Contrastive Posterior Distillation Sampling for Image Editing

Yoav Orenbach
208847749
orenbach@mail.tau.ac.il

Yasmin Khen
314920489
yasminkhen@mail.tau.ac.il

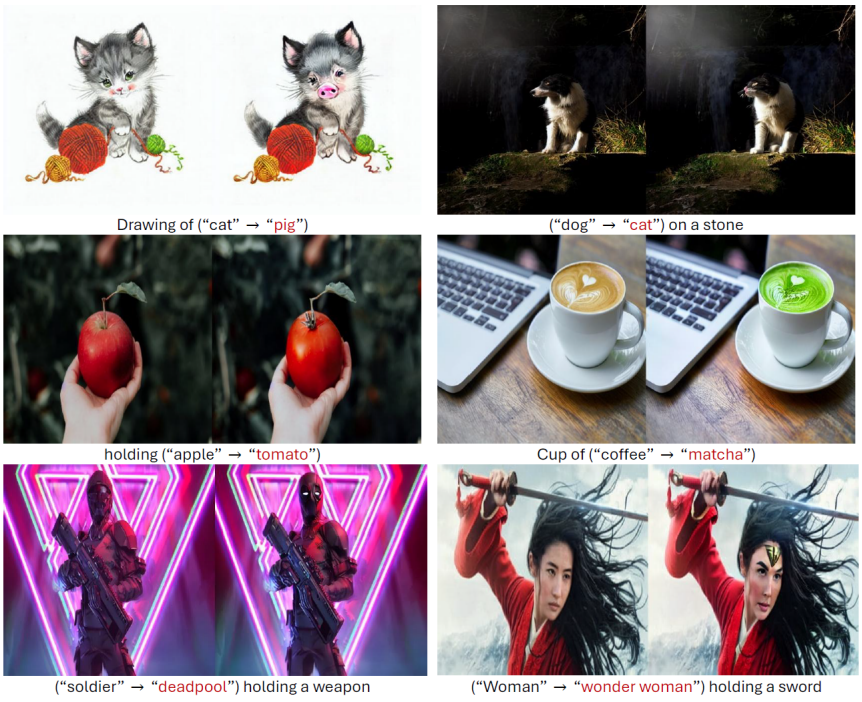

Figure 1: Text conditioned image editing results via Contrastive Posterior Distillation Sampling. **CPDS** achieves a balance between transforming the content of a source image to a target text prompt while keeping the structural elements and details of the source.

Abstract

Text conditioned image editing transforms a given image and text prompt into a desired target prompt. Editing represents a significant challenge, requiring both faithful transformation according to a target prompt while preserving the source image's identity. We introduce Contrastive Posterior Distillation Sampling (CPDS), a novel

method that combines structural consistency preservation from Contrastive Denoising Score (CDS) with the identity-preserving optimization of Posterior Distillation Sampling (PDS). CPDS achieves superior results with up to 50% improvement in perceptual similarity metrics while maintaining alignment with target text prompts. We further enhance CPDS by replacing random patch selection used to compute the contrastive loss with a Meta-Learner network that predicts which feature patches to preserve, improving structural correspondence without increasing inference time. Our approach demonstrates significant qualitative and quantitative improvements over existing state-of-the-art methods across multiple image editing tasks. Project page:
https://github.com/YoavOrenbach/ContrastivePosteriorDistillationSampling

1 Introduction

The ability to edit images based on text prompts has become increasingly important with the advancement of generative models. While Latent Diffusion Models (LDMs) have made significant strides in text-to-image generation task [5, 6, 7], creating photorealistic content from textual descriptions, the challenge of text-conditioned image editing remains particularly difficult. Editing differs from generation in that it requires considerations of both the target text and the original source content, thereby emphasizing two key aspects:

1. Alignment with the target text prompt - transforming the image to match new attributes.
2. Preservation of the source structural details - maintaining the identity and composition of the original.

These dual requirements create a delicate balance that current methods struggle to maintain consistently. When edits significantly alter object characteristics (such as changing species or style), preserving structural consistency becomes especially challenging.

Recent approaches have attempted to address these challenges through different mechanisms. Delta Denoising Score (DDS) [3] - an image editing technique that reduced the noisy gradients inherent in earlier Score Distillation Sampling (SDS) framework [4], has led to better maintaining of background details and sharper editing outputs. Unfortunately, in DDS the structural details of the source image are often neglected and the optimization function of DDS lacks an explicit term for identity preservation.

