

---

# Appendix of DenseDPO: Fine-Grained Temporal Preference Optimization for Video Diffusion Models

---

Anonymous Author(s)

Affiliation

Address

email

- 1 **We highly encourage the readers to check out the website in the supplementary material for**  
2 **video results of baselines and DenseDPO.** Please open the `index.html` file within the `website/`  
3 folder in the browser.
- 4 Our Appendix contains the following sections:
- 5 • Appendix **A** provides details about the training and testing datasets, baselines, evaluation settings,  
6 and the implementation details of our model;
  - 7 • Appendix **B** analyzes a potential cause of why StructuralDPO underperforms VanillaDPO, from  
8 the perspective of model unlearning;
  - 9 • Appendix **C** provides more experimental results of our DenseDPO;
  - 10 • Appendix **D** discusses limitations, future works, and potential positive and negative societal  
11 impacts of this work;
  - 12 • Appendix **E** presents the pseudo code for StructuralDPO and DenseDPO.

## 13 A Detailed Experimental Setup

14 In this section, we provide full details on the datasets, baselines, evaluation settings, and the training  
15 and inference implementation details of our model.

### 16 A.1 Training Data

17 Following common practice in diffusion model post-training [6, 18, 37], we curate a high-quality data  
18 subset from existing large-scale video datasets [5, 40]. We mostly follow [31] to filter the resolution,  
19 duration, aesthetic quality, and motion strength of videos. Inspired by VideoAlign [22], we further  
20 apply GPT-4o [2] to only retain prompts with non-trivial motion.

#### GPT-4o Prompt Filtering Template

Please help me classify if a text prompt contains challenging dynamics. Ignore camera motion and description of the background in the text prompt. Only focus on foreground objects. In general, we are looking for complex motion that requires precise, coordinated movement such as doing sports.

Here are some good examples:

1. A skateboarder performs jumps.
2. On a rainy rooftop, a pair of hip-hop dancers lock and pop in perfect sync.
3. A figure skater executes a powerful leap.
4. A woman transitions gracefully on an aerial hoop under golden hour light.
5. An acrobat holding a handstand on a mat.
6. A martial artist performs a spinning hook kick in a misty bamboo forest.

Additional rules:

1. Only keep real-time videos, and remove any video that is slow-motion, time-lapse, or aerial shot;
2. Remove videos with more than five major subjects in the scene, such as sports games or a group of people doing something;
3. Remove videos with any of the following content: screen-in-screen, screen recording, special effects from video editing, cartoon, animation, TV news, and video games;
4. When not violating the previous rules, you can keep videos with any of the following content: eating, cutting, or any action that causes big deformation of the main object, such as "A person takes a bite of a hamburger / cuts a steak / squeezes a sponge / makes a dough";
5. Be conservative. If you are uncertain about a prompt, please classify it as "no".

Please reply "yes" or "no" in the first line of your response. Then, please explain your decision in the second line. I will now provide the prompt for you to classify:

[PROMPT]

21  
22 The GPT-4o template we used is shown above. Overall, our final dataset contains around 55k  
23 high-quality text-video pairs, which we used to generate preference data.

24 **Data processing.** The training dataset contains videos of different lengths, resolutions, and aspect  
25 ratios. Following common practice [30, 43], we use data bucketing, which groups videos into a fixed  
26 set of sizes. Overall, we sample videos up to 512 resolution and 5s during training.

### 27 A.2 Baselines

28 **Pre-trained model.** Our base text-to-video model is a latent Diffusion Transformer framework [29].  
29 It leverages a MAGVIT-v2 [42] as the autoencoder and a stack of 32 DiT blocks as the denoiser  $G_\theta$ .  
30 The autoencoder is similar to the one in CogVideoX [41], which downsamples the spatial dimensions  
31 by  $8\times$  and the temporal dimension by  $4\times$ . Each DiT block is similar to the one in Open-Sora [19],  
32 which consists of a 3D self-attention layer running on all video tokens, a cross-attention layer between  
33 video tokens and T5 text embeddings [32] of the input prompt, and an MLP.  
34 The base model adopts the rectified flow training objective [21, 24]. We mostly follow the design  
35 choice of Stable Diffusion 3 [7], e.g., Logit-Normal distribution of noise levels.

36 **SFT** fine-tunes the pre-trained model on the 55k high-quality dataset described in Appendix A.1 for  
37 5k iterations. We did not observe a clear difference between full-model and LoRA [16] fine-tuning,  
38 and thus choose LoRA fine-tuning as it is more efficient.

**VanillaDPO** follows the common direct preference optimization (DPO) practice for video diffusion models [22, 34, 39]. We randomly select 30k text prompts from the curated dataset, and generate 2 videos of 5s per prompt to obtain a comparison pair. We then ask human annotators to choose a better video or a tie by considering text alignment, visual quality, and temporal consistency. 2 labelers are assigned to a pair, with a reviewer to correct any potential errors. This leads to around 10k winning-losing pairs after removing ties. Indeed, human preferences in video are influenced by multiple, sometimes inversely correlated, factors, making it hard to obtain a clear preference.

**StructuralDPO** uses the same 30k prompts as VanillaDPO to construct comparison pairs. For a text prompt, we sample a guidance level  $\eta \sim \mathcal{U}(0.65, 0.8)$ , and 2 Gaussian noises from 2 random seeds. Then, we add the noise to the ground-truth video corresponding to the text prompt according to  $\eta$ , and denoise from it to generate a video. We perform the same human preference annotation process, again leading to around 10k non-tie data pairs.

We have tried using different  $\eta$  when generating the video pair. Usually, the video generated with more guidance is preferred over the other one (e.g., has fewer artifacts). However, we observe that the model quickly achieves a low DPO loss when trained on this data, yet the quality of generated videos is not improved. We hypothesize that different  $\eta$  leads to statistical differences in generated videos, and the model learns shortcuts using such cues instead of truly understanding human preferences.

**DPO training hyper-parameters.** Both VanillaDPO and StructuralDPO adopt the Flow-DPO loss with a constant  $\beta$ , as prior work shows it achieves the best performance [22]. Following prior works [22, 39], we use  $\beta = 500$  and apply LoRA [16] with rank 128 to fine-tune the video model. We train with the AdamW optimizer [25] and a global batch size of 256 for 1k steps. The peak learning rate is set to  $1 \times 10^{-5}$  and it is linearly warmed up from 0 in the first 250 steps. A gradient clipping of 1.0 is applied to stabilize training. We implement all models using PyTorch [28] and conduct training on 64 NVIDIA A100 GPUs, which takes around 16 hours.

**Inference hyper-parameters.** We use the rectified flow sampler [24] with 40 sampling steps and a classifier-free guidance [12] scale of 8 to generate horizontal videos of  $512 \times 288$  resolution. A timestep shifting [7] of 5.66 is applied to further improve results.

### A.3 DenseDPO Implementation Details

For a fair comparison with baselines, we only label dense human preferences on 10k randomly sampled video pairs from the StructuralDPO training data, which costs a similar amount of human annotation time compared to labeling 30k global binary preferences. The segment length  $s$  is set to 1s. Overall, more than 80% of video pairs have at least 1 non-tie segment and can be used in DPO training, greatly improving the data efficiency over using global preferences. Since the DPO loss is now averaged over all non-tie tokens within a training video pair, we sample video clips that have more than 20% non-tie segments to avoid a small effective batch size. All other training and inference hyper-parameters are the same as DPO baselines.

### A.4 Evaluation Datasets

Since our main focus is improving the temporal quality of the pre-trained video model and VanillaDPO, we evaluate on two motion-rich prompt sets.

