

Received 12 January 2026, accepted 5 February 2026, date of publication 10 February 2026, date of current version 20 February 2026.

Digital Object Identifier 10.1109/ACCESS.2026.3663135

RESEARCH ARTICLE

Online Adaptive Slimmable Network for Source-Free Unsupervised Domain Adaptation

SEUNGMO SEO^{1,2}, JONGSU YOUN^{1,2}, SEUNGJIN JUNG^{1,2}, MINJI KWAK¹,
HEEGON JIN¹, HOJOON JUNG², AND JONGWON CHOI^{1,2}, (Member, IEEE)

¹Department of Artificial Intelligence, Chung-Ang University, Seoul 06974, South Korea

²Department of Advanced Imaging, GSAIM, Chung-Ang University, Seoul 06974, South Korea

Corresponding author: Jongwon Choi (choijw@cau.ac.kr)

This work was supported in part by the Chung-Ang University Research Scholarship Grants, in 2023; and in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by Korean Government [Ministry of Science and Information and Communications Technology (MSIT)], Development of Autonomous Performance Improvement Technology for Video Surveillance System Based on Edge-Analysis Server Connection, through the Artificial Intelligence Graduate School Program, Chung-Ang University, under Grant RS-2024-00437576 and Grant RS-2021-II211341.

ABSTRACT When the target domain is well defined, only a small portion of a deep neural network is often sufficient to achieve satisfactory performance. However, traditional model compression method typically requires access to the original dataset and additional labeled annotations, an approach that is both costly and often restricted by privacy concerns. In this study, we propose a novel slimmable adaptation framework designed for source-free unsupervised domain adaptation (SFUDA). Specifically, we introduce a block transformable network that can be slimmed into multiple sub-models without requiring any additional fine-tuning or retraining. Furthermore, we develop an online multi-path stabilization strategy that enhances the performance of these slimmed sub-models using only unlabeled target data. Comprehensive experiments across various scenarios confirm that the proposed method enables the slimmable model to adapt effectively to diverse domain shifts. Our approach surpasses existing unsupervised domain adaptation and model compression techniques that rely on labeled data.

INDEX TERMS Source-free unsupervised domain adaptation, slimmable neural network, neural network compression.

I. INTRODUCTION

Modern deep neural networks have achieved remarkable performance across a wide range of visual recognition tasks. However, deploying these models across diverse hardware platforms and rapidly evolving visual domains remains challenging. In practical scenarios, it is desirable for a single model to flexibly balance computational efficiency and generalization performance, especially when adapting to target domains that lack labeled data. To address domain shift without relying on labeled target data or direct access to source domain samples, Source-Free Unsupervised Domain Adaptation (SFUDA) has emerged as a promising approach [7], [8], [9]. SFUDA methods aim to adapt a

pre-trained source model to an unlabeled target domain without requiring access to the original source data, which is advantageous in privacy-sensitive or resource-limited environments. Representative approaches, such as SHOT [7] and DCPL [37], focus on aligning target features through self-supervised objectives. However, these methods assume a fixed model architecture and cannot accommodate dynamically re-sizable networks.

Slimmable neural networks have recently emerged as a promising solution that enables a single model to operate across multiple widths or scales, thereby dynamically adjusting to various computational resource constraints [2], [3], [4]. Recent studies including OFA [17] and SlimDA [16] have investigated slimmable and self-distillation frameworks that adaptively train networks across multiple widths. However, these approaches primarily emphasize multi-capacity

The associate editor coordinating the review of this manuscript and approving it for publication was Thomas Canhao Xu¹.

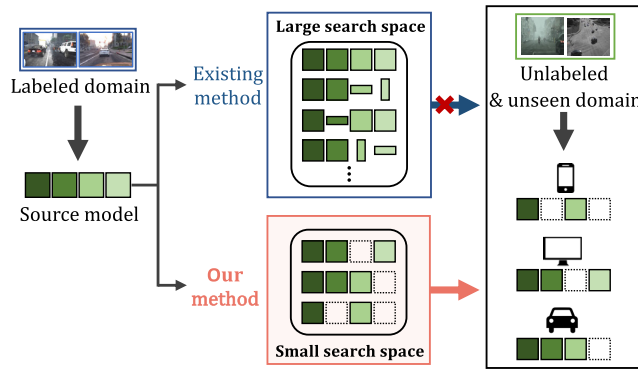


FIGURE 1. Challenging Task: Slimmable Adaptation without Labeled Data. This figure illustrates the slimmable SFUDA setting, where a source model is adapted to an unlabeled target domain while selecting among multiple sub-model configurations. Block-level compression reduces the configuration complexity compared to conventional depth/width scaling.

generalization under labeled or source-available conditions. They often optimize each sub-model independently or use uni-directional teacher-student distillation, which limits consistent adaptation across different capacities when domain shifts occur.

In this work, we unify the research directions of source-free adaptation and slimmable learning by proposing a unified slimmable adaptation framework that jointly optimizes multiple sub-models through block-level compression and hierarchical co-adaptation. Fig. 1 summarizes the slimmable adaptation setting and the resulting sub-model configuration complexity. Unlike OFA or SlimDA, our method facilitates bi-directional knowledge exchange between the full model and its sub-networks through a gradient-stabilized teacher mechanism, ensuring consistent learning behavior across varying models. This co-adaptive framework transforms the conventional one-way distillation paradigm into a mutual optimization process, allowing all sub-models to collaboratively adapt to the target domain without retraining or manual configuration.

Furthermore, we introduce a *block-level network compression* strategy that adjusts model capacity at the block level, simplifying the construction of multiple sub-models and enhancing both training efficiency and adaptability to diverse target environments. Specifically, this strategy incorporates a *bypassing block*, a lightweight structural unit that substitutes the original residual block by directly forwarding either its input or the bottleneck layer's output. This replacement effectively eliminates redundant parameters, enabling efficient block-wise compression while maintaining the representational integrity of the network. As a result, computational cost and model size are considerably reduced without compromising expressive performance.

To promote consistent and balanced adaptation, we further introduce a *stabilization loss* designed to mitigate gradient imbalances between the full model and its sub-models. Combined with an *adaptive teacher mechanism* that dynamically guides each sub-model based on the full model's

predictions, this design ensures uniform performance across all sub-model configurations. It aligns with the core objective of slimmable networks, optimizing multiple model capacities simultaneously under domain shift. Our main contributions are summarized as follows:

- We propose a novel slimmable network method tailored for the source-free unsupervised domain adaptation (SFUDA) setting, enabling simultaneous adaptation of multiple sub-models without access to labeled data or the source domain.
- We propose a block-level network compression strategy that adjusts model capacity at the block level and streamlines the construction of multiple sub-models, thereby improving training efficiency and adaptability across diverse target environments.
- We develop a stabilization loss to mitigate gradient imbalances between the full model and its sub-models, fostering harmonized optimization across capacities.
- We incorporate an adaptive teacher mechanism that dynamically guides sub-model learning based on the full model's predictions, promoting robust performance under domain shift.
- We demonstrate the effectiveness of our approach through extensive experiments on multiple domain adaptation benchmarks and present detailed ablation studies to evaluate the contribution of each component.

II. RELATED WORKS

A. SOURCE-FREE UNSUPERVISED DOMAIN ADAPTATION

Unsupervised Domain Adaptation (UDA) aims to enhance model performance when a distribution shift exists between a labeled source domain and an unlabeled target domain [5], [6]. ADDA [5] employs a generative adversarial network to minimize this performance gap, whereas, WDAN [6] utilizes the weighted mean maximum discrepancy to incorporate prior probabilities from both domains.

Unlike conventional UDA, Source-Free Unsupervised Domain Adaptation (SFUDA) eliminates the requirement to access source data during adaptation [7], [8], [10]. For instance, SSM [10] minimizes the maximum mean discrepancy to achieve adaptation without source data, while SFOD-SED [8] applies a self-entropy descent strategy for SFUDA in object detection tasks. SHOT [7] enhances adaptation through a pseudo-labeling mechanism. SFUDA methods have been extended to various tasks [9], [11]; for example, G-SFDA [9] seek to preserve source domain performance after adaptation.

Recent advances in SFUDA primarily focus on refining pseudo-labels and improving loss functions. DCPL [37] mitigates pseudo-label noise, thereby improving accuracy across multiple SFUDA benchmarks. Similarly, ISFDA [38] revisits SHOT and enhances the adaptation process by reducing the divergence between its proposed loss terms.

Nevertheless, most SFUDA approaches overlook the practical challenge of adapting under hardware or computational constraints. In real-world applications, models often operate

on device with limited resources. Therefore, it is crucial to design architectures that adapt not only to the target domain but also to the hardware capacity. Addressing this issue is vital for achieving scalable and efficient domain adaptation in practice.

B. KNOWLEDGE DISTILLATION

Knowledge distillation (KD) is a well-established approach for model compression that transfers knowledge from a large pre-trained teacher model to a smaller student model [12], [13], [14]. Hinton et al. [12] initially introduced the concept of KD by training a student model to replicate the softened logits of the teacher network.

In RKD [14], the teacher's inter-sample relational structure is transferred to the student model. Although these methods enhance student model performance, they often require an additional training phase for the teacher model. This extra step reduces efficiency, particularly in lightweight settings or online learning scenarios [13].

To overcome this, KDCL [15] removes the dependency on a pre-trained teacher by employing multiple peer models that collaboratively share knowledge, an approach conceptually related to ensemble learning. Unlike traditional knowledge distillation frameworks, which require manual specification of the student architecture, our proposed framework automatically determines an optimized, compressed architecture suited to both the target domain and the computational environment.

C. SLIMMABLE NEURAL NETWORK

Slimmable neural networks have achieved remarkable success in improving computational efficiency by providing flexible neural architectures that enable users to select sub-models according to specific requirements [2], [16]. The OFA [17] introduces a one-for-all strategy that unifies the training and search processes to enhance the usability of slimmable networks. Similarly, DS-Net [4] employs a gating mechanism to generate diverse sub-models customized to user preferences. Recent developments have further advanced this field through methods such as contrastive self-supervised training for slimmable networks [16], [39] and knowledge distillation in manifold space to optimize sub-model training [36].

Despite these advancements, slimmable networks still face major challenges in source-free unsupervised domain adaptation (SFUDA). Most existing methods heavily rely on labeled validation data to optimize sub-models, which is impractical in SFUDA settings [16]. Moreover, the need for large validation datasets not only increases labeling costs but also introduces potential privacy concerns [3], [4], [17]. Another limitation lies in the exhaustive search procedures often required to evaluate thousands of sub-models (exceeding 10K in some cases), leading to substantial computational overhead [17]. Furthermore, these approaches typically scale channel widths or network depths uniformly,

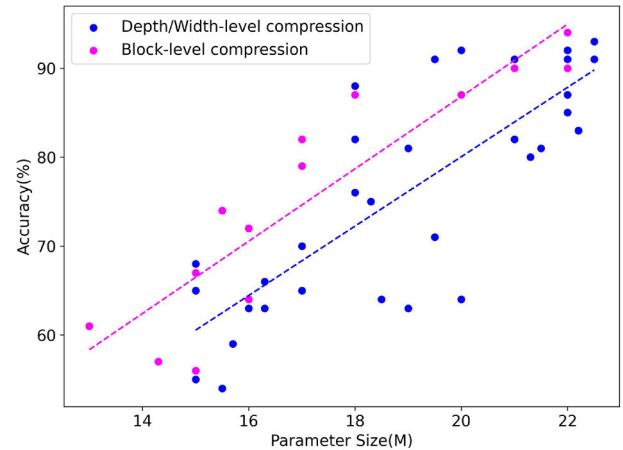


FIGURE 2. The Efficiency of Block-level Compression. From the perspective of model size-performance correlation, block-level compression achieves a higher correlation coefficient than depth- or width-based compression methods.

resulting in a vast search space that complicates optimization and further increases computational complexity.

