In this Appendix, we provide more details of ENL-DEE Details.

A. Algorithm

Algorithm 1 outlines the training process for ENL-DEE, which integrates early exit classifiers into a pre-trained model to enhance efficiency by optimizing domain-specific loss functions. Algorithm 2 describes the inference procedure, where the early exit mechanism dynamically determines the output layer based on confidence thresholds, reducing computational overhead.

Algorithm 1 BEE-NTL Training: Bayesian Early-Exit with Entropy-Based Routing

Require: pre-trained backbone M (params $\boldsymbol{\delta}_M$); exits E; epochs e; source $\mathcal{D}_s$, target $\mathcal{D}_t$; coefficients $\alpha, \beta \geq 0$; temperature $\tau > 0$; domain routing priors $\boldsymbol{\pi}^{(s)}, \boldsymbol{\pi}^{(t)}$.
for $k = 1$ to e **do**
  Sample mini-batches $\{(\boldsymbol{x}_s, y_s)\} \sim \mathcal{D}_s$ and $\{(\boldsymbol{x}_t, y_t)\} \sim \mathcal{D}_t$
  Forward pass for each exit $i = 1, \ldots, E$ and for $\boldsymbol{x} \in \{\boldsymbol{x}_s, \boldsymbol{x}_t\}$:
    Amortized posterior: $m(\boldsymbol{z}_i \mid \boldsymbol{x}) = \mathcal{N}\big(\boldsymbol{z}_i \mid f_e^{\mu}(\boldsymbol{x}), f_e^{\Sigma}(\boldsymbol{x})\big)$
    Reparameterization: $\boldsymbol{z}_i = f(\boldsymbol{x}, \epsilon), \qquad \epsilon \sim \mathcal{N}(0, I)$
    Exit-i head prediction: $p_{\boldsymbol{\theta}}^{(i)}(y \,|\, \boldsymbol{x})$
    Entropy (Eq. 6) and Entropy-based routing (Eq. 7).
  Source loss (Eq. 8):
    $K(i) = \mathbb{I}[i = E] - \mathbb{I}[i < E]$)
    $\mathcal{L}_s(\boldsymbol{x}_s, y_s) = -\mathcal{L}_{\text{ELBO}}^{(s)}(\boldsymbol{x}_s, y_s) + \alpha \sum_{i=1}^{E} \eta_i(\boldsymbol{x}_s)\, K(i)\, H_i(\boldsymbol{x}_s)$
  Target loss (Eq. 10):
    $J(i) = -K(i)$
    $\mathcal{L}_t(\boldsymbol{x}_t, y_t) = \mathcal{F}^{(t)}(\boldsymbol{x}_t, y_t) + \alpha \sum_{i=1}^{E} \eta_i(\boldsymbol{x}_t)\, J(i)\, H_i(\boldsymbol{x}_t)$
  Total objective (Eq. 11):
    $\mathcal{J} = \mathbb{E}_{(\boldsymbol{x}_s, y_s) \sim \mathcal{D}_s}\big[\mathcal{L}_s(\boldsymbol{x}_s, y_s)\big] + \beta\, \mathbb{E}_{(\boldsymbol{x}_t, y_t) \sim \mathcal{D}_t}\big[\mathcal{L}_t(\boldsymbol{x}_t, y_t)\big]$
  Update $\boldsymbol{\theta}$ by gradient descent on $\mathcal{J}$)
end for

Algorithm 2 ENL-DEE **Testing** (Entropy-based Early Exit).

Require: trained M^*; input $\boldsymbol{x}$; entropy thresholds $\mathbf{R} = \{r_1, \ldots, r_E\}$; temperature τ.
for $i = 1$ to E **do**
  Compute $p_{\theta}^{(i)}(y \mid \boldsymbol{x})$ and entropy $H_i(\boldsymbol{x})$
  if $H_i(\boldsymbol{x}) \geq r_i$ **then**
    Return exit i prediction $\arg\max_c \; p_{\theta}^{(i)}(y = c \mid \boldsymbol{x})$
  end if
end for

B. The architecture details of exit classifier

If M is ViT, we add four exit classifiers in victim model M, as shown in Table 9.

The architecture details of exit classifiers for other networks, such as Resnet-series are similar to this.

C. Derivation of the ELBO Bound

Starting from the mixture evidence in Eq. equation 4,

$$p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) = \sum_{i=1}^{E} \eta_i(\boldsymbol{x}) \int p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i)\, p_{\boldsymbol{\theta}}(\boldsymbol{z}_i \mid \boldsymbol{x}, d = i)\, d\boldsymbol{z}_i. \tag{13}$$

Table 9. The architecture details of exit classifiers, N is the number of classes.

