Enhancing RT-DETR Efficiency with Mixture of Experts Approach and Matrix Decomposition

Thanh Thien Nguyen, Quoc Cuong Nguyen, and Duc Lung Vu

***Abstract*—In real-time object detection, convolutional neural networks (CNNs) have traditionally dominated the field. Recently, however, RT-DETR - a transformer-based object detection model - has emerged as a competitor to the CNN-based YOLO series by tackling limitations introduced by non-maximum suppression (NMS) in YOLO. Despite its strong potential, RT-DETR requires extensive runtime optimizations, such as conversion to a TensorRT environment, to achieve competitive processing speeds. Additionally, RT-DETR's scaling approach focuses solely on adjustments within the decoder stage. In this paper, we propose a novel enhancement by integrating Mixture of Experts (MoE) and matrix decomposition techniques into RT-DETR's encoder stage. This enhanced encoder significantly reduces computational complexity while preserving accuracy. Our model achieves a 50% reduction in FLOPs of encoder for a 640x640 input size, with only a minimal 0.4% drop in average precision (AP) on the COCO dataset, compared to the original RT-DETR. The official implementation code of our method is available at https://github.com/quoccuonglqd/RT-DETR**

Link to graphical and video abstracts, and to code: https://latamt.ieeer9.org/index.php/transactions/article/view/9524

***Index Terms*—object detection, model compression, transformer, mixture of experts, matrix decomposition, deep learning**

I. Introduction

MODERN real-time object detection aims to identify and localize objects in images swiftly, enabling applications in fields such as video surveillance, autonomous driving, and video object tracking. Traditionally, this domain has been dominated by CNN-based architectures like the YOLO series [1]. Despite the transformative impact of Transformer architectures in other areas, CNN-based approaches have continued to outperform Transformer-based models for real-time object detection due to the latter's computational demands. Recently, Real-Time Detection Transformer (RT-DETR) [2] has emerged as a promising Transformer-based alternative to YOLO, addressing the latency caused by non-maximum suppression (NMS) in post-processing, which hinders end-to-end detection in YOLO models. Unlike YOLO, DETR models eliminate the need for this manual component; however, DETR's high computational cost limits its suitability for real-time tasks. RT-DETR enhances DETR's structure by redesigning multi-scale feature processing in the encoder and introducing an uncertainty-minimal query selection scheme to improve object query optimization.

Nevertheless, RT-DETR still requires post-processing optimizations—such as conversion to half precision and TensorRT—to match YOLO's real-time speed. This highlights the need for architectural improvements to reduce computational complexity without relying on external optimizations, especially those studying adjustments inside the model's architecture.

Some works has tried to mitigate the heavy computational burden. A few of them try the approach of improving the model architecture. For example, Hou et al [3] proposed hierarchical salience filtering refinement, in which the transformer encoding are only conducted on filtered discriminative queries. Zhu et al [4] combined the sparse spatial sampling of deformable convolution, and the relation modelling capability of Transformers, to high complexity issues of DETR. Li et al [5] developed a lite version of DETR by introducing new encoder block to update high-level features and low-level features in an interleaved way.

Despite the progress, few work pays attention to the use of MoEfication method, which employs the Mixture of Expert [6] module and have shown promising scalability in Transformer-based models in other domains, such as large language models. RT-DETR's current scaling mechanism relies on adjusting only the decoder layers, presenting an opportunity to explore Mixture of Experts (MoE) in the encoder to improve scalability. In our work, we incorporate multiple expert networks in the encoder and investigate expert redundancy to identify essential parameters. Specifically, we propose decomposing the weight matrix in the encoder's fully connected layers into the product of two low-rank matrices, sharing the first matrix across all experts as a global feature representation. This decomposition substantially reduces the parameters in the MoE architecture.

Our main contribution can be summarized as following:

- We introduce a novel MD-MoE (Matrix Decomposition-Mixture of Experts) module to enhance the transformer layer in RT-DETR's hybrid encoding, significantly improving scalability and reducing complexity.
- Compared to the original RT-DETR, our method (with four experts) achieves approximately 50% reduction in GFLOPs for 640x640 input images, with only a 0.4% drop in average precision (AP) on the COCO dataset [7].
- To the best of our knowledge, this is the first exploration of Mixture of Experts and low-rank decomposition within transformer-based object detection models.

The associate editor coordinating the review of this manuscript and approving it for publication was Ruth Aguilar (*Corresponding author: Quoc Cuong Nguyen*).

T. T. Nguyen, Quoc Cuong Nguyen, and D. L. Vu are with University of Information Technology, Vietnam National University, Ho Chi Minh City, Vietnam (e-mails: thiennt@uit.edu.vn, cuongnq.17@grad.uit.edu.vn, and lungvd@uit.edu.vn).