To address these challenges, we propose a block-level network compression framework that refines the search process through fine-grained optimization. This design eliminates the need for labeled validation data and statistically reduces computational costs by avoiding exhaustive search. Additionally, our framework integrates online training, which enables efficient and systematic sub-model generation. These innovations collectively mitigate the limitations of existing slimmable neural network methods.

III. MOTIVATION

This research aims to address two key limitations observed in previous studies on slimmable networks. The first limitation is that many existing approach explore numerous compressed sub-models by varying the network width and depth [4], [16], [17]. As these configurations change, the slimmable network requires extensive computation to identify the optimal sub-model [17]. Such computational overhead is impractical for real-world adaptive scenarios, where models must be rapidly adjusted to diverse deployment environments.

The second limitation is that most existing slimmable methods rely on pre-trained teacher models optimized for a single architecture and dataset [4], [12], [17]. Consequently, when adapting to a new domain, the teacher model must be retrained before constructing the slimmable network, leading to considerable computational cost.

To overcome these challenges, we propose a block-transformable network and an adaptive teacher network. The block-transformable network minimizes the search space for optimal sub-models by constraining the number of architectural combinations. Meanwhile, the adaptive teacher network jointly updates both the teacher and sub-models, enabling more efficient and effective adaptation of the slimmable network to new domains.

TABLE 1. Controlled comparison of fixed-teacher and adaptive-teacher models under SFUDA domain shifts.

Method	A $\rightarrow$ D	A $\rightarrow$ W	D $\rightarrow$ A	D $\rightarrow$ W	W $\rightarrow$ A	W $\rightarrow$ D
Fixed-teacher	51.29	43.01	31.03	60.19	38.92	55.59
Adaptive-teacher	55.30	47.10	33.61	62.87	40.13	57.31

A. BLOCK-LEVEL NETWORK COMPRESSION

To mitigate the challenge of extensive search ranges, we employ block-level network compression, which reduces the number of potential architectural configurations. We assess its effectiveness by comparing the compression efficiency of block-level compression with that of conventional depth- and width-based methods. Specifically, we compute the correlation coefficient between the compressed model size and classification accuracy [18].

In our experiments, we randomly select sub-models generated via block-level compression and depth/width-level compression from ResNet-50 [19]. These sub-models are then evaluated on the DSLR domain of the Office-31 dataset [20], and results are presented in Fig. 2.

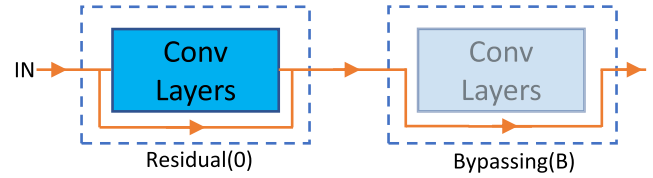
The experimental findings demonstrate that block-level compression achieves a higher correlation coefficient (0.90) compared to depth/width level compression (0.80). As shown in Fig. 2, block-level network compression enhances performance even at equivalent parameter sizes relative to traditional compression methods that vary depth and width [21]. Therefore, block-level network compression not only narrows the search range but also improves the compression efficiency while maintaining model accuracy. Building on this observation, we design a block-level slimmable network, described in detail in Sec. IV.

B. ADAPTIVE TEACHER THROUGH ONLINE TRAINING

To demonstrate the necessity of an adaptive teacher, we conduct a controlled adaptation experiment comparing the performance of fixed and adaptive teacher models. Specifically, two student models (ResNet-34 [19]) are trained using logits distilled from the *fixed* and *adaptive* teacher models (ResNet-50), respectively. In the *fixed* teacher model, the student training begins only after the teacher model has fully adapted to the target domain. In contrast, in the *adaptive* teacher setting, both teacher and student models are updated simultaneously, enabling them to adapt collaboratively in real time.

The SFUDA framework [7] is employed to adapt the teacher models to the target domain, while the student networks are trained using standard knowledge distillation [12]. However, the fixed-teacher approach overlooks the adaptation of sub-models, resulting in logits optimized solely for the entire network architecture.

For optimal performance, the fine-tuning process should consider various sub-model configurations within the slimmable network, rather than depending exclusively on a single complete model. We evaluate this motivation experiment on Office-31 [20] (Amazon, DSLR, and Webcam)

**FIGURE 3. Original Residual Block and Bypassing Block. ‘Conv Layers’ denote the convolution layers in the original residual block. The transformed forward path is represented by a path code, where the original residual block and the bypassing block are labeled as 0 and B, respectively.**

and report representative cross-domain shifts (e.g., Amazon(A) $\rightarrow$ DSLR(D) and Amazon(A) $\rightarrow$ Webcam(W)) using consistent arrow notation. In Table 1, the adaptive-teacher configuration consistently outperforms the baseline across all SFUDA tasks, confirming the effectiveness of joint teacher-student optimization under conditions of domain shifts.

IV. BLOCK TRANSFORMABLE NETWORK

Building on the motivations in Sections III-A and III-B, we introduce an *online adaptation mechanism* in which the teacher and student models are adapted simultaneously to unseen target domains through the proposed *block-level slimmable network*. Additionally, an online sub-model search strategy is developed to identify the optimal configuration during slimmable training, thereby improving adaptation efficiency.

To construct a slimmable network that integrates multiple models with varying compression ratios, we design a block transformable network (BTN). Transforming the BTN into a compressed sub-model is straightforward; it can be accomplished by removing specific residual blocks. We first design a BTN that supports flexible block removal without structural dependency, and then constrain the number of potential sub-models to reduce search costs while maintaining stable post-compression performance.

For efficient architectural transformation, we focus on neural networks with residual connections, including ResNet [19], ResNext [22], and WideResNet [23]. We adopt ResNet as the primary backbone due to its modular design, robust performance, and broad applicability in both domain adaptation and slimmable network research. Its residual connections align seamlessly with our block-level transformations, enabling effective compression and optimization while mitigating vanishing gradient issues. To ensure the generality, we also extend our experiments to WideResNet, demonstrating the architecture-agnostic property of the proposed method.

We define a residual block as a structural unit incorporating residual connections, composed of multiple convolution layers, activation functions, and batch normalization layers. Traditionally, the number and size of residual blocks vary depending on the network architecture. However, the proposed BTN framework supports compression irrespective

of architectural variations, ensuring flexibility and robustness during adaptation.

A. BYPASSING BLOCKS

In BTN, certain residual blocks are randomly replaced with bypassing blocks, to achieve block-wise network compression through structural transformation. The original residual block computes its output by adding the transformed feature, obtained through internal layers, to the input via a residual connection. In contrast, the bypassing block outputs the input without transformation, effectively eliminating the parameters associated with the corresponding residual block. This substitution leads to considerable model compression.

In deep neural networks, residual connections often have a bottleneck structure that reduces the input channel dimension to match the output dimension. In such cases, the bypassing block outputs the feature map produced by the bottleneck layer, which is fed by the input, rather than skipping the entire block. Fig. 3 illustrates the schematic difference between the original residual block and the bypassing block.

B. BLOCK TRANSFORMATION

Because a residual network typically consists of multiple residual blocks, varying the number and position of blocks replaced by bypassing blocks allows the generation of diverse sub-models. Consequently, a single BTN can embed numerous compressed sub-models, including the complete model without compression.

To uniquely identify each architecture, we define a path code in which each character represents the type of residual block in sequential order. In the representation, ‘0’ denotes the original residual block, and ‘B’ represents the bypassing block. For example, a path code of (0000) corresponds to the original uncompressed model. If bypassing blocks replace the second and fourth residual blocks, the path code becomes (0B0B).

Let S_k denote the set of all path codes in which k blocks are replaced with bypassing blocks. When K represents the total number of residual blocks, each path code in S_k contains $(K - k)$ ‘0’ symbols, and the number of k ‘B’ symbols. Thus, S_0 is a group of the path code where no block transformation is applied, consisting of only one path code: $S_0 = \{(0000)\}$. As a simple example, for a model with two residual blocks, $S_1 = \{B0\}, \{0B\}$.

V. ONLINE MULTI-PATH STABILIZATION

Using the proposed Block Transformable Network (BTN), we address the challenge of slimmable adaptation under source-free unsupervised domain adaptation (SFUDA). The learning process consists of two phases: a supervised pre-training phase, where the BTN learns discriminative features from labeled source data, and an unsupervised adaptation phase, where the model adapts to the target domain through online optimization. As illustrated in Fig. 4, we first pre-train the complete model on labeled source data and then perform online adaptation on unlabeled target data via multi-path

optimization over the complete model and sampled sub-models.

A. PRE-TRAINING PHASE

In the pre-training phase, only the complete model (denoted as f^{s_0} is employed without any bypassing blocks. The model is trained using labeled source data ($\mathbf{X}, \mathbf{Y}$) under a standard classification objective with label smoothing [25], [26]. The cross-entropy loss function is defined as:

$$\mathcal{L}_{ce} = \mathbb{E}_{\mathbf{x} \in \mathbf{X}, \mathbf{y} \in \mathbf{Y}} \left[-g(\mathbf{y}) \log(\sigma(f^{s_0}(\mathbf{x}))) \right], \quad (1)$$

where $\sigma(\cdot)$ denotes the softmax function, and $g(\mathbf{y})$ represents the smoothed version of the one-hot label $\mathbf{y}$ as:

$$g(\mathbf{y}) = (1 - \alpha)\mathbf{y} + \alpha \mathbf{1}_y / C. \quad (2)$$

Here, C indicates the total number of classes, and α is the label-smoothing coefficient, set to 0.1 following [25], [26]. This pre-training phase establishes a robust representation for the subsequent domain adaptation stage.

B. ADAPTATION PHASE

During adaptation, the BTN is fine-tuned using the unlabeled target data $\mathbf{X}_t$. The pre-trained model is decomposed into two components: a feature extractor and a classifier. Only the feature extractor is updated, based on the assumption that it can effectively generalize to target-domain representations.

In this phase, four complementary objectives jointly guide the adaptation process: (i) *entropy minimization*, (ii) *pseudo-supervised regularization*, (iii) *multi-path knowledge distillation*, and (iv) *gradient stabilization*. Each objective addresses a distinct aspect of adaptation, as described below.

