Towards weight-space interpretation of Low-Rank Adapters for Diffusion Models

Jacek Duszenko[1] and Piotr Bielak[1]

Department of Artificial Intelligence,
Wrocław University of Science and Technology
jacek.duszenko@gmail.com, piotr.bielak@pwr.edu.pl

Abstract. Low-rank adapters (LoRAs) have emerged as an efficient method for customizing large-scale diffusion models, but their internal representations remain poorly understood. We present a comprehensive investigation of the interpretability of adapter weight-spaces for image diffusion models. To that end, we open-source a dataset of 100,000 Stable Diffusion adapters fine-tuned across a hierarchy of image concepts amounting to 264 leaf classes, complete with training metadata. Through systematic analysis, we demonstrate that adapter weights encode meaningful semantic information about their training data, enabling direct interpretation without image generation. We evaluate multiple weight-space representations, including raw parameters, statistical summaries, and learned embeddings, to determine their effectiveness in predicting training data characteristics. To demonstrate real-world impact, we apply our findings to the critical task of detecting potentially harmful content on newly introduced NSFW (Not Safe For Work) toy dataset of Stable Diffusion LoRAs fine-tuned on harmful content. This work advances the interpretability of adapter-based fine-tuning and provides practical tools for understanding and auditing adapted diffusion models.

Keywords: Weight-space models · Metanetworks · Low-rank adapters · Image diffusion · Interpretability

1 Introduction

Text-to-image diffusion models, particularly Stable Diffusion [1], have revolutionized the field of AI-generated imagery by enabling widespread access to high-quality image synthesis. Although these models offer impressive general-purpose capabilities, many applications require customized image generation aligned with specific artistic styles, concepts, or domain requirements. This customization typically requires fine-tuning the model on specialized datasets.

However, full model fine-tuning presents significant computational challenges, often requiring extensive GPU resources and training time. Low-Rank Adaptation (LoRA) [2] has emerged as an efficient alternative, allowing for a lightweight model adaptation through training small adapter modules while keeping the base model frozen. This approach has led to a proliferation of publicly shared LoRA adaptations across various communities.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

A critical challenge has emerged with the widespread adoption of these adapter modules: the lack of transparency regarding their encoded content and behaviors. Users who incorporate third-party LoRA modules into their generation pipeline have no reliable way to verify the nature of the adaptations without trial-and-error image generation. This raises concerns about potential inappropriate, biased, or harmful content being inadvertently introduced into the generation process.

Current approaches to understanding LoRA adaptations rely primarily on empirical testing through image generation, which is both time-consuming and computationally expensive. Moreover, this black-box testing approach may fail to reveal the full scope of encoded behaviors. There is a clear need for methods to analyze and interpret LoRA adapters directly without requiring the execution of the full diffusion model.

In this work, we experiment with various weight-space representations and predictive models to interpret LoRA adapters. Our approach enables direct examination of adapter weights eliminating the need for image generation or model execution, which are both resource and time intensive. This methodology has immediate practical applications for popular model repositories such as Civit.ai [3] and Hugging Face [4], where thousands of community-contributed LoRA adapters are shared. Such platforms could leverage the weight-space approach to automatically categorize and tag adapters based on their encoded content, significantly improving content organization and enabling more effective content moderation - all without the computational overhead of running the adapters themselves.

We summarize our contributions as follows:

1. We prepare and open-source the largest dataset of LoRA adapters for Stable Diffusion fine-tuned across a variety of image categories. We publish multiple versions differing in size (number of LoRA adapters), i.e. 1K, 10K, 50K and 100K adapters. Our dataset includes not only the final adapter weights, but also complete training metadata.
2. We provide an experimental evaluation of various weight-space representations and report their performance in classification of the fine-tuning set class.
3. Additionally, we contribute a small real-world use case dataset for predicting whether LoRA adapters were trained on harmful content and empirically prove that it's possible to do solely by looking at adapter's weights.
4. We make our models and code publicly available, ensuring reproducibility of our experiments, to accelerate weight-space research for LoRA adapters.

2 Related Work

Image synthesis. Image synthesis has advanced significantly with Generative Adversarial Networks (GANs) [5], which enabled realistic image generation through a generator-discriminator framework. Diffusion models [6,7] offered more stable training and better image quality by iteratively denoising images, though

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

at the cost of slower inference. CLIP [8] enabled text-conditioned generation in both GANs [9,10,11] and diffusion models [12,13,14]. Stable Diffusion [1] improved efficiency by operating in latent space, reducing computational costs while preserving quality, and became widely adopted due to its open-source nature, spurring innovations such as parameter-efficient fine-tuning.

