VT-DUDA: Visual Token Conditioning for Diffusion-guided Unsupervised Domain Adaptation

Anonymous authors
Paper under double-blind review

Abstract

Unsupervised domain adaptation (UDA) aims to learn a target-domain classifier from labeled source data and unlabeled target data under distribution shift. Recent diffusion-based UDA methods approach this problem by synthesizing labeled target-style images and training on the resulting synthetic data. However, their performance depends heavily on the conditioning design: class prompts provide only coarse guidance, while domain adaptation modules mainly control appearance, which may leave target-style synthesis insufficiently specified. We propose VT-DUDA, a visual-token conditioning framework for diffusion-guided UDA. Instead of relying only on text prompts, VT-DUDA uses source images to provide additional instance-level visual context for target-style synthesis. Specifically, it converts each source image into a compact set of visual tokens and injects them, together with text embeddings, into the standard cross-attention pipeline of a latent diffusion model. This provides instance-dependent conditioning beyond text alone, while synthesis is performed with the target-domain adapter branch. Because guidance is represented explicitly as a token sequence, the same interface also permits inference-time manipulation of the conditioning signal through token selection and token-strength adjustment. The proposed method preserves the standard diffusion objective and can be integrated into existing adapter-based diffusion frameworks without modifying the backbone. Across Office-31, Office-Home, and VisDA-2017, VT-DUDA improves average target-domain accuracy over strong discriminative and diffusion-based UDA baselines. The results suggest that, in generation-based UDA, a stronger conditioning interface can improve the downstream usefulness of synthetic target-style data.

1 Introduction

Unsupervised domain adaptation (UDA) aims to learn a target-domain classifier from labeled source data and unlabeled target data under distribution shift. Classical UDA methods address this problem primarily in feature space through discrepancy minimization Li et al. (2020); Long et al. (2015; 2018), adversarial alignment Ganin et al. (2016); Saito et al. (2018), or prediction- and entropy-based regularization Xu et al. (2019); Rangwani et al. (2022); Zhang et al. (2023b; 2019); Xia et al. (2022); Wu et al. (2025); Jin et al. (2020). More recently, diffusion-based approaches Zhuang et al. (2024); Zhang et al. (2025); Rombach et al. (2022); Ho et al. (2020); Nichol & Dhariwal (2021); Song et al. (2020b); Song & Ermon (2020); Dhariwal & Nichol (2021) have explored a data-centric alternative: instead of aligning source and target features directly, they adapt a generative model to synthesize labeled target-style images and then train a classifier on the resulting synthetic set. In this view, the quality of the constructed synthetic supervision becomes an important factor in downstream adaptation.

Latent diffusion models are a natural choice in this setting because they support flexible conditioning through cross-attention Nichol et al. (2022); Podell et al. (2024); Rombach et al. (2022); Vaswani et al. (2017); Ho & Salimans (2022); Zhang et al. (2023a); Ye et al. (2023). In principle, target-style synthesis can be guided by class prompts, domain indicators, or lightweight adaptation modules. In practice, however, diffusion-guided UDA often relies on conditioning that maps distinct source instances of the same class to the same class-level text signal. That is, on the source side, multiple labeled source images from the same class are typically

associated with the same class-level text condition (e.g., "a bed"), even though different source exemplars may contain different visual details that could be useful for target-style synthesis. On the target side, when target labels are unavailable, the text condition may be reduced to a generic prompt such as "an image." As a result, the denoiser may be trained under conditions that provide limited instance-level guidance for synthetic data construction.

In this paper, we study the role of conditioning specificity in diffusion-guided UDA and propose **VT-DUDA**, a **V**isual-**T**oken conditioning framework for **D**iffusion-guided **U**nsupervised **D**omain **A**daptation. The core idea is to augment text conditioning with instance-level visual tokens extracted from individual images. Specifically, given an image, VT-DUDA maps it to a compact sequence of visual tokens and concatenates these tokens with text embeddings through the standard cross-attention interface of a latent diffusion model. On the source side, the visual tokens are paired with class prompts, so that different labeled source instances sharing the same class label are not represented by exactly the same conditioning context. In this way, the class prompt specifies the category, while the visual tokens provide instance-specific information associated with the current source sample. On the target side, where class labels are unavailable, the visual tokens are paired with a generic prompt such as "An image," providing additional instance-level context beyond text alone for unlabeled target denoising.

A key design choice is to introduce target-side visual tokens not only to enrich the target-side conditioning signal, but also to align the token-conditioned interface used during training and generation. Source-side supervision makes the tokenized context depend on both class labels and source-instance visual features during class-conditioned synthesis. Target-side denoising, in turn, exposes the target branch to the same token-conditioned cross-attention format on target-domain data. As a result, after fine-tuning, visual tokens extracted from a source image are concatenated with the class prompt and fed to the target branch at generation time. This enriches class-conditioned prompting with instance-dependent visual context, reducing the extent to which different source samples of the same class share identical conditioning inputs.

This perspective motivates evaluating diffusion-guided UDA by the downstream usefulness of the generated dataset. The goal is not only to generate plausible target-style images, but to construct synthetic data that is useful for downstream target-domain classification. We therefore focus on the *utility* of the generated dataset for adaptation. In particular, when the conditioning interface is weak, increasing the generation budget alone may not necessarily improve downstream performance. VT-DUDA is designed to increase conditioning specificity within each class, with the goal of improving the downstream usefulness of synthetic target-style data under limited generation budgets.

The tokenized design also has two practical properties. First, because visual guidance is represented explicitly as a sequence of tokens, it permits inference-time manipulation through token subset selection and token-strength scaling. Second, the proposed conditioning interface remains compatible with inversion-based translation procedures such as DDIM inversion Song et al. (2020a); Mokady et al. (2023); Meng et al. (2021), allowing token-conditioned synthesis and translation-based augmentation to be used within the same framework.

Our contributions are summarized as follows:

- We propose **VT-DUDA**, a visual-token conditioning framework for diffusion-guided unsupervised domain adaptation that addresses insufficient instance-level guidance in existing diffusion-based UDA methods. Our method augments the standard cross-attention interface of an adapter-based latent diffusion model with instance-dependent visual tokens, enriching source-side class prompts and target-side generic prompts without modifying the diffusion backbone, VAE, or denoising objective.
- Within this framework, we study two token-conditioned routes for constructing synthetic target-style training data for UDA: pure-noise target-style synthesis and DDIM-inversion-based target-style translation. Both routes use the same visual-token conditioning interface, providing a unified way to generate labeled target-style samples from source-domain supervision under different generation protocols.
- Experiments on Office-31, Office-Home, and VisDA-2017 show that VT-DUDA improves average downstream target-domain accuracy over strong discriminative and diffusion-based baselines. We

also observe that the tokenized conditioning interface supports inference-time token manipulation, and that combining pure-noise synthesis with inversion-based translation can provide additional gains in our setting.

2 Related Work

Unsupervised domain adaptation. Unsupervised domain adaptation (UDA) seeks to transfer knowledge from a labeled source domain to an unlabeled target domain under distribution shift. A large body of prior work addresses this problem in feature space through discrepancy minimization Li et al. (2020); Long et al. (2015; 2018), adversarial alignment Ganin et al. (2016); Saito et al. (2018), or prediction- and entropy-based regularization Xu et al. (2019); Rangwani et al. (2022); Zhang et al. (2023b; 2019); Xia et al. (2022); Wu et al. (2025); Jin et al. (2020). These approaches typically aim to learn domain-invariant representations for a fixed discriminative model. In contrast, generative approaches revisit UDA from a data-centric perspective, where adaptation is achieved by constructing target-style training data and then learning a classifier from the resulting synthetic domain.

Diffusion-based adaptation under domain shift. Diffusion models have recently been explored as tools for UDA and related cross-domain learning problems Zhuang et al. (2024); Zhang et al. (2025); Rombach et al. (2022). Rather than aligning source and target features directly, these methods leverage the generative capacity of latent diffusion models to synthesize target-style images or translate source images toward the target domain, after which downstream supervised learning is performed on the generated data. This line of work benefits from two properties of modern diffusion systems: pretrained generative priors for modeling appearance changes across domains Ho et al. (2020); Nichol & Dhariwal (2021); Song et al. (2020b); Dhariwal & Nichol (2021), and parameter-efficient adaptation through mechanisms such as adapters and LoRA Hu et al. (2022); Houlsby et al. (2019); Karimi Mahabadi et al. (2021); Chen et al. (2022). Our work belongs to this family of diffusion-guided UDA methods, but focuses on a different design axis: the conditioning interface used to construct synthetic supervision.

Conditioning in diffusion models. An important property of latent diffusion models is their flexible conditioning interface. Through cross-attention, the denoiser can incorporate signals such as text prompts, learned embeddings, or auxiliary context tokens Nichol et al. (2022); Podell et al. (2024); Rombach et al. (2022); Vaswani et al. (2017); Zhang et al. (2023a); Ye et al. (2023); Hertz et al. (2022). Existing conditioning mechanisms are primarily developed for general-purpose generation, editing, or style control, where the main goals are controllability, personalization, or fidelity Garibi et al. (2025); Gal et al. (2023); Ruiz et al. (2023); Shah et al. (2024); Kumari et al. (2023); Zhang et al. (2023a); Ye et al. (2023); Hertz et al. (2022); Mokady et al. (2023); Brooks et al. (2023); Wei et al. (2023). In diffusion-guided UDA, however, the objective is different: the generated images should serve as *useful labeled training data* for downstream adaptation. Conditioning quality is therefore task-dependent. A conditioning mechanism that is useful for generic generation may not be the one that yields the most useful synthetic target-style samples for downstream adaptation.

Motivated by this distinction, we study how visual tokens can enrich the conditioning signal in diffusion-guided UDA. VT-DUDA introduces a compact visual-token representation and injects it, together with text embeddings, into the standard cross-attention pipeline of a latent diffusion model. Notably, the same token-augmented conditioning interface is used at both fine-tuning time and generation time. During fine-tuning, visual tokens enrich the conditioning seen by both the source and target branches beyond text-only prompts. During generation, visual tokens extracted from labeled source images are paired with class prompts to provide additional instance-level guidance for target-style synthesis. This train-generation consistency is a design choice of VT-DUDA, keeping the conditioning format aligned between training and generation. The resulting design is easy to integrate into adapter-based diffusion adaptation frameworks and allows inference-time manipulation through token selection and token-strength scaling.

3 Methodology

3.1 Problem Setup

We consider the standard UDA setting with a labeled source domain and an unlabeled target domain. Let $\mathcal{D}_{\mathrm{s}} = \{(x_i^{\mathrm{s}}, y_i^{\mathrm{s}})\}_{i=1}^{N_{\mathrm{s}}}$ denote the source dataset, where $x_i^{\mathrm{s}} \in \mathbb{R}^{H\times W\times 3}$ is an RGB image and $y_i^{\mathrm{s}} \in \{1,\dots,C\}$ is its class label over C shared categories. Let $\mathcal{D}_{\mathrm{t}} = \{x_j^{\mathrm{t}}\}_{j=1}^{N_{\mathrm{t}}}$ denote the unlabeled target dataset, drawn from a different image distribution but sharing the same label space. Equivalently, the two domains can be viewed as samples from distributions $\mathbb{P}_{\mathrm{s}}(X,Y)$ and $\mathbb{P}_{\mathrm{t}}(X,Y)$ defined on a common label space, where target labels are unavailable during training. Our goal is to learn a classifier that performs well on the target domain.

We adopt a generation-based view of UDA. Rather than aligning source and target features directly, we first adapt a conditional diffusion model using labeled source images and unlabeled target images, then synthesize a labeled target-style training set, and finally train a discriminative classifier on the resulting data. In this pipeline, the diffusion model is used to generate target-style samples conditioned on labeled source inputs, while the downstream UDA objective is applied at the classifier-training stage.

3.2 Preliminaries: Adapter-based Latent Diffusion

LoRA adaptation. Following LoRA Hu et al. (2022), given a frozen weight matrix $W_0 \in \mathbb{R}^{m\times n}$, LoRA parameterizes a low-rank update

$$\Delta W = W_{\mathrm{up}} W_{\mathrm{down}}, \tag{1}$$

where $W_{\mathrm{up}} \in \mathbb{R}^{m\times r}$, $W_{\mathrm{down}} \in \mathbb{R}^{r\times n}$, and $r \ll \min(m,n)$. For an input feature $x \in \mathbb{R}^n$, the adapted forward pass is

$$y = W_0 x + \lambda \Delta W x = W_0 x + \lambda W_{\mathrm{up}} W_{\mathrm{down}} x, \tag{2}$$

where $\lambda > 0$ is a scaling factor.

Latent diffusion objective. Let E_{VAE} and D_{VAE} denote the encoder and decoder of a pretrained VAE Kingma & Welling (2013), both frozen throughout training. A training image x_0 is mapped to a latent code $z_0 = E_{\mathrm{VAE}}(x_0) \in \mathbb{R}^{h\times w\times d}$. Given a diffusion timestep $\tau \in \{1,\dots,T\}$, the forward noising process is

$$z_\tau = \sqrt{\bar{\alpha}_\tau}\, z_0 + \sqrt{1-\bar{\alpha}_\tau}\, \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \tag{3}$$

where $\bar{\alpha}_\tau = \prod_{k=1}^{\tau}(1-\beta_k)$ under a variance schedule $\{\beta_k\}_{k=1}^{T}$.

Let $\epsilon_\theta(\cdot)$ denote the U-Net noise predictor with parameters θ. Given a text prompt p, a frozen text encoder produces a token sequence $e(p)$, which is passed to the U-Net through cross-attention. Standard adapter-based fine-tuning minimizes the denoising objective Ho et al. (2020); Nichol & Dhariwal (2021); Song et al. (2020b); Song & Ermon (2020)

$$\mathcal{L}_{\mathrm{diff}}(\Delta\theta) = \mathbb{E}_{z_0,\tau,\epsilon,p}\Big[\|\epsilon - \epsilon_{\theta_0+\Delta\theta}(z_\tau, \tau, e(p))\|_2^2\Big], \tag{4}$$

where θ_0 denotes frozen backbone parameters and $\Delta\theta$ denotes trainable adapter parameters.

Domain-specific adapter branches. To model source- and target-domain appearance within a shared diffusion backbone, we use two domain-specific adapter branches indexed by a binary variable $\sigma \in \{0,1\}$, where $\sigma = 0$ denotes the source branch and $\sigma = 1$ denotes the target branch. For each adaptable weight matrix, we maintain a pair of low-rank updates $\Delta W^{(0)}$, $\Delta W^{(1)}$, and the active adapted weight is

$$W^{(\sigma)} = W_0 + \lambda \Delta W^{(\sigma)}. \tag{5}$$

We use $\Delta\theta = \{\Delta\theta^{(0)}, \Delta\theta^{(1)}\}$ to denote the collection of all domain-specific adapter parameters. The same domain indicator σ is used consistently during diffusion training and sample generation.

