Towards Robust Edge Model Adaptation via Elastic Architecture Search

Xianhang Chu, Xu Yang*, Kun Wei, Xi Wang

School of Electronic Engineering, Xidian University, Xi'an, Shaanxi, China
xhchu@stu.xidian.edu.cn, {xuyang.xd,weikunsk,wangxi6317}@gmail.com

Abstract

Continual test-time adaptation (CTTA) enables online model adjustment under dynamic distribution shifts in real-world environments. However, most existing CTTA frameworks adopt fixed model architectures, lacking the structural flexibility required for deployment across heterogeneous edge devices with varying computational capacities. To address this, we propose an elastic framework for edge CTTA that performs resource-aware dynamic model search based on a pre-trained Supernet. This enables architectural flexibility by generating personalized models tailored to the resource constraints of different edge devices. Considering the evolving distribution of unlabeled data on edge devices during deployment, we introduce a pluggable, lightweight fine-tuning mechanism. By inserting low-rank adapters into the frozen backbone, the model enables continual self-supervised adaptation with minimal computational overhead. To improve the quality of future architecture search without retraining, we propose a structure-aware knowledge reflux mechanism that distills adaptation experience from fine-tuned edge models into structurally similar paths within the Supernet. Experiments on multiple benchmarks validate that our method achieves state-of-the-art performance while significantly reducing resource consumption, with re-searched models after knowledge reflux showing further improvements.

Introduction

Deep neural networks typically achieve impressive performance in a similar train-test distribution, however, their effectiveness often diminishes in real-world scenarios where the distributions fluctuate over time. Autonomous systems (Arnold et al. 2019) deployed on edge devices, for example, frequently encounter diverse conditions—such as varying illumination between day and night or weather changes (Sakaridis, Dai, and Van Gool 2021; Xu et al. 2024b), including fog, rain, and snow—that can compromise model robustness. To maintain stable performance under these dynamic conditions, continual adaptation is critical. Continual Test-Time Adaptation (Wang et al. 2022) (CTTA) has emerged as a promising solution, allowing models to progressively adjust during deployment by utilizing incoming test samples exclusively. This approach obviates the need for source data(Quiñonero-Candela et al. 2022) and enables real-time adaptability in complex and evolving environments.

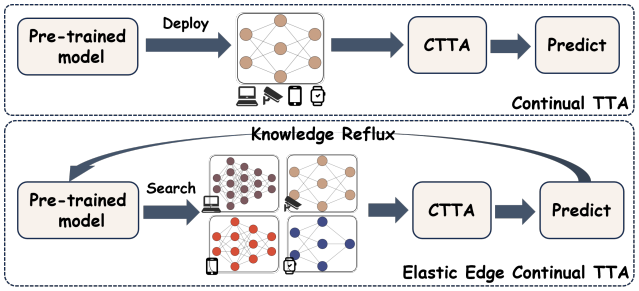

Figure 1: Traditional CTTA methods typically deploy a fixed pre-trained model across all edge devices, which limits its adaptation to heterogeneous resource constraints. In contrast, the Elastic Edge CTTA framework performs device-specific elastic submodel search from a pre-trained model according to the computational budget of device. After continual adaptation, the acquired knowledge is distilled back into the pre-trained model in a structure-aware manner, enhancing its internal paths and improving the quality of future submodel searches.

CTTA has recently been committed to improving the scalability and practicality of the model, making significant progress (Wang et al. 2022; Döbler, Marsden, and Yang 2023). However, most existing approaches rely on the online adaptation of a fixed architecture, which limits adaptability in resource-constrained edge environments. A single model structure cannot accommodate the diverse computational capabilities and adaptation needs of heterogeneous edge devices. This underscores the need for a more flexible edge adaptation paradigm, where model capacity can be dynamically adjusted to match device capabilities, thereby overcoming resource bottlenecks and enabling more efficient continual adaptation.

To address that, we propose Elastic Edge CTTA, shown in Fig. 1, a novel paradigm that adapts not only the model parameters but also its architecture to meet the resource and adaptation needs of edge devices. Instead of deploying a fixed model, our approach dynamically generates a specialized model for each device to strike a balance between ex-

*Corresponding author.

pressiveness and efficiency. Furthermore, the adaptation process is designed to form a closed loop, allowing edge-side experience to enhance the system over time. Therefore, a key challenge lies in deriving model architectures that can flexibly accommodate diverse edge constraints while maintaining adaptation capability after deployment.

Neural Architecture Search (NAS) has emerged as a powerful tool for discovering high-performance model architectures under various constraints (Ma et al. 2024). While several works have explored applying NAS to domain adaptation (DA) tasks by incorporating target-domain signals during training, these approaches remain limited to offline settings where source and target data are accessible during search (Peng et al. 2020; Broni-Bediako et al. 2024; Hoyer, Dai, and Van Gool 2022; Xu et al. 2024c; Li et al. 2020; Deng et al. 2021). Consequently, the resulting architectures are statically optimized and cannot adapt to evolving data distributions after deployment. Furthermore, most NAS methods designed for domain adaptation ignore device heterogeneity and generate a single shared architecture across all deployment environments. In CTTA scenarios, edge models must adapt to device-specific resource constraints while maintaining continual adaptability to changing environments. Therefore, integrating architecture search with online adaptation presents a critical challenge for applying NAS to CTTA tasks.

This paper proposes a novel elastic learning paradigm for CTTA in dynamic edge scenarios, offering solutions for model architecture design, dynamic adaptation, and continuous evolution. Specifically, a pre-trained binary Supernet serves as the foundation for resource-aware architecture search, enabling the generation of personalized sub-models adapted to diverse edge constraints. Unlike prior CTTA approaches that adopt fixed architectures, our method dynamically selects optimal edge models tailored to device-specific resource budgets. Furthermore, to enhance the adaptability of the edge one, we design a pluggable and lightweight online fine-tuning strategy by inserting low-rank adapters into a frozen backbone, enabling continual self-supervised adaptation with minimal overhead. Finally, a structure-aware knowledge reflux mechanism distills adapted edge models back to structurally aligned Supernet paths, improving the quality of future architecture search and enabling continuous model evolution. The proposed method demonstrates competitive accuracy, efficient resource usage, and robust adaptability across various benchmark datasets.

