

Dynamic DNNs and Runtime Management for Efficient Inference on Mobile/Embedded Devices

DATE PhD Forum 2024

Lei Xun

School of Electronics & Computer Science
University of Southampton, UK
l.xun@soton.ac.uk

Advisor: Jonathon Hare, Geoff V. Merrett

School of Electronics & Computer Science
University of Southampton, UK
{jsh2, gvm}@ecs.soton.ac.uk

I. INTRODUCTION

Deep neural network (DNN) inference is increasingly being executed on mobile and embedded platforms [6, 40] due to several key advantages in latency [26, 36], privacy [9] and always-on availability [26, 24]. However, efficient DNN deployment on mobile and embedded platforms is challenging due to limited computing resources [21, 1]. Although many hardware accelerators [10, 8, 18, 17, 14] and static model compression methods [35, 11] were proposed by previous works, at system runtime, multiple applications are typically executed concurrently and compete for hardware resources. This raises two main challenges:

- **Runtime Hardware Availability:** Modern System-on-Chips (SoCs), comprising CPUs, GPUs, and NPUs, face challenges due to the varying availability of hardware resources at runtime. This fluctuation arises from different core combinations and changes in voltage and clock frequencies. While static model compression can initially optimize DNN models to fit on targeted hardware and meet performance goals, the issue is the unpredictable availability of these resources during runtime. Different and dynamic workloads on SoCs make it hard to consistently meet performance targets, as the hardware resources available to DNN models change and are unknown during the initial compression [33, 12, 2, 3].
- **Runtime Application Variability:** A single DNN model, such as large language models (LLMs), can serve as the backbone for various applications like translation, text generation, and ChatBot, each requiring different performance trade-offs. For example, a ChatBot needs LLMs to have low latency for quick responses, whereas translation and text generation need LLMs to focus on accuracy. These performance targets can also change through user settings/preferences at runtime, posing a significant challenge in the design stage. The current solution of using multiple static models with different performance trade-offs is not feasible for mobile and embedded platforms due to limited memory resources. An ideal approach would be a single, adaptable DNN model that dynamically adjusts its performance trade-offs to meet the specific requirements of each application and user at runtime.

II. RUNTIME SYSTEM-LEVEL PERFORMANCE TRADE-OFF MANAGEMENT

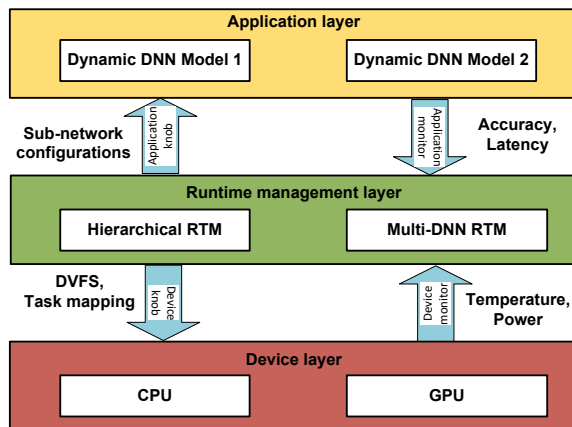


Fig. 1. The high-level diagram of the proposed runtime system, which contains three abstract layers that are connected through knobs and monitors.

Previous works have addressed aforementioned challenges through dynamic neural networks that contain sub-networks with different performance trade-offs [22, 28, 39, 38, 37, 34] or runtime hardware resource management [16, 4, 20, 19, 27].

We proposed a combined method [32, 33], a system was developed for DNN performance trade-off management, combining the runtime trade-off opportunities in both algorithms and hardware (Fig 1). The runtime system contains three abstract layers which are connected through knobs and monitors. Application layers contain multiple concurrent dynamic neural networks. The device layer is mobile and embedded heterogeneous SoCs. The runtime management layer is the highest-level layer with central control algorithms for tuning both application knobs (i.e. dynamic DNN sub-networks), and device knobs (e.g. DVFS and task mapping) to meet dynamically changing application performance targets and hardware constraints which are both monitored in real-time. The key contribution of our works [32, 33, 30, 31, 29] is system-level performance trade-off management, the application layer and runtime management layer are co-designed based on the characterisation of the underlay hardware platforms to boost the space and granularity of performance trade-offs. This is different to the previous works which only focus on standalone system components while isolating others.