3.3 Visual Token Conditioning

We introduce a visual-token conditioning interface that augments text prompts and domain control with additional instance-specific visual features.

At a high level, VT-DUDA consists of three stages. First, we jointly train a domain-adaptive diffusion model and an image-to-token encoder using labeled source images and unlabeled target images. Second, for each labeled source instance, we extract visual tokens from the source image and combine them with the class prompt to synthesize labeled target-style samples under the target branch of the diffusion model. Third, we train a downstream classifier on the resulting synthetic target-style dataset, optionally together with an unsupervised target regularizer. The same tokenized conditioning interface can also be used for inference-time token manipulation and for inversion-based translation.

3.3.1 Image-to-Token Encoder

We introduce an image-conditioned encoder G_ψ that maps an input image $v \in \mathbb{R}^{H\times W\times 3}$ to a sequence of M visual tokens,

$$U = G_\psi(v) = [u_1; \dots; u_M] \in \mathbb{R}^{M\times d_c}, \qquad u_m \in \mathbb{R}^{d_c}, \tag{6}$$

where d_c denotes the cross-attention context dimension of the diffusion model.

G_ψ first extracts a global visual representation from the input image using a ResNet-based backbone. This representation is then mapped to an M-dimensional latent code $a(v) \in \mathbb{R}^M$, and each scalar component $a_m(v)$ is transformed into a d_c-dimensional visual token through a token-specific mapping g_m:

$$u_m = g_m\big(a_m(v)\big), \qquad m = 1, \dots, M. \tag{7}$$

The resulting token sequence U is concatenated with text embeddings and used as additional conditioning in cross-attention.

This parameterization converts image-level information into a fixed number of conditioning tokens. The resulting token sequence U is concatenated with text embeddings and used as additional conditioning in cross-attention. The encoder is trained jointly with the diffusion model using only the denoising objective, without auxiliary token-level supervision.

$$U_i^{\mathrm{s}} = G_\psi(x_i^{\mathrm{s}}), \qquad U_j^{\mathrm{t}} = G_\psi(x_j^{\mathrm{t}}). \tag{8}$$

3.3.2 Hybrid Cross-Attention Context

Let $e(p) \in \mathbb{R}^{M_p\times d_c}$ denote the token-wise embedding sequence of a text prompt p. For a labeled source sample $(x_i^{\mathrm{s}}, y_i^{\mathrm{s}})$, we use a class prompt $p_{\mathrm{s}}(y_i^{\mathrm{s}})$, e.g., "A [class]". For a target sample x_j^{t}, we use a generic target prompt p_{t}, e.g., "An image".

We write

$$R_i^{\mathrm{s}} = e\big(p_{\mathrm{s}}(y_i^{\mathrm{s}})\big) \in \mathbb{R}^{M_p^{\mathrm{s}}\times d_c}, \qquad R_j^{\mathrm{t}} = e(p_{\mathrm{t}}) \in \mathbb{R}^{M_p^{\mathrm{t}}\times d_c}. \tag{9}$$

The source and target conditioning contexts are then formed by concatenating text tokens and visual tokens along the sequence dimension:

$$\begin{aligned} S_i^{\mathrm{s}} &= \big[R_i^{\mathrm{s}}\,;\, \kappa_{\mathrm{s}} U_i^{\mathrm{s}}\big] \in \mathbb{R}^{(M_p^{\mathrm{s}}+M)\times d_c}, \\ S_j^{\mathrm{t}} &= \big[R_j^{\mathrm{t}}\,;\, \kappa_{\mathrm{t}} U_j^{\mathrm{t}}\big] \in \mathbb{R}^{(M_p^{\mathrm{t}}+M)\times d_c}, \end{aligned} \tag{10}$$

where $\kappa_{\mathrm{s}}, \kappa_{\mathrm{t}} > 0$ control the relative strength of visual-token guidance in the source and target branches, respectively.

This design leaves the denoiser architecture and diffusion objective unchanged, and modifies only the conditioning sequence passed to cross-attention. On the source side, the visual tokens make the conditioning depend on the current labeled sample in addition to its class prompt. On the target side, the generic prompt provides limited class information; adding target-image tokens allows the target branch to be trained under the same token-augmented conditioning format used later during generation.

3.3.3 Joint Diffusion Training

We jointly optimize the domain-specific adapter parameters $\Delta\theta$ and the token encoder parameters ψ. For each labeled source sample $(x_i^{\mathrm{s}}, y_i^{\mathrm{s}}) \sim \mathcal{D}_{\mathrm{s}}$, we construct the source conditioning context S_i^{s} using the class prompt $p_{\mathrm{s}}(y_i^{\mathrm{s}})$. For each unlabeled target image $x_j^{\mathrm{t}} \sim \mathcal{D}_{\mathrm{t}}$, we construct the target conditioning context S_j^{t} using the generic prompt p_{t}.

Let

$$z_{i,0}^{\mathrm{s}} = E_{\mathrm{VAE}}(x_i^{\mathrm{s}}), \qquad z_{j,0}^{\mathrm{t}} = E_{\mathrm{VAE}}(x_j^{\mathrm{t}}) \tag{11}$$

denote the corresponding latent codes, and let $z_{i,\tau}^{\mathrm{s}}$ and $z_{j,\tau}^{\mathrm{t}}$ be their noised versions obtained through Equation 3. We denote by

$$\epsilon_{\theta_0+\Delta\theta}(z, \tau, S, \sigma) \tag{12}$$

the conditioned noise predictor, where S is the hybrid conditioning context and $\sigma \in \{0, 1\}$ selects the active domain-specific adapter branch.

The joint diffusion training objective is

$$\begin{aligned}\mathcal{L}_{\mathrm{train}}(\Delta\theta, \psi) = \lambda_{\mathrm{s}}\, \mathbb{E}_{(x_i^{\mathrm{s}}, y_i^{\mathrm{s}})\sim\mathcal{D}_{\mathrm{s}},\ \tau,\ \epsilon}\Big[\big\|\epsilon - \epsilon_{\theta_0+\Delta\theta}(z_{i,\tau}^{\mathrm{s}}, \tau, S_i^{\mathrm{s}}, 0)\big\|_2^2\Big] \\ + \lambda_{\mathrm{t}}\, \mathbb{E}_{x_j^{\mathrm{t}}\sim\mathcal{D}_{\mathrm{t}},\ \tau,\ \epsilon}\Big[\big\|\epsilon - \epsilon_{\theta_0+\Delta\theta}(z_{j,\tau}^{\mathrm{t}}, \tau, S_j^{\mathrm{t}}, 1)\big\|_2^2\Big],\end{aligned} \tag{13}$$

where $\lambda_{\mathrm{s}}, \lambda_{\mathrm{t}} > 0$ balance the source and target denoising terms. In our default setting, source and target mini-batches are sampled with equal frequency and we use $\lambda_{\mathrm{s}} = \lambda_{\mathrm{t}} = 1$. Unless otherwise stated, τ is sampled uniformly from $\{1, \ldots, T\}$ and $\epsilon \sim \mathcal{N}(0, I)$.

The first term optimizes denoising on labeled source samples under class-conditioned, exemplar-dependent inputs, while the second term optimizes the target branch on unlabeled target samples under the same token-augmented interface. The token encoder is trained through the same denoising objective as the diffusion adapters, without additional alignment, reconstruction, or token-level supervision. The overall training and generation pipeline of VT-DUDA is illustrated in Fig. 1.

3.4 Generation-based UDA with Token-conditioned Synthesis

After training the token-conditioned diffusion model, we construct a synthetic labeled target-style dataset

$$\widetilde{\mathcal{D}}_{\mathrm{t}} = \{(\tilde{x}_k^{\mathrm{t}}, \tilde{y}_k)\}_{k=1}^{N_{\mathrm{syn}}}. \tag{14}$$

This synthetic dataset is constructed by assigning source labels to target-style samples generated from source-conditioned inputs.

Target-style synthesis from Gaussian noise. Let $z_T \sim \mathcal{N}(0, I)$ denote the initial latent noise, and let $\mathcal{G}(z_T; S, \sigma)$ denote a fixed diffusion sampler that maps an initial noisy latent, a conditioning context S, and a domain indicator σ to a final latent prediction $\tilde{z}_0$. To synthesize a labeled target-style sample, we first draw a labeled source instance $(x_i^{\mathrm{s}}, y_i^{\mathrm{s}}) \sim \mathcal{D}_{\mathrm{s}}$, extract its visual tokens $U_i^{\mathrm{s}} = G_\psi(x_i^{\mathrm{s}})$, and construct the target-style conditioning context with a generation-time token-strength coefficient $\eta_{\mathrm{gen}} > 0$:

$$S^{\mathrm{t}}(y_i^{\mathrm{s}}; i) = \Big[e\big(p_{\mathrm{s}}(y_i^{\mathrm{s}})\big)\,;\, \eta_{\mathrm{gen}} U_i^{\mathrm{s}}\Big]. \tag{15}$$

Unless otherwise stated, we use $\eta_{\mathrm{gen}} = \kappa_{\mathrm{t}}$, and generate a target-style latent under the target branch using this conditioning context:

$$\tilde{z}_0^{\mathrm{t}} = \mathcal{G}\big(z_T; S^{\mathrm{t}}(y_i^{\mathrm{s}}; i), \sigma = 1\big), \qquad z_T \sim \mathcal{N}(0, I), \tag{16}$$

and decode it into image space as $\tilde{x}^{\mathrm{t}} = D_{\mathrm{VAE}}(\tilde{z}_0^{\mathrm{t}})$.

We assign the synthetic label $\tilde{y}^{\mathrm{t}} = y_i^{\mathrm{s}}$ by inheriting the source label associated with the conditioning source image. As in prior generation-based UDA pipelines, this design assumes that the generated sample retains sufficient class information from the source-conditioned input to be used as a labeled training example.

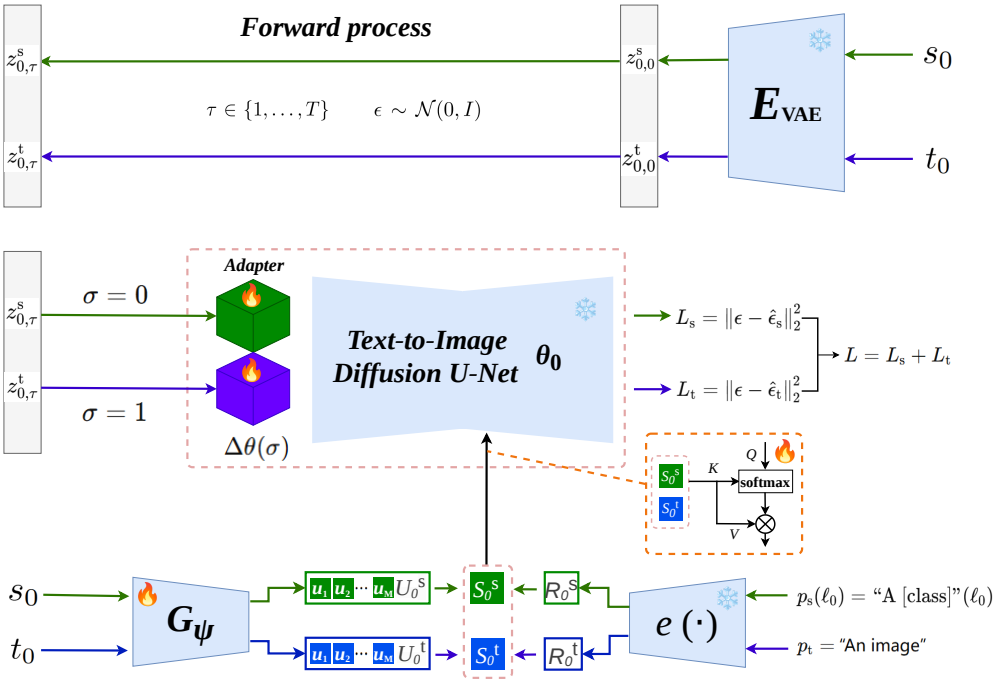

Figure 1: Overview of VT-DUDA. We first jointly train domain-specific diffusion adapters and an image-to-token encoder on source and target images. We then extract visual tokens from labeled source images, combine them with class prompts, and synthesize labeled target-style samples under the target branch of the diffusion model. Finally, the resulting synthetic target-style dataset is used to train a downstream UDA classifier. The same tokenized conditioning interface can also be used for inference-time manipulation and translation-based augmentation.

Downstream UDA training. Let f_ϕ denote the downstream classifier, with class posterior $p_\phi(y \mid x)$. Once the synthetic dataset $\widetilde{\mathcal{D}}_{\mathrm{t}}$ has been constructed, we train f_ϕ using supervised learning on $\widetilde{\mathcal{D}}_{\mathrm{t}}$, optionally combined with an unsupervised target regularizer on $\mathcal{D}_{\mathrm{t}}$:

$$\mathcal{L}_{\mathrm{uda}}(\phi) = \mathcal{L}_{\mathrm{sup}}\big(\widetilde{\mathcal{D}}_{\mathrm{t}}; \phi\big) + \lambda_{\mathrm{u}}\, \mathcal{L}_{\mathrm{unsup}}\big(\mathcal{D}_{\mathrm{t}}; \phi\big), \tag{17}$$

where

$$\mathcal{L}_{\mathrm{sup}}\big(\widetilde{\mathcal{D}}_{\mathrm{t}}; \phi\big) = \mathbb{E}_{(\tilde{x},\tilde{y})\sim\widetilde{\mathcal{D}}_{\mathrm{t}}}\Big[-\log p_\phi(\tilde{y} \mid \tilde{x})\Big]. \tag{18}$$

The unsupervised term $\mathcal{L}_{\mathrm{unsup}}$ is instantiated by the chosen downstream UDA method, e.g., MCC or ELS in our experiments. VT-DUDA therefore changes the synthetic data construction stage while remaining compatible with different classifier-side UDA objectives.

Translation-based augmentation. The same visual-token interface is compatible with inversion-based translation procedures. Let $z^{\mathrm{s}}_{i,0} = E_{\mathrm{VAE}}(x^{\mathrm{s}}_i)$ be the source latent, and let $\mathcal{I}$ denote an inversion operator that maps this latent to an approximate terminal latent

$$z^{\mathrm{inv}}_{i,T} = \mathcal{I}(z^{\mathrm{s}}_{i,0}). \tag{19}$$

A translated target-style sample can then be obtained by running the sampler under the target branch using the same generation-time conditioning context $S^{\mathrm{t}}(y^{\mathrm{s}}_i; i)$:

$$\tilde{z}^{\mathrm{inv}\to\mathrm{t}}_{i,0} = \mathcal{G}\big(z^{\mathrm{inv}}_{i,T}; S^{\mathrm{t}}(y^{\mathrm{s}}_i; i), \sigma = 1\big), \qquad \tilde{x}^{\mathrm{inv}\to\mathrm{t}}_i = D_{\mathrm{VAE}}\big(\tilde{z}^{\mathrm{inv}\to\mathrm{t}}_{i,0}\big). \tag{20}$$

Such translated samples can be incorporated as an additional source of synthetic training data. In VT-DUDA, however, the conditioning mechanism itself remains unchanged: both Gaussian-noise synthesis and translation-based augmentation are driven by the same token-conditioned interface.

