CleanDIFT: Diffusion Features without Noise

Supplementary Material

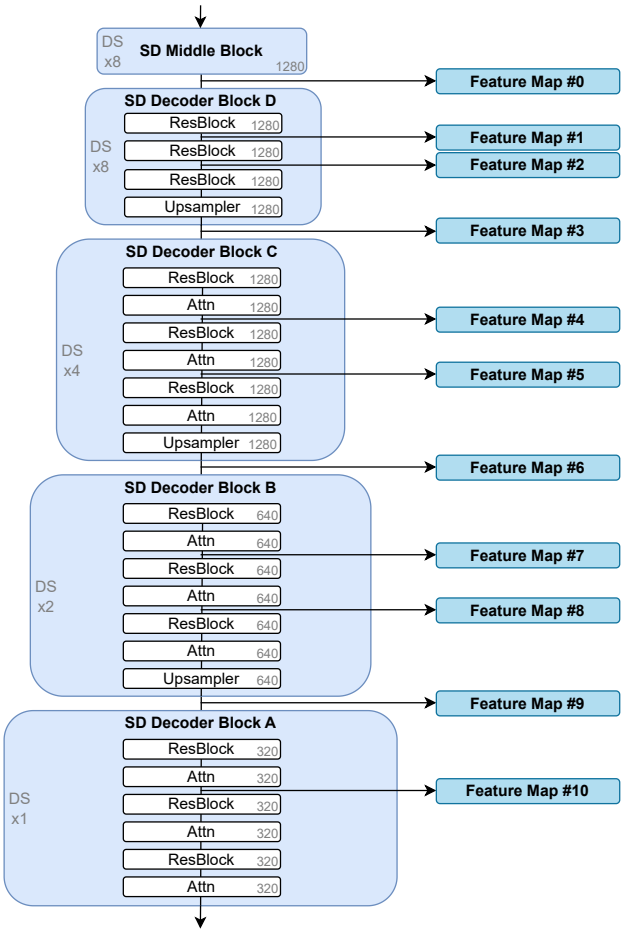

Figure 11. We align the feature maps between our feature extractor and the diffusion model at multiple stages within the network to enable the usage of multiple feature maps for downstream tasks. In total, we extract and align features at $K = 11$ stages of the SD U-Net decoder. The downsampling factor for the different blocks is denoted as *DS* and the channel dimension is shown on the right side of each block.

A. Architecture Details

An illustration of where exactly we extract and align feature maps is provided in Figure 11. The decoder architecture is identical for SD 1.5 and SD 2.1, therefore Figure 11 applies to both models. DIFT [61] extracts feature map #6. A Tale of Two Features [65] and Telling Left from Right [66] both extract feature maps #2, #6, and #8.

Figure 12. Additional qualitative results for semantic correspondence matching using DIFT [61] with the standard SD 2.1 ($t = 261$) and our CleanDIFT features. Our clean features show significantly less incorrect matches than the standard diffusion features, especially along texture-less edges.

B. Additional Quantitative Evaluations

Unsupervised Semantic Correspondence In Tab. 4, we provide an extended version of Tab. 1, in which we report the PCK metric per category of the SPair71k dataset [36]. The categories for which we observe the largest gain when comparing our CleanDIFT features to standard diffusion features are *TV*, *Plant*, and *Chair*. These classes are characterized by long, texture-less edges: the bezel of a TV monitor, the pot of a plant, and the legs of a chair. Supporting this observation, the performance gain for samples from the *Plant* class mostly comes from keypoints not on the plant itself but on the pot of the plant. This is further illustrated in Figure 5. We conclude that our CleanDIFT features are particularly effective for matching corresponding keypoints located along texture-less edges.

