Discriminative Score Function: Turning Pretrained Models into Functional Generative Priors

Junhoo Lee* **Hyeonjin Kim** **Sangbum Han** **Nojun Kwak**
Seoul National University
junhoo.lee@kaist.ac.kr

Abstract

Discriminative models are trained with task-specific objectives such as classification, detection, or representation learning, yet their practical utility depends on generalizing beyond the training task to unseen samples from the same domain. This suggests that a discriminative model may retain an implicit trace of the data manifold beyond its task labels. We ask whether this trace can be exposed as a generative signal without task-specific guidance. We introduce the Discriminative Score Function (DSF), which extracts an image-space generative field from the loss-gradient geometry of a pretrained discriminative model. DSF provides a training-free update rule that moves images toward model-consistent visual regions without retraining or task-specific targets. We apply DSF to diverse pretrained discriminative models, including ResNet classifiers, DETR detectors, and DINOv2 ViT encoders. DSF generates unconditional images from noise and supports conditional guidance, editing, inpainting, and explanation through the same field. DSF-generated samples also provide effective zero-shot quantization calibration when real images or labels are unavailable, preserving feature geometry better than data-free calibration baselines. These results show that discriminative training leaves a generative functional signal for synthesis and calibration. Project page: https://junhoo.me/dsf.

1 Introduction

"What I cannot create, I do not understand."
— Richard P. Feynman [14]

Discriminative models are trained through task objectives such as classification, detection, or representation learning, but their usefulness depends on something broader: they must generalize to unseen samples from the same visual domain. This generalization suggests that a discriminative model captures data-manifold structure beyond the specific task used to train it [31, 32]. Such structure is closely related to the goal of generative modeling, which is to move toward or sample from the data manifold itself. If this view is correct, the model should be able to induce unconditional generation without a task prior or a specified generation objective. We ask whether this implicit manifold structure can be exposed and reused as a standalone generative objective in a general way.

Existing discriminative-to-generative methods typically rely on external or task-specific handles, including class labels, activation objectives, reconstruction losses, architecture-specific statistics such as BatchNorm priors, or auxiliary generative models [17, 29, 30, 50]. These handles already specify what should be produced, reconstructed, or activated. As a result, the generated images do not by themselves show that the discriminative model exposes the data manifold; they also do not provide a general solution for arbitrary discriminative models. This limitation is especially visible beyond

*Now at KAIST.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

classification, where detectors and self-supervised encoders may not provide class probabilities or labels at all. A suitable answer therefore needs a label-free interface that avoids task objectives and external generators while applying across diverse pretrained discriminative models.

We answer this question by introducing the Discriminative Score Function (DSF), a functional generative field extracted from a pretrained discriminative model. DSF converts the model's loss-gradient geometry into an image-space update direction. Unlike prior work, DSF evaluates whether a candidate image lies on the model-implied manifold by measuring its alignment in functional space. This alignment moves the image toward visual structure that is consistent with the pretrained model itself. In this operational sense, DSF turns a discriminative model into a generative model and provides a solution for discriminative models operating in continuous, high-dimensional image space.

The key challenge is that the manifold structure captured by a discriminative model is implicit. We therefore reinterpret score-based generation as motion toward a data manifold, and replace the inaccessible image-space manifold distance with a functional distance induced by the Loss Tangent Kernel (LTK) [5], a loss-aware variant of NTK [23]. This yields a computable field in which candidate images are updated according to their alignment with the pretrained model's loss-gradient geometry. As a result, DSF provides a functional score induced by a discriminative model. With a label-free augmentation-invariance surrogate in LTK functional space, the same formulation applies to classifiers, detectors, and self-supervised models under one rule across architectures.

We show that DSF is a general solution for reusing pretrained discriminative models as generative models, and that this solution is practically useful. DSF generalizes across both architecture and objective, spanning convolutional classifiers such as ResNet [19], detection transformers such as DETR [3], self-supervised ViT encoders such as DINOv2 [11, 40], and contrastive vision-language models such as CLIP. In all cases, DSF generates model-consistent images and support conditional generation, editing, inpainting, and explanation through a unified interface. Finally, we show that DSF is not only a generative interface but also a practical tool for model compression. In zero-shot quantization, DSF-generated images provide effective calibration data without access to real data or labels, outperforming existing data-free alternatives and demonstrating the broader utility of the model-induced generative geometry.

2 Related works

Score-based generative models generate samples by approximating the probability distribution of the target data [22, 45]. Since direct approximation is infeasible, a diffusion process is used: noise transforms the target distribution into a known distribution such as Gaussian [21, 44], and the reverse process recovers the target. In essence, these models iteratively transition from a Gaussian distribution to the data manifold. Training such models requires delicate methodologies, including learning a score function instead of modeling the full distribution [20], and handling data differently at each time step. Noisy classification also enables class-conditional generation along the trajectory [10].

Discriminative models as generative models are uncommon outside cases where discrimination and generation already overlap, such as autoregressive next-token prediction. In standard vision, discriminative models do not define an image-generation process, while generative models are often reused for discriminative tasks [27, 33]. DSF studies the inverse direction: extracting a generative field from a fixed pretrained discriminative vision model. Prior attempts to generate images from classification models mainly follow two directions: gradient-based visualization and maximal-margin reconstruction. Gradient-based methods, including DeepDream [37, 39], feature visualization, and related energy-based approaches, optimize images to activate a specified class, neuron, or energy. Meaningful synthesis typically requires additional handles such as architecture-specific priors [50], frequency regularization [13], total variation [36], auxiliary generative models [16], classifier fine-tuning [49], or joint generative-discriminative training [41]. Maximal-margin methods instead use stationarity conditions and links between support vector machines and neural networks to reconstruct or generate training data [2, 17, 35]. These methods often rely on overfitted classifiers, simple architectures, or small datasets [15]; even higher-resolution extensions [30] remain confined to classification. DSF targets a broader setting: extracting a label-free functional field from arbitrary pretrained discriminative vision models for generation in continuous image space.

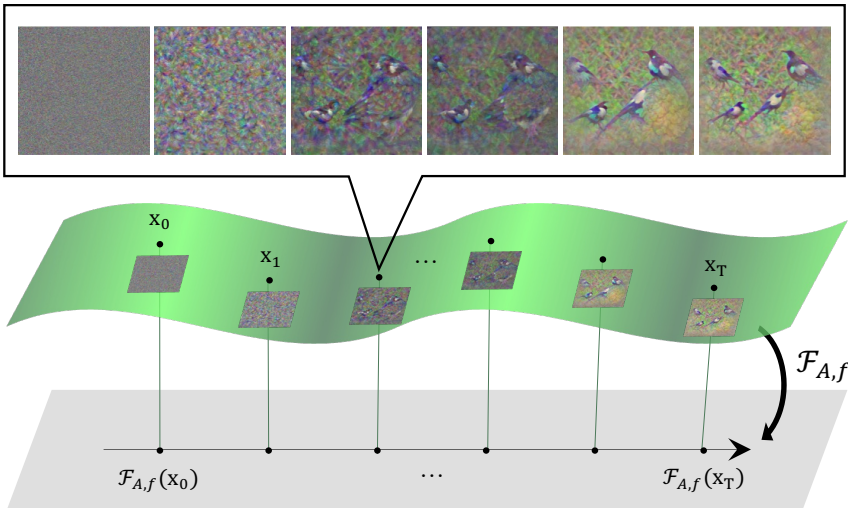


Figure 1: DSF as a functional generative prior. A candidate image is mapped through a fixed discriminative model into a loss-gradient functional space; DSF then provides an update direction that supports unconditional generation, conditional guidance, editing, inpainting, and explanation through the same interface.

Position of our paper. **(1) Motivation**: We expose and reuse the data-distribution-like visual structure hidden inside task-trained discriminative models. **(2) Framework**: DSF extracts a label-free functional score from loss-gradient geometry, giving fixed classifiers, detectors, and self-supervised encoders a unified generative interface. **(3) Evidence**: Generation results show that the interface works across architectures and objectives, while zero-shot quantization shows that DSF provides a practical calibration solution without real data, labels, or auxiliary generators in this setting.

