Token Compression Meets Compact Vision Transformers: A Survey and Comparative Evaluation for Edge AI

Phat Nguyen* and Ngai-Man Cheung*
*Singapore University of Technology and Design (SUTD)
E-mail: {tienphat_nguyen,ngaiman_cheung}@sutd.edu.sg

***Abstract*—Token compression techniques have recently emerged as powerful tools for accelerating Vision Transformer (ViT) inference in computer vision. Due to the quadratic computational complexity with respect to the token sequence length, these methods aim to remove less informative tokens before the attention layers to improve inference throughput. While numerous studies have explored various accuracy–efficiency trade-offs on large-scale ViTs, two critical gaps remain. First, there is a lack of unified survey that systematically categorizes and compares token compression approaches based on their core strategies (e.g., pruning, merging, or hybrid) and deployment settings (e.g., fine-tuning vs. plug-in). Second, most benchmarks are limited to standard ViT models (e.g., ViT-B, ViT-L), leaving open the question of whether such methods remain effective when applied to structurally compressed transformers, which are increasingly deployed on resource-constrained edge devices. To address these gaps, we present the first systematic taxonomy and comparative study of token compression methods, and we evaluate representative techniques on both standard and compact ViT architectures. Our experiments reveal that while token compression methods are effective for general-purpose ViTs, they often underperform when directly applied to compact designs. These findings not only provide practical insights but also pave the way for future research on adapting token optimization techniques to compact transformer-based networks for edge AI and AI agent applications.**

I. Introduction

Vision Transformers (ViTs)[16] have emerged as a powerful and general-purpose architecture for various visual understanding tasks, spanning image[17], video[18], and multimodal domains[19], due to their powerful representation learning and strong generalization. However, their quadratic scaling with token count and large model sizes pose significant challenges for deployment on resource-constrained devices. To alleviate this, two complementary families of optimization methods have emerged. Structural compression adapts classical pruning and neural architecture search (NAS) techniques to ViTs. For example, AutoFormer[20] and ElasticViT[21] employ a two-stage search to derive compact ViT variants. In parallel, token optimization leverages the variable-length capabilities of self-attention to dynamically drop or merge less informative tokens under a predefined keep ratio, reducing inference cost without modifying the core network.[6], [9], [14]

These two paradigms differ in perspective. Structural compression adopts a model-centric, data-agnostic view: once the architecture is chosen. Token optimization, in contrast, is data-centric: it adapts to the information content of each sample: some methods act as plug-in modules in pretrained ViTs, while others require partial or complete retraining to learn optimal reduction ratios.

TABLE I
Summary of Token Compression Methods. Each method is categorized by its compression approach (pruning, merging, or hybrid — a combination of both), its reduction type (static: fixed keep-rate pruning; dynamic: adaptive keep-rate pruning; hard: exclusive token merging; soft: weighted averaging of token embeddings), and whether training is required.

Method	Approach	Compression Type	Training Required
EViT[1]	Pruning	Static	✗
DynamicViT[2]	Pruning	Static	✓
Cropr[3]	Pruning	Static	✓
ATS[4]	Pruning	Dynamic	✗
SPViT[5]	Pruning	Dynamic	✓
ToMe[6]	Merging	Hard	✗
TokenPooling[7]	Merging	Hard	✗
TCFormer[8]	Merging	Hard	✗
PiToMe[9]	Merging	Hard	✗
SiT[10]	Merging	Soft	✓
Sinkhorn[11]	Merging	Soft	✓
PatchMerger[12]	Merging	Soft	✓
DTEM[13]	Merging	Soft	✓
ToFu[14]	Hybrid	-	✗
DiffRate[15]	Hybrid	-	✓

Since these approaches are orthogonal, a question arises: can we combine them to achieve models that are both structurally and data-aware? This topic has not been explored in prior work, yet it is highly important given the stringent computational constraints of many emerging edge AI and AI agent applications. A straightforward strategy is to first apply structural compression - obtaining a *compact network* - then apply token optimization on top. In this work, we investigate this pipeline through extensive experiments.

Our contributions are three-fold:

1) We present a comprehensive survey and taxonomy of token compression methods, categorizing them by their compression strategy and deployment requirements (as summarized in Table I).
2) We conduct the experiments of representative token compression techniques applied to structure-compressed

ViTs, assessing their suitability for edge-deployable models.

