State of the Art on: Input-Aware Dynamic Quantization in Deep Neural Networks

Mehmet Emre Akbulut, mehmetemre.akbulut@mail.polimi.it

1. Introduction to the research topic

Tiny Machine Learning (TinyML) is a subset of Machine Learning that serves as a link between the ML domain and the embedded system ecosystem. It enables us to employ advanced machine learning capabilities on edge devices with limited memory, computation power and energy consumption while offering reduced latency and better privacy through on-device data processing.

Besides all the advantages mentioned above, there are many challenges when deploying machine learning models into resource-constrained devices. Most of the challenges originate from heterogeneity and resource constraints of the edge devices with lower memory capacity, compute power and energy consumption [4],[16]. Considering the intersection with machine learning, our research mainly belongs to the computer vision and pattern recognition field. In this direction, our research aims to propose solutions related to image classification tasks with deep neural networks. On the other hand, in the TinyML field, a variety of approaches and methods are used to overcome these issues. One of the main techniques used to compress and deploy these models on devices with limited resources is low-precision quantization. Quantization algorithms are mainly grounded on the idea of reducing the precision of weights, biases, and activations. In other words, a 32-bit full precision model is compressed to a low-bit representation by employing bit widths from 8-bit to 1-bit.

As we conduct research related to compressing machine learning models by maintaining their capability, the **conferences** related to our research mainly belong to computer vision and machine learning fields. In this context, the most prestigious **conferences** are IEEE Computer Vision and Pattern Recognition, European Conference on Computer Vision, and IEEE International Joint Conference on Neural Network. IEEE/CFV CVPR is an annual conference on computer vision and pattern recognition. It is regarded as one of the most important conferences in the field. According to Google Scholar Metrics (2024), it is the second highest impact computing venue after Nature [1]. ECCV is another important conference in which papers we are inspired by are published, which is co-organized by the Computer Vision Foundation. Additionally, the IEEE International Joint Conference is another conference through which we can aim to publish our research. Considering the interest and funding coming from the best companies and universities and the "h-5 index" metric, along with NeurIPS, ICML, and ICLR, these are the most prestigious **conferences**. Also, IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Neural Networks and Learning Systems, and Proceedings of the IEEE International Conference on Computer Vision are some of the most prestigious **journals** available, considering their impact scores [3], [2].

1.1. Preliminaries

Above all, the fundamental concept for the understanding of the research topic is **Quantization**. In Figure 1, a schematic overview of matrix-multiply operation in neural network accelerator hardware is shown [13]. A MAC operation starts by loading bias b_n. After, $W_{n,m}$ and x_m are loaded into the array in order to calculate $C_{n,m} = W_{n,m}x_m$ multiplication. Lastly, biases are added to the calculated value in accelerators.

$$A_n = b_n + \sum_m C_{n,m}$$

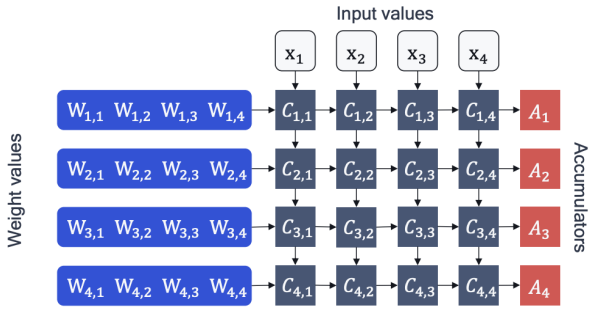


Figure 1: A schematic overview of matrix-multiply logic in neural network accelerator hardware [13].

Most neural network models are trained with 32-bit floating point so our values are stored and calculated with 32-bit representation with the requirement that the accelerator support FP logic. Consequently, we can significantly reduce the amount of used resources applying quantization techniques through using lower bit representation.

The main idea behind is the conversion of a value from floating point representation to an integer one. Briefly, a floating point vector $\hat{x}$ can be expressed by an integer vector x_{int} multiplied by a scale factor s_x:

$$\hat{x} = s_x \cdot x_{int} \approx x$$

where s_x is a floating-point scale factor and x_{int} is an integer vector, e.g., INT8.Together with them, $\hat{x}$ is the quantized vector.

This representation reduces the energy needed for data transfer, computation, and memory, which eventually helps to deploy machine learning models on edge devices. Apart from these, **Uniform/Non-Uniform Quantization** and **Symmetric/Asymmetric Quantization** are related concepts when dealing with quantization schemes. **Quantization-Aware Training (QAT)** is another concept that is developed to train neural networks with simulated quantization in order to make the network aware of quantization priorly.
In order to cope with challenges and apply known algorithms in the TinyML ecosystem, some frameworks such as TensorFlow Lite Micro are used [7]. Additionally, Pytorch supports various approaches for quantization [14].