As preserving structural consistency is recognized as crucial in image manipulation, two new methods have addressed this issue and improved upon the DDS framework, achieving state-of-the-art results and outperforming existing baselines.

The first method, called Contrastive Denoising Score (CDS) [1], integrates the application of Contrastive Unpaired Translation (CUT) loss [8] into pretrained latent diffusion models such as Stable Diffusion [6], within the DDS framework. CDS utilizes the rich spatial information in the self-attention features of LDMs by randomly selecting sets of patches from the same spatial locations to compute the CUT loss, enabling it to preserve structural consistency.

The second method, called Posterior Distillation Sampling (PDS) [2], aims to match the stochastic latents of the source and the optimized target. PDS demonstrated that its process

resembles aligning forward process posteriors of the source and the target, ensuring that the target's generative process trajectory does not significantly deviate from that of the source.

We observe that these methods offer complementary strengths that could be combined to achieve superior results, especially when considering that the PDS optimization function incorporates a term dedicated to preserving the identity of the source, which DDS lacks. This led us to extend the editing capabilities of CDS by integrating the CUT loss into the PDS framework rather than the DDS framework, thus creating a new method called Contrastive Posterior Distillation Sampling (CPDS). CPDS is achieved by using the CDS pipeline, but instead of computing the DDS loss, it computes the PDS loss and continues in the same manner to utilize the spatial information for the CUT loss computation. Qualitative results comparing CPDS to CDS, PDS, and DDS show that CPDS outperforms all other methods, preserving much of the structural details of the original image. Quantitative results show that similarity scores between the image and text prompt for CPDS are on par with other methods, hence indicating that CPDS transformed the content in alignment with a target text prompt. However, the scores for perceptual similarity between the source and generated image show CPDS reaches up to 50% improvement over the other methods.

Moreover, since failure cases can arise from unfavorable random patch selections when computing the CUT loss, we propose to replace it with a more robust method that learns what patches to select from the self-attention layers. Specifically, we tested a Meta-Learner (small fully connected neural network) and 3 machine learning classifiers: Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and a Random Forest as patch selectors. Quantitative results show that a smart patch selector can achieve slightly better structural consistency, while qualitative results remain similar. However, all machine learning selectors had slow inference time when predicting patches, thus affecting the overall image generation time and proving unsuitable for the image editing task. In contrast, the Meta-Learner selector had the same inference time as a random selector but with superior quantitative scores, hence proving to be the more suited option for both CDS and CPDS.

All in all, we make the following contributions:

- A novel method, Contrastive Posterior Distillation Sampling (CPDS), that incorporates the CUT loss from CDS into the PDS framework, achieving significantly better balance between preserving structural details and text alignment.
- A Meta-Learner network that replaces random patch selection with predicted patches, improving perceptual similarity while maintaining inference efficiency.

2 Related Work

The emergence of open-source generative models, particularly Latent Diffusion Models (LDMs), has catalyzed extensive exploration in image editing. We review key methods that form the foundation of our work:

2.1 Delta Denoising Score (DDS)

Score Distillation Sampling (SDS) [4] first proposed using pre-trained text-to-image diffusion models as priors for image generation, leveraging the gradient of the diffusion loss function:

$$\mathcal{L}_{SDS}(\theta; y) = ||\epsilon_\phi^\omega\left(z_t\left(\theta\right), y, t\right) - \epsilon||^2$$

Where $\epsilon \sim \mathcal{N}\left(0, I\right)$ is noise, y is the target prompt, z_t represent a noisy latent, $t \sim \mathcal{N}\left(0, 1\right)$ is a timestep, and $\epsilon_\phi^\omega\left(z_t\left(\theta\right), y, t\right)$ is the predicted noise from the text-conditioned diffusion model.

DDS [3] expands the SDS framework for image editing, utilizing not only the target text prompt y but also a reference pair of image $\hat{z}_t$ and text $\hat{y}$ with the DDS loss being:

$$\mathcal{L}_{DDS}(\theta; y) = ||\epsilon_\phi^\omega\left(z_t\left(\theta\right), y, t\right) - \epsilon_\phi^\omega\left(\hat{z}_t\left(\theta\right), \hat{y}, t\right)||^2$$

2.2 Contrastive Denoising Score (CDS)

CDS [1] incorporates the CUT loss [8] loss into the DDS framework to enhance structural consistency. During the denoising process of the DDS gradient computation, CDS extracts the intermediate features $h_\ell, \hat{h}_\ell$ of self attention layer ℓ. Next, random patches are selected from the feature map h_ℓ, where patches at the corresponding location of feature map $\hat{h}_\ell$ are considered positive, and non-corresponding patches within the feature map serve as negatives. The objective of the CUT loss is to maximize the mutual information between positive patches while simultaneously minimizing the mutual information between negatives, effectively encouraging structural correspondence between source and target.