Exit Index	First layer	Second layer
Exit1	$Linear(192, 5 * N)$	$Linear(5 * N, N)$
Exit2	$Linear(192, 5 * N)$	$Linear(5 * N, N)$
Exit3	$Linear(192, 5 * N)$	$Linear(5 * N, N)$
Exit4	$Linear(192, 5 * N)$	$Linear(5 * N, N)$

Step 1: Insert a routing posterior and apply Jensen. Let $q(d \mid \boldsymbol{x})$ be any distribution over $\{1, \ldots, E\}$. Multiplying and dividing by $q(d \mid \boldsymbol{x})$ gives

$$\begin{aligned}\log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) &= \log \sum_{i=1}^{E} q(d=i \mid \boldsymbol{x}) \, \frac{\eta_i(\boldsymbol{x})}{q(d=i \mid \boldsymbol{x})} \int p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i) \, p_{\boldsymbol{\theta}}(\boldsymbol{z}_i \mid \boldsymbol{x}, d=i) \, d\boldsymbol{z}_i \\ &= \log \mathbb{E}_{d \sim q(\cdot \mid \boldsymbol{x})} \left[\frac{\eta_d(\boldsymbol{x})}{q(d \mid \boldsymbol{x})} \int p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_d) \, p_{\boldsymbol{\theta}}(\boldsymbol{z}_d \mid \boldsymbol{x}, d) \, d\boldsymbol{z}_d \right] \\ &\geq \mathbb{E}_{d \sim q(\cdot \mid \boldsymbol{x})} \left[\log \frac{\eta_d(\boldsymbol{x})}{q(d \mid \boldsymbol{x})} + \log A_d(\boldsymbol{x}) \right], \quad A_d(\boldsymbol{x}) := \int p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_d) \, p_{\boldsymbol{\theta}}(\boldsymbol{z}_d \mid \boldsymbol{x}, d) \, d\boldsymbol{z}_d. \end{aligned} \tag{14}$$

Recognizing the first expectation as the negative KL divergence yields

$$\log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) \geq -\mathbb{KL}\big(q(d \mid \boldsymbol{x}) \,\|\, \boldsymbol{\eta}(\boldsymbol{x})\big) + \mathbb{E}_{d \sim q(\cdot \mid \boldsymbol{x})}[\log A_d(\boldsymbol{x})]. \tag{15}$$

Step 2: Insert the fixed Gaussian encoder and apply Jensen again. For each exit i, define a frozen Gaussian encoder

$$m(\boldsymbol{z}_i \mid \boldsymbol{x}) = \mathcal{N}\big(\boldsymbol{z}_i \mid f_e^{\mu}(\boldsymbol{x}), f_e^{\Sigma}(\boldsymbol{x})\big),$$

where f_e is the backbone feature extractor. Then,

$$A_i(\boldsymbol{x}) = \int m(\boldsymbol{z}_i \mid \boldsymbol{x}) \, \frac{p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i) \, p_{\boldsymbol{\theta}}(\boldsymbol{z}_i \mid \boldsymbol{x}, d=i)}{m(\boldsymbol{z}_i \mid \boldsymbol{x})} \, d\boldsymbol{z}_i = \mathbb{E}_{\boldsymbol{z}_i \sim m(\boldsymbol{z}_i \mid \boldsymbol{x})} \left[\frac{p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i) \, p_{\boldsymbol{\theta}}(\boldsymbol{z}_i \mid \boldsymbol{x}, d=i)}{m(\boldsymbol{z}_i \mid \boldsymbol{x})} \right], \tag{16}$$

$$\log A_i(\boldsymbol{x}) \geq \mathbb{E}_{\boldsymbol{z}_i \sim m(\boldsymbol{z}_i \mid \boldsymbol{x})} \Big[\log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i) + \log p_{\boldsymbol{\theta}}(\boldsymbol{z}_i \mid \boldsymbol{x}, d=i) - \log m(\boldsymbol{z}_i \mid \boldsymbol{x}) \Big]. \tag{17}$$