**VideoJAM-bench** [3] contains 128 prompts focusing on real-world scenarios with challenging motion, ranging from human actions to physical phenomena.

**MotionBench** collects more diverse prompts from existing prompt sets [18, 31, 38]. We run GPT-4o to select prompts with dynamic actions as described in Appendix A.1, resulting in 419 prompts. This prompt set contains more diverse scenes, subjects, and action types.

Note that, to ensure a fair comparison between methods, we use the original text prompt without any prompt rewriting or extension process.

### A.5 Evaluation Metrics

Inspired by prior works [22, 39], we identify three key aspects in text-to-video generation: visual quality, text alignment, and motion quality. Since VanillaDPO has the motion bias issue that leads

to visually pleasing videos with reduced motion strength, we want to evaluate both the smoothness (i.e., temporal consistency) and the strength (i.e., dynamic degree) of the video in a disentangled way. Thus, we cannot use metrics like VideoReward [22] as it only contains a “motion quality” dimension.

**VBench** [17] is a comprehensive benchmark that tests different aspects of a video generation model. When testing on custom prompt sets, it supports six dimensions covering the visual quality, temporal consistency, and dynamic degree. Following the official evaluation protocol, we run each model to generate 5 videos using each prompt with 5 random seeds.

**VisionReward** [39] is a state-of-the-art video reward model that fine-tunes a pre-trained Vision Language Model (VLM) [15] on large-scale human video preference data. It breaks down the human preference into 64 aspects, which can be categorized into 9 dimensions. We take the “Alignment” dimension as text alignment, merge “Composition”, “Quality”, “Fidelity” dimensions into visual quality, merge “Stability”, “Physics”, “Preservation” dimensions into temporal consistency, and take the “Dynamic” dimension as dynamic degree to report in the final result.

## A.6 VLM-based Preference Labeling

**VLM labelers.** We aim to evaluate the effectiveness of existing VLMs in video preference learning. This includes both off-the-shelf models and models fine-tuned for the preference prediction task. For fine-tuned VLMs, we input both videos and take the one with a higher score as the winning data:

- *VideoScore* [11] is fine-tuned on human-labeled scores on 1-3s short videos. We average all five dimensions of its output as the overall score of a video;
- *LiFT* [38], *VideoReward* [22], and *VisionReward* [39] are all fine-tuned on >5s long videos generated by modern video models. We average their output dimensions or take the overall dimension (if presented) as the final score of a video.

For off-the-shelf VLMs, we take *GPT o3* [1] as we find it to outperform GPT-4o [2] due to the visual reasoning capability. We follow the official guide<sup>1</sup> to prompt the model as follows:

### GPT o3 Video Preference Labeling Template

Please help me compare two videos generated by our text-to-video diffusion model. I will provide you with frames sampled from the two videos. The two videos are structurally similar (e.g. global layout and motion are similar), so I only want to compare their details. Please assess which one has a higher quality, i.e., the video that contains fewer artifacts. Pay close attention to visual artifacts such as:

- Additional fingers or legs, deformed human limbs, morphing human faces or body parts;
- Blurry or distorted objects, slight motion blur is fine, but the object should not be completely distorted;
- Abrupt changes in the object, such as objects appearing/disappearing unexpectedly, or anything that should not happen in the real world, e.g., rigid object deforming or melting.

Please only answer “tie” (two videos have equal quality), “first” (the first video has fewer artifacts), and “second” (the second video has fewer artifacts), followed by a simple explanation.

Be conservative in your answer. If you see similar amounts of artifacts in both videos, please choose “tie”. Only select “first” or “second” when one video is clearly better than the other.

These are frames from the first video, sampled at 8 FPS:

Video 1 frames

These are frames from the second video, sampled at 8 FPS:

Video 2 frames

Please compare the two videos and tell me which one is better.

To improve accuracy, we apply a self-consistency check by reversing the order of Video 1 and Video 2. If the prediction on both orders are the same, we keep it. Otherwise, we treat it as a tie. This simple strategy improves the accuracy on short segments by around 10%. We note that there might be better strategy in prompt construction, such as concatenating paired video frames side-by-side, or organizing frames into a grid. We leave further investigations for future works.

<sup>1</sup>[https://cookbook.openai.com/examples/gpt\\_with\\_vision\\_for\\_video\\_understanding](https://cookbook.openai.com/examples/gpt_with_vision_for_video_understanding)



Finally, we design *GPT o3 Segment* that partitions long videos into short 1s segments to process separately. This gives the dense preference labels compatible with our DenseDPO framework. To obtain a global preference for the entire long video, we simply apply majority voting.

**Preference prediction setup.** Prior work [22] pointed out that existing VLMs excel at processing short videos, while falling short on long videos. Therefore, we evaluate two cases:

- *Short Segment* that predicts human preferences on 1s clips. We report the accuracy on the 10k dense human preference labels used in DenseDPO training;
- *Long Video* that predicts human preferences on the entire video, except for GPT o3 Segment that aggregates segment-level results. We report the accuracy on the 30k binary human preference labels used in StructuralDPO training, which is a superset of the previous case.

When calculating the prediction accuracy, we skip tie labels and only compute results on segments or videos with non-tie ground-truth preference labels.

**DPO training setup.** We apply StructuralDPO on binary preference labels produced by GPT o3 and VisionReward as it achieves the highest accuracy. We also apply DenseDPO on dense preference labels produced by GPT o3 Segment. Notably, we run VLMs to label video pairs generated from *all* 55k training data to explore the limit of automatic preference learning performance.

## B Caveat of Guided Sampling

### B.1 Analyzing the Learning Signal

As discussed in Sec. 3.2 of the main text, guided sampling is attractive since it fixes the structure in the preference pair. This neutralizes the motion bias between videos and focuses the comparison on visual artifacts. However, StructuralDPO with guided sampling achieves inferior results. Analyzing its learning signal reveals that it can paradoxically push the model to “*unlearn*” the real data distribution. Intuitively, this happens because the learning signal from a losing sample typically dominates over the winning one in those regions of a video, which correspond to the real data distribution. In this section, we investigate this phenomenon and discuss potential remedies to the issue.

For our analysis, we will use the original DPO notation [36] to simplify the exposition and emphasize that the argument applies for the most general diffusion DPO setup. Diffusion DPO training loss is formulated as:

$$\mathcal{L}(\theta) = -\mathbb{E}_{(\mathbf{x}_0^w, \mathbf{x}_0^l) \sim \mathcal{D}, t \sim \mathcal{U}(0, T), \mathbf{x}_t^w \sim q(\mathbf{x}_t^w | \mathbf{x}_0^w), \mathbf{x}_t^l \sim q(\mathbf{x}_t^l | \mathbf{x}_0^l)} \left[ \log \sigma \left( -\beta T \left( \right. \right. \right. \quad (1)$$

$$\left. + \mathbb{D}_{\text{KL}}(q(\mathbf{x}_{t-1}^w | \mathbf{x}_{0,t}^w) \| p_\theta(\mathbf{x}_{t-1}^w | \mathbf{x}_t^w)) - \mathbb{D}_{\text{KL}}(q(\mathbf{x}_{t-1}^w | \mathbf{x}_{0,t}^w) \| p_{\text{ref}}(\mathbf{x}_{t-1}^w | \mathbf{x}_t^w)) \right. \quad (2)$$

$$\left. - \mathbb{D}_{\text{KL}}(q(\mathbf{x}_{t-1}^l | \mathbf{x}_{0,t}^l) \| p_\theta(\mathbf{x}_{t-1}^l | \mathbf{x}_t^l)) + \mathbb{D}_{\text{KL}}(q(\mathbf{x}_{t-1}^l | \mathbf{x}_{0,t}^l) \| p_{\text{ref}}(\mathbf{x}_{t-1}^l | \mathbf{x}_t^l)) \right) \Big], \quad (3)$$