3 Preliminaries

Kernel Methods are conventional machine learning techniques where a bivariate kernel $k(\cdot,\cdot)$ represents the inner product in a high-dimensional feature space, i.e., $k(x,y) = \langle \phi(x), \phi(y) \rangle$. Reproducing Kernel Hilbert Space (RKHS) guarantees the existence of the space ϕ where the closeness of $\phi(x)$ and $\phi(y)$ implies the closeness of x and y. Fixing one input gives the functional object $k(\cdot, x)$.

Neural Tangent Kernel tries to explain the generalization ability of deep learning models using conventional kernel methods. It suggests that for every deep learning network being trained, a corresponding kernel exists. NTK $\Theta(x,x')$ is defined by the gradients of the given deep learning model, $\Theta(x,x') = \nabla_\theta f(x)^\top \nabla_\theta f(x')$, reflecting how the model trains in the input space. Because NTK ignores loss dynamics, Loss Tangent Kernel (LTK) [5] adds loss derivatives:

$$K(x,x') = \nabla_f \ell(f(x;\theta))^\top \Theta(x,x') \nabla_f \ell(f(x';\theta)).$$

4 Discriminative Score Function

In this section, we derive the Discriminative Score Function (DSF) as a functional generative field induced by a pretrained discriminative model. First, Sec. 4.1 reviews the geometric view of score-based generation as motion toward a data manifold. Sec. 4.2 then introduces a training map that sends datasets to trained models, allowing us to discuss the manifold through functional objects. Finally, Sec. 4.3 replaces this ideal map with an LTK-based surrogate and defines the DSF update that moves candidate images toward model-consistent visual structure through the same field.

4.1 Connection between score-based model and Geometry

The goal of generative model is estimating data distribution $p(x)$ using a model f. There are two important points to consider. First, $p(x)$ is unknown and what we do know is the sampled dataset $\mathcal{D}_s \sim p(x)$. Second, image space $\mathbb{R}^{hw}$ is high-dimensional, while the data manifold $\mathcal{M} \subseteq \mathbb{R}^{hw}$ occupies a small, hard-to-approximate subset for sampling.

The direct way to solve this issue is to first sample x_0 from a known distribution and iteratively update it for T steps to generate x_T toward the manifold $\mathcal{M}$. Since we have no access to $\mathcal{M}$, the above process should be approximated using $\mathcal{D}_s \subseteq \mathcal{M}$ in practice for sampling:

$$\begin{aligned} x_{t+1} &= x_t - \eta_t \nabla_x d(x_t, \mathcal{M}) \coloneqq x_t - \eta_t \nabla_x \int_{\mathcal{M}} d(x_t, x) dx \\ &\simeq x_t - \eta_t \nabla_x \sum_i d(x_t, \zeta_i), \end{aligned} \tag{1}$$

where, ζ_i is the ith sample of $\mathcal{D}_s$ and η_t is the step size.

Direct implementation of Eq. (1) is almost impossible for two reasons: 1. It is difficult to find a metric $d(\cdot, \cdot)$ that represents $\mathcal{M}$ well, since the manifold occupies only a small, uneven portion of the high-dimensional space. 2. The amount of calculation is proportional to the number of samples in our data $\mathcal{D}_s$, *i.e.*, proportional to $|\mathcal{D}_s|$ for each update.

Score-based models resolves the implementation problem through a probability-space translation:

$$x_{t+1} = x_t + \eta_t \nabla_x \log p(x_t). \tag{2}$$

Here, the term score function refers to the gradient of the log probability density function, $\nabla_x \log p(x_t)$. Replacing the score function in Eq. (2) with $-\nabla_x d(x_t, \mathcal{M})$ brings us back to Eq. (1). Appendix. D details the connection between the score function and $d(x_t, \mathcal{M})$ used above.

As we only have a finite dataset, the distance d in Eq. 1 should satisfy the following condition:

$$d(\cdot, \mathcal{D}_s) \simeq d(\cdot, \mathcal{D}_t), \quad \forall \mathcal{D}_t \subseteq \mathcal{M}, \tag{3}$$

where $\mathcal{D}_s$ and $\mathcal{D}_t$ represent the training dataset and any test dataset drawn from the manifold $\mathcal{M}$, respectively. Intuitively, this means the metric should generalize the manifold effectively even when samples are sparse. Thus any subset from $\mathcal{M}$ can reliably estimate $d(\cdot, \mathcal{M})$ for generation.

A discriminative model does not explicitly provide this distance, but its ability to generalize from $\mathcal{D}_s$ to unseen samples from the same manifold indicates stable loss and representation geometry across draws from $\mathcal{M}$. We use this geometry to construct a functional surrogate for $d(\cdot, \mathcal{M})$.

4.2 Map the manifold to function space

To directly utilize the discriminative model, we need to map the data manifold to the function space. Training provides this map: same-manifold datasets should produce similar models.

We formalize this with a Functor $\mathcal{F}$ that maps a training setup to a trained model $f^\star$:

$$\mathcal{F}: \quad (\mathcal{D} \times \mathcal{A} \times f) \to f^\star, \tag{4}$$

where $\mathcal{D}$ is the dataset, $\mathcal{A}$ is the training algorithm (*e.g.*, SGD), and f is the network architecture.

Fixing $\mathcal{A}$ and f (*e.g.*, ResNet-50 and SGD), we obtain $\mathcal{F}_{\mathcal{A},f}: \mathcal{D} \to f^\star$. This Functor maps data, including generated candidates, to function space for direct use of the discriminative model (Fig. 2).

Since $\mathcal{F}_{\mathcal{A},f}(\mathcal{D}_s)$ is the trained discriminative model itself and generalizes to any test set from $\mathcal{M}$, we can replace the manifold distance in Eq. (1) and obtain the iterative process in Fig. 1:

$$\begin{aligned} x_{t+1} &= x_t - \eta_t \nabla_x d(\mathcal{F}_{\mathcal{A},f}(x), \mathcal{F}_{\mathcal{A},f}(\mathcal{M})) \\ &\simeq x_t - \eta_t \nabla_x d(\mathcal{F}_{\mathcal{A},f}(x), \mathcal{F}_{\mathcal{A},f}(\mathcal{D}_s)). \end{aligned} \tag{5}$$

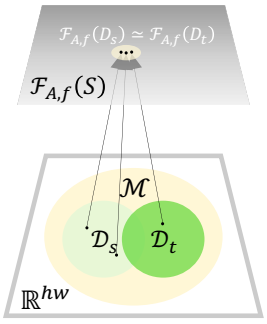


Figure 2: Training maps nearby datasets on the same manifold to nearby trained functions.

This substitution is valid because $\mathcal{F}_{\mathcal{A},f}$ is locally continuous–small changes in the dataset produce small changes in the trained model [1, 12, 18, 48]. This framework directly leverages the discriminative model since $\mathcal{F}_{\mathcal{A},f}(\mathcal{D}_s)$ itself is the trained discriminative model. What remains is to derive a good functor $\mathcal{F}$. This uses the trained model as a local functional representative of the data manifold: if nearby samples induce similar trained functions, then moving a candidate image toward the model-consistent functional region provides an image-space generative direction.

4.3 Direct Use of Discriminative Model through LTK

Loss Tangent Kernel as a Functor To directly compute the Functor $\mathcal{F}$ without explicit training, we employ the Loss Tangent Kernel (LTK) [5]. LTK integrates loss derivatives with the Neural Tangent Kernel (NTK), giving the training-free functional proxy for the map we need.