Contributions. The highlights of the paper are threefold: 1) Based on a comprehensive analysis of existing methods, we propose the first elastic learning paradigm for CTTA. A resource-aware architecture search generates personalized edge models for diverse devices, followed by lightweight online adaptation to handle distribution shifts during deployment. To support progressive model refinement, we further introduce a knowledge reflux mechanism that feeds the adaptation experience back into the Supernet, enabling continual evolution without retraining. 2) We design a modular fine-tuning mechanism compatible with mainstream adaptation losses, which inserts low-rank trainable adapters into a frozen binary backbone. This pluggable scheme allows flexible integration with various supervision signals used in CTTA, while maintaining high efficiency. 3) Extensive experiments show that the searched models achieve a strong balance between accuracy and efficiency. After knowledge reflux, re-searched models exhibit further improvements, confirming the effectiveness of the strategy in enhancing future search quality.

Related Work

Test-time adaptation. Test-time adaptation (TTA), also known as source-free domain adaptation (Boudiaf et al. 2023; Yang et al. 2021; Zang et al. 2023), is the process of adapting a source model to a target domain distribution without relying on any source domain data. Contrastive learning has also found applications in TTA (Liang, He, and Tan 2024; Yang et al. 2024).

Continual Test-Time Adaptation (CTTA) refers to the setting where the target domain evolves over time, posing additional challenges to conventional TTA techniques. The pioneering work in (Wang et al. 2022) introduces a teacher-student framework that leverages bi-average pseudo labels and stochastic weight resets. Inspired by the observation that mean teacher outputs tend to exhibit greater stability than single model predictions (Tarvainen and Valpola 2017), subsequent self-supervised CTTA methods (Gan et al. 2023; Liu et al. 2023; Xu, Yan, and Deng 2025; Döbler, Marsden, and Yang 2023) extend this paradigm. Meanwhile, another line of research improves model adaptation by minimizing prediction entropy, either through updating normalization statistics (Wang et al. 2020; Gong et al. 2022) or selectively fine-tuning model components (Song et al. 2023; Ni et al. 2023; Xu et al. 2024a; Yan et al. 2024). Despite their progress, these approaches rely on a fixed pre-trained backbone, limiting deployment efficiency on heterogeneous edge devices with diverse resource constraints.

Neural Architecture Search. Neural Architecture Search (NAS) has become a central paradigm for automating the design of neural networks, achieving notable improvements over manually crafted Convolutional Neural Networks in numerous applications (Liu et al. 2018; Real et al. 2019; Ren et al. 2021; Liu et al. 2019; Guo et al. 2020). Among various NAS approaches, one-shot methods (Bender et al. 2018; Liu, Simonyan, and Yang 2018; Guo et al. 2020; Su et al. 2021) are widely adopted due to their ability to amortize the cost of architecture evaluation through weight sharing.

The search process can be organized in two general paradigms: decoupled and joint optimization. In the decoupled setting, the supernet is fixed, after which candidate architectures are evaluated using techniques such as random sampling (Bender et al. 2018; Li and Talwalkar 2020), evolutionary search (Guo et al. 2020), or Monte Carlo Tree Search (MCTS)(Wang et al. 2021). In contrast, joint optimization alternates between supernet training and search guidance updates, as seen in differentiable NAS(Liu, Simonyan, and Yang 2018), reinforcement learning-based controllers (Pham et al. 2018), or distribution-based methods (Su et al. 2021). While joint strategies can improve search effectiveness, they typically incur higher computational overhead due to repeated model updates. Tree-based

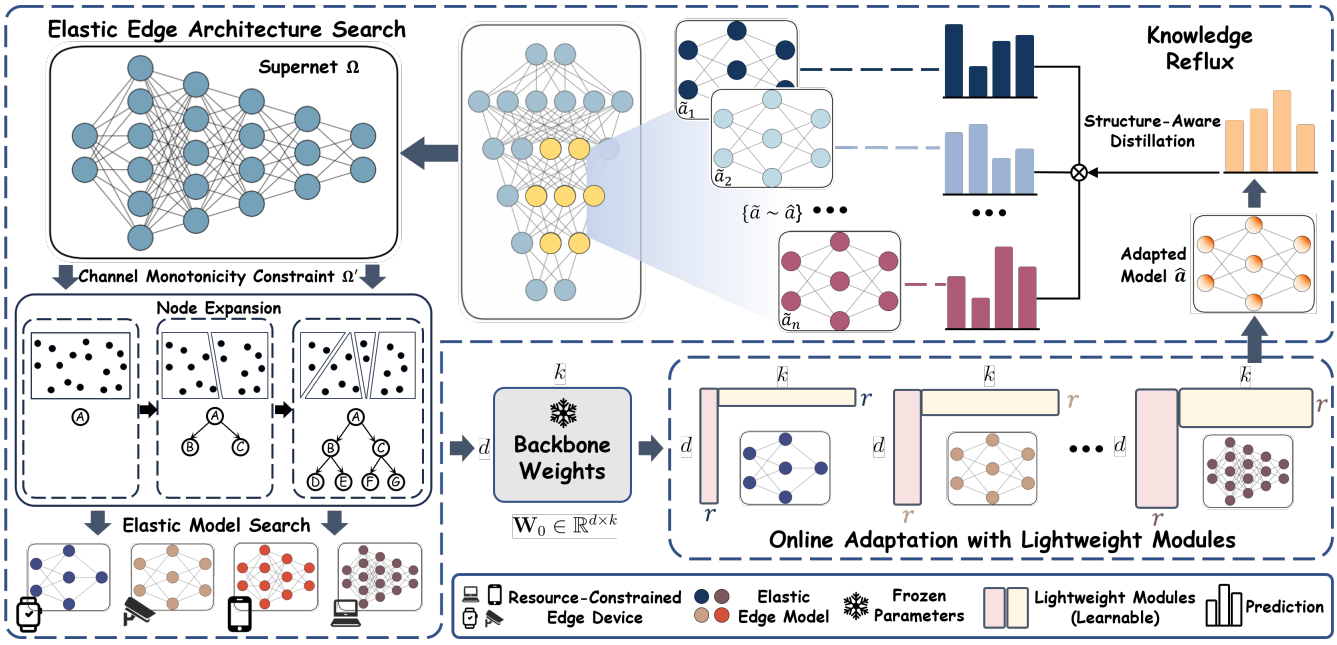

Figure 2: An overview of our method. The framework consists of three parts: (1) Elastic Edge Architecture Search generates device-specific submodels from a pre-trained binary Supernet. A channel monotonicity constraint is applied to reduce redundant configurations, and MCTS is used to progressively construct layer-wise architectures under resource constraints. (2) Lightweight Online Adaptation retains the binary backbone and inserts low-rank trainable modules into selected layers for localized updates. (3) Structure-Aware Knowledge Reflux selects structurally similar subpaths from Ω' and distills supervision from the fine-tuned edge model. Each student shares weights with the Supernet and is optimized using both hard labels and soft outputs from the teacher model.

search, such as MCTS, introduces a hierarchical perspective on the architecture space and mitigates weight sharing interference, making it well-suited for fine-grained exploration. Considering that, we propose a new strategy to elastically search for performance-optimal models tailored to the computational budgets of different devices.