2.3 Posterior Distillation Sampling (PDS)

PDS [2] approaches identity preservation differently by matching the stochastic latents of the source and target. This ensures that their generative trajectories remain similar despite being guided by different prompts. Denoting the stochastic latents of the source and the target as $\tilde{z}_t^{src}$ and $\tilde{z}_t^{trg}$, the PDS objective function is as follows:

$$\mathcal{L}_{PDS} = \mathbb{E}_{t,\epsilon_{t-1},\epsilon_t}\left[||\tilde{z}_t^{trg} - \tilde{z}_t^{src}||^2\right]$$

This expectation is taken over timesteps and noise variables, so rather than matching noise variables as in SDS and DDS, the stochastic latents of the source and the target are matched via the optimization.

3 Preliminaries

Our work builds upon these foundations to create a more effective image editing approach.

The input requirements for generating an edited image are straightforward:

- A source image

- A source text prompt describing the input
- A target text prompt specifying desired attributes for the output

Preprocessing consists simply of resizing the image and encoding both text prompts for use with a pre-trained LDM, specifically Stable Diffusion v1.4.

For comprehensive evaluation, we collected 250 images of cats along with various images from the LAION 5B dataset [11] to test different image editing tasks, which we detail in Section 5.2.

4 Methods

4.1 Contrastive Posterior Distillation Sampling (CPDS)

Our primary contribution, CPDS, integrates the structural consistency benefits of CDS with the identity preservation of PDS. The key insight is that while PDS provides an explicit optimization term for identity preservation, it lacks the spatial correspondence mechanism that CDS provides through contrastive learning on self-attention features. CPDS follows this process:

1. Compute noise variables $\epsilon_\phi^\omega\left(z_t\left(\theta\right), y, t\right)$ and $\epsilon_\phi^\omega\left(\hat{z}_t\left(\theta\right), \hat{y}, t\right)$ as in DDS.
2. Calculate their posterior means to derive the stochastic latents $\tilde{z}_t^{src}$ and $\tilde{z}_t^{trg}$ for the PDS loss.
3. Extract feature maps $h_\ell, \hat{h}_\ell$ from self-attention layers.
4. Select corresponding patches and compute the CUT loss as in CDS.

By combining these complementary approaches, CPDS achieves both trajectory alignment through PDS and spatial correspondence through the CUT loss. This combination results in significantly improved structural preservation between input and output images while still effectively transforming content according to the target prompt as shown in section 5.1. Figure 2 illustrates the overall pipeline of CPDS.

4.2 Smart Path Selector

A limitation of the CUT loss in both CDS and our CPDS method is its reliance on random patch selection, which can sometimes lead to suboptimal structural correspondence. To address this, we developed a more robust approach to patch selection that learns to select positive patches, hence improving the CUT loss.

We note that instead of replacing the random patch selection, we could have trained an encoder to extract spatial information from the input image as in the original CUT algorithm, however, it is inefficient and we sought a lightweight solution that would maintain inference efficiency. Our primary innovation is a Meta-Learner - a small fully connected neural network that learns to predict which patches to select based on feature similarity patterns.

The Meta-Learner is trained on feature maps obtained during the denoising process. Specifically:

1. The network takes as input features from map h_ℓ.
2. Labels are generated by computing the norm $||h_\ell - \hat{h}||_2^2$ for each patch.
3. These norm values are converted to binary labels using an 80th percentile threshold, where values below the threshold receive a label of 1 (indicating a positive patch).
4. The Meta-Learner learns to predict these labels, effectively identifying which patches have corresponding features in the source and aid the CUT loss objective.

We also explored machine learning classifiers including Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Random Forest as alternative patch selectors. However, while these achieved high accuracy, their inference time proved prohibitively slow for practical use in the image editing pipeline.