3) We demonstrate that, when directly applied to compact backbones, existing token compression methods underperform, revealing a critical gap: token reduction algorithms must be specifically adapted to the architectural and resource constraints of compact transformers.

II. Related Works

A. Structure Compression

To address the need of computation-intensive applications [22]–[24] and to enable efficient deployment of Vision Transformers on resource-constrained settings such as edge AI [25], [26] and AI agent [27], [28] applications, structural compression has become a key research direction. Its goal is to mitigate the over-parameterization of ViTs when applied to downstream tasks, by reducing redundant computation and model size. Structural compression strategies are primarily grouped in two categories: Sparsity-based pruning, which removes redundant weights or activations, and Neural Architecture Search (NAS), which automatically finds efficient ViT designs under task-specific constraints.

Model pruning introduces sparsity by eliminating unimportant weights or neurons, effectively reducing the runtime cost of matrix multiplications and lowering latency. Techniques like channel pruning (NViT[29]) or width and depth pruning (WDPruning[30]) have shown promising results on Vision Transformers. Knowledge distillation can also be applied to enhance the performance of structural compressed networks [31], [32].

Neural Architecture Search (NAS), by contrast, seeks to directly define compact ViT architectures optimized for both accuracy and efficiency. For instance, AutoFormer proposes a one-shot NAS framework that uses weight-sharing (entanglement) across transformer blocks to jointly train a large supernet containing thousands of subnetworks. Once trained, a lightweight evolutionary search is conducted to select the best-performing subnet. Follow-up works have improved this paradigm by expanding the search space[33] or refining supernet training (e.g., NasViT[34], ViTAS[35]). In our study, we adopt the subnets discovered by the original AutoFormer framework as compact ViT backbones, and analyze how well token compression techniques integrate with these structure-optimized models.

B. Token Compression

The transformer-based designs support processing with token sequences of variable length, yet not all tokens are important to represent the meaning of the input sequence (e.g. background regions of an image) [8]. The token reduction techniques aim to detect and drops less important tokens in some layers in the rest of the network inference. Technically, a lightweight scoring module can be inserted at selected transformer layers to rank each token's importance and then compress the least useful ones during inference ([1], [2], [5]).

While the recent work [36] presents a categorization of several token compression methods to support a controlled experimental analysis of reduction patterns on the vanilla Vision Transformer [16], the focus of this work is an empirical study. In contrast, our work provides an extensive and systematic survey of token compression techniques. First, we introduce a taxonomy with a new category, hybrid compression, which integrates both pruning and merging within a single framework. Second, we broaden significantly the scope to include a wider range of recent developments, including plug-in, learnable, and adaptive approaches applicable to various vision tasks beyond classification, such as detection and segmentation.

III. A taxonomy of Token Compression methods

Inspired by [36], we start with discussion of two token compression paradigms: pruning and merging. We improve upon [36] by providing a broader and up-to-date view, then incorporate recent methods into both groups and introduce a new category, *hybrid compression*, which integrates both strategies in a unified design.

A. Token Pruning

Token pruning methods can be categorized into two main types based on how they determine the number of tokens to retain.

1) Static Keep Rate Pruning: In the static keep-rate setting, a fixed number of tokens is preserved at each reduction stage. For instance, EViT[1] selects the top-K most important tokens based on their attention to the CLS token and aggregates the pruned tokens into a single fused token using a weighted average. DynamicViT[2] also operates under a static token budget, but introduces a differentiable scoring module that learns to predict per-token importance, enabling end-to-end trainable token selection. Token Cropr [3]. extends token pruning beyond classification by introducing lightweight, task-aware auxiliary modules (one appended to each transformer block). Each auxiliary "Cropr" head uses a cross-attention mechanism with trainable queries to compute token-level importance scores. These scores are then supervised directly by the task-specific loss (e.g., segmentation, detection, classification), which allows the network to dynamically prune tokens most relevant to the end objective.

