Intelligent Systems*, vol. 38, no. 3, pp. 56–62, 2023.