Method

Problem Definition and Motivation

Edge model may encounter out-of-distribution samples caused by natural variations or device-specific factors during inference, leading to overall performance degradation. In contrast, continual test-time adaptation (CTTA) can effectively solve the problem, which uses a pre-trained model on a source domain $D_S = \{(x_i, y_i)\}$, followed by adaptive updates across a series of target domains D_t, where n denotes the number of consecutive target domain sequences. Throughout this process, the source domain data remains inaccessible, and the data for each target domain is utilized only once. The distribution of target domains ($D_t, t = 1, 2, 3...$) changes continuously over time.

Constrained by the mismatch between model complexity and device-specific resource budgets, we propose an Elastic Edge CTTA framework, as illustrated in Fig. 2. In this framework, a pre-trained binary Supernet defines a discrete architectural space Ω, from which a personalized model is elastically searched for edge device under its resource constraint. Formally, the goal is to search for an architecture $a \in \Omega$. The selected model is then deployed and equipped with a pluggable lightweight online adaptation mechanism that enables continual adjustment to distribution shifts using unlabeled target samples. Finally, the adapted model $\hat{a}$ is used to guide a structure-aware distillation process that updates structurally similar paths within the Supernet, allowing knowledge from adapted edge models to be integrated without full retraining.

Elastic Edge Architecture Search

Redundant and structurally invalid configurations are prevalent in the initial architectural space, resulting in a large and inefficient search domain. To reduce the search space and maintain structural consistency, we impose a channel-width monotonicity constraint, which requires that the channel size of any lower layer does not exceed that of its preceding layer. This constraint prunes infeasible candidates and preserves a coherent subspace $\Omega' \subseteq \Omega$. With shared weights, candidate architectures from Ω' can be efficiently evaluated on the validation set without individual training.

The architecture search is formulated as selecting an optimal structure $a \in \Omega'$ that satisfies the resource constraint while maximizing estimated accuracy:

$$a = \arg\max_{a' \in \Omega'} \text{Acc}(a') \quad \text{s.t. Cost}(a') \leq \text{Budget}, \qquad (1)$$

where Budget denotes the resource limit of the target device obtained via profiling tools, $\text{Cost}(a')$ is the estimated

parameter count of architecture a', and $\text{Acc}(a')$ is its accuracy estimate under shared weights on the validation set.

Monte Carlo Tree Search (MCTS) (Browne et al. 2012) is adopted for architecture construction. It balances exploration and exploitation in sequential decisions, making it suitable for constructing layer-wise architectures. Each node in the search tree represents a partially defined architecture, and each transition corresponds to the addition of a new layer configuration.

At each layer, candidate configurations are generated by selecting the channel width, kernel size, and number of groups. From each partial architecture, the remaining layers are completed via random sampling to construct a full candidate a', whose accuracy and computational cost are estimated using shared weights. Given the sensitivity of binary networks to structural changes, multiple forward passes are performed for each a', and their results are averaged to reduce evaluation noise. We define the reward for each candidate as follows:

$$R(a') = \frac{1}{K} \sum_{k=1}^{K} \left[\alpha \cdot \text{Acc}^{(k)}(a') - \beta \cdot \text{Cost}(a') \right], \quad (2)$$

where α and β are weighting factors for accuracy and cost, and K is the number of sampling runs. After each simulation, the reward is propagated along the search path to update the statistics of visited nodes. During node selection, MCTS employs the Upper Confidence Bound (UCB) strategy to balance exploration and exploitation:

$$\text{UCB}(s) = \bar{R}_s + \eta \cdot \sqrt{\frac{\ln N_p}{N_s}}, \quad (3)$$

where $\bar{R}_s$ is the average reward of candidate architectures sampled from node s, N_p and N_s are the visit counts of the parent node and the current node respectively, and η is the exploration coefficient.

The search is terminated after a fixed number of simulations T. Finally, among all sampled candidates, we select the architecture with the highest reward under the resource constraint as the target sub-network:

$$a = \arg\max_{a' \in \Omega'} R(a') \quad \text{s.t.} \quad \text{Cost}(a') \leq \text{Budget}. \quad (4)$$

The selected architecture a is then deployed to the corresponding edge device, where it undergoes online adaptation to address distribution shifts in unseen target domains.

Lightweight Online Adaptation

Architectures searched by conventional NAS methods are typically optimized on static datasets and fixed training objectives, making them less effective when deployed in distribution-shifted environments, especially under source-free continual test-time adaptation scenarios. To address this issue, we introduce a lightweight online fine-tuning mechanism that modifies only a small set of inserted trainable modules while keeping the binary backbone fixed.

Let the frozen backbone weights of the sub-model be denoted by $\mathbf{W}_0 \in \mathbb{R}^{d \times k}$, where d and k are the input and output channel dimensions of the current layer, respectively. We insert low-rank residual modules into the final layer to refine the original binary weights using a bottleneck structure, expressed as:

$$\mathbf{W} = \mathbf{W}_0 + \mathbf{B}\mathbf{A}, \quad \mathbf{B} \in \mathbb{R}^{d \times r}, \quad \mathbf{A} \in \mathbb{R}^{r \times k}, \quad (5)$$

$\mathbf{B}$ and $\mathbf{A}$ are the trainable parameters, and r is the rank that controls the trade-off between representational capacity and computational overhead. As larger models typically possess richer feature spaces, we increase r for complex architectures to ensure sufficient adaptation capacity, while still maintaining $r \ll \min(d, k)$ for efficiency.