(i) Entropy Minimization. We adopt the self-entropy loss [27], which encourages the model to produce more confident predictions for target samples by minimizing output uncertainty:

$$\begin{aligned} \mathcal{L}_{self} = \mathbb{E}_{\mathbf{x} \in \mathbf{X}_t} \left[-\sigma(f^{s_0}(\mathbf{x})) \log(\sigma(f^{s_0}(\mathbf{x}))) \right] \\ + \sigma \left(\mathbb{E}_{\mathbf{x} \in \mathbf{X}_t} [f^{s_0}(\mathbf{x})] \right) \log \left(\mathbb{E}_{\mathbf{x} \in \mathbf{X}_t} [f^{s_0}(\mathbf{x})] \right), \end{aligned} \quad (3)$$

The first term minimizes the prediction entropy for each target sample, compelling the model to make more confident (i.e., low-entropy) predictions. The second term enhances global diversity by maximizing the entropy of the averaged predictions, thereby preventing the model from collapsing into a single dominant class.

(ii) Pseudo-Supervised Regularization. To further stabilize adaptation, we employ synthetic pseudo-labels [7], generated by clustering the feature representations of target samples. Let $\hat{\mathbf{Y}}$ denote the pseudo-label set. The corresponding loss is formulated as:

$$\mathcal{L}_{syn} = \mathbb{E}_{\mathbf{x} \in \mathbf{X}_t, \hat{\mathbf{y}} \in \hat{\mathbf{Y}}} \left[-g(\hat{\mathbf{y}}) \log(\sigma(f^{s_0}(\mathbf{x}))) \right]. \quad (4)$$

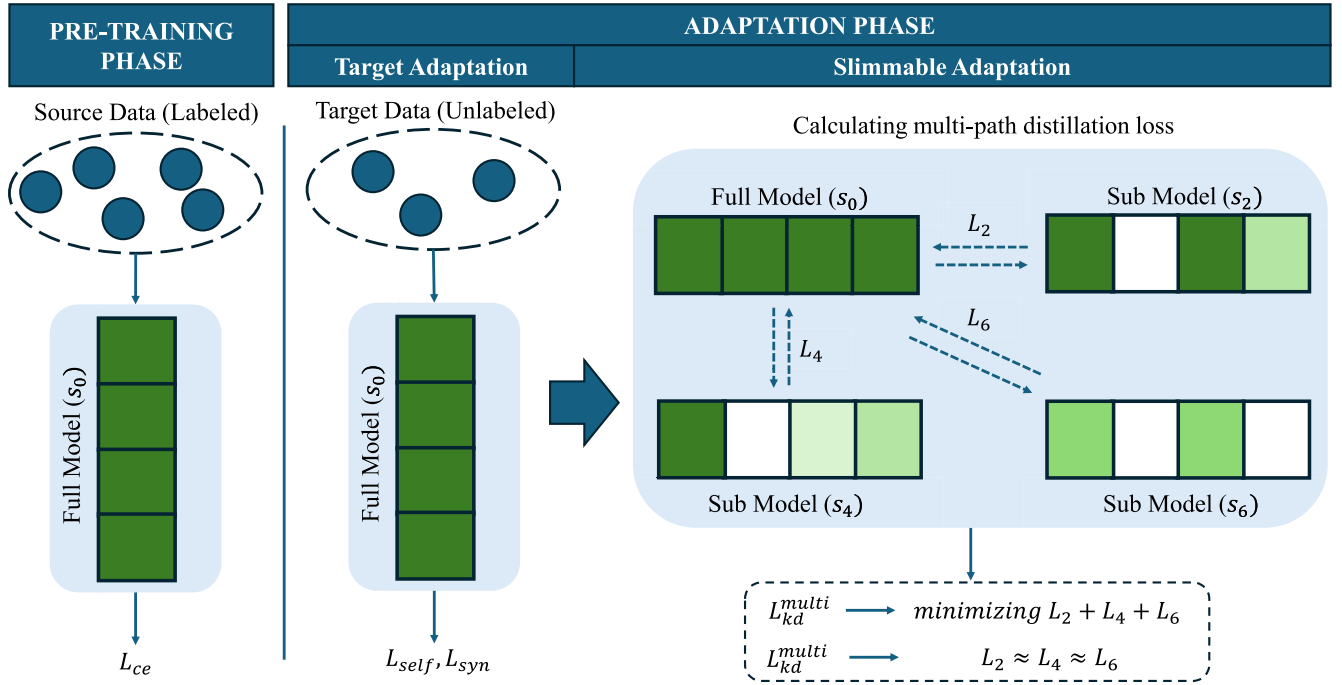


FIGURE 4. Overall framework of Training Block Transformable Network (BTN). The framework consists of two key stages: (1) the *pre-training phase*, where the whole model s_0 is trained on labeled source data using the classification loss $\mathcal{L}_{ce}$; and (2) the *adaptation phase*, where the model is fine-tuned on unlabeled target data through multi-path optimization. During adaptation, both the complete model and sub-models (s_2, s_4, s_6) are jointly optimized using four complementary loss functions: self-entropy ($\mathcal{L}_{self}$), synthetic label ($\mathcal{L}_{syn}$), multi-path distillation ($\mathcal{L}_{kd}^{multi}$), and stabilization losses ($\mathcal{L}_{stable}$). This unified training pipeline facilitates domain adaptation while ensuring efficient optimization of the slimmable sub-model.

Here, $g(\hat{y})$ applies label smoothing as defined in Eq. 2. This pseudo-supervised regularization term constrains model predictions through the clustered label structures, enhancing class separability within the target domain.

(iii) Multi-Path Knowledge Distillation. Because BTN comprises multiple sub-models that share parameters with the complete model, a multi-path knowledge distillation (KD) strategy is employed to align their predictions. The KL divergence between the complete model (s_0) and each sub-model (s) is computed as:

$$\mathcal{L}_{kd}^s = \mathbb{E}_{\mathbf{x} \in \mathbf{X}_t} \left[KL \left(\sigma(f^{s_0}(\mathbf{x})/\tau), \sigma(f^s(\mathbf{x})/\tau) \right) \right], \quad (5)$$

where τ is the temperature parameter that softens the predicted distributions. Rather than distilling knowledge from a single model at a time, we perform joint distillation across multiple paths as follows:

$$\mathcal{L}_{kd}^{multi} = \sum_{s \in \{s_2, s_4, s_6\}} \mathcal{L}_{kd}^s, \quad (6)$$

This approach ensures consistent learning across different sub-model configurations and encourages shared representational semantics among models of varying capacities.

(iv) Gradient Stabilization. Traditional slimmable methods [16], [36] often suffer from imbalanced optimization, where certain sub-models dominate the shared parameter updates due to uneven gradient magnitudes. Such an imbalance leads to unstable convergence and inconsistent

performance across sub-models. To formally describe this phenomenon, we define the gradient discrepancy between two sub-models (s and s') as:

$$\Delta_{s,s'} = \|\nabla_{\theta} L_{kd}^s - \nabla_{\theta} L_{kd}^{s'}\|. \quad (7)$$

Without stabilization, this discrepancy typically increases over training iterations due to accumulated imbalance:

$$\Delta_{s,s'}(t+1) = \Delta_{s,s'}(t) + |\nabla_{\theta} L_{kd}^s - \nabla_{\theta} L_{kd}^{s'}|_{\text{extra}}, \quad (8)$$

where $|\nabla_{\theta} L_{kd}^s - \nabla_{\theta} L_{kd}^{s'}|_{\text{extra}} > 0$ indicates the additional imbalance term caused by dominant paths.

To tackle this effect, we introduce the stabilization loss, which penalizes large deviations among the KD losses of sub-models:

$$L_{\text{stable}} = \mathbb{E}_{s \in S} \left[\|L_{kd}^s - \mathbb{E}_{s' \in S} [L_{kd}^{s'}]\|_H \right], \quad (9)$$

where $\|\cdot\|_H$ denotes the Huber norm. Minimizing L_{stable} reduces gradient variance across sub-models, leading to the following convergence behavior:

$$\Delta_{s,s'}(t+1) \leq \Delta_{s,s'}(t). \quad (10)$$

This theoretically grounded mechanism constrains excessive gradient deviations, ensuring that no sub-model disproportionately influences shared parameters. Consequently, the training process becomes more stable and balanced, enabling all sub-models to achieve consistent adaptation performance under varying capacities.

Total Objective. The overall objective for the adaptation phase integrates the four complementary learning signals described above: entropy minimization ($\mathcal{L}_{self}$), pseudo-supervised regularization ($\mathcal{L}_{syn}$), multi-path knowledge distillation ($\mathcal{L}_{kd}^{multi}$), and gradient stabilization ($\mathcal{L}_{stable}$). The complete optimization objective is expressed as:

$$\mathcal{L}_{total} = \mathcal{L}_{self} + \lambda_1 \mathcal{L}_{syn} + \lambda_2 \mathcal{L}_{kd}^{multi} + \lambda_3 \mathcal{L}_{stable}, \quad (11)$$

where λ_1 , λ_2 , and λ_3 are hyper-parameters controlling the relative influence of each component.

Entropy minimization drives the model to produce confident predictions on the target domain, whereas pseudo-supervised regularization anchors these predictions using synthetic labels derived from feature clustering. The multi-path distillation term enforces consistency between the complete model and its sub-models, promoting coherent knowledge transfer across capacities. Finally, the stabilization loss constrains gradient variance among sub-models, ensuring balanced optimization and preventing domination by any specific path.

Together, these components form a unified learning objective that harmonizes self-supervised target adaptation, knowledge transfer, and structural regularization within a single optimization framework. This synergy enables the BTN to achieve stable and consistent adaptation across all sub-model configurations without requiring access to source data.

C. PATH CODE REDUCTION

During the adaptation phase, the objective is to identify sub-models that effectively represent the target domain while remaining consistent with the whole model. To achieve this, the number of path codes is systematically reduced at fixed intervals (every 30 epochs), prioritizing sub-models that demonstrate optimal performance. The evaluation of path codes is based on the average Kullback-Leibler (KL) divergence between the probabilistic outputs of the complete model (f^{s_0}) and the corresponding sub-model (f^s).

A lower KL divergence indicates greater similarity between the outputs of the sub-model and the full model, emphasizing the sub-model's ability to preserve consistent performance during adaptation. At each reduction step, the average KL divergence (Eq. 5) is computed for all path codes in the current set ($\mathbf{S}_k^t$). Path codes exhibiting the smallest average KL divergence are retained, thereby reducing the set size by one-third. This iterative refinement continues until only one path code remains ($\mathbf{S}_k^T$), which is considered the optimal representation of the target domain.