Step 3: Adopt a standard prior for z. Assuming $p_{\boldsymbol{\theta}}(\boldsymbol{z}_i \mid \boldsymbol{x}, d=i) = p(\boldsymbol{z}_i) = \mathcal{N}(0, I)$, substitute Eq. equation 17 into Eq. equation 15:

$$\begin{aligned}\log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) &\geq -\mathbb{KL}\big(q(d \mid \boldsymbol{x}) \,\|\, \boldsymbol{\eta}(\boldsymbol{x})\big) + \sum_{i=1}^{E} q(d=i \mid \boldsymbol{x}) \mathbb{E}_{\boldsymbol{z}_i \sim m(\boldsymbol{z}_i \mid \boldsymbol{x})} \Big[\log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i) - \log \tfrac{m(\boldsymbol{z}_i \mid \boldsymbol{x})}{p(\boldsymbol{z}_i)} \Big] \\ &= \sum_{i=1}^{E} q(d=i \mid \boldsymbol{x}) \Big[\mathbb{E}_m \log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i) - \mathbb{KL}\big(m(\boldsymbol{z}_i \mid \boldsymbol{x}) \,\|\, p(\boldsymbol{z}_i)\big) \Big] - \mathbb{KL}\big(q(d \mid \boldsymbol{x}) \,\|\, \boldsymbol{\eta}(\boldsymbol{x})\big). \end{aligned} \tag{18}$$

Step 4: Replace the routing regularizer with the domain prior. We decompose the discrete regularizer as

$$-\mathbb{KL}(q \| \eta) = -\mathbb{KL}(q \| \pi) + \mathbb{E}_q \Big[\log \tfrac{\pi_d}{\eta_d} \Big],$$

leading to

$$\log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}) \geq \sum_{i=1}^{E} q(d=i \mid \boldsymbol{x}) \Big[\mathbb{E}_m \log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i) - \mathbb{KL}\big(m(\boldsymbol{z}_i \mid \boldsymbol{x}) \,\|\, p(\boldsymbol{z}_i)\big) \Big] - \mathbb{KL}\big(q(d \mid \boldsymbol{x}) \,\|\, \boldsymbol{\pi}\big) + \mathbb{E}_q \Big[\log \tfrac{\pi_d}{\eta_d(\boldsymbol{x})} \Big]. \tag{19}$$

Step 5: Entropy-based routing and the final bound. In practice we set $q(d = i \mid \boldsymbol{x}) \equiv \eta_i(\boldsymbol{x})$ (entropy-based routing), which makes the last term constant and removable. Thus, the final *Exit-Mixture ELBO* is:

$$\mathcal{L}_{\text{ELBO}}(\boldsymbol{x}, y) = \sum_{i=1}^{E} \eta_i(\boldsymbol{x}) \Big[\mathbb{E}_{\boldsymbol{z}_i \sim m(\boldsymbol{z}_i \mid \boldsymbol{x})} \log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{z}_i) - \mathbb{KL}\big(m(\boldsymbol{z}_i \mid \boldsymbol{x}) \,\|\, p(\boldsymbol{z}_i)\big) \Big] - \mathbb{KL}\big(q(d \mid \boldsymbol{x}) \,\|\, \boldsymbol{\pi}\big) \;\; \leq \;\; \log p_{\boldsymbol{\theta}}(y \mid \boldsymbol{x}). \tag{20}$$

The final KL term aligns the data-dependent routing $q(d \mid \boldsymbol{x})$ with the corresponding domain prior $\boldsymbol{\pi}$. Based on the frozen backbone, we define a fixed Gaussian encoder

$$m(\boldsymbol{z}_i \mid \boldsymbol{x}) = \mathcal{N}\big(\boldsymbol{z}_i \mid f_e^{\mu}(\boldsymbol{x}), f_e^{\Sigma}(\boldsymbol{x})\big),$$

where f_e outputs the mean $f_e^{\mu}(\boldsymbol{x})$ and covariance $f_e^{\Sigma}(\boldsymbol{x})$ for the latent variable $\boldsymbol{z}_i$. Using the reparameterization trick,

$$\boldsymbol{z}_i = f(\boldsymbol{x}, \epsilon), \qquad \epsilon \sim \mathcal{N}(0, I),$$

we obtain a differentiable formulation that separates the stochastic noise ϵ from the model parameters, enabling efficient gradient-based optimization while keeping the backbone frozen.