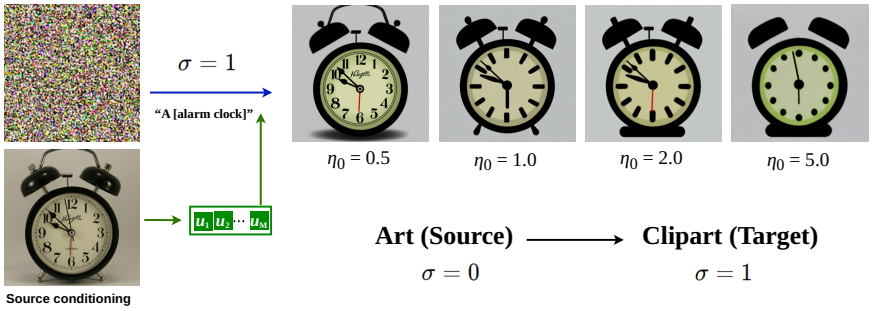

Figure 2: Token-strength scaling for target-style synthesis (Art→Clipart). For each labeled source instance, we extract visual tokens, combine them with the class prompt, and generate a target-style sample under the target branch from Gaussian noise. The figure illustrates the effect of varying the token strength $\eta \in \{0.5, 1.0, 2.0, 5.0\}$ on the generated samples.

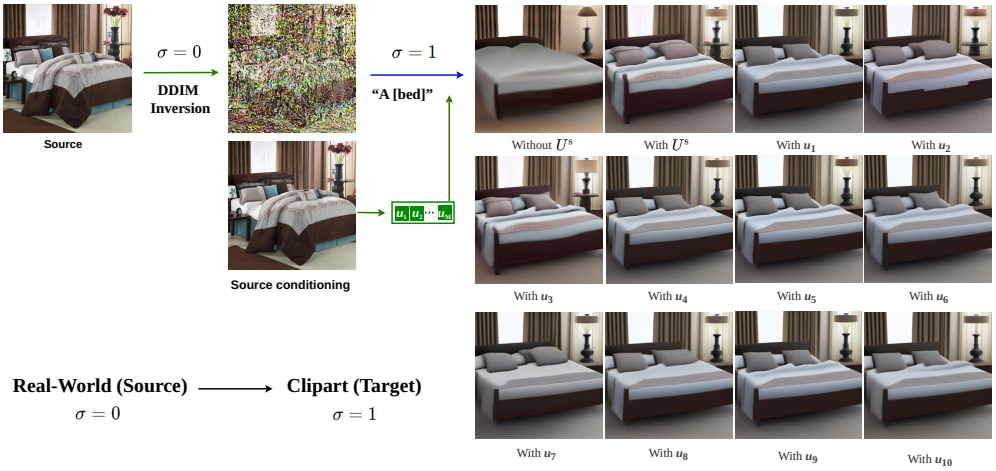

Figure 3: Token-subset manipulation under translation-based augmentation (Real-World→Clipart). Starting from a labeled source image in the Real-World domain, we first obtain an inverted latent and then re-synthesize it under the target branch using different token subsets. The figure shows how the tokenized conditioning interface allows explicit modification of the visual-token context during target-style translation.

3.5 Inference-time Token Manipulation

Because visual guidance is represented as an explicit token sequence, the conditioning context can be modified directly at inference time without retraining. Given a prompt p and a reference image v, we extract visual tokens

$$U = G_\psi(v) \in \mathbb{R}^{M \times d_c}. \tag{21}$$

Let $\Omega \subseteq \{1, \ldots, M\}$ be a selected token index set and let $\eta > 0$ be a token-strength coefficient. We define the manipulated conditioning context

$$S_{\text{gen}} = \Big[e(p)\,;\, \eta\, U_\Omega\Big], \qquad U_\Omega \in \mathbb{R}^{|\Omega| \times d_c}. \tag{22}$$

These operations require no retraining and preserve the original diffusion backbone and adapter parameters. They do not assume that individual visual tokens are semantically disentangled or directly interpretable. Rather, they provide a direct way to modify the composition and scale of the visual-token context passed to cross-attention. Figure 2 visualizes token-strength scaling under pure-noise target-style synthesis, while Fig. 3 illustrates token-subset manipulation in the translation-based setting.

4 Experiments

4.1 Experimental Setup

We evaluate VT-DUDA on three standard UDA benchmarks: Office-31 Saenko et al. (2010), Office-Home Venkateswara et al. (2017), and VisDA-2017 Peng et al. (2017). Office-31 contains 31 categories across three domains: Amazon (A), Webcam (W), and DSLR (D). Office-Home contains 65 categories across four domains: Art (Ar), Clipart (Cl), Product (Pr), and Real-World (Rw). VisDA-2017 contains a large synthetic-to-real shift with 12 categories. We follow the standard splits and report target-domain classification accuracy (%).

We compare VT-DUDA with representative discriminative UDA baselines and diffusion-based data-augmentation baselines. In particular, we instantiate VT-DUDA on top of MCC Jin et al. (2020) and ELS Zhang et al. (2023b), and compare against both the original discriminative methods and diffusion-based baselines including Terra Zhuang et al. (2024) and DCDM Zhang et al. (2025). For VT-DUDA, the *full configuration* combines pure-noise target-style synthesis with inversion-based translation. Importantly, in our method these are two data-construction routes under the same token-conditioned generation framework: both use the proposed visual-token interface, and the translation route also conditions on visual tokens extracted from the source image being inverted. We additionally report an *inversion-free* protocol in which the synthetic labeled training set is constructed only from pure-noise target-style generation. This lighter setting is used to examine how far the proposed conditioning mechanism remains effective without the extra translation route.

Unless otherwise stated, VT-DUDA generates 50 images per class on Office-31 and Office-Home, and 1000 images per class on VisDA-2017. For Terra, we follow the same synthetic-data budget in our comparisons. For DCDM, we report the results under its published generation setting, which uses a larger synthetic-data budget: 200 generated images per class on Office-31 and Office-Home, and 2000 generated images per class on VisDA-2017. The downstream classifier uses ResNet-50 on Office-31/Office-Home and ResNet-101 on VisDA-2017. The image-to-token encoder G_ψ is implemented with a ResNet-18 backbone He et al. (2016). We set the token count to $M = 10$ in the default configuration. All results are averaged over three runs.

Our empirical objective is practical: we evaluate whether the proposed conditioning design improves the downstream usefulness of synthetic target-style data for UDA. We therefore treat the target-domain classification accuracy obtained from the constructed synthetic set as the primary criterion throughout this section. Additional implementation details, evaluation protocols, and supplementary results are provided in the Appendix A.

4.2 Main Results

Tables 1 and 2 summarize the main UDA results under the full configuration described above. Across all three benchmarks, VT-DUDA achieves the strongest average performance when instantiated on top of MCC and ELS.

A first observation is that VT-DUDA consistently improves over the underlying discriminative UDA baselines. Relative to MCC, VT-DUDA improves the average accuracy by +4.34% on Office-Home (76.58 vs. 72.24), +2.62% on Office-31 (92.23 vs. 89.61), and +4.16% on VisDA-2017 (87.48 vs. 83.32). Relative to ELS, the gains are +5.50% on Office-Home (77.34 vs. 71.84), +2.63% on Office-31 (92.84 vs. 90.21), and +5.02% on VisDA-2017 (88.42 vs. 83.40). These margins indicate that the proposed token-conditioned synthetic-data construction substantially strengthens both downstream UDA objectives.

A second observation is that VT-DUDA also improves over strong diffusion-based data-augmentation baselines. Under the full setting, the best non-VT diffusion results are 76.26 on Office-Home (ELS+Terra), 91.80 on Office-31 (ELS+DCDM), and 86.86 on VisDA-2017 (ELS+Terra). The corresponding best VT-DUDA results are 77.34, 92.84, and 88.42, yielding gains of +1.08%, +1.04%, and +1.56%, respectively. This shows that the proposed conditioning design remains effective in the strongest practical generation setting and yields improvements over existing diffusion-based augmentation pipelines across all three benchmarks.

Table 1: Transfer accuracies (%) on Office-Home under the UDA setting. Results are averaged over 12 transfer tasks (Avg). The best result in each column is highlighted in **bold**.

Method	**Ar→Cl**	**Ar→Pr**	**Ar→Rw**	**Cl→Ar**	**Cl→Pr**	**Cl→Rw**	**Pr→Ar**	**Pr→Cl**	**Pr→Rw**	**Rw→Ar**	**Rw→Cl**	**Rw→Pr**	**Avg**
ERM Vapnik (2013)	44.06	67.12	74.26	53.26	61.96	64.54	51.91	38.90	72.94	64.51	43.84	75.39	59.39
DANN Ganin et al. (2016)	52.53	62.57	73.20	56.89	67.02	68.34	58.37	54.14	78.31	70.78	60.76	80.57	65.29
CDAN Long et al. (2018)	54.21	72.18	78.29	61.97	71.43	72.39	62.96	55.68	80.68	74.71	61.22	83.68	69.12
AFN Xu et al. (2019)	52.58	72.42	76.96	64.90	71.14	72.91	64.08	51.29	77.83	72.21	57.46	82.09	67.99
MDD Zhang et al. (2019)	56.37	75.53	79.17	62.95	73.21	73.55	62.56	54.86	79.49	73.84	61.45	84.06	69.75
SDAT Rangwani et al. (2022)	58.20	77.46	81.35	66.06	76.45	76.41	63.70	56.69	82.49	76.02	62.09	85.24	71.85
MSGD Xia et al. (2022)	58.70	76.90	78.90	70.10	76.20	76.60	69.00	57.20	82.30	74.90	62.70	84.50	72.40
MCC Jin et al. (2020)	56.83	79.81	82.66	67.80	77.02	77.82	66.98	55.43	81.79	73.95	61.41	85.44	72.24
ELS Zhang et al. (2023b)	57.79	77.65	81.62	66.59	76.74	76.43	62.69	56.69	82.12	75.63	62.85	85.35	71.84
Diffusion-based methods													
MCC+Terra Zhuang et al. (2024)	63.49	81.51	83.46	72.52	82.89	81.25	73.20	61.66	83.16	74.36	63.45	84.41	75.45
ELS+Terra Zhuang et al. (2024)	64.62	82.33	83.60	71.19	84.25	80.31	73.00	63.57	83.81	76.20	**66.56**	85.70	76.26
MCC+DCDM Zhang et al. (2025)	58.23	80.33	82.91	70.14	79.15	81.36	68.49	57.75	83.44	74.18	63.81	85.67	73.71
ELS+DCDM Zhang et al. (2025)	60.35	78.81	82.74	69.59	80.53	79.55	65.16	58.26	83.11	75.81	64.18	85.55	73.64
MCC+VT-DUDA	**65.49**	83.39	**85.36**	**74.38**	81.88	**83.01**	**75.06**	63.44	82.10	76.25	65.30	83.30	76.58
ELS+VT-DUDA	63.26	**83.74**	85.03	72.63	**85.67**	78.89	71.55	**64.91**	**85.29**	**77.58**	65.17	**87.16**	**77.34**

Table 2: Transfer accuracies (%) on Office-31 under the UDA setting, with VisDA-2017 mean accuracy (%) reported in the last column. The best result in each column is highlighted in **bold**.

Dataset	**Office-31**							**VisDA-2017**
Method	**A→W**	**D→W**	**W→D**	**A→D**	**D→A**	**W→A**	**Avg**	**mean**
ERM Vapnik (2013)	77.07	96.60	99.20	81.08	64.11	64.01	80.35	51.47
DANN Ganin et al. (2016)	89.85	97.95	99.90	83.26	73.28	73.75	86.33	79.02
CDAN Long et al. (2018)	92.42	98.62	**100.00**	91.44	74.61	72.80	88.32	80.74
AFN Xu et al. (2019)	91.82	98.77	**100.00**	95.12	72.43	70.71	88.14	74.64
MDD Zhang et al. (2019)	93.55	98.66	**100.00**	93.92	75.29	73.95	89.23	81.10
SDAT Rangwani et al. (2022)	91.32	98.83	**100.00**	95.25	76.97	73.19	89.26	83.23
MSGD Xia et al. (2022)	95.50	99.20	**100.00**	95.60	77.30	77.00	90.80	84.60
MCC Jin et al. (2020)	94.09	98.32	99.67	94.25	75.89	75.46	89.61	83.32
ELS Zhang et al. (2023b)	93.84	98.78	**100.00**	95.78	77.72	75.13	90.21	83.40
Diffusion-based methods								
MCC+Terra Zhuang et al. (2024)	94.55	99.03	**100.00**	96.46	78.64	79.37	91.34	85.39
ELS+Terra Zhuang et al. (2024)	94.09	**99.21**	**100.00**	96.25	78.67	79.45	91.28	86.86
MCC+DCDM Zhang et al. (2025)	95.51	98.58	99.93	95.31	78.26	78.43	91.01	86.56
ELS+DCDM Zhang et al. (2025)	96.90	98.91	**100.00**	97.46	**79.79**	77.74	91.80	86.29
MCC+VT-DUDA	96.40	98.98	**100.00**	98.34	78.53	81.13	92.23	87.48
ELS+VT-DUDA	**98.36**	97.77	**100.00**	**99.46**	78.11	**83.34**	**92.84**	**88.42**

These full-setting results should be interpreted as the performance of the complete VT-DUDA pipeline. In our method, pure-noise synthesis and inversion-based translation are not two unrelated components; both are driven by the same visual-token-conditioned interface. The main results therefore evaluate VT-DUDA in its intended full form. To further examine the role of the conditioning design under a lighter generation protocol, we report an inversion-free variant in Sec. 4.3.

4.3 Inversion-free VT-DUDA and Budget-aware Comparison

Table 3 reports the inversion-free variant of VT-DUDA, where the synthetic labeled training set is constructed only from pure-noise target-style generation. This setting is useful for two reasons. First, it shows whether the proposed token-conditioned interface remains effective without the extra translation route. Second, it provides a direct way to examine data efficiency, since VT-DUDA uses only 50 generated images per class on Office-31 and Office-Home, and 1000 on VisDA-2017.