where  $(\mathbf{x}_0^w, \mathbf{x}_0^l)$  is the winning-losing preference pair,  $T$  is the number of diffusion steps,  $t \sim \mathcal{U}(0, T)$  is the noise level distribution, and  $q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t | \alpha_t \mathbf{x}_0, \sigma_t I)$ . In Diffusion-DPO [36], the objective is further simplified to:

$$\mathcal{L}(\theta) = -\mathbb{E}_{(\mathbf{x}_0^w, \mathbf{x}_0^l) \sim \mathcal{D}, t \sim \mathcal{U}(0, T), \mathbf{x}_t^w \sim q(\mathbf{x}_t^w | \mathbf{x}_0^w), \mathbf{x}_t^l \sim q(\mathbf{x}_t^l | \mathbf{x}_0^l)} \left[ \log \sigma \left( -\beta T \omega(t) \left( \right. \right. \right. \quad (4)$$

$$\left. \|\epsilon^w - \epsilon_\theta(\mathbf{x}_t^w, t)\|_2^2 - \|\epsilon^w - \epsilon_{\text{ref}}(\mathbf{x}_t^w, t)\|_2^2 - \|\epsilon^l - \epsilon_\theta(\mathbf{x}_t^l, t)\|_2^2 + \|\epsilon^l - \epsilon_{\text{ref}}(\mathbf{x}_t^l, t)\|_2^2 \right) \Big], \quad (5)$$

where  $\omega(t)$  is a weighting function. Let’s denote:

$$\Delta_\theta^w = \|\epsilon^w - \epsilon_\theta(\mathbf{x}_t^w, t)\|_2^2 \quad \Delta_\theta^l = \|\epsilon^l - \epsilon_\theta(\mathbf{x}_t^l, t)\|_2^2 \quad (6)$$

$$\Delta_{\text{ref}}^w = \|\epsilon^w - \epsilon_{\text{ref}}(\mathbf{x}_t^w, t)\|_2^2 \quad \Delta_{\text{ref}}^l = \|\epsilon^l - \epsilon_{\text{ref}}(\mathbf{x}_t^l, t)\|_2^2 \quad (7)$$

Then, the loss gradient is:

$$\nabla_\theta \mathcal{L}(\theta) = -\nabla_\theta \mathbb{E} \left[ \log \sigma \left( -\beta T \omega(t) \cdot (\Delta_\theta^w - \Delta_{\text{ref}}^w - (\Delta_\theta^l - \Delta_{\text{ref}}^l)) \right) \right] \quad (8)$$

$$= \mathbb{E} \left[ (1 - \sigma(\cdot)) \beta T \omega(t) \cdot \nabla_\theta (\Delta_\theta^w - \Delta_{\text{ref}}^w - (\Delta_\theta^l - \Delta_{\text{ref}}^l)) \right] \quad (9)$$

$$= \mathbb{E} \left[ C \cdot \nabla_\theta (\Delta_\theta^w - \Delta_\theta^l) \right], \quad (10)$$

151 where  $C = (1 - \sigma(\cdot))\beta T\omega(t)$ .

152 One can show that  $C > 0$  since  $\sigma(\cdot) \in (0, 1)$  and  $\beta, T, \omega(t) > 0$ . Since  $C > 0$ , the per-pixel  
 153 direction of the incoming gradient w.r.t. the model outputs is determined entirely by the interplay  
 154 between  $\nabla_{\theta}\Delta_{\theta}^w$  and  $\nabla_{\theta}\Delta_{\theta}^l$ . This fact becomes crucial for StructuralDPO with guided sampling for  
 155 the following reason.

156 In guided sampling (see Algo. 2 of the main text), we generate a winning/losing sample  $\mathbf{x}_n^* =$   
 157  $(1 - \eta)\mathbf{x}_0 + \eta\epsilon^*$  using a real video  $\mathbf{x}_0$ . This makes them carry similar structure, which means that the  
 158 winning and losing samples share many pixels. Moreover, these shared pixels normally correspond to  
 159 the ground-truth data distribution. Let’s split the pixels into two sets,  $\mathcal{I}_{\text{same}}$  and  $\mathcal{I}_{\text{unique}}$ :

$$\mathcal{I}_{\text{same}}(\mathbf{x}^w, \mathbf{x}^l) = \{p \mid \mathbf{x}_0^w[p] \approx \mathbf{x}_0^l[p] \approx \mathbf{x}_0[p]\}, \quad (11)$$

$$\mathcal{I}_{\text{unique}}(\mathbf{x}_0^w, \mathbf{x}_0^l) = \{p \mid p \notin \mathcal{I}_{\text{same}}\}, \quad (12)$$

160 where  $p$  is a pixel location (e.g., typically, triplet  $(i, j, k)$  denoting the frame, height, width indices).  
 161 In this way,  $\mathcal{I}_{\text{same}}$  stores pixel locations which remained intact during the forward diffusion process  
 162 of guided sampling and subsequent denoising with  $\mathbf{G}_{\theta}$ . Fig. 2 of the main text and Fig. 6 illustrate  
 163 this: many pixels in the winning and losing samples are identical and correspond to the original  
 164 ground-truth video. In this way, one can argue that  $\mathcal{I}_{\text{same}}$  corresponds to the real data distribution.

165 Let’s denote:

$$\Delta_{(\cdot)}^*[\mathcal{I}] = \|\epsilon^*[\mathcal{I}] - \epsilon_{(\cdot)}^*(\mathbf{x}_t^*, t)[\mathcal{I}]\|_2^2, \quad (13)$$

166 i.e., the diffusion loss in particular pixel locations  $\mathcal{I}$ . Now, if  $\Delta_{\theta}^w[\mathcal{I}_{\text{same}}] < \Delta_{\theta}^l[\mathcal{I}_{\text{same}}]$  (meaning that  
 167 the diffusion error in ground-truth pixel locations of a losing sample is higher than that of the winning  
 168 one), then the gradient will be dominated by the negative contribution of  $\Delta_{\theta}^l[\mathcal{I}_{\text{same}}]$ , and the model  
 169 will be *unlearning* real data distribution. Turns out, this is exactly what is happening in practice:

- 170 • Prior to DPO, the model undergoes extensive supervised fine-tuning (SFT), so it is rea-  
 171 sonable to expect that at initialization, we have  $\mathbb{D}_{\text{KL}}(q(\mathbf{x}_{t-1}^w \mid \mathbf{x}_{0,t}^w) \parallel p_{\theta}(\mathbf{x}_{t-1}^w \mid \mathbf{x}_t^w)) <$   
 172  $\mathbb{D}_{\text{KL}}(q(\mathbf{x}_{t-1}^l \mid \mathbf{x}_{0,t}^l) \parallel p_{\theta}(\mathbf{x}_{t-1}^l \mid \mathbf{x}_t^l))$ , meaning that the model is closer in distribution  
 173 to winning pairs and interprets the entire image as unlikely when there are artifacts  
 174 present in some part of it. The condition  $\mathbb{D}_{\text{KL}}(q(\mathbf{x}_{t-1}^w \mid \mathbf{x}_{0,t}^w) \parallel p_{\theta}(\mathbf{x}_{t-1}^w \mid \mathbf{x}_t^w)) <$   
 175  $\mathbb{D}_{\text{KL}}(q(\mathbf{x}_{t-1}^l \mid \mathbf{x}_{0,t}^l) \parallel p_{\theta}(\mathbf{x}_{t-1}^l \mid \mathbf{x}_t^l))$  implies  $\Delta_{\theta}^w < \Delta_{\theta}^l$  (see Appendix 2 of [36]).
- 176 • The model suffers from an internal distribution shift in the presence of artifacts and its predictions  
 177 in good pixel locations deteriorate in the presence of artifacts. Besides, it might be shifting its  
 178 capacity towards rectifying the artifacts, so the rest of the output suffers.