For the target domain, minimizing the reversed ELBO gives the free-energy form:

$$\mathcal{F}^{(t)}(\boldsymbol{x}_t, y_t) = \sum_{i=1}^{E} \eta_i(\boldsymbol{x}_t) \Big[\mathbb{E}_{\boldsymbol{z}_i \sim m(\boldsymbol{z}_i \mid \boldsymbol{x}_t)} \log p_{\boldsymbol{\theta}}(y_t \mid \boldsymbol{z}_i) - \mathbb{KL}\big(m(\boldsymbol{z}_i \mid \boldsymbol{x}_t) \,\|\, p(\boldsymbol{z}_i)\big) \Big] + \mathbb{KL}\big(q(d \mid \boldsymbol{x}_t) \,\|\, \boldsymbol{\pi}^{(t)}\big), \tag{21}$$

$$\mathcal{L}_t(\boldsymbol{x}_t, y_t) = \mathcal{F}^{(t)}(\boldsymbol{x}_t, y_t) + \alpha \sum_{i=1}^{E} \eta_i(\boldsymbol{x}_t)\, J(i)\, H_i(\boldsymbol{x}_t), \tag{22}$$

where the first term penalizes predictive fit and rewards latent simplicity, while softly aligning routing $q(d \mid \boldsymbol{x}_t)$ to the shallow-biased prior $\boldsymbol{\pi}^{(t)}$. The second term enforces shallow low-entropy (early confident exit) and deep high-entropy (discourage late exit).

D. Extended Evaluation on Additional Datasets

Consistent with the observations in the main text, our method often leads to reduced performance on out-of-domain (target domain) datasets. It achieves optimal performance in many cases.

D.1. Extended Evaluation on VLCS Datasets

As shown in Table 10, our method demonstrates strong performance.

Table 10. Model accuracy on *Target-Specified Domain (VLCS Dataset)* task, we use ViT as our model, *S/T* denotes *Source/Target*, SUN09(S) and VOC2007(V).

Methods	**S/T**	S	V	PG
SL	S	70.10 ± 0.02	48.66 ± 0.23	0.00 ± 0.02
	V	70.00 ± 0.55	64.94 ± 0.72	0.00 ± 0.07
NTL	S	68.60 ± 0.36	48.07 ± 0.38	-0.91 ± 0.34
	V	46.92 ± 0.57	35.37 ± 1.29	-6.49 ± 0.33
CUTI	S	63.11 ± 0.92	33.83 ± 0.72	7.84 ± 0.84
	V	45.12 ± 0.41	64.80 ± 0.78	$\mathbf{24.74 \pm 0.83}$
ENL-DEE	S	63.22 ± 0.41	33.12 ± 0.24	$\mathbf{8.66 \pm 0.34}$
	V	66.22 ± 0.23	48.17 ± 0.97	24.04 ± 0.97

D.2. Extended Evaluation on Office-Home Dataset

As shown in Table 11, our method achieves the best performance across all domain combinations in this dataset.

Table 11. Model accuracy on *Target-Specified Domain (Office-Home Dataset)* task, we use ViT as our model, *S/T* denotes *Source/Target*.

Methods	**S/T**	Real-World	Art	Clipart	Product	PG
SL	Real-World	61.60 ± 0.24	30.17 ± 0.94	36.24 ± 1.67	49.89 ± 0.25	0.00 ± 0.13
	Art	30.80 ± 0.33	37.20 ± 0.84	17.89 ± 1.12	16.48 ± 0.94	0.00 ± 0.34
	Clipart	40.46 ± 0.78	23.55 ± 0.48	74.10 ± 0.66	38.83 ± 1.26	0.00 ± 0.81
	Product	39.08 ± 0.99	20.25 ± 1.17	30.96 ± 0.82	75.40 ± 0.43	0.00 ± 0.17
NTL	Real-World	47.33 ± 1.24	21.49 ± 2.02	4.59 ± 1.93	11.29 ± 2.03	12.04 ± 2.11
	Art	10.80 ± 1.29	24.67 ± 2.39	2.52 ± 1.29	1.58 ± 0.93	4.22 ± 1.96
	Clipart	7.82 ± 1.32	4.13 ± 0.94	61.93 ± 0.73	4.97 ± 0.35	16.47 ± 0.72
	Product	11.95 ± 0.34	3.31 ± 0.93	4.59 ± 2.30	67.40 ± 0.49	$\mathbf{15.48 \pm 0.91}$
CUTI	Real-World	39.70 ± 0.72	12.81 ± 1.05	5.50 ± 2.42	10.84 ± 1.91	7.15 ± 1.47
	Art	9.20 ± 0.25	22.73 ± 1.87	3.90 ± 0.27	2.48 ± 1.61	2.06 ± 1.35
	Clipart	7.36 ± 0.73	2.89 ± 1.66	53.37 ± 1.05	3.84 ± 1.82	8.85 ± 1.28
	Product	11.26 ± 0.43	3.31 ± 2.01	5.05 ± 1.77	64.33 ± 1.55	12.49 ± 1.64
ENL-DEE	Real-World	51.36 ± 2.22	18.18 ± 1.25	6.19 ± 0.74	10.38 ± 1.30	$\mathbf{16.94 \pm 1.03}$
	Art	22.76 ± 0.90	34.43 ± 2.14	3.21 ± 1.2	2.93 ± 1.20	$\mathbf{9.32 \pm 0.92}$
	Clipart	13.79 ± 1.05	6.61 ± 2.05	66.53 ± 1.58	10.16 ± 0.94	$\mathbf{16.52 \pm 1.11}$
	Product	17.47 ± 0.35	6.61 ± 0.26	9.17 ± 0.45	71.1 ± 1.24	14.71 ± 1.26