III. DYNAMIC SUPER-NETWORKS

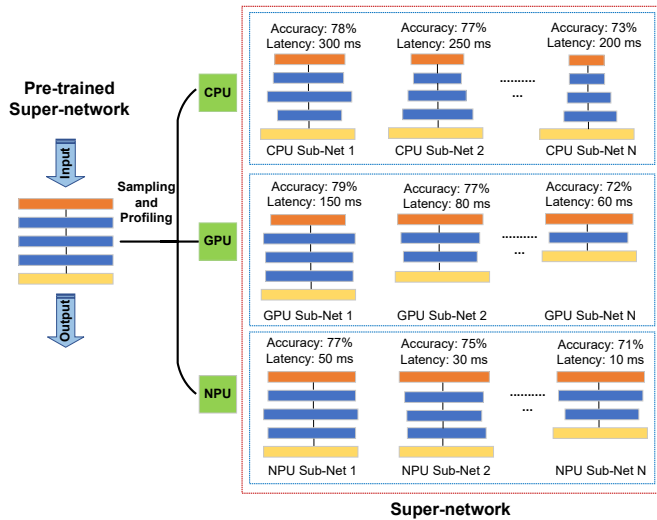


Fig. 2. Dynamic Super-network. It samples and combines different efficient sub-network libraries from backbone super-networks for all heterogeneous cores, and to build dynamic neural networks without training.

We co-designed novel dynamic neural networks to maximise system-level performance and energy efficiency. We identified three common issues with all previous dynamic neural networks: (1) significant training time cost, (2) conflict with the SOTA neural architecture search (NAS) pipeline, and (3) inference inefficiency on heterogeneous hardware.

To address these problems, we proposed Dynamic Super-network [13, 15], as shown in Fig 2, which directly samples efficient sub-networks on (near) the performance trade-off Pareto-front from the backbone super-networks to create a library and build dynamic neural networks directly without any training process (i.e. only sampling and performance profiling). The sampling and profiling process is repeated for different heterogeneous cores (e.g. CPU, GPU, NPU) on SoCs, this is because the most efficient DNN model architectures for different cores are different, and often contrary [7]. Super-networks have the flexibility to generate very different sub-network model architectures, whereas previous dynamic neural networks can scale either layer-wise [23, 25] or channel-wise [22, 28, 39, 38, 37, 34]. In the end, we obtained different libraries of efficient sub-networks for all heterogeneous cores using only one set of model weights (i.e. the weights of the full super-network) which are stored using on-chip memory. Each library has multiple sub-networks with different performance trade-offs, which all share their weights with each other, and with sub-networks in other libraries, as well as the super-network. Each library is efficient on the corresponding cores, and when moving the DNN model between cores (e.g. from GPU to CPU), a different sub-network from the target hardware (i.e. CPU) model library is selected, and this is done by applying masks to the backbone super-network so it can be partially executed, this act as model/weights selection from the backbone model rather than full model/weights switching.