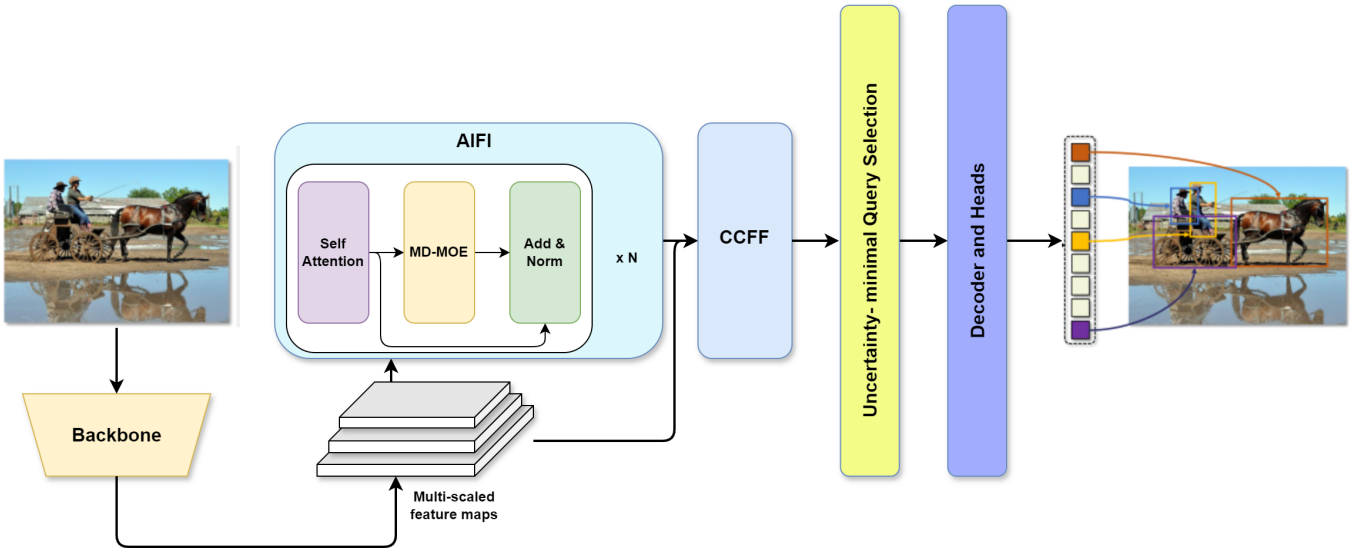

Fig. 1. Overview of the updated RT-DETR. The model begins by passing the input through a backbone network. We use the last 3 feature maps to capture multi-scale information. The encoder part consists of AIFI and CCFF transforms multi-scale features into a sequence of image features. Next, an uncertainty-minimal query selection strategy chooses a fixed subset of these features to initialize object queries. The decoder finally utilizes these object queries to predict the location and class labels of detected objects. **Our modification**: We use a new module name MD-MOE instead of traditional feed-forward network in the AIFI part.

II. Related Work

A. Real-time End-to-End Detector

Traditional object detectors rely on anchor points to predict bounding boxes, with anchor-based methods typically employing CNN backbones. Anchor points are positioned at the center of each sliding window, providing candidate positions for object localization. These detectors are classified into two main types: one-stage and two-stage models. In two-stage detectors, such as Faster R-CNN [8] and Mask R-CNN [9], dense class-agnostic predictions are initially generated to propose candidate foreground regions, a process typically handled by a Region Proposal Network (RPN) [8]. Features from these proposals are extracted from backbone network outputs and fed into separate heads to predict object scores, bounding box coordinates, and classification labels. Redundant detections are removed through Non-Maximum Suppression (NMS), enabling high-performance results. However, since the two stages are separated and non-end-to-end, these detectors often fail to meet real-time requirements. In contrast, one-stage detectors predict classes and anchor offsets across the entire feature map without region proposal steps, offering a direct, streamlined approach. Prominent models in this category include YOLO [1], SSD [10], and RetinaNet [11]. YOLO, with its extensive development, has become synonymous with real-time object detection.

The end-to-end object detection landscape shifted significantly with Carion et al.'s [12] introduction of DETR (DEtection TRansformer), which eliminated the need for manual post-processing components such as NMS. DETR uses a transformer-based encoder-decoder architecture (built on the Vision Transformer (ViT) [13]), integrating global context through the encoder and applying cross-attention in the decoder to make direct final predictions based on object queries. This architecture removed the need for hand-designed anchor generation, attracting substantial attention and inspiring numerous variants that address DETR's remaining challenges. For instance, Deformable DETR [14], DN-DETR [15], and Group DETR [16] improve small-object detection performance, matching the two-stage detector results while accelerating convergence.Conditional DETR [17], Anchor DETR [18], and DINO [19] reduce optimization difficulties. To address computational costs, Efficient DETR [20] reduces encoder and decoder layers, and Lite DETR [5] lowers complexity by decreasing the frequency of low-level feature updates in the encoder.

Recently, RT-DETR [2] further optimized DETR's computational cost by redesigning the encoder. With external optimizations such as half-precision and TensorRT conversion, RT-DETR achieves performance and accuracy comparable to state-of-the-art YOLO models. However, to make Transformer-based models feasible for real-world deployment, further improvements are necessary. In our research, we focus on enhancing the model architecture of RT-DETR itself, rather than relying on external optimizations like pruning, quantization, or runtime conversions.

B. Object Detection Model Compression

The field of model compression for object detection has seen significant development, with various approaches devised to optimize model speed and reduce computational costs. Major strategies include pruning, quantization, knowledge distillation, and tensor decomposition, each with distinct mechanisms and advantages. (1) Pruning involves reducing network size by eliminating unnecessary parameters, classified broadly into two categories: magnitude pruning and data-driven pruning. Magnitude pruning [21], [22] is a straightforward technique