The LTK K measures the similarity between data points through their loss gradients:

$$\begin{aligned} K(x,x') \triangleq \langle k(\cdot,x), k(\cdot,x')\rangle = \nabla_f \ell(f(x))^\top \Theta \nabla_f \ell(f(x')) = \nabla_\theta \ell(f(x))^\top \nabla_\theta \ell(f(x')) \\ \nabla_f \ell(f(x)) \in \mathbb{R}^{C\times 1},\ \nabla_\theta \ell(f(x)) \in \mathbb{R}^{P\times 1},\ \Theta \in \mathbb{R}^{C\times C}, \end{aligned} \tag{6}$$

where C is the number of classes, P is the number of parameters and Θ is the NTK. The final equality is the parameter-gradient form of the same LTK inner product: with $J_\theta f(x)$ denoting the model Jacobian, $\Theta(x,x') = J_\theta f(x) J_\theta f(x')^\top$ and $\nabla_\theta \ell(f(x)) = J_\theta f(x)^\top \nabla_f \ell(f(x))$. It therefore evaluates the kernel through the corresponding loss-gradient features used below.

Instantiating the functional distance with kernel alignment after replacing $\mathcal{F}_{\mathcal{A},f}(x)$ with $k(\cdot,x)$ in Eq. (5), and writing the dataset-level loss-gradient aggregate as $\bar{g}_{\mathcal{D}}(\theta) = |\mathcal{D}_s|^{-1} \sum_i \nabla_\theta \ell(f(\zeta_i;\theta))$, gives

$$x_{t+1} = x_t - \eta_t \nabla_{x_t} d(k(\cdot,x_t), k(\cdot,\mathcal{D}_s)) \tag{7}$$

$$= x_t + \eta_t \nabla_{x_t} K(x_t, \mathcal{D}_s) \tag{8}$$

$$= x_t + \eta_t \nabla_{x_t} \{\nabla_\theta \ell(f(x_t;\theta))^T \bar{g}_{\mathcal{D}}(\theta)\} \tag{9}$$

$$\simeq x_t - \eta_t \nabla_{x_t} d(\nabla_\theta \ell(f(x_t;\theta)), -\theta) \tag{10}$$

Eq. (10) is an aggregate LTK alignment. In finite-width networks, loss-gradient features are sample-dependent, so the dataset-side representer of the alignment is the mean loss-gradient direction $\bar{g}_{\mathcal{D}}$. With access to training data, this anchor can be instantiated directly by averaging empirical loss gradients. DSF instantiates the same aggregate-gradient representer in the data-free regime using the pretrained parameter direction. For an ℓ_2-regularized stationary solution, $\nabla_\theta L_{\mathcal{D}}(\theta) + \rho\theta \simeq 0$, so $-\theta$ recovers this aggregate direction up to scale; more generally, it acts as a trajectory-level trace of the accumulated descent direction during training. Appendix C relates this replacement to kernel mean embeddings, trajectory matching, and regularized stationarity.

Label-Free Loss via Training Dynamics Matching Direct use of label-dependent losses prevents unconditional generation and may be unavailable in practice. We address this by introducing a label-free surrogate loss that exhibits similar training dynamics for DSF across architectures.

We select augmentation invariance loss because it naturally decreases during training, mirroring label-dependent loss behavior as models improve generalization in trained networks [28, 43, 46].

More specifically, we instantiate the candidate feature in the template using augmentation invariance:

$$\ell_{aug}(x_t;\theta) = 1 - \cos(h(A_1(x_t);\theta), h(A_2(x_t);\theta)), \tag{11}$$

where $A_1, A_2 \sim \mathcal{A}$ are two sampled augmentations and $h(\cdot;\theta)$ is the architecture-specific DSF representation. This operational loss, also used in Algorithm 1, supplies the gradient feature

$$g_{aug}(x_t;\theta) = \nabla_\theta \ell_{aug}(x_t;\theta).$$

Substituting this feature into the alignment template gives

$$x_{t+1} = x_t - \eta_t \nabla_x d(g_{aug}(x_t;\theta), -\theta), \tag{12}$$

In eq. (12), the vector field $-\nabla_x d(g_{aug}(x_t;\theta), -\theta)$ is our Discriminative Score Function (DSF). It uses loss-gradient geometry to move candidates toward the pretrained model's functional manifold.

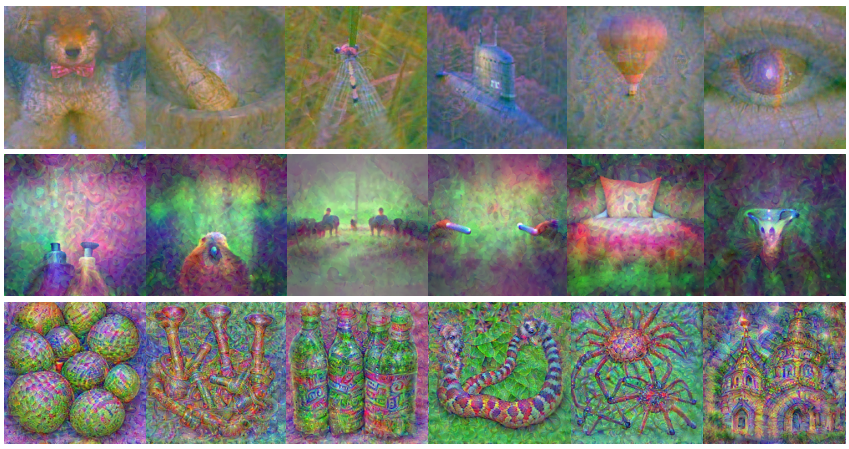


Figure 3: Examples of **unconditionally generated images** using various discriminative models. Top: DINOv2/LVD-124M (self-supervised learning). Middle: DeTR/COCO (object detection). Bottom: ResNet/Imagenet (image classification).

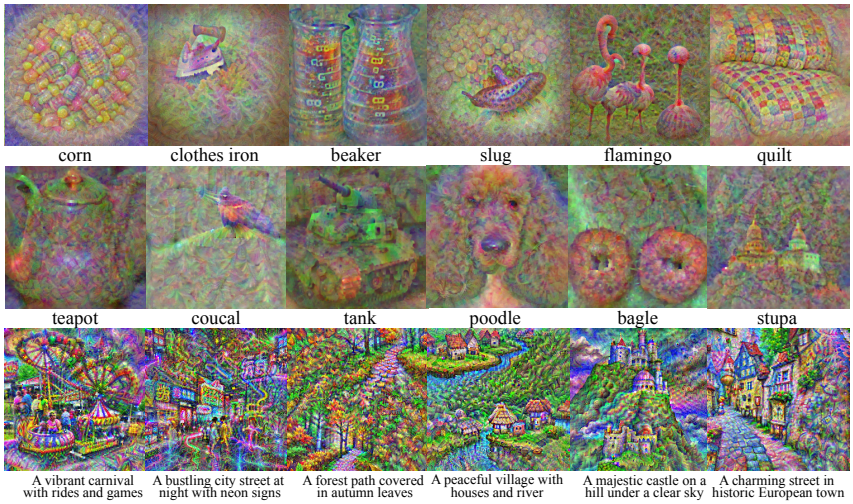


Figure 4: Examples of **conditionally generated images** using DSF. Top: Both the score model and the conditional model used ResNet/Imagenet. Middle: The score model used DINOv2/LVD-124M, and the conditional model used ResNet/Imagenet. Bottom: The score model used ResNet/Imagenet, and the conditional model used CLIP.

In implementation, d is a layerwise normalized ℓ_1 distance over the selected layers $\mathcal{S}$ to reduce the effect of outlier parameters, aligning the induced update direction $-g_\ell$ with the model-side direction θ_ℓ, where $g_\ell = \nabla_{\theta_\ell} \ell_{aug}(x;\theta)$. We add a small ϵ in the normalization for stability. The image update differentiates this scalar alignment objective with respect to x. Appendix C gives details.

5 Experiments

Experimental setup. We use public pretrained ResNet [19], DINOv2 [40], and DETR [3] models without post-training. Images are optimized directly from random noise using the DSF update. For generation experiments, we report unconditional and conditional samples across classification, self-supervised representation learning, and object detection models. For zero-shot quantization, we use DSF-generated samples as calibration data for a fixed DINOv2 feature extractor and compare them with real-image and data-free calibration sources. Appendix C provides optimization details, architecture-specific layer choices, and quantization protocols used in all reported experiments.