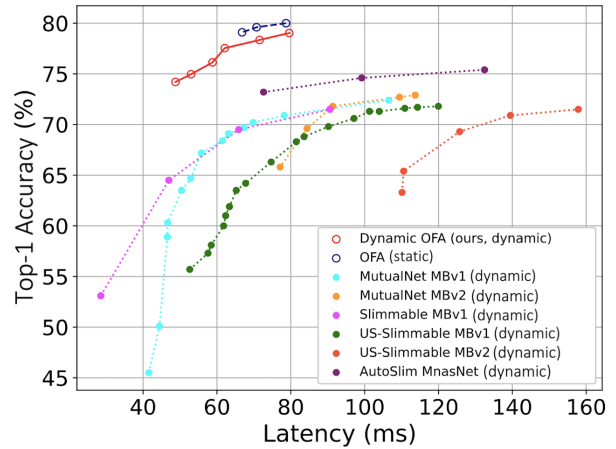


Fig. 3. The performance trade-offs between ImageNet Top-1 accuracy and latency of our Dynamic OFA model [13] on the GPU of Jetson Xavier NX platform, comparing against SOTA static OFA backbone model [5] and dynamic DNN models [39, 38, 37, 34]. Dynamic OFA model is 2.4x faster (at similar accuracy) or has 5.1% higher Top-1 ImageNet accuracy (at similar latency) than AutoSlim-MnasNet [37].

We illustrated the Dynamic Super-network approach through a dynamic version of ‘once-for-all network [5]’ (namely Dynamic-OFA), which can scale the ConvNet architecture to fit heterogeneous computing resources efficiently [13] and has good generalisation for different model architectures such as Transformer [15]. As shown in Fig 3, compared to the SOTA dynamic neural networks [39, 38, 37, 34], our experimental results using ImageNet on the GPU of Jetson Xavier NX show that the Dynamic-OFA is 2.4x faster for similar ImageNet Top-1 accuracy, or 5.1% higher accuracy at similar latency. We also see a similar level of improvement on the CPU. Dynamic OFA has a slightly lower accuracy than its fine-tuned static versions (i.e. individual models with separate weights), but these models have significant memory overhead and runtime switching costs.

IV. RUNTIME RESOURCE MANAGEMENT FOR DYNAMIC SUPER-NETWORKS

We illustrated the runtime management approach through a hierarchical runtime resource manager that tunes both dynamic neural networks and DVFS at runtime to meet the hardware constraints (e.g. power consumption), and algorithm performance targets (e.g. accuracy, latency). Compared with the Linux DVFS governor schedutil, our runtime approach achieves up to a 19% energy reduction and a 9% latency reduction in single model deployment scenario, and an 89% energy reduction and a 23% latency reduction in a two concurrent model deployment scenario.

ACKNOWLEDGMENTS

These works were supported by the Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/S030069/1. Experimental data can be found at: <https://doi.org/10.5258/SOTON/D1804>. Code is available open-source at <https://github.com/UoS-EEC/DynamicOFA>.

