Deep Support Vectors

Junhoo Lee Hyunho Lee Kyomin Hwang
Nojun Kwak*
Seoul National University
{mrjunoo, hhlee822, kyomin98, nojunk}@snu.ac.kr

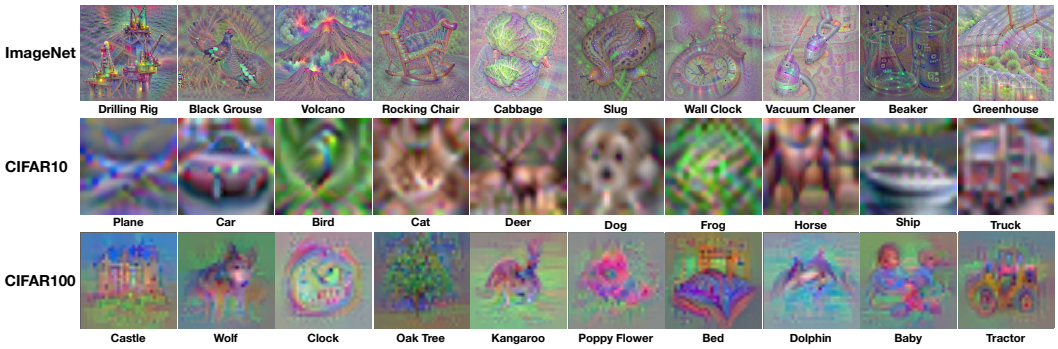

Figure 1: Generated images using a model trained with the ImageNet, CIFAR10, and CIFAR100 datasets, respectively. **Each image was generated without referencing the original training data.**

Abstract

Deep learning has achieved tremendous success. However, unlike SVMs, which provide direct decision criteria and can be trained with a small dataset, it still has significant weaknesses due to its requirement for massive datasets during training and the black-box characteristics on decision criteria. This paper addresses these issues by identifying support vectors in deep learning models. To this end, we propose the DeepKKT condition, an adaptation of the traditional Karush-Kuhn-Tucker (KKT) condition for deep learning models, and confirm that generated Deep Support Vectors (DSVs) using this condition exhibit properties similar to traditional support vectors. This allows us to apply our method to few-shot dataset distillation problems and alleviate the black-box characteristics of deep learning models. Additionally, we demonstrate that the DeepKKT condition can transform conventional classification models into generative models with high fidelity, particularly as latent generative models using class labels as latent variables. We validate the effectiveness of DSVs using common datasets (ImageNet, CIFAR10 and CIFAR100) on the general architectures (ResNet and ConvNet), proving their practical applicability. (See Fig. 1)

1 Introduction

Although deep learning has gained enormous success, it requires huge amounts of data for training, and its black-box characteristics regarding decision criteria result in a lack of reliability. For example, CLIP [24] needs 400 million image pairs for training and Stable Diffusion XL (SDXL) [25] requires

*Corresponding Author

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-3752

5 billion images. This implies only a small number of groups can train foundation models from scratch. Also, the black-box nature makes it hard to anticipate the model's performance in different environments. For example, suppose we are to classify pictures of deer and most training deer images contain antlers. For the test images taken in winter, the performance will be worse as deer shed their antlers in winter. As modern deep learning models do not provide any decision criterion *i.e.*, black box, we cannot determine whether the domain of the model has shifted, or if the model is biased in advance, thus cannot anticipate the performance drop in this case.

Interestingly, these problems do not occur in previous state-of-the-art, support vector machines (SVMs), which require substantially less data, enabling almost anyone to train a model from scratch. Also, as it encodes the decision boundary explicitly, SVM can reconstruct the support vectors from the training dataset using the KKT condition. Since it is a white box, one can anticipate the test's performance in advance. In the deer classification example, if the model's support vectors of deer have prominent antlers, using that SVM is not appropriate for photos taken in winter.

In this paper, we tackle the natural limitations of deep learning – the need for large data and black-box characteristics – by extracting SVM features in deep learning models. In doing so, we introduce the DeepKKT condition for deep models, which corresponds to the KKT condition in traditional SVMs. By either selecting deep support vectors (DSVs) from training data or generating them from already trained deep learning models, we show DSVs can play a similar role to conventional support vectors. Like support vectors can reconstruct SVM, we can reconstruct the deep models from scratch only with DSVs. Also, we show that DSVs encode the decision criterion visually, providing a global explanation for the trained deep model. Expanding beyond conventional support vectors, DSVs suggest that a trained deep classification model can also function as a latent generative model by utilizing logits as latent variables and applying DeepKKT conditions.

To this end, we generalize the KKT condition and define the DeepKKT condition considering that the data handled by a deep model is high-dimensional and multi-class. We demonstrate that the selected data points (selected DSVs) among the training data satisfying the DeepKKT condition are closer to the decision boundary than other training data, as evidenced by comparing entropy values. Also, we show that the calculated Lagrangian multiplier can reveal the level of uncertainty of the model for the sample in question. Additionally, we demonstrate that the DSVs outperform existing algorithms in the few-shot dataset distillation setting, where only a portion of the training set is used, indicating that DSVs exhibit characteristics similar to SVMs. Moreover, we confirm that modifying existing images using information obtained from DSVs allows us to change their class at will, verifying that DSVs can meaningfully explain the decision boundary. Finally, by using soft labels as latent variables in ImageNet, we generate unseen images with high fidelity.

Our contributions are as follows:

- By generalizing the KKT condition for deep models, we propose the DeepKKT condition to extract Deep Support Vectors, which are applicable to general architectures such as ConvNet and ResNet.
- We verify that DeepKKT can be used to extract and generate DSVs for common image datasets such as CIFAR10, CIFAR100, SVHN, and ImageNet.
- DSVs are better than existing algorithms in few-shot dataset distillation problems.
- DeepKKT acts as a novel model inversion algorithm that can be applied in practical situations.
- By using the DeepKKT condition, we not only show the trained deep models can reconstruct data, but also can serve as latent generative models by using logits as latent.

2 Related Works

2.1 SVM in Deep Learning

Numerous studies have endeavored to establish a connection between deep learning and conventional SVMs. In the theoretical side, [28] demonstrated that, in an overparameterized setting with linearly separable data, the learning dynamics of a model possessing a logistic tail are equivalent to those of a support vector machine, with the model's normalized weights converging towards a finite value. Following this, [15] extended this equivalence to feedforward networks. This line of research relies on strong assumptions such as full-batch training and non-residual architecture without data

augmentation. There also exists a body of work on integrating SVM principles into deep learning, often referred to as DeepSVM, aiming to leverage SVM's desirable properties [30, 29, 27, 23, 21]. DeepSVM integrates SVM components, specifically using them as feature extractors to derive meaningful, human-crafted features.

In contrast, our work does not modify or incorporate SVM architectures. Instead, we focus on identifying support vectors directly within deep learning models, thereby bridging the gap between deep learning and support vector machines in a more fundamental manner. Despite these advancements, there remains a lack of research that directly connects support vectors through a theoretical lens of equivalence. In this study, we address this gap by introducing the DeepKKT condition, a KKT condition tailored for deep learning, allowing us to apply the concept of support vectors in a practical deep learning context.

We show that reconstructing support vectors in deep models is indeed feasible, and obtaining high-quality support vectors is achievable under much less restrictive conditions compared to prior work.

2.2 Model Inversion Through the Lens of Maximum Margian

There is a line of research utilizing the stationarity condition, a part of the KKT condition, for model inversion. [6] firstly exploited the KKT condition for model generation, adopting SVM-like architectures. They normally conducted experiments with binary classification, a 2-layer MLP, and full-batch gradient descent. [1] extended these experiments to multi-label classification by adapting their existing architecture to a multi-class SVM structure. To ensure the generated samples lie on the data manifold, they initialized with the dataset's mean, implying the adoption of some prior knowledge of the data. Similarly, [34] generated images through the stationarity condition, also adopting the mean-initialization and conducting experiments on the CIFAR10 [11], MNIST [4] and downsampled CelebA [14] datasets. Their work generally focused on low-dimensional, labeled datasets with a small number of classes such as CIFAR10 and MNIST, consistent with the traditional SVM setting of binary-labeled, low-dimensional datasets. In contrast, we extended our experiments to high-dimensional datasets with many classes, an area traditionally dominated by deep learning. Specifically, we conducted experiments on ImageNet [26] using a pretrained ResNet50 model following the settings described in the original paper [7].

Furthermore, previous works have concentrated on **reconstructing the training** dataset. In contrast, similar to generative models, our work focuses on **generating unseen** data from noise using a classification model. Additionally, we emphasize the original meaning of 'support vectors'. Unlike other approaches, our Deep Support Vectors (DSVs) adhere to the traditional role of support vectors: they explain the decision criteria, and a small number of DSVs can effectively reconstruct the model.