Table 3: Average accuracies (%) under the inversion-free protocol. VT-DUDA and Terra use only pure-noise target-style generation, with 50 generated images per class on Office-31 and Office-Home and 1000 on VisDA-2017. DCDM is included for reference under its published larger-budget setting (200 per class on Office-31/Office-Home and 2000 on VisDA-2017).

Method	**Office-Home Avg**	**Office-31 Avg**	**VisDA mean**
MCC Jin et al. (2020)	72.24	89.61	83.32
ELS Zhang et al. (2023b)	71.84	90.21	83.40
MCC+Terra Zhuang et al. (2024)	71.84	91.34	85.39
ELS+Terra Zhuang et al. (2024)	72.18	91.28	86.86
MCC+DCDM Zhang et al. (2025)	73.71	91.01	86.56
ELS+DCDM Zhang et al. (2025)	73.64	91.80	86.29
MCC+VT-DUDA	73.50	91.37	86.92
ELS+VT-DUDA	**73.72**	**91.92**	**87.56**

Even under this lighter protocol, VT-DUDA remains consistently stronger than the underlying discriminative UDA baselines. Under MCC, the gains over the original discriminative baseline are +1.26% on Office-Home, +1.76% on Office-31, and +3.60% on VisDA-2017. Under ELS, the corresponding gains are +1.88%, +1.71%, and +4.16%.

Compared with Terra under the same pure-noise budget, VT-DUDA also performs better on Office-Home. Under MCC, VT-DUDA improves over Terra by +1.66%, while under ELS the corresponding gain is +1.54%. Because Terra and VT-DUDA use the same generation budget in this comparison, this result provides cleaner evidence for the benefit of visual-token conditioning.

The budget-aware comparison is particularly revealing on Office-31 and VisDA-2017. On these two benchmarks, inversion-free VT-DUDA already surpasses the full Terra configuration, even though the latter additionally uses inversion-based translation. Specifically, on Office-31, MCC+VT-DUDA reaches 91.37% versus 91.34% for MCC+Terra, and ELS+VT-DUDA reaches 91.92% versus 91.28% for ELS+Terra. On VisDA-2017, MCC+VT-DUDA reaches 86.92% versus 85.39% for MCC+Terra, and ELS+VT-DUDA reaches 87.56% versus 86.86% for ELS+Terra. In other words, on these two benchmarks, using only 50 generated images per class on Office-31 and 1000 on VisDA-2017 is already sufficient for VT-DUDA to outperform a stronger Terra pipeline that additionally includes inversion-based translation.

A similarly strong pattern appears in the comparison with DCDM, despite DCDM using a much larger synthetic-data budget. On Office-31, inversion-free VT-DUDA outperforms DCDM under both downstream objectives (91.37% vs. 91.01% for MCC and 91.92% vs. 91.80% for ELS) while using 50, rather than 200, generated images per class. On VisDA-2017, VT-DUDA again exceeds DCDM under both objectives (86.92% vs. 86.56% for MCC and 87.56% vs. 86.29% for ELS) while using 1000, rather than 2000, generated images per class. On Office-Home, VT-DUDA remains competitive with DCDM under this smaller budget: ELS+VT-DUDA slightly exceeds ELS+DCDM (73.72% vs. 73.64%), while MCC+VT-DUDA is within 0.21 points of MCC+DCDM (73.50% vs. 73.71%). To visualize how the generated target-style samples relate to the target domain, we present t-SNE plots in Fig. 4. The synthesized samples occupy a feature region that is closer to the unlabeled target domain than the original source-domain samples. Taken together, these results indicate that the proposed token-conditioned interface improves not only final accuracy but also the downstream utility per generated sample.

4.4 Inversion-based Translation within the Same Token-conditioned Framework

The comparison between Tables 1, 2, and 3 shows that inversion-based translation further improves the strongest practical configuration. In VT-DUDA, inversion-based translation is an additional synthetic-data construction route driven by the same visual-token-conditioned generation framework. From this perspective, the inversion-free results and the full results answer two different questions. The inversion-free results show that the proposed conditioning interface is already effective under a lighter pure-noise protocol, while the full results show that it remains strong when combined with an additional inversion-based augmentation

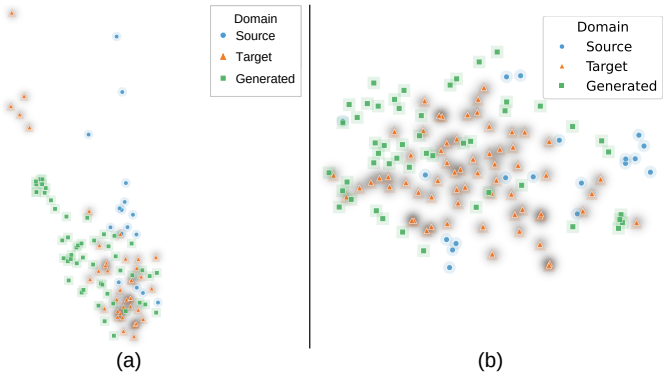

Figure 4: t-SNE visualization on Office-Home under the UDA setting. We compare source-domain features, target-domain features, and features from the generated target-style domain. (a) Ar→Cl for class *Webcam*. (b) Ar→Rw for class *Toys*.

Table 4: Ablation results on Office-Home (% accuracy).

Ablation setting	**Value**	**MCC+VT-DUDA**	**ELS+VT-DUDA**
Ar→Cl: number of generated images per class			
Generated images / class	10	40.18	47.74
Generated images / class	25	50.54	52.83
Generated images / class	50	57.77	59.24
Generated images / class	100	**61.15**	**63.46**
Cl→Rw: number of visual tokens M (50 generated images / class)			
Visual tokens M	5	82.16	**83.44**
Visual tokens M	10	**82.59**	83.30
Visual tokens M	20	81.34	82.53
Visual tokens M	50	79.32	80.10

route. We therefore view inversion-based translation as complementary to token-conditioned synthesis: it can further strengthen the complete pipeline.

4.5 Ablation Studies

We conduct ablations under the inversion-free protocol so that the observed trends can be attributed directly to the token-conditioned generator without adding the extra translation route.

Table 4 shows two representative trends. First, increasing the number of generated images per class consistently improves downstream performance. This is expected because the downstream classifier is trained on synthetic target-style data, and a larger synthetic set provides more supervision. Second, varying the number of visual tokens M shows that a compact token set already works well, whereas overly long token sequences degrade performance. In our experiments, $M = 5$ or $M = 10$ is sufficient to obtain the strongest results, while increasing the token count to 20 or 50 hurts accuracy. We present this as an empirical observation about conditioning efficiency. Appendix A.4 further provides detailed transfer-wise results on Office-Home and Office-31, together with per-class results on VisDA-2017, under the inversion-free protocol.

5 Conclusion

We introduced VT-DUDA, a visual-token conditioning framework for diffusion-guided unsupervised domain adaptation. The method augments the cross-attention context of an adapter-based latent diffusion model with instance-level visual tokens, enriching text-based conditioning without modifying the diffusion backbone or objective. Experiments on Office-31, Office-Home, and VisDA-2017 show that this design improves the downstream utility of synthetic target-style data over strong baselines, including under limited generation budgets. These results suggest that conditioning design matters in generation-based UDA. Evaluating this approach beyond image classification and the diffusion setting considered here remains future work.

References

Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18392–18402, 2023.

Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, and D. Cohen-or. An image is worth one word: Personalizing text-to-image generation using textual inversion. In *The Eleventh International Conference on Learning Representations*, 2023.

Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. March, and V. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

D. Garibi, S. Yadin, R. Paiss, O. Tov, S. Zada, A. Ephrat, T. Michaeli, I. Mosseri, and T. Dekel. Tokenverse: Versatile multi-concept personalization in token modulation space. *ACM Transactions On Graphics (TOG)*, 44(4):1–11, 2025.

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

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Y. Jin, X. Wang, M. Long, and J. Wang. Minimum class confusion for versatile domain adaptation. In *European Conference on Computer Vision*, pp. 464–480. Springer, 2020.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in neural information processing systems*, 34:1022–1035, 2021.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

N. Kumari, B. Zhang, R. Zhang, E. Shechtman, and J.-Y. Zhu. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1931–1941, 2023.

M. Li, Y.-M. Zhai, Y.-W. Luo, P.-F. Ge, and C.-X. Ren. Enhanced transport distance for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13936–13944, 2020.

M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pp. 97–105. PMLR, 2015.

M. Long, Z. Cao, J. Wang, and M. I. Jordan. Conditional adversarial domain adaptation. *Advances in Neural Information Processing Systems*, 31, 2018.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6038–6047, 2023.

A. Q. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. Mcgrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning*, pp. 16784–16804. PMLR, 2022.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PMLR, 2021.

X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017.

D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024.

H. Rangwani, S. K. Aithal, M. Mishra, A. Jain, and V. B. Radhakrishnan. A closer look at smoothness in domain adversarial training. In *International Conference on Machine Learning*, pp. 18378–18399. PMLR, 2022.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22500–22510, 2023.

K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision*, pp. 213–226. Springer, 2010.

K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3723–3732, 2018.

V. Shah, N. Ruiz, F. Cole, E. Lu, S. Lazebnik, Y. Li, and V. Jampani. Ziplora: Any subject in any style by effectively merging loras. In *European Conference on Computer Vision*, pp. 422–438. Springer, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

V. Vapnik. *The nature of statistical learning theory.* Springer science & business media, 2013.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5018–5027, 2017.

Yuxiang Wei, Yabo Zhang, Zhilong Ji, Jinfeng Bai, Lei Zhang, and Wangmeng Zuo. Elite: Encoding visual concepts into textual embeddings for customized text-to-image generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 15943–15953, 2023.

Z.-Z. Wu, C.-J. Du, X.-Q. Wang, L. Zou, F. Cheng, T. Li, F.-D. Nian, T. Weise, and X.-F. Wang. Domain adaptation via feature disentanglement for cross-domain image classification. *Applied Soft Computing*, 172:112868, 2025.

H. Xia, T. Jing, and Z. Ding. Maximum structural generation discrepancy for unsupervised domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3434–3445, 2022.

R. Xu, G. Li, J. Yang, and L. Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1426–1435, 2019.

Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3836–3847, 2023a.

Y. Zhang, T. Liu, M. Long, and M. Jordan. Bridging theory and algorithm for domain adaptation. In *International Conference on Machine Learning*, pp. 7404–7413. PMLR, 2019.

Y. Zhang, X. Wang, J. Liang, Z. Zhang, L. Wang, R. Jin, and T. Tan. Free lunch for domain adversarial training: Environment label smoothing. In *ICLR*, 2023b.

Y. Zhang, S. Chen, W. Jiang, Y. Zhang, J. Lu, and J. T. Kwok. Domain-guided conditional diffusion model for unsupervised domain adaptation. *Neural Networks*, 184:107031, 2025.

Z. Zhuang, Y. Zhang, X. Wang, J. Lu, Y. Wei, and Y. Zhang. Time-varying lora: Towards effective cross-domain fine-tuning of diffusion models. *Advances in Neural Information Processing Systems*, 37: 73920–73951, 2024.

A Appendix

A.1 Implementation Details

We instantiate VT-DUDA on top of Stable Diffusion XL (SDXL) Podell et al. (2024). We use the SDXL VAE, keep the VAE frozen throughout training, and use the same VAE encoder E_{VAE} and decoder D_{VAE} as defined in Sec. 3.2.

Low-rank adaptation. We freeze the pretrained UNet and both SDXL text encoders, and optimize only lightweight adapter parameters together with the image-to-token encoder. In the UNet, LoRA modules are attached to all attention projection matrices, the LoRA rank is set to $r = 32$. To align the implementation with the notation in Equation 5, the effective domain-specific update for an adaptable weight matrix is

$$\Delta W^{(\sigma)} = W_{\mathrm{up}}\big(I + \sigma\Omega\big)^{\top} W_{\mathrm{down}}, \qquad \sigma \in \{0, 1\}, \tag{23}$$

where $\sigma = 0$ denotes the source branch and $\sigma = 1$ denotes the target branch. Hence, $\Delta W^{(0)} = W_{\mathrm{up}} W_{\mathrm{down}}$ and $\Delta W^{(1)} = W_{\mathrm{up}}(I + \Omega)^{\top} W_{\mathrm{down}}$. This realizes the domain-dependent adapters in Equation 5 through a shared low-rank basis $(W_{\mathrm{up}}, W_{\mathrm{down}})$ and a target-activated modulation matrix Ω. In practice, the source and target forward passes are executed sequentially within each iteration using the binary branch indicator σ. No additional LoRA scaling coefficient is used, i.e., the effective scaling in Eq. equation 2 is $\lambda = 1$ in our implementation.

Image-to-token encoder. The image-to-token encoder G_ψ in Equation 6 uses a **ResNet-18** backbone. Specifically, the ResNet-18 backbone extracts a global visual representation from an input image v, which is projected to an M-dimensional latent code $a(v) \in \mathbb{R}^M$ with $M = 10$. Each scalar component $a_m(v)$ is then mapped to a visual token $u_m \in \mathbb{R}^{d_c}$ through an independent token-specific MLP g_m, following Eq. equation 7. In our implementation, these token-specific heads use a $1 \to 64 \to 128 \to d_c$ multilayer perceptron with ELU activations. The cross-attention dimension d_c is set automatically to the concatenated text-conditioning dimension of the two frozen SDXL text encoders. The token encoder is trained jointly with the diffusion adapters using only the denoising objective in equation 13.

Prompt construction. Let $p_{\mathrm{s}}(y)$ and p_{t} denote the source and target prompts introduced in Sec. 3.3.2. In the implementation, the source prompt is instantiated from the source image folder name using a simple class template, e.g., `"A backpack"`, while the target prompt is the generic template $p_{\mathrm{t}} =$ `"An image"`. Thus, source prompts are label-dependent, whereas target prompts are domain-generic.

Fusion of text and visual conditioning. We follow the hybrid context formulation in Equation 10. Text conditioning is obtained from the two frozen SDXL text encoders. Given a prompt p, each text encoder outputs a token sequence, and we use the penultimate hidden state from each encoder. These two token sequences are concatenated along the channel dimension to form the token-wise text embedding $R = e(p) \in \mathbb{R}^{M_p \times d_c}$. For the experiments here, the original size and target size are both 1024×1024, and the crop offset is fixed to $(0, 0)$.

For a source image x_i^{s} and a target image x_j^{t}, we extract visual tokens $U_i^{\mathrm{s}} = G_\psi(x_i^{\mathrm{s}})$ and $U_j^{\mathrm{t}} = G_\psi(x_j^{\mathrm{t}})$, with $M = 10$ tokens per image. We then construct the cross-attention contexts

$$S_i^{\mathrm{s}} = \big[R_i^{\mathrm{s}}; \kappa_{\mathrm{s}} U_i^{\mathrm{s}}\big],$$
$$S_j^{\mathrm{t}} = \big[R_j^{\mathrm{t}}; \kappa_{\mathrm{t}} U_j^{\mathrm{t}}\big],$$

exactly as in Equation 10. In our default setting, $\kappa_{\mathrm{s}} = \kappa_{\mathrm{t}} = 1$. No extra fusion block is added: text tokens and visual tokens are concatenated directly along the sequence dimension and passed to the UNet cross-attention layers.