D. SUMMARY OF THE ADAPTATION PHASE

As illustrated in Alg. 1, the adaptation phase begins with a pre-trained complete model f^{s_0} and a set of path code groups $\mathbf{S}_k = \{S_0, S_2, S_4, S_6\}$. Let $\mathbf{S}_k^t$ denote the set of candidate path codes for group k at epoch t (with $\mathbf{S}_k^0 = \mathbf{S}_k$ initialized from $\mathbf{S}_k$). For periodic path code reduction, we score each candidate using the epoch-averaged KL divergence (averaged

Algorithm 1 Pseudo-Code for the Adaptation Phase

Require: Pre-trained model f^{s_0} , path code groups $\mathbf{S}_k^0$.
Ensure: Single slimmable network f^{s_0} and optimized path codes for each of the path code groups $\mathbf{S}_k$

- 1: **for** each training epoch $t = 1, 2, \dots, T$ **do**
- 2: **Sub-model Sampling:** Randomly sample N path codes $\{s_2, s_4, s_6\}$ from $\mathbf{S}_k^t$
- 3: **Compute Losses:**
- 4: 1. Self-entropy loss: $\mathcal{L}_{self}$ (Eq. 3)
- 5: 2. Synthetic label loss: $\mathcal{L}_{syn}$ (Eq. 4)
- 6: 3. Multi-path guided distillation loss: $\mathcal{L}_{kd}^{multi}$ (Eq. 6)
- 7: 4. Stabilization loss: $\mathcal{L}_{stable}$ (Eq. 9)
- 8: Update f^{s_0} from $\mathcal{L}_{total}$ (Eq. 11)
- 9: **if** t is a multiple of 30 **then**
- 10: **Path Code Reduction:**
- 11: 1. Compute $\mathcal{L}_{kd}^s$ for each path code s in $\mathbf{S}_k^t$ (Eq. 5)
- 12: 2. Select the top $\frac{|\mathbf{S}_k^t|}{3}$ path codes with the smallest $\mathcal{L}_{kd}^s$ values
- 13: 3. Form the reduced path code group $\mathbf{S}_k^{t+1}$ with the selected path codes
- 14: **end if**
- 15: **end for**

over target mini-batches within the current epoch) and retain the top candidates accordingly.

During each training epoch, sub-models corresponding to S_2 , S_4 , and S_6 are sampled, and their respective path codes are refined using a combination of self-entropy, synthetic label, multi-path distillation, and stabilization losses. Periodic path code reduction gradually narrows the search space, thereby enhancing training efficiency.

The final output of the adaptation phase is a slimmable network, in which the whole model f^{s_0} corresponds to s_0 . Optimized path codes for S_2 , S_4 , and S_6 are derived, enabling the deployment of specific sub-model configurations adapted to diverse computational environments and hardware constraints. Once these optimized path codes are obtained, the compressed models can be deployed directly without requiring additional training, ensuring efficient adaptation while maintaining the whole model's performance. This process guarantees that the slimmable neural network retains the predictive accuracy of the full model while supporting efficient, domain-robust deployment of compressed sub-models across varying computational and hardware conditions.¹

VI. EXPERIMENTS

We evaluate the proposed online method across multiple source-free unsupervised domain adaptation (SFUDA) scenarios. Our approach is compared with state-of-the-art UDA and SFUDA methods. In addition, comparative experiments involving network compression and slimmable models are conducted under the same settings. It is important to note that, unlike previous compression and slimming methods that rely

¹ Code is available at <https://github.com/seosm52525252/BTN>

TABLE 2. Experimental results of Slimmable adaptation on Office-31.

Method	slimmable	source	$A \rightarrow D$	$A \rightarrow W$	$D \rightarrow A$	$D \rightarrow W$	$W \rightarrow A$	$W \rightarrow D$	avg
CR	-	-	100%	100%	100%	100%	100%	100%	100%
CDAN	-	O	92.9	94.1	69.3	98.6	71.1	94.1	86.7
DMRL	-	O	93.4	90.4	73.0	99.1	71.2	99.1	87.7
SHOT	-	-	94.0	90.1	74.7	98.4	74.3	99.9	88.6
DCPL	-	-	93.3	94.2	73.4	99.0	73.5	99.9	88.8
OFA	O	-	29.3	36.2	45.4	24.0	51.5	14.7	33.5
SlimDA	O	O	99.8	90.7	73.7	91.2	73.7	99.1	87.6
SlimCLR	O	-	90.0	85.3	70.3	96.1	65.3	98.8	84.3
Ours-S_0	O	-	95.8	93.0	70.1	98.1	71.2	99.8	<u>88.0</u>
CR	-	-	90%	90%	90%	90%	90%	90%	90%
CDAN	-	O	89.9	90.0	62.0	88.8	65.1	90.2	81.0
DMRL	-	O	92.1	89.5	65.0	88.0	66.3	94.3	82.5
SHOT	-	-	93.5	90.3	71.1	97.3	71.3	99.6	<u>87.1</u>
DCPL	-	-	90.8	91.0	69.3	90.0	68.8	98.9	84.8
E-PRUNE	-	-	84.5	90.4	75.4	90.9	76.5	88.2	84.3
KD	-	-	81.7	91.5	80.2	92.3	81.8	82.3	84.9
OFA	O	-	34.1	34.8	50.4	19.9	53.7	14.5	34.6
SlimDA	O	O	98.6	87.3	68.8	91.0	70.1	98.9	85.7
SlimCLR	O	-	88.2	82.5	63.1	91.3	60.5	96.0	80.3
Ours-S_2	O	-	95.8	93.0	70.2	98.1	71.1	99.8	88.0
CR	-	-	84%	80%	80%	84%	84%	84%	82%
CDAN	-	O	86.1	83.4	53.1	88.3	63.2	89.5	77.2
DMRL	-	O	87.0	84.5	62.8	80.3	61.0	90.8	79.4
SHOT	-	-	92.3	85.8	64.5	97.2	64.2	98.5	<u>83.7</u>
DCPL	-	-	90.2	82.8	63.8	82.1	66.8	96.3	80.3
E-PRUNE	-	-	77.5	78.5	70.6	80.6	70.8	82.7	76.8
KD	-	-	80.9	89.9	85.5	90.5	74.8	81.1	<u>83.7</u>
OFA	O	-	32.9	35.6	50.9	19.4	53.7	14.1	34.4
SlimDA	O	O	90.3	80.1	61.0	86.3	65.0	96.2	79.8
SlimCLR	O	-	78.9	75.5	58.2	82.1	52.2	95.3	73.7
Ours-S_4	O	-	95.6	93.1	70.8	98.1	71.1	99.8	88.0
CR	-	-	60%	60%	75%	75%	75%	75%	70%
CDAN	-	O	58.2	60.3	50.8	65.2	37.5	63.2	55.8
DMRL	-	O	60.3	57.1	43.2	58.2	32.6	66.1	52.9
SHOT	-	-	57.6	62.5	45.5	71.5	47.5	76.1	60.1
DCPL	-	-	57.8	60.8	43.0	73.1	46.3	75.3	59.3
E-PRUNE	-	-	68.5	75.7	67.6	79.0	67.6	67.5	71.0
KD	-	-	81.5	89.4	81.3	91.5	76.2	81.3	<u>83.5</u>
OFA	O	-	31.7	36.0	50.6	18.9	53.8	14.3	34.2
SlimDA	O	O	82.3	72.1	55.3	68.1	63.2	92.3	72.2
SlimCLR	O	-	73.0	56.2	51.0	62.5	43.1	82.4	61.3
Ours-S_6	O	-	95.6	93.1	70.0	98.0	71.0	99.8	87.9

on offline training schemes using labeled data, our method adopts an online training strategy, eliminating the need for source labels.

A. EXPERIMENTAL SETUP

1) DATASETS

We validate the proposed method on Office-31 [20] and Office-Home [30] datasets. The Office-31 dataset is a standard UDA benchmark dataset comprising three domains:

Amazon (A), DSLR (D), and Webcam (W), each containing 31 object categories. The Office-Home dataset, a more challenging benchmark, includes four domains: Art (A), Clip-Art (C), Product Image (P), and Real-World Image (R). All domains have 65 object categories in total.

2) NETWORK ARCHITECTURE

Our framework is built upon the residual network (ResNet) [19], forming a block transformable network (BTN).

TABLE 3. Experimental results of Slimmable adaptation on Office-Home.

Method	slimmable	source	$A \rightarrow C$	$A \rightarrow P$	$A \rightarrow R$	$C \rightarrow A$	$C \rightarrow P$	$C \rightarrow R$	$P \rightarrow A$	$P \rightarrow C$	$P \rightarrow R$	$R \rightarrow A$	$R \rightarrow C$	$R \rightarrow P$	avg
CR	-	-	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
SHOT	-	-	57.1	78.1	81.5	68.0	78.2	78.1	67.4	54.9	82.2	73.3	58.8	84.3	71.8
OFA	O	-	40.5	58.3	43.2	29.1	60.0	41.1	23.5	39.1	41.7	29.1	41.6	58.3	42.1
SlimDA	O	O	52.8	72.3	77.2	63.5	72.3	73.3	64.8	52.2	79.7	72.4	57.8	82.8	68.4
SlimCLR	O	-	56.2	74.8	75.1	60.9	76.4	65.8	66.1	54.8	75.5	68.2	51.2	78.6	66.9
Ours-S₀	O	-	54.7	78.2	79.8	66.2	76.9	75.7	66.7	52.9	80.3	72.3	57.7	84.2	70.5
CR	-	-	90%	90%	90%	90%	90%	90%	94%	90%	90%	90%	90%	90%	90%
SHOT	-	-	52.6	71.1	80.2	66.1	68.2	62.8	63.6	51.0	78.2	70.9	56.1	77.9	66.5
E-PRUNE	-	-	61.6	70.7	60.3	51.7	69.8	60.0	57.4	62.5	61.9	52.2	62.0	70.8	61.8
OFA	O	-	40.3	58.1	43.6	29.3	59.9	40.9	23.7	38.9	41.8	29.3	41.4	58.4	42.1
SlimDA	O	O	52.8	70.1	72.6	56.9	68.4	74.1	61.0	41.7	78.6	71.9	54.0	79.1	65.1
SlimCLR	O	-	51.3	70.7	71.2	51.2	71.8	57.0	59.1	49.6	72.0	63.1	41.4	66.1	60.3
Ours-S₂	O	-	54.7	78.1	79.8	66.3	76.9	75.8	66.7	53.2	80.5	72.3	57.5	83.8	70.5
CR	-	-	84%	84%	84%	84%	85%	80%	84%	84%	66%	80%	80%	80%	81%
SHOT	-	-	42.9	63.6	72.6	56.2	59.3	58.2	46.4	42.8	71.9	67.1	47.2	70.0	58.1
E-PRUNE	-	-	57.7	65.6	55.8	44.0	68.1	49.9	44.4	59.2	41.8	40.5	55.3	63.3	53.8
OFA	O	-	40.6	58.1	43.4	29.0	59.9	40.6	23.8	39.0	41.9	29.2	41.8	58.6	42.1
SlimDA	O	O	51.0	68.9	63.7	47.1	63.9	65.1	51.6	43.6	72.8	66.8	46.2	64.9	58.8
SlimCLR	O	-	50.5	61.7	58.1	48.6	66.4	54.1	52.1	39.1	68.0	59.7	38.4	58.3	54.5
Ours-S₄	O	-	54.8	78.9	79.5	65.2	77.1	76.2	66.8	53.2	80.5	72.0	58.0	83.9	70.5
CR	-	-	75%	75%	75%	78%	75%	75%	78%	78%	60%	60%	75%	75%	73%
SHOT	-	-	38.2	46.1	60.2	32.9	47.1	43.3	32.3	40.8	56.4	62.3	36.1	56.3	45.9
E-PRUNE	-	-	48.9	59.1	43.8	38.0	59.5	46.2	37.7	52.2	41.4	36.6	50.1	60.2	47.9
OFA	O	-	40.7	58.5	43.6	29.1	59.8	40.8	23.7	39.8	42.2	28.9	41.8	58.5	42.2
SlimDA	O	O	46.3	62.8	60.1	36.5	58.1	58.8	47.3	42.6	65.0	63.6	41.6	59.8	53.5
SlimCLR	O	-	46.5	58.7	52.7	42.1	59.7	51.3	42.8	39.3	64.6	49.7	38.7	56.7	50.2
Ours-S₆	O	-	54.1	77.7	78.4	65.0	75.6	74.1	64.7	52.0	80.0	67.8	56.2	82.4	69.0

A pre-trained ResNet-50 model is employed as the feature extractor. To generate pseudo-labels, we insert an embedding layer between the feature extractor and the classifier. As the model extension, we further demonstrate the adaptability of our method on Wide-ResNet [23].