Before deployment, we perform a warm-up initialization of the inserted modules. The parameters $\mathbf{B}$ and $\mathbf{A}$ are trained with cross-entropy loss using the source domain data D_S, providing a stable initialization without incurring additional communication overhead. Once deployed, the model receives a sequence of unlabeled target domain samples every t steps, denoted as $D_t^{(t)} = \{x_i^{(t)}\}$. For each sample $x_i^{(t)}$ in the current batch, the prediction $p_i^{(t)}$ is computed using the updated parameters from the previous round:

$$p_i^{(t)} = f(x_i^{(t)}; \mathbf{W}_0 + \mathbf{B}^{(t-1)}\mathbf{A}^{(t-1)}), \quad (6)$$

where $f(\cdot;\cdot)$ denotes the forward function composed of the frozen binary backbone and the inserted adaptation modules. Pseudo-labels $\hat{y}_i^{(t)}$ are generated for unlabeled target samples based on the most probable class in $p_i^{(t)}$:

$$\hat{y}_i^{(t)} = \arg\max_c p_i^{(t)}[c]. \quad (7)$$

Using $\hat{y}_i^{(t)}$ as the supervision signal, only the inserted modules are updated. After a gradient update, the adapted parameters at step t are denoted as $\tilde{\mathbf{B}}^{(t)}$ and $\tilde{\mathbf{A}}^{(t)}$. To ensure stability during the update process, we apply an exponential moving average (EMA) strategy to smooth the parameter updates at each step, defined as:

$$\mathbf{B}^{(t)} = \lambda \cdot \mathbf{B}^{(t-1)} + (1 - \lambda) \cdot \tilde{\mathbf{B}}^{(t)}, \quad (8)$$

$$\mathbf{A}^{(t)} = \lambda \cdot \mathbf{A}^{(t-1)} + (1 - \lambda) \cdot \tilde{\mathbf{A}}^{(t)}, \quad (9)$$

where $\lambda \in [0, 1)$ is the EMA decay coefficient that balances the influence of historical and current updates.

Additionally, we support the integration of pluggable supervision signals. We define a regularization term $\mathcal{R}^{(t)}$ at each time step t as:

$$\mathcal{R}^{(t)} = \sum_j \gamma_j \cdot \mathcal{L}_j(x_i^{(t)}, f(x_i^{(t)})), \quad (10)$$

where each $\mathcal{L}_j$ denotes the j-th auxiliary objective (e.g., entropy minimization or batch normalization alignment), and γ_j is its weighting factor. This design enables the seamless injection of task-specific guidance without altering the core adaptation pipeline.

We define the core TTA loss at step t as:

$$\mathcal{L}_{\text{TTA}}^{(t)} = \frac{1}{|\mathcal{D}_t^{(t)}|} \sum_{x_i \in \mathcal{D}_t^{(t)}} \mathcal{L}_{\text{CE}} \left(\hat{y}_i^{(t)}, f\left(x_i; \mathbf{W}_0 + \mathbf{B}^{(t)}\mathbf{A}^{(t)} \right) \right), \quad (11)$$

only $\mathbf{B}^{(t)}$ and $\mathbf{A}^{(t)}$ are updated, guided by both pseudo-labels and the optional supervision in $\mathcal{R}^{(t)}$, while EMA smoothing ensures stability across time steps. This modular formulation enables robust and extensible adaptation under evolving target distributions.

The overall loss function optimized at step t is given by:

$$\mathcal{L}_{\text{total}}^{(t)} = \mathcal{L}_{\text{TTA}}^{(t)} + \mathcal{R}^{(t)}. \tag{12}$$

Knowledge Reflux

After completing the model search and online adaptation on the edge, we further propose a structure-aware knowledge reflux mechanism to feed effective knowledge from the adapted edge model back into the Supernet. The fine-tuned model $\hat{a}$ serves as the teacher. A set of student models $\mathcal{S}(\hat{a}) \subseteq \Omega'$ is selected from the top-M most frequently visited architectures during the search. Each student $\tilde{a} \in \mathcal{S}(\hat{a})$ is instantiated by extracting the corresponding sub-path from the Supernet and directly reusing its weights.

The use of visit frequency as a selection criterion is consistent with the reward driven nature of MCTS, as frequently visited architectures tend to achieve better performance and exhibit structural similarity to the adapted model $\hat{a}$. This selection ensures that the chosen student models remain close to the teacher in structure and are well supported by prior search evidence.

Distillation is performed using a dataset $\mathcal{D}_{\text{dist}} \subseteq D_S$ consistent with the Supernet pretraining. The student paths $\tilde{a}$ are supervised by the teacher $\hat{a}$ under a standard soft-label distillation loss that combines cross-entropy and Kullback–Leibler divergence:

$$\mathcal{L}_{\text{dist}} = \mathbb{E}_{\tilde{a}\sim\mathcal{S}(\hat{a})}\, \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{dist}}} \Big[\mathcal{L}_{\text{CE}}(\tilde{a}(x), y) + \zeta \cdot \text{KL}(\hat{a}(x) \,\|\, \tilde{a}(x)) \Big], \tag{13}$$

where $\mathcal{L}_{\text{CE}}(\cdot)$ denotes the cross-entropy loss between the student prediction and the ground-truth label y. The KL divergence term encourages the student to match the soft output distribution of the teacher $\hat{a}$. The weighting factor ζ balances the influence of hard-label supervision and distribution matching.

All student structures share and update the weights of the Supernet during distillation, transferring knowledge to regions of the architectural space that are structurally related to $\hat{a}$. This localized update improves the generalization and structural expressiveness of the Supernet within a limited update scope, without requiring full retraining.

Experiments

Experimental Settings

Datasets. We evaluate our method on two widely used distribution shift benchmarks: ImageNet-C (Hendrycks and Dietterich 2019a) and CIFAR100-C (Hendrycks and Dietterich 2019b). ImageNet-C applies four major categories of corruptions—noise, blur, weather, and digital—on the ImageNet validation set, resulting in 15 distinct corruption types. Each corruption is further divided into 5 severity levels (from 1 to 5), with higher levels indicating stronger distribution shifts. CIFAR100-C is constructed similarly by introducing the same 15 corruption types and 5 severity levels to the CIFAR-100 test set, forming a corrupted evaluation set of 10,000 images across 100 classes.