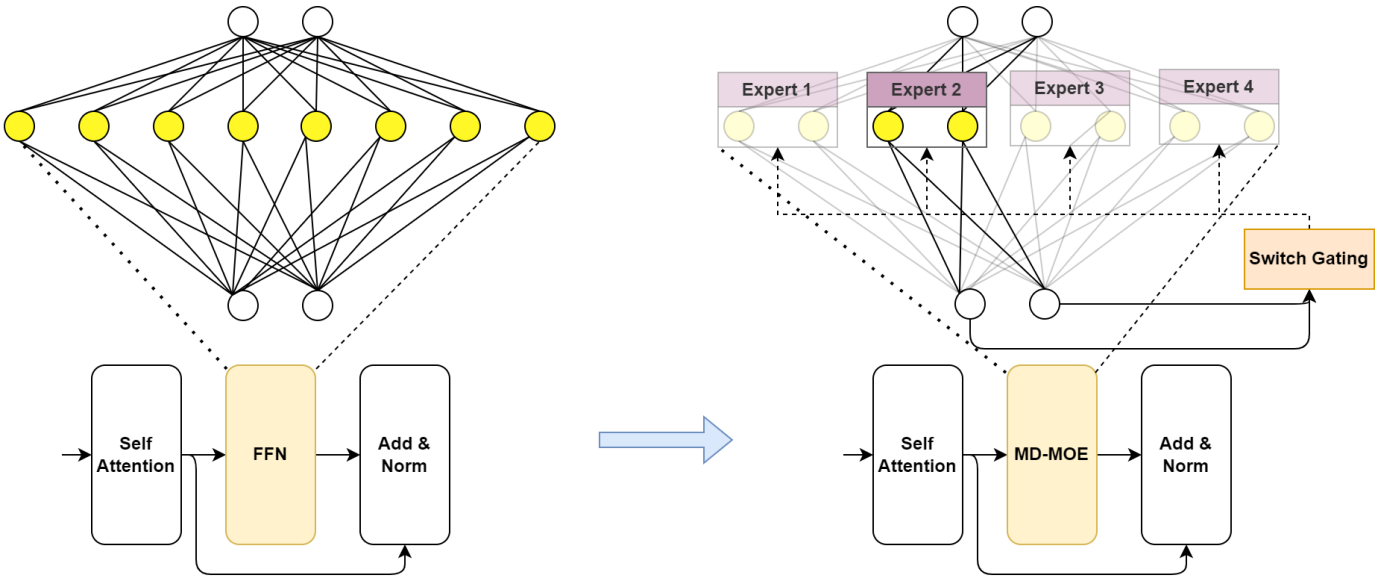

Fig. 2. The structure of the MoE layer used in the model. The layer FFN in the original self-attention block is splitted into multiple smaller FFN (called expert). Each expert's hidden layer size is set to be the original one divided by the number of experts. A routing mechanism (top 1 in this case) associates each token to one of the experts

that removes parameters based on their absolute values. In contrast, data-driven pruning [23], [24] analyzes network performance on a given dataset to identify less significant parameters, allowing selective pruning either during or after training. (2) Quantization compresses the model by representing parameters with fewer bits, classified into two types: quantization-aware training and post-training quantization. The distinction lies in whether quantization progress is considered during or after the training process. Post-training quantization methods [25] aim to reduce precision after the model is trained. Whereas, quantization-aware training [26] allows quantizing the model and fine-tuning it simultaneously. This kind of quantization can mitigate performance degradation caused by quantization. (3) Knowledge Distillation transfers learned knowledge from a complex model (teacher model) to a simplified model (student model) through three main techniques. Firstly, in feature-based knowledge distillation [27]–[30], the student model is trained to mimic the intermediate feature representations of the teacher model. Secondly, soft label-based knowledge distillation [31], [32] guides the student model using the softened output probability of the teacher model. Thirdly, we have the knowledge distillation based on some additional information guidance [33]–[36]. The student model captures the relational knowledge between different components of the input as represented by the teacher model. These components may comprise probabilistic distribution, graph, feature embedding, ... (4) Tensor decomposition is also a compression technique applied in a few papers [37]–[40]. It decomposes tensors into lower-rank vectors or matrices, enhancing network efficiency by reducing computational complexity. In our research, tensor decomposition is utilized to break down weight matrices in the MoE model, allowing the sharing of low-rank matrices among experts, thereby reducing model redundancy.

Beside external optimizations, efficient model architectures are designed using compact, specialized components. These include efficient convolution operators [41]–[43], lightweight blocks/modules [44]–[46], and streamlined detection heads [47]–[49]. Our research introduces a novel approach, MoEfication, which redesigns transformer-based architectures for object detection using MoE, an approach that is relatively unexplored in the context of model compression for object detection.

C. Mixture of Experts

Mixture of Experts (MoE) is an ensemble learning technique that partitions a predictive modeling problem into subtasks, each handled by specialized expert models, with a gating network selecting the appropriate expert for each input. This approach enhances model capacity, particularly in large-scale applications like deep learning and large language models (LLMs). The concept of MoE was introduced in [6]. This work proposed a supervised learning procedure for systems with multiple networks, each learning a subset of training cases. The gating network dynamically selects experts, demonstrating improved training efficiency, reaching target accuracy in half the epochs compared to conventional models.

Over the next two decades, MoE saw significant development. Yuksel et al [50] made a survey to discuss fundamental models for regression and classification, training with the expectation-maximization algorithm, and improvements like mixtures of Gaussian process experts. It covered alternative training methods, such as variational learning and localized ME training, and addressed model selection, including optimal expert numbers and tree depth. With the rise of deep learning, understanding MoE's mechanisms became crucial. Chen et al [51] studied how MoE layers improve neural network performance. It explored why MoE does not collapse into a single model, emphasizing the cluster structure of underlying

problems and the non-linearity of experts. Empirical results showed MoE's success in classification tasks with intrinsic cluster structures, hard for single experts like two-layer CNNs, but solvable with MoE layers.