2) Dynamic Keep Rate Pruning: Dynamic Pruning technique adjusts the token reduction ratios adaptively for each input sample. ATS [4] relies on a stochastic sampling strategy to preserve fewer tokens when attention patterns focus on particular areas. Such an adaptive approach allows the model to optimize computational allocation according to the complexity of each input sample. SPViT [37] addresses efficient inference in Vision Transformers by proposing a soft token pruning strategy that adapts per input and per layer. The method stems from the observation that different attention heads in a ViT capture diverse and complementary features, making it suboptimal to treat token importance uniformly across heads. To account for this, SPViT introduces a token selector module that computes head-wise token importance scores and then

aggregates them into a global score via a learnable weighted combination.

B. Token Merging

The objective of token merging method is to reduce information loss by avoiding token dropping, but aim to combine similar tokens into representative ones. And the merging mechanism is typically built to pairing or clustering similar tokens. Based on the merging mechanism, these methods can be categorized into hard merging and soft merging.

1) Hard-Merging: In hard merging setting, discrete clustering algorithms are employed to assign tokens to distinct groups, and the merged tokens are computed as average or weighted representations of each group (ToMe[6], TokenPooling[38], TCFormer[39]). PiToMe [9] aims address the limitation of ToMe's pairing strategy that tends to remove informative tokens in deeper layers. To overcome this, PiToMe introduces an efficient metric called the energy score, which quantifies the importance of each token based on its contribution to the overall feature spectrum. In this formulation, background tokens that dominate large, redundant regions deemed to have high energy and are prioritized for merging, while low-energy tokens often carrying fine-grained or informative details, are preserved.

2) Soft-Merging: Soft merging approaches operate by enabling tokens to participate in multiple merged outputs through convex combinations calculated using a learned assignment matrices (SiT[10]) or query-based assignment (Sinkhorn[11], PatchMerger[12]). DTEM [13] proposes a differentiable token merging technique that leverages a lightweight yet effective auxiliary embedding module, which is decoupled from the main transformer layers. This module is specifically designed to compute token similarity for merging, using dedicated token embeddings that are independent of the backbone representation. The merging module can be trained either end-to-end with the transformer or modularly, allowing flexibility in optimization and deployment.

C. Hybrid Compression

While token pruning and token merging have usually been treated as separate paradigms, several works demonstrate that integrating both techniques can produce more effective and adaptable token compression results. Intuitively, hybrid compression methods aim to leverage the strengths of two compression schemes: the strengths of token pruning in efficiently removing clearly redundant tokens, and the strengths of token merging to preserve semantic information by fusing similar tokens.

ToFu [14] proposes an adaptive plug-in token compression strategy that can be applied without further re-training. Specifically, ToFu examines the model's output behavior when token embeddings are interpolated: if the output is sensitive to the change, it indicates that the tokens carry distinct information and pruning is favored to remove redundant ones; if the output is smooth, suggesting redundancy, merging is applied instead. To further improve compression quality, ToFu introduces a norm-preserving interpolation function to maintain the magnitude of merged tokens and reduce the risk of distribution shifts.

DiffRate [15] proposes a hybrid token compression framework that incorporates both pruning and merging operations under layer-wise differentiable compression ratios. The method employs a softmax-based re-parameterization technique allowing the gradients backpropagation through the compression ratio, enabling it to be optimized end-to-end.

D. Categorization by deployment requirements

Token compression methods differ in how they are deployed. Some, such as PiToMe [9] and ToFu [14], can be used as plug-ins without additional training. Others, like DTEM [13] or DiffRate [15], require full or partial fine-tuning to learn scoring modules or adaptive compression rates. Table I summarizes the recent token compression techniques following the taxonomy introduced in this work, along with their deployment requirements.

IV. Empirical Evaluation on Compact ViTs

A. Experiment settings

Setup. We evaluate token compression methods on the standard image classification task using the ImageNet-1K dataset, which contains 1.28M training images and 50,000 validation images. As the backbone architecture, we adopt AutoFormer-S as our representative *compact transformer*, pretrained on ImageNet-1K via supervised learning. To assess the impact of token compression, we report Top-1 classification accuracy, GFLOPs, and inference throughput measured in images per second (img/s) to evaluate both predictive performance and computational efficiency. For throughput measurement, we run all methods on a single NVIDIA A6000 GPU with a batch size of 128.

