Late Breaking Results: Adaptive Ensembles of Dynamic DNNs for Collaborative Edge Inference

***Abstract*—Edge computing enables low-latency and privacy-preserving DNN inference, yet heterogeneous and dynamically changing device resources make it difficult to satisfy real-time constraints. In this paper, we present AdaEnsemble, an adaptive and collaborative ensemble inference framework that integrates Dynamic DNNs with deadline-aware scheduling. The system profiles accuracy and latency offline and selects both model widths and participating devices at runtime to maximize accuracy under a given deadline. Experiments on heterogeneous edge devices show that AdaEnsemble adapts effectively to different latency requirements and consistently outperforms the state-of-art.**

***Index Terms*—Ensemble DNNs, Distributed Inference**

I. Introduction

Compared to cloud-based processing, edge computing provides lower latency, reduced power consumption, and improved privacy [1]. However, edge devices exhibit limited and heterogeneous resources that may fluctuate due to deployment variability or contention from concurrent tasks. Modern edge environments further involve multiple cooperating devices, increasing potential computational capacity but also introducing challenges in managing distributed and dynamic resources.

Existing approaches attempt to address these challenges in several ways. Dynamic DNNs [2] enhance single-device adaptability by enabling runtime selection among pre-trained widths, providing fine-grained control over the accuracy–latency trade-off. Yet, their optimization is restricted to individual devices and cannot leverage the additional capacity available in multi-device systems. Distributed inference methods parallelize computation across devices, but layer-wise dependencies induce substantial intermediate communication, making performance sensitive to network fluctuations and requiring complex scheduling [3]–[5]. Ensemble-based collaborative inference [6], [7] avoids such dependencies by aggregating outputs from independently executed models, but existing ensembles remain static in their model configurations, which are fixed at training time, limiting their effectiveness under time-varying conditions.

To address these limitations, we introduce AdaEnsemble, Fig. 1, an adaptive and collaborative ensemble inference framework built on Dynamic DNNs, that enables runtime adjustment of both participating devices and their inference capacities. This flexibility improves the system's ability to adapt to resource variability and deadline constraints. We present a decision-making algorithm to jointly consider accuracy and latency to select optimal device set and model widths for each input.

The contributions are: (1) we extend ensemble inference beyond fixed model configurations by incorporating Dynamic DNNs to achieve flexible accuracy–performance trade-offs; (2) we propose a runtime mechanism that selects model widths and devices under practical edge constraints; (3) experiments on heterogeneous edge devices demonstrate that AdaEnsemble adapts effectively to different deadlines and consistently achieves higher accuracy than state-of-art baselines.

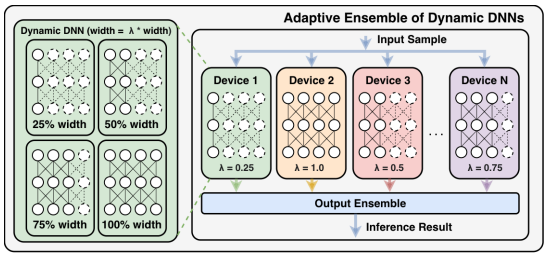

Fig. 1. Overview of adaptive ensemble of Dynamic DNNs.

II. Adaptive Ensemble of Dynamic DNNs

A. Training of Ensemble Dynamic DNNs

Our framework uses a set of Dynamic DNNs that support multiple width configurations through channel scaling. As shown in Fig. 1, a width hyperparameter adjusts the number of channels in each layer, enabling proportional control of computational cost at runtime.

To improve the accuracy of all widths, we adopt the training scheme of Slimmable Neural Network [2], which applies intra-model knowledge distillation by using the full-width model to supervise sub-models during training. We train multiple Dynamic DNNs with different random initializations under identical settings, producing models with similar per-width accuracy while maintaining the diversity required for effective ensemble inference. Unlike methods that require modified training dataset or workflow of ensemble [6], [7], our approach introduces no extra training or system complexity.