MoE has become pivotal in scaling LLMs. Fedus et al [52] introduced a sparsely-activated expert Transformer model. It simplified MoE routing, enabling models with up to 1.6 trillion parameters, using selective precision training with bfloat16 and increased regularization for stability. This work achieved 7x pre-training speed increases compared to dense models like T5-Base. Mistral AI released Mixtral-8x7B [53]. This Sparse Mixture of Experts (SMoE) model, with 47B total parameters but only 13B active, outperformed Llama 2 70B [54] on most benchmarks, offering 6x faster inference. DeepSeek AI's 2024 contribution [55], introduced DeepSeekMoE-16B. With 16.4B parameters, it employed fine-grained expert segmentation (activating 6 out of 64 routed experts plus 2 shared) and shared expert isolation, trained on 2T tokens. It achieved comparable performance to LLaMA2 7B [54] with 40% computations, deployable on a single 40GB GPU without quantization, showcasing economical training and efficient inference. However, despite its success, there have been relatively few attempts to adapt MoE to object detection.

III. Method

In this section, we first review the overall architecture inherited from RT-DETR, followed by an introduction to our MoEfication strategy applied within the hybrid encoder. The final part describes how this approach leverages matrix decomposition to share parameters among experts, significantly reducing computational costs.

A. Overall Architecture

We show the overall architecture in Fig. 1. Our model architecture is based on the RT-DETR design, comprising a CNN backbone, a hybrid encoder, and a transformer decoder. Multi-scale features are generated by extracting the final three feature maps from the backbone. Following RT-DETR, only the final feature map is processed through the AIFI (Attention-based Intra-scale Feature Interaction) module, which contains transformer encoder layers to integrate global contextual information. This design choice leverages the implicit contextual information captured from earlier layers within the final feature map, avoiding redundancy and potential confusion that might arise from performing feature interaction on shallower layers.

Subsequently, cross-scale interaction is executed in the CCFF (CNN-based Cross-scale Feature Fusion) module, where the output of AIFI interacts with the remaining two feature maps. The CCFF module employs several RepConv [49] units to effectively fuse multi-scale features into a unified representation. Our modification focus on the AIFI part, replacing the each fully connected layer with a variant of mixture-of-expert layer called MD-MOE. We explain this part in further detail in the next section.

The output of the hybrid encoder, enriched with both intra-scale and cross-scale information, is then refined through a query selection process. Here, a fixed number K of encoder features are selected to initialize the object queries for the decoder, streamlining the optimization process. Finally, a transformer decoder, along with multiple auxiliary prediction heads, decodes these object queries to detect objects within the image.

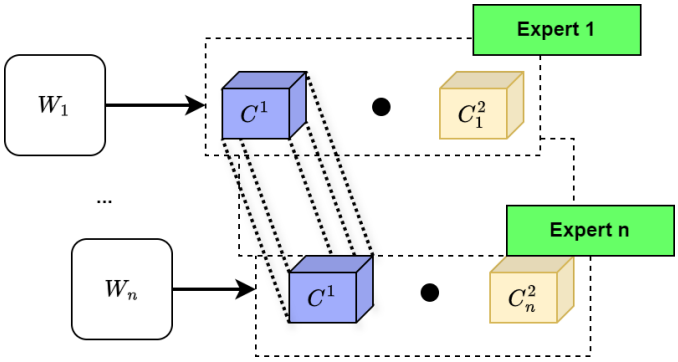

Fig. 3. Illustration of the proposed parameter-sharing scheme. We decompose each expert's weight matrix into a sequential product of two low-rank matrices. In our method, the first matrix is shared across all experts, allowing for parameter efficiency and consistency in feature representation. During optimization, backpropagation updates the second matrix individually for each expert, while the shared first matrix is consistently updated across all experts.

B. MoEficated Attention Block

To optimize the feed-forward layers within the transformer encoder, we propose a MoEfication approach, where each feed-forward network (FFN) layer is replaced by a Mixture-of-Experts (MoE) module. This module is illustrated in the Fig. 2. In the original RT-DETR design, the entire model activates for each input, leading to a quadratic increase in inference cost due to the self-attention module's quadratic complexity with respect to input length [56]. Retaining the standard FFN layer may also pose challenges for efficient exploration of network weight redundancy. In our approach, each FFN layer F is substituted with n expert networks $E_1, E_2, ..., E_n$, where each expert is selectively activated for specific inputs. A "gating network" G, implemented as a linear layer, directs the input to the appropriate expert network based on the input characteristics.

The output y of this MoE layer is computed by combining the outputs of the experts, weighted by the gating network's output:

$$y = \sum_{i=1}^{n} E_i(x) G(x)_i \tag{1}$$

In this formulation, each expert's output is denoted as $E_i(x)$, while $G(x)_i$ represents the routing weight assigned to expert i by the gating network G. To optimize computational efficiency, we leverage the sparsity in G's output. Specifically, when $G(x)_i = 0$, we can bypass the computation of $E_i(x)$, focusing only on computing outputs for the relevant experts. The routing mechanism is designed to produce a one-hot vector, activating only a single expert at each inference step. This is achieved by applying the Softmax function to the gating

network's output $G(x)$, then setting the entry with the highest probability to 1 and all others to 0. This approach ensures that only one expert is activated based on the routing decision.

$$G(x) = Softmax(x \cdot W_g) \tag{2}$$

This gating module can be trained simultaneously with the remaining of the model. For every input, there will always a chosen expert. The gate value of that expert has the nonzero gradient value with respect to the weights of the gating network.

C. MD-MOE: Parameter-Efficient MoE Architecture

The Mixture-of-Experts (MoE) module offers a scalable approach for simplifying our model's structure and reducing its computational load. By activating only a single expert for each input, based on the router's decision, we significantly lower the processing requirements. Despite this efficiency, the total number of parameters across all experts is equal to that of a single, original feed-forward layer, and including the parameters from the routing module leads to an increase in model size. While computational efficiency is improved, the expanded model size presents challenges for deployment.