Compression methods. To provide a comprehensive evaluation of token compression techniques applied to compact transformers, we select representative methods from each category in the proposed taxonomy. For *plug-in methods* that do not require retraining, we include ToMe [6], PiToMe [9], and ToFu [14]. For *trainable methods* that require retraining or fine-tuning, we evaluate Cropr [3] (pruning-based), DTEM [13] (merging-based), and DiffRate [15] (hybrid compression). For all retrainable methods, we follow each method's official implementation and training procedure as originally proposed for standard Vision Transformer backbones, without manually re-tuning them for AutoFormer. This allows us to assess how readily these methods generalize to compact architectures without additional adaptation.

B. Experiment results

Table II reports the image classification performance of the compact transformer AutoFormer-S [20] when various token compression methods are applied. The first row shows the performance of the original model without any compression. We evaluate two experimental settings:

TABLE II
TOP-1 ACCURACY COMPARISON OF TOKEN COMPRESSION METHODS ON AUTOFORMER-S IN OFF-THE-SHELF AND RETRAINED SETTINGS (ACC1(OTS) VS. ACC1(RE-TRAIN)), WITH INFERENCE EFFICIENCY MEASURED BY GFLOPS AND THROUGHPUT (IMG/S).

Method	Acc1(ots) ↑	Acc1(re-train) ↑	GFLOPs ↓	img/s ↓
AutoFormer-S	81.66	81.66	4.92	988
ToMe	29.45	79.22	3.27	1550
PiToMe	30.10	78.74	3.27	1435
ToFu	30.69	78.17	3.27	1507
Cropr	-	69.47	4.26	1131
DiffRate	-	77.47	3.27	1557
DTEM	-	42.63	3.27	1620

TABLE III
ABLATION EXPERIMENTS FOR OFF-THE-SHELF SETTING WITH DIFFERENT COMPRESSION RATIOS ON AUTOFORMER-S. TOP-1 ACCURACY IS USED AS EVALUATION METRIC.

#pruned tokens	ToMe	ToFu	PiToMe	GLOPS
0	81.66	81.66	81.66	4.92
3	30.60	30.72	29.88	4.36
6	29.50	30.03	29.78	3.81
9	29.45	30.69	30.10	3.27
12	28.80	30.77	29.71	2.75
15	27.42	31.54	29.68	2.24
18	22.03	31.99	29.48	1.87

1) Off-the-shelf Setting: In this setting, we apply three parameter-free token compression methods, ToMe [6], PiToMe [9], and ToFu [14], as plug-in modules without retraining. As shown in Table II, the Top-1 accuracy scores (*Acc1(ots)*) drop drastically for all three methods, with reductions of nearly 50% compared to the original model. To examine whether the performance degradation is caused by overly aggressive token reduction, we conduct an ablation study by varying the number of preserved tokens. As shown in Table III, even relatively mild reductions (e.g., with compression ratios of 3 or 6 tokens) result in a sharp decline in classification accuracy. This suggests that either critical task-related features are being discarded, or the compressed token representations are misaligned with the pretrained network parameters, making the model unable to extract them effectively.

These results indicate that compact transformers are highly sensitive to plug-in token compression techniques. When applied directly without carefull adaptation, such methods can severely impair the model's ability to preserve discriminative features for image classification. In parallel, we also observe a clear reduction in computational cost as the number of preserved tokens decreases. Specifically, GFLOPs drop progressively from 4.36 to 1.87 as the compression ratios increase from 3 to 18 tokens (Table III). This trend demonstrates that token compression offers a tangible benefit in reducing inference cost, even when applied to an already optimized compact model like AutoFormer.

2) Retraining Setting: In this setting, we treat token compression modules as additional components integrated into the model and perform full network retraining after applying compression. The resulting Top-1 accuracies are denoted as *Acc1(re-train)* in Table II.

Motivated by the observations in the off-the-shelf setting, we first examine whether retraining can recover the performance of the three plug-in compression methods (ToMe, PiToMe, and ToFu). After fine-tuning the entire network, all three methods show substantial improvements in accuracy: approximately +50% for ToMe, and +48% for both PiToMe and ToFu. These results suggest that a major cause of performance degradation in the off-the-shelf setting could be a mismatch between the compressed token embeddings and the pretrained model weights. Retraining effectively realigns the network to the modified token embedding set, allowing it to process the compressed inputs more effectively.