2.3 Dataset Distillation

Dataset distillation [2, 31, 13] fundamentally aims to reduce the size of the original dataset while maintaining model performance. The achievement also involves addressing privacy concerns and alleviating communication issues between the server and client nodes. The dataset distillation problem is typically addressed under the following conditions: 1) Access to the entire dataset for gradient matching, 2) Possession of snapshots from all stages of the model's training phase, which are impractical settings for practical usage [12]. Furthermore, these algorithms typically require Hessian computation, which imposes a heavy computational burden.

In SVM, the model can be reconstructed using support vectors. This reconstruction is more practical compared to previous dataset distillation methods, as it does not require any of the restrictive conditions. Likewise, because Deep Support Vectors (DSVs) also do not require these conditions and are Hessian-free, they can play the role of distillation under practical conditions.

3 Preliminaries

3.1 Notation

In the SVM formulation, $\tilde{w}(:= [w; b])$ represents the concatenated weight vector w and bias b. Each data instance, expanded to include the bias term, is denoted by $\tilde{x}_i(:= [x_i; 1])$, while the corresponding binary label is represented by $y_i \in \{\pm 1\}$. The Lagrange multipliers are denoted by α_i's.

 https://doi.org/10.52202/079017-3752

Transitioning to the context of deep learning, we denote the parameter vector of a neural network model by θ, which, upon optimization, yields θ^* as the set of learned weights. The mapping function $\Phi(x_i;\theta)$ represents the transformation of input data into a C-dimensional logit in a C-class classification problem in a manner dictated by the parameters θ, *i.e.*, $\Phi(x_i;\theta) = [\Phi_1(x_i;\theta), \cdots, \Phi_C(x_i;\theta)]^T \in \mathbb{R}^C$. We define the **score** as the logit of a target class, *i.e.*, $\Phi_{y_i}(x_i;\theta)$. If the score is the largest among logits, *i.e.*, $\arg\max_c \Phi_c(x_i;\theta) = y_i$, then it correctly classifies the sample. The Lagrange multipliers adapted to the optimization in deep learning are represented by λ_i's. (x, y) denotes a pair of input and output and $\mathcal{I}$ is the index set with $|\mathcal{I}| = n$.

3.2 Support Vector Machines

The fundamental concept of Support Vector Machines (SVMs) is to find the optimal hyperplane that classifies the given data. This hyperplane is defined by the closest data points to itself known as support vectors, and the distance between the support vectors and the hyperplane is termed the margin. The hyperplane must classify the classes correctly while maximizing the margin. This leads to the following KKT conditions that an SVM must satisfy: **(1) Primal feasibility:** $\forall i,\ y_i\tilde{w}^T\tilde{x}_i \geq 1$, **(2) Dual feasibility:** $\forall i,\ \alpha_i \geq 0$, **(3) Complementary slackness:** $\alpha_i \left(y_i\tilde{w}^T\tilde{x}_i - 1\right) = 0$ and **(4) Stationarity:** $\tilde{w} = \sum_{i=1}^n \alpha_i y_i \tilde{x}_i$.

The primal and dual conditions ensure these critical values correctly classify the data while being outside the margin. The complementary slackness condition mandates that support vectors lie on the decision boundary. The stationarity condition ensures that the derivative of the Lagrangian is zero.

The final condition, stationarity, offers profound insights into SVMs. It underscores that support vectors encode the decision boundary $\tilde{w}$. Consequently, identifying the decision hyperplane in SVMs is tantamount to pinpointing the corresponding support vectors. This implies that with a trained model at our disposal, we can reconstruct the SVM in two distinct ways:

1. **Support Vector Selection:** From the trained model, we can extract support vectors among the training data that inherently encode the decision hyperplane.

2. **Support Vector Synthesis:** Alternatively, it is feasible to generate or synthesize support vectors, even in the absence of a training set, which can effectively represent the decision hyperplane by generating samples that satisfy $|\tilde{w}^T\tilde{x}| = 1$.

4 Deep Support Vector

This section presents the specific conditions that DSVs (Deep Support Vectors) must satisfy and discusses how to get an optimization loss to meet these conditions.

4.1 DeepKKT

SVM's Relationship with Hinge loss We start our discussion by focusing on the *hinge loss,* a continuous surrogate loss for the primal feasibility, and its gradient:

$$\begin{aligned} \text{Hinge Loss: } L_h(x_i, y_i; \tilde{w}) &= \frac{1}{n}\sum_{i=1}^n \max(0, 1 - y_i(\tilde{w}^T\tilde{x}_i)), \\ \nabla_{\tilde{w}} L_h &= \frac{1}{n}\sum_{i=1}^n \begin{cases} 0, & \text{if } y_i(\tilde{w}^T\tilde{x}_i) \geq 1 \\ -y_i\tilde{x}_i, & \text{otherwise.} \end{cases} \end{aligned} \tag{1}$$

With Eq. (1), the stationarity condition of SVM becomes

$$w^* := -\sum_{i=1}^n \alpha_i \nabla_w L_h(x_i, y_i; w^*), \quad \text{s.t. } \alpha_i \geq 0. \tag{2}$$

Generalization of conventional KKT conditions In this paper, we extend the KKT conditions to deep learning. In doing so, the two main hurdles of a deep network different from a linear binary SVM are 1) the nonlinearity of $\Phi(x_i;\theta)$ taking the role of $\tilde{w}^T\tilde{x}_i$ and 2) multi-class nature of a deep learning model.

img/cls	ratio (%)	Random	selected DSVs
50	1	46.16 ± 1.93	48.91 ± 0.90
10	0.2	30.08 ± 1.96	33.69 ± 2.05
1	0.02	14.26 ± 0.99	16.83 ± 0.29

Table 1: In coreset selection benchmarks using the CIFAR-10 dataset, the DeepKKT condition is used as the selection criterion. Images with the highest λ for each class were chosen to train a network.

Considering that the role of the primal feasibility condition is to correctly classify x_i into y_i, we can enforce the score $\Phi_{y_i}(x_i;\theta)$ for the correct class y_i to take the maximum value among all the logits with some margin ϵ, *i.e.*,

$$\Phi_{y_i}(x_i;\theta^*) - \max_{c\neq y_i, c\in[C]} \Phi_c(x_i;\theta^*) \geq \epsilon, \tag{3}$$

We can relax this discontinuity with a continuous surrogate function $-L(\Phi(x_i;\theta^*), y_i)$, which is the negative loss function to maximize. Note that if we take the cross-entropy loss for L, it becomes

$$-L_{ce} = \Phi_{y_i}(x_i;\theta^*) - \log \sum_{c=1}^{C} \exp(\Phi_c(x_i;\theta^*)), \tag{4}$$

which takes a similar form as Eq. (3) and the negative loss can be maximized to meet the condition.

Now that we found the analogy between $y_i\tilde{w}^T\tilde{x}_i$ and $-L(\Phi(x_i;\theta^*), y_i)$, $y_i\tilde{x}_i(= \nabla_{\tilde{w}}(y_i\tilde{w}^T\tilde{x}_i))$ corresponds to $-\nabla_{\theta^*}L(\Phi(x_i;\theta^*), y_i)$. Thus, the stationary condition $\tilde{w} = \sum_{i=1}^{n}\alpha_i y_i\tilde{x}_i$ in SVMs can be translated into that of deep networks such that

$$\theta^* = -\sum_{i=1}^{n} \lambda_i \nabla_{\theta^*} L(\Phi(x_i;\theta^*), y_i). \tag{5}$$

This condition is a generalized formulation of Eq. (2), where we substitute the linear model $\tilde{w}^T\tilde{x}$ with a nonlinear model $\Phi(x;\theta)$, and the binary classification hinge loss L_h with multi-class classification loss L. Furthermore, the stationarity condition reflects the dynamics of overparameterized deep learning models. We provide an analogy with respect to [28] in Appendix C.

However, these conditions are not enough for deep learning. As mentioned before, we are interested in dealing with high dimensional manifolds. Compared to the problems dealt with in the classical SVMs, the input dimensions of deep learning problems are typically much higher. In this case, the data are likely to lie along a low-dimensional latent manifold $\mathcal{M}$ inside the high-dimensional space. To make a generated sample be a plausible DSV, it not only should satisfy the generalized KKT condition but also should lie in an appropriate data manifold, *i.e.*, $x \in \mathcal{M}$.

Finally, we can rewrite the new **DeepKKT condition** as follows:

$$\begin{aligned}
&\text{Primal feasibility:} && \forall\, i \in \mathcal{I}, \quad \arg\max_c \ \Phi_c(x_i;\theta^*) = y_i \\
&\text{Dual feasibility:} && \forall\, i \in \mathcal{I}, \quad \lambda_i \geq 0, \\
&\text{Stationarity:} && \theta^* = -\sum_{i=1}^{n} \lambda_i \nabla_\theta \mathcal{L}(\Phi(x_i;\theta^*), y_i), \\
&\text{Manifold:} && \forall\, i \in \mathcal{I}, \quad x_i \in \mathcal{M}.
\end{aligned} \tag{6}$$

4.2 Deep Support Vectors, From Dust to Diamonds

Sec. 4.1 explored the KKT conditions in the context of deep learning and how these conditions can be used to formulate a loss function. It is important to note that our goal is not to construct the model Φ, but rather to generate **support vectors of an already-trained model Φ with its parameter θ^***.