5.1 Unconditional Generation

Unconditional generation provides the most direct test of whether a fixed discriminative model induces a usable generative field. In this setting, no class label, text prompt, target activation, real image, or external generator is provided. We initialize images from random noise and optimize them only with the DSF update in Eq. (12). If the optimized samples exhibit coherent visual structure, the signal must come from the pretrained discriminative model's functional geometry.

We evaluate this setting across three representative discriminative objectives: classification with ResNet, object detection with DETR, and self-supervised representation learning with DINO. All models are public pretrained checkpoints and are used without modification. Fig. 3 shows the resulting unconditional samples. DSF produces model-dependent visual structure in all cases: the classifier tends to generate object-centric patterns, DINO produces broad natural-image structure from its self-supervised representation, and DETR yields scene-like samples with multiple object-like regions. These results demonstrate that DSF exposes a reusable functional prior across pretrained discriminative models with different architectures and objectives, indicating that discriminative training can leave a functional-space direction without generator training from fixed models.

5.2 Conditional Generation

The same DSF field can also be combined with external guidance fields. To test this, we implement a simple classifier-guided-diffusion variant [10] with a conditional loss term during sampling:

$$x_{t+1} = x_t \underbrace{-\nabla_x d(\nabla_\theta f(x_t;\theta)\lambda, -\theta)}_{\text{score}} + \underbrace{\nabla_x \log p_\phi(y|x_t)}_{\text{condition}}. \tag{13}$$

Here, p_ϕ denotes an external guidance model. The experiment tests whether DSF can serve as the prior term while a separate discriminative or vision-language model supplies the condition. Fig. 4 shows that DSF combines with pretrained classifiers or CLIP across architectures.

5.3 Zero-Shot Quantization

A practical advantage of DSF is deployment-time calibration without real data. Post-training quantization requires representative inputs to estimate activation ranges and feature statistics, but zero-shot calibration has no access to real calibration images or labels [25]. This is particularly challenging for self-supervised encoders such as DINOv2 [40], which lack classifier logits, label-conditioned objectives, and BatchNorm statistics that can guide conventional synthetic calibration.

DSF provides a model-matched calibration source because its samples are generated from the functional geometry of the same model being quantized. We use DSF-generated images to calibrate a fixed DINOv2 feature extractor and compare against real ImageNet images as an oracle reference, together with data-free calibration sources including RegCache [26], Gaussian noise, and MimiQ [7]. Since the target is a representation model rather than a classifier, we evaluate downstream usability and preservation of full-precision feature geometry after quantization in this setting.

Table 1 shows that DSF is comparable to the real-image oracle across 8-, 6-, and 4-bit quantization, while consistently outperforming data-free calibration sources. The gap is most visible at lower bit-widths, where generic data-free sources more severely distort the representation geometry. These results suggest that DSF samples cover relevant manifold regions and preserve feature structure.

6 Generative Properties

Beyond synthesis and calibration, DSF acts as a reusable image-space field induced by a fixed discriminative model. We study qualitative uses of this field: since the update acts directly on images without labels, masks, or a trained generator, it supports editing, restoration, explanation, and sampling heuristics with minimal task-specific changes.

Table 1: **Zero-shot quantization with DSF calibration.** DSF matches or exceeds the real-image oracle in nearly all settings and consistently outperforms data-free calibration sources for a quantized DINOv2 feature extractor. All sources use 100 calibration images and the same scale-selection protocol; higher is better.

Calibration	Downstream usability						Representation geometry								
	k-NN↑			Head↑			fpQ@5↑			CLS↑			Patch↑		
	8	6	4	8	6	4	8	6	4	8	6	4	8	6	4
Real images†	**.659**	**.605**	.580	.269	.195	.187	.882	**.873**	.823	.939	.798	.708	.968	.874	.753
DSF (ours)	**.659**	**.605**	**.589**	**.283**	**.215**	**.202**	**.883**	.872	**.825**	**.947**	**.814**	**.710**	**.973**	**.885**	**.760**
RegCache	.629	.564	.500	.196	.148	.142	.874	.859	.787	.902	.744	.608	.963	.849	.671
Gaussian	.592	.435	.077	.152	.051	.003	.874	.791	.058	.825	.515	.102	.913	.622	.153
MimiQ	.588	.357	.078	.152	.036	.001	.877	.691	.032	.833	.435	.095	.917	.553	.141

Note. Bit-width subcolumns denote 8-, 6-, and 4-bit quantization. † denotes ImageNet calibration images and is an oracle reference.

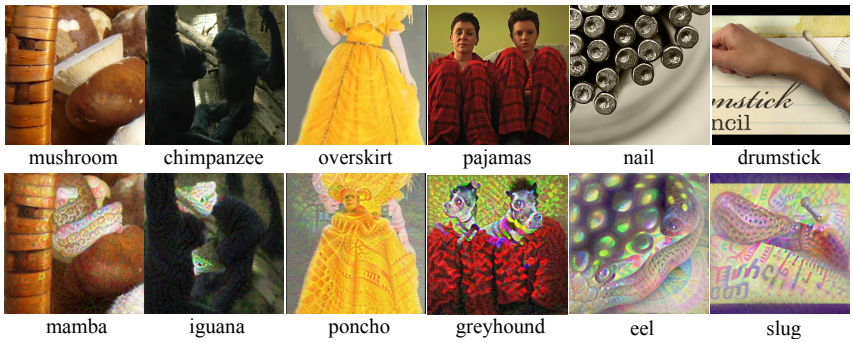


Figure 5: Examples of editing through DSF. Top: original images; bottom:he top row contains original images while the bottom row shows edited images.

6.1 Image Editing

What a diffusion model learns is the reverse process of converting an image to pure noise. Based on this property, editing tasks in diffusion include sending the object image to a noise-like image and retrieving it back. The condition is injected into this process in the form of masking [8]. For the case of DSF, what it conducts is not the denoising process but the direct estimation of manifold distribution $p(x)$. Editing is therefore possible at image level without adding noise to the target image. Masking is also unnecessary. The result of directly applying conditional generation to a real image without adding noise or masking is shown in Fig. 5. The outstanding point is that the semantic has changed without largely deforming the input images. For example, in the photo of a chimpanzee, a spot of light fallen at the background is naturally transformed to a head of iguana. The animal body is lightly deformed to form the iguana's body and shadow. This was possible because the score function consistently restrains the edit from straying away from the manifold.

6.2 Image Inpainting

Image inpainting visually restores damaged or omitted parts of a target image. To enable this, a model must generate new content according to the surrounding context; diffusion models are suitable for inpainting tasks in this sense. Similarly, DSF can serve as an effective inpainting model. To evaluate its performance, we masked random image regions with empty patches. Although DSF is calculated on the entire image, updates were restricted to masked areas. To guide restoration, we introduced a simple boundary-consistency regularizer. With this straightforward setup and direct DSF application, the missing regions were successfully restored, as shown in Fig. 6.

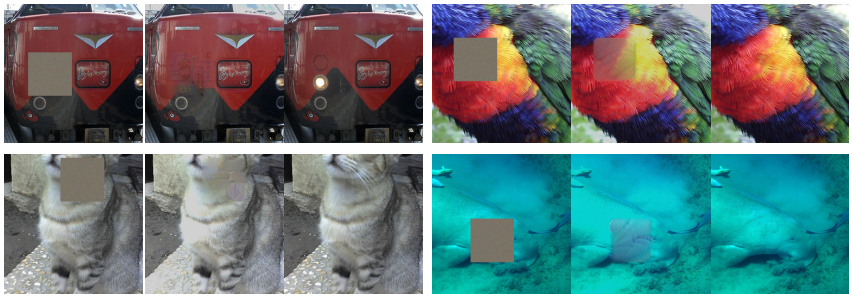


Figure 6: Examples of the inpainting task. For each set, the left is the inpainted image, the middle is the reconstructed one, and the right is the original image.