Since the feature data is only available during the inference steps of generating the output image, we jointly train the selectors alongside the editing process, saving feature maps and updating the selector every 10 iterations. This approach allows the patch selector to adapt to the specific characteristics of each image editing task while maintaining computational efficiency.

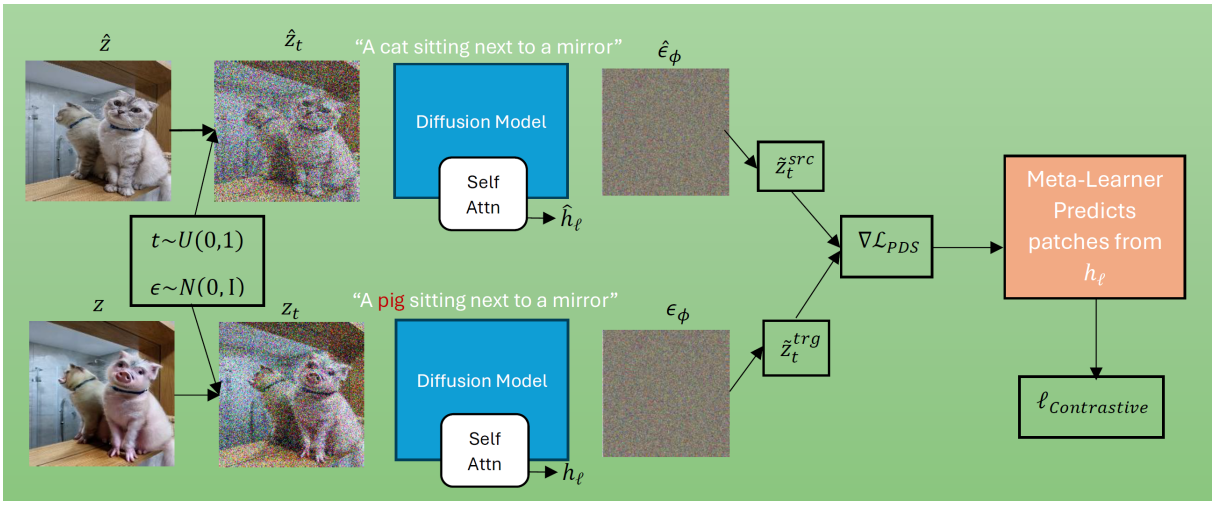

Figure 2: Overall pipeline of CPDS

5 Experiments

We conducted comprehensive experiments to evaluate CPDS against CDS, PDS, and DDS across various image editing tasks, as well as to assess the impact of different patch selectors.

5.1 Qualitative Results

Figure 3 compares the edited outputs of CPDS, CDS, PDS, and DDS on various image editing tasks. All methods successfully transform images according to the target prompt without modifying unrelated regions such as backgrounds. However, DDS, PDS, and CDS frequently generate images with severely deformed structures compared to the source. In

contrast, CPDS successfully preserves the original structural information while accomplishing the desired edits.

The first task of transforming a cat to a pig demonstrates this clearly—the original shape of the cat is well-preserved in CPDS, shows slight distortion in CDS, and is severely altered in PDS and DDS. Similarly, in other tasks, new structures are formed or distorted in DDS, CDS, and PDS outputs, while CPDS maintains consistent structural correspondence with the source.

