

# Efficient Down-sampling in Hybrid Neural Networks using Adversarial Autoencoders

Jonghyeon Nam<sup>†</sup>, JoonSeok Kim<sup>†</sup>, Eunji Kwon<sup>‡</sup>, and Seokhyeong Kang<sup>†</sup>  
<sup>†</sup>*Pohang University of Science and Technology, Pohang, Republic of Korea*  
<sup>‡</sup>*Kookmin University, Seoul, Republic of Korea*

*Abstract—*

Early convolutional layers in hybrid neural networks enable efficient down-sampling but pose a significant burden on inference latency and energy consumption. We propose a method to replace the conventional down-sampling block with lightweight autoencoders to enhance applicability in edge devices. We introduce an adversarial training strategy to align the autoencoder's latent features with the original stem, ensuring compatibility with succeeding layers. Applying our method to MobileViTV2-050 yields a 1.23x speedup and a 47% reduction in Energy-Delay Product (EDP) with only a 1.0% accuracy drop on ImageNet-1K.

## I. INTRODUCTION

Hybrid neural networks, which combine Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs) [1], have become a standard for mobile vision tasks [3, 4]. As illustrated in Fig. 1, these networks typically follow a hierarchical structure where a stack of convolutional layers, known as the stem, down-samples high-resolution inputs before passing the extracted features to subsequent attention blocks [5]. While effective for extracting local features, this stem is computationally expensive. For instance, in MobileViTV2-100 [3], the stem accounts for 32.8% of the total inference latency.

Motivated by the manifold hypothesis that high-dimensional data lies on a lower-dimensional manifold [6], we propose replacing the heavy convolutional stem depicted in Fig. 1 with a lightweight autoencoder (AE) [7]. Unlike standard AEs that focus solely on reconstruction, we employ adversarial training [8] to ensure the encoder generates features compatible with the pre-trained network. Our approach significantly reduces latency, and Energy-Delay Product (EDP) [9] on edge devices (mobile CPU and GPU) while maintaining competitive accuracy.

## II. PROPOSED METHODS

### A. Lightweight Down-sampling Using Autoencoders

Conventional down-sampling stems typically rely on a stack of MobileNetV2 (MV2) blocks, which can introduce significant computational overhead due to their depth. To address this, we replace this heavy architecture with a lightweight encoder composed of standard convolutional layers and efficient pooling operations. This structural substitution drastically shortens the data path, directly contributing to reduced inference latency. Furthermore, the encoder is specifically engineered to generate output feature maps with dimensions identical to those of the

This work was supported by Korea Institute for Advancement of Technology(KIAT) grant funded by the Korea Government(MOTIE) (RS-2024-00401466, HRD Program for Industrial Innovation) and by Institute of Information communications Technology Planning Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2023-00222085, Development of memory module and memory compiler for non-volatile PIM).

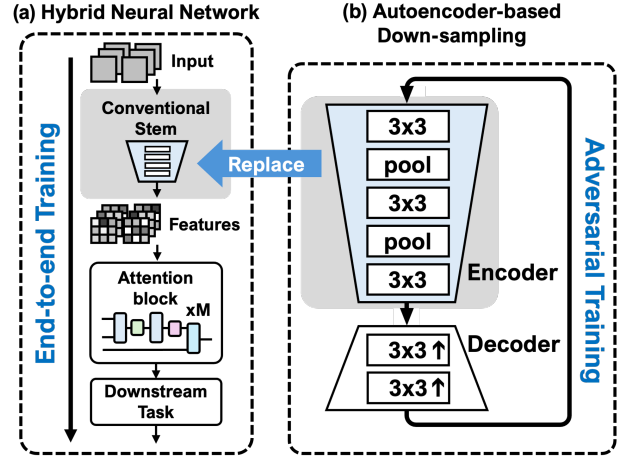


Fig. 1: Illustration of applying our proposed method on MobileViT [3]. (a) shows a hybrid neural network with a conventional stem, and (b) shows our proposed autoencoder-based down-sampling method.

original stem, thereby ensuring seamless and direct integration into the existing hybrid architecture.