179 Given the above two observations, we conclude that the model will be unlearning the real data  
 180 distribution in the StructuralDPO setting. While the above analysis is outlined for diffusion models  
 181 with  $\epsilon$ -prediction parametrization [13], it holds for  $v$ -prediction as well (used in both our and many  
 182 contemporary works that align rectified flow models [24]) with argument derivation being basically  
 183 the same. We also emphasize that even marginal domination of the gradient from a losing sample  
 184 over the one from the winning sample results in such “unlearning” behavior.

185 Several prior works observed a similar behavior in DPO on language models [8, 27, 33]. For example,  
 186 DPO-Positive [27] shows that on datasets with short edit distances between winning and losing  
 187 samples, DPO may lead to a reduction in the model’s likelihood on the preferred examples.

188 DenseDPO is a natural way to eliminate this shortcoming of StructuralDPO since it is designed to  
 189 provide dense per-pixel DPO objective and would allow to treat the pixels from  $\mathcal{I}_{\text{same}}$  (i.e., similar  
 190 pixels) as ties, thus removing any loss on them. In the ideal world, we would love to have per-pixel  
 191 preference labels for DenseDPO training, but, as our work demonstrates, even coarse-level temporal  
 192 ones allow us to recover and improve the DPO performance.

193 One can also investigate other strategies to mitigate the issue by taking into account the similarity  
 194 between pixels (e.g., uncertainty-based or margin-aware DPO [14, 23]) or reformulating the DPO in  
 195 some novel way without maximizing the loss of losing samples. We leave this for future work.

## 196 B.2 Empirical Verification

197 While the previous section presents our theoretical argument, we must verify empirically whether  
 198 the assumption regarding the dominance of the losing sample’s loss over the winning sample indeed  
 199 holds. Here, we address this question through empirical analysis using the Flux-dev [20] model.  
 200 We deliberately chose a popular, open-source  $v$ -prediction model to ensure our conclusions remain  
 201 general and reproducible rather than specific to our internal video model.

202 We constructed a synthetic dataset containing controlled artifacts to facilitate this analysis. Specifi-  
 203 cally, we selected 5,195 real-world images with a resolution of  $1024^2$  and artificially corrupted them  
 204 by applying blur to their central patches. Each image is encoded using the FluxAE encoder, resulting  
 205 in a latent tensor of dimensions  $128 \times 128 \times 16$ . We then applied increasing levels of corruption to  
 206 the central  $32 \times 32$  patch (representing 6.25% of the image) with the following blur intensities:

- 207 1. No corruption (0% blur)
- 208 2. Blur with  $k = 2$  (6.25% of the patch size)
- 209 3. Blur with  $k = 4$  (12.5% of the patch size)
- 210 4. Blur with  $k = 8$  (25% of the patch size)
- 211 5. Blur with  $k = 16$  (50% of the patch size)

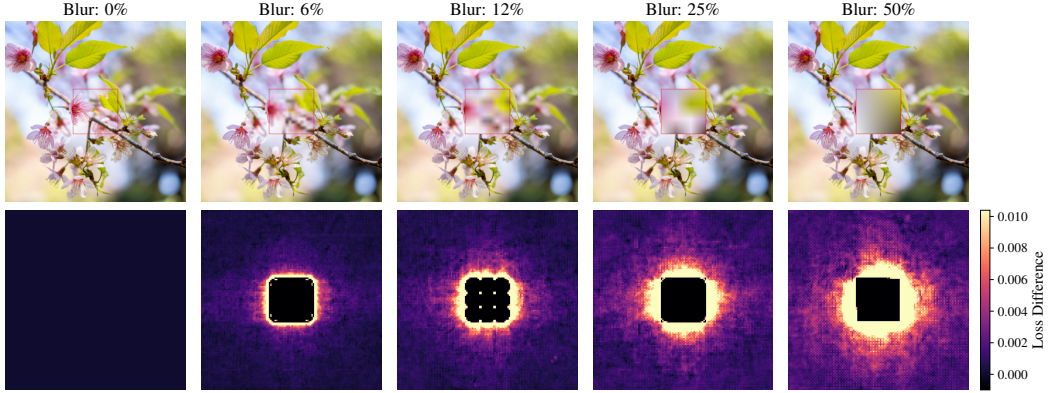


Figure 1: **Visualization of losing sample loss dominance in *uncorrupted* regions.** Top: example images with progressively increasing blur in the central region (note, that the visualization corresponds to an equally-corrupted RGB image, rather than the decoded corrupted latent tensor). Bottom: the per-pixel loss difference  $\delta_{\mathcal{L}}$  between losing and winning samples averaged across latent channels. Positive values indicate the dominance of losing sample losses, driving the model to unintentionally degrade predictions in artifact-free areas. For clarity, we clamp maximum values in the visualization to the maximum loss difference observed in uncorrupted regions, preventing extreme outliers from dominating the heatmap.

212 The resulting dataset comprises five image variants, progressively more corrupted, visualized in Fig. 1  
 213 (top). This setup allows separate measurement of losses in corrupted and uncorrupted regions.  
 214 Subsequently, we randomly sample a timestep  $t \sim \mathcal{U}[0, 1]$ , estimate velocities  $\mathbf{v}_{\text{Flux}}(\mathbf{x}_t^w, t)$  and  
 215  $\mathbf{v}_{\text{Flux}}(\mathbf{x}_t^l, t)$ , and compute per-pixel losses (averaged over the 16 latent channels  $C$ ):

$$\mathcal{L}^w = \frac{1}{C} \sum_{c=1}^C (\mathbf{v}_{\text{Flux}}(\mathbf{x}_t^w, t)[c] - (\epsilon - \mathbf{x}_0^w)[c])^2 \quad (14)$$

$$\mathcal{L}^l = \frac{1}{C} \sum_{c=1}^C (\mathbf{v}_{\text{Flux}}(\mathbf{x}_t^l, t)[c] - (\epsilon - \mathbf{x}_0^l)[c])^2, \quad (15)$$

216 where  $(\cdot)^2$  denotes element-wise squaring. Next, for each sample, we calculate the loss difference  
 217  $\delta_{\mathcal{L}} \in \mathbb{R}^{h \times w}$  as follows:

$$\delta_{\mathcal{L}} = \mathcal{L}^l - \mathcal{L}^w. \quad (16)$$

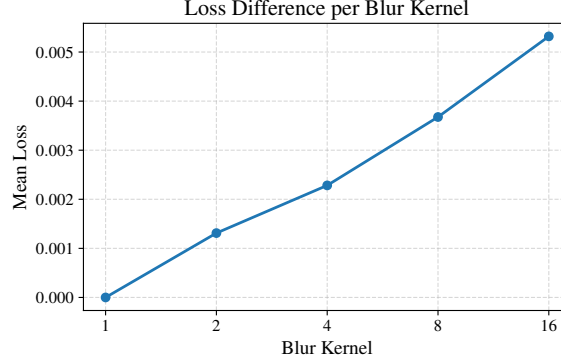


Figure 2: Average loss difference  $\text{mean}(\delta_{\mathcal{L}})$  in *uncorrupted regions* as a function of artifact severity. Increasing blur severity amplifies the losing sample’s negative learning signal in good (uncorrupted) regions, illustrating the risk of inadvertently unlearning correct predictions.