Model adaptation. While fine-tuning entire models has been the traditional approach to customization, parameter-efficient adaptation methods have recently gained prominence. Low-Rank Adaptation (LoRA) [2] has emerged as a particularly effective approach. For a given weight matrix $\mathbf{W} \in \mathbb{R}^{d\times k}$, LoRA introduces a low-rank decomposition: $\Delta\mathbf{W} = \mathbf{BA}$ where $\mathbf{B} \in \mathbb{R}^{d\times r}$ and $\mathbf{A} \in \mathbb{R}^{r\times k}$ with rank $r \ll \min(d, k)$. The final weight matrix becomes: $\mathbf{W}' = \mathbf{W} + \alpha\Delta\mathbf{W}$ where α is a scaling factor. This decomposition typically reduces trainable parameters by several orders of magnitude. Variations include AdaLoRA [15], which dynamically adjusts rank during training, and adapter layers [16], which insert trainable modules between existing layers. In diffusion models, LoRA has become the standard for community adaptations due to its efficiency and ease of implementation. To prevent overfitting of the original base pipeline on the training set, the standard practice is to combine LoRA adaptation with DreamBooth [17] due to its ability to preserve prior class features while enabling personalized subject adaptation with a small dataset. We employed DreamBooth while creating the dataset.

Weight space models for LoRA adapters. Several recent works have focused on using LoRA weights for various tasks. Salama et al. (2024) [18] propose a task of predicting the number of training samples used to fine-tune a model based on LoRA weights. Their method exploits the relationship between the singular value spectrum of LoRA matrices and dataset size and leverages a simple K-NN approach for training set size prediction. They evaluate their approach on a dataset of 2 thousand fine-tuned LoRA adapters with snapshots of their weights amounting to 25 thousand samples. In contrast, our dataset doesn't use multiple in-training snapshots of a single adapter's weights as samples.

Putterman et al. (2024) [19] draw on geometric deep learning, focusing on designing GL-invariant and equivariant architectures for predicting fine-tuned model properties, e.g. accuracy on downstream tasks, training data membership and training data attributes. Their models are trained on a curated less-noisy subset of CelebA[20] dataset to predict physical binary attributes of the person's face diffusion model was personalized to. In contrast, our dataset comprises of diverse hierarchical concepts not constrained to physical attributes of a human face. Their dataset comprises of about 8 thousand adapters' weights in total.

Dravid et al. (2024) [21] focus on representing adapters in the principal component space of rank-one LoRA weights, enabling linear edits for visual concept transfers, sampling new adapters and inverting images into the weight space. Their dataset has roughly 60 thousand samples.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

Horwitz et al. (2024) [22] propose a representation learning approach by propagating learnable vectors through a frozen dense matrix and subsequently projecting and aggregating the outputs. Although they demonstrate the efficacy of their method on a classification task analogous to our experimental setup, their empirical validation is limited to 5,000 LoRA adapters. Their methodology, although theoretically extensible, relies on selecting a single LoRA adapter weight matrix to derive model-wide representations, a choice that is determined through hyperparameter optimization on a validation set of 500 samples. Furthermore, their supervised learning framework, which jointly optimizes the representation space and classification head, introduces potential limitations in scenarios where labeled data is scarce, a common constraint in weight-space modeling.

3 Proposed dataset

We present a comprehensive dataset of fine-tuned LoRA adapters along with their associated training metadata. The code used in the dataset creation process is available at `https://github.com/JacekDuszenko/weightspace-lora-for-diffusion`

Source images. The dataset is constructed using a carefully curated subset of ImageNet [23], leveraging its hierarchical structure of visual concepts. We selected 10 distinct hierarchies from ImageNet, encompassing general concepts such as dog, cat, airplane, car, fruit, and vegetable. The leaf nodes within each hierarchy correspond to specific instances of these concepts, such as particular dog or cat breeds.

Sample size and LoRA hyperparameters. For each leaf node, we sample between 4 and 15 images without replacement to create training sets for LoRA adaptation, resulting in inputs for 200,000 distinct LoRA adapters. These source images are made available as a separate dataset which we open source at `https://hf.co/datasets/jacekduszenko/weightspace-images`. Using the well known `Stable-Diffusion-v1-5` as our base model, we perform DreamBooth fine-tuning without prior preservation loss to create the LoRA adapters. Each adapter is trained using consistent hyperparameters: $rank = \alpha = 1$ and a learning rate of $\mathrm{lr} = 1\mathrm{e}{-4}$, with training conducted for 200 iterations.

Fine-tuning. The training process employs a standardized textual prompt format: `a photo of sks CLASS`, where `CLASS` represents one of the ten general categories, and `sks` serves as a unique identifier for each specific concept, following standard DreamBooth methodology. The adapters are configured to modify the query (W_q) and value (W_v) projections within the attention layers of the Stable Diffusion U-Net. For each adapter, we store both the A and B LoRA matrices for each layer. Given the 64 layers in the architecture, this results in

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

128 LoRA matrices per adapter, with matrices represented as $320,640,768$ or 1024-dimensional vectors due to the rank-1 configuration.

The dataset maintains balanced representation across categories, with adapters trained on each general category comprising 10% of the total dataset. These trained adapters represent 264 unique concepts corresponding to the leaf classes from which the training images were sampled. For each adapter, we preserve comprehensive metadata including epoch-wise loss values, adapter gradient norms, weight norms, and training duration.

Hardware resources. The training infrastructure utilized 8 NVIDIA A100 GPUs in a distributed setting, requiring a total of 2096 GPU hours to complete.