To reconstruct support vectors from a trained deep learning model, sampling or synthesizing support vectors is essential. By replacing the optimization variable θ with x, we shift our focus to utilizing the DeepKKT conditions for generating or evaluating input x rather than θ. This adjustment necessitates a consideration of the data's inherent characteristics, specifically its multiclass nature and the tendency of the data to reside on a lower-dimensional manifold in an ambient space.

 https://doi.org/10.52202/079017-3752

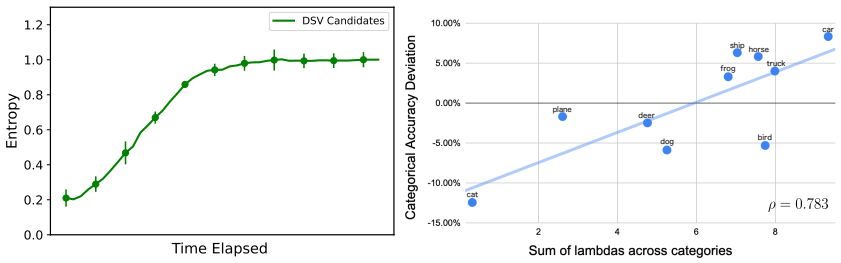

Figure 2: Characteristics of DSVs; (Left) Entropy change of DSV candidates over time, (Right) Correlation between classwise mean test accuracy and the sum of λ's

Primal Feasibility Firstly, the primal feasibility condition in Eq. (6) mandates support vectors be correctly classified by the trained model θ^*. As presented in Sec. 4.1, instead of Eq. (3), we use a surrogate function for the loss:

$$L_{\text{primal}} = \frac{1}{n}\sum_{i=1}^{n} L_i, \quad \text{where} \quad L_i = \begin{cases} 0 & \text{if } \arg\max_c \Phi_c(x_i;\theta^*) = y_i, \\ L(\Phi(x_i;\theta^*), y_i) & \text{otherwise.} \end{cases} \tag{7}$$

This is designed to match the primal condition by mimicking the Hinge loss. When each DSV is correctly classified, no loss is incurred. Otherwise, we adjust the DSVs to align them with the correct target, effectively optimizing the support vectors. This approach also implicitly enforces the complementary slackness condition, as L_i decreases confidence in the incorrect classification.

Here, L can be any loss function and we have employed the cross-entropy loss in our experiments.

Stationarity Secondly, the stationarity condition can be used directly as a loss function. Since we are extracting DSVs from the trained model $\Phi(\cdot;\theta^*)$, we construct this loss as follows:

$$L_{\text{stat}} = D(\theta^*, -\sum_{i=1}^{n} \lambda_i \nabla_\theta L(\Phi(x_i;\theta^*), y_i)). \tag{8}$$

For the distance measure D, any metric can be used; we have chosen to use the l_1 distance to suppress the effect of outliers. It is crucial to remember that our objective is to find DSVs and the optimization is done for the primal and dual variables x_i and λ_i and not for the parameter θ. For this, we require one forward pass and two backward passes; one for $\nabla_\theta L$ and the other for $\nabla_{x_i} D$. The overall computational cost is quite low, as we optimize only a small number of samples.

Moreover, as shown in Algorithm 1 (Appendix H), we satisfy the dual condition by ensuring the Lagrange multipliers λ_i's are greater than zero and disqualify any x_i's from being a support vector candidate if during optimization λ_i becomes less than zero. The condition that is not explicitly satisfied is the complementary slackness. To directly fulfill the functional boundary for support vectors as specified in Sec. 3.2, we would need to be able to calculate the distance between functions, which is not only abstract but also requires a second-order computation cost. Therefore, we adopted a relaxed version of the KKT conditions that excludes this requirement. Furthermore, as demonstrated in Sec. 5.1, we have shown that DSVs implicitly satisfy the complementary slackness condition. This implies that DSVs meet every condition introduced in conventional SVM.

Manifold Condition: Reflecting High-Dimensional Dynamics of Deep Learning As modern deep learning deals with extremely high-dimensional spaces, imposing additional constraints other than the primal feasibility and stationarity conditions is needed so that DSVs reside in the desired data manifold. To achieve this, we add a manifold condition, which enforces that the DSVs lie on the data manifold. By selecting DSVs that are in the intersection of the solution subspace and the data manifold, we can properly represent both the model and the training dataset.

To extract DSVs from the manifold, we assume that the model is well-trained, meaning it maintains consistent decisions despite data augmentation. In other words, the model should classify DSVs invariantly even after augmentation. To ensure this, we enforce that the augmented DSVs ($\mathcal{A}(x)$ where $\mathcal{A}$ denotes augmentation function) also meet the primary and stationarity conditions.

Also, we exploit traditional image prior [33, 16], total variance L_{tot} and size of the norm L_{norm} to make DSVs lie in the data manifold. L_{tot} is calculated by summing the differences in brightness between neighboring pixels, reducing unnecessary noise in an image, and maintaining a natural appearance. L_{norm}, taking a similar role, penalizes the outlier and preserves important pixels.

Finally our DSV is obtained as follows where $\mathbb{E}_{\mathcal{A}}$ represents expectation over augmentations:

$$\text{DSV} = \arg\min_{x} \mathbb{E}_{\mathcal{A}} \left[L_{\text{stationarity}}(\mathcal{A}(x)) + \beta_1 L_{\text{primal}}(\mathcal{A}(x)) + \beta_2 L_{\text{tot}}(x) + \beta_3 L_{\text{norm}}(x) \right]. \quad (9)$$

One might wonder if there is a better sampling strategy than using DSVs, such as sampling far from the decision boundary instead of near it. We argue that in a high-dimensional data manifold, most data points are located close to the decision boundary because, in a data-scarce, high-dimensional space, every sample matters and thus would serve as a DSV.

5 Experiments

5.1 DSVs: Revival of Support Vectors in Deep Learning

DSVs meet SVM characteristics As discussed in Section 3.2, the principle of complementary slackness within the KKT conditions suggests that support vectors should be situated on the decision boundary, implying that support vectors typically exhibit high uncertainty from a probabilistic perspective, *i.e.*, they possess high entropy. While DeepKKT does not explicitly incorporate the complementary slackness condition due to computational costs and ambiguity, Fig. 2a suggests that DSVs implicitly fulfill this condition; During the training process, we observe an increase in the entropy of DSV candidates, hinting that the generated DSVs are close to the decision boundary.

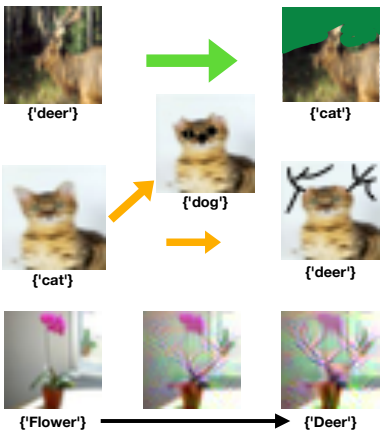

Figure 3: Model predictions for original versus DSV-informed edited images. (Top) Images were altered manually based on decision criteria derived from DSVs, influencing the model's prediction. (Bottom) Images were altered based on DeepKKT loss.

In addition, we can infer the importance of a sample in the decision process by utilizing the DeepKKT condition. We trained the Lagrangian multiplier λ for each test image. Figure 2b shows a strong correlation between the sum of λ values for each class and its test accuracy. This finding is intriguing because, despite the model achieving nearly 100% accuracy during training due to overparameterization, DSVs provide insights into categorical generalization in the test phase. Not only does measuring its credibility indicate that a large λ refers to an 'important' image for training, but λ could also serve as a natural core-set selection measure. Table 1 shows this to be true. On the CIFAR-10 [11] dataset, we selected images with high λ values and retrained the network with the selected images. In this case, the selected DSVs show higher test acccuracies compared to random selection. This characteristic resembles that of support vectors, as the model can be reconstructed with support vectors.

Finally, Fig. 1 demonstrates the high fidelity of the generated DSVs, providing practical evidence that these DSVs lie on the data manifold. Similar to how support vectors are reconstructed in an SVM, the DeepKKT condition enables the reconstruction of these vectors without referencing training data. This shows the effectiveness and adaptability of our DeepKKT in capturing key data features.