REFERENCES

- [1] M. Almeida, S. Laskaridis, A. Mehrotra, L. Dudziak, I. Leontiadis, and N. D. Lane. "Smart at what cost? Characterising Mobile Deep Neural Networks in the wild". In: *ACM Internet Measurement Conference*. 2021.
- [2] H. Bai, M. Cao, P. Huang, and J. Shan. "BatchQuant: Quantized-for-all Architecture Search with Robust Quantizer". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021.
- [3] Y. Bai, L. Chen, S. Ren, and J. Xu. "Automated Customization of On-Device Inference for Quality-of-Experience Enhancement". In: *IEEE Transactions on Computers (TC)* (2022).
- [4] K. R. Basireddy, A. K. Singh, B. M. Al-Hashimi, and G. V. Merrett. "AdaMD: Adaptive Mapping and DVFS for Energy-Efficient Heterogeneous Multicores". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* (2019).
- [5] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han. "Once-for-All: Train One Network and Specialize it for Efficient Deployment". In: *International Conference on Learning Representations (ICLR)*. 2020.
- [6] H. Cai, J. Lin, Y. Lin, Z. Liu, H. Tang, H. Wang, et al. "Enable Deep Learning on Mobile Devices: Methods, Systems, and Applications". In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 27.3 (2022), pp. 1–50.
- [7] H. Cai, L. Zhu, and S. Han. "ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware". In: *International Conference on Learning Representations (ICLR)*. 2019.
- [8] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze. "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks". In: *IEEE Journal of Solid-State Circuits* 52.1 (2016), pp. 127–138.
- [9] X. Dai, I. Spasić, B. Meyer, S. Chapman, and F. Andres. "Machine Learning on Mobile: An On-device Inference App for Skin Cancer Detection". In: *International Conference on Fog and Mobile Edge Computing (FMEC)*. 2019.
- [10] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, et al. "EIE: Efficient Inference Engine on Compressed Deep Neural Network". In: *International Symposium on Computer Architecture (ISCA)*. 2016.
- [11] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han. "AMC: AutoML for Model Compression and Acceleration on Mobile Devices". In: *European Conference on Computer Vision (ECCV)*. 2018.
- [12] H. Hoffmann, A. Jantsch, and N. D. Dutt. "Embodied Self-Aware Computing Systems". In: *Proceedings of the IEEE* 108.7 (2020), pp. 1027–1046.
- [13] W. Lou, L. Xun, A. Sabet, J. Bi, J. Hare, and G. V. Merrett. "Dynamic-OFA: Runtime DNN Architecture Switching for Performance Scaling on Heterogeneous Embedded Platforms". In: *Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2021.
- [14] A. Moss, H. Lee, L. Xun, C. Min, F. Kawsar, and A. Montanari. "Ultra-low Power DNN Accelerators for IoT: Resource Characterization of the MAX78000". In: *SenSys Conference 4th Workshop on AIChallengeIoT*. 2022.
- [15] H. Parry, L. Xun, A. Sabet, J. Bi, J. Hare, and G. V. Merrett. "Dynamic Transformer for Efficient Machine Translation on Embedded Devices". In: *ACM/IEEE 3rd Workshop on Machine Learning for CAD (MLCAD)*. 2021.
- [16] B. K. Reddy, A. K. Singh, D. Biswas, G. V. Merrett, and B. M. Al-Hashimi. "Inter-cluster Thread-to-core Mapping and DVFS on Heterogeneous Multi-cores". In: *IEEE Transactions on Multi-Scale Computing Systems* 4.3 (2017), pp. 369–382.
- [17] M. Safarpour, T. Z. Deng, J. Massingham, L. Xun, M. Sabokrou, and O. Silvén. "Low-Voltage Energy Efficient Neural Inference by Leveraging Fault Detection Techniques". In: *Nordic Circuits and Systems Conference (NorCAS)*. 2021.
- [18] M. Safarpour, L. Xun, G. V. Merrett, and O. Silvén. "A High-Level Approach for Energy Efficiency Improvement of FPGAs by Voltage Trimming". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* (2021).
- [19] A. K. Singh, K. R. Basireddy, A. Prakash, G. V. Merrett, and B. M. Al-Hashimi. "Collaborative Adaptation for Energy-Efficient Heterogeneous Mobile SoCs". In: *IEEE Transactions on Computers (TC)* 69.2 (2019), pp. 185–197.
- [20] A. K. Singh, A. Prakash, K. R. Basireddy, G. V. Merrett, and B. M. Al-Hashimi. "Energy-Efficient Run-Time Mapping and Thread Partitioning of Concurrent OpenCL Applications on CPU-GPU MPSoCs". In: *ACM Transactions on Embedded Computing Systems (TECS)* 16.5s (2017), pp. 1–22.