Method	Our Features	PCK@α_{bbox} = 0.1 per category (↑)																		
		Aero	Bike	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Dog	Horse	Motor	Person	Plant	Sheep	Train	TV	**All**
DIFT [61]	✗	63.41	55.10	80.40	34.55	46.15	52.26	48.02	75.86	39.46	75.57	55.00	61.71	53.32	46.53	56.36	57.68	71.30	63.63	59.57
	✓	63.72	55.90	80.50	35.40	49.36	53.46	48.08	75.78	43.10	76.20	55.69	61.01	54.17	49.14	62.56	58.37	74.63	**71.54**	61.43 ▲1.86
A Tale of Two Features [65]	✗	71.26	62.23	87.01	37.24	53.78	54.32	51.20	78.61	46.50	78.93	64.43	69.47	62.23	69.27	59.28	68.03	65.40	53.81	63.73
	✓	71.12	62.70	87.42	38.33	**54.78**	54.67	51.20	78.52	47.86	79.38	64.88	69.18	62.61	69.72	**62.82**	68.87	67.51	59.04	64.81 ▲1.08
Telling Left from Right [66]	✗	**78.14**	**66.37**	**89.60**	43.74	53.29	66.61	59.94	**82.66**	51.75	**82.79**	**68.95**	74.91	65.84	**71.67**	57.71	**72.24**	83.46	49.66	68.64
	✓	77.17	65.65	89.58	**44.24**	54.27	**67.24**	**60.63**	82.33	**56.57**	82.53	68.37	**75.91**	**65.99**	71.37	62.29	70.42	**84.58**	59.84	**69.99** ▲1.35

Table 4. Reproduced results for zero-shot unsupervised semantic correspondence matching, evaluated on SPair71k [36]. The three categories for which we observe the largest overall gains are marked in blue. We report PCK@α = 0.1 with an error margin relative to bounding box sizes on the test split of SPair71k, aggregated per point and per category. We compare our reproductions against the papers' reported numbers in Tab. 5

Method	Eval Method	PCK@α (↑)	
		α_{img} = 0.1	α_{bbox} = 0.1
DIFT [61]	reported	-	59.50
	reproduced	66.53	59.57
A Tale of Two Features [65]	reported	-	64.00
	reproduced	72.31	63.73
Telling Left from Right [66]	reported	-	69.60
	reproduced	77.07	68.64

Table 5. Reproduced vs reported numbers for zero-shot semantic correspondences, evaluated on SPair71k [36]. A Tale of Two Features [65] and Telling Left from Right [66] report higher PCK values than our reproduction because they utilize a conditioning mechanism on CLIP image embeddings from [63] that was fine-tuned for panoptic segmentation. As this task is related to semantic correspondence matching, we do not consider using this conditioning mechanism fair in comparison to other zero-shot approaches for semantic correspondences. Therefore, we exclude it from our reproductions.

Supervised Semantic Correspondence We test our method in the supervised setting of TLFR [66] that uses an aggregation network [34, 65] to fuse feature maps. In this setting, our features achieve a PCK_{bbox}@α=0.1 per image of 83.9, outperforming SD2.1 features (83.2). This shows that even in a supervised setting, our features provide additional information that cannot be extracted by additional supervised training.

Distilled Text Conditioning In our standard setting, we train CleanDIFT with image-text pairs since that is what the diffusion model expects as input. As a result, the model also requires a fitting text prompt for optimal feature performance at inference time. Here, we experiment with distilling the text conditioning during our fine-tuning to directly yield optimal features without the need for a prompt. To that end, we train a CleanDIFT version that depends neither on explicit nor implicit captions [65, 66] by distilling the text conditioning directly into its features. This results in a 1.2 PCK_{bbox}@α=0.1 gain over DIFT SD2.1 features with 8x ensembling and no text prompt when extracting features. Compared to simply training CleanDIFT without captions and no text conditioning during inference, the distillation improves performance by 0.4 PCK_{bbox}@α=0.1.