B. IMPLEMENTATION DETAILS

1) HYPER-PARAMETERS

We employ a stochastic gradient descent (SGD) optimizer with momentum 0.9 and a learning rate of 0.01 [31]. The hyper-parameter λ_1 is set to 1 in all experiments, and λ_2 for $\mathcal{L}_{kd}^{multi}$ is set to 1 in Office-31 and 0.5 in Office-home, respectively. The hyper-parameter λ_3 is set to 0.2 in Office-31 and 0.1 in Office-home, respectively. Temperature τ is set to 4. We use 64 inputs for the single batch and our method is implemented on a single GPU (A6000).

2) PATH CODE SETUP

Path code groups (S_0 , S_2 , S_4 , and S_6), are defined based on ResNet-50, which consists of 16 residual blocks. Excluding the initial n blocks, the remaining residual blocks are divided into 5 subsets. A probability list, $[p_1, p_2, p_3, p_4, p_5]$, is constructed such that the sum of all elements equals 1, where each element represents the probability of converting the corresponding subset block into a bypassing block. Instead of utilizing all possible path code from each groups, we randomly sample 27 path codes per path code group based on the probability list during initialization. In our experiments, $n = 3$ and the probability list for BTN is defined as $[0.1, 0.2, 0.4, 0.2, 0.1]$.

3) BASELINES

In the absence of prior studies integrating slimmable networks into the SFUDA setting, we conduct extensive

baseline experiments. Starting from a teacher model adapted to the unlabeled target domain using SHOT method [7], we perform an additional compression step following the protocol of each existing compression approach. To investigate more aggressive compression, we include supervised compression techniques and slimmable methods such as E-PRUNE [34], Knowledge Distillation (KD) [12], and Once-For-All (OFA) [17], that require label supervision. For this method, we use 30% of the target annotations. Since these approaches are not designed for slimmable adaptation, we mark the ‘slimmable’ column as ‘-’ in Tables 2 and 3. For comparison with slimmable networks, we evaluate OFA [17], SlimDA [16], and SlimCLR [39]. To assess adaptation performance, we also include CDAN [32] and DMRL [33] under the standard UDA settings, which require access to source data as indicated in the ‘source’ column of Tables 2 and 3. For SFUDA evaluation, we perform slimmable adaptation using DCPL [37] and SHOT.

C. QUANTITATIVE COMPARISON

At the conclusion of the proposed framework, we obtain three optimized sub-models from S_2 , S_4 , and S_6 , each exhibiting a distinct compression ratio (CR) depending on the target domain. For fair comparisons, we evaluate all competing methods at matched CR settings [35], and report results on Office-31 and Office-Home in Tables 2 and 3.

1) COMPARISON WITH MODEL-SPECIFIC COMPRESSION BASELINES (E-PRUNE, KD)

Compared to model-specific compression methods that produce a single task-specific compressed model, BTN consistently yields stronger sub-models under the same CR while maintaining compactness. On Office-31 (Table 2), under the most compressed setting reported (CR 60% and 70%),

our smallest model S_6 achieves an average accuracy of 87.9, outperforming E-PRUNE (71.0) and KD (83.5) by 16.9 and 4.4 points, respectively. This advantage persists at milder compression: at CR 82%, S_4 reaches 88.0 on average, exceeding E-PRUNE (76.8) and KD (83.7) by 11.2 and 4.3 points. On Office-Home, which includes larger domain gaps, the margin becomes more pronounced: at CR 60% and 75%, S_6 achieves 69.0 average accuracy, surpassing E-PRUNE (47.9) by 21.1 points. Overall, these results confirm that BTN's block-level compression and adaptive distillation produce sub-models that remain accurate even when aggressively compressed, whereas model-specific baselines degrade substantially as domain transfer.

2) COMPARISON WITH SLIMMABLE BASELINES (OFA, SlimDA, SlimCLR)

We next compare BTN against prior slimmable learning approaches adapted to SFUDA. On Office-31, at CR 60% and 75%, S_6 achieves substantially higher performance (87.9) than OFA (34.2), SlimDA (72.2), and SlimCLR (61.3). Even at higher capacities, BTN maintains a clear lead: at CR 82%, S_4 reaches 88.0, while SOTA slimmable method (SlimDA) reports 79.8 (gains of 8.2 points). On Office-Home, under average CR 73% S_6 , BTN outperforms SlimDA (53.5) and SlimCLR (50.2) by 15.5 and 18.8 points. These improvements align with the motivation of our design: in SFUDA, fixed or one-way guidance from a single full model is insufficient for maintaining consistent sub-model quality across capacities, while BTN's co-adaptation keeps sub-models competitive without relying on an additional search step.

3) COMPARISON WITH SFUDA/UDA METHODS (SHOT, DCPL)

Finally, we position BTN against standard SFUDA/UDA baselines that assume a single fixed architecture. On Office-31, BTN's full model (S_0) achieves 88.0 average accuracy, exceeding CDAN (86.7) and DMRL (87.7). More importantly for the slimmable setting, BTN's compressed sub-models remain comparable to (and often stronger than) these fixed-architecture baselines: At an average CR of 70% (S_6), BTN outperforms the SOTA SFUDA baselines (SHOT and DCPL) by a large margin (27.8 points). On Office-Home, compared with SHOT, BTN demonstrates strong robustness under more challenging domain discrepancies. Moreover, under strong compression (S_6 setting), our approach still outperforms SHOT (45.9) by 13.1 points.

In summary, Tables 2 and 3 show that BTN provides (i) consistently stronger sub-models than model-specific compression, (ii) substantially more reliable slimmable adaptation than prior slimmable baselines, and (iii) competitive (often superior) performance relative to fixed-architecture SFUDA methods, while offering multiple deployable sub-models from a single training process.

TABLE 4. Compression results using a large-scale self-supervised backbone.

Method	CR = 100%		CR = 90%		CR = 80%		CR = 60%	
	A→D	A→W	A→D	A→W	A→D	A→W	A→D	A→W
SHOT	96.8	94.8	94.7	87.2	92.0	88.1	62.5	60.7
SlimDA	99.8	93.0	98.1	87.6	88.6	85.0	76.3	68.5
Ours	96.4	94.3	96.4	94.5	96.4	91.4	96.0	90.2

D. COMPUTATIONAL COST

The proposed algorithm requires approximately 155 minutes to complete the target adaptation process on the Office-31 dataset, whereas OFA takes about 1,143 minutes to complete the same sequence. This corresponds to a runtime reduction of approximately 86.4%, i.e., a $7.4\times$ speedup over the existing slimmable baseline (OFA).

Unlike OFA, our method eliminates the need for independent teacher training or multiple training phases for slimmable networks. Experimental results clearly demonstrate that our method is markedly more efficient than existing slimmable network frameworks, such as OFA. Furthermore, the optimal path is determined during training, eliminating the need for additional computational resources to identify the best-performing model. Therefore, we conclude that our method efficiently performs the SFUDA task with substantially reduced computational overhead.

E. SCALABILITY UNDER A LARGE-SCALE SELF-SUPERVISED BACKBONE

In Table 4, we evaluate scalability of our method using a ResNet-50 backbone pre-trained with DINO on ImageNet [40]. In Office-31 dataset, the source model is trained on *Amazon* with labels and adapted to the unlabeled *DSL*R and *Webcam* targets without access to source data (SFUDA). Our method maintains high accuracy across all CRs and remains stable despite reduced model capacity. For example, we observe an over-20-point accuracy improvement at $CR = 60\%$. These findings demonstrate that block-level slimmable adaptation with multi-path optimization is effective when applied to large-scale pre-trained backbone in the SFUDA scenarios.

F. ADDITIONAL EXPERIMENTS ON THE CHALLENGING BENCHMARK

We further evaluate the proposed framework using the VisDA-C dataset [41], a challenging benchmark characterized by substantial appearance discrepancies between synthetic and real domains. Following the SFUDA protocol, the ResNet-101 backbone is first trained on the synthetic source domain and subsequently adapted to the real target domain without access to labeled data.

As illustrated in Fig. 5, the proposed method exhibits strong robustness under varying compression ratios. While SHOT achieves 80.2 accuracy at full capacity (S_0 , $CR = 100\%$), its performance sharply declines to 51.5 accuracy at the most compressed configuration (S_{12} , $CR = 52\%$).

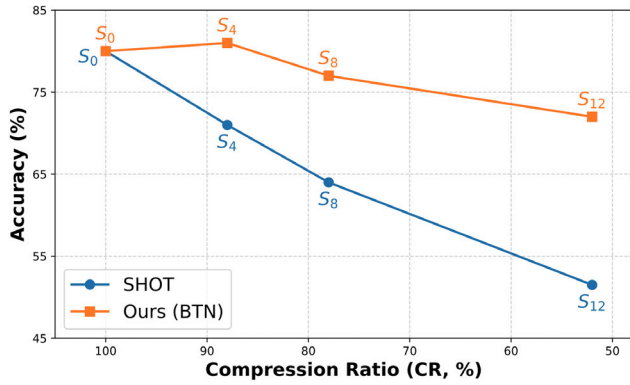


FIGURE 5. Slimmable Adaptation on VisDA-C Benchmark.

In contrast, our approach maintains stable performance, showing 79.6 accuracy at full capacity and retaining 72.0 accuracy even at 52% compression. Intermediate configurations also follow this trend, with our method consistently outperforming SHOT by considerable margins at S_4 ($CR = 88\%$) and S_8 ($CR = 78\%$). Overall, these results clearly indicate that the proposed BTN effectively preserves performance under high compression, achieving both scalability and robust SFUDA adaptation on a highly challenging benchmark.

VII. ANALYSIS

In this section, we present a comprehensive validation and analysis of the proposed approach. We evaluate the stability of the proposed online mechanism, and include the results of the ablation experiments. Additionally, we provide extended evaluations to demonstrate the model's scalability and adaptability. This in-depth investigation emphasizes the robustness and flexibility of the proposed method.

A. SUB-MODEL STABILITY DURING SLIMMABLE ADAPTATION

To validate the effectiveness of the proposed approach, we analyze sub-model performance during slimmable adaptation as shown in Fig. 6b. For comparison, we also describe the *baseline* (SHOT) as shown in Fig. 6a. Our method shows that the sub-models come closer to the performance of the full model when training proceeds as shown in Fig. 6b. In contrast, we show that the performance of the sub-model hardly changes after 700 iterations as shown in Fig. 6a. Furthermore, by using our method, the performance of each sub-model is higher than the performance of the entire model of *baseline*.

To optimize sub-model selection, we use the average of KL-divergence to reduce the path codes, thereby generating optimized sub-models for each target domain. These sub-models exhibit stable training behavior, as depicted in Fig. 6b. However, in the *baseline* (SHOT), identifying which sub-model possesses the greatest target-adaptive potential is challenging as shown in Fig. 6a. This limitation arises from

TABLE 5. Ablation results.