To tackle this, we explore the question: "Are all parameters within the experts necessary?" In the field of language modeling, research has shown that redundancy exists across experts. Studies such as [52] and [57] have identified overlapping information among experts, suggesting that not every parameter is essential. Treating each expert as fully independent necessitates an extensive set of parameters, but we propose optimizing this by identifying shared parameters among closely related networks.

Our hypothesis is that the weight matrix for each expert in the MoE layer can be represented in a low-dimensional, shareable format. We validate this hypothesis by proposing a layer named MD-MOE, which is an updated version of normal MOE layer. Let W_i represent the weight matrix for expert i. Using Singular Value Decomposition (SVD), each weight matrix W_i is decomposed into a product of two smaller matrices, denoted as C_i^1 and C_i^2. We then modify the training process to share the first component, C^1, across all experts, as shown in Fig. 3. In each gradient descent step, the shared C^1 is updated, regardless of which expert is active, effectively encouraging it to capture global information common to all experts.

D. Training Pipeline

Our training pipeline consists of the following steps:

- Pretrain a model, such as RT-DETR, where standard feed-forward layers in the hybrid encoder are replaced with MoE modules.
- Apply SVD to decompose each expert's weight matrix and establish shared parameters by setting the first matrix to be common across experts.
- Fine-tune the model, allowing the shared matrices to update for all inputs, thereby solidifying the role of C^1 as a repository of shared knowledge among the experts.

TABLE I
COMPARISON OF THE PROPOSED METHODS WITH THE ORIGINAL RT-DETR AND THE BASELINE PRUNING APPROACH. 'MD' DENOTES THE MODEL CONFIGURATION WHERE PARAMETERS ARE SHARED AMONG EXPERTS AND SUBSEQUENTLY FINE-TUNED

Method	AP^{val}	GFLOPS	#Params (M)
YOLOv7-X [58]	52.9	189	45
YOLOv8-X [2]	53.9	257	68
DETR-DC5 [12]	43.3	187	41
Deformable DETR [14]	46.8	177	40
Lite-DETR [5]	46.7	123	41
Efficient-DETR-R50 [20]	45.1	210	35
Efficient-DETR-R101 [20]	45.7	289	54
RT-DETR-L [2]	54.3	189	76
RT-DETR-S [2]	48.1	397.5	21.95
RT-DETR-S + Pruning FFN (25%)	47.8	345.0	21.95
RT-DETR-S + Pruning FFN (50%)	47.7	292.6	21.95
RT-DETR-S + Pruning FFN (75%)	47.4	240.2	21.95
RT-DETR-S + Pruning FFN (85%)	47.0	219.2	21.95
Ours (4 experts)	47.7	240.6	21.95
Ours (8 experts)	47.6	214.8	21.95
Ours (16 experts)	43.2	209.5	21.95
Ours (4 experts + MD)	47.5	227.5	21.77
Ours (8 experts + MD)	47.4	211.5	21.77
Ours (16 experts + MD)	41.6	208.3	21.77

IV. EXPERIMENTS

In this section, we conduct an extensive series of experiments to assess the proposed method's performance and efficiency. We start by detailing the selected models, datasets, baseline methods, and experimental parameter configurations. Following this, we evaluate the accuracy and computational performance of the MoEfication model relative to baseline methods. Additionally, we examine the impact of our parameter-sharing approach using matrix decomposition.

A. Experimental Setup

To validate our proposed approach on the object detection benchmark, we adopt the RT-DETR setup and evaluate on the COCO [7] val2017 dataset. All training occurs exclusively on train2017 images, without incorporating any extra data sources. Detection accuracy is assessed using Average Precision (AP) across various IOU thresholds, while model performance is measured in terms of GFLOPS of encoder and parameter count. We opt for GFLOPS over the commonly used inference speed metric, as the latter is influenced by multiple hardware-dependent factors, such as precision level, processor type, cache size, and parallelism capabilities, ... Since our primary focus is on optimizing model architecture, GFLOPS provides a more stable theoretical performance metric. For consistency, GFLOPS values are reported for an input size of 640x640.

We use standard ImageNet [59] pre-trained ResNet-50 [60] as backbones to extract feature maps. We share with other detectors a common input size of 640x640. For the MoE layer configuration, we experiment with expert counts of 4 and 8, resulting in hidden sizes of 256 and 128 per expert, respectively. For the baseline methods, we reimplement one given limited follow-up research on RT-DETR regarding performance optimization. We employ a widely used pruning

framework [56] to design a baseline pruning approach, which targets only the feed-forward network components within the hybrid encoder. We choose the pruning ratio of 25%, 50%, 75%, 85%, with 75% and 85% yielding GFLOPS values comparable to our 4-expert and 8-expert MoE configurations. Our method is implemented using Pytorch library.

In our experimental setup, we follow the common setup as in the RTDETR paper to use GFLOPs is used as a crucial metric. GFLOPs serve as an effective metric for measuring efficiency because they directly quantify the computational cost of a model, independent of hardware-specific factors. Unlike model size, which only reflects the number of parameters and does not account for actual runtime complexity, GFLOPs provide a more practical measure of how much computation is required for inference. Compared to FPS (frames per second), which is influenced by hardware, software optimizations, and parallelization, GFLOPs offer a hardware-agnostic metric that allows for fairer comparisons.

Our experiments are conducted on a system with dual NVIDIA RTX 3090 GPUs. The MoEfication models undergo pretraining for 120 epochs, after which matrix decomposition is applied for parameter sharing, followed by fine-tuning for an additional 50 epochs. The batch size is set to 16 across all training and fine-tuning stages.