1.2. Research topic

Besides all the advantages mentioned above, there are many challenges to solve in order to deploy machine learning models into resource-constrained devices. Most of the challenges originate from heterogeneity and resource constraints of the edge devices with lower memory capacity, compute power, energy consumption [4],[16]. In the field of TinyML, a variety of approaches and methods are used to overcome these issues. One of the main techniques used to compress and deploy these models on devices with limited resources is low-precision quantization. Quantization algorithms are mainly grounded on the idea of reducing the precision of weights, biases, and activations. In other words, a 32-bit full precision model is compressed to a low-bit representation by employing bit widths from 8-bit to 1-bit. Various quantization techniques are available in the literature so far. **The thesis aims to explore the solutions to reduce memory and computation overhead at run-time by employing a dynamic quantization approach while focusing on choosing efficient bit precisions for different layers considering the given input**. Eventually, an instance-aware dynamic quantization framework that reduces computation overhead while ensuring minimal accuracy loss. In other words, from a set of candidates varying from 1 to k bit-width $(b^1, b^2, ..., b^k)$, for each layer L_n, efficient bit precision is chosen depending on the given input to the model with n layers.

The proposed dynamic quantization framework aims to reduce computation overhead without changing the current architecture while enabling faster execution by enabling simpler operations instead of full precision floating point operations, adapting the model to the given input.

2. Main related works

2.1. Classification of the main related works

The research will explore the efficient quantization schemes in the context of TinyML. In detail, we will focus on mixed precision combined with dynamic quantization concerning a given input, specifically targeting image classification tasks. Here is the classification of the main related works:

1. Quantization techniques on deep neural networks to reduce memory consumption and computation overhead
2. Efficient mixed-precision quantization
3. Dynamic quantization and TinyML solutions in dynamic settings

2.2. Brief description of the main related works

2.2.1 Quantization techniques on deep neural networks

Neural Network Quantization has been one of the remarkable areas in the field of machine learning for recent years. Earlier works start with quantization network weights with certain bit-widths without focusing on activation precision [8], [10], [22]. Naturally, the main narrative in these papers was reducing model size and also storage efficiency. Later, activation bit-widths are taken into consideration by some fundamental papers in the field [15], [21], [6]. The DoReFa-Net [21] uses the "straight-through estimator" [20] to train convolutional neural networks (CNNs) using low bitwidth weights, activations, and gradients. On the other hand, PACT [6], PArametrized Clipping acTivations, is a quantization technique for activations, which enables minimum accuracy loss when working in low precisions. Note that, in these works, researchers generally propose methods based on a quantization with the same precision for all layers.

2.2.2 Efficient mixed-precision quantization

With the help of various hardware platforms, a mixed precision approach has been conducted in recent years. Because search space is very large $O(M^N)$ Neural Architecture Search methods are used to determine the optimal precision values for layers [18]. [18] and the following works contributed to the literature by explaining an SGD-based differentiable neural architecture search for mixed precision and DAG approach for candidate architectures and adding probabilistics to optimize. Also, it is shown that **mixed precision** training can substantially reduce the computational resources required, specifically in terms of memory bandwidth and processing time, making it feasible to train larger and more complex models within the same hardware constraints [17]. This paper proposes a regularization way for better training of mixed precision DNNs. Researchers who use gradient-based algorithms (SGD) to find bit-widths add quantization step size as a learning parameter to minimize a newly defined loss function. In [17], STE approach is used to show that the quantizer's parameters, including the bitwidth, can be learned with gradient methods if a good parametrization is chosen. They propose to learn the step size and dynamic range. The bitwidth can be inferred from them by showing that learning directly the bit-width is not optimal. In [18], a novel, effective, and efficient differentiable neural architecture search (DNAS) framework is proposed to solve the precision selection problem. They represent the architecture search space with a stochastic super net where nodes represent intermediate data tensors of the super net (e.g., feature maps of a ConvNet) and edges represent operators (e.g., convolution layers in a ConvNet). Any candidate architecture can be seen as a child network (sub-graph) of the supernet.