Method	$A \rightarrow D$	$A \rightarrow W$	$D \rightarrow A$	$D \rightarrow W$	$W \rightarrow A$	$W \rightarrow D$	avg
<i>baseline</i> (S_0)	95.0	92.0	73.6	98.2	72.8	99.6	88.5
<i>w.o PR</i> (S_0)	95.0	91.7	67.5	98.1	70.1	99.6	87.0
<i>w.o L_{stable}</i> (S_0)	94.7	90.4	68.5	97.9	69.0	99.8	86.7
<i>Ours-full</i> (S_0)	95.8	93.0	70.1	98.1	71.2	99.8	88.0
<i>baseline</i> (S_2)	94.5	91.1	73.2	98.2	72.4	99.6	88.1
<i>w.o PR</i> (S_2)	95.3	89.3	66.1	98.1	68.4	99.8	86.2
<i>w.o L_{stable}</i> (S_2)	94.7	90.4	68.5	97.9	69.0	99.8	86.7
<i>Ours-full</i> (S_2)	95.8	93.0	70.2	98.1	71.1	99.8	88.0
<i>baseline</i> (S_4)	92.3	85.8	64.5	97.2	64.2	98.5	83.7
<i>w.o PR</i> (S_4)	87.8	83.0	53.6	93.0	61.6	98.0	79.5
<i>w.o L_{stable}</i> (S_4)	94.7	90.4	68.1	97.5	69.0	99.8	86.5
<i>Ours-full</i> (S_4)	95.6	93.1	70.0	98.1	71.1	99.8	88.0
<i>baseline</i> (S_6)	57.6	62.5	45.5	71.5	47.5	76.1	60.1
<i>w.o PR</i> (S_6)	84.7	81.5	48.3	90.2	60.9	98.0	77.3
<i>w.o L_{stable}</i> (S_6)	94.7	90.5	68.1	97.5	68.1	99.8	86.4
<i>Ours-full</i> (S_6)	95.6	93.1	70.0	98.0	71.0	99.8	87.9

the absence of a systematic mechanism to evaluate each sub-model's adaptability to the target domain. In contrast, the proposed online searching scheme, effectively enhances the adaptability of multiple sub-models and isolates those with the highest potential. Overall, the proposed method achieves substantial performance gains and demonstrates enhanced stability during slimmable adaptation.

In addition, we report the training losses of the optimized sub-models during slimmable adaptation in Fig. 7. All sub-models (S_2 , S_4 , and S_6) show a consistent decrease in loss and converge to low values as training proceeds, indicating stable optimization across different capacities. Notably, while the losses exhibit larger fluctuations in early iterations, they become stable after approximately 700 iterations, which aligns with the stabilization trend observed in the performance curves.

B. ABLATION STUDY

We conduct a comprehensive series of ablation experiments, as summarized in the Table 5.

1) IMPACT OF STABILIZATION LOSS

One ablation test evaluates the effectiveness of the stabilization loss (Eq. 9), denoted as *w.o L_{stable}* . It is important to emphasize that achieving high performance for a full model in the target domain does not guarantee that the sub-models will perform well, as indicated by the *baseline* results. However, the ablation findings reveal that applying Eq. 9 to minimize divergences among sub-models notably enhances their performance. Specifically, in the highly compressed S_6 configuration, the inclusion of stabilization loss boosts accuracy by 1.9 points (from 68.1 to 70.0) in the challenging $D \rightarrow A$ scenario and by 2.9 points (from 68.1 to 71.0) in the $W \rightarrow A$ scenario. On average, the proposed method achieves a 1.5-point performance gain over the model without stabilization loss across all sub-model capacities. This enhancement increases the generalization capability of the slimmable network in the target domain, thereby facilitating deployment across diverse environments. Furthermore, this mechanism strengthens the overall stability and robustness of the network.

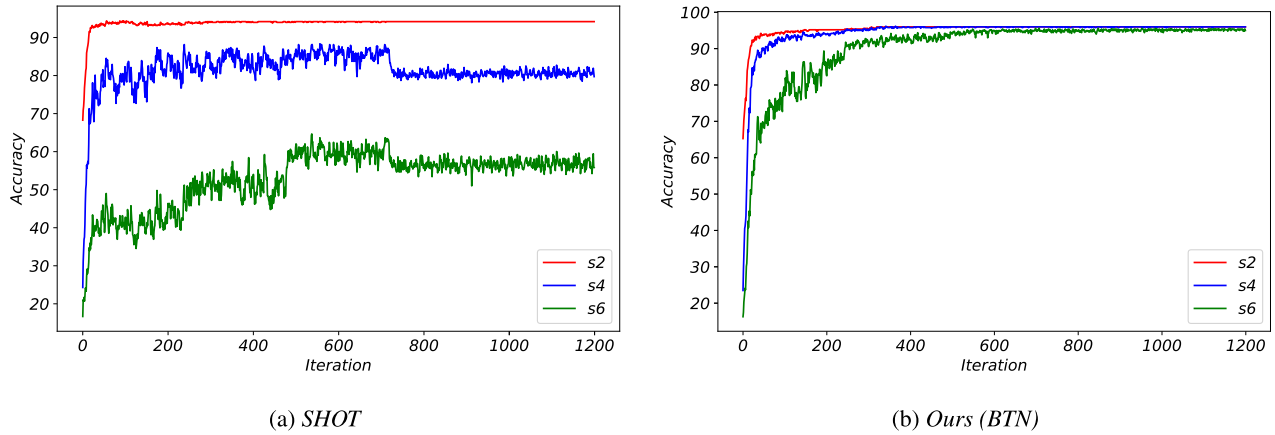


FIGURE 6. Performance Progression of Sub-models during Adaptation. This figure illustrates how the performance of sub-models evolves throughout the adaptation phase. Whereas the existing SFUDA method (SHOT) guarantees performance improvement only for a single model (s2), the other sub-models show limited enhancement. In contrast, the proposed method achieves consistent and progressive performance gains across all sub-models as adaptation advances.

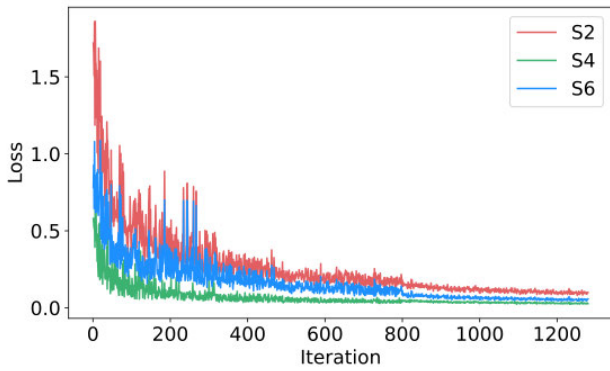


FIGURE 7. Training Losses of Sub-Models. The results show that all sub-models stably converge to low overall loss, with reduced loss dispersion across sub-models, demonstrating consistent optimization during training.

2) IMPACT OF PATH CODE REDUCTION

To analyze the contribution of the proposed Path Code Reduction (PR), we compare the performance of models with and without PR (*w.o PR*) across various sub-model capacities (S_0 – S_6) in Table 5. Our full framework consistently outperforms the baseline while removing PR leads to significant performance degradation. For instance, at S_4 (82% average capacity), the proposed method achieves 88.0, showing an 8.5-point improvement over the *w.o PR* (79.5). Similarly, at S_6 (70% average capacity), our method records 87.9, outperforming the baseline (60.1) by a substantial margin of 27.8, whereas removing PR limits performance margin by 17.2 points.

These results indicate that PR effectively eliminates redundant path codes and preserves the most informative sub-models during adaptation, thereby enhancing robustness under constrained capacity. The stable performance observed across all scales further validates that PR is essential for

TABLE 6. Compression result on Wide-ResNet.

Method	$A \rightarrow D$	$A \rightarrow W$	$D \rightarrow A$	$D \rightarrow W$	$W \rightarrow A$	$W \rightarrow D$	avg
ResNet (S_0)	95.8	93.0	70.1	98.1	71.2	99.8	88.0
WideNet (S_0)	92.2	89.2	74.1	98.1	74.7	99.6	88.0
ResNet (S_2)	95.8	93.0	70.2	98.1	71.1	99.8	88.0
WideNet (S_2)	92.2	89.2	74.2	98.1	74.7	99.6	88.0
ResNet (S_4)	95.6	93.1	70.0	98.1	71.1	99.8	88.0
WideNet (S_4)	92.2	89.2	74.2	98.0	74.8	99.6	88.0
ResNet (S_6)	95.6	93.1	70.0	98.0	71.0	99.8	87.9
WideNet (S_6)	92.2	89.2	74.0	98.1	74.1	99.6	87.9

maintaining an optimal balance between efficiency and accuracy in slimmable adaptation.

C. STATISTICAL SIGNIFICANCE TESTS

We evaluate the statistical robustness of each method under various compression ratios, by averaging the results over five independent runs on the Office-31 dataset.

As shown in Fig. 8, SlimDA demonstrates rapidly increasing variance as compression intensifies, from ± 1.1 to ± 9.4 in $A \rightarrow D$ and from ± 3.2 to ± 8.3 in $A \rightarrow W$, indicating unstable adaptation under reduced capacity. In contrast, our method maintains consistently low variance across all compression levels (from ± 0.8 to ± 4.2 in $A \rightarrow D$, from ± 1.6 to ± 3.1 in $A \rightarrow W$) while preserving high mean accuracy (from 95.3% to 93.8%, and from 93.6% to 90.5%, respectively). These results confirm that the proposed *path code reduction* mechanism effectively stabilizes multi-path optimization by eliminating unreliable sub-models, leading to statistically robust and consistent performance even under high compression.

D. HYPER-PARAMETER SENSITIVITY

We further analyze the sensitivity of our method to the key hyper-parameters λ_1 , λ_2 , and λ_3 in the total loss (11). Experiments are conducted on the complicated domain transfer setting ($W \rightarrow A$ domain of Office-31 under 75%

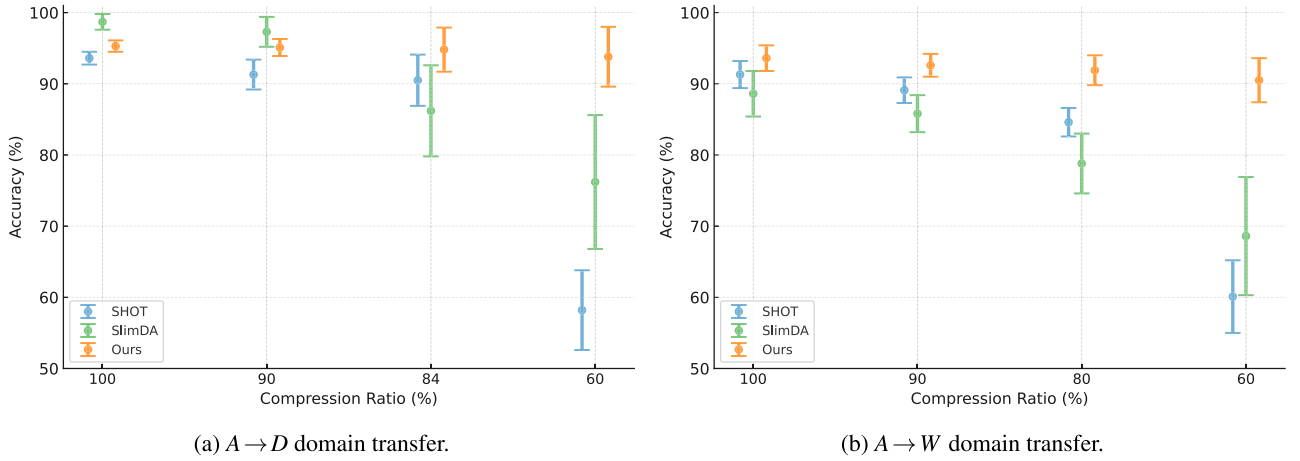


FIGURE 8. Statistical Robustness under Compression. This figure compares the mean accuracy and variance across different compression ratios for the $A \rightarrow D$ and $A \rightarrow W$ domain transfers on the Office-31. The proposed method maintains low variance and stable accuracy across varying model capacities, highlighting its strong statistical robustness under compression.