DSVs for Few Shot Dataset Distillation DeepKKT emerges as a pioneering algorithm tailored for practical dataset distillation. DSVs addresses two critical concerns: 1) Protecting private information through data synthesis, and 2) Reducing the communication load by minimizing the size of data transmission. Traditional distillation algorithms encounter a fundamental paradox; as data predominantly originate from edge devices like smartphones [18, 17, 10], the requirement to access the entire dataset introduces significant communication overhead and heightens privacy concerns.

https://doi.org/10.52202/079017-3752

img/cls	shot/class	ratio (%)	DC [37]	DSA [35]	DM [36]	DSVs
1	0	0	-	-	-	21.68 ± 0.80
	1	0.02	16.48±0.81	15.41±1.91	13.03±0.15	**22.69** ± 0.38
	10	0.2	19.66±0.78	21.15±0.58	22.42±0.43	-
	50	1	25.90±0.62	26.01±0.70	24.42±0.29	-
	500	10	28.06±0.61	28.20±0.63	25.06±1.20	-
10	0	0	-	-	-	30.35 ± 0.99
	10	0.2	25.06±1.20	26.67±1.04	29.77±0.66	**37.90** ± 1.69
	50	1	36.44±0.52	36.63±0.52	36.63±0.52	-
	500	10	43.55±0.50	44.66±0.59	47.96±0.95	-
50	0	0	-	-	-	39.35 ± 0.54
	50	1	41.22±0.90	41.29±0.45	48.93±0.92	**53.56** ± 0.73
	500	10	52.00±0.59	52.19±0.53	60.59±0.41	-

Table 2: Performance of Few-shot learning on CIFAR10. 'img/cls' and 'shot/class' refer to the per-class number of generated images and the training samples used in generating the distilled dataset, respectively. 'ratio' is the ratio of the seen samples among the entire training samples. 0 shot refers to the distillation task performed without any access to the training data.

Our DeepKKT relies solely on a pre-trained model without relying on the training dataset. This unique approach eliminates the need for edge devices to store or process large volumes of private data. As shown in Table 2, while traditional methods suffer significant performance drops under these scenarios and are incapable of implementing zero-shot scenarios. Conversely DeepKKT remains effective, requiring only minimal data: a single image per sample (*i.e.*, initialization with real data), or in some cases, no images at all. For the zero-image setting, we initialized the images with data from other datasets to ensure diversity.

DSVs Encode the Decision Criteria Visually Our findings suggest that DSVs not only satisfy the conditions of classical support vectors but also offers a global explanation of visual information. Fig. 3 experimentally verifies our claims and illustrates the practical use of DSVs, *e.g.*, analysis of Fig. 1-Cifar10 reveals the decision criteria for classifying deer, cats, and dogs: 1) DSVs highlight antlers in deer, signifying them as a distinctive characteristic. 2) Pointed triangular ears are a recurring feature in DSVs of cats. 3) For dogs, a trio of facial dots holds significant importance. Using these observations, we altered a deer's image by erasing its antlers and reshaping its ears to a pointed contour, which reduced the model's confidence in classifying it as a deer, and caused the model to misclassify it as a cat. Similarly, by smoothing the ears of a cat image to diminish its classification confidence and then adding antlers or three facial dots, we influenced the model to reclassify the image as a deer or a dog, respectively. Additionally, the DeepKKT-altering case in Fig. 3-Bottom supports our assertions. Altering a flower image to resemble a deer class by changing the target class in the primal and dual feasibility loss, antlers grew similar to our manipulation.

This discovery holds significant implications about making models responsible; it introduces a qualitative aspect to assessing model performance. Consider a deer classification problem again. The model in our study would be less suitable, as evidenced by Fig. 1-Cifar10-deer, which indicates the model's reliance on antlers for identifying deer – a feature not present in winter. This shows that DSVs enable us to conduct causal predictions by qualitatively analyzing models, as SVM does.

5.2 Unlocking the potential of classifier as generator with DeepKKT

Practical Model Inversion with DeepKKT In cloud environments or APIs, models are deployed with the belief that although they are sometimes trained with sensitive information, their black-box nature prevents users from inferring the data. This belief makes it possible to deploy sensitive models. However, as demonstrated in Fig 1, this belief is no longer valid. Fig. 4 further illustrates that model inversion remains feasible even in practical scenarios such as transfer learning scenarios, where only specific layers of a foundation model are fine-tuned. Remarkably, DeepKKT condi-

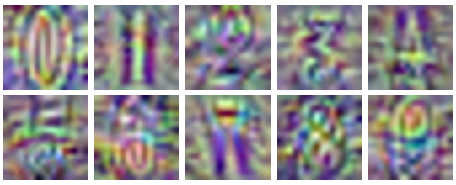

Figure 4: DSVs generated by a model that underwent transfer learning from CIFAR-10 to SVHN. During transfer learning, only the last layer is updated by SVHN.

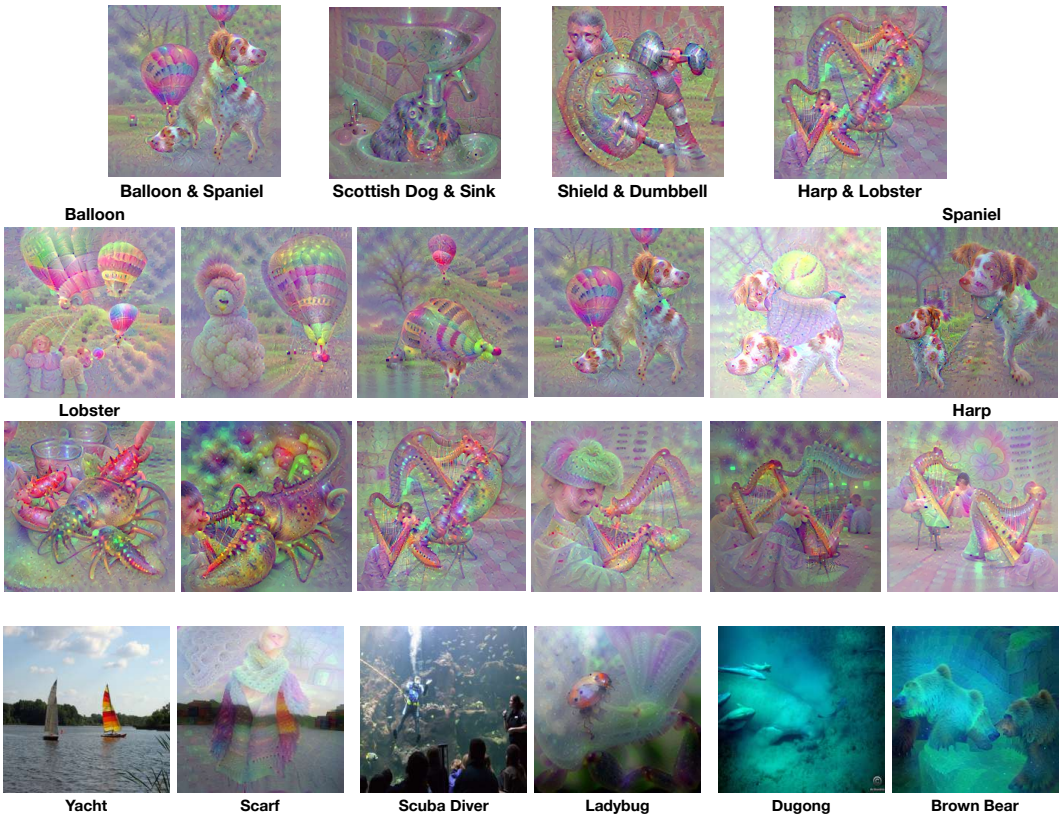

Figure 6: (Top) Generated DSVs using soft labels: δ was set 0.6, *i.e.*, soft label $y = 0.4y_{\text{left}} + 0.6y_{\text{right}}$. (Middle) Examples of latent (soft-label) interpolation. (Bottom) Image Editing through latent.

tions enable model inversion even in these challenging environments, suggesting that they can be applied to a subset of the parameter space rather than the entire parameter space *i.e.*, a more relaxed condition.

Classifier as Latent generative model Considering the impressive capabilities of DSVs in the image generation domain, and the geometric interpretation that DSVs are samples near the decision boundaries, it is noted that enforcing the DeepKKT condition resembles the diffusion process. In each iteration, the DSV develops through the DeepKKT condition as follows:

$$x_{t+1} = x_t - \eta \cdot ([\nabla_x L_{\text{stat}}(\mathcal{A}(x_t)) + \beta_2 L_{\text{tot}}(x_t) + \beta_3 L_{\text{norm}}(x_t)] + \beta_1 \nabla_x L_{\text{primal}}(\mathcal{A}(x_t))) \,. \quad (10)$$

This is similar to the generalized form of the score-based diffusion process:

$$x_{t+1} = x_t + \epsilon_t \cdot (\nabla_x \log p(x_t) + \gamma \nabla_x \log p(y|x_t)) \,. \quad (11)$$

The first three loss terms in Eq. (10) aim to maximize the score ($\nabla_x \log p(x_t)$), while the last term, the primal feasibility term, corresponds to the guidance term ($\gamma \nabla_x \log p(y|x_t)$). As shown in Fig. 5, when only the primal loss term is used, meaningful DSV samples are not generated. This indicates that the other losses (stationarity and manifold terms) function update the image towards manifold *i.e.*, score function. From this perspective, an arbitrarily assigned label y can be used as a latent variable for guidance.