B. Results

Table I presents a detailed comparison of our method against existing notable object detection models. It also shows the result for the baseline method of pruning and the original RT-DETR, both utilizing a ResNet-50 backbone. The first section of the table reports results from various methods, with performance metrics extracted from their respective papers. The results for YOLOv8-X are sourced from the RT-DETR paper. The second section of the table showcases our experimental results, where the first four rows serve as a baseline comparison using different levels of pruning on the FFN of RT-DETR-S. The following six rows present our proposed approach, evaluating different configurations based on the number of experts and the inclusion of a Matrix Decomposition (MD) technique.

The incorporation of the Mixture of Experts (MoE) layer demonstrates superior performance compared to magnitude pruning. When using 4 and 8 experts, the GFLOPS of our method aligns with pruning rates of 75% and 85%, respectively, while achieving higher AP scores in both cases. Notably, when compared to pruning 50%, the model with 4 experts achieves the same AP score but requires 50 fewer GFLOPS. Furthermore, *adopting a parameter sharing strategy mitigates the increase in model parameters typically associated with MoE layers*. This scheme achieves a minimal 0.2% drop in performance while significantly reducing the parameter count, still outperforming the corresponding pruning results.

We observe that using 4 or 8 experts provides a reasonable balance between accuracy and computational efficiency. However, increasing the number of experts beyond this point leads to a noticeable drop in performance. Since we maintain a fixed total number of parameters, adding more experts requires reducing the size of each individual expert. This trade-off improves FLOPS efficiency but may restrict the representational capacity of each expert, limiting the overall effectiveness of the model. The decline in AP when using 16 experts suggests that at a certain point, the reduced capacity of individual experts outweighs the benefits of expert specialization.

TABLE II
SPEED COMPARISON WITH THE ORIGINAL RT-DETR ON NVIDIA RTX 3090 GPUS

Method	FPS
RT-DETR-S [2]	1.74
Ours (4 experts)	1.74
Ours (8 experts)	1.74
Ours (16 experts)	1.76
Ours (4 experts + MD)	1.52
Ours (8 experts + MD)	1.32
Ours (16 experts + MD)	1.36

The number of parameters remains largely consistent across experiments because the modifications—such as moefication, matrix decomposition, and pruning—target only the encoder's fully connected layers, which have a relatively small parameter footprint. However, moefication slightly increases the parameter count, countering our goal of optimization. To address this, the parameter-sharing scheme proves effective, reducing the parameter count further with only a 0.2% accuracy drop. Notably, even with this slight decrease in accuracy, the performance still surpasses the baseline pruning results.

In Table II, we report the actual inference time measured on NVIDIA RTX 3090 GPUs with the batch size of 1. Although our methods achieve better FLOPs, the actual inference time remains nearly unchanged. In fact, when using MD, there is a slight increase in inference time. However, this behavior is expected, as FLOPs alone do not fully determine the speed of model execution. Inference time can be influenced by several additional factors, including memory bandwidth, parallelism, and hardware utilization. For instance, even if a model has lower FLOPs, inefficient memory access patterns or frequent data transfers between different memory levels (such as GPU memory and caches) can introduce latency. Similarly, operations that cannot be efficiently parallelized—such as sequential dependencies in the model—can limit the benefits of reduced FLOPs.

V. CONCLUSION

In this paper, we present a new method for optimizing the computational cost of the transformer-based detector RT-DETR. Our approach consists of 2 key components: the use of MoE layer to alter the original FFN layer in original self-attention module, and a parameter-sharing scheme to share the global parameters among experts. Extensive experiments have shown that our approach outperforms the baseline methods of magnitude pruning.

ACKNOWLEDGMENTS

This research is funded by Vietnam National University Ho Chi Minh City (VNU-HCM) under grant number DS2024-

26-03 and supported by the Vingroup Innovation Foundation (VINIF).

REFERENCES

[1] J. Redmon, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, doi: https://doi.org/10.1109/cvpr.2016.91.

[2] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, and J. Chen, "Detrs beat yolos on real-time object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 965–16 974, doi: https://doi.org/10.1109/cvpr52733.2024.01605.

[3] X. Hou, M. Liu, S. Zhang, P. Wei, and B. Chen, "Salience detr: Enhancing detection transformer with hierarchical salience filtering refinement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 574–17 583, doi: https://doi.org/10.1109/cvpr52733.2024.01664.

[4] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," in *International Conference on Learning Representations*, doi: https://doi.org/10.48550/arXiv.2010.04159.

[5] F. Li, A. Zeng, S. Liu, H. Zhang, H. Li, L. Zhang, and L. M. Ni, "Lite detr: An interleaved multi-scale encoder for efficient detr," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 18 558–18 567, doi: https://doi.org/10.1109/cvpr52729.2023.01780.

[6] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural computation*, vol. 3, no. 1, pp. 79–87, 1991, doi: https://doi.org/10.1162/neco.1991.3.1.79.

[7] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755, doi: https://doi.org/10.1007/978-3-319-10602-1_48.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016, doi: https://doi.org/10.1109/tpami.2016.2577031.

[9] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969, doi: https://doi.org/10.1109/iccv.2017.322.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 21–37, doi: https://doi.org/10.1007/978-3-319-46448-0_2.

[11] T.-Y. Ross and G. Dollár, "Focal loss for dense object detection," in *proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2980–2988, doi: https://doi.org/10.1109/iccv.2017.324.

[12] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-toend object detection with transformers. in eccv," *Springer*, vol. 1, no. 2, p. 4, 2020, doi: https://doi.org/10.1007/978-3-030-58452-8_13.