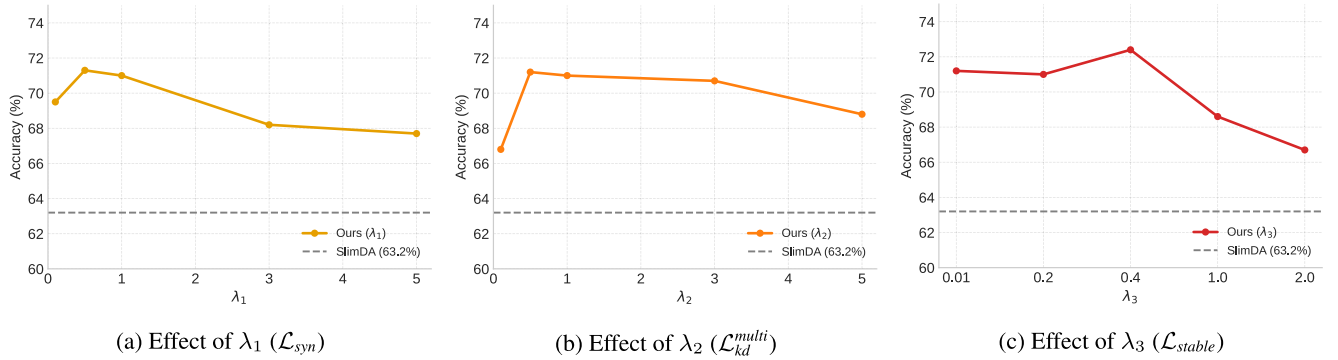


FIGURE 9. Hyper-parameter Sensitivity Analysis. Model performance on the $W \rightarrow A$ domain under 75% compression. Each plot varies one hyper-parameter ($\lambda_1, \lambda_2, \lambda_3$) while keeping others fixed. The dashed line indicates the baseline method (SlimDA).

compression), where each parameter is varied independently while the others are fixed. The results are illustrated in Fig. 9.

The performance remains stable across a broad range of λ_1 and λ_2 values, showing that the contributions of the synthetic-label and multi-path distillation losses are robust to moderate scaling. The accuracy reaches its highest value (71.3%) when $\lambda_1=0.5$ and remains nearly unchanged around $\lambda_1=1.0$, while the variation of λ_2 between 0.5 and 1.5 yields consistent accuracy near 71%. The stabilization coefficient λ_3 exhibits a clear optimum around $\lambda_3=0.4$, where the model achieves 72.4% accuracy. Smaller values under-regularize gradient discrepancies, whereas excessively large values ($\lambda_3>1.0$) constrain sub-model updates, leading to slight degradation. Overall, the method maintains high accuracy under all tested configurations, consistently outperforming the SlimDA baseline (63.2%), which confirms that the proposed multi-loss formulation is both stable and resilient to variations in hyper-parameter settings.

E. MODEL EXTENSION

We further validate the versatility of the proposed method by implementing it within the Wide-ResNet architecture [23]

with experimental results presented in Table 6. This experiment is conducted on Office-31, where both architectures comprise 50 layers representing the whole model. The results of the Wide-ResNet implementation, referred to as *WideNet*, demonstrate that our method achieves remarkable performance when applied to this architecture. In more complex domain transfer scenarios (e.g., $D \rightarrow A$, $W \rightarrow A$), *WideNet* attains comparable or superior performance to *ResNet*. This improvement can be attributed to Wide-ResNet's broader layer structure, which provides a larger representational capacity to manage complex tasks effectively. However, applying this characteristic to simpler domain transfers (e.g., $A \rightarrow W$, $A \rightarrow D$) is more challenging because the large number of parameters makes it difficult to adapt efficiently to subtle domain variations.

These findings emphasize the structure-agnostic nature of our method, which consistently performs well across different residual architectures, including ResNet and Wide-ResNet. By leveraging modularity and residual connections, the proposed framework adapts seamlessly to architectures of varying complexity, thereby ensuring both flexibility and generalization. While smaller models are more efficient for

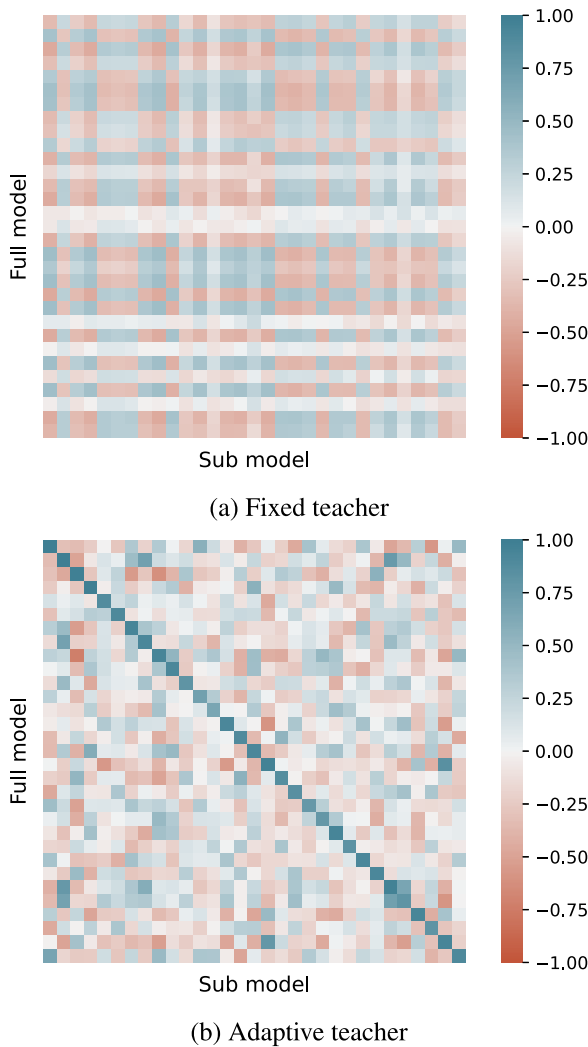


FIGURE 10. Covariance Visualization. Compared with (a) the existing slimmable method using a fixed teacher, (b) the proposed online framework with an adaptive teacher produces sub-models that are more similar to the full model.

simpler tasks, large models are preferable for handling more complex problems. In summary, our method demonstrates robust and consistent performance across various residual architectures, allowing users to select the most appropriate model size based on the complexity of the target domain.

F. FEATURE SIMILARITY VISUALIZATION

To examine the alignment between sub-models and the full model, we visualize the covariance of their feature representations. The visualization results demonstrate that sub-models trained with our adaptive mechanism produce features that are highly similar to those of the full model, as illustrated in Fig. 10. This alignment ensures that the compressed sub-models preserve the capability of the full model even under diverse conditions.

This feature similarity has substantial practical implications. In real-world scenarios where computational resources

TABLE 7. Comparison with Inplace KD.

Method	$A \rightarrow D$	$A \rightarrow W$	$D \rightarrow A$	$D \rightarrow W$	$W \rightarrow A$	$W \rightarrow D$	avg
baseline (S_0)	95.0	92.0	73.6	98.2	72.8	99.6	88.5
Inplace KD (S_0)	90.1	90.1	69.3	96.5	65.2	99.8	85.1
Ours (S_0)	95.8	93.0	70.1	98.1	71.2	99.8	88.0
baseline (S_2)	94.5	91.1	73.2	98.2	72.4	99.6	88.1
Inplace KD (S_2)	90.3	89.4	69.3	96.5	65.2	99.8	85.0
Ours S_2	95.8	93.0	70.2	98.1	71.1	99.8	88.0
baseline (S_4)	92.3	85.8	64.5	97.2	64.2	98.5	83.7
Inplace KD (S_4)	90.2	89.2	69.3	96.5	65.2	99.8	85.0
Ours (S_4)	95.6	93.1	70.0	98.1	71.1	99.8	88.0
baseline (S_6)	57.6	62.5	45.5	71.5	47.5	76.1	60.1
Inplace KD (S_6)	90.1	89.2	68.0	96.5	65.0	99.8	84.7
Ours (S_6)	95.6	93.1	70.0	98.0	71.0	99.8	87.9

are often limited, deploying compressed sub-models that preserve similar feature representations enables high accuracy while minimizing resource consumption. For instance, in edge computing environments or real-time applications, the ability to utilize lightweight models without sacrificing accuracy is critical. The proposed method's adaptive sub-model training mechanism maintains consistent feature similarity, making it particularly effective in such settings. Consequently, it ensures stable performance across diverse tasks and domains, underscoring both the robustness and practical utility of the proposed approach.

VIII. DISCUSSION

In this section, we examine the potential advantages of the proposed approach compared with Inplace knowledge distillation (*Inplace KD*), a common strategy employed in existing slimmable network methods. We also highlight possible directions for improvement and future work to enhance the effectiveness of our framework.

A. COMPARISON WITH INPLACE KD

Existing slimmable methods generally adopt *Inplace KD*, which minimizes the KL divergence between a pre-trained teacher model and its smaller sub-models [2], [16], [17]. However, this method typically requires a substantial number of training epochs, emphasizing the need for a more practical and efficient approach to slimmable network adaptation. Therefore, it is essential to reconsider the conventional training paradigm for slimmable networks [2], [16], [17]. While slimmable networks are designed to deploy sub-models across diverse devices, *Inplace KD* focuses only on minimizing the KL divergence from a single sub-model. In contrast, by applying Eq. 5, we can estimate the KL divergence between the complete model and path codes of varying sizes. This equation serves as a regularization term, preventing the slimmable network from over-optimizing towards either the full model or any specific path code.

To validate these points, we conduct a comprehensive comparison test, as presented in Table 7. For the *Inplace KD*, we minimize the KL divergence of the sampled s_6 , rather than using Eq. 5 and Eq. 9. Following the existing implementation of slimmable methods [2], [4], [17], a fixed teacher model was used for *Inplace KD*. As presented in Table 7, *Inplace KD* achieves performance comparable to, but overall inferior

to, our method. Although the sub-model's accuracies appear similar, Inplace KD underperforms, particularly at lower compression ratios, where the accuracy falls below the *baseline*.