To experimentally verify this, we performed a latent interpolation task and image editing, which is common

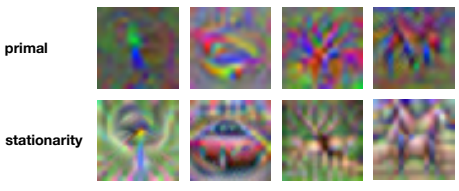

Figure 5: Results showing DeepKKT images created solely by the primal condition or by the stationary condition. A sole usage of the primal condition shows low fidelity.

https://doi.org/10.52202/079017-3752

in generative models [8, 3] By mixing different labels ($y_i = (1-\delta)y_a + \delta y_b$, where $y_a \neq y_b$), we generated DSVs as depicted in Fig. 6. When generating DSVs with these mixed soft labels, the generated DSVs semantically represent the midpoint between the two classes. A generated DSV either simply contains both images (the case of “balloon" and “spaniel") or semantically ‘fuse’ objects (the case of “lobster" and “harp", producing an image of a harp made out of lobster claws). For the image editing task, we assigned the latent variable to the desired class and then aligned the image using DeepKKT loss. The result was quite surprising: the method successfully transferred the image to the desired class while maintaining the original structure. For example, the sail of a yacht was seamlessly transformed into the shape of a scarf. This task was impossible with other methods; in diffusion models, for instance, a mask would be needed to edit the image seamlessly.

The fact that the generated images correctly merge the semantics of the classes suggests a couple of significant implications: 1) 1) **New Generative Model**: This approach offers a new type of generative model as an alternative to GANs and diffusion models. It can handle the same task without the need for training a specific model, as it leverages existing classification models for generative purposes. Furthermore, it is lightweight compared to diffusion models. For example, as the model size of a pretrained ResNet50 for ImageNet is only one-twentieth of that for SDXL [25], DSVs show a potential to leverage existing classification models for generative purposes. 2) **Exploration of Classification Model Generalization**: Unlike other generative models, classification models are trained simply to predict the label of an image. Yet, in latent interpolation and editing tasks, they demonstrate an understanding of semantics. This implies that, despite being trained to memorize class labels, the models grasp the overall semantics of the dataset. As they can generate seemingly unseen samples by interpolation and editing.

6 Conclusion

In this paper, we redefined support vectors in nonlinear deep learning models through the introduction of Deep Support Vectors (DSVs). We demonstrated the feasibility of generating DSVs using only a pretrained model, without accessing to the training dataset. To achieve this, we extended the KKT (Karush-Kuhn-Tucker) conditions to DeepKKT conditions and the proposed method can be applied to any deep learning models.

Akin to SVMs, the DeepKKT condition effectively encodes the decision boundary into DSVs. DSVs can reconstruct the model, making them useful for dataset distillation. Additionally, their visual encoding of the decision criteria can serve as a global explanation, helping to understand the model’s overall behavior and decisions. Furthermore, the DeepKKT condition transforms a classification model into a generative model with high fidelity. Not only can it sample data, but it also generalizes well, allowing the use of labels as latent variables.

Acknowledgement

Thank you for Hyunjin Kim, Wonhak Park, and Yeji Song for detailed discussion and feedback. This work was supported by NRF grant (2021R1A2C3006659) and IITP grants (RS-2022-II220953, RS-2021-II211343), all funded by MSIT of the Korean Government.

References

[1] Gon Buzaglo, Niv Haim, Gilad Yehudai, Gal Vardi, and Michal Irani. Reconstructing training data from multiclass neural networks, 2023.

[2] 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.

[3] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation, 2018.

[4] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[5] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[6] 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.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[8] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018.

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *CoRR*, abs/1610.02527, 2016.

[11] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[12] Hyunho Lee, Junhoo Lee, and Nojun Kwak. Practical dataset distillation based on deep support vectors, 2024.

[13] Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 1100–1113. Curran Associates, Inc., 2022.

[14] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738, 2015.

[15] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. In *International Conference on Learning Representations*, 2020.

[16] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. *CoRR*, abs/1412.0035, 2014.

 https://doi.org/10.52202/079017-3752

[17] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[18] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016.

[19] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *CoRR*, abs/1610.08401, 2016.

[20] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[21] Onuwa Okwuashi and Christopher E. Ndehedehe. Deep support vector machine for hyperspectral image classification. *Pattern Recognition*, 103:107298, 2020.

[22] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703, 2019.

[23] Zhiquan Qi, Bo Wang, Yingjie Tian, and Peng Zhang. When ensemble learning meets deep learning: a new deep support vector machine for classification. *Knowledge-Based Systems*, 107:54–60, 2016.

[24] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[25] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn 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.

[26] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

[27] Hichem Sahbi. Totally deep support vector machines. *CoRR*, abs/1912.05864, 2019.

[28] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. In *International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, 2018. Poster presented at ICLR 2018.

[29] Yichuan Tang. Deep learning using linear support vector machines. *arXiv preprint arXiv:1306.0239*, 2013.

[30] Jingyuan Wang, Kai Feng, and Junjie Wu. Svm-based deep stacking networks. In *Proceedings of the AAAI conference on artificial intelligence*, pages 5273–5280, 2019.

[31] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

[32] Han Xu, Yao Ma, Haochen Liu, Debayan Deb, Hui Liu, Jiliang Tang, and Anil K. Jain. Adversarial attacks and defenses in images, graphs and text: A review, 2019.

[33] Hongxu Yin, Arun Mallya, Arash Vahdat, José M. Álvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. *CoRR*, abs/2104.07586, 2021.

[34] Runpeng Yu and Xinchao Wang. Generator born from classifier. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[35] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. *CoRR*, abs/2102.08259, 2021.

[36] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. *CoRR*, abs/2110.04181, 2021.

[37] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *CoRR*, abs/2006.05929, 2020.

 https://doi.org/10.52202/079017-3752

A Societal Impacts

Our paper is closely related to Responsible AI (RAI), especially in enabling qualitative assessments of models. Our approach provides visual and intuitive explanations of a model's decision-making criteria, offering insights that are both explanatory and responsible. Our approach of utilizing DSVs for RAI enables global explanations, surpassing traditional Explainable AI (XAI) methodologies, which usually focus on local explanations for individual inputs and cannot provide a global decision criterion. Furthermore, since our method is based on model inversion, it ensures safety and privacy. While the synthesized sets in Fig. 7 might appear similar to the selected sets, they do not replicate specific sample features. This is because DSVs represent a more generalized decision boundary, avoiding the inclusion of image-specific features. Consequently, DSVs enable all models using logistic loss to be more responsible.

Figure 7: Comparison of synthesized images (first row) created using the DeepKKT condition initiated from noise, and selected images (second row) from the CIFAR-10 training dataset. The selected images were chosen based on λ values, *i.e.*, each image has the highest λ in each class. Both synthesized and selected images demonstrate similarity at the pixel level sharing common features.

B Limitations and Future work

In this paper, we propose the DeepKKT condition, which can be applied universally to any deep models to generate deep support vectors (DSVs) that function similarly to support vectors in SVMs. However, it should be noted that the equivalence between DSVs in deep learning models and support vectors in SVMs is only described intuitively, not rigorously. We have shown experimentally in Fig. 7 and intuitively in Sec. C why the DeepKKT condition should be as we suggested, but we have not derived it with rigorous math. Proving this rigorously would be a meaningful research topic.

C Intutive explanation of DeepKKT condition

In DeepKKT, many conditions make sense, except for one. For instance, the primal feasibility condition and the manifold condition are reasonable, and the dual feasibility condition can be regarded as importance sampling. However, the most counterintuitive part is the stationarity condition:

$$L_{\text{stat}} = D(\theta^*, -\sum_{i=1}^{n} \lambda_i \nabla_\theta L(\Phi(x_i;\theta^*), y_i)) \tag{12}$$

In this section, we will explain the dynamics of DSVs in an overparameterized deep network and how it is connected to deep learning. Below is a quick analogy of [28] to illustrate this connection.