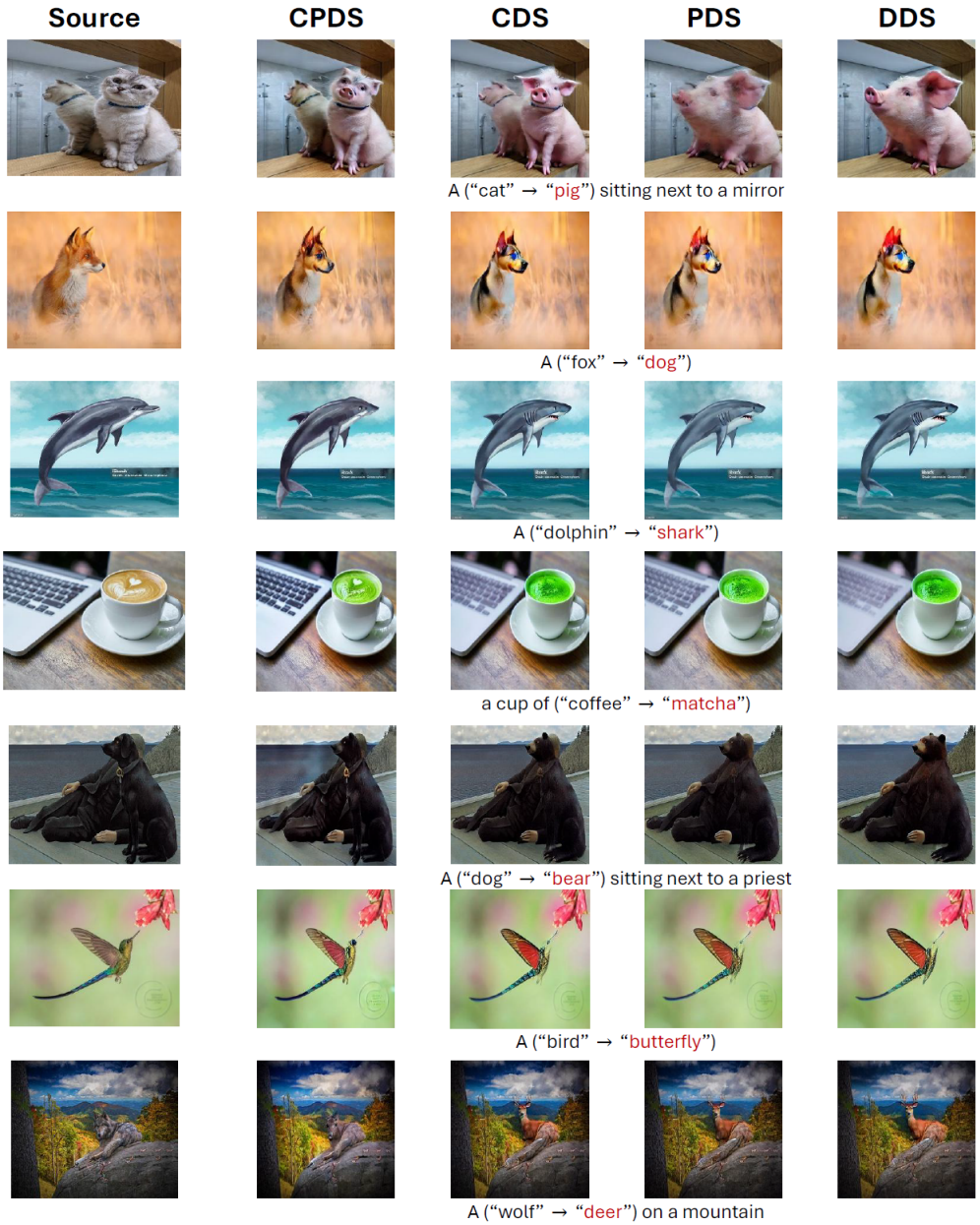

Figure 3: Comparison of CPDS, CDS, PDS, and DDS on different tasks. CPDS demonstrates outstanding performance in retaining structural consistency.

5.2 Quantitative Results

For quantitative evaluation, we collected 250 different images of cats and conducted three tasks: (1) cat → dog, (2) cat → cow, and (3) cat → pig using CPDS, CDS, PDS, and DDS.

We measured performance using two established metrics also used by CDS and PDS:

- CLIP score [9]: Measures the similarity between edited 2D renderings and target text prompts, where a higher score indicates a higher similarity.
- LPIPS score [10]: Measures the perceptual similarity between two images, where a low score indicates better structural preservation.

Table 1 presents the results across all three tasks:

	cat2dog		cat2cow		cat2pig	
	CLIP(↑)	LPIPS(↓)	CLIP(↑)	LPIPS(↓)	CLIP(↑)	LPIPS(↓)
DDS	0.24972	0.0759269	**0.247354**	0.103936	0.247094	0.113857
CDS	0.249663	0.0537139	0.244481	0.0961218	0.246517	0.0987702
PDS	**0.249963**	0.0586451	0.245436	0.10757	**0.247256**	0.101784
CPDS	0.249635	**0.0428247**	0.24435	**0.0558181**	0.246405	**0.0628885**

Table 1: Average scores of DDS, CDS, PDS, and CPDS on 250 images of cats for the cat2dog, cat2cow, cat2pig tasks

The results demonstrate that CPDS significantly outperforms other methods in LPIPS scores, achieving almost 50% improvement in some cases. This indicates substantially better preservation of the source image's structure. Crucially, CPDS maintains competitive CLIP scores, confirming that the target prompt's attributes are still accurately reflected in the generated images. This quantitative evidence supports our claim that CPDS introduces only the minimal necessary changes to the input image while effectively meeting the requirements specified in the target text prompts.