Figure 7: Comparison between DSF and an existing feature visualization method: (Left) DSF, (Middle) real ImageNet images of the targeted class, and (Right) visualizations produced by [13].



Figure 8: Unconditional images generated with DINOv2 [40]; sampling uses linear step scheduling and fixed noise injection.

6.3 Global Explanation

DSF can also be used as a global explanation tool. Unlike naive feature visualization, which often follows adversarial activation gradients, DSF constrains optimization by a functional prior induced by the pretrained model. This distinction is important: if DSF produces more complete and contextually plausible images than activation maximization, then the generated samples can reveal dataset biases and feature entanglements in the model's learned representation. For example, when applied to a ResNet trained on ImageNet, DSF repeatedly generates "analog clock" images showing 10:10 and "plate" images with contextual elements such as wine glasses and dishes. These outputs suggest that DSF can expose common co-occurrences and biases encoded by the discriminative model, while comparison with [13] separates manifold-guided visualization from activation-driven artifacts.

6.4 Unlocking the potential of DSF through diffusion model

DSF is an image-space generative field, so standard sampling heuristics can be used to improve the trajectory. Our base experiments use a deliberately simple fixed-step update; diffusion models, however, typically rely on step-size schedules and intermediate noise injection to correct bias and improve sample quality [20, 44]. We therefore apply the simplest version of these ideas: linear step-size decay with proportional noise injection. As shown in Fig. 8, unconditional generation from DINOv2 [40] exhibits fewer artifacts and more coherent structures than the fixed-step baseline in Fig. 3. This improvement shows that DSF can benefit from better samplers and suggests a direct path for improving discriminative-model-induced generation with no change to the field.

7 Conclusion

In this paper, we introduced the Discriminative Score Function (DSF), a training-free mechanism for extracting a generative field from a pretrained discriminative model. DSF does not require labels, data, or architectural modification; instead, it uses the model's own loss-gradient geometry for synthesis. Our experiments show that this geometry can support generative ability to discriminative

ones. Practically, DSF-generated samples improve zero-shot quantization calibration, indicating that the induced field captures model-specific activation structure rather than merely producing visual artifacts. The broader implication is that discriminative models can serve as more than predictors: they can expose reusable generative structure tied to their learned representations.

Limitations. DSF is viewed as a training-free generative interface induced by a discriminative model, not a replacement for high-fidelity diffusion generators. Its gradient proxy can depend on architecture and training recipe, and parameter-gradient updates make sampling costlier than first-order image optimization. Broader domains, and stronger samplers remain future work.

Acknowledgments and Disclosure of Funding

This work was supported by the Korean Government through the grants from IITP (RS-2021-II211343 and RS-2025-25442338) and KOCCA (RS-2024-00398320).

References

[1] Olivier Bousquet and André Elisseeff. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526, 2002.

[2] Gon Buzaglo, Niv Haim, Gilad Yehudai, Gal Vardi, Yakir Oz, Yaniv Nikankin, and Michal Irani. Deconstructing data reconstruction: Multiclass, weight decay and general losses. *arXiv preprint arXiv:2307.01827*, 2023.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *CoRR*, abs/2005.12872, 2020. URL `https://arxiv.org/abs/2005.12872`.

[4] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 4750–4759, June 2022.

[5] Yilan Chen, Wei Huang, Hao Wang, Charlotte Loh, Akash Srivastava, Lam M. Nguyen, and Tsui-Wei Weng. Analyzing generalization of neural networks through loss path kernels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL `https://openreview.net/forum?id=8Ba7VJ7xiM`.

[6] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829, 2023.

[7] Kanghyun Choi, Hyeyoon Lee, Dain Kwon, SunJong Park, Kyuyeun Kim, Noseong Park, Jonghyun Choi, and Jinho Lee. Mimiq: Low-bit data-free quantization of vision transformers with encouraging inter-head attention similarity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 16037–16045, 2025.

[8] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance, 2022. URL `https://arxiv.org/abs/2210.11427`.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

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

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL `https://arxiv.org/abs/2010.11929`.

[12] Andre Elisseeff, Theodoros Evgeniou, Massimiliano Pontil, and Leslie Pack Kaelbing. Stability of randomized learning algorithms. *Journal of Machine Learning Research*, 6(1), 2005.

[13] Thomas Fel, Thibaut Boissin, Victor Boutin, Agustin Picard, Paul Novello, Julien Colin, Drew Linsley, Tom Rousseau, Rémi Cadène, Laurent Gardes, et al. Unlocking feature visualization for deeper networks with magnitude constrained optimization. *arXiv preprint arXiv:2306.06805*, 2023.

[14] Richard P. Feynman. What I cannot create I do not understand. Inscription on Feynman's blackboard at the time of his death, 1988. URL https://feynman.com/fun/quotes/. Transcribed on the Richard Feynman website; accessed October 14, 2026.

[15] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

[16] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

[17] Niv Haim, Gal Vardi, Gilad Yehudai, michal Irani, and Ohad Shamir. Reconstructing training data from trained neural networks. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Sxk8Bse3RKO.

[18] Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International conference on machine learning*, pages 1225–1234. PMLR, 2016.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

[20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. URL https://arxiv.org/abs/2006.11239.

[22] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[23] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *CoRR*, abs/1806.07572, 2018. URL http://arxiv.org/abs/1806.07572.

[24] Ziwei Ji and Matus Telgarsky. Directional convergence and alignment in deep learning. *CoRR*, abs/2006.06657, 2020. URL https://arxiv.org/abs/2006.06657.

[25] Minjun Kim, Jaehyeon Choi, Jongkeun Lee, Wonjin Cho, and U Kang. Zero-shot quantization: A comprehensive survey. *arXiv preprint arXiv:2505.09188*, 2025. URL https://arxiv.org/abs/2505.09188.

[26] Seunghyeon Kim, Taesun Yeom, Jinho Kim, Wonpyo Park, Kyuyeun Kim, and Jaeho Lee. Activation quantization of vision encoders needs prefixing registers. *arXiv preprint arXiv:2510.04547*, 2025.

[27] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. *Advances in neural information processing systems*, 27, 2014.

[28] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

[29] Dongkwan Lee, Junhoo Lee, and Nojun Kwak. Deep edge filter: Return of the human-crafted layer in deep learning. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2510.13865.

[30] Junhoo Lee, Hyunho Lee, Kyomin Hwang, and Nojun Kwak. Deep support vectors. In *Advances in Neural Information Processing Systems*, 2024.

[31] Junhoo Lee, Seungyeon Kim, and Nojun Kwak. Unlocking the potential of diffusion language models through template infilling. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6273–6287, 2026. doi: 10.18653/v1/2026.acl-long.284. URL https://aclanthology.org/2026.acl-long.284/.

[32] Junhoo Lee, Mijin Koo, and Nojun Kwak. CSF: Black-box fingerprinting via compositional semantics for text-to-image models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2604.16363.

[33] Alexander C Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2206–2217, 2023.

[34] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL http://arxiv.org/abs/1405.0312.

[35] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJeLIgBKPS.

[36] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196, 2015.

[37] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks. *Google research blog*, 17(6):1–10, 2015.

[38] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends in Machine Learning*, 10(1–2):1–141, 2017.

[39] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11): e7, 2017.

[40] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

[41] Shibani Santurkar, Andrew Ilyas, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Image synthesis with a single (robust) classifier. *Advances in Neural Information Processing Systems*, 32, 2019.

[42] Alexander J. Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pages 13–31. Springer, 2007.

[43] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020.

[44] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *CoRR*, abs/2010.02502, 2020. URL https://arxiv.org/abs/2010.02502.

[45] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *CoRR*, abs/2011.13456, 2020. URL https://arxiv.org/abs/2011.13456.

[46] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

[47] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models, 2024. URL https://arxiv.org/abs/2211.10438.