A deep learning model follows the following ODE:

$$w_{t+1} = w_t - \eta \nabla L(x, y; w_t). \tag{13}$$

Here, η is the learning rate and t is the optimization step. The loss L does not go to zero since deep learning models usually exploit a loss function with a logistic tail, such as the cross-entropy loss, and the gradient of the least confident sample (support vector) dominates overall gradient. Thus, there exists a convergence of the gradient direction $g_\infty := \hat{\nabla} L$. There also exists a time T where the

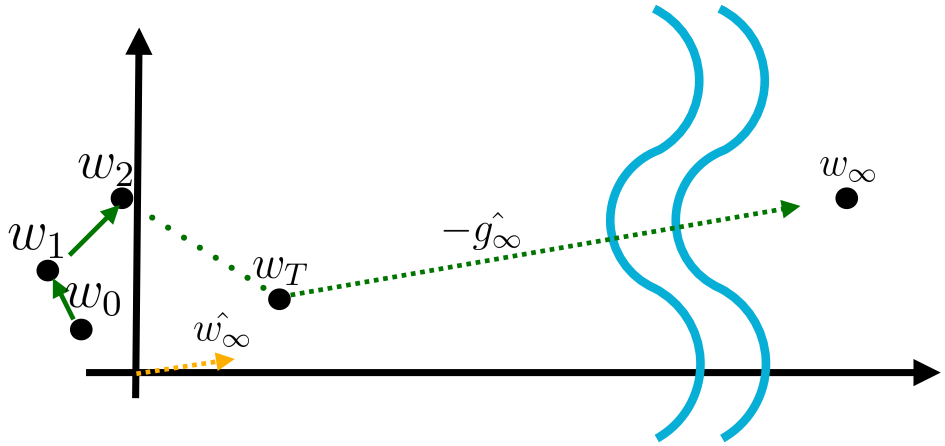

Figure 8: The stationarity condition with a logistic loss. Even though the direction of the gradient $\hat{g}$ converges, the size of the gradient does not go to zero. Therefore, the direction of the converged gradient weight $\hat{w_\infty}$ aligns with $\hat{g}$.

gradient direction converges to $g_\infty - \varepsilon$ for a sufficiently small ε. As illustrated in Fig. 8, w moves toward the direction of $-g_\infty$. Therefore, $\hat{w}_\infty \approx -g_\infty$.

This is for what stationarity condition wants to seek. The direction of g_∞, by using only a few support vectors.

D Implementation Details

To obtain the results in Table 2 and Fig. 4, the ConvNet architecture [5] was used for pretraining $\Phi(\cdot;\theta)$ on the SVHN dataset [20], a digit dataset with dimensions similar to CIFAR-10 [11]. For ImageNet, we used the ResNet50 model [7] with the original setting in the paper. Specifically, we used the pretrained model in torchvision library in pytorch [22]. For visualizing synthesized DSVs in ImageNet, we increased the contrast in 224x224 dimensions. When calculating L_{stat}, we averaged the distance per parameter. In Alg. 1, η was set to 5.

To synthesize DSVs in ImageNet, we used translation, crop, cutout, flip, and noise for augmentation, with hyperparameters set to 0.125, 0.2, 0.15, 0.5, and 0.01, respectively. In Eq. (9), we set α to 2e-5, β to 40, and γ to 1e-6. When calculating $L_{\text{stationarity}}$, we averaged the distance per parameter.

For dataset distillation in Table 2, we used translation, crop, flip, and noise for augmentation, with hyperparameters set to 0.125, 0.2, and 0.5, respectively. In Eq. (9), we set α to 2e-3, and both β and γ to 0. For retraining models with synthesized images, we used a learning rate of 1e-4 while the other parameters set to the default values of the Adam optimizer [9].

To obtain the pretrained weight θ^* for CIFAR10 and CIFAR100, we chose the ConvNet architecture [5], a common choice in deep learning. This architecture includes sequential convolutional layers followed by max pooling, and a single fully-connected layer for classification. The learning rate was set to 10^{-3} with a weight decay of 0.005 using the Adam optimizer. Additionally, we employed flipping and cropping techniques, with settings differing from those used for DSVs reconstruction to ensure fair comparison. For pretraining Φ on the Street View House Numbers (SVHN) dataset [20], a digit dataset with dimensions similar to CIFAR-10 [11], we exclusively trained the fully-connected layer of the CIFAR-10 pre-trained ConvNet. This approach resulted in a training accuracy of 80%.

E DSVs by Selection

Fig. 9 shows the selected images with large Lagrangian multipliers λ's, which correspond to the candidates used in Fig. 2b. Surprisingly, there is a meaningful match between the selected DSVs and the synthesized DSVs in the CIFAR-10 dataset, as shown in Fig. 7. This implies that synthesizing DSVs corresponds to reviving training data that lie on the boundary manifolds.

https://doi.org/10.52202/079017-3752

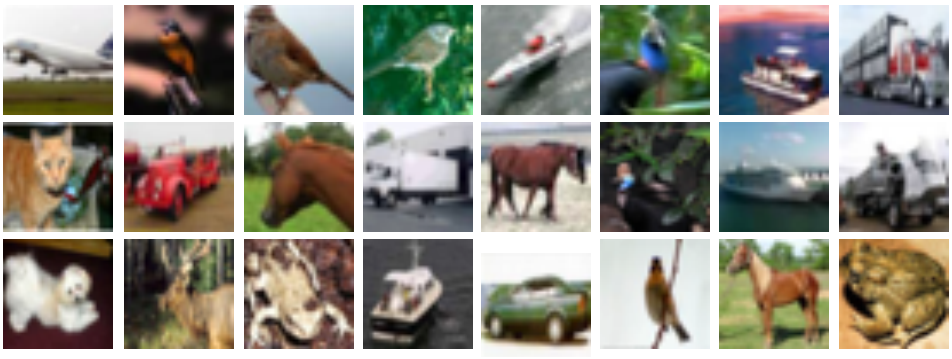

Figure 9: Images of DSV candidates (Selected in the CIFAR-10 dataset).

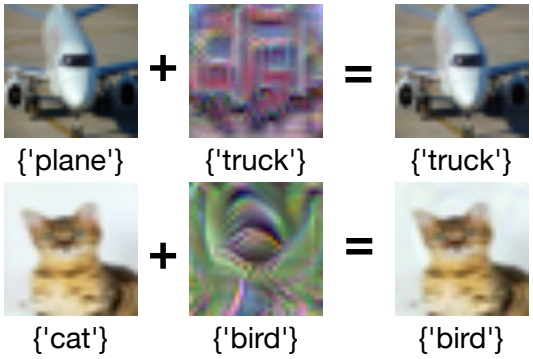

Figure 10: Examples of adversarial attack in the CIFAR-10 dataset

F More Characteristics of DSVs

DSVs Are Full of Discriminative Information In Fig. 10, we conducted an experiment by mixing a randomly sampled image from the real dataset with an image from the DSVs. Upon observation, the mixed image is virtually indistinguishable from an image obtained solely from the real dataset. It is noteworthy to highlight this situation resembles that of an adversarial attack [32, 19], yet we did not apply gradient descent to the image; we simply mixed two images. This suggests that the discriminative informational density in a single DSV image is substantially greater than that in a randomly sampled image. The fact that the DSV's characteristics remained dominant in the classification, underscores the significant role of DSVs in explaining the model's classification ability.

G More Examples

In Fig. 11, examples of latent interpolations between target labels are presented. The smoothness of these interpolations within the latent space indicates that the semantic information learned from the training data has been effectively applied during the DSV generation process. This observation provides evidence that the DeepKKT optimization successfully conducts the generative process.

Fig. 12 and 13 provide examples of deep support vectors generated using the CIFAR-100 and ImageNet datasets, respectively. Fig. 14 presents additional examples related to image editing.

Fig. 15 empirically supports on our assertion on decision criterion. Starting from CIFAR100 random images and CIFAR10-pretrained models, we edited the image CIFAR10 labels as latents. The edited images changes the image following decision criterions in generated DSVs. 1) For editing images to deer, antler grows. 2) For dog editing, facial dots are generated. 3) For cat editing, pointed traiangler features are generated.

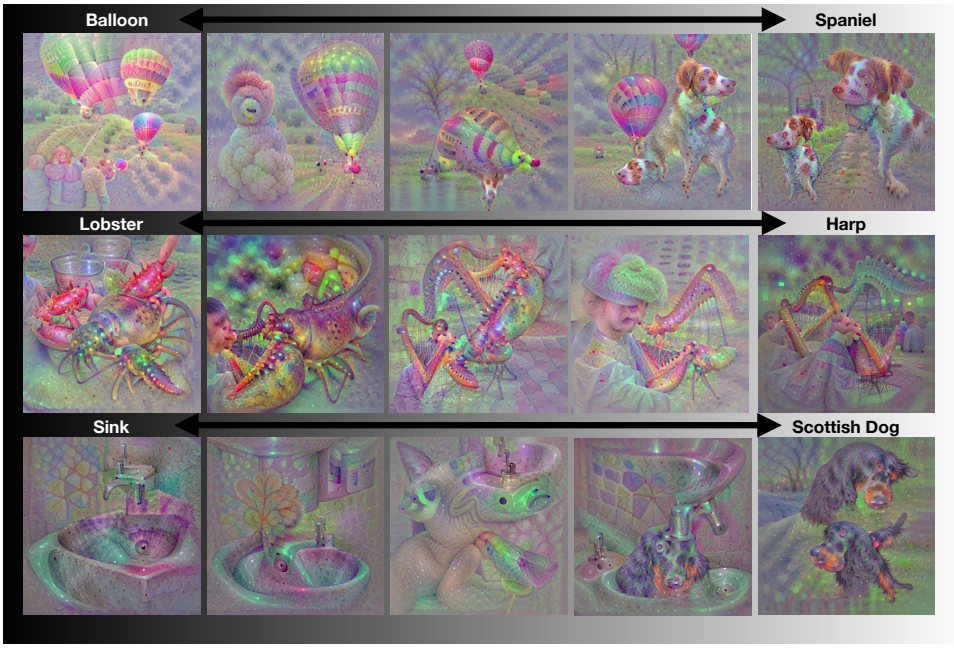

Figure 11: More examples of latent interpolation in the ImageNet dataset

H Algorithm

Alg. 1 presents our algorithm of generating Deep Support Vectors (DSVs). Initialized either from a noise $x_i^s \sim \mathcal{N}(0, I)$ or a real sample, it iterates to obtain the primal X^S and dual Λ^S variables.