Figure 4 provides a qualitative comparison of these three editing tasks on a representative image from our test set.

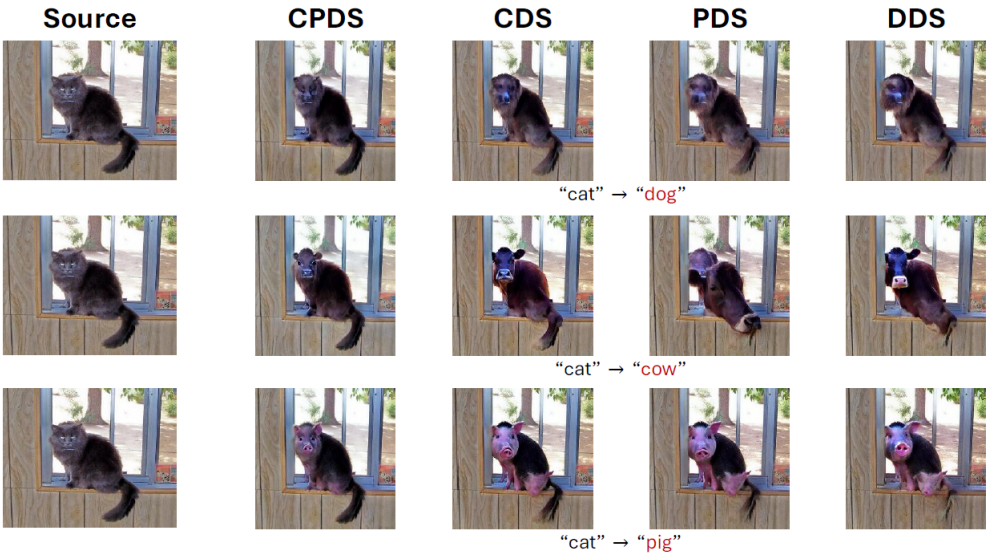

Figure 4: Comparison of CPDS, CDS, PDS, and DDS on the three cat editing tasks.

5.3 Ablation Study

An important factor in both PDS and our CPDS method is the learning rate used during optimization. While CDS and DDS use a learning rate of 0.1, PDS employs a higher rate of 0.8. Since CPDS combines elements from both approaches, we conducted an ablation study to determine the optimal learning rate.

Figure 5: Ablation study on CPDS learning rate effects.

As shown in Figure 5, a learning rate of 0.1 produces minimal changes to the original image, failing to satisfy the target prompt requirements. As the learning rate increases, the desired semantic transformations become more apparent. However, at a learning rate of 1.2, we observed background distortion, suggesting that excessive rates could compromise structural preservation. Based on these findings, we adopted a learning rate of 0.8 for CPDS, balancing transformation effectiveness with structural preservation.

5.4 Patch Selector Results

To evaluate our Smart Patch Selector, we compared different implementations using the cat → pig task with CDS as the base method. Table 2 presents CLIP and LPIPS scores for the

random patch selector (baseline), Meta-Learner, and KNN selector (we did not include the SVM and Random-Forest patch selectors as they had unacceptable inference time).

	cat2dog	
	CLIP(↑)	LPIPS(↓)
Random	**0.247089**	0.0895743
Meta-Learner	0.246973	0.0838054
KNN	0.246875	**0.080417**

Table 2: Patch selectors CLIP and LPIPS score on the cat2pig task

While the quantitative differences appear modest, the random selector achieves the highest CLIP score, while the KNN selector shows the best LPIPS score. This pattern suggests that smart patch selection can enhance structural consistency. However, the critical factor is inference time - the KNN selector, despite its slightly better LPIPS score, introduces significant computational overhead that undermines practical applicability.

In addition, since it is a classification task where the selectors are classifying the feature maps and the top patches are predicted to be positive patches, we also measured the classification accuracy of each selector during training as shown in Figure 6.