Training objective and batch composition. We optimize the joint objective in Equation 13 with equal weights $\lambda_{\mathrm{s}} = \lambda_{\mathrm{t}} = 1$. Each optimization step processes one minibatch of paired source and target samples.

Each iteration contains 8 source images and 8 target images. The source and target losses are computed in two sequential passes through the same frozen diffusion backbone with different branch indicators:

$$\mathcal{L}_{\text{train}} = \mathcal{L}_{\text{s}} + \mathcal{L}_{\text{t}}.$$

where $\mathcal{L}_{\text{s}}$ uses $\sigma = 0$ and context S_i^{s}, and $\mathcal{L}_{\text{t}}$ uses $\sigma = 1$ and context S_j^{t}. For both passes, the image is encoded to the latent space with the frozen VAE, a diffusion timestep τ is sampled uniformly, and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ is added using the pretrained DDPM scheduler, following Eq. equation 3. The UNet is then trained to predict either ϵ or the corresponding velocity target depending on the scheduler configuration. In the SDXL setup used here, we follow the scheduler configuration provided by the pretrained model and use the standard mean-squared denoising loss. Importantly, within each iteration, the same sampled τ and the same noise tensor ϵ are reused for the source and target passes, so the two losses are matched under identical corruption conditions.

Optimization and training configuration. We train on a single NVIDIA A100 (80GB) using mixed precision `fp16`. The VAE is always kept in `float32` for numerical stability, and the image-to-token encoder is also optimized in `float32`. The launch configuration uses 8-bit AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and weight decay 10^{-4} for the UNet adapter parameters. The optimizer uses separate parameter groups: the low-rank factors $(W_{\text{up}}, W_{\text{down}})$ are optimized with learning rate 5×10^{-5}, while the domain-modulation parameters Ω are optimized with learning rate 5×10^{-3}, corresponding to `lr_theta`=100. The image-to-token encoder G_ψ is optimized with learning rate 10^{-4} and weight decay 10^{-4}. Only the UNet adapters and G_ψ are updated.

We train for 8000 optimization steps with per-device batch size 8, gradient accumulation 1, and maximum gradient norm 1.0. The learning-rate scheduler is constant with zero warmup. Data loading uses 8 worker processes. The random seed is fixed to 0.

Data preprocessing. All training images are first corrected with EXIF transpose, converted to RGB if necessary, resized to 1024×1024 using bilinear interpolation, and then randomly cropped to 1024×1024. After cropping, each image is converted to a tensor and normalized channel-wise to $[-1, 1]$ using mean $(0.5, 0.5, 0.5)$ and standard deviation $(0.5, 0.5, 0.5)$. No horizontal flipping or additional color augmentation is applied in this training pipeline. The same normalized tensor is used both for VAE encoding and for visual-token extraction by G_ψ.

A.2 Dataset and Evaluation Protocol Details

This appendix consolidates the benchmark setup, evaluation protocol, and the two synthetic-data construction routes used in VT-DUDA. We first describe the datasets and evaluation protocols in text form, and then give the complete algorithms for (i) pure-noise target-style synthesis and (ii) DDIM-inversion-based target-style translation.

A.2.1 Benchmarks and evaluation protocol

We evaluate VT-DUDA on three standard unsupervised domain adaptation benchmarks.

Office-31. Office-31 contains three domains, Amazon (A), Webcam (W), and DSLR (D), with 31 shared object categories. Following the standard UDA protocol, we evaluate all six directed transfer tasks:

$$\mathrm{A}\to\mathrm{W},\ \mathrm{D}\to\mathrm{W},\ \mathrm{W}\to\mathrm{D},\ \mathrm{A}\to\mathrm{D},\ \mathrm{D}\to\mathrm{A},\ \mathrm{W}\to\mathrm{A}.$$

For this benchmark, the downstream classifier uses a ResNet-50 backbone, and we report target-domain classification accuracy for each transfer task together with the average over all six tasks.

Office-Home. Office-Home contains four domains, Art (Ar), Clipart (Cl), Product (Pr), and Real-World (Rw), with 65 shared categories. We follow the standard protocol and evaluate all twelve directed transfer tasks formed by ordered pairs of distinct domains. The downstream classifier again uses a ResNet-50 backbone, and we report target-domain classification accuracy for each task together with the average over all twelve tasks.

VisDA-2017. VisDA-2017 contains a large synthetic-to-real domain shift with 12 categories. We follow the standard training/validation protocol, using the synthetic split as source and the real validation split as target. For this benchmark, the downstream classifier uses a ResNet-101 backbone, and we report the standard target-domain mean accuracy on the real validation split.

Across all benchmarks, the source and target domains share the same label space. Source labels are available only for the source domain, while target labels are not used during training. All reported numbers are averaged over three runs.

A.2.2 Synthetic-data construction protocol

Let $(x_i^{\mathrm{s}}, y_i^{\mathrm{s}}) \in \mathcal{D}_{\mathrm{s}}$ denote a labeled source sample, and let $U_i^{\mathrm{s}} = G_\psi(x_i^{\mathrm{s}})$ be the corresponding visual-token sequence extracted by the image-to-token encoder. During target-style generation, VT-DUDA uses the target-side conditioning context

$$S_i^{\mathrm{t}} = \Big[e\big(p_{\mathrm{s}}(y_i^{\mathrm{s}})\big)\ ;\ \eta_{\mathrm{gen}}\, U_i^{\mathrm{s}}\Big], \tag{24}$$

where $e(\cdot)$ is the text-embedding operator and $\eta_{\mathrm{gen}} > 0$ is the generation-time token-strength coefficient. Unless otherwise stated, we use $\eta_{\mathrm{gen}} = 1$.

VT-DUDA constructs synthetic target-style training data using two routes: (i) inversion-free pure-noise synthesis and (ii) DDIM-inversion-based target-style translation.

Inversion-free protocol. The inversion-free protocol uses only the pure-noise route. Unless otherwise stated, this protocol generates 50 images per class on Office-31 and Office-Home, and 1000 images per class on VisDA-2017. In this setting, the synthetic labeled target-style dataset is $\widetilde{\mathcal{D}}_{\mathrm{t}} = \widetilde{\mathcal{D}}_{\mathrm{t}}^{\mathrm{noise}}$.

Full protocol. The full protocol augments the pure-noise route with the DDIM-inversion-based translation route. Both routes use the same visual-token-conditioned interface and assign the source label y_i^{s} to the resulting target-style sample. Accordingly, the final synthetic labeled target-style dataset is

$$\widetilde{\mathcal{D}}_{\mathrm{t}} = \widetilde{\mathcal{D}}_{\mathrm{t}}^{\mathrm{noise}} \ \cup\ \widetilde{\mathcal{D}}_{\mathrm{t}}^{\mathrm{inv}\to\mathrm{t}}. \tag{25}$$

Sampler and branch assignment. In the pure-noise route, synthesis is performed under the target branch ($\sigma = 1$). Final image generation is instantiated with DPM-Solver multistep sampling using 25 inference steps. In the DDIM-based route, source images are first inverted under the source branch ($\sigma = 0$) and then translated under the target branch ($\sigma = 1$) on the same reduced timestep schedule.

A.2.3 Downstream training and evaluation

After synthetic-data construction, we train a downstream target classifier on the resulting synthetic target-style dataset, optionally together with an unsupervised target regularizer, as described in the main paper. We instantiate this stage using MCC and ELS. The benchmark-specific backbones and evaluation metrics are exactly those described above.

A.2.4 Pure-noise target-style synthesis with visual-token conditioning

The first synthetic-data route generates target-style samples directly from Gaussian noise under the target branch of the token-conditioned diffusion model. For a labeled source instance $(x_i^{\mathrm{s}}, y_i^{\mathrm{s}})$, we first extract the source visual tokens U_i^{s}, build the target-style conditioning context S_i^{t} according to equation 24, and then run the diffusion sampler under the target branch ($\sigma = 1$) from an initial Gaussian latent.

To keep the formulation sampler-agnostic, let $\Phi_{\tau_k \to \tau_{k-1}}$ denote the update rule of the chosen diffusion sampler from timestep τ_k to τ_{k-1}. In our experiments, final image generation is instantiated with DPM-Solver multistep sampling using 25 inference steps. Let $0 = \tau_0 < \tau_1 < \cdots < \tau_K = T$ denote the inference schedule. Starting from $z_{\tau_K} \sim \mathcal{N}(0, I)$, the target-branch denoising process iterates

$$\hat{\epsilon}_k = \epsilon_{\theta_0+\Delta\theta}\big(z_{\tau_k}, \tau_k, S_i^{\mathrm{t}}, \sigma = 1\big), \tag{26}$$

$$z_{\tau_{k-1}} = \Phi_{\tau_k \to \tau_{k-1}}\big(z_{\tau_k}, \hat{\epsilon}_k, S_i^{\mathrm{t}}, \sigma = 1\big), \qquad k = K, \ldots, 1. \tag{27}$$

The final latent z_{τ_0} is decoded by the VAE decoder to obtain the synthetic target-style image $\tilde{x}^{\mathrm{t}}$, and the label is inherited from the conditioning source image: $\tilde{y}^{\mathrm{t}} = y_i^{\mathrm{s}}$.

A.2.5 DDIM inversion and target-style translation

The second synthetic-data route starts from a source image rather than Gaussian noise. Given a labeled source sample $(x_i^{\mathrm{s}}, y_i^{\mathrm{s}})$, we first encode it into the latent space, $z_{i,0}^{\mathrm{s}} = E_{\mathrm{VAE}}(x_i^{\mathrm{s}})$, and compute the source visual tokens $U_i^{\mathrm{s}} = G_\psi(x_i^{\mathrm{s}})$. We then form two conditioning contexts:

$$S_i^{\mathrm{s}} = \Big[e\big(p_{\mathrm{s}}(y_i^{\mathrm{s}})\big)\,;\, \kappa_{\mathrm{s}} U_i^{\mathrm{s}}\Big], \tag{28}$$

$$S_i^{\mathrm{t}} = \Big[e\big(p_{\mathrm{s}}(y_i^{\mathrm{s}})\big)\,;\, \eta_{\mathrm{gen}} U_i^{\mathrm{s}}\Big]. \tag{29}$$

The first is used during DDIM inversion under the source branch ($\sigma = 0$), and the second is used during target-style resynthesis under the target branch ($\sigma = 1$).

We use deterministic DDIM inversion on the reduced timestep schedule $0 = \tau_0 < \tau_1 < \cdots < \tau_K = T$. At inversion step k, given the current latent $z_{\tau_{k-1}}^{\mathrm{inv}}$, we predict

$$\hat{\epsilon}_k^{\mathrm{s}} = \epsilon_{\theta_0+\Delta\theta}\big(z_{\tau_{k-1}}^{\mathrm{inv}}, \tau_{k-1}, S_i^{\mathrm{s}}, \sigma = 0\big), \tag{30}$$

estimate the clean latent

$$\hat{z}_{0,k-1}^{\mathrm{s}} = \frac{z_{\tau_{k-1}}^{\mathrm{inv}} - \sqrt{1 - \bar{\alpha}_{\tau_{k-1}}}\,\hat{\epsilon}_k^{\mathrm{s}}}{\sqrt{\bar{\alpha}_{\tau_{k-1}}}}, \tag{31}$$

and move to the next noisier state

$$z_{\tau_k}^{\mathrm{inv}} = \sqrt{\bar{\alpha}_{\tau_k}}\,\hat{z}_{0,k-1}^{\mathrm{s}} + \sqrt{1 - \bar{\alpha}_{\tau_k}}\,\hat{\epsilon}_k^{\mathrm{s}}. \tag{32}$$

After K steps, we obtain the approximate terminal latent $z_{\tau_K}^{\mathrm{inv}}$.

Algorithm 1 Pure-noise target-style synthesis with visual-token conditioning. The procedure is sampler-agnostic and is instantiated with DPM-Solver multistep sampling and $K = 25$ inference steps in our experiments.

Require: Labeled source dataset $\mathcal{D}_{\mathrm{s}}$, trained token-conditioned diffusion model $\epsilon_{\theta_0+\Delta\theta}$, VAE decoder D_{VAE}, source prompt constructor $p_{\mathrm{s}}(\cdot)$, image-to-token encoder G_ψ, class-wise generation budgets $\{B_c\}_{c=1}^{C}$, generation-time token strength η_{gen} (default: $\eta_{\mathrm{gen}} = 1$), inference schedule $\{\tau_k\}_{k=0}^{K}$, target-branch sampler updates $\Phi_{\tau_k \to \tau_{k-1}}$

Ensure: Synthetic labeled target-style dataset $\widetilde{\mathcal{D}}_{\mathrm{t}}^{\mathrm{noise}}$

```
Initialize D̃_t^noise ← ∅
for each class c ∈ {1, ..., C} do
    for b = 1, ..., B_c do
        Sample (x_i^s, y_i^s) ~ D_s such that y_i^s = c
        Extract source visual tokens U_i^s ← G_ψ(x_i^s)
        Form target-style conditioning context S_i^t ← [e(p_s(y_i^s)) ; η_gen U_i^s]
        Sample initial latent z_{τ_K} ~ N(0, I)
        for k = K, K−1, ..., 1 do
            Predict target-branch noise ε̂_k ← ε_{θ0+Δθ}(z_{τ_k}, τ_k, S_i^t, 1)
            Update latent z_{τ_{k−1}} ← Φ_{τ_k→τ_{k−1}}(z_{τ_k}, ε̂_k, S_i^t, 1)
        end for
        Decode x̃^t ← D_VAE(z_{τ_0})
        Add (x̃^t, y_i^s) to D̃_t^noise
    end for
end for
return D̃_t^noise
```

Target-style translation then runs deterministic DDIM sampling from this inverted latent under the target branch. At translation step k, we predict

$$\hat{\epsilon}_k^{\mathrm{t}} = \epsilon_{\theta_0+\Delta\theta}\big(z_{\tau_k}^{\mathrm{t}}, \tau_k, S_i^{\mathrm{t}}, \sigma = 1\big), \tag{33}$$

estimate the clean latent

$$\hat{z}_{0,k}^{\mathrm{t}} = \frac{z_{\tau_k}^{\mathrm{t}} - \sqrt{1-\bar{\alpha}_{\tau_k}}\,\hat{\epsilon}_k^{\mathrm{t}}}{\sqrt{\bar{\alpha}_{\tau_k}}}, \tag{34}$$

and update

$$z_{\tau_{k-1}}^{\mathrm{t}} = \sqrt{\bar{\alpha}_{\tau_{k-1}}}\,\hat{z}_{0,k}^{\mathrm{t}} + \sqrt{1-\bar{\alpha}_{\tau_{k-1}}}\,\hat{\epsilon}_k^{\mathrm{t}}, \qquad k = K, \ldots, 1. \tag{35}$$

The final latent $z_{\tau_0}^{\mathrm{t}}$ is decoded into the translated target-style image $\tilde{x}_i^{\mathrm{inv}\to\mathrm{t}}$, to which we assign the label y_i^{s}.