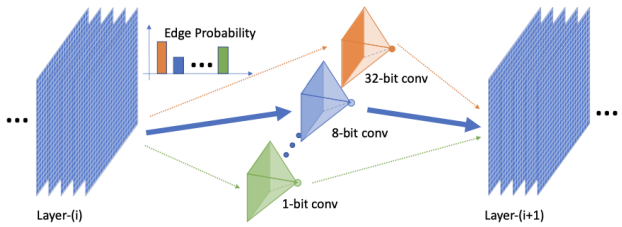


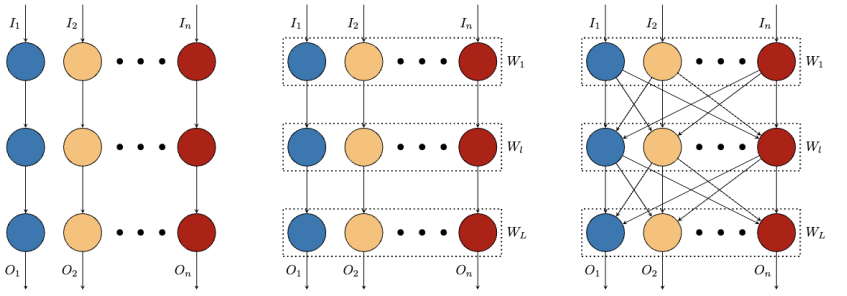
Figure 2: Mixed Precision Neural Architecture Search [18]

2.2.3 Dynamic quantization and TinyML solutions in dynamic settings

Even though mixed precision enables the development of different architectures through choosing different bit-widths for each layer, existing research proposes fixed, predefined bit-widths which are not changing without re-training. This situation limits the capability of the model. Consequently, some dynamic quantization solutions are available in the literature. In the dynamic network quantization literature, one of the very first works was published in 2018 [19]. In this paper, a Dynamic Network Quantization framework utilizes bit widths of layers via bidirectional LSTM which exploits the reinforcement learning. The framework was based on weight quantization and uses RL which leads to high computation overhead. Also, layer bit widths are fixed during inference.

Furthermore, AdaBits were proposed to allow dynamically adjust bit precision of the model during inference [9]. The main drawback of this paper is that all layers share the same bit precision so the framework cannot exploit the advantages of mixed precision.

In [5], a bit predictor, Bit-Mixer, which focuses on choosing bit precisions for each layer on inference time is proposed. Bit-Mixer optimizes the bit precisions considering the resource availability and performance instead of focusing on the input.



(a) **Independent:** Each bit-width requires training a new network with independent weights.

(b) **Adabits:** A single network can be quantized to any of n bit-widths at runtime. All layers inside the network share the *same* bit-width.

(c) **Proposed method (Bit-Mixer):** A single network whose individual layers can be quantized at runtime to any bit-width, without any re-training, resulting in an exponential number of mixed precision networks that one can choose from to fit the device characteristics and computational resources available on-the-fly.

Figure 3: Comparison between prior network quantization paradigms [5]

One of the earliest research focused on the input data is D^2NN [11] which executes a subset of the neurons depending on the given input.

In [12], authors present an Instance-Aware DQNet framework which consists of a predictor bit-controller

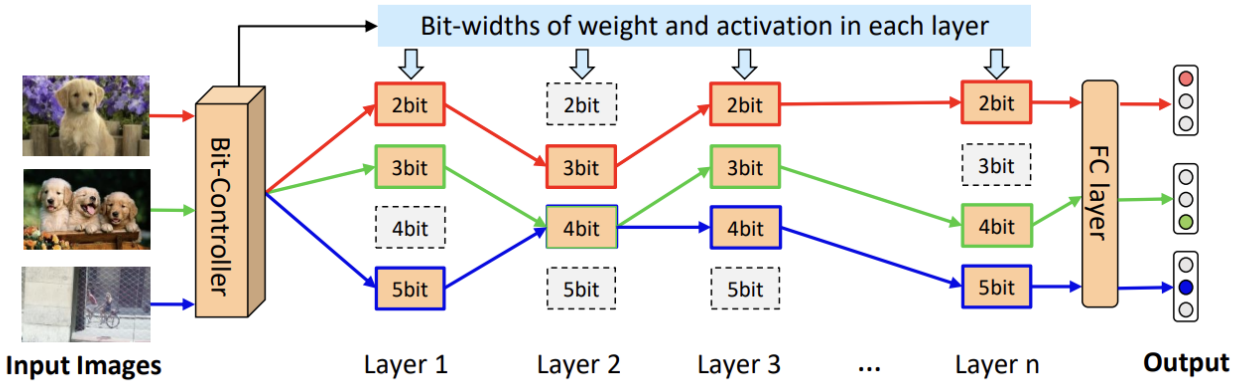


Figure 4: Dynamic network quantization scheme [12]

network trained together with the model itself. This framework is the most similar work to our thesis so we will focus on the possible directions considering the results of this paper.