[48] Huan Xu and Shie Mannor. Robustness and generalization. *Machine learning*, 86:391–423, 2012.

[49] Xiulong Yang, Hui Ye, Yang Ye, Xiang Li, and Shihao Ji. Generative max-mahalanobis classifiers for image classification, generation and more. In *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part II 21*, pages 67–83. Springer, 2021.

[50] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8715–8724, 2020.

[51] Chulhee Yun, Shankar Krishnan, and Hossein Mobahi. A unifying view on implicit bias in training linear neural networks. In *International Conference on Learning Representations (ICLR)*, 2021. URL `https://arxiv.org/abs/2010.02501`.

A Impact Statement

DSF is a methodological contribution for reusing fixed pretrained discriminative vision models as training-free generative priors. A positive impact is that it can reduce the need to train separate generative models for some inspection, calibration, and diagnostic workflows, which may lower computation and data-access requirements. Its zero-shot calibration use case is especially relevant when real calibration data cannot be shared because of privacy, licensing, or deployment constraints.

The same capability also has risks. Generated images should not be interpreted as faithful reconstructions of the original training set or as complete descriptions of the data manifold; they are model-induced visualizations and may amplify biases, spurious correlations, or artifacts encoded in the pretrained model. If used to synthesize calibration or debugging data, DSF samples may inherit undesirable correlations from the source model. We therefore frame DSF primarily as an analysis and deployment-support tool, report its limitations, and do not release a new high-risk pretrained generator or dataset.

B Existing Assets and Licenses

We use existing public models, datasets, and baseline methods only for research evaluation, cite their original sources, and do not redistribute third-party datasets or pretrained checkpoints as new assets. ImageNet calibration images are used under the ImageNet access terms for non-commercial research and educational use. COCO is referenced through public DETR/COCO models and, where data are used, follows the COCO terms and associated image licenses. Public code dependencies and baselines are used according to their repository licenses.

Table 2: Existing assets used or referenced in the experiments.

Asset	Use in this paper	License or terms followed
ResNet / torchvision [19]	Public pretrained classifier used as a DSF score model	PyTorch/torchvision code and model access are used under the PyTorch/torchvision BSD-style license terms; ImageNet-trained weights are used only for research evaluation.
DINOv2 [40]	Public pretrained self-supervised ViT used for generation and quantization diagnostics	DINOv2 code and model weights are released under Apache-2.0; pretrained checkpoints are used as fixed evaluation models and are not redistributed.
DETR / COCO [3, 34]	Public pretrained detector used as a DSF score model	DETR is released under Apache-2.0; COCO data are used or referenced under the official COCO terms and associated image licenses.
ImageNet [9]	Oracle calibration images and downstream evaluation reference	Accessed under ImageNet terms for non-commercial research and educational use; images are not redistributed.
CLIP / OpenCLIP [6]	External guidance model for conditional generation	Public CLIP/OpenCLIP implementations and checkpoints are used under their respective repository/model licenses, including MIT-licensed OpenAI CLIP code, and are not redistributed.
SmoothQuant [47]	Post-training quantization transform used in zero-shot calibration diagnostics	SmoothQuant code is released under the MIT license.
RegCache and MimiQ [7, 26]	Data-free calibration baselines	Cited baseline methods are used for comparison under their authors' public paper/code terms where applicable.

C Implementation Details

Operational DSF update Algorithm 1 summarizes the implementation used in our experiments and uses the same two-view cosine augmentation-invariance loss as Eq. (12). The scalar objective

optimized over images is not a class logit or a target-neuron activation. Instead, DSF first forms a label-free augmentation-invariance loss, differentiates this loss with respect to a selected subset of model parameters, and aligns the resulting parameter-gradient direction with $-\theta$ using the layerwise normalized ℓ_1 distance d. The image-space field is the input gradient of this alignment objective.

Algorithm 1 Operational DSF update used for image optimization.

Require: fixed model f_θ, selected score layers $\mathcal{S}$, trainable image x, augmentation distribution $\mathcal{A}$, step size η_t.

1: Sample two augmented views $A_1(x)$ and $A_2(x)$, and compute latent features $h(A_1(x))$, $h(A_2(x))$.

2: Compute the label-free loss

$$\ell_{aug}(x;\theta) = 1 - \cos(h(A_1(x)), h(A_2(x))).$$

3: For each selected layer $\ell \in \mathcal{S}$, compute $g_\ell = \nabla_{\theta_\ell} \ell_{aug}(x;\theta)$, while preserving the graph for differentiation through g_ℓ.

4: Form the normalized alignment objective

$$L_{\mathrm{DSF}}(x) = \sum_{\ell \in \mathcal{S}} \left\| \frac{g_\ell}{\|g_\ell\|_1 + \epsilon} + \frac{\theta_\ell}{\|\theta_\ell\|_1 + \epsilon} \right\|_1 .$$

5: Skip bias vectors and layers with unused gradients.

6: Update the image:

$$x \leftarrow \Pi_{[0,1]}(x - \eta_t \nabla_x L_{\mathrm{DSF}}(x)),$$

optionally adding TV/norm regularization, conditional guidance, and Langevin noise depending on the experiment.

Parameter direction as a data-free aggregate-gradient proxy Here we spell out why the normalized alignment objective contains a plus between g and θ. The LTK alignment in Eq. (10) is naturally written using a dataset-level aggregate gradient. This matters in finite-width networks: individual parameter-gradient features are sample-dependent and anisotropic, so practical kernel-gradient constructions use an averaged or aggregate direction as the representative dataset-side object. Let $L_{\mathcal{D}}(\theta) = \sum_i \ell(f(\zeta_i;\theta))$ be the empirical training objective and let $g_{\mathcal{D}}(\theta) = \nabla_\theta L_{\mathcal{D}}(\theta)$. If the training set were available, $g_{\mathcal{D}}$ could be estimated directly. DSF instead uses the pretrained parameter direction as a data-free proxy for the aggregate descent direction used by DSF.

Gradient descent updates parameters by

$$\theta_{t+1} = \theta_t - \eta_t g_{\mathcal{D}}(\theta_t), \tag{14}$$

so the learned parameter displacement records accumulated *negative* empirical loss-gradient directions. Thus, a candidate image whose loss-gradient is consistent with the training-induced aggregate should produce an update direction $-g(x)$ that points along θ, up to normalization and layerwise scaling. This is why we minimize $\|g_\ell/\|g_\ell\| + \theta_\ell/\|\theta_\ell\|\|$: making g_ℓ and θ_ℓ opposite is exactly making the update direction $-g_\ell$ align with the model-side descent trace θ_ℓ. The sign is fixed by the gradient-descent update convention rather than treated as a tunable hyperparameter.

This anchor also agrees with two standard views of trained discriminative models. In a regularized stationary point, the first-order condition has the following form at convergence in practice

$$\nabla_\theta L_{\mathcal{D}}(\theta) + \rho\theta \simeq 0, \tag{15}$$

which directly identifies the empirical loss-gradient aggregate with $-\theta$ up to the regularization scale, or equivalently identifies the empirical update direction with θ. In separable or homogeneous settings, implicit-bias and directional-convergence analyses show that gradient-based training drives the weight direction toward margin- or training-gradient-induced directions even when the norm continues to grow [24, 35, 51]. DSF does not claim that $-\theta$ equals every sample gradient, nor that the equality is exact for every modern pretrained architecture. Rather, $-\theta$ is a model-side proxy for the unavailable dataset-average gradient representer. The practical assumption is that the learned parameter direction summarizes a useful trace of training-induced aggregate descent.