In *Inplace KD*, the divergence loss focuses exclusively on optimizing the smaller sub-models, despite the inherent interdependence among sub-models within a slimmable network. This isolated optimization constrains the overall model performance, even when sub-models such as s_0 and s_2 exhibit reasonable potential. Conversely, our method samples multiple path codes of varying model sizes, enabling more balanced optimization and superior performance across the entire network. Moreover, as discussed in Sec. VI-C, our approach facilitates beneficial collaboration among sub-models, further enhancing network robustness and adaptability.

B. LIMITATIONS AND FUTURE WORKS

While BTN demonstrates strong performance as a slimmable network capable of efficiently generating sub-models of varying sizes, its full model accuracy remains slightly lower than that of model-specific SFUDA methods optimized for single fixed architecture. Bridging this performance gap may require integrating advanced domain adaptation strategies specifically designed to enhance the whole model. Such integration could enable BTN to achieve or surpass the performance of specialized SFUDA methods while preserving its inherent flexibility and computational efficiency. This presents an avenue for further refinement to strengthen BTN's competitiveness across a broader range of applications. Moreover, Eq. 11 introduces additional computational overhead due to the estimation of divergence from path codes. Although the proposed online training mechanism alleviates much of this cost, challenges may persist in highly resource-constrained environments. Despite these limitations, BTN markedly advances UDA for slimmable networks by offering a systematic, efficient training strategy that attains state-of-the-art performance.

Future research should extend the evaluation of BTN beyond conventional classification tasks. Although our experiments utilize widely recognized benchmarks such as Office-31 and Office-Home, these datasets primarily emphasize classification and lack the diversity necessary to capture real-world complexity. To enhance the generalizability of our approach, future studies could investigate its performance in more advanced tasks, such as object detection and semantic segmentation, using larger and more diverse datasets. Such investigations would yield more profound insights into BTN's potential to address a broader range of real-world challenges.

C. EXPECTED IMPACT OF BTN IN REAL-WORLD APPLICATIONS

The proposed BTN framework holds substantial promise for real-world deployment scenarios, where models must operate efficiently under diverse and dynamic hardware constraints. Unlike conventional adaptation approaches that

require separate retraining or fine-tuning for each deployment target, BTN produces a family of sub-models with varying computational capacities from a single training process. Each sub-model can be directly deployed to meet device-specific constraints such as GPU memory limits, energy consumption, or real-time latency requirements.

This capability enables seamless integration into heterogeneous infrastructures. For example, compact sub-models can efficiently serve edge devices or autonomous systems with limited computational resources, whereas the full-capacity model can support high-accuracy inference on cloud servers. Furthermore, the consistency among sub-models, ensured by the path code reduction mechanism, simplifies large-scale system maintenance by preventing accuracy drift across devices. Overall, BTN translates research-stage adaptability into practical, deployable efficiency, offering a unified framework for scalable, resource-aware AI deployment in real-world environments.

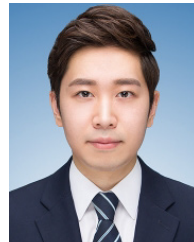
IX. CONCLUSION

In this paper, we introduce a novel BTN designed to generate multiple compressed models from a single network within a source-free unsupervised domain adaptation (SFUDA) setting. Built upon a ResNet-based architecture, the whole model is trained using an innovative online multi-path stabilization scheme. Comprehensive experimental results demonstrate that the proposed BTN consistently outperforms existing model compression approaches when adapting compressed models to unseen target domains. Moreover, the framework effectively preserves model performance, avoiding the degradation typically observed in existing domain adaptation algorithms, even under its slimmable architecture. In the future, we plan to extend our experiments to further validate the generalization-ability of the proposed algorithm in real-world applications, such as autonomous driving systems and surveillance systems.

REFERENCES

- [1] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, no. 55, pp. 1997–2017, 2019.
- [2] J. Yu and T. Huang, "Universally slimmable networks and improved training techniques," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1803–1811.
- [3] T.-W. Chin, A. S. Morcos, and D. Marculescu, "Joslim: Joint widths and weights optimization for slimmable neural networks," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Cham, Switzerland: Springer, 2021, pp. 119–134.
- [4] C. Li, G. Wang, B. Wang, X. Liang, Z. Li, and X. Chang, "Dynamic slimmable network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8607–8617.
- [5] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7167–7176.
- [6] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo, "Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2272–2281.
- [7] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 6028–6039.

- [8] X. Li, W. Chen, D. Xie, S. Yang, P. Yuan, S. Pu, and Y. Zhuang, "A free lunch for unsupervised domain adaptive object detection without source data," in *Proc. Assoc. Advancement Artif. Intell.*, 2021, pp. 8474–8481.
- [9] S. Yang, Y. Wang, J. van de Weijer, L. Herranz, and S. Jui, "Generalized source-free domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 8978–8987.
- [10] J. Nath Kundu, N. Venkat, M. V. Rahul, and R. Venkatesh Babu, "Universal source-free domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4544–4553.
- [11] Z. Qiu, Y. Zhang, H. Lin, S. Niu, Y. Liu, Q. Du, and M. Tan, "Source-free domain adaptation via avatar prototype generation and adaptation," in *Proc. 30th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 2921–2927.
- [12] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS) Workshop*, 2014, pp. 1–11.
- [13] J. H. Cho and B. Hariharan, "On the efficacy of knowledge distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4794–4802.
- [14] W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3962–3971.
- [15] Q. Guo, X. Wang, Y. Wu, Z. Yu, D. Liang, X. Hu, and P. Luo, "Online knowledge distillation via collaborative learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11017–11026.
- [16] R. Meng, W. Chen, S. Yang, J. Song, L. Lin, D. Xie, S. Pu, X. Wang, M. Song, and Y. Zhuang, "Slimmable domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 7141–7150.
- [17] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once-for-all: Train one network and specialize it for efficient deployment," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020, pp. 1–14.
- [18] I. Cohen, Y. Huang, J. Chen, J. Benesty, J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise Reduction in Speech Processing*. Burlington, MA, USA: Academic, 2009, pp. 1–4.
- [19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
- [20] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer*, 2010, pp. 213–226.
- [21] X. Dong, S. Chen, and S. J. Pan, "Learning to prune deep neural networks via layer-wise optimal brain surgeon," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 4857–4867.
- [22] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1492–1500.
- [23] S. Zagoruyko and N. Komodakis, "Wide residual networks," 2016, *arXiv:1605.07146*.
- [24] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer*, 2016, pp. 646–661.
- [25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.
- [26] R. Müller, S. Kornblith, and G. Hinton, "When does label smoothing help?" in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2019, pp. 4694–4703.
- [27] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Proc. CAP*, vol. 367, 2005, pp. 281–296.
- [28] R. Gomes, A. Krause, and P. Perona, "Discriminative clustering by regularized information maximization," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, Sep. 2010, pp. 703–711.
- [29] J. T. Barron, "A general and adaptive robust loss function," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4326–4334.
- [30] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5385–5394.
- [31] J. Duda, "SGD momentum optimizer with step estimation by online parabola model," 2019, *arXiv:1907.07063*.
- [32] M. Long, Z. Cao, J. Wang, and M. L. Jordan, "Conditional adversarial domain adaptation," in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 1352–1362.
- [33] Y. Wu, D. Inkpen, and A. El-Roby, "Dual mixup regularized learning for adversarial domain adaptation," in *Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer*, 2020, pp. 540–555.
- [34] M. Lin, R. Ji, S. Li, Y. Wang, Y. Wu, F. Huang, and Q. Ye, "Network pruning using adaptive exemplar filters," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 7357–7366, Dec. 2022.
- [35] 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, Oct. 2021.
- [36] X. Yang and X. Wang, "Neural metalmorphosis," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2024, pp. 1–20.
- [37] I. Diamant, A. Rosenfeld, I. Achituve, J. Goldberger, and A. Netzer, "De-confusing pseudo-labels in source-free domain adaptation," in *Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer*, 2025, pp. 108–125.
- [38] Y. Mitsuzumi, A. Kimura, and H. Kashima, "Understanding and improving source-free domain adaptation from a theoretical perspective," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, May 2024, pp. 28515–28524.
- [39] Z. Shuai, Z. Linchao, W. Xiaohan, and Y. Yi, "Slimmable networks for contrastive self-supervised learning," *Int. J. Comput. Vis.*, vol. 133, no. 3, pp. 1222–1237, 2025.
- [40] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9630–9640.
- [41] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko, "VisDA: The visual domain adaptation challenge," 2017, *arXiv:1710.06924*.



SEUNGMO SEO received the B.S. degree in electronics and electrical engineering and the M.S. degree in artificial intelligence from Chung-Ang University, Seoul, South Korea, in 2020 and 2022, respectively, where he is currently pursuing the Ph.D. degree, under the supervision of Prof. Jongwon Choi. His research interests include machine learning, with a focus on transfer learning and efficient AI, as well as their applications in real-world scenarios.



JONGSU YOUN received the B.S. degree in digital imaging of integrative engineering and the M.S. degree in artificial intelligence from Chung-Ang University, South Korea, in 2021 and 2023, respectively, where he is currently pursuing the Ph.D. degree, under the supervision of Prof. Jongwon Choi. His research interests include multimodality, especially audio and vision.



SEUNGJIN JUNG received the B.S. degree in information and communication engineering from Hoseo University, South Korea, in 2019, and the M.S. degree in digital imaging from Chung-Ang University, South Korea, in 2022, where he is currently pursuing the Ph.D. degree in artificial intelligence. His current research interests include machine learning, domain generalization, post-hoc calibration for model reliability, and spurious correlation for data debiasing.



MINJI KWAK received the B.S. degree in information and statistics from Dongduk Women's University, Seoul, South Korea, in 2022, and the M.S. degree in AI imaging from Chung-Ang University, Seoul, in 2024. She conducted this research while at Chung-Ang University and is currently conducting research with the HDC-Laboratory, Seoul. Her research interests include human pose estimation and domain adaptation.



HOJOON JUNG received the B.S. degree from the School of Photography and Motion Picture, Kyung-Il University, South Korea, in 2021, and the M.S. degree from the Department of Cultural Properties, Chung-Ang University, Seoul, South Korea, in 2023. He is currently pursuing the Ph.D. degree in AI imaging with Chung-Ang University. His current research interests include computer vision and 3-D reconstruction.



HEEGON JIN received the B.S. degree in digital imaging and the M.S. degree in artificial intelligence from Chung-Ang University, Seoul, South Korea, in 2021 and 2023, respectively. From 2023 to 2025, he was with the NCSOFT NLP Center, Seongnam. He conducted this research while at Chung-Ang University and is currently with Global AI Platform. His current research interests include knowledge distillation, cross architecture knowledge transfer, and data-efficient optimization.



JONGWON CHOI (Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from KAIST, Daejeon, South Korea, in 2012 and 2014, respectively, and the Ph.D. degree in electrical engineering from Seoul National University, Seoul, South Korea, in 2018. From 2018 to 2020, he was with the Research Intelligence Research Center, Samsung SDS, Seoul, as a Research Engineer. In 2020, he joined the Department of Advanced Imaging, Chung-Ang University, Seoul, where he is currently an Associate Professor. His research interests include deepfake detection, semi-supervised learning, and deep learning architecture.

...