[13] D. Alexey, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv: 2010.11929*, 2020, doi: https://doi.org/10.48550/arXiv.2010.11929.

[14] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020, doi: https://doi.org/10.48550/arXiv.2010.04159.

[15] F. Li, H. Zhang, S. Liu, J. Guo, L. M. Ni, and L. Zhang, "Dn-detr: Accelerate detr training by introducing query denoising," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 13 619–13 627, doi: https://doi.org/10.1109/cvpr52688.2022.01325.

[16] Q. Chen, X. Chen, J. Wang, S. Zhang, K. Yao, H. Feng, J. Han, E. Ding, G. Zeng, and J. Wang, "Group detr: Fast detr training with group-wise one-to-many assignment," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 6633–6642, doi: https://doi.org/10.1109/iccv51070.2023.00610.

[17] D. Meng, X. Chen, Z. Fan, G. Zeng, H. Li, Y. Yuan, L. Sun, and J. Wang, "Conditional detr for fast training convergence," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 3651–3660, doi: https://doi.org/10.1109/iccv48922.2021.00363.

[18] Y. Wang, X. Zhang, T. Yang, and J. Sun, "Anchor detr: Query design for transformer-based detector," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, no. 3, 2022, pp. 2567–2575, doi: https://doi.org/10.1609/aaai.v36i3.20158.

[19] H. Zhang, F. Li, S. Liu, L. Zhang, H. Su, J. Zhu, L. M. Ni, and H.-Y. Shum, "Dino: Detr with improved denoising anchor boxes for end-to-end object detection," *arXiv preprint arXiv:2203.03605*, 2022, doi: https://doi.org/10.48550/arXiv.2203.03605.

[20] Z. Yao, J. Ai, B. Li, and C. Zhang, "Efficient detr: improving end-to-end object detector with dense prior," *arXiv preprint arXiv:2104.01318*, 2021, doi: https://doi.org/10.48550/arXiv.2104.01318.

[21] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," *arXiv preprint arXiv:1608.08710*, 2016, doi: https://doi.org/10.48550/arXiv.1608.08710.

[22] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," *Advances in neural information processing systems*, vol. 28, 2015, doi: https://doi.org/10.48550/arXiv.1506.02626.

[23] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," *arXiv preprint arXiv:1611.06440*, 2016, doi: https://doi.org/10.48550/arXiv.1611.06440.

[24] 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 *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 784–800, doi: https://doi.org/10.1007/978-3-030-01234-2_48.

[25] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, "Pruning and quantization for deep neural network acceleration: A survey," *Neurocomputing*, vol. 461, pp. 370–403, 2021, doi: https://doi.org/10.1016/j.neucom.2021.07.045.

[26] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh *et al.*, "Mixed precision training," *arXiv preprint arXiv:1710.03740*, 2017, doi: https://doi.org/10.48550/arXiv.1710.03740.

[27] Q. Li, S. Jin, and J. Yan, "Mimicking very efficient network for object detection," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2017, pp. 6356–6364, doi: https://doi.org/10.1109/cvpr.2017.776.

[28] L. Qi, J. Kuen, J. Gu, Z. Lin, Y. Wang, Y. Chen, Y. Li, and J. Jia, "Multi-scale aligned distillation for low-resolution detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 443–14 453, doi: https://doi.org/10.1109/cvpr46437.2021.01421.

[29] R. Sun, F. Tang, X. Zhang, H. Xiong, and Q. Tian, "Distilling object detectors with task adaptive regularization," *arXiv preprint arXiv:2006.13108*, 2020, doi: https://doi.org/10.48550/arXiv.2006.13108.

[30] Z. Yang, Z. Li, X. Jiang, Y. Gong, Z. Yuan, D. Zhao, and C. Yuan, "Focal and global knowledge distillation for detectors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4643–4652, doi: https://doi.org/10.1109/cvpr52688.2022.00460.

[31] R. Mehta and C. Ozturk, "Object detection at 200 frames per second," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0, doi: https://doi.org/10.1007/978-3-030-11021-5_41.

[32] Z. Zheng, R. Ye, P. Wang, D. Ren, W. Zuo, Q. Hou, and M.-M. Cheng, "Localization distillation for dense object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9407–9416, doi: https://doi.org/10.1109/cvpr52688.2022.00919.

[33] P. De Rijk, L. Schneider, M. Cordts, and D. Gavrila, "Structural knowledge distillation for object detection," *Advances in Neural Information Processing Systems*, vol. 35, pp. 3858–3870, 2022, doi: https://doi.org/10.48550/arXiv.2211.13133.

[34] J. Guo, K. Han, Y. Wang, H. Wu, X. Chen, C. Xu, and C. Xu, "Distilling object detectors via decoupled features," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 2154–2164, doi: https://doi.org/10.1109/cvpr46437.2021.00219.

[35] G. Li, X. Li, Y. Wang, S. Zhang, Y. Wu, and D. Liang, "Knowledge distillation for object detection via rank mimicking and prediction-guided feature imitation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, no. 2, 2022, pp. 1306–1313, doi: https://doi.org/10.1609/aaai.v36i2.20018.

[36] X. Dai, Z. Jiang, Z. Wu, Y. Bao, Z. Wang, S. Liu, and E. Zhou, "General instance distillation for object detection," in *Proceedings of the*

IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 7842–7851, doi: https://doi.org/10.1109/cvpr46437.2021.00775.

[37] H. Kuang, L. Chen, L. L. H. Chan, R. C. Cheung, and H. Yan, "Feature selection based on tensor decomposition and object proposal for night-time multiclass vehicle detection," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 71–80, 2018, doi: https://doi.org/10.1109/tsmc.2018.2872891.