About θ proxy The construction above is analogous to several standard ways of replacing many sample-dependent objects by one aggregate representative. In kernel mean embeddings, a distribution is represented in an RKHS by the mean feature vector $\mu_{\mathcal{D}} = |\mathcal{D}|^{-1} \sum_i \phi(\zeta_i)$, and evaluating a new point uses $\langle \phi(x), \mu_{\mathcal{D}} \rangle$ rather than a single sample feature [38, 42]. The LTK view makes the same

move in a loss-gradient feature space: the feature is $\phi_\theta(z) = \nabla_\theta \ell(f(z;\theta))$, and the dataset-side object is the aggregate $\bar{g}_\mathcal{D} = |\mathcal{D}|^{-1} \sum_i \phi_\theta(\zeta_i)$ [5, 23]. This is natural in finite-width networks, where individual gradient features vary across samples, while the aggregate serves as the representative direction used by kernel alignment.

DSF differs only in that the training set is unavailable. We therefore need a model-side replacement for $\bar{g}_\mathcal{D}$. The final parameters provide such a replacement because training trajectories are built by accumulating dataset-gradient updates. This is the same general source of information exploited by trajectory-matching dataset distillation, where synthetic data are optimized to reproduce parameter trajectories [4], and by training-data reconstruction analyses, where stationarity and weight decay reveal data-dependent gradient information in trained weights [2, 17]. Under the ideal ℓ_2-regularized stationary condition, the replacement is exact up to scale; outside this ideal case, the parameter direction is an operational aggregate-gradient proxy, not a pointwise equality. Layerwise normalization and late-layer selection preserve this directional interpretation while reducing sensitivity to architecture-specific scale, outliers, and parameterization across architectures in practice.

Layers and normalization The subset $\mathcal{S}$ is chosen for computational tractability, and late layers provide the high-level representation geometry used by DSF. We use layerwise normalized ℓ_1 alignment to compare directions robustly under layer-scale variation and outlier coordinates.

Architecture instantiation Table 3 lists the representation $h(x)$ and parameter subset $\mathcal{S}$ used to instantiate the same DSF update across architectures. For DETR, DSF is applied to the encoder representation and does not use Hungarian matching, box regression loss, or detector-head targets.

Table 3: Architecture and objective coverage of DSF. The same DSF update is used across target discriminative models, while CLIP is used as an external guidance model.

Model	Model type	Objective	Representation and score layers
Target models			
ResNet	ConvNet classifier	classification	AvgPool/classifier features; conv. and residual-block layers
DETR	detection transformer	detection	encoder features; encoder projection layers
DINOv2 ViT	ViT encoder	self-supervised learning	final normalized features; late attention and MLP layers
External guidance model			
CLIP	vision–language encoder	contrastive image–text learning	visual embedding; visual encoder layers

Optimization and compute Images are optimized directly as trainable tensors and clamped to $[0, 1]$ after updates. The augmentation policy used in the main sweeps includes crop, additive noise, color jitter/drop, and cutout. The crop operation preserves the original tensor size by padding the cropped view back to the input resolution. For scheduled runs, learning-rate and TV multipliers use the same piecewise schedule: 1.0, 0.8, a linear decay from 0.8 to 0.25, a linear decay from 0.25 to 0.05, and then 0.05, with thresholds scaled by the run-specific step-rate. The Langevin-noise schedule uses $5 \cdot 10^{-4}$, $2 \cdot 10^{-4}$, a linear decay to 10^{-4}, and finally 10^{-5}. Because Step 3 in Algorithm 1 preserves the parameter-gradient graph, DSF requires mixed second-order differentiation: the first derivative forms $g_\ell = \nabla_{\theta_\ell} \ell_{aug}$, and the image update differentiates the alignment objective through g_ℓ. In A6000 sweeps with batch size 50, this costs about 1.4–1.7 seconds per image.

Zero-shot quantization protocol For Table 1, the target model is a fixed DINOv2 feature extractor. We apply post-training SmoothQuant with $\alpha = 0.75$ and evaluate 8-, 6-, and 4-bit quantization. Each calibration source contributes 100 images. The real-image source uses ImageNet calibration images and therefore serves as an oracle reference for the ImageNet downstream evaluation rather than a zero-shot method. DSF uses images generated from the target model itself, without real calibration images, class labels, or an external generator. We compare against data-free calibration sources including RegCache, Gaussian noise, and MimiQ under the same calibration budget. Gaussian noise is the data-free lower baseline, while RegCache and MimiQ do not use real ImageNet calibration images. Calibration-source construction and scale selection do not use evaluation labels.

The activation-scale multiplier is selected separately for each source and bit-width by a small sweep. Table 4 reports the selected multipliers. In the main table, we omit this column because it is an implementation choice rather than a result metric.

Table 4: Activation-scale multipliers used for zero-shot quantization diagnostics. Each value is selected from the calibration-source sweep used in Table 1.

Calibration source	8-bit scale	6-bit scale	4-bit scale
Real images	2.0	0.75	0.75
DSF	2.0	0.75	1.0
RegCache	1.5	1.0	1.25
Gaussian noise	2.0	1.25	1.5
MimiQ	2.0	1.25	0.75

The reported diagnostics measure different views of feature preservation. k-NN top-1 classifies each validation image by nearest neighbors in the frozen feature space. fpQ@5 measures whether nearest neighbors obtained with quantized features agree with those from the full-precision feature space. CLS and Patch report cosine similarity between full-precision and quantized CLS-token and patch-token representations. Head reports ImageNet top-1 accuracy using a frozen full-precision linear head on top of the quantized feature extractor; new integer heads are unstable here.

D Generalization of Score Model

This section explains how the score function can be understood geometrically. Consider the following generation process through a score-based model on the manifold of interest for sampling:

$$x_{t+1} = x_t + \lambda_t \nabla \log p(x_t), \tag{16}$$

where $p(x)$ is the probability distribution corresponding to the desired manifold of data. Our goal is to approximate $p(x)$ in a way that captures the underlying structure of this manifold.

A natural way to view $p(x)$ is as a continuous generalization of **Kernel Density Estimation (KDE)**. In KDE, an empirical density estimate $\hat{p}(x)$ can be written over dataset samples as:

$$\hat{p}(x) = \frac{1}{n} \sum_{i=1}^{n} K(x, x_i). \tag{17}$$

Extending this notion to a continuous integral over the manifold M, we can write:

$$p(x) = \int_M K(x, u)\, du. \tag{18}$$

From this perspective, the corresponding **score function**, the gradient of the log-density, becomes:

$$\nabla_x \log p(x) = \nabla_x \log \int_M K(x, u)\, du. \tag{19}$$

Geometrically, this gradient term can be interpreted as a "pull" of x toward high-density regions on the manifold M, where the desired data manifold is most concentrated under this view.

Moreover, when viewed purely in terms of geometry, one may simplify the problem by omitting the log for computational convenience or stability. In that case, the **diffusion-like** or **flow-based** perspective leads to the more direct generalization we use below for DSF sampling:

$$\nabla_x \int_M d(x, u)\, du, \tag{20}$$

where $d(x, u)$ is a distance or kernel-based measure between x and points u in the manifold M. This views the iterative update as moving x along the manifold toward higher-density regions.



Figure 9: More images generated by ResNet50/ImageNet.