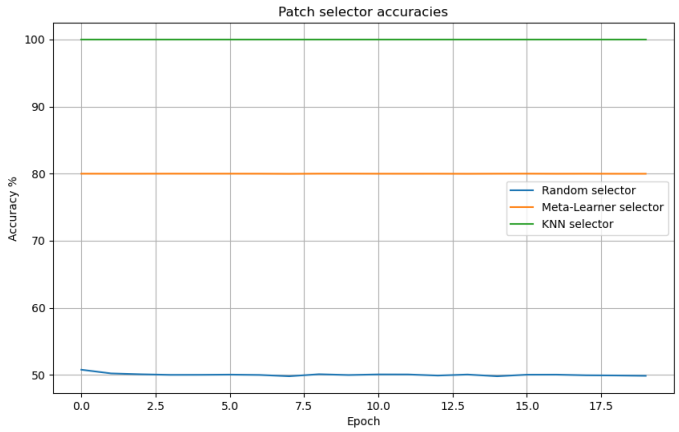

Figure 6: Patch selectors accuracy results for classifying the feature maps as positive/negative.

The accuracy results follow the trend of the CLIP and LPIPS scores. The random selector achieves approximately 50% accuracy (as expected), while the Meta-Learner reaches around 80%, and KNN approaches 100%. Given that our goal is to maintain inference efficiency comparable to random selection while improving structural correspondence, the Meta-Learner represents the optimal compromise between accuracy and computational cost.

6 Conclusion

In this work, we introduced Contrastive Posterior Distillation Sampling (CPDS), a novel approach that combines the structural consistency benefits of CDS with the identity preservation capabilities of PDS. Our comprehensive evaluations demonstrate that CPDS significantly outperforms existing methods in preserving source image structure while effectively implementing target prompt transformations.

CPDS excels at preserving not only background elements but also object features not explicitly specified in the target prompt. While this high degree of preservation might sometimes be excessive for tasks requiring more dramatic transformations, it significantly expands the potential of controllable image editing.

Our secondary contribution, the Meta-Learner patch selector, offers improved structural correspondence over random selection without increasing inference time. By predicting which feature patches to preserve, the Meta-Learner enhances the effectiveness of the CUT loss mechanism in both CDS and CPDS.

Future research directions include developing more sophisticated patch selection mechanisms, exploring adaptive learning rate strategies for different editing tasks, and extending the CPDS framework to other generative models beyond Stable Diffusion. As new approaches to structural identity preservation emerge, they could potentially be integrated with our contrastive learning framework to further enhance editing performance while maintaining the delicate balance between transformation and preservation.

7 Appendix

7.1 Implementation Details

The code for CPDS and our patch selectors is available at:
https://github.com/YoavOrenbach/ContrastivePosteriorDistillationSampling.

Our implementation builds upon the original CDS repository at:
https://github.com/HyelinNAM/ContrastiveDenoisingScore and incorporates elements from the PDS repository at:
https://github.com/KAIST-Visual-AI-Group/PDS.

We will now delve into the specifics of the code changes, integrations, and new additions:

7.2 CPDS Implementation

Since our goal was the integration of the PDS framework, the first thing to do was to add the PDS loss into the CDS source code. Specifically, in the loss.py file we defined an abstract class called $FrameworkLoss$ and made the already defined $DDSLoss$ class inherit from it, and created a new class called $PDSLoss$ which inherits from the $DDSLoss$ class. The inheritance structure allows both losses to share common functionality such as defining a noise input given a latent and timestep, i.e., z_t and t, predicting the noise, i.e., computing $\epsilon_\phi^\omega\left(z_t\left(\theta\right), y, t\right)$, and finally, further processing this noise to get the final prediction. In the case of DDS there is no further processing as $\epsilon_\phi^\omega\left(z_t\left(\theta\right), y, t\right)$, and $\epsilon_\phi^\omega\left(\hat{z}_t\left(\theta\right), \hat{y}, t\right)$ are the variable

used in its optimization function. However, for PDS there is a further processing step, specifically to compute a posterior mean for the noise prediction and getting the stochastic latents $\tilde{z}_t^{src}$ and $\tilde{z}_t^{trg}$ of the PDS optimization function. The abstract class design facilitates future framework integration without code modifications.

Next, to use the desired loss, we followed the original CDS pipeline. In particualr, we replaced the pipelince_cds.py file with a pipeline_cpds.py file that supports the same arguments as CDS, i.e., input image, text prompt, target text prompt, and weights, but also an argument for the type of method configuration desired which could be either: CPDS, CDS, PDS, or DDS. In the pipeline itself, the functions for encoding the text prompt and preparing the latents remained, however, when defining a framework loss, we used the DDS loss or the PDS loss according to the desired method. Namely, DDS and CDS use the DDS loss and PDS and CPDS use the PDS loss. Similarly, the optimizer used is the SGD optimizer but with a learning rate of 0.1 for CDS and DDS, and a learning rate of 0.8 for PDS and CPDS as detailed in section 5.3.