A.2.6 Remarks on protocol consistency

Both synthetic-data routes are driven by the same token-conditioned interface. In the pure-noise route, the source label and source visual tokens provide the class and instance information used to synthesize a target-style sample from Gaussian noise. In the DDIM-based route, the same source visual tokens are used first to invert the source image under the source branch and then to resynthesize it under the target branch. This design keeps the source-conditioned signal consistent across the two synthetic-data construction routes while allowing the final sample to be rendered in the target style.

Algorithm 2 DDIM-inversion-based target-style translation with visual-token conditioning.

Require: Selected labeled source subset $\mathcal{S}_{\text{inv}} \subseteq \mathcal{D}_{\text{s}}$, trained token-conditioned diffusion model $\epsilon_{\theta_0+\Delta\theta}$, VAE encoder E_{VAE}, VAE decoder D_{VAE}, source prompt constructor $p_{\text{s}}(\cdot)$, image-to-token encoder G_ψ, source-side token strength κ_{s}, generation-time token strength η_{gen} (default: $\eta_{\text{gen}} = 1$), reduced DDIM schedule $\{\tau_k\}_{k=0}^{K}$

Ensure: Synthetic labeled translated dataset $\widetilde{\mathcal{D}}_{\text{t}}^{\text{inv}\to\text{t}}$

Initialize $\widetilde{\mathcal{D}}_{\text{t}}^{\text{inv}\to\text{t}} \leftarrow \varnothing$
for each $(x_i^{\text{s}}, y_i^{\text{s}}) \in \mathcal{S}_{\text{inv}}$ **do**
  Encode source image to latent $z_{i,0}^{\text{s}} \leftarrow E_{\text{VAE}}(x_i^{\text{s}})$
  Extract source visual tokens $U_i^{\text{s}} \leftarrow G_\psi(x_i^{\text{s}})$
  Form source-side inversion context $S_i^{\text{s}} \leftarrow \Big[e\big(p_{\text{s}}(y_i^{\text{s}})\big)\,;\,\kappa_{\text{s}} U_i^{\text{s}}\Big]$
  Form target-side resynthesis context $S_i^{\text{t}} \leftarrow \Big[e\big(p_{\text{s}}(y_i^{\text{s}})\big)\,;\,\eta_{\text{gen}} U_i^{\text{s}}\Big]$
  Initialize $z_{\tau_0}^{\text{inv}} \leftarrow z_{i,0}^{\text{s}}$
  for $k = 1, \ldots, K$ **do** ▷ Deterministic DDIM inversion under source branch
    $\hat{\epsilon}_k^{\text{s}} \leftarrow \epsilon_{\theta_0+\Delta\theta}\big(z_{\tau_{k-1}}^{\text{inv}}, \tau_{k-1}, S_i^{\text{s}}, 0\big)$
    $\hat{z}_{0,k-1}^{\text{s}} \leftarrow \dfrac{z_{\tau_{k-1}}^{\text{inv}} - \sqrt{1-\bar{\alpha}_{\tau_{k-1}}}\,\hat{\epsilon}_k^{\text{s}}}{\sqrt{\bar{\alpha}_{\tau_{k-1}}}}$
    $z_{\tau_k}^{\text{inv}} \leftarrow \sqrt{\bar{\alpha}_{\tau_k}}\,\hat{z}_{0,k-1}^{\text{s}} + \sqrt{1-\bar{\alpha}_{\tau_k}}\,\hat{\epsilon}_k^{\text{s}}$
  end for
  Initialize $z_{\tau_K}^{\text{t}} \leftarrow z_{\tau_K}^{\text{inv}}$
  for $k = K, K-1, \ldots, 1$ **do** ▷ Deterministic DDIM resynthesis under target branch
    $\hat{\epsilon}_k^{\text{t}} \leftarrow \epsilon_{\theta_0+\Delta\theta}\big(z_{\tau_k}^{\text{t}}, \tau_k, S_i^{\text{t}}, 1\big)$
    $\hat{z}_{0,k}^{\text{t}} \leftarrow \dfrac{z_{\tau_k}^{\text{t}} - \sqrt{1-\bar{\alpha}_{\tau_k}}\,\hat{\epsilon}_k^{\text{t}}}{\sqrt{\bar{\alpha}_{\tau_k}}}$
    $z_{\tau_{k-1}}^{\text{t}} \leftarrow \sqrt{\bar{\alpha}_{\tau_{k-1}}}\,\hat{z}_{0,k}^{\text{t}} + \sqrt{1-\bar{\alpha}_{\tau_{k-1}}}\,\hat{\epsilon}_k^{\text{t}}$
  end for
  Decode $\tilde{x}_i^{\text{inv}\to\text{t}} \leftarrow D_{\text{VAE}}(z_{\tau_0}^{\text{t}})$
  Add $(\tilde{x}_i^{\text{inv}\to\text{t}}, y_i^{\text{s}})$ to $\widetilde{\mathcal{D}}_{\text{t}}^{\text{inv}\to\text{t}}$
end for
return $\widetilde{\mathcal{D}}_{\text{t}}^{\text{inv}\to\text{t}}$

Table 5: Standard deviations (%) over three runs on Office-Home under the UDA setting.

Method	**Ar→Cl**	**Ar→Pr**	**Ar→Rw**	**Cl→Ar**	**Cl→Pr**	**Cl→Rw**	**Pr→Ar**	**Pr→Cl**	**Pr→Rw**	**Rw→Ar**	**Rw→Cl**	**Rw→Pr**
ERM Vapnik (2013)	0.25	0.26	0.42	0.17	0.20	0.15	0.07	0.17	0.05	0.34	0.33	0.01
DANN Ganin et al. (2016)	0.44	0.72	0.38	0.02	0.30	0.39	0.58	0.47	0.59	0.84	0.14	0.51
CDAN Long et al. (2018)	0.25	0.62	0.22	0.37	0.58	0.30	0.57	0.36	0.16	0.33	0.23	0.35
AFN Xu et al. (2019)	0.16	0.30	0.06	0.23	0.31	0.14	0.32	0.15	0.02	0.19	0.18	0.22
MDD Zhang et al. (2019)	0.51	0.32	0.06	0.24	0.73	0.41	0.36	0.53	0.24	0.03	0.09	0.11
SDAT Rangwani et al. (2022)	0.51	0.44	0.24	0.13	0.41	0.01	1.46	0.40	0.11	0.46	0.19	0.29
MCC Jin et al. (2020)	0.59	0.22	0.16	0.27	0.52	0.16	0.16	0.38	0.25	0.35	0.35	0.23
ELS Zhang et al. (2023b)	0.83	0.45	0.38	0.08	0.46	0.19	0.39	0.39	0.08	0.02	0.44	0.05
Diffusion-based methods												
MCC+Terra Zhuang et al. (2024)	0.21	0.11	0.14	0.25	0.28	0.18	0.18	0.29	0.25	0.15	0.06	0.11
ELS+Terra Zhuang et al. (2024)	0.06	0.30	0.14	0.30	0.37	0.21	0.10	0.18	0.13	0.68	0.24	0.16
MCC+VT-DUDA	0.32	0.17	0.05	0.41	0.13	0.26	0.37	0.19	0.08	0.31	0.22	0.29
ELS+VT-DUDA	0.12	0.45	0.29	0.08	0.52	0.31	0.19	0.41	0.25	0.88	0.03	0.34

Table 6: Standard deviations (%) over three runs on Office-31 under the UDA setting.

Method	**A→W**	**D→W**	**W→D**	**A→D**	**D→A**	**W→A**
ERM Vapnik (2013)	0.11	0.00	0.00	1.22	0.15	0.11
DANN Ganin et al. (2016)	1.34	0.06	0.08	0.68	0.65	0.39
CDAN Long et al. (2018)	1.75	0.18	0.00	1.19	0.79	0.45
AFN Xu et al. (2019)	0.63	0.07	0.00	0.53	0.50	0.32
MDD Zhang et al. (2019)	1.00	0.15	0.00	0.10	0.68	0.18
SDAT Rangwani et al. (2022)	1.83	0.12	0.00	1.03	0.67	0.34
MSGD Xia et al. (2022)	0.50	0.30	0.00	0.30	0.40	0.50
MCC Jin et al. (2020)	0.38	0.08	0.09	1.47	0.50	0.20
ELS Zhang et al. (2023b)	0.51	0.06	0.00	0.20	0.54	0.16
Diffusion-based methods						
MCC+Terra Zhuang et al. (2024)	0.06	0.06	0.00	0.09	0.18	0.12
ELS+Terra Zhuang et al. (2024)	0.17	0.06	0.00	0.48	0.28	0.11
MCC+VT-DUDA	0.12	0.10	0.00	0.04	0.18	0.15
ELS+VT-DUDA	0.15	0.12	0.00	0.10	0.20	0.18

A.3 Standard Deviations of Main Experimental Results

To complement the mean accuracies reported in the main paper, we provide the standard deviations over three independent runs for the main UDA experiments on Office-Home, Office-31, and VisDA-2017. These results quantify the run-to-run stability of the compared methods under the same experimental protocol as in Sec. 4.

For transparency, we report standard deviations only for methods for which three-run records were available in our experimental logs. In particular, DCDM Zhang et al. (2025) is not included in this subsection because only the published mean accuracies were available to us.

Table 7: Standard deviations (%) over three runs on VisDA-2017 under the UDA setting.

Method	**aero**	**bicycle**	**bus**	**car**	**horse**	**knife**	**motor**	**person**	**plant**	**skate**	**train**	**truck**
ERM Vapnik (2013)	9.90	2.64	3.26	2.20	1.35	3.60	1.41	1.03	1.80	3.97	0.79	0.67
DANN Ganin et al. (2016)	0.39	1.94	0.38	2.80	0.80	3.40	0.76	0.86	0.72	2.00	0.32	2.72
CDAN Long et al. (2018)	0.38	3.72	2.55	1.36	0.53	0.52	0.14	2.58	0.67	0.49	2.61	2.43
AFN Xu et al. (2019)	0.69	3.84	1.80	2.55	1.48	2.51	0.48	2.08	2.47	3.93	1.11	1.27
MDD Zhang et al. (2019)	2.40	9.46	1.18	0.66	0.85	4.01	0.65	1.81	1.21	4.30	1.58	0.33
SDAT Rangwani et al. (2022)	1.40	2.64	1.60	1.67	0.48	0.92	0.82	0.24	0.78	0.84	1.36	0.60
MCC Jin et al. (2020)	0.12	0.92	2.91	0.39	0.28	0.54	0.80	0.87	0.15	0.77	0.55	2.25
ELS Zhang et al. (2023b)	0.93	1.20	1.39	0.47	0.15	0.95	1.38	0.73	1.59	1.02	1.35	0.27
Diffusion-based methods												
MCC+Terra Zhuang et al. (2024)	0.21	0.59	0.12	0.69	0.60	0.60	0.56	0.35	0.40	0.35	0.47	0.88
ELS+Terra Zhuang et al. (2024)	0.34	0.79	0.41	1.31	0.06	0.39	0.85	0.43	0.92	0.64	0.26	0.19
MCC+VT-DUDA	0.15	0.72	0.36	0.81	0.49	0.86	0.28	0.62	0.11	0.55	0.23	0.95
ELS+VT-DUDA	0.65	0.42	0.78	1.15	0.51	0.12	0.99	0.23	1.38	0.88	0.05	0.57

A.4 Additional Ablation Studies

To expand the pure-noise results summarized in Table 3 of the main paper, we report detailed transfer-wise results on Office-Home and Office-31, together with per-class results on VisDA-2017. Throughout this subsection, the reported **VT-DUDA** results are obtained using *only* pure-noise target-style generation under visual-token conditioning, without DDIM inversion. Concretely, VT-DUDA uses 50 generated images per class on Office-Home and Office-31, and 1000 generated images per class on VisDA-2017.

The reference diffusion baselines follow their respective comparison settings. On **Office-Home**, Terra is compared under the same pure-noise budget as VT-DUDA, namely 50 generated images per class, while DCDM is reported under its published setting with 200 generated images per class. On **Office-31**, the reported Terra result corresponds to its full configuration, which combines pure-noise generation with DDIM-inversion-based augmentation, whereas DCDM again uses 200 generated images per class. On **VisDA-2017**, the reported Terra result also corresponds to its full configuration with DDIM inversion, while DCDM uses 2000 generated images per class. We make these distinctions explicit because the purpose of this appendix is to assess the downstream effectiveness and sample efficiency of VT-DUDA under a lighter synthesis budget.

Tables 8 and 9 provide the detailed task-wise results corresponding to the averages reported in the main text. On Office-Home, the budget-matched comparison with Terra directly isolates the effect of conditioning quality: VT-DUDA improves over Terra on 11 out of 12 transfer tasks under both MCC and ELS. Compared with the larger-budget DCDM reference, ELS+VT-DUDA slightly improves over ELS+DCDM in average accuracy (73.72% vs. 73.64%), while MCC+VT-DUDA remains within 0.21 points of MCC+DCDM (73.50% vs. 73.71%), despite using only one quarter as many generated samples per class. This result supports the claim that the synthetic samples produced by VT-DUDA are more useful per sample for downstream adaptation.

On Office-31, the comparison is even stricter, since the Terra reference additionally benefits from DDIM-inversion-based augmentation and DCDM uses a larger synthetic-data budget. Even under this stronger comparison, VT-DUDA still achieves the best average accuracy under both downstream objectives: 91.37% for MCC and 91.92% for ELS, exceeding both Terra (91.34%, 91.28%) and DCDM (91.01%, 91.80%). Although the task-wise pattern is more mixed than on Office-Home, the average gain shows that the proposed visual-token conditioning improves the usefulness of the synthesized target-style data rather than merely increasing the number of generated samples.

Table 10 further reports the per-class results on VisDA-2017. Here VT-DUDA uses only pure-noise generation with 1000 images per class, while Terra uses its full configuration with DDIM inversion and DCDM uses 2000 images per class. Despite this lighter synthesis setting, VT-DUDA achieves the highest mean accuracy under both MCC and ELS. Relative to Terra, MCC+VT-DUDA improves 11 out of 12 categories, and ELS+VT-DUDA also improves 11 out of 12 categories. Relative to DCDM, MCC+VT-DUDA improves 7 out of 12 categories, while ELS+VT-DUDA improves 10 out of 12 categories. These class-wise results are consistent with the benchmark-level averages and indicate that the gains of VT-DUDA arise from systematically better synthetic supervision rather than from isolated categories.