- [21] V. Sze, Y.-H. Chen, J. Emer, A. Suleiman, and Z. Zhang. "Hardware for Machine Learning: Challenges and Opportunities". In: *Custom Integrated Circuits Conference (CICC)*. 2017.
- [22] H. Tann, S. Hashemi, R. Bahar, and S. Reda. "Runtime Configurable Deep Neural Networks for Energy-Accuracy Trade-off". In: *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. 2016.
- [23] S. Teerapittayanon, B. McDanel, and H.-T. Kung. "BranchyNet: Fast Inference via Early Exiting from Deep Neural Networks". In: *International Conference on Pattern Recognition (ICPR)*. 2016.
- [24] S. I. Venieris, M. Almeida, R. Lee, and N. D. Lane. "NAWQ-SR: A Hybrid-Precision NPU Engine for Efficient On-Device Super-Resolution". In: *IEEE Transactions on Mobile Computing* (2023).
- [25] X. Wang, F. Yu, Z.-Y. Dou, T. Darrell, and J. E. Gonzalez. "SkipNet: Learning Dynamic Routing in Convolutional Networks". In: *European Conference on Computer Vision (ECCV)*. 2018.
- [26] C.-J. Wu, D. Brooks, K. Chen, D. Chen, S. Choudhury, M. Dulkan, et al. "Machine Learning at Facebook: Understanding Inference at the Edge". In: *International Symposium on High Performance Computer Architecture (HPCA)*. 2019.
- [27] Z. Xu, D. Yang, C. Yin, J. Tang, Y. Wang, and G. Xue. "A Co-Scheduling Framework for DNN Models on Mobile and Edge Devices with Heterogeneous Hardware". In: *IEEE Transactions on Mobile Computing* (2021).
- [28] Z. Xu, F. Yu, C. Liu, and X. Chen. "ReForm: Static and Dynamic Resource-Aware DNN Reconfiguration Framework for Mobile Device". In: *Design Automation Conference (DAC)*. 2019.
- [29] L. Xun, J. Hare, and G. V. Merrett. "Dynamic DNNs Meet Runtime Resource Management for Efficient Heterogeneous Computing". In: *Workshop on Novel Architecture and Novel Design Automation (NANDA)*. 2023.
- [30] L. Xun, B. M. Al-Hashimi, J. Hare, and G. V. Merrett. "Runtime DNN Performance Scaling through Resource Management on Heterogeneous Embedded Platforms". In: *tinyML EMEA Technical Forum*. 2021.
- [31] L. Xun, B. M. Al-Hashimi, J. Hare, and G. V. Merrett. "Dynamic DNNs Meet Runtime Resource Management on Mobile and Embedded Platforms". In: *UK Mobile, Wearable and Ubiquitous Systems Research Symposium (MobiUK)*. 2022.
- [32] L. Xun, L. Tran-Thanh, B. M. Al-Hashimi, and G. V. Merrett. "Incremental Training and Group Convolution Pruning for Runtime DNN Performance Scaling on Heterogeneous Embedded Platforms". In: *ACM/IEEE 1st Workshop on Machine Learning for CAD (MLCAD)*. 2019.
- [33] L. Xun, L. Tran-Thanh, B. M. Al-Hashimi, and G. V. Merrett. "Optimising Resource Management for Embedded Machine Learning". In: *Design, Automation and Test in Europe Conference (DATE)*. 2020.
- [34] T. Yang, S. Zhu, M. Mendieta, P. Wang, R. Balakrishnan, M. Lee, et al. "MutualNet: Adaptive ConvNet via Mutual Learning from Different Model Configurations". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (2021).
- [35] T.-J. Yang, A. Howard, B. Chen, X. Zhang, A. Go, M. Sandler, et al. "NetAdapt: Platform-Aware Neural Network Adaptation for Mobile Applications". In: *European Conference on Computer Vision (ECCV)*. 2018.
- [36] F. Yu, S. Bray, D. Wang, L. Shangquan, X. Tang, C. Liu, et al. "Automated Runtime-Aware Scheduling for Multi-Tenant DNN Inference on GPU". In: *International Conference On Computer-Aided Design (ICCAD)*. 2021.
- [37] J. Yu and T. Huang. "AutoSlim: Towards One-Shot Architecture Search for Channel Numbers". In: *arXiv preprint arXiv:1903.11728* (2019).
- [38] J. Yu and T. S. Huang. "Universally Slimmable Networks and Improved Training Techniques". In: *International Conference on Computer Vision (ICCV)*. 2019.
- [39] J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang. "Slimmable Neural Networks". In: *International Conference on Learning Representations (ICLR)*. 2019.
- [40] T. Zhao, Y. Xie, Y. Wang, J. Cheng, X. Guo, B. Hu, et al. "A Survey of Deep Learning on Mobile Devices: Applications, Optimizations, Challenges, and Research Opportunities". In: *Proceedings of the IEEE* 110.3 (2022), pp. 334–354.