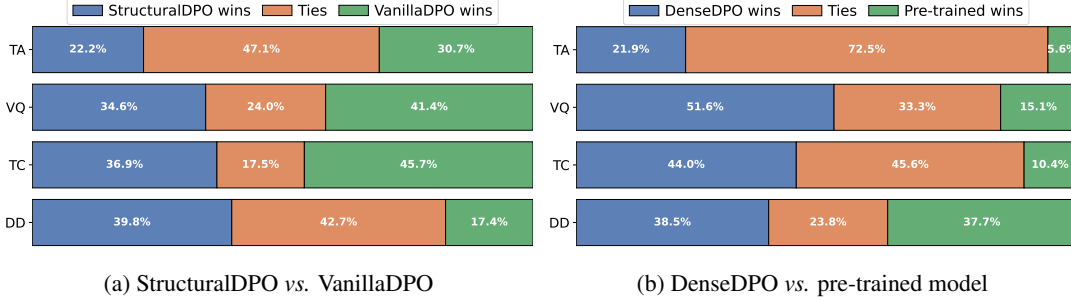


Figure 3: **Human evaluation** on the VideoJAM-bench dataset. TA, VQ, TC, DD stand for text alignment, visual quality, temporal consistency, and dynamic degree.

218  $\delta_{\mathcal{L}}$  indicates the extent to which the losing sample’s loss surpasses that of the winning sample. If  
 219 this dominance occurs in  $\mathcal{I}_{\text{same}}$ , it implies the model is *unlearning* these areas due to the negative  
 220 contribution from the losing sample.

221 To reduce the variance, loss computations use the same noise for corrupted and uncorrupted images.  
 222 Fig. 1 (top) illustrates a representative example of corruption. Visualizations in Fig. 1 (bottom) clearly  
 223 demonstrate the loss dominance of the losing samples in the *uncorrupted regions*. As we maximize  
 224 the loss of the losing sample and minimize that of the winning sample, this results in the model  
 225 increasing loss in uncorrupted image regions when trained with DPO to mitigate blurring artifacts.

226 Fig. 2 provides a quantitative analysis, demonstrating how increasing artifact severity intensifies the  
 227 negative learning signal from losing samples in uncorrupted image areas.

## 228 C Additional Experimental Results

### 229 C.1 DPO with Human Labels

230 **Human evaluation.** Fig. 3 presents additional user study on VideoJAM-bench. Fig. 3a shows that  
 231 StructuralDPO outperforms VanillaDPO in dynamic degree as it performs DPO on video pairs with  
 232 similar motion. Yet, it underperforms in other dimensions. Fig. 3b shows that DenseDPO consistently  
 233 beats the pre-trained model in all dimensions.

234 **Qualitative results.** We show more videos generated by DenseDPO in Fig. 4. Fig. 5 compares  
 235 DenseDPO with baselines. Overall, DenseDPO aligned model generates videos with high visual  
 236 quality, rich motion, and precise text alignment. **Please check out the website in our supplementary**  
 237 **material for video results.**



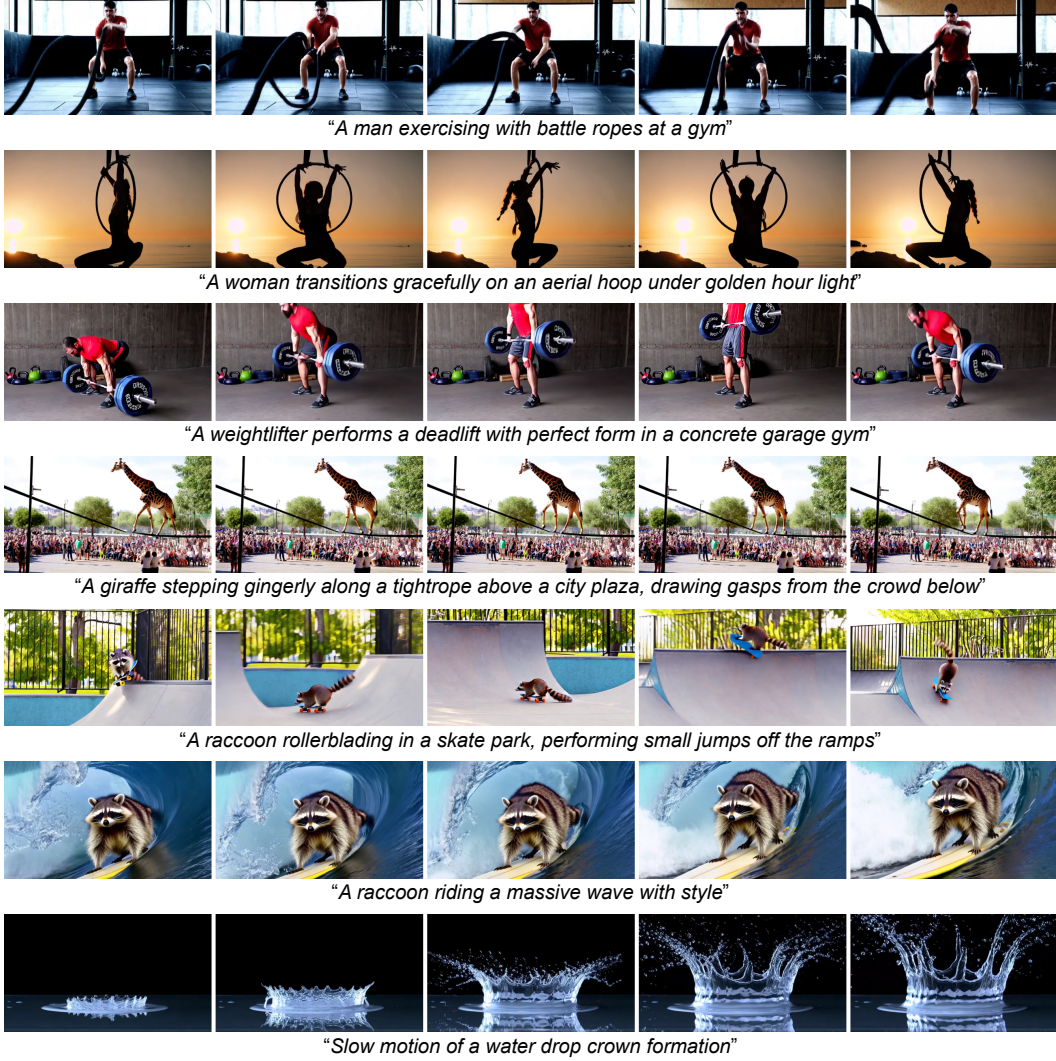


Figure 4: **Text-to-video results with our DenseDPO aligned model.** Here, we show generation of challenging human activities, novel animal actions, and physical phenomena. **Please check out the website in our supplementary material for video results.**

## 238 C.2 DPO with VLM Labels

239 **Motion bias in VLMs.** As discussed in the main paper, there is a motion bias in human preference  
 240 annotation—human labelers tend to favor artifact-free slow-motion clips over dynamic clips with  
 241 artifacts. Ge et al. [9] pointed out that video metrics such as FVD [35] are also biased towards  
 242 per-frame visual quality rather than temporal motion quality. Here, we study whether more advanced  
 243 VLM-based video reward models also suffer from this issue. We test two state-of-the-art models,  
 244 VisionReward [39] and VideoReward [22]. We randomly sample 10k videos from the VanillaDPO  
 245 training data, each having 121 frames (5s). For each video, we construct a static version of it by  
 246 duplicating one frame of it, and we compare this static video with the original video.