B. Deadline-aware Ensemble Inference

Fig. 2 illustrates the AdaEnsemble system architecture designed to support adaptive inference. A Master device coordinates multiple Worker devices, each capable of executing a Dynamic DNN at a selected width. The Master maintains a *Profiling Database* containing the latency of each device–width pair and the accuracy of all feasible ensemble configurations. These offline characterizations allow the system to make informed runtime decisions. When an input sample with deadline T_d arrives, the *Resource Allocator* predicts the end-to-end latency for each device, taking into account both computation and communication, and selects the subset of Workers and model widths that maximizes ensemble accuracy while meeting

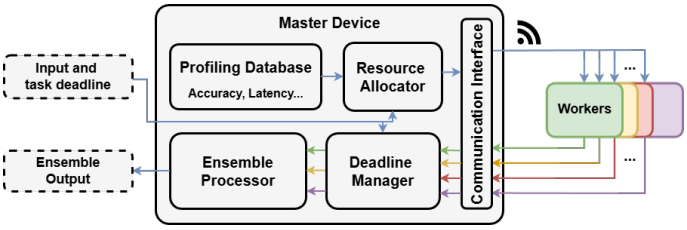

Fig. 2. System architecture of distributed ensemble inference

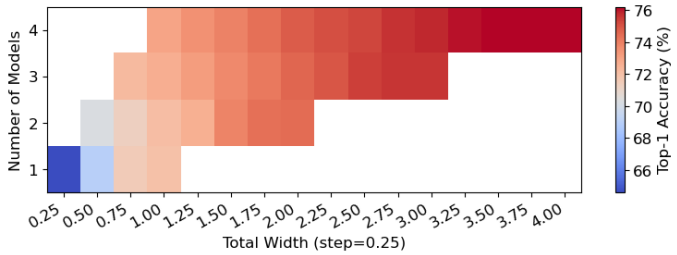

Fig. 3. Ensemble accuracy from different combinations of Dynamic DNNs.

the deadline. The Master distributes the input along with the selected widths to the corresponding Workers via the *Communication Interface*. During execution, all devices run inference in parallel. The *Deadline Manager* collects results and discards outputs that arrive after T_d, ensuring that the system never violates real-time constraints. Valid outputs are forwarded to the *Ensemble Processor*, which aggregates them by averaging output probability distributions and returns the final prediction.

III. Experimental Results and Analysis

We evaluate our approach in a heterogeneous edge environment consisting of multiple typical edge devices, as summarized in Table I. Communication between devices is performed using TCP/IP over wired gigabit Ethernet. Our experiments use PyTorch for DNN inference with GPU acceleration enabled. We adopt the widely used ResNet-18 [8] as the backbone network and extend it to a Dynamic ResNet-18 to support adaptive computation. The dynamic model supports four width configurations, 0.25, 0.50, 0.75, and 1.0, obtained using a training granularity of 0.25. For the model set, we train eight Dynamic ResNet-18 models with different random initializations and select the four models with the highest validation accuracy.

TABLE I
Cluster Configuration

Cluster	Device	Enabled Processing Units	Memory
Master	Jetson Nano	2 CPUs@1.47GHz	4 GB
Worker	Jetson Nano	2 CPUs@1.47GHz	4 GB
	Raspberry Pi5	4 CPUs@1.5GHz	8 GB
	Jetson Xavier NX	4 CPUs@1.49GHz	8 GB
	Jetson Nano	2 CPUs@1.47GHz, GPU@153MHz	4 GB

1) Ensemble Accuracy: We perform offline profiling to evaluate the accuracy of all feasible ensemble configurations (Fig. 3). When the total width is small (e.g. 0.75 or 1.0), multiple lightweight sub-models can match, or even exceed, the accuracy of a single larger model. Because sub-models run in parallel, the overall latency remains close to that of a single sub-model, showing that ensemble based cooperative inference can improve both accuracy and latency under limited capacity. As the total width increases, ensemble accuracy also improves, revealing a clear tradeoff between computational cost/latency

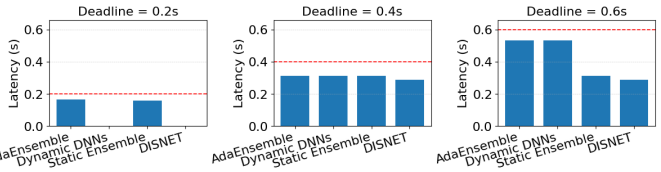

Fig. 4. Comparison of inference latency under different deadlines.