Implementation details. We followed the implementation details specified in previous works to ensure consistency. We use the Adam optimizer with a learning rate of 1e-5 for Vit-S (Dosovitskiy 2020) and 1e-3 for Resnet50 (He et al. 2016). We set the architecture search budget to T=500 simulations and use K=3 runs to reduce evaluation noise. The reward weights are set to α=1.0 and β=0.05. The EMA decay coefficient is fixed at λ=0.9, and we select M=5 student models for distillation. For latency and resource measurements, we simulate deployment using static graph profiling tools. The theoretical FLOPs and parameter counts are computed using standard model profiling tools.

Evaluation Metric. We report evaluation results on both corruption robustness and classification performance. For TTA tasks, we use Average Accuracy (Avg.Acc.) over all corruption types and severity levels (following prior works), alongside computational cost metrics including FLOPs, Params, and Memory (Mem.), which quantifies peak GPU memory usage during inference. For clean classification performance, we report Best Top-1 Accuracy (Best Acc.) on standard validation sets. In addition, we report the Latency (ms/image) for real-time inference comparison across architectures with different complexities.

Comparisons with the State-of-the-arts

TTA Method. We compare our proposed method with several state-of-the-art CTTA approaches, including BN (Schneider et al. 2020), Tent (Wang et al. 2020), CoTTA (Wang et al. 2022), SAR (Niu et al. 2023), VDP (Gan et al. 2023), ViDA (Liu et al. 2023), REM (Han, Na, and Hwang 2025) and PALM (Maharana, Zhang, and Guo 2025). To ensure a fair comparison, we select two widely used backbones with comparable parameter scales: ResNet-50 (25.6M) and ViT-S (22M). These architectures represent strong baselines within the 25M parameter range, balancing low computational cost and high adaptation performance. For each compared method, we replace the original backbone with either ResNet-50 or ViT-S to build two sets of baseline models, allowing a comprehensive evaluation of our method against both convolutional and transformer-based architectures.

Based on this setup, we apply our method to search and fine-tune a binarized model under a 22M parameter budget, consistent with ViT-S. As shown in Tab. 1, our base method (using only CE loss) achieves an average accuracy of 56.23% on ImageNet-C, already outperforming all CNN-based methods and most ViT-based TTA baselines, including TENT (49.96%) and ViDA (52.17%). This demonstrates the strong robustness of our architecture even without additional regularization. When enhanced with auxiliary TTA losses, performance further improves: incorporating SAR, REM, and PALM results in 57.13%, 57.60%, and a peak of 58.01% accuracy, respectively. In each case, our framework not only benefits from these objectives but also surpasses

Method	Arch.	Params.	Gau.	Shot	Imp.	Def.	Gla.	Mot.	Zoom	Snow	Fro.	Fog	Bri.	Con.	Ela.	Pix.	JPEG	Avg. ↑
BN	CNN	25.6M	15.8	16.7	15.3	18.7	19.3	29.8	41.7	35.8	35.0	50.5	65.9	31.0	49.3	51.7	42.0	33.71
TENT			28.0	30.1	28.1	29.9	29.5	42.2	49.7	46.2	41.5	57.7	67.1	30.0	55.7	58.3	52.5	43.10
CoTTA			19.9	31.8	35.3	30.4	34.4	40.2	43.2	39.3	38.6	47.7	51.8	36.0	43.5	46.7	43.1	38.79
SAR			29.6	38.4	37.8	31.5	32.8	41.1	48.6	42.9	40.2	53.3	63.7	37.7	57.3	52.3	52.3	43.97
REM			28.7	40.8	37.7	31.7	33.0	42.5	49.7	43.2	40.8	54.0	63.9	38.2	53.4	56.9	54.1	44.57
PALM			30.5	40.8	38.5	33.1	34.5	44.7	50.3	43.6	41.5	54.4	64.1	39.2	54.1	56.9	55.3	45.43
TENT	ViT	22M	29.4	38.1	37.4	44.8	41.2	50.9	44.2	51.6	51.1	60.4	71.2	58.5	61.6	61.5	61.0	49.93
CoTTA			45.6	58.6	58.4	40.1	50.9	50.4	44.4	46.1	51.4	52.2	57.6	35.6	55.8	55.4	50.0	50.51
VDP			27.2	30.8	56.4	57.8	22.9	53.2	60.8	61.5	58.7	48.2	74.5	73.0	50.3	46.8	49.6	49.96
ViDA			29.9	39.3	58.0	58.8	34.8	54.8	63.5	62.1	63.4	54.4	68.5	58.1	50.4	45.3	52.9	52.17
SAR			40.1	47.8	49.9	47.4	46.8	54.5	50.2	55.6	58.2	63.0	73.1	58.2	51.7	64.8	55.1	55.15
REM			39.9	48.5	53.4	47.4	47.1	55.3	52.8	59.5	60.5	63.7	74.8	59.9	53.1	66.7	65.5	56.54
PALM			40.5	49.7	53.9	47.6	48.2	56.4	53.1	59.9	60.4	63.8	75.2	60.8	53.7	67.0	66.3	57.10
Ours (CE)	BNN	22M	39.7	47.9	53.2	47.5	46.8	55.0	52.7	58.3	60.1	63.4	74.9	59.4	52.6	66.4	65.5	56.23
Ours + SAR			40.5	48.3	53.9	48.2	47.3	55.6	**54.1**	59.7	60.9	64.4	75.2	60.9	54.1	66.8	**67.1**	57.13
Ours + REM			**41.3**	49.2	54.0	47.7	47.8	56.2	53.3	**61.2**	60.9	63.8	75.9	60.4	53.7	**68.0**	66.1	57.30
Ours + PALM			41.1	**50.6**	**56.3**	**48.6**	**48.8**	**57.0**	53.9	61.0	**61.4**	**64.6**	**76.5**	**61.6**	**54.3**	67.5	66.9	**58.01**

Table 1: Comparison with different CTTA methods on ImageNet-C (corruption severity level 5), with the model size fixed at 22M parameters. And rank $r = 2$ is selected to ensure minimal parameter overhead in the adaptation modules. The results include our method in its base form (adapted using only CE loss) and with additional supervision signals from other methods. Despite the low parameter budget, our method achieves the highest accuracy.