247 Tab. 1 presents the winning rate of static vs. original video. Surprisingly, VisionReward favors  
 248 static videos over original videos with a sizable gap, indicating a clear motion bias. In contrast,  
 249 VideoReward prefers original videos in around 80% of cases. We note that both VisionReward and  
 250 VideoReward output multi-dimensional scores (e.g., visual quality, dynamic degree), and aggregate  
 251 them to predict the binary human preference. VideoReward simply averages all dimensions, and thus  
 252 is not biased to any dimension. VisionReward instead first labels human preferences on video pairs,  
 253 and then learns per-dimension weights via logistic regression. This inevitably inherits the motion

Table 1: **Winning rate of static video vs. original video measured by video reward models.** The static video is constructed by duplicating a frame to the video length, where we tested frame 0, 24, 48, and 96 here. A lower winning rate means the video reward model is more sensitive to motion.

Method	Frame 0 vs. Original video	Frame 24 vs. Original video	Frame 48 vs. Original video	Frame 96 vs. Original video
VisionReward [39]	70.63%	68.28%	69.84%	69.06%
VideoReward [22]	20.33%	17.76%	17.72%	17.91%

Table 2: **Ablation on segment length  $s$  of dense preference labels.** We report VBench [17] and VisionReward [39] metrics on VideoJAM-bench [3]. All models are only trained on 5k videos.

Method	VBench Metrics						VisionReward Metrics			
	Aesthetic Quality	Imaging Quality	Subject Consistency	Background Consistency	Motion Smoothness	Dynamic Degree	Text Alignment	Visual Quality	Temporal Consistency	Dynamic Degree
Pre-trained	54.65	55.85	88.29	91.50	92.40	84.16	0.770	0.192	0.354	0.680
$s = 0.5$	<u>55.57</u>	<b>57.03</b>	<b>89.88</b>	<b>92.76</b>	<u>92.52</u>	<u>84.25</u>	<b>0.811</b>	<b>0.326</b>	<b>0.601</b>	<u>0.643</u>
$s = 1$ (Ours)	<b>55.62</b>	<u>56.13</u>	<u>89.23</u>	<u>92.48</u>	<b>92.99</b>	<b>84.25</b>	<u>0.806</u>	<u>0.305</u>	<u>0.589</u>	<b>0.655</b>
$s = 2$	55.40	56.10	89.04	91.96	92.54	84.15	0.795	0.291	0.557	0.623

bias in human labels. Indeed, Tab. 25 in the VisionReward paper [39] reveals that human preference is *negatively* correlated with object dynamics, while *positively* correlated with temporal smoothness.

These results suggest that it is better to label per-dimension scores and predict them, instead of predicting an overall score. The bias in human preferences will be leaked into the reward model if we train the model to regress it. We note that our analysis is still preliminary. Further investigations similar to [9] are required to fully understand the bias in recent VLM-based video reward models.

**GPT dense preference label.** We visualize some dense preferences predicted by GPT o3 Segment in Fig. 6. Overall, GPT is able to identify obvious artifacts such as distorted faces and deformed limbs. With our carefully designed prompt and self-consistency check, it only predicts a preference when there is a clear difference between two segments. However, it still does not understand complex motion, such as playing tennis and cartwheels. This is partially because GPT o3 Segment only has access to 1s video clips, which is too short to finish these actions.

### C.3 Ablation Study

**Dense label granularity.** We study the impact of the segment length  $s$  in dense preference labels. By default,  $s = 1$  is used in all our experiments. Here, we tested  $s = 0.5$  and  $s = 2$ . Due to the high annotation cost, we only label 5k videos for each segment setting in this study. Tab. 2 compares DenseDPO trained on 5k videos using different  $s$ .  $s = 1$  consistently outperforms  $s = 2$  due to more fine-grained preference annotation. Interestingly,  $s = 0.5$  performs similarly to  $s = 1$ . We hypothesize that this is because  $0.5s$  is too short—a longer context window is needed to assess the temporal aspect of videos. In addition, labeling dense preference at  $s = 0.5$  is  $2\times$  expensive compared to  $s = 1$ . Therefore, we choose to label 1s segments in our experiments.

## D Limitations, Future Works, and Broader Impacts

Similar to prior works [22, 39], we also observed unstable training and reward hacking when fine-tuning the entire model. As a result, we have to rely on LoRA training and early stopping. This is in stark contrast to DPO in large language model, where DPO training is relatively stable. More investigation on diffusion DPO basics is need to resolve this issue.

To mitigate the motion bias in VanillaDPO, we propose guided sampling [26] to generate structurally-similar pairs. However, this reduces the variations in comparison pairs, degrading the DPO performance. We note that image-to-video generation with the same conditioning image is another way to

283 retain similar structure between video pairs. In addition, it allows more variations in the generated  
284 videos, which may improve StructuralDPO performance.

285 Although this work focuses on improving visual quality and consistency while holding motion  
286 dynamics constant, the same methodology can be extended to other dimensions. For example, slightly  
287 perturbing the text prompt during guided synthesis could support better alignment with textual  
288 descriptions. It may even lead to preference labels “for-free”, e.g., the video generated with the  
289 original prompt is often better than the one generated with the perturbed prompt. Similar approaches  
290 have also been studied in [4], where they intentionally distort a video via inversion.

291 **Potential positive societal impacts.** Text-to-video generation is an important task in computer vision.  
292 DenseDPO designs a post-training algorithm that improves the motion quality of pre-trained video  
293 diffusion models, which may potentially benefit other fields such as computer graphics and robotics.  
294 We believe this work will benefit the whole research community and the society.

295 **Potential negative societal impacts.** Since we fine-tune large-scale pre-trained generative models,  
296 our final model still inherits limitations of these base models, such as dataset bias. One can explore  
297 post-training with safety-based preferences to mitigate this issue. Further study on the impact of  
298 selection bias is required for mitigating negative societal impacts that could arise from this work [10].

299 **Safeguards.** For a potential future code release, we will be careful to apply automated content filters  
300 (e.g., NSFW classifiers) to the model output. There are no immediate plans to release model weights.