Next, we evaluate adaptive compression methods that are designed to be trained jointly with the network: DiffRate [15], Cropr [3], and DTEM [13]. Among them, DiffRate demonstrates the most favorable accuracy-efficiency trade-off, achieving a 1.5× throughput speedup while maintaining Top-1 accuracy above 77%. In contrast, Cropr obtains limited gains in throughput and significantly reduces accuracy to below 70% with a relative GLOPs improvement. Interestingly, DTEM fails to converge during training in our setting, despite achieving the highest inference throughput. This may be attributed to its merging mechanism, which constructs a decoupled embedding space for trainable compression, which works well on standard ViTs but may require more careful fine-tuning on compact models like AutoFormer. This result indicates that adaptive compression configurations tuned for standard Vision Transformers may not directly transfer to compact architectures such as AutoFormer. These findings highlight the need for a method-specific or architecture-aware adaptation approach when applying token compression to compact models.

V. CONCLUSIONS

In this work, we present a comprehensive survey of recent token optimization techniques for Vision Transformers, covering a wide range of compression approaches—including pruning, merging, and hybrid strategies. While many of these methods demonstrate strong accuracy-efficiency trade-offs on standard ViTs, and some can be applied as plug-in modules without retraining, we explore a more practical scenario: applying token compression on already compressed (compact) ViTs for aggressive deployment settings.

Importantly, our empirical results indicates that token compression is not a one-size-fits-all solution, particularly when applied to compact backbones without adaptation. However, performance can be significantly recovered through retraining, suggesting that alignment between token representations and network parameters is critical. In addition, token compression can further reduce inference cost on compact architectures, highlighting its complementary role to the overall optimization pipeline. In summary, our study motivates future work toward a unified framework that jointly considers both structure-aware and data-centric optimization strategies, offering efficient and adaptive transformer models tailored for resource-constrained deployment scenarios such as edge AI and AI agent applications.

REFERENCES

[1] Y. Liang, C. GE, Z. Tong, Y. Song, J. Wang, and P. Xie, "EVit: Expediting vision transformers via token reorganizations," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=BjyvwnXXVn_.

[2] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C.-J. Hsieh, "Dynamicvit: Efficient vision transformers with dynamic token sparsification," in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021. [Online]. Available: https://openreview.net/forum?id=jB0Nlbwlybm.

[3] B. Bergner, C. Lippert, and A. Mahendran, "Token cropr: Faster vits for quite a few tasks," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 9740–9750.

[4] M. Fayyaz, S. A. Koohpayegani, F. R. Jafari, *et al.*, "Adaptive token sampling for efficient vision transformers," in *European Conference on Computer Vision*, Springer, 2022, pp. 396–414.

[5] Z. Kong, P. Dong, X. Ma, *et al.*, "Spvit: Enabling faster vision transformers via latency-aware soft token pruning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[6] D. Bolya, C.-Y. Fu, X. Dai, P. Zhang, C. Feichtenhofer, and J. Hoffman, "Token merging: Your ViT but faster," in *International Conference on Learning Representations*, 2023.

[7] D. Marin, J.-H. R. Chang, A. Ranjan, A. Prabhu, M. Rastegari, and O. Tuzel, "Token pooling in vision transformers for image classification," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2023, pp. 12–21.

[8] W. Zeng, S. Jin, W. Liu, *et al.*, "Not all tokens are equal: Human-centric visual analysis via token clustering transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 101–11 111.

[9] C. Tran, D. MH Nguyen, M.-D. Nguyen, *et al.*, "Accelerating transformers with spectrum-preserving token merging," *Advances in Neural Information Processing Systems*, vol. 37, pp. 30 772–30 810, 2024.

[10] Z. Zong, K. Li, G. Song, *et al.*, "Self-slimmed vision transformer," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[11] J. B. Haurum, M. Madadi, S. Escalera, and T. B. Moeslund, "Multi-scale hybrid vision transformer and sinkhorn tokenizer for sewer defect classification," *Automation in Construction*, vol. 144, p. 104 614, 2022, ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2022.104614. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580522004848.

[12] C. Renggli, A. S. Pinto, N. Houlsby, B. Mustafa, J. Puigcerver, and C. Riquelme, "Learning to merge tokens in vision transformers," *arXiv preprint arXiv:2202.12015*, 2022.

[13] D. H. Lee and S. Hong, "Learning to merge tokens via decoupled embedding for efficient vision transformers," in *Conference on Neural Information Processing Systems*, 2024.