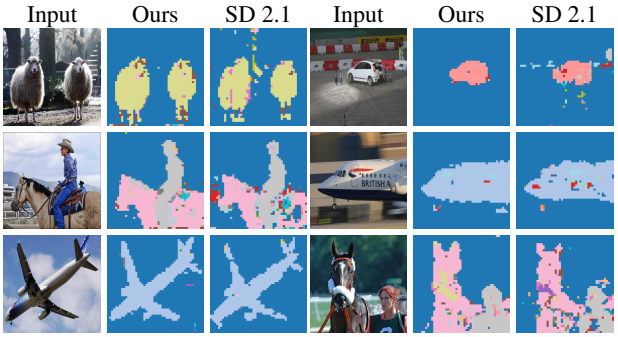

Figure 13. Additional qualitative results for semantic segmentation from diffusion features on Pascal VOC [16]. Standard SD features use $t = 100$ as the timestep, which we found to perform best quantitatively (c.f. Figure 9).

C. Additional Qualitative Samples

We provide additional qualitative samples for semantic correspondence matching in Figure 12 and for semantic segmentation in Figure 13. Figure 16 shows a PCA visualization of our features revealing that they are less noisy than standard features.

D. Depth Probes Analysis

We show a more thorough analysis of the depth probe experiment presented in Sec. 4.3. We show the full set of linear probes for depth prediction on our projected features, i.e. the outputs of the projection heads for different timesteps. A comparison of the performance across timesteps is provided in Figure 15. We observe that the performance of linear probes trained on the projected features decreases for large timesteps, albeit significantly less severe than for standard diffusion features due to the absence of noise. The best performance on projected features is achieved at timestep $t = 499$, while the best performance for standard diffusion features is achieved at timestep $t = 299$.

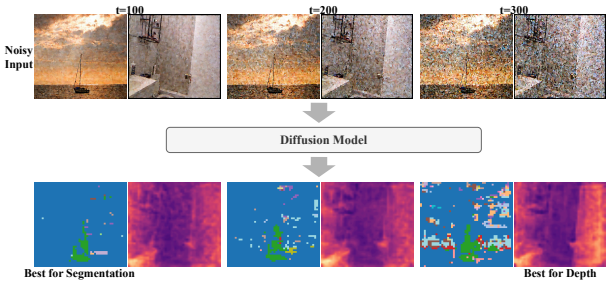

Figure 14. Depending on the downstream application, different diffusion timesteps result in optimal feature representations. For semantic segmentation, $t = 100$ is optimal resulting in a much cleaner segmentation map compared to higher timesteps. However, for depth estimation, the low timestep yields inaccurate depth estimates and a higher timestep is necessary ($t = 300$). CleanDIFT remedies the dependence on the timestep and yields optimal features for every downstream task without additional tuning (Figure 1).

SPair71k (Test)	COYO (Generic, Ours)	ImageNet (mismatched)
61.42	61.43	60.78

Table 6. SPair $\text{PCK}_{\text{bbox}}@\alpha$=0.1 when training on different datasets. A generic dataset performs best and training on the test dataset does not yield any additional gains.

E. Additional Projection Head Ablations

We investigate the influence of different components of the projection head architecture and the influence of pre-training the projection heads following the setup in Sec. 4.6. In our main configuration, we use a FiLM layer [44] in each FFN block to adaptively scale activations depending on the timestep t. We replace the FiLM layers with Adaptive RMS (AdaRMS) norm layers [64] and observe a performance degradation of 0.1 percentage points for PCK_{bbox}. We conclude that removing the scale and shift information of the model's feature by normalization is harmful and cannot be recovered by subsequent scaling and shifting.

Our main configuration for the projection heads uses the SwiGLU [56] gating mechanism as an activation function in each FFN block. We investigate the influence of removing this gating mechanism from our FFNs, effectively leaving us with Swish layers [48]. When removing the gating mechanism, PCK_{bbox} slightly decreases by 0.12 percentage points. Additionally, we experiment with fine-tuning the projection heads before training our feature extraction model. After pre-training the projection heads, we fine-tune them in two settings: fine-tuning both the feature extraction model and projection heads, and training only the feature extraction model while locking the pre-trained projection heads. When fine-tuning both the feature extraction model and projection heads, we achieve the exact same performance as our main configuration which does not use pre-training. When locking the projection heads during fine-tuning, the performance slightly decreases by 0.16 percentage points for PCK_{bbox}.