Dataset versions. To facilitate various research applications, we release four versions of the dataset at different scales, with uniform distribution across general class labels. These datasets, including LoRA weights and training metadata, are available at the following URLs:

- `LoRA-WS-1k` (`https://hf.co/datasets/jacekduszenko/lora-ws-1k`),
- `LoRA-WS-10k` (`https://hf.co/datasets/jacekduszenko/lora-ws-10k`),
- `LoRA-WS-50k` (`https://hf.co/datasets/jacekduszenko/lora-ws-50k`),
- `LoRA-WS-100k` (`https://hf.co/datasets/jacekduszenko/lora-ws-100k`).

NSFW classification toy dataset. Additionally, we develop a small Not Safe For Work detection dataset comprising LoRA adapters fine-tuned on two distinct classes: not safe for work content of one of the class from [24] and neutral concepts from our ImageNet subset. With the exception of fixed training set size of 5 images, adapters are trained using procedures and parameters identical to those of the main dataset, with complete preservation of the training data and the weights of the trained LoRAs. This supplementary dataset consists of 160 samples of adapters distributed evenly across the binary label and is available at `https://hf.co/datasets/jacekduszenko/lora-ws-nsfw`.

4 Weight-space representations

Traditional representation learning methods build representation (embedding) vectors for various data types such as images, audio, text or graphs [25], often utilizing neural networks combined with a carefully crafted objective function, to convert the input objects into vector form. In contrast, weight-space representation learning aims at building representation vectors for neural networks. Their weights are processed either directly, by means of simple statistical operations [26], or by other neural networks [27].

In this paper, we focus on direct vector processing and statistical operations, leaving more advanced neural network approaches for future work. In particular, we examine the following approaches:

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

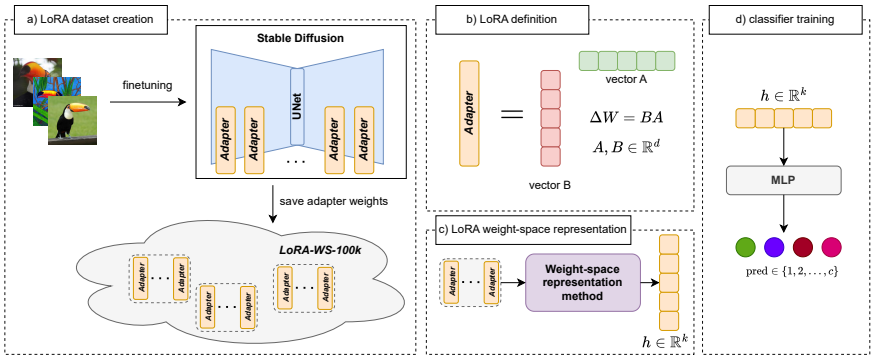

Fig. 1. Overview of our setup. **a)** The LoRA weight-space datasets were created by Stable Diffusion finetuning using low-rank adapters. For each sample of images, we finetune Stable Diffusion and save all the LoRAs from the UNet component. **b)** A single low-rank adapter consists of two vectors A and B (we assume rank $= 1$, so instead of matrices, we obtain vectors). **c)** The weight-space representation learning methods take as input a collection of LoRAs (from a single Stable Diffusion finetuning instance) and compute a single vector representation h which captures the characteristics of the adapters. **d)** To evaluate our hypothesis that LoRA weights encode sufficient information to retrieve the class of the input finetuning images, we use the weight-space representations to train an MLP classifier and report the classification performance on the test set.

- `FlatVec` – the most popular baseline approach in weight-space representation learning; for each adapter layer, the weight matrices are flattened into vectors, next all the vectors are concatenated into one long vector; in our case the final vector dimensionality is $dim(\texttt{FlatVec}) = 99648$,
- `FlatVecPCA@K` – the high dimensionality of the `FlatVec` approach is unfeasible in most scenarios, therefore, we reduce the dimensionality of `FlatVec`'s output vector by using Principal Component Analysis and keep K components, so that $dim(\texttt{FlatVecPCA@K}) = K$,
- `StatsFlatVec` – another popular weight-space representation approach is to compute basic statistics of the model weights; previous works [26,27] have shown that such representation provides quite competitive results, even against complex neural network approaches; in our case, we take the `FlatVec`'s output vector and compute the following statistics for it: for the `LoRA-WS-1k` and `LoRa-WS-10k` datasets – mean, std, median, min, max, kurtosis, skew, which results in $dim(\texttt{StatsFlatVec}) = 7$, and for `LoRA-WS-50k` – the same statistics except for kurtosis and skew (due to Out-Of-Memory errors during evaluation), which results in $dim(\texttt{StatsFlatVec}) = 5$,
- `StatsLayer` – given that each adapter layers consists of two matrices $\mathbf{A}$ and $\mathbf{B}$, we compute the statistics (same as for `StatsFlatVec`) but for each matrix and adapter layer independently, and finally we concatenate all the vectors, which results in $dim(\texttt{StatsLayer}) = 896$,

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

– `StatsLayerDense` – we first compute the