[14] M. Kim, S. Gao, Y.-C. Hsu, Y. Shen, and H. Jin, "Token fusion: Bridging the gap between token pruning and token merging," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 1383–1392.

[15] M. Chen, W. Shao, P. Xu, *et al.*, "Diffrate: Differentiable compression rate for efficient vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 17 164–17 174.

[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *ICLR*, 2021.

[17] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*, Springer, 2020, pp. 213–229.

[18] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: A video vision transformer," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 6836–6846.

[19] A. Radford, J. W. Kim, C. Hallacy, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, PmLR, 2021, pp. 8748–8763.

[20] M. Chen, H. Peng, J. Fu, and H. Ling, "Autoformer: Searching transformers for visual recognition," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 12 270–12 280.

[21] C. Tang, L. L. Zhang, H. Jiang, *et al.*, "Elasticvit: Conflict-aware supernet training for deploying fast vision transformer on diverse mobile devices," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 5829–5840.

[22] M. Abdollahzadeh, T. Malekzadeh, and N. M. Cheung, "Revisit multimodal meta-learning through the lens of multi-task learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021.

[23] F. Mentzer, G. Toderici, D. Minnen, *et al.*, "Vct: A video compression transformer," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[24] N. M. Cheung, O. C. Au, M. C. Kung, P. H. W. Wong, and C. H. Liu, "Highly parallel rate-distortion optimized intra-mode decision on multicore graphics processors," *IEEE Transactions on Circuits and Systems for Video Technology*, 2009.

[25] N.-T. Tran, D.-K. Le Tan, A.-D. Doan, *et al.*, "On-device scalable image-based localization via prioritized cascade

search and fast one-many ransac," *IEEE Transactions on Image Processing*, vol. 28, no. 4, pp. 1675–1690, 2018.

[26] J. Sander, A. Cohen, V. R. Dasari, B. Venable, and B. Jalaian, "On accelerating edge ai: Optimizing resource-constrained environments," *arXiv preprint arXiv:2501.15014*, 2025.

[27] Z. Zhou, X. Ning, K. Hong, *et al.*, "A survey on efficient inference for large language models," *arXiv preprint arXiv:2404.14294*, 2024.

[28] Y. Chen and X. Li, "Rlrc: Reinforcement learning-based recovery for compressed vision-language-action models," *arXiv preprint arXiv:2506.17639*, 2025.

[29] H. Yang, H. Yin, P. Molchanov, H. Li, and J. Kautz, "Nvit: Vision transformer compression and parameter redistribution," 2021.

[30] F. Yu, K. Huang, M. Wang, Y. Cheng, W. Chu, and L. Cui, "Width & depth pruning for vision transformers," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 3143–3151.

[31] K. Chandrasegaran, N. T. Tran, Y. Zhao, and N. M. Cheung, "Revisiting label smoothing and knowledge distillation compatibility: What was missing?" In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2022.

[32] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[33] M. Chen, K. Wu, B. Ni, *et al.*, "Searching the search space of vision transformer," *Advances in Neural Information Processing Systems*, vol. 34, pp. 8714–8726, 2021.

[34] C. Gong and D. Wang, "Nasvit: Neural architecture search for efficient vision transformers with gradient conflict-aware supernet training," *ICLR Proceedings 2022*, 2022.

[35] X. Su, S. You, J. Xie, *et al.*, "Vitas: Vision transformer architecture search," in *European Conference on Computer Vision*, Springer, 2022, pp. 139–157.

[36] J. B. Haurum, S. Escalera, G. W. Taylor, and T. B. Moeslund, "Which tokens to use? investigating token reduction in vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 773–783.

[37] Z. Kong, P. Dong, X. Ma, *et al.*, "Spvit: Enabling faster vision transformers via latency-aware soft token pruning," in *European conference on computer vision*, Springer, 2022, pp. 620–640.

[38] D. Marin, J.-H. R. Chang, A. Ranjan, A. Prabhu, M. Rastegari, and O. Tuzel, "Token pooling in vision transformers for image classification," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2023, pp. 12–21.

[39] W. Zeng, S. Jin, W. Liu, *et al.*, "Not all tokens are equal: Human-centric visual analysis via token clustering transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 101–11 111.