Algorithm 1 Support Vector Refinement for Deep Learning Model

Require: Pretrained classifier $\Phi(\cdot;\theta)$, loss function L, augmentation function set A, number of DSV candidate N, number of class C, hyperparameters α, β
Ensure: Freeze classifier $\Phi(\cdot;\theta)$
Initialize $N \times C$ number of support vector candidates
for $i = 1$ to C **do**
    sample N number of (x_i^s, λ_i^s) for label $y_i^s = i$
end for
Define $X^S = \{x_i^s \mid i \in [C], s \in [N]\}$
Define $\Lambda^S = \{\lambda_i^s \mid i \in [C], s \in [N]\}$
repeat
    $L_{\text{primal}}(X^S) = \sum_{s=1}^{N} \sum_{i=1}^{C} L(\Phi(x_i^s;\theta), y_i^s)$
    $L_{\text{stationary}}(X^S) = \|\theta + \sum_{s=1}^{N} \sum_{i=1}^{C} \lambda_i^s y_i^s \nabla_\theta \Phi(x_i^s;\theta)\|_2^2$
    $L_{\text{kkt}}(X^S) = \beta_1 \cdot L_{\text{primal}}(X^S) + L_{\text{stationary}}(X^S)$
    $L_{\text{prior}} = \beta_2 \cdot L_{\text{tot}}(X) + \beta_3 L_{\text{norm}}(X)$
    Sample $f_A \in A$
    Define $AX^S = \{f_A(x_i^s) \mid x_i^s \in X^S\}$
    $L_{\text{akkt}}(X^S) = L_{\text{kkt}}(AX^S)$
    $L_{\text{total}}(X^S) = L_{\text{kkt}}(X^S) + \eta \cdot L_{\text{akkt}}(X^S) + L_{\text{prior}}$
    Update $X^S \leftarrow X^S + \nabla_{X^S} L_{\text{total}}(X^S)$
    Update $\Lambda^S \leftarrow \Lambda^S + \nabla_{\Lambda^S} L_{\text{total}}(X^S)$
    Remove x_i^ss for corresponding $\lambda_i^s < 0$
until X^S converges
return Set of DSV : X^S