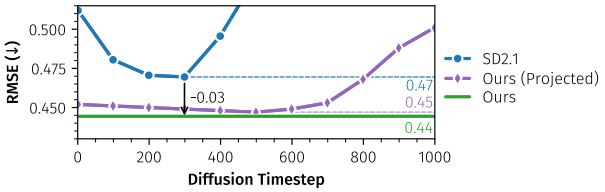

Figure 15. Metric depth prediction for NYUv2 [39] using linear probes. We investigate our proposed projection heads' outputs by training linear probes for depth prediction on them, following the procedure described in Sec. 4.3. This figure extends the results presented in Tab. 2 by showcasing the performance over timesteps.

Backbone	PCK@α Gain (↑)	
	$\alpha_{\text{img}} = 0.1$	$\alpha_{\text{bbox}} = 0.1$
SDXL [45]	1.7	1.6
SDXL Turbo [54]	3.2	3.7
PIXART-α [7]	2.7	2.2
Flux [14]	9.4	8.1

Table 7. We assess the effectiveness of our features for other diffusion backbones such as a much larger UNet (SDXL), diffusion transformers (PIXART-α, Flux), and a distilled model (SDXL Turbo). The PCK quantifies the gain obtained by using our features for semantic correspondences using the standard DIFT [61] setup when compared to the standard features. We show that using our CleanDIFT features leads to better performance for all backbones.

F. Other Diffusion Backbones

We evaluate whether our method applies to other diffusion backbones by evaluating semantic correspondence performance in a DIFT [61] setting and find gains in all cases (cf. Tab. 7). This shows that diffusion models suffer from noisy features independent of their size and architecture, and that CleanDIFT can successfully remedy that. Note that for SDXL (Turbo) and Flux we trained LoRAs [25] instead of fully fine-tuning the entire model due to their large model size. We adapt the self-attention and FFN layers with LoRAs of rank 64. We swept all models for the optimal timestep and feature map and used that for the PCK calculation.

Figure 16. Principal component visualization of our features. One 4-component feature PCA is computed per column, with the first component being thresholded at 0 to obtain the foreground and the remaining being mapped to RGB color. CleanDIFT produces similarly semantically useful features as DIFT, while exhibiting less noise.

G. Dataset Considerations

We evaluated different datasets and find simple choices to suffice. Specifically, a random (only filtered to a minimum size of 768^2) subset of COYO-700M with $\sim$3k images sufficed for our optimal results. COYO-700M is a dataset that is close to the original model's pretraining setting and not tailored to any of the categories relevant to our downstream evals. We show zero-shot semantic correspondence results across different fine-tuning datasets in Tab. 6. Overfitting on the test setting, e.g., by training on SPair71k while also evaluating on that dataset does not give gains over our non-tailored version, but using a dataset that has little overlap with the distribution of the target task (e.g., ImageNet) results in reduced performance.

H. Evaluation Design Choices

During our efforts to reproduce the numbers reported by [61, 65, 66], we found a variety of differences between evaluation pipelines that influence the results. We list them here to provide some clarity for future comparisons.

CLIP Image Embedding Conditioning A Tale of Two Features [65] and Telling Left from Right [66] employ a conditioning mechanism on CLIP image embeddings from [63] that was fine-tuned for panoptic segmentation. Specifically, they multiply the CLIP image embedding element-wise with a learned tensor. This reweighed CLIP conditioning is then added to the embedding of an empty prompt and subsequently passed to the U-Net as the prompt embedding. Additionally, another learned tensor is used to element-wise scale the CLIP image embedding and then add it to the timestep embedding.

Sliding Window Both A Tale of Two Features [65] and Telling Left from Right [66] use a sliding window approach to account for input resolutions higher than the native model input resolution. Specifically, they perform forward passes for overlapping patches, each patch having the model input resolution. Additionally, the methods use different resizing strategies to handle non-square images. DIFT [61] simply resizes the non-square input images to the square input resolution of the evaluation pipeline. A Tale of Two Features [65] and Telling Left from Right [66] resize the input image such that the longer side matches the evaluation resolution and pad the remaining part of the square image with zeros.