Finally, to perform the desired inference step and generate the image while calling the PDS loss or DDS loss, we created two separate functions: _loss_step and _contrasive_loss_step where the DDS and PDS methods use the _loss_step function to perform the basic loss calculation, and CDS and CPDS use the _contrasive_loss_step to perform both the PDS/DDS loss computation before using its output to compute the CUT loss.

7.3 Patch Selector Implementation

In a similar fashion, to support multiple patch selectors, we created a file called patch_selector.py containing an abstract class called *PatchSelector* with functions relevant for all patch selector types. These are functions to select the patches, functions to train the selectors, and functions to get the selectors' accuracy for evaluation purposes.

We created the derived classes: *RandomPatchSelector* class, a $MetaLearnerPatchSelector$ class, and a $MLPatchSelector$ class with the SVM, KNN, and Random forest classes inheriting from it as they operate in the same manner.

The random patch selector class maintains the original CDS behavior, returning random patches as in the original code, and the training step does not perform any action. The MetaLearner class implements a fully-connected network with two hidden layers optimized using Adam with a learning rate of 0.001. For selecting patches we returned the top K patches of the network prediction following a sigmoid on the output layer, and for the training step we used the Binary cross Entropy loss for this binary classification task. All the Machine learning selectors are implemented using classifiers from the scikit-learn library, such that the patches predicted are the top probabilities chosen by the classifier, and the training step is a regular fit of the classifier given data and labels.

Next, to save all the data (which is the feature maps $h_\ell, \hat{h}_\ell$), we created a $PatchImportanceTracker$ class that saves all features h_ℓ as input data, and converts the norm of the feature maps to binary labels as described in section 4.2.

Finally, to apply it on CDS and CPDS, instead of using the $CUTLoss$ class defined in loss.py, we created an $EnhancedCutLoss$ class that inherits from it. This class gets at initialization the type of patch selector object to be used, and instead of selecting random

patches for computing the CUT loss, it calls the patch selector function to select predicted patches. We note that since the reference and target feature maps could be of varying sizes, they were padded with zeros to enable the classifiers to work with constant input sizes. In addition, since these are training-based methods, every time the Enhanced CUT loss is computed, the feature maps are saved in the $PatchImportanceTracker$ class, and the patch selectors are trained every 10 inference steps. We chose to jointly train as the feature maps are only available during the inference steps of generating the image, and found every 10 iterations to be a sweet spot for overall inference time, data collected, and patches predicted.

7.4 Evaluation Framework

To evaluate the CPDS method and the different patch selectors, we created a new evaluate.py file. Our evaluation framework loads the stable diffusion model, and calls the CPDS pipeline with the desired configuration method (DDS, CDS, PDS, CPDS) or the patch selector (Random, Meta-Learner, KNN, SVM, Random-Forest) using sample images or the cats dataset we collected with pre-made text prompts. The framework computes CLIP and LPIPS scores for each generated image and aggregates results for comparative analysis across methods and patch selectors.

References

[1] Nam, Hyelin and Kwon, Gihyun and Park, Geon Yeong and Ye, Jong Chul. Contrastive Denoising Score for Text-guided Latent Diffusion Image Editing. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). June 2024.

[2] Koo, Juil and Park, Chanho and Sung, Minhyuk. Posterior Distillation Sampling. CVPR 2024.

[3] Amir Hertz, Kfir Aberman, and Daniel Cohen-Or. Delta denoising score. In ICCV, 2023.

[4] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3D using 2D diffusion. In ICLR, 2023.

[5] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125, 1 (2):3, 2022.

[6] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-resolution image ¨ synthesis with latent diffusion models. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 10684–10695, 2022.

[7] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo-Lopes, Burcu Karagol Ayan, Tim

Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In Advances in Neural Information Processing Systems, 2022.

[8] Taesung Park, Alexei A. Efros, Richard Zhang, and JunYan Zhu. Contrastive learning for unpaired image-to-image translation. In European Conference on Computer Vision, 2020.

[9] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In ICML, 2021.

[10] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In CVPR, 2018.

[11] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. Laion-5b: An open large-scale dataset for training next generation image-text models, 2022.