### B. Adversarial Training for Feature Alignment

Training an autoencoder solely with reconstruction loss ( $L_{recon}$ ) often fails to capture the semantic information required by the subsequent transformer blocks. To address this, we introduce a discriminator ( $D$ ) to distinguish between the features generated by our encoder ( $z$ ) and the features from the original stem ( $m \sim p_{CD}(x)$ ), as illustrated in Fig. 2

The training proceeds in two phases:

1) *Autoencoder Update Phase*: The encoder is first trained to minimize the reconstruction error (MSE) to embed a high-quality manifold into the latent space. The reconstruction loss is defined as:

$$L_{recon} = \mathbb{E}_{z \sim q(z), z \sim q(z|x)} \|x - y\|^2 \quad (1)$$

Simultaneously, the encoder acts as a generator to fool the discriminator ( $L_{gen}$ ), guiding it to mimic the distribution of the original stem:

$$L_{gen} = -\mathbb{E}_{z \sim q(z|x)} \log D(z) \quad (2)$$

The total autoencoder loss combines these two objectives:

$$L_{AE} = \lambda_{recon} L_{recon} + \lambda_{gen} L_{gen} \quad (3)$$

2) *Discriminator Update Phase*: The discriminator is trained to maximize the probability of assigning the correct label to both real features from the original stem and fake features from the encoder. The loss functions are defined as:

$$L_{real} = -\mathbb{E}_{m \sim p_{CD}} [\log D(m)] \quad (4)$$

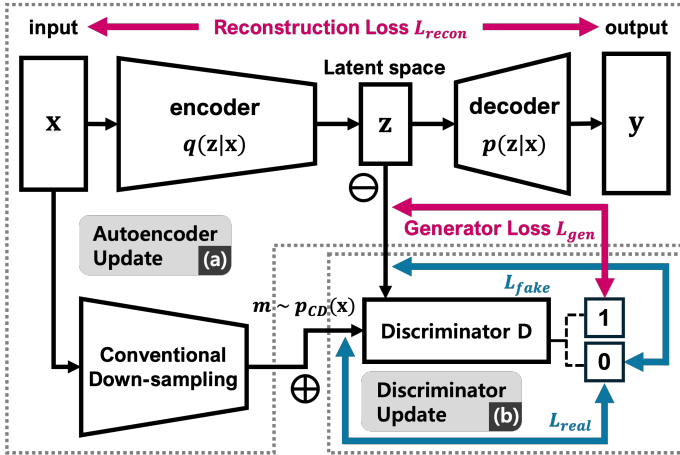


Fig. 2: The adversarial training scheme of the autoencoder consists of two phases: (a) the autoencoder update phase and (b) the discriminator update phase. The loss for the autoencoder update phase is colored in red, while the loss for the discriminator update is shown in blue.

$$L_{fake} = -\mathbb{E}_{z \sim q(z)} [\log(1 - D(z))] \quad (5)$$

The total discriminator loss is calculated as the average of these two terms ( $L_{dis} = 0.5(L_{real} + L_{fake})$ ).

After this pre-training, the discriminator and decoder are discarded. The trained encoder replaces the original stem, and the entire hybrid network undergoes end-to-end fine-tuning to recover any accuracy loss.

### III. EXPERIMENTS

#### A. Setup and Metrics

We evaluated our method on ImageNet-1K using MobileViT [3] and MobileViTV2 [4] variants implemented via the `timm` library [10]. Performance was measured on Raspberry Pi 4 (mobile CPU) and NVIDIA Jetson Nano (mobile GPU). We report Top-1 accuracy, Latency, and EDP [9].

#### B. Efficiency Improvements

Fig. 3 highlights the performance improvements achieved by our method across different hardware platforms. As shown in the latency charts, our autoencoder-based approach consistently reduces inference time, achieving up to a 1.41 $\times$  speedup on mobile GPU (MobileViTV2-100). Furthermore, the bottom

TABLE I: Comparison of Top-1 Accuracy, and Accuracy Decrease for down-sampling networks trained via knowledge distillation (KD) versus our proposed adversarial training method (Ours). Accuracy Decrease indicates the performance drop in top-1 accuracy of KD and Ours relative to the Original. The variants of each method are evaluated at the standard resolution ( $224 \times 224$ ).

Model	Top-1 Acc.			Acc. Decrease (%)	
	Original	KD	Ours	KD	Ours
MobileViT-XXS	68.9	63.2	67.3	5.7	<b>1.6</b>
MobileViT-XS	74.6	69.0	72.3	5.6	<b>2.3</b>
MobileViT-S	78.3	72.9	76.3	5.4	<b>2.0</b>
MobileViTV2-050	70.2	66.1	69.2	4.1	<b>1.0</b>
MobileViTV2-100	77.9	72.8	75.6	5.1	<b>2.3</b>