Method	Arch.	FLOPs	Params	Avg. (%) ↑	Mem.(MB) ↓
BN	CNN	4.1G	25.6M	55.5	91
TENT				54.1	926
CoTTA				59.8	2064
SAR				55.9	838
REM				61.8	762
PALM				64.6	1538
TENT	ViT	4.6G	22M	67.9	1537
CoTTA				65.2	3230
VDP				68.0	2157
ViDA				72.7	1479
SAR				71.4	1285
REM				72.8	1183
PALM				74.6	1742
Ours (CE)	BNN	4.1G	22M	73.5	945
Ours + SAR				74.7	960
Ours + REM				75.2	**738**
Ours + PALM				**76.9**	1027

Table 2: Comparison of different adaptation methods on CIFAR100-C in terms of computational cost, accuracy, and memory usage. Our method achieves competitive accuracy while maintaining lower resource consumption.

the original implementations of the corresponding methods (e.g., PALM: 57.10%), highlighting its superior compatibility and effectiveness.

To further assess generality, we incorporate regularization schemes from existing methods into our framework and, under all settings, obtain consistently better results under the same parameter and computational budgets, which supports its plug-and-play use with diverse test-time adaptation strategies. We also report GFLOPs for each model and note that, although GFLOPs can underestimate the hardware efficiency of binary neural networks due to reduced memory access and energy cost, our method still uses significantly fewer FLOPs than all full-precision counterparts.

Tab. 2 compares performance on CIFAR100-C using ResNet-50 and ViT-S as CNN and ViT backbone under matched resource budgets. Our binarized submodel achieves the highest average accuracy (73.5%) among all methods,

Method	Best Acc. (%)	FLOPs (M)	Params (M)	Latency ms/img
MobileNetV2	72.0	300	3.4	2.9
GreedyNAS-C	75.2	312	3.9	2.4
EAEPSO	75.6	280	6.0	2.1
IS-DARTS	76.2	284	4.7	1.9
HEP-NAS	76.3	280	4.9	1.5
EEAS (a)	**76.5**	**280**	**3.4**	**1.4**
EEAS + fine-tuning ($\hat{a}$)	**77.3**	**280**	**3.5**	**1.4**
EEAS + re-search (a^*)	**77.9**	**280**	**3.4**	**1.2**

Table 3: Comparison of different NAS methods on the ImageNet dataset in terms of accuracy, computational cost, and search efficiency. And rank $r = 1$ is selected to ensure minimal parameter overhead in the adaptation modules.

with only 4.1 GFLOPs and 945MB memory usage, demonstrating superior efficiency and adaptation performance. Although some CNN-based methods (e.g., BN and SAR) have lower memory usage, their accuracy drops significantly (e.g., BN: 55.5%), highlighting the trade-off between memory savings and adaptation capability. ViT-based methods generally achieve higher accuracy than CNNs but require substantially more memory (e.g., CoTTA: 3230MB, VDP: 2157MB), making them less suitable for edge deployment. In contrast, our BNN maintains a favorable balance of accuracy and efficiency. When further enhanced with auxiliary losses from existing TTA methods, performance improves consistently: 74.7% with SAR, 75.2% with REM, and 76.9% with PALM. In all cases, our method not only benefits from the added supervision but also outperforms the original implementations of these methods, validating its strong adaptability and plug-and-play capability.

NAS Method. Tab. 3 presents the performance of our proposed resource-aware architecture search method on the ImageNet-1K classification task, compared against several mainstream lightweight NAS methods, including MobileNetV2, GreedyNAS-C (You et al. 2020), EAEPSO

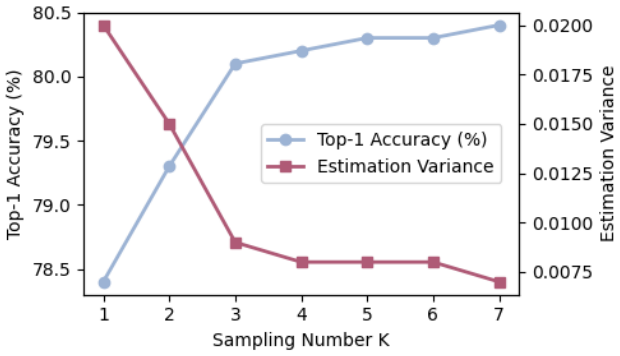

Figure 3: Effect of sampling number K on search performance during architecture evaluation. The blue and red curves indicate Top-1 accuracy, respectively, where higher K improves stability and final accuracy, with diminishing returns beyond $K = 3$.

Budgets	Acc. (%) w/ Constraint	Acc. (%) w/o Constraint
3 M	54.1	52.3
6 M	59.6	57.4
15 M	65.8	63.0
40 M	69.3	67.5

Table 4: Ablation study on the monotonicity constraint of channel width. Estimated top-1 accuracy of models searched under different parameter budgets on ImageNet (3M/6M/15M/40M with $r = 1/1/2/4$). Applying the constraint consistently improves performance across budgets.

(Yuan, Xue, and Zhang 2023), IS-DARTS (He et al. 2024), and HEP-NAS (Li et al. 2025). All models are designed under a similar computational budget of approximately 280M FLOPs, with parameter counts ranging from 3M to 6M. Additionally, all baseline methods adopt a one-shot search strategy that outputs a single network architecture satisfying resource constraints, consistent with our setting of generating a single sub-model via one-time search.

The results show that our binary model achieves 76.5% top-1 accuracy, outperforming all baselines and demonstrating the effectiveness of our search framework for large-scale classification. Notably, this is achieved with only 3.4M parameters and 1.4ms latency, surpassing efficiency-focused methods like GreedyNAS-C(2.4ms) and HEP-NAS(1.5ms). These results indicate that our strategy effectively balances accuracy and efficiency under tight resource budgets.

We further compare the adapted model ($\hat{a}$), fine-tuned with lightweight modules, and the distilled model (a^*), re-searched after knowledge reflux. Due to added low-rank parameters, $\hat{a}$ has slightly more parameters(3.5M). In contrast, a^* is searched under the original 3.4M constraint, inheriting knowledge while retaining minimal size. With identical FLOPs, a^* achieves the best accuracy(77.9%) and lowest latency(1.2 ms), validating the effectiveness of our structure-aware reflux mechanism.