2.3. Discussion

The existing solutions in dynamic quantization suffer from the lack of an instance-aware approach that considers the complexity of the given inputs to the model. Even though similar works exist, more robust and applicable framework methods are not available in the literature. Based on this, the main open issue is *"How can we design and implement an instance-aware dynamic quantization framework that adapts bit precision considering given input for devices with limited memory and computation power while maintaining model accuracy?"*. Additionally, testing and real-world deployment of these solutions are also another open problem in the current literature.

References

[1] scholar.google.com, 2024. https://scholar.google.com/citations?view_op=top_venues [Accessed: October 2024].

[2] Scimago journal rank artificial intelligence, 2024. https://www.scimagojr.com/journalrank.php?category=1702 [Accessed: October 2024].

[3] Scimago journal rank computer vision and pattern recognition, 2024. https://www.scimagojr.com/journalrank.php?category=1707 [Accessed: October 2024].

[4] Banbury, C. R., Reddi, V. J., Lam, M., Fu, W., Fazel, A., Holleman, J., Huang, X., Hurtado, R., Kanter, D., Lokhmotov, A., Patterson, D. A., Pau, D., sun Seo, J., Sieracki, J., Thakker, U., Verhelst, M., and Yadav, P. Benchmarking tinyml systems: Challenges and direction. *ArXiv abs/2003.04821* (2020).

[5] Bulat, A., and Tzimiropoulos, G. Bit-mixer: Mixed-precision networks with runtime bit-width selection. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), 5168–5177.

[6] Choi, J., Wang, Z., Venkataramani, S., Chuang, P. I.-J., Srinivasan, V., and Gopalakrishnan, K. Pact: Parameterized clipping activation for quantized neural networks. *ArXiv abs/1805.06085* (2018).

[7] David, R., Duke, J., Jain, A., Reddi, V. J., Jeffries, N., Li, J., Kreeger, N., Nappier, I., Natraj, M., Regev, S., Rhodes, R., Wang, T., and Warden, P. Tensorflow lite micro: Embedded machine learning on tinyml systems. *ArXiv abs/2010.08678* (2020).

[8] Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. *arXiv: Computer Vision and Pattern Recognition* (2015).

[9] Jin, Q., Yang, L., and Liao, Z. A. Adabits: Neural network quantization with adaptive bit-widths. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 2143–2153.

[10] Leng, C., Li, H., Zhu, S., and Jin, R. Extremely low bit neural network: Squeeze the last bit out with admm. In *AAAI Conference on Artificial Intelligence* (2017).

[11] Liu, L., and Deng, J. Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution. *ArXiv abs/1701.00299* (2017).

[12] Liu, Z., Wang, Y., Han, K., Ma, S., and Gao, W. Instance-aware dynamic neural network quantization. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 12424–12433.

[13] Nagel, M., Fournarakis, M., Amjad, R. A., Bondarenko, Y., van Baalen, M., and Blankevoort, T. A white paper on neural network quantization. *ArXiv abs/2106.08295* (2021).

[14] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. *ArXiv abs/1912.01703* (2019).

[15] Rastegari, M., Ordonez, V., Redmon, J., and Farhadi, A. Xnor-net: Imagenet classification using binary convolutional neural networks. *ArXiv abs/1603.05279* (2016).

[16] Sanchez-Iborra, R., and Skarmeta, A. F. Tinyml-enabled frugal smart objects: Challenges and opportunities. *IEEE Circuits and Systems Magazine 20* (2020), 4–18.

[17] Uhlich, S., Mauch, L., Cardinaux, F., Yoshiyama, K., García, J. A., Tiedemann, S., Kemp, T., and Nakamura, A. Mixed precision dnns: All you need is a good parametrization. In *International Conference on Learning Representations* (2019).

[18] Wu, B., Wang, Y., Zhang, P., Tian, Y., Vajda, P., and Keutzer, K. Mixed precision quantization of convnets via differentiable neural architecture search. *ArXiv abs/1812.00090* (2018).

[19] Xu, Y., Zhang, S., Qi, Y., Guo, J., Lin, W., and Xiong, H. Dnq: Dynamic network quantization. *2019 Data Compression Conference (DCC)* (2018), 610–610.

[20] Yin, P., Lyu, J., Zhang, S., Osher, S., Qi, Y., and Xin, J. Understanding straight-through estimator in training activation quantized neural nets. *ArXiv abs/1903.05662* (2019).

[21] Zhou, S., Ni, Z., Zhou, X., Wen, H., Wu, Y., and Zou, Y. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *ArXiv abs/1606.06160* (2016).

[22] Zhu, C., Han, S., Mao, H., and Dally, W. J. Trained ternary quantization. *ArXiv abs/1612.01064* (2016).