[38] X. Zhang, Y. Gong, C. Qiao, and W. Jing, "Multiview deep learning based on tensor decomposition and its application in fault detection of overhead contact systems," *The visual computer*, vol. 38, no. 4, pp. 1457–1467, 2022, doi: https://doi.org/10.1007/s00371-021-02080-y.

[39] L. Meneghetti, N. Demo, and G. Rozza, "A proper orthogonal decomposition approach for parameters reduction of single shot detector networks," in *2022 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2022, pp. 2206–2210, doi: https://doi.org/10.1109/icip46576.2022.9897513.

[40] L. Huyan, Y. Li, D. Jiang, Y. Zhang, Q. Zhou, B. Li, J. Wei, J. Liu, Y. Zhang, P. Wang *et al.*, "Remote sensing imagery object detection model compression via tucker decomposition," *Mathematics*, vol. 11, no. 4, p. 856, 2023, doi: https://doi.org/10.3390/math11040856.

[41] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258, doi: https://doi.org/10.1109/cvpr.2017.195.

[42] T. Zhang, G.-J. Qi, B. Xiao, and J. Wang, "Interleaved group convolutions," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4373–4382, doi: https://doi.org/10.1109/iccv.2017.469.

[43] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500, doi: https://doi.org/10.1109/cvpr.2017.634.

[44] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: efficient convolutional neural networks for mobile vision applications (2017)," *arXiv preprint arXiv:1704.04861*, vol. 126, 2017, doi: https://doi.org/10.48550/arXiv.1704.04861.

[45] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114, doi: https://doi.org/10.48550/arXiv.1905.11946.

[46] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, "Ghostnet: More features from cheap operations," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 1580–1589, doi: https://doi.org/10.1109/cvpr42600.2020.00165.

[47] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 734–750, doi: https://doi.org/10.1007/s11263-019-01204-1.

[48] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6569–6578, doi: https://doi.org/10.1109/iccv.2019.00667.

[49] T. Wang, X. Zhu, J. Pang, and D. Lin, "Fcos3d: Fully convolutional one-stage monocular 3d object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 913–922, doi: https://doi.org/10.1109/iccvw54120.2021.00107.

[50] S. E. Yuksel, J. N. Wilson, and P. D. Gader, "Twenty years of mixture of experts," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 8, pp. 1177–1193, 2012, doi: https://doi.org/10.1109/TNNLS.2012.2200299.

[51] Z. Chen, Y. Deng, Y. Wu, Q. Gu, and Y. Li, "Towards understanding mixture of experts in deep learning," *arXiv preprint arXiv:2208.02813*, 2022, doi: https://doi.org/10.48550/arXiv.2208.02813.

[52] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," *Journal of Machine Learning Research*, vol. 23, no. 120, pp. 1–39, 2022, doi: https://doi.org/10.48550/arXiv.2101.03961.

[53] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. d. l. Casas, E. B. Hanna, F. Bressand *et al.*, "Mixtral of experts," *arXiv preprint arXiv:2401.04088*, 2024, doi: https://doi.org/10.48550/arXiv.2401.04088.

[54] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023, doi: https://doi.org/10.48550/arXiv.2307.09288.

[55] D. Dai, C. Deng, C. Zhao, R. Xu, H. Gao, D. Chen, J. Li, W. Zeng, X. Yu, Y. Wu *et al.*, "Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models," *arXiv preprint arXiv:2401.06066*, 2024, doi: https://doi.org/10.48550/arXiv.2401.06066.

[56] F. D. Keles, P. M. Wijewardena, and C. Hegde, "On the computational complexity of self-attention," in *International Conference on Algorithmic Learning Theory*. PMLR, 2023, pp. 597–619, doi: https://doi.org/10.48550/arXiv.2209.04881.

[57] Y. J. Kim, A. A. Awan, A. Muzio, A. F. C. Salinas, L. Lu, A. Hendy, S. Rajbhandari, Y. He, and H. H. Awadalla, "Scalable and efficient moe training for multitask multilingual models," *arXiv preprint arXiv:2109.10465*, 2021, doi: https://doi.org/10.48550/arXiv.2109.10465.

[58] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 7464–7475, doi: https://doi.org/10.1109/CVPR52729.2023.00721.

[59] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255, doi: https://doi.org/10.1109/cvprw.2009.5206848.

[60] 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, doi: https://doi.org/10.1109/cvpr.2016.90.

Thanh Thien Nguyen received B.S. degree in Information Technology in 2013 and M.Sc. degree in Computer Science in 2018 from the University of Science, Vietnam National University Ho Chi Minh City. He is currently pursuing a Ph.D. degree in Computer Science at University of Information Technology, Vietnam National University Ho Chi Minh City, with a focus on efficient deep learning. His research interests include machine learning, computer vision and their applications.

Quoc Cuong Nguyen received the B.S. degree in computer science from the Honors Program, University of Information Technology, Vietnam National University Ho Chi Minh City, where he is currently pursuing the M.S. degree. He was a Research Assistant with VinUni-Illinois Smart Health Center, VinUniversity, in 2022. His research interests include machine learning, deep learning, and computer vision.

Duc Lung Vu received B.S. and M.Sc. degrees in Computer Engineering from the Peter the Great St.Petersburg Polytechnic University in 1998 and 2000, respectively. He got his Ph.D. in Computer Science from Saint Petersburg Electrotechnical University in 2006. He has been working at the University of Information Technology, Vietnam National University Ho Chi Minh City, as an Associate Professor since 2015 and Chancellor of the school since 2020. His research interests include machine learning, human-computer interaction, embedded systems and digital system design on FPGA.