Memory Usage	Top-1 Acc. % a (Initial)	$\hat{a}$ (Fine-tuned)	a^* (Re-search)
400 MB	73.1	75.3	76.5
600 MB	75.2	77.0	78.4
800 MB	77.6	78.9	80.1
1600 MB	78.9	80.2	81.3

Table 5: Top-1 accuracy versus memory usage on ImageNet-1K (batch size = 16) for different stages of our pipeline. Circles, squares, and triangles represent the initial models (a), fine-tuned models ($\hat{a}$), and distilled-and-researched models (a^*), respectively.

Ablation Study

Simulation Sampling Times. Fig. 3 shows how the sampling number K affects the stability and quality of architecture evaluation. Increasing K significantly reduces estimation variance, yielding more reliable rewards and higher final accuracy of the searched submodels, which confirms the benefit of multi-sample averaging in mitigating evaluation noise for weight-sharing binary networks. However, gains saturate beyond $K = 3$, while computational cost grows roughly linearly with K, so we set $K = 3$ by default as a favorable trade-off between performance and efficiency.

Monotonicity Constraint on Channel Width. Tab. 4 visualizes the impact of the channel width monotonicity constraint on the search space. Without the constraint, arbitrary width transitions are allowed, producing a redundant and complex structure space; with the constraint, only non-increasing transitions are permitted, yielding a more compact and efficient space. The plot compares sub-model accuracy under different parameter budgets, where constrained models consistently perform better, with gains of about 2.2% and 1.9% under the 6M and 15M budgets.

Knowledge Reflux. Tab. 5 presents the performance progression of our pipeline across three stages under identical memory constraints. The initially searched models (a) exhibit limited performance, reflecting insufficient adaptation to the target domain. Fine-tuning on the edge yields personalized models ($\hat{a}$) with consistent accuracy improvements, validating the benefit of lightweight online adaptation. Notably, the re-searched models (a^*), obtained from the distilled Supernet, outperform the initial a models without additional fine-tuning, and in some cases even exceed $\hat{a}$.

Conclusion

We propose an elastic CTTA framework for edge devices that combines resource-aware architecture search, lightweight online adaptation, and structure-aware knowledge reflux. Personalized models are sampled from a pre-trained binary Supernet, adapted with low-rank adapters at minimal overhead, and their adaptation experience is distilled into structurally similar paths to guide subsequent search. Experiments across benchmarks show strong accuracy and efficiency, and models re-searched after knowledge reflux obtain further gains, confirming robust and sustainable edge adaptation.

Acknowledgments

This work is supported in part by the National Key Research and Development Program of China (No. 2023YFC3305600), Joint Fund of Ministry of Education of China (8091B022149, 8091B02072404), National Natural Science Foundation of China (62132016, 62571412, and 62571393), Key Research and Development Program of Shaanxi (2024GX-YBXM-127) and National Key Laboratory Foundation of China (Grant No. HTKJ2024KL504011).

References

Arnold, E.; Al-Jarrah, O. Y.; Dianati, M.; Fallah, S.; Oxtoby, D.; and Mouzakitis, A. 2019. A survey on 3d object detection methods for autonomous driving applications. *IEEE Transactions on Intelligent Transportation Systems*, 20(10): 3782–3795.

Bender, G.; Kindermans, P.-J.; Zoph, B.; Vasudevan, V.; and Le, Q. 2018. Understanding and simplifying one-shot architecture search. In *International conference on machine learning*, 550–559. PMLR.

Boudiaf, M.; Denton, T.; van Merriënboer, B.; Dumoulin, V.; and Triantafillou, E. 2023. In Search for a Generalizable Method for Source Free Domain Adaptation.

Broni-Bediako, C.; et al. 2024. Unsupervised Domain Adaptation Architecture Search with Self-Training for Land Cover Mapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*.

Browne, C. B.; Powley, E.; Whitehouse, D.; Lucas, S. M.; Cowling, P. I.; Rohlfshagen, P.; Tavener, S.; Perez, D.; Samothrakis, S.; and Colton, S. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1): 1–43.

Deng, X.; Zuo, Y.; Tian, Y.; and Newsam, S. 2021. AutoDAPT: Automated Segmentation Network Search for Unsupervised Domain Adaptation. arXiv:2106.13227.

Döbler, M.; Marsden, R. A.; and Yang, B. 2023. Robust mean teacher for continual and gradual test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7704–7714.

Dosovitskiy, A. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Gan, Y.; Bai, Y.; Lou, Y.; Ma, X.; Zhang, R.; Shi, N.; and Luo, L. 2023. Decorate the Newcomers: Visual Domain Prompt for Continual Test-Time Adaptation. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, 7595–7603.

Gong, T.; Jeong, J.; Kim, T.; Kim, Y.; Shin, J.; and Lee, S.-J. 2022. NOTE: Robust continual test-time adaptation against temporal correlation. *Advances in Neural Information Processing Systems*, 35: 27253–27266.

Guo, Z.; Zhang, X.; Mu, H.; Heng, W.; Liu, Z.; Wei, Y.; and Sun, J. 2020. Single path one-shot neural architecture search with uniform sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, 544–560. Springer.

Han, J.; Na, J.; and Hwang, W. 2025. Ranked Entropy Minimization for Continual Test-Time Adaptation. In *Proceedings of the 42nd International Conference on Machine Learning*.

He, H.; Liu, L.; Zhang, H.; and Zheng, N. 2024. IS-DARTS: Stabilizing DARTS through Precise Measurement on Candidate Importance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 12367–12375. AAAI.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hendrycks, D.; and Dietterich, T. 2019a. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*.

Hendrycks, D.; and Dietterich, T. 2019b. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *International Conference on Learning Representations*.

Hoyer, L.; Dai, D.; and Van Gool, L. 2022. DAP-NAS: Improving Network Architectures and Training Strategies for Domain-Adaptive Semantic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Li, J.; Zhang, J.; Wang, F.; and Ma, L. 2025. HEP-NAS: Towards Efficient Few-shot Neural Architecture Search via Hierarchical Edge Partitioning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Li, L.; and Talwalkar, A. 2020. Random search and reproducibility for neural architecture search. In *Uncertainty in artificial intelligence*, 367–377. PMLR.

Li, Y.; Yang, Z.; Wang, Y.; and Xu, Z. 2020. Adapting Neural Architectures Between Domains. In *Advances in Neural Information Processing Systems*, 789–798.