https://doi.org/10.52202/079017-3752

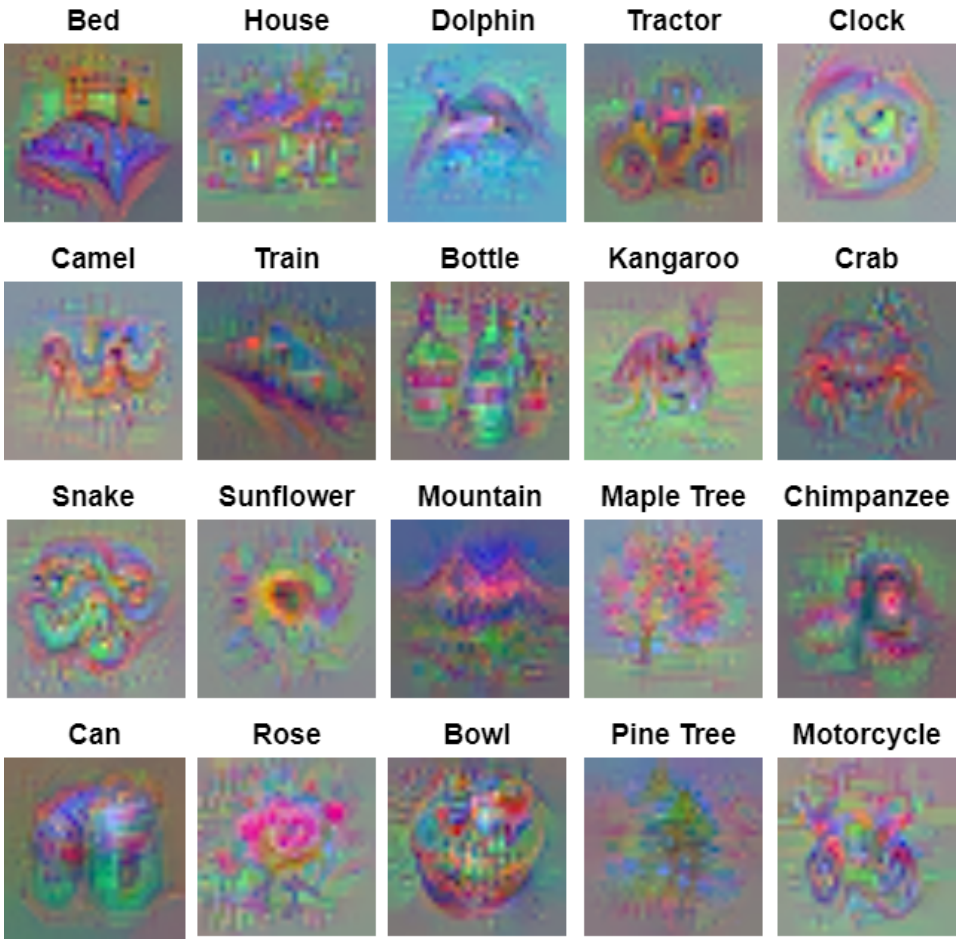

Figure 12: More examples of generated images with CIFAR-100 dataset

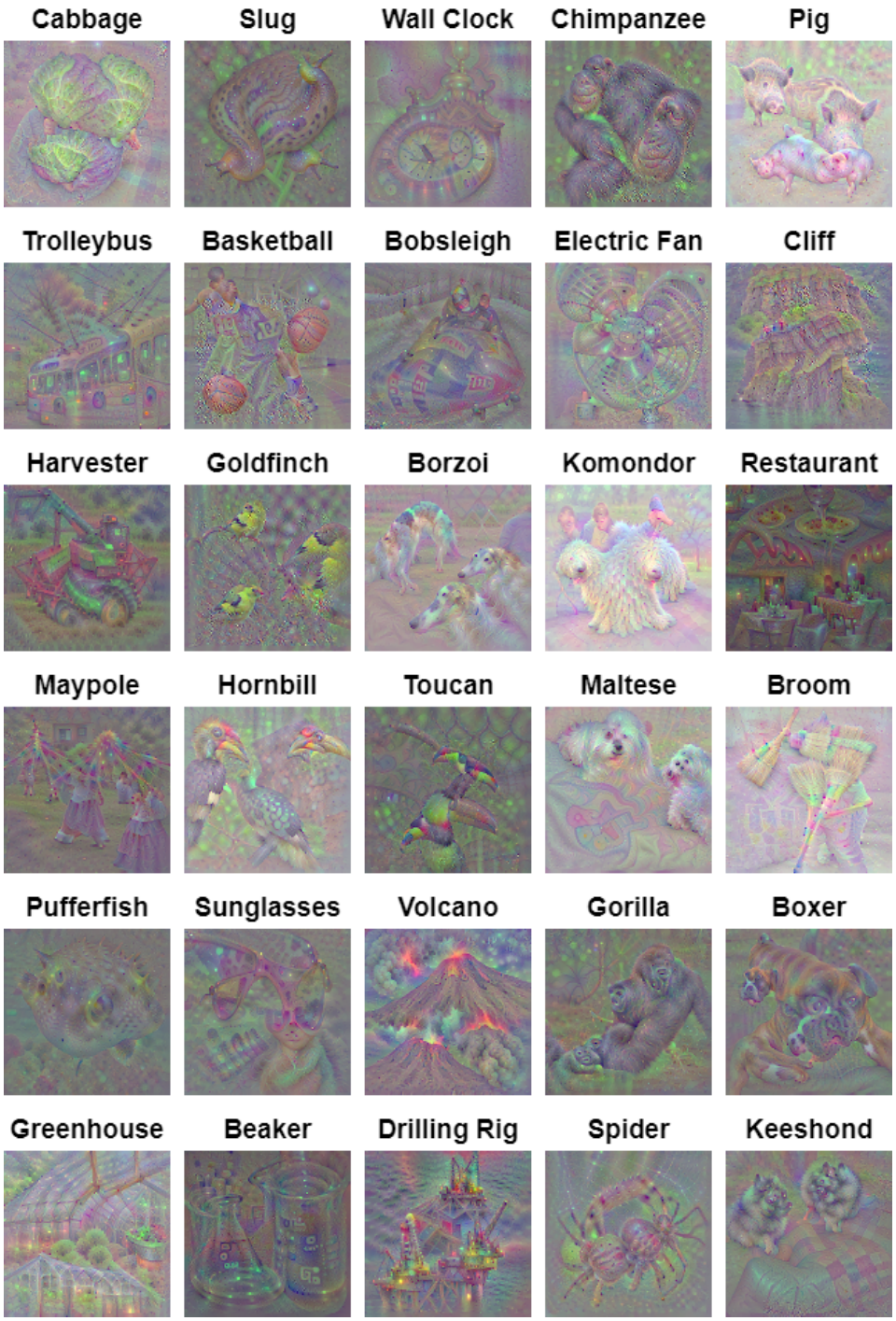

Figure 13: More examples of generated images with ImageNet dataset

https://doi.org/10.52202/079017-3752

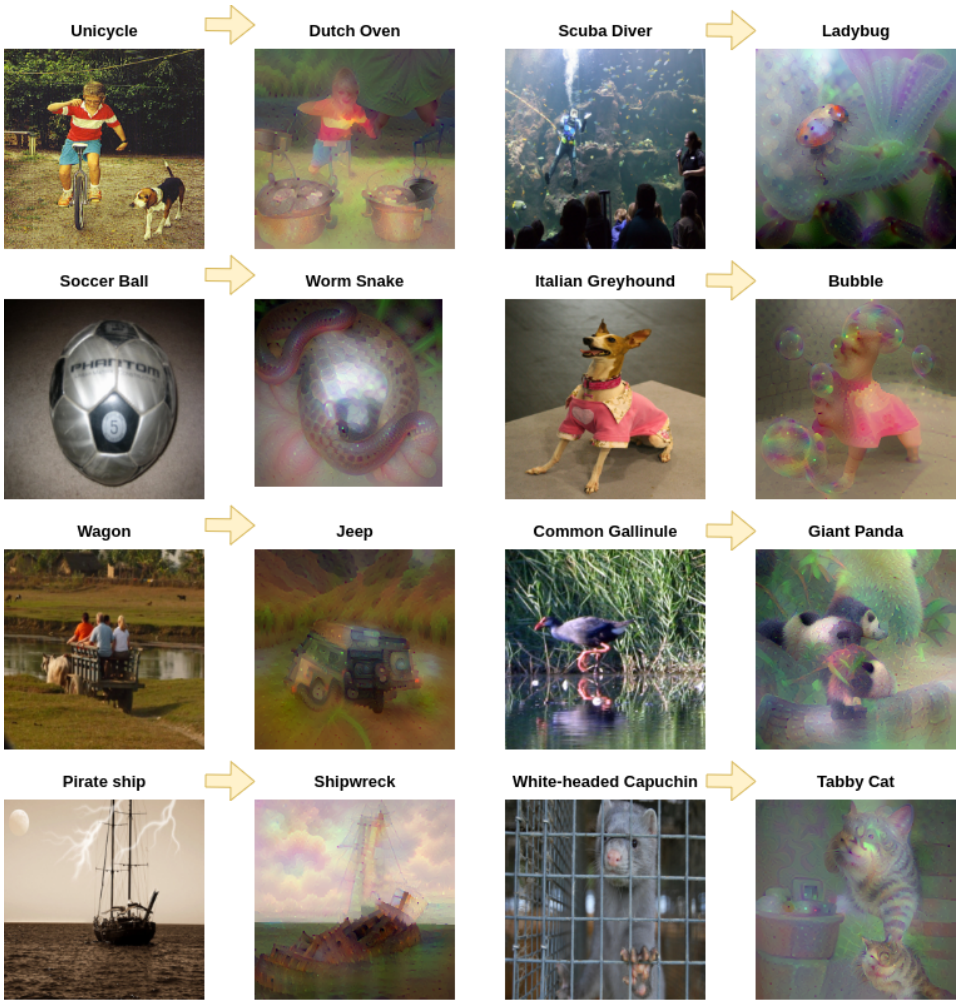

Figure 14: More examples of image editing. The images to the left of the arrows represent the initial images before training, while those to the right depict the edited images after training.

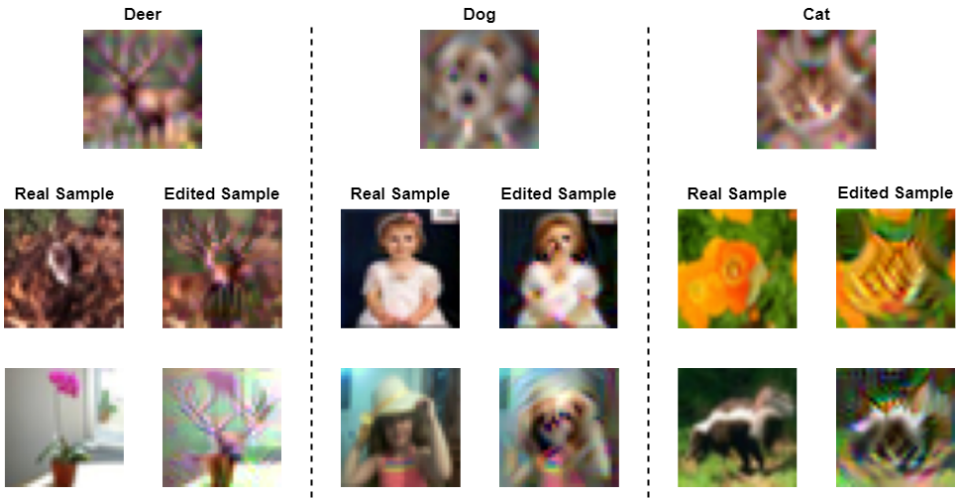

Figure 15: More examples of image editing. The images to the left of the arrows represent the initial images before training, while those to the right depict the edited images after training.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

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 provide a clear overview of the paper's primary contributions, particularly highlighting the introduction of the DeepKKT condition and the concept of Deep Support Vectors (DSVs). These contributions are accurately reflected in the body of the paper through theoretical formulations, experimental evidence, and practical applications of DSVs, aligning with the claims of the paper.

   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. A No or NA answer to this question will not be perceived well by the reviewers.
   - 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**

https://doi.org/10.52202/079017-3752

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

Answer: [Yes]

Justification: See limitation section in Sec. B The limitations are presented in Section \ref{sec:limitations}. The paper notes that the DeepKKT condition's equivalence to traditional support vector representations remains intuitive rather than rigorously proven. It also acknowledges the computational challenges of enforcing all KKT conditions explicitly, especially in high-dimensional deep learning contexts.

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 any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

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

Answer: [Yes]

Justification: The theoretical framework is grounded in adapted KKT conditions, with assumptions explicitly stated for applying these in high-dimensional, multi-class deep learning settings. Although some proofs are briefly outlined, the core theoretical justifications are complete, with further details in the appendix for additional clarification.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- 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, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

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 (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides comprehensive details on the model architectures, datasets, augmentation strategies, and hyperparameters used in the experiments. Key implementation choices, such as optimizer configurations and specific data augmentation techniques, are described in the supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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: We submitted code in supplementary.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

https://doi.org/10.52202/079017-3752

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setup includes clear specifications on data splits, selected hyperparameters, and optimizers for each experimental task. Additional settings, such as augmentation parameters and model architecture details, are included, enabling a comprehensive understanding of the experimental environment.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in 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: [Yes]

   Justification: The paper includes tables with error bars representing the variability of DSV performance on benchmarks. The error bars are correctly computed, taking into account variations across training runs, which supports the robustness of the claims made.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).

- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper specifies the use of GPUs for all major experiments and provides approximate training times. Resources are sufficiently detailed to allow for replication, indicating required compute types and time estimates for reproducibility.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: he work aligns with the NeurIPS Code of Ethics, emphasizing privacy and responsible AI principles, especially concerning the responsible use of DeepKKT for dataset distillation and model inversion, without accessing sensitive data.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

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 paper discusses societal impacts, especially the benefits of interpretability in AI models and potential concerns about model inversion's misuse. It acknowledges the ethical considerations associated with generating data from sensitive models, encouraging responsible handling.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate

https://doi.org/10.52202/079017-3752

to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

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 (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not release high-risk models or data that require specific safeguards.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper appropriately references all datasets (e.g., CIFAR-10, ImageNet) and model architectures, following proper citation practices.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

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

    Answer: [NA]

    Justification: The paper does not introduce any new assets, such as unique datasets or models, that require documentation.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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 (if any)?

    Answer: [NA]

    Justification: The paper does not involve human subjects or crowdsourcing experiments.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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 (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve human subjects research, so IRB approval is not applicable.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-3752