Table 8: Detailed transfer accuracies (%) on Office-Home. VT-DUDA uses only pure-noise target-style generation with 50 generated images per class and no DDIM inversion. Terra is reported under the same pure-noise 50-per-class setting, while DCDM is included as a larger-budget reference with 200 generated images per class. Results are averaged over 12 transfer tasks (Avg). The best performance in each column is highlighted in **bold**.

Method	**Ar→Cl**	**Ar→Pr**	**Ar→Rw**	**Cl→Ar**	**Cl→Pr**	**Cl→Rw**	**Pr→Ar**	**Pr→Cl**	**Pr→Rw**	**Rw→Ar**	**Rw→Cl**	**Rw→Pr**	**Avg**
ERM Vapnik (2013)	44.06	67.12	74.26	53.26	61.96	64.54	51.91	38.90	72.94	64.51	43.84	75.39	59.39
DANN Ganin et al. (2016)	52.53	62.57	73.20	56.89	67.02	68.34	58.37	54.14	78.31	70.78	60.76	80.57	65.29
CDAN Long et al. (2018)	54.21	72.18	78.29	61.97	71.43	72.39	62.96	55.68	80.68	74.71	61.22	83.68	69.12
AFN Xu et al. (2019)	52.58	72.42	76.96	64.90	71.14	72.91	64.08	51.29	77.83	72.21	57.46	82.09	67.99
MDD Zhang et al. (2019)	56.37	75.53	79.17	62.95	73.21	73.55	62.56	54.86	79.49	73.84	61.45	84.06	69.75
SDAT Rangwani et al. (2022)	58.20	77.46	81.35	66.06	76.45	76.41	63.70	56.69	82.49	76.02	62.09	85.24	71.85
MSGD Xia et al. (2022)	58.70	76.90	78.90	70.10	76.20	76.60	69.00	57.20	82.30	74.90	62.70	84.50	72.40
MCC Jin et al. (2020)	56.83	79.81	82.66	67.80	77.02	77.82	66.98	55.43	81.79	73.95	61.41	**85.44**	72.24
ELS Zhang et al. (2023b)	57.79	77.65	81.62	66.59	76.74	76.43	62.69	56.69	82.12	75.63	**62.85**	85.35	71.84
Diffusion-based methods													
MCC+Terra Zhuang et al. (2024)	58.15	80.19	80.41	69.20	80.13	80.02	70.03	53.61	81.10	71.35	56.42	81.48	71.84
ELS+Terra Zhuang et al. (2024)	58.66	80.71	80.94	69.74	80.68	78.95	69.70	54.20	81.68	71.92	56.98	82.03	72.18
MCC+DCDM Zhang et al. (2025)	58.23	80.33	**82.91**	70.14	79.15	81.36	68.49	57.75	**83.44**	74.18	63.81	**85.67**	73.71
ELS+DCDM Zhang et al. (2025)	**60.35**	78.81	82.74	69.59	80.53	79.55	65.16	**58.26**	83.11	**75.81**	**64.18**	85.55	73.64
MCC+VT-DUDA	57.77	**81.56**	82.81	72.03	**81.54**	82.59	**71.37**	57.65	82.54	72.36	57.15	82.62	73.50
ELS+VT-DUDA	59.24	80.41	81.72	**73.60**	80.72	**83.30**	70.72	56.88	82.23	73.81	59.14	82.89	**73.72**

Table 9: Detailed transfer accuracies (%) on Office-31, with the corresponding VisDA-2017 mean accuracy (%) reported in the last column. VT-DUDA uses only pure-noise target-style generation with 50 generated images per class and no DDIM inversion. Terra is reported under its full configuration combining pure-noise generation (50 per class) with DDIM-inversion-based augmentation, while DCDM is included under its published 200-per-class setting. The best performance in each column is highlighted in **bold**.

Dataset	**Office-31**							**VisDA-2017**
Method	**A→W**	**D→W**	**W→D**	**A→D**	**D→A**	**W→A**	**Avg**	**mean**
ERM Vapnik (2013)	77.07	96.60	99.20	81.08	64.11	64.01	80.35	51.47
DANN Ganin et al. (2016)	89.85	97.95	99.90	83.26	73.28	73.75	86.33	79.02
CDAN Long et al. (2018)	92.42	98.62	**100.00**	91.44	74.61	72.80	88.32	80.74
AFN Xu et al. (2019)	91.82	98.77	**100.00**	95.12	72.43	70.71	88.14	74.64
MDD Zhang et al. (2019)	93.55	98.66	**100.00**	93.92	75.29	73.95	89.23	81.10
SDAT Rangwani et al. (2022)	91.32	98.83	**100.00**	95.25	76.97	73.19	89.26	83.23
MSGD Xia et al. (2022)	95.50	99.20	**100.00**	95.60	77.30	77.00	90.80	84.60
MCC Jin et al. (2020)	94.09	98.32	99.67	94.25	75.89	75.46	89.61	83.32
ELS Zhang et al. (2023b)	93.84	98.78	**100.00**	95.78	77.72	75.13	90.21	83.40
Diffusion-based methods								
MCC+Terra Zhuang et al. (2024)	94.55	99.03	**100.00**	96.46	78.64	79.37	91.34	85.39
ELS+Terra Zhuang et al. (2024)	94.09	**99.21**	**100.00**	96.25	78.67	79.45	91.28	86.86
MCC+DCDM Zhang et al. (2025)	95.51	98.58	99.93	95.31	78.26	78.43	91.01	86.56
ELS+DCDM Zhang et al. (2025)	96.90	98.91	**100.00**	97.46	**79.79**	77.74	91.80	86.29
MCC+VT-DUDA	96.22	98.10	99.95	**99.88**	77.01	77.05	91.37	86.92
ELS+VT-DUDA	**97.58**	97.52	99.52	98.02	77.88	**81.01**	**91.92**	**87.56**

Table 10: Per-class transfer accuracies (%) on VisDA-2017. VT-DUDA uses only pure-noise target-style generation with 1000 generated images per class and no DDIM inversion. Terra is reported under its full configuration with DDIM-inversion-based augmentation, while DCDM is included under its published 2000-per-class setting. The best performance in each column is highlighted in **bold**.

Method	**aero**	**bicycle**	**bus**	**car**	**horse**	**knife**	**motor**	**person**	**plant**	**skate**	**train**	**truck**	**mean**
ERM Vapnik (2013)	81.71	22.46	54.08	**76.21**	74.83	10.69	83.81	18.71	80.88	28.66	79.66	5.98	51.47
DANN Ganin et al. (2016)	94.75	73.47	83.46	47.91	87.00	88.30	88.47	77.18	88.16	90.05	87.21	42.26	79.02
CDAN Long et al. (2018)	94.55	74.41	82.22	58.92	90.56	96.22	89.71	78.90	86.11	89.06	84.81	43.42	80.74
AFN Xu et al. (2019)	93.13	54.76	81.03	69.74	92.36	75.88	92.11	73.83	93.16	55.55	**90.48**	23.63	74.64
MDD Zhang et al. (2019)	92.68	65.26	82.29	66.78	91.68	92.09	**93.18**	79.67	92.12	84.95	83.85	48.66	81.10
SDAT Rangwani et al. (2022)	94.51	83.56	74.28	65.78	93.00	95.83	89.61	80.04	90.86	91.47	84.95	54.93	83.23
MSGD Xia et al. (2022)	97.50	83.40	**84.40**	69.40	95.90	94.10	90.90	75.50	**95.50**	94.60	88.10	44.90	84.60
MCC Jin et al. (2020)	95.26	86.14	77.12	69.98	92.83	94.84	86.52	77.78	90.26	90.98	85.68	52.52	83.32
ELS Zhang et al. (2023b)	94.76	83.38	75.44	66.45	93.16	95.14	89.09	80.13	90.77	91.06	84.09	57.36	83.40
Diffusion-based methods													
MCC+Terra Zhuang et al. (2024)	96.20	87.27	78.77	70.59	94.18	95.49	85.08	85.48	92.24	93.20	86.26	59.88	85.39
ELS+Terra Zhuang et al. (2024)	95.98	87.12	81.60	70.84	95.14	96.29	88.47	87.78	94.75	94.06	86.47	**63.83**	86.86
MCC+DCDM Zhang et al. (2025)	96.43	87.28	83.16	74.37	94.59	96.13	88.60	81.90	92.98	94.45	87.26	61.56	86.56
ELS+DCDM Zhang et al. (2025)	96.20	84.79	83.15	73.28	94.76	96.58	90.99	82.21	92.98	93.37	87.49	59.70	86.29
MCC+VT-DUDA	97.11	**88.49**	82.36	72.34	95.88	96.42	89.23	**88.07**	94.91	94.16	87.14	56.93	86.92
ELS+VT-DUDA	**97.79**	87.91	83.19	73.14	**96.06**	**96.81**	91.25	88.01	95.15	**94.71**	88.94	57.76	**87.56**

A.5 Additional Qualitative Results

In this subsection, we provide additional qualitative results to complement the quantitative findings reported in the main paper and in Appendix A.4. Our purpose is to visualize the behavior of the proposed visual-token-conditioned generation framework under the two synthetic-data construction routes used in VT-DUDA, namely pure-noise target-style synthesis and DDIM-inversion-based target-style translation. We further provide an additional visualization of inference-time token-strength control in the pure-noise setting.

We begin with qualitative results obtained using Algorithm 1. In this setting, a labeled source image provides both the class prompt and the corresponding visual-token conditioning, and the target-style sample is synthesized from Gaussian noise under the target branch ($\sigma = 1$). Figure 5 summarizes representative pure-noise synthesis results on Office-Home. The figure contains all twelve transfer tasks, with one representative class shown for each domain shift. Figure 6 provides the corresponding qualitative visualization on VisDA-2017, where six representative examples are shown in a single composed figure.

We then present qualitative results obtained using Algorithm 2. In this setting, a labeled source image is first inverted under the source branch ($\sigma = 0$), and is then translated to the target style under the target branch ($\sigma = 1$) using the same class prompt together with the visual tokens extracted from the source image. Figure 7 shows representative results on Office-Home for all twelve transfer tasks, again using one representative class per transfer direction.

Finally, Figure 8 visualizes inference-time token-strength scaling for pure-noise synthesis in the Art→Clipart setting. Here, we extract concept tokens from a class-matched source image and generate target-style samples from Gaussian noise under the target condition ($\sigma = 1$), while varying the token strength $\eta_0 \in \{0.5, 1.0, 1.5, 2.0\}$. This figure illustrates that the proposed tokenized conditioning interface permits direct control of the conditioning signal at inference time without modifying the diffusion backbone or re-training the model.

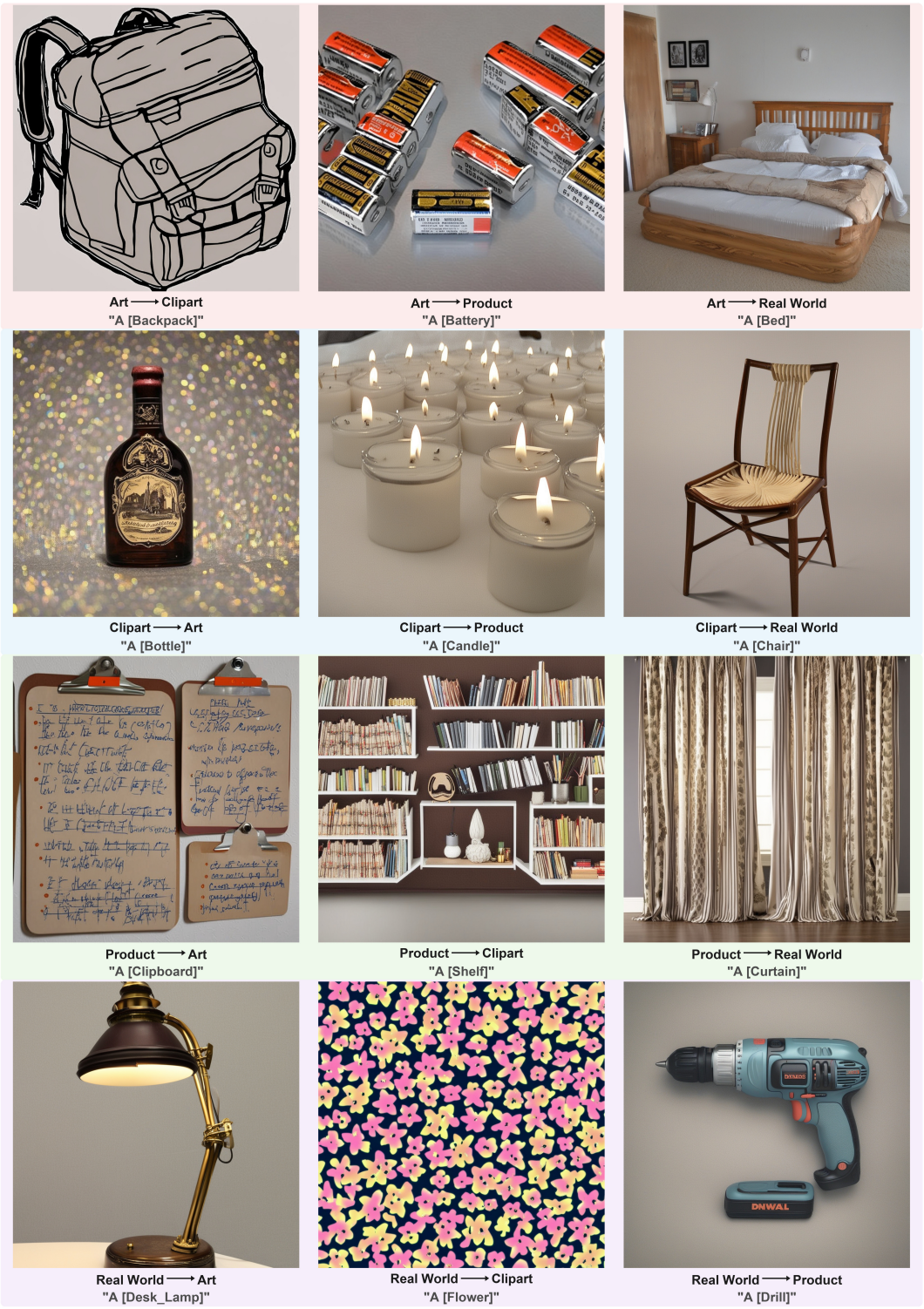

Figure 5: Additional qualitative results for pure-noise target-style synthesis on Office-Home using Algorithm 1. The figure is composed of representative results for all twelve Office-Home transfer tasks. For each domain shift, we show one representative class generated from Gaussian noise under the target branch ($\sigma = 1$), conditioned on the class prompt and the visual tokens extracted from a class-matched source image. These examples illustrate how the proposed visual-token conditioning provides instance-dependent guidance for target-style synthesis across diverse domain pairs.