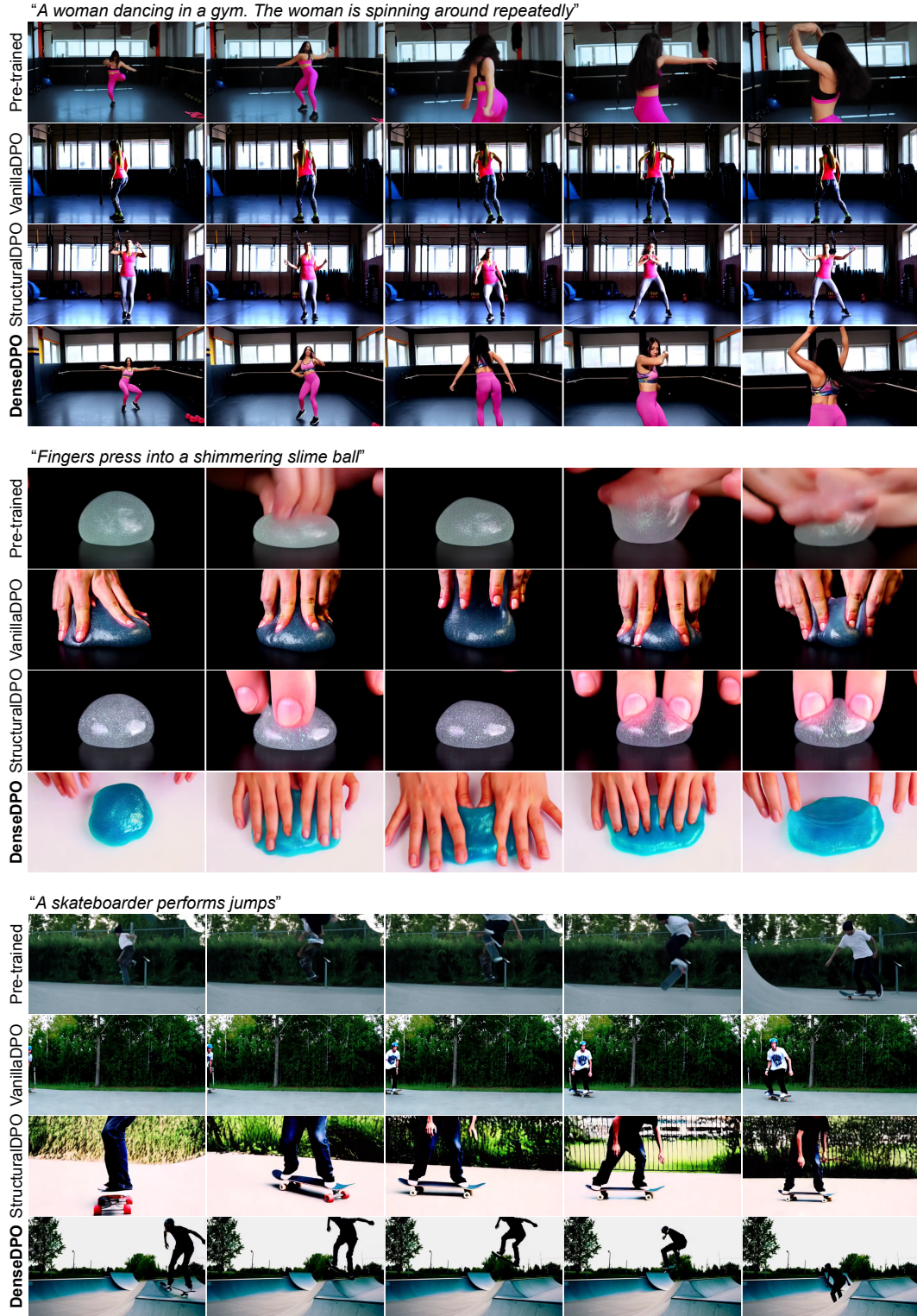


Figure 5: **Qualitative comparison with baselines.** Pre-trained model often generates deformed human body or unnatural object composition. VanillaDPO fixes these artifacts, but with significantly reduced dynamics. StructuralDPO retains the dynamics, but generates oversaturated frames or some artifacts. DenseDPO strikes the best balance over these dimensions. **Please check out the website in our supplementary material for video results.**





Figure 6: **Uncurated samples of GPT o3 predicted dense preference labels.** Each sample consists of a pair of structurally similar videos generated via guided sampling. Videos are sampled at 2 FPS. A top red bar means GPT prefers the first example, and a bottom green bar means GPT prefers the second example, otherwise it is a tie. We highlight some obvious artifacts with blue rectangles.

## E Pytorch-style Pseudo Code for StructuralDPO and DenseDPO

StructuralDPO applies the same Flow-DPO objective as VanillaDPO, which is adopted from [22]:

```
def flow_dpo_loss(model, ref_model, x_0, x_1, c, l, beta):
    """
    # model: Flow model that takes text embeddings c and timestep t
    #         as inputs and predicts velocity c
    # ref_model: Pre-trained model that is frozen
    # x_0: The first video in the pair, shape [B, T, C, H, W]
    # x_1: The second video in the pair, shape [B, T, C, H, W]
    # c: Text embedding of the prompt
    # l: Preference label, shape [B], each item is either +1 or -1
    #     +1 means x_0 is better than x_1, -1 means the other way
    # beta: DPO regularization hyper-parameter
    # returns: Flow-DPO loss value
    """
    # Add noise to videos
    t = logit_normal_sampler(x_0.shape[0])
    noise = torch.randn_like(x_0)
    noisy_x_0 = (1 - t) * x_0 + t * noise
    noisy_x_1 = (1 - t) * x_1 + t * noise

    # Compute velocity prediction loss
    v_0_pred = model(noisy_x_0, c, t)
    v_1_pred = model(noisy_x_1, c, t)
    v_ref_0_pred = ref_model(noisy_x_0, c, t)
    v_ref_1_pred = ref_model(noisy_x_1, c, t)
    v_0_gt = noise - x_0
    v_1_gt = noise - x_1
    model_0_err = ((v_0_pred - v_0_gt) ** 2).mean(dim=[1, 2, 3, 4])
    model_1_err = ((v_1_pred - v_1_gt) ** 2).mean(dim=[1, 2, 3, 4])
    ref_0_err = ((v_ref_0_pred - v_0_gt) ** 2).mean(dim=[1, 2, 3, 4])
    ref_1_err = ((v_ref_1_pred - v_1_gt) ** 2).mean(dim=[1, 2, 3, 4])

    # Compute DPO loss
    diff_0 = model_0_err - ref_0_err # Shape [B]
    diff_1 = model_1_err - ref_1_err # Shape [B]
    inside_term = -0.5 * beta * l * (diff_0 - diff_1)
    loss = -1 * log(sigmoid(inside_term)).mean()
    return loss
```

342 DenseDPO extends the Flow-DPO loss to frame-level (or token-level for latent models):

```
343
344 def flow_dense_dpo_loss(model, ref_model, x_0, x_1, c, l, beta):
345     """
346     # model: Flow model that takes text embeddings c and timestep t
347     #         as inputs and predicts velocity c
348     # ref_model: Pre-trained model that is frozen
349     # x_0: The first video in the pair, shape [B, T, C, H, W]
350     # x_1: The second video in the pair, shape [B, T, C, H, W]
351     # c: Text embedding of the prompt
352     # l: Dense preference label, shape [B, T], can be +1, 0, or -1
353     #     +1 means x_0 is better than x_1, -1 means the other way
354     #     0 means a tie
355     # beta: DPO regularization hyper-parameter
356     # returns: Flow-DPO loss value
357     """
358     # Add noise to videos
359     t = logit_normal_sampler(x_0.shape[0])
360     noise = torch.randn_like(x_0)
361     noisy_x_0 = (1 - t) * x_0 + t * noise
362     noisy_x_1 = (1 - t) * x_1 + t * noise
363
364     # Compute velocity prediction loss
365     v_0_pred = model(noisy_x_0, c, t)
366     v_1_pred = model(noisy_x_1, c, t)
367     v_ref_0_pred = ref_model(noisy_x_0, c, t)
368     v_ref_1_pred = ref_model(noisy_x_1, c, t)
369     v_0_gt = noise - x_0
370     v_1_gt = noise - x_1
371     model_0_err = ((v_0_pred - v_0_gt) ** 2).mean(dim=[2, 3, 4])
372     model_1_err = ((v_1_pred - v_1_gt) ** 2).mean(dim=[2, 3, 4])
373     ref_0_err = ((v_ref_0_pred - v_0_gt) ** 2).mean(dim=[2, 3, 4])
374     ref_1_err = ((v_ref_1_pred - v_1_gt) ** 2).mean(dim=[2, 3, 4])
375
376     # Compute DPO loss
377     diff_0 = model_0_err - ref_0_err # Shape [B, T]
378     diff_1 = model_1_err - ref_1_err # Shape [B, T]
379     inside_term = -0.5 * beta * l * (diff_0 - diff_1)
380     inside_term = inside_term[l != 0] # Only take non-tie frames
381     loss = -1 * log(sigmoid(inside_term)).mean()
382     return loss
383
```