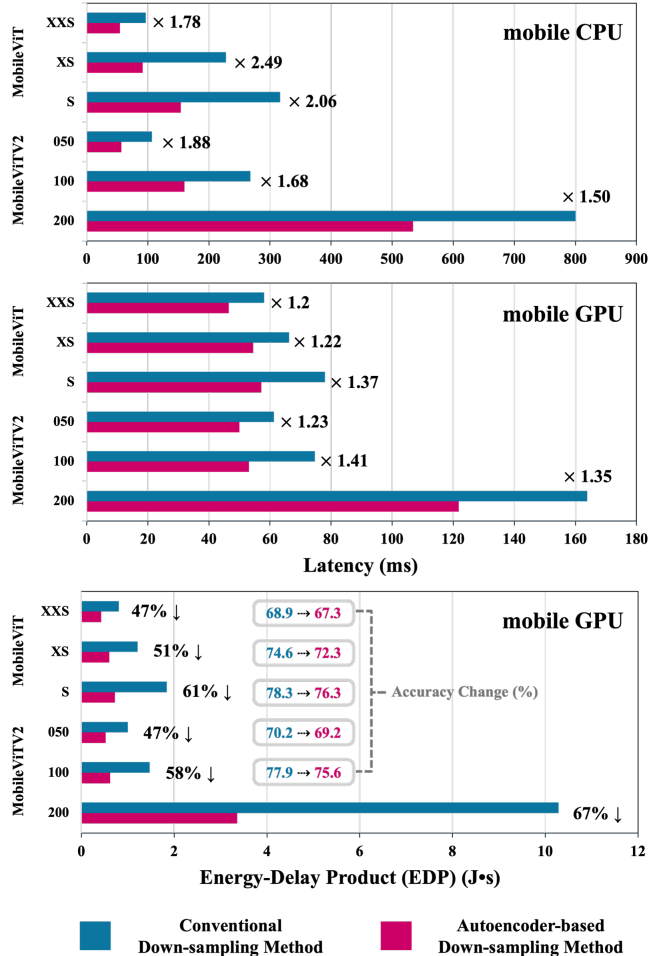


Fig. 3: Comparison of mobile CPU/GPU latency, EDP [9] on mobile GPU, and top-1 accuracy between conventional down-sampling and autoencoder-based down-sampling methods on MobileViT Variants, all evaluated at a fixed input resolution of  $224 \times 224$ . The top-1 accuracy for MobileViTV2-200 is excluded.

chart demonstrates a significant enhancement in energy efficiency, with EDP reductions ranging from 47% to 67%, while maintaining competitive top-1 accuracy with only marginal decreases.

#### C. Adversarial Training vs. Knowledge Distillation (KD)

To validate the effectiveness of our method, we compared it with KD [11] in an ablation study. As shown in Table I, simple KD resulted in significant accuracy degradation. In contrast, our method achieved a nearly threefold reduction in accuracy loss, demonstrating that feature alignment via a discriminator is critical for maintaining representational quality.

### IV. CONCLUSION

We presented a novel down-sampling method for hybrid neural networks using adversarial autoencoders. By replacing heavy convolutional stems with lightweight encoders trained to mimic the original feature manifold, we achieved significant gains in inference speed and energy efficiency on edge hardware. Our results demonstrate that this approach offers a practical solution for deploying complex vision models in resource-constrained mobile environments.

## REFERENCES

- [1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", *International Conference on Learning Representations (ICLR)*, 2021.
- [2] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", *arXiv preprint arXiv:1704.04861*, 2017.
- [3] Sachin Mehta, and Mohammad Rastegari, "MobileViT: Lightweight, General-purpose, and Mobile-friendly Vision Transformer", *Proceedings of the 10th International Conference on Learning Representations (ICLR)*, 2022.
- [4] Sachin Mehta, and Mohammad Rastegari, "Separable Self-attention for Mobile Vision Transformers", *arXiv preprint arXiv:2206.02680*, 2022.
- [5] Weiyang Luo, Yujia Li, Raquel Urtasun, and Richard Zemel, "Understanding the Effective Receptive Field in Deep Convolutional Neural Networks", *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.
- [6] Yoshua Bengio, Aaron Courville, and Pascal Vincent, "Representation Learning: A Review and New Perspectives", *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35(8):1798-1828, 2013.
- [7] Geoffrey E. Hinton, and Richard S. Zemel, "The "Wake-Sleep" Algorithm for Unsupervised Neural Networks", *Advances in Neural Information Processing Systems (NeurIPS)*, 1995.
- [8] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey, "Adversarial Autoencoders", *arXiv preprint arXiv:1511.05644*, 2015.
- [9] Mark Horowitz, Thomas Indermaur, and Ricardo Gonzalez "Low-Power Digital Design", *Proceedings of 1994 IEEE Symposium on Low Power Electronics*, pp. 8-11, 1994.
- [10] Ross Wightman, "pytorch-image-models", *Timm*, <https://github.com/rwightman/pytorch-image-models>.
- [11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the Knowledge in a Neural Network", *arXiv preprint arXiv:1503.02531*, 2015.