D.3. Extended Evaluation on ImageNet200/200-C Dataset

As shown in Table 12, our method achieves the best performance across all domain combinations in this dataset.

To investigate the impact of distribution shifts and sample sizes, our evaluation encompasses a broad spectrum of scenarios, ranging from moderate to large shifts (e.g., the partially overlapping CIFAR10↔STL10). To more precisely quantify the relationship between performance and shift magnitude, we further extend our benchmarks to ImageNet200-C. By evaluating across varying corruption levels (Levels 1–3), as shown in Table 12, we demonstrate that ENL-DEE consistently achieves significant improvements even as the domain shift intensifies. This robust performance on a large-scale dataset like ImageNet200 further validates the scalability and effectiveness of our approach.

Table 12. Model accuracy on *Target-Specified Domain* (ImageNet200/200-C Dataset), *S/T Source/Target*.

Methods	**S/T**	S (brightness)	Level 1	Level 2	Level 3	PG
SL	S	52.41 ± 0.38	45.80 ± 1.32	37.40 ± 1.37	25.30 ± 1.15	0.00 ± 0.84
NTL	S	13.03 ± 1.23	1.50 ± 0.45	1.30 ± 0.69	0.80 ± 1.09	-4.41 ± 0.14
CUTI	S	16.50 ± 0.93	2.10 ± 1.20	1.80 ± 0.91	1.80 ± 0.94	-1.64 ± 1.20
ENL-DEE	S	48.43 ± 0.92	3.80 ± 0.24	0.50 ± 0.39	0.00 ± 1.04	$\mathbf{30.75 \pm 0.96}$

D.4. Sensitivity Analysis on Temperature Hyperparameter

Table 13 demonstrates that ENL-DEE is relatively insensitive to variations in the temperature parameter τ. Performance gains are consistently observed for τ values between 0.5 and 2.0. For ease of implementation, we fix τ across all backbone architectures, showcasing the generalizability of this parameter choice.

Table 13. Sensitivity to the temperature τ.

Methods	**S/T**	S (brightness)	Level 1	Level 2	Level 3	PG
SL	S	52.41 ± 0.38	45.80 ± 1.32	37.40 ± 1.37	25.30 ± 1.15	0.00 ± 0.84
ENL-DEE	S ($\tau = 0.5$)	50.13 ± 0.55	10.90 ± 1.03	7.30 ± 0.73	0.50 ± 0.94	27.65 ± 1.03
ENL-DEE	S ($\tau = 1.0$)	48.43 ± 0.92	3.80 ± 0.24	0.50 ± 0.39	0.00 ± 1.04	$\mathbf{30.75 \pm 0.96}$
ENL-DEE	S ($\tau = 2.0$)	41.33 ± 0.35	1.60 ± 0.86	0.00 ± 1.21	0.00 ± 1.53	24.55 ± 1.01

D.5. Efficiency Benchmarking on Large-Scale Datasets

To provide a comprehensive evaluation, we extend our comparison to the ImageNet-200 and ImageNet-200-C benchmarks. Beyond predictive performance, we rigorously analyze the computational efficiency of ENL-DEE by reporting the number of trainable parameters, training duration, FLOPs, and inference latency. As summarized in Table 14, the results demonstrate that

ENL-DEE achieves a superior balance between accuracy and efficiency, maintaining significantly lower overhead across all metrics compared to baseline models.

Table 14. Training time, FLOPs, inference time comparison.