Liang, J.; He, R.; and Tan, T. 2024. A comprehensive survey on test-time adaptation under distribution shifts. *International Journal of Computer Vision*, 1–34.

Liu, C.; Chen, L.-C.; Schroff, F.; Adam, H.; Hua, W.; Yuille, A. L.; and Fei-Fei, L. 2019. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 82–92.

Liu, C.; Zoph, B.; Neumann, M.; Shlens, J.; Hua, W.; Li, L.-J.; Fei-Fei, L.; Yuille, A.; Huang, J.; and Murphy, K. 2018. Progressive neural architecture search. In *Proceedings of the European conference on computer vision (ECCV)*, 19–34.

Liu, H.; Simonyan, K.; and Yang, Y. 2018. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*.

Liu, J.; Yang, S.; Jia, P.; Lu, M.; Guo, Y.; Xue, W.; and Zhang, S. 2023. ViDA: Homeostatic Visual Domain Adapter for Continual Test Time Adaptation. *arXiv preprint arXiv:2306.04344*.

Ma, L.; Kang, H.; Yu, G.; Li, Q.; and He, Q. 2024. Single-domain generalized predictor for neural architecture search

system. *IEEE Transactions on Computers*, 73(5): 1400–1413.

Maharana, S. K.; Zhang, B.; and Guo, Y. 2025. PALM: Pushing Adaptive Learning Rate Mechanisms for Continual Test-Time Adaptation. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*. Code: https://github.com/sarthaxxxx/PALM.

Ni, J.; Yang, S.; Liu, J.; Li, X.; Jiao, W.; Xu, R.; Chen, Z.; Liu, Y.; and Zhang, S. 2023. Distribution-aware continual test time adaptation for semantic segmentation. *arXiv preprint arXiv:2309.13604*.

Niu, S.; Wu, J.; Zhang, Y.; Wen, Z.; Chen, Y.; Zhao, P.; and Tan, M. 2023. Towards stable test-time adaptation in dynamic wild world. *arXiv preprint arXiv:2302.12400*.

Peng, X.; Huang, Z.; Zhu, Y.; and Saenko, K. 2020. Network Architecture Search for Domain Adaptation. arXiv:2008.05706.

Pham, H.; Guan, M.; Zoph, B.; Le, Q.; and Dean, J. 2018. Efficient neural architecture search via parameters sharing. In *International conference on machine learning*, 4095–4104. PMLR.

Quiñonero-Candela, J.; Sugiyama, M.; Schwaighofer, A.; and Lawrence, N. D. 2022. *Dataset shift in machine learning*. Mit Press.

Real, E.; Aggarwal, A.; Huang, Y.; and Le, Q. V. 2019. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, 4780–4789.

Ren, P.; Xiao, Y.; Chang, X.; Huang, P.-Y.; Li, Z.; Chen, X.; and Wang, X. 2021. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4): 1–34.

Sakaridis, C.; Dai, D.; and Van Gool, L. 2021. ACDC: The adverse conditions dataset with correspondences for semantic driving scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10765–10775.

Schneider, S.; Rusak, E.; Eck, L.; Bringmann, O.; Brendel, W.; and Bethge, M. 2020. Improving robustness against common corruptions by covariate shift adaptation. *Advances in neural information processing systems*, 33: 11539–11551.

Song, J.; Lee, J.; Kweon, I. S.; and Choi, S. 2023. Ecotta: Memory-efficient continual test-time adaptation via self-distilled regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11920–11929.

Su, X.; Huang, T.; Li, Y.; You, S.; Wang, F.; Qian, C.; Zhang, C.; and Xu, C. 2021. Prioritized architecture sampling with monto-carlo tree search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10968–10977.

Tarvainen, A.; and Valpola, H. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30.

Wang, D.; Shelhamer, E.; Liu, S.; Olshausen, B.; and Darrell, T. 2020. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*.

Wang, L.; Xie, S.; Li, T.; Fonseca, R.; and Tian, Y. 2021. Sample-efficient neural architecture search by learning actions for monte carlo tree search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9): 5503–5515.

Wang, Q.; Fink, O.; Van Gool, L.; and Dai, D. 2022. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7201–7211.

Xu, C.; Lyu, G.; Yan, J.; Yang, M.; and Deng, C. 2024a. LLM Knows Body Language, Too: Translating Speech Voices into Human Gestures. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5004–5013. Bangkok, Thailand: Association for Computational Linguistics.

Xu, C.; Yan, J.; and Deng, C. 2025. Keep and Extent: Unified Knowledge Embedding for Few-Shot Image Generation. *IEEE Transactions on Image Processing*, 34: 2315–2324.

Xu, C.; Yan, J.; Yang, M.; and Deng, C. 2024b. Rethinking Noise Sampling in Class-Imbalanced Diffusion Models. *IEEE Transactions on Image Processing*, 33: 6298–6308.

Xu, C.; Yan, J.; Yang, Y.; and Deng, C. 2024c. Implicit Compositional Generative Network for Length-Variable Co-Speech Gesture Synthesis. *IEEE Transactions on Multimedia*, 26: 6325–6335.

Yan, J.; Yin, Z.; Xu, C.; Deng, C.; and Huang, H. 2024. Retrieval across any domains via large-scale pre-trained model. In *Forty-first International Conference on Machine Learning*.

Yang, S.; Wang, Y.; van de Weijer, J.; Herranz, L.; and Jui, S. 2021. Generalized Source-Free Domain Adaptation. *international conference on computer vision*.

Yang, X.; Chen, X.; Li, M.; Wei, K.; and Deng, C. 2024. A Versatile Framework for Continual Test-Time Domain Adaptation: Balancing Discriminability and Generalizability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 23731–23740.

You, S.; Huang, T.; Yang, M.; Wang, F.; Qian, C.; and Zhang, C. 2020. Greedynas: Towards fast one-shot nas with greedy supernet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1999–2008.

Yuan, X.; Xue, B.; and Zhang, M. 2023. Particle Swarm Optimization for Efficiently Evolving Deep Convolutional Neural Networks Using an Autoencoder-Based Encoding Strategy. *IEEE Transactions on Evolutionary Computation*, 1190–1204.

Zang, Z.; Shang, L.; Yang, S.; Wang, F.; Sun, B.; Xie, X.; and Li, S. Z. 2023. Boosting Novel Category Discovery Over Domains with Soft Contrastive Learning and All in One Classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 11858–11867.