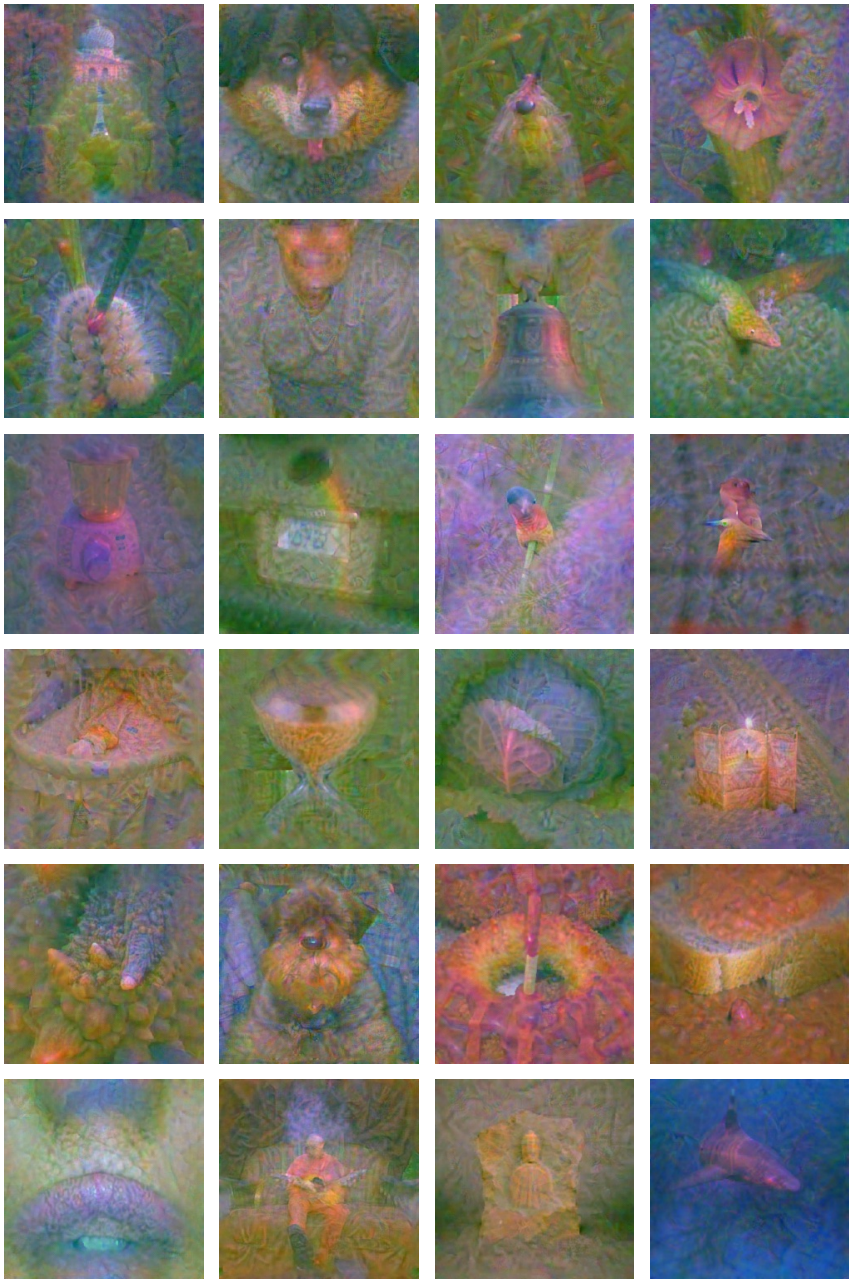


Figure 10: More images generated by DINOv2/LVD-124M.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the scope as extracting a training-free functional generative field from fixed pretrained discriminative vision models, with experiments across ResNet, DETR, DINOv2, and CLIP-guided settings. The claims are tied to the reported generation, editing, inpainting, explanation, and zero-shot quantization results rather than to unrestricted generative modeling.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See limitations Section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out strong assumptions and how robust the results are to violations of these assumptions.
   - The authors should reflect on the scope of the claims made, such as if the approach was only tested on a few datasets or with a few runs.
   - The authors should discuss computational efficiency and scaling where applicable.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete proof?

   Answer: [N/A]

   Justification: The paper provides a geometric and kernel-inspired derivation of DSF rather than formal theorems with proof obligations. The assumptions behind the operational proxy are explained in Sec. 4.3 and Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material.
   - Informal proof sketches in the main paper should be complemented by formal proofs in the appendix when applicable.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper?

   Answer: [Yes]

Justification: The experiments section describes the experimental settings and Appendix C gives the operational update, selected score layers, augmentation policy, optimization schedules, quantization protocol, and calibration-source details needed to reproduce the reported results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- Reproducibility can be provided through code, data, model access, or detailed instructions appropriate to the contribution.
- The paper should make clear how to reproduce the proposed algorithm and empirical evaluation.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The supplementary materials include the anonymized code and instructions for reproducing the DSF image optimization and zero-shot quantization experiments reported in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments requiring code.
   - Code and data instructions should include exact commands and environment details when provided.
   - NeurIPS encourages release, but a justified No is acceptable when release is not possible at submission time.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details, including data splits, hyperparameters, how they were chosen, and evaluation protocols?

   Answer: [Yes]

   Justification: The paper specifies the pretrained public models, DSF update, layer choices, optimization schedules, calibration budget, SmoothQuant settings, bit-widths, activation-scale sweep, and evaluation metrics in the experiments section and Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core paper to a level needed to understand the results.
   - Full details can be provided with code, in an appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper reports deterministic diagnostic tables and qualitative generations, but does not report error bars, confidence intervals, or significance tests. The main quantitative comparison is therefore presented as a controlled diagnostic comparison under a fixed protocol rather than as a statistical significance claim.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - Error bars, confidence intervals, or statistical significance tests should be reported where they support the main claims.
   - The variability source and calculation method should be stated when error bars are reported.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix C reports the mixed second-order nature of the DSF update and the measured A6000 batch cost for the main sweeps. The paper also identifies the public pretrained models and image resolution/calibration budgets used in the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers, memory where relevant, and time required to reproduce experiments.
   - The paper should disclose whether substantially more compute was used in preliminary or failed experiments when applicable.

9. **Code Of Ethics**

   Question: Have you read the NeurIPS Code of Ethics and ensured that your research conforms to it?

   Answer: [Yes]

   Justification: The work uses public pretrained vision models and standard image benchmarks for methodological research, and the paper discusses bias-revealing behavior and potential societal implications in the Impact Statement.

   Guidelines:

   - If there are special circumstances requiring deviation from the Code of Ethics, explain them while preserving anonymity.

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The Impact Statement discusses potential positive impacts through reuse of pretrained discriminative models and reduced need for separate generative training, as well as the ability to reveal model or dataset biases through generated visualizations.

    Guidelines:

    - If appropriate, discuss potential negative societal impacts such as misuse, fairness, privacy, or security considerations.
    - Foundational research not tied to a deployment may still mention direct paths to negative applications when they exist.
    - Mitigation strategies may be discussed where relevant.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse?

    Answer: [N/A]

    Justification: The paper does not release a new high-risk pretrained model, dataset, or scraped corpus. It studies a training-free method applied to existing public pretrained vision models.

    Guidelines:

    - The answer NA means that the paper poses no such risks or does not release such assets.
    - High-risk models or datasets should be released with appropriate use restrictions or safeguards.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets used in the paper properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Appendix 2 lists the public pretrained models, datasets, and baseline methods used in the experiments, cites their original sources, and states the corresponding license or terms followed. Third-party datasets and checkpoints are not redistributed as new assets.

Guidelines:

- Cite the original paper that produced the code package, dataset, or model.
- State the version and license of each asset where possible.
- Respect copyright and terms of service for any data source.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not release a new dataset, benchmark, pretrained model, or code asset as part of the current submission. Generated images are experimental outputs used to evaluate the method.

Guidelines:

- The answer NA means that the paper does not release new assets.
- New assets should include documentation about training, license, limitations, and intended use.
- Assets should be anonymized at submission time where applicable.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation?

Answer: [N/A]

Justification: The paper does not use crowdsourcing or conduct research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing or research with human subjects.
- Details about participant instructions, screenshots, and compensation should be provided when applicable.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board approvals were obtained?

Answer: [N/A]

Justification: The paper does not involve human-subject studies, participant data collection, or crowdsourcing, so IRB approval is not applicable.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing or research with human subjects.
- IRB or equivalent approval should be described when human-subject research is conducted.
- Initial submissions should not include information that breaks anonymity.

16. **Declaration of LLM Usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

Answer: [N/A]

Justification: LLMs are not an important, original, or non-standard component of the DSF method or experiments. Any writing or formatting assistance, if used, does not affect the core methodology, scientific claims, or experimental results.

Guidelines:

- The answer NA means that the core method development does not involve LLMs as an important, original, or non-standard component.
- Writing, editing, or formatting use that does not affect the core methodology does not require declaration.