Methods	SL	NTL	CUTI	ENL-DEE
Trainable Params (M)	85.81	85.81	85.81	**7.75**
Training Time (h)	0.1188	0.1998	0.2048	**0.0028**
FLOPs (GFLOPs/img)	2.91	2.91	2.91	**2.04**
Inference Time (s)	0.6128	0.6207	0.6242	**0.4713**

D.6. Ablation Study on Various Transformer Architectures

Table 15. Model accuracy on *Target-Specified Domain*(Office-Home Dataset) task, we use different model network architectures as our models, *S/T* denotes *Source/Target*, *ViT* is Vision Transformer, *SWIN* is Swin Transformer. The last column (PG) shows the absolute average accuracy gain (higher is better) of our method over the corresponding supervised-learning (SL) baseline with the same backbone and S/T pair; therefore PG is **0.00** for SL by definition (it serves as the reference).

Methods	**S/T**	Real-World	Art	Clipart	Product	PG
SL (ViT)	Real-World	61.60 ± 0.24	30.17 ± 0.94	36.24 ± 1.67	49.89 ± 0.25	0.00 ± 0.13
	Art	30.80 ± 0.33	37.20 ± 0.84	17.89 ± 1.12	16.48 ± 0.94	0.00 ± 0.34
	Clipart	40.46 ± 0.78	23.55 ± 0.48	74.10 ± 0.66	38.83 ± 1.26	0.00 ± 0.81
	Product	39.08 ± 0.99	20.25 ± 1.17	30.96 ± 0.82	75.40 ± 0.43	0.00 ± 0.17
ENL-DEE	Real-World	51.36 ± 2.22	18.18 ± 1.25	6.19 ± 0.74	10.38 ± 1.30	$\mathbf{16.94 \pm 1.03}$
	Art	22.76 ± 0.90	34.43 ± 2.14	3.21 ± 1.2	2.93 ± 1.20	$\mathbf{9.32 \pm 0.92}$
	Clipart	13.79 ± 1.05	6.61 ± 2.05	66.53 ± 1.58	10.16 ± 0.94	$\mathbf{16.52 \pm 1.11}$
	Product	17.47 ± 0.35	6.61 ± 0.26	9.17 ± 0.45	71.1 ± 1.24	$\mathbf{14.71 \pm 1.26}$
SL (SWIN)	Real-World	65.31 ± 0.06	31.40 ± 0.43	36.93 ± 1.19	48.98 ± 1.05	0.00 ± 0.72
	Art	18.62 ± 1.22	29.80 ± 0.91	13.99 ± 1.17	7.90 ± 0.82	0.00 ± 1.01
	Clipart	37.24 ± 1.39	26.86 ± 1.36	73.89 ± 1.25	36.12 ± 1.52	0.00 ± 1.36
	Product	42.99 ± 1.02	27.27 ± 1.03	33.94 ± 0.99	78.80 ± 0.97	0.00 ± 1.08
ENL-DEE SWIN	Real-World	54.43 ± 1.71	19.01 ± 0.86	7.11 ± 1.22	13.54 ± 1.63	$\mathbf{15.00 \pm 1.44}$
	Art	15.40 ± 1.37	28.10 ± 0.83	3.21 ± 1.49	2.93 ± 0.37	$\mathbf{4.62 \pm 1.28}$
	Clipart	8.97 ± 1.64	7.02 ± 0.92	66.73 ± 2.11	6.09 ± 1.29	$\mathbf{18.89 \pm 1.73}$
	Product	18.62 ± 1.33	7.02 ± 0.95	7.57 ± 1.25	72.40 ± 1.83	$\mathbf{15.51 \pm 1.54}$

D.7. Study on Target-Free Setting

Table 16. Model accuracy on *Target-Free (PACS Dataset)* task, Vision Transformer Architecture, *S/T* denotes *Source/Target*.

Methods	**S/T**	S	C	A	P	PG
SL	P	34.44 ± 1.42	35.90 ± 0.92	39.71 ± 1.49	92.80 ± 0.95	0.0 ± 0.95
NTL	P	23.21 ± 0.94	15.81 ± 1.44	23.53 ± 0.25	82.60 ± 0.83	-31.05 ± 0.95
CUTI	P	32.14 ± 1.01	26.92 ± 0.94	28.43 ± 0.23	86.80 ± 0.34	1.52 ± 0.95
ENL-DEE	P	24.23 ± 0.45	33.33 ± 0.77	37.25 ± 0.53	92.20 ± 1.14	$\mathbf{4.48 \pm 0.76}$