A. Implementation details

**Diffusion pre-training.** We follow official implementations of DDPM, EDM and DiT for generative diffusion pre-training. The networks used in DDPM and EDM are UNets based on Wide ResNet with multiple convolutional down-sampling and up-sampling stages. Single head self-attention layers are used in the residual blocks at some resolutions. For CIFAR-10, we retrieve official checkpoints1 from their codebases. For Tiny-ImageNet, we use official (or equivalent) implementations and similar configurations to train unconditional diffusion models by ourselves. The setting is in Table 5. Transformer-based DiT-XL/2 pre-trained on 2562 ImageNet is retrieved from its official codebase3, and we do not train a smaller version (e.g. DiT-B/2) due to the high computational cost. The used off-the-shelf VAE model for latent compression is retrieved from Stable Diffusion4, which has a down-sample factor of 8.

**Linear probing and fine-tuning.** We use very simple settings for linear probing and fine-tuning experiments (see Table 6 and Table 7) and we intentionally do not tune the hyper-parameters such as Adam $\beta_1/\beta_2$ or weight decays. In contrast with common practices in representation learning, we do not use additional normalization layers before linear classifiers since we find it also works well.

To train latent-space DiTs for recognition efficiently, we store the extracted latent codes through the VAE encoder and train DiTs in an offline manner. We encode 10 versions of the training set with data augmentations and randomly sample one version per epoch at the training. This approach may suffer from insufficient augmentation, and increasing augmentation versions or training with online VAE encoder may improve the recognition accuracy.

**Supervised training from scratch.** In Figure 4, we present recognition accuracies of truncated UNet encoders trained from scratch and compare them to supervised Wide ResNets. The setting is in Table 8. We intentionally train these supervised models for long duration (200 epochs) to reach maximum performance for fair comparisons.

B. Layer-noise combinations in grid search

In Section 3.2 we have shown that the layer-noise combination affects representation quality heavily. We perform grid searching to find a good enough, if not the best, combination for each model and dataset. For 18-step or 50-step EDM models, we train linear classifiers for 10 epochs with each layer and timestep. For 1000-step DDPM or DiT, we increase the timestep by 5 or 10 to search more efficiently. Table 9 shows the combinations adopted in Section 4.

---

1 https://github.com/pesser/pytorch_diffusion
2 https://github.com/NVlabs/edm
3 https://github.com/facebookresearch/DeepImageText
4 Hugging Face/Diffusers