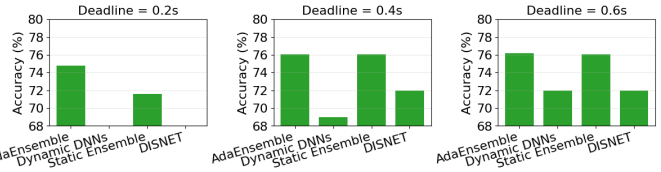

Fig. 5. Comparison of inference accuracy under different deadlines.

and accuracy. Each square in the figure represents the mean accuracy of all width combinations with the same total width and number of models; however the runtime still relies on the accuracy of individual combinations when making decisions.

2) System Performance: We evaluate AdaEnsemble under three deadline constraints (0.6s, 0.4s, and 0.2s) and compare it against representative approaches used in edge inference, including Dynamic DNNs, static ensemble inference, and distributed collaborative inference. As the single-device adaptive baseline, we adopt the Dynamic DNN mechanism from [2]. For the static ensemble baseline, we implement the model-assignment strategy in [6], which deploys one fixed-width model per device based on a predefined latency target. We additionally reproduce the distributed layer-block partitioning method proposed in DISNET [4].

Latency and accuracy results for all methods are presented in Figs. 4 and 5. With a 0.6s deadline, AdaEnsemble and Dynamic DNNs select larger widths and meet the latency requirement, but AdaEnsemble achieves higher accuracy by leveraging multiple devices. Static Ensemble and DISNET remain unchanged across deadlines and therefore provide fixed performance. With a 0.4s deadline, AdaEnsemble scales its widths to satisfy the tighter latency constraint and achieves accuracy similar to Static Ensemble; this is expected as the static models are trained for this latency target. Dynamic DNNs must reduce width aggressively and consequently suffer a large accuracy drop. Under the strict 0.2s deadline, only AdaEnsemble and Static Ensemble can return results before the deadline. Static Ensemble fails on most devices, leaving only the Raspberry Pi output available and reducing accuracy to that of its single model. AdaEnsemble adapts both device selection and model widths, enabling cooperative inference and achieving the highest accuracy among all methods.

IV. Conclusions

AdaEnsemble is a collaborative ensemble inference framework that coordinates Dynamic DNNs across heterogeneous edge devices. Through adaptive management of hardware and model configurations, it consistently returns timely results and achieves higher accuracy than state-of-the-art approaches.

V. Acknowledgments

Research data associated with this paper will be made available on acceptance.

REFERENCES

[1] L. Xun, M. Hu, H. Zhao, A. K. Singh, J. Hare, and G. V. Merrett, "Fluid dynamic dnns for reliable and adaptive distributed inference on edge devices," in *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2024, pp. 1–2.

[2] J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang, "Slimmable neural networks," in *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[3] J. Mao, X. Chen, K. W. Nixon, C. Krieger, and Y. Chen, "Modnn: Local distributed mobile computing system for deep neural network," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2017, pp. 1396–1401.

[4] E. Samikwa, A. Di Maio, and T. Braun, "Disnet: Distributed micro-split deep learning in heterogeneous dynamic iot," *IEEE internet of things journal*, vol. 11, no. 4, pp. 6199–6216, 2023.

[5] L. Zhou, M. H. Samavatian, A. Bacha, S. Majumdar, and R. Teodorescu, "Adaptive parallel execution of deep neural networks on heterogeneous edge devices," in *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. ACM, 2019, pp. 195–208.

[6] X. Feng, C. Luo, B. Wei, J. Zhang, J. Li, H. Wang, W. Xu, M. C. Chan, and V. C. M. Leung, "Time-constrained ensemble sensing with heterogeneous iot devices in intelligent transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 12 949–12 960, 2023.

[7] S. Kumazawa, J. Yu, K. Kawamura, T. V. Chu, and M. Motomura, "Toward improving ensemble-based collaborative inference at the edge," *IEEE Access*, vol. 12, pp. 6926–6940, 2024.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.