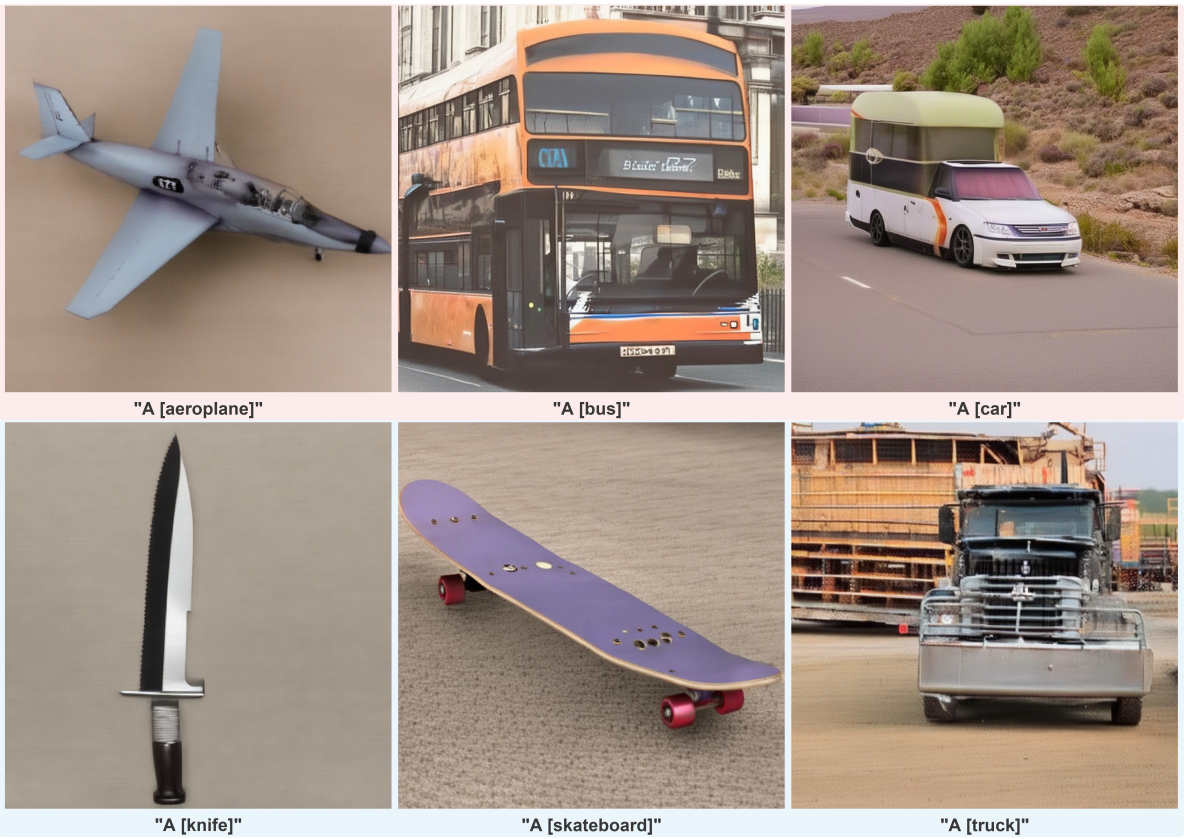

Figure 6: Additional qualitative results for pure-noise target-style synthesis on VisDA-2017 using Algorithm 1. The figure contains six representative generated examples. In each case, the target-style image is synthesized from Gaussian noise under the target branch ($\sigma = 1$), using the class prompt together with the visual-token conditioning extracted from a class-matched source image.

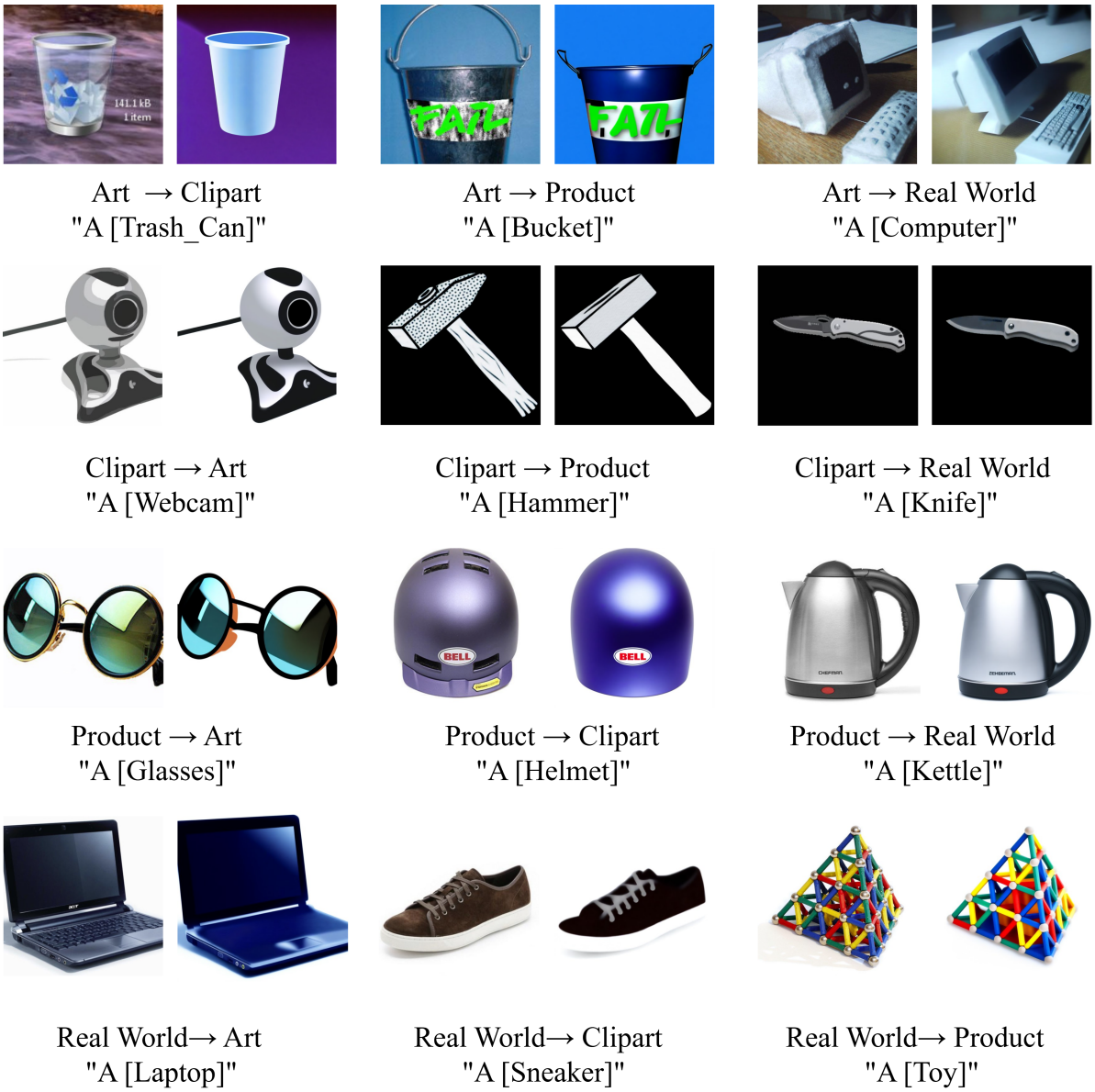

Figure 7: Additional qualitative results for DDIM-inversion-based target-style translation on Office-Home using Algorithm 2. The figure is composed of representative results for all twelve Office-Home transfer tasks. For each domain shift, we show one representative class obtained by first inverting a labeled source image under the source branch ($\sigma = 0$) and then re-synthesizing it under the target branch ($\sigma = 1$) with the same class prompt and source-image visual tokens. The visualization shows that the proposed visual-token-conditioned interface is compatible not only with pure-noise synthesis but also with inversion-based target-style translation.

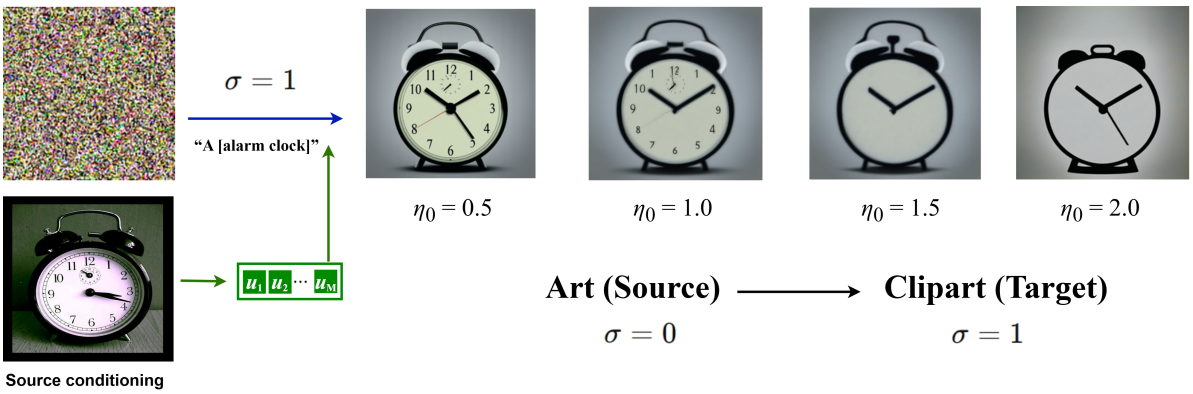

Figure 8: Token-strength scaling for pure-noise synthesis (Art→Clipart). We extract concept tokens from a class-matched source image and generate target-style samples from pure noise under the target condition σ=1, while varying the token strength $\eta_0 \in \{0.5, 1.0, 1.5, 2.0\}$. The resulting visualization shows how changing the strength of the visual-token conditioning affects the synthesized target-style samples, highlighting the controllability of the proposed tokenized conditioning interface at inference time.

A.6 Comparison with the SDXL Prior

To further clarify the role of diffusion adaptation in generation-based UDA, we include an additional comparison against target-like sample generation that relies only on the prior knowledge of a pretrained SDXL model. The goal of this comparison is to distinguish improvements obtained from *stronger prompt design and sample selection on top of an off-the-shelf text-to-image prior* from improvements obtained by *explicitly adapting the generative model and its conditioning interface for the source-target transfer*. To this end, we adopt the SDXL-prior baselines reported in Terra and compare them with our method under the same Office-Home benchmark and the same ELS downstream UDA setting.

For clarity, we briefly summarize how each SDXL-prior baseline is constructed. These baselines synthesize target-like samples directly from the SDXL prior with progressively stronger prompt engineering and post-selection:

- **SDXL (random):** Samples are generated using only the class prompt "A [CLASS]", where $[CLASS]$ denotes the category label.
- **SDXL (styles):** First, GPT-4 is asked to generate 50 diverse style prompts. These style prompts are then inserted into the template "A [CLASS], an everyday object in office and home, in the style of [STYLE]" to synthesize images.
- **SDXL (target):** Based on the previous setting, the style placeholder $[STYLE]$ is replaced by the target-domain name itself (e.g., "Clipart") so that the SDXL prior is prompted more directly toward the target domain.
- **SDXL (target styles):** Instead of using only the domain name, GPT-4 is asked to generate 50 prompts that explicitly describe the target-domain style, yielding richer target-style prompts for synthesis.
- **SDXL (selected):** Finally, the samples generated by *SDXL (target styles)* are further filtered with a confidence-based active-learning-style selection strategy to remove poor-quality or misclassified synthetic samples.

This sequence of baselines is useful because it forms a progressively stronger test of what can be achieved by the SDXL prior alone. In particular, it starts from plain class prompting, then adds generic style diversity, then explicitly injects target-domain information, then refines that information into more detailed target-style prompts, and finally adds sample selection to improve the quality of the synthetic set. Therefore, if our method still outperforms these increasingly strong SDXL-prior baselines, the gain cannot be attributed merely to prompt engineering or filtering on top of a powerful pretrained text-to-image model.

The results are shown in Table 11. As the SDXL-prior baselines become stronger, the average UDA accuracy increases from 69.92% for *SDXL (random)* to 73.99% for *SDXL (selected)*. This confirms that more target-aware prompting and sample filtering do improve the usefulness of synthetic target-like data. However, our method still achieves 77.34% average accuracy, which is +3.35 points higher than the strongest SDXL-prior baseline.

The SDXL-prior baselines already test several increasingly sophisticated ways of exploiting a strong pretrained generative prior. Nevertheless, they remain consistently below our method. This suggests that, for downstream UDA, it is not sufficient to generate visually target-like samples using better prompts alone. Instead, the generative process must be adapted so that the conditioning signal better preserves class semantics while aligning the generated image with the target domain. Our visual-token-conditioned framework is designed precisely for this purpose: it enriches class prompting with instance-dependent visual information and trains the target branch under the same token-conditioned interface used during generation. The comparison with the SDXL prior therefore supports our central claim that the utility of synthetic data in diffusion-guided UDA depends not only on the strength of the pretrained prior, but also on the design of the adaptation and conditioning mechanism.

Table 11: Comparison with SDXL-prior-based target-like sample generation on Office-Home under the UDA setting with ELS. The five SDXL-prior baselines are taken from Terra Zhuang et al. (2024), while the last row reports our method under the same benchmark and downstream UDA setting. The best result in each column is highlighted in **bold**.

Method	**Ar→Cl**	**Ar→Pr**	**Ar→Rw**	**Cl→Ar**	**Cl→Pr**	**Cl→Rw**	**Pr→Ar**	**Pr→Cl**	**Pr→Rw**	**Rw→Ar**	**Rw→Cl**	**Rw→Pr**	**Avg**
SDXL (random)	56.88	73.64	80.38	69.18	73.64	80.40	68.93	56.54	80.38	68.93	56.54	73.64	69.92
SDXL (styles)	55.23	77.09	80.26	68.11	77.09	80.26	68.11	55.23	80.45	68.11	55.23	77.04	70.18
SDXL (target)	59.70	75.51	82.26	66.67	75.51	82.26	66.67	59.70	82.26	66.67	59.70	75.51	71.04
SDXL (target styles)	60.76	79.52	81.68	70.95	79.52	81.68	70.95	60.76	81.68	70.95	60.76	79.52	73.23
SDXL (selected)	61.63	79.81	82.19	71.98	79.73	81.82	71.69	61.58	82.07	72.76	62.15	80.42	73.99
ELS+VT-DUDA	**63.26**	**83.74**	**85.03**	**72.63**	**85.67**	78.89	**71.55**	**64.91**	**85.29**	**77.58**	**65.17**	**87.16**	**77.34**