## References

- [1] OpenAI o3. <https://openai.com/index/o3-o4-mini-system-card/>, 2025.
- [2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
- [3] Hila Chefer, Uriel Singer, Amit Zohar, Yuval Kirstain, Adam Polyak, Yaniv Taigman, Lior Wolf, and Shelly Sheynin. VideoJAM: Joint appearance-motion representations for enhanced motion generation in video models. *arXiv preprint arXiv:2502.02492*, 2025.
- [4] Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Juncheng Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengchen Ma, et al. SkyReels-V2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.
- [5] Tsai-Shien Chen, Aliaksandr Siarohin, Willi Menapace, Ekaterina Deyneka, Hsiang-wei Chao, Byung Eun Jeon, Yuwei Fang, Hsin-Ying Lee, Jian Ren, Ming-Hsuan Yang, et al. Panda-70M: Captioning 70m videos with multiple cross-modality teachers. In *CVPR*, 2024.
- [6] Xiaoliang Dai, Ji Hou, Chih-Yao Ma, Sam Tsai, Jialiang Wang, Rui Wang, Peizhao Zhang, Simon Vandenhende, Xiaofang Wang, Abhimanyu Dubey, et al. Emu: Enhancing image generation models using photogenic needles in a haystack. *arXiv preprint arXiv:2309.15807*, 2023.
- [7] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024.
- [8] Duanyu Feng, Bowen Qin, Chen Huang, Zheng Zhang, and Wenqiang Lei. Towards analyzing and understanding the limitations of dpo: A theoretical perspective. *arXiv preprint arXiv:2404.04626*, 2024.
- [9] Songwei Ge, Aniruddha Mahapatra, Gaurav Parmar, Jun-Yan Zhu, and Jia-Bin Huang. On the content bias in fréchet video distance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7277–7288, 2024.
- [10] Google. Safety & fairness considerations for generative models, 2023. URL <https://developers.google.com/machine-learning/resources/safety-gen-ai>.
- [11] Xuan He, Dongfu Jiang, Ge Zhang, Max Ku, Achint Soni, Sherman Siu, Haonan Chen, Abhranil Chandra, Ziyang Jiang, Aaran Arulraj, et al. VideoScore: Building automatic metrics to simulate fine-grained human feedback for video generation. In *EMNLP*, 2024.
- [12] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.
- [14] Jiwoo Hong, Sayak Paul, Noah Lee, Kashif Rasul, James Thorne, and Jongheon Jeong. Margin-aware preference optimization for aligning diffusion models without reference. In *ICLR Workshop*, 2025.
- [15] Wenyi Hong, Weihang Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yan Wang, Yean Cheng, Shiyu Huang, Junhui Ji, Zhao Xue, et al. CogVLM2: Visual language models for image and video understanding. *arXiv preprint arXiv:2408.16500*, 2024.
- [16] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.
- [17] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. VBench: Comprehensive benchmark suite for video generative models. In *CVPR*, 2024.



- [18] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. HunyuanVideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.
- [19] PKU-Yuan Lab and Tuzhan AI etc. Open-Sora-Plan, April 2024. URL <https://doi.org/10.5281/zenodo.10948109>.
- [20] Black Forest Labs. Flux. <https://github.com/black-forest-labs/flux>, 2024.
- [21] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023.
- [22] Jie Liu, Gongye Liu, Jiajun Liang, Ziyang Yuan, Xiaokun Liu, Mingwu Zheng, Xiele Wu, Qiulin Wang, Wenyu Qin, Menghan Xia, Xintao Wang, Xiaohong Liu, Fei Yang, Pengfei Wan, Di Zhang, Kun Gai, Yujiu Yang, and Wanli Ouyang. Improving video generation with human feedback. *arXiv preprint arXiv:2501.13918*, 2025.
- [23] Runtao Liu, Haoyu Wu, Zheng Ziqiang, Chen Wei, Yingqing He, Renjie Pi, and Qifeng Chen. VideoDPO: Omni-preference alignment for video diffusion generation. In *CVPR*, 2025.
- [24] Xingchao Liu, Chengyue Gong, et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.
- [25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.
- [26] 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. In *ICLR*, 2022.
- [27] Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddhartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*, 2024.
- [28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019.
- [29] William Peebles and Saining Xie. Scalable diffusion models with transformers. *ICCV*, 2023.
- [30] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.
- [31] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.
- [32] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 2020.
- [33] Noam Razin, Sadhika Malladi, Adithya Bhaskar, Danqi Chen, Sanjeev Arora, and Boris Hanin. Unintentional Unalignment: Likelihood displacement in direct preference optimization. In *ICLR*, 2025.
- [34] Team Seaweed, Ceyuan Yang, Zhijie Lin, Yang Zhao, Shanchuan Lin, Zhibei Ma, Haoyuan Guo, Hao Chen, Lu Qi, Sen Wang, Feng Cheng, Feilong Zuo, Xuejiao Zeng, Ziyang Yang, Fangyuan Kong, Meng Wei, Zhiwu Qing, Fei Xiao, Tuyen Hoang, Siyu Zhang, Peihao Zhu, Qi Zhao, Jiangqiao Yan, Liangke Gui, Sheng Bi, Jiashi Li, Yuxi Ren, Rui Wang, Huixia Li, Xuefeng Xiao, Shu Liu, Feng Ling, Heng Zhang, Houmin Wei, Huafeng Kuang, Jerry Duncan, Junda Zhang, Junru Zheng, Li Sun, Manlin Zhang, Renfei Sun, Xiaobin Zhuang, Xiaojie Li, Xin Xia, Xuyan Chi, Yanghua Peng, Yuping Wang, Yuxuan Wang, Zhongkai Zhao, Zhuo Chen, Zuquan Song, Zhenheng Yang, Jiashi Feng, Jianchao Yang, and Lu Jiang. Seaweed-7B: Cost-effective training of video generation foundation model, 2025.

- 480 [35] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski,  
481 and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges.  
482 *arXiv preprint arXiv:1812.01717*, 2018.
- 483 [36] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam,  
484 Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using  
485 direct preference optimization. In *CVPR*, 2024.
- 486 [37] Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao,  
487 Jianxiao Yang, Jianyuan Zeng, et al. Wan: Open and advanced large-scale video generative  
488 models. *arXiv preprint arXiv:2503.20314*, 2025.
- 489 [38] Yibin Wang, Zhiyu Tan, Junyan Wang, Xiaomeng Yang, Cheng Jin, and Hao Li. LiFT: Lever-  
490 aging human feedback for text-to-video model alignment. *arXiv preprint arXiv:2412.04814*,  
491 2024.
- 492 [39] Jiazheng Xu, Yu Huang, Jiale Cheng, Yuanming Yang, Jiajun Xu, Yuan Wang, Wenbo Duan,  
493 Shen Yang, Qunlin Jin, Shurun Li, Jiayan Teng, Zhuoyi Yang, Wendi Zheng, Xiao Liu, Ming  
494 Ding, Xiaohan Zhang, Xiaotao Gu, Shiyu Huang, Minlie Huang, Jie Tang, and Yuxiao Dong.  
495 VisionReward: Fine-grained multi-dimensional human preference learning for image and video  
496 generation. *arXiv preprint arXiv:2412.21059*, 2024.
- 497 [40] Hongwei Xue, Tiankai Hang, Yanhong Zeng, Yuchong Sun, Bei Liu, Huan Yang, Jianlong Fu,  
498 and Baining Guo. Advancing high-resolution video-language representation with large-scale  
499 video transcriptions. In *CVPR*, 2022.
- 500 [41] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming  
501 Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. CogVideoX: Text-to-video diffusion  
502 models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.
- 503 [42] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen,  
504 Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, et al. Language model beats  
505 diffusion-tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023.
- 506 [43] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun  
507 Zhou, Tianyi Li, and Yang You. Open-Sora: Democratizing efficient video production for all,  
508 March 2024. URL <https://github.com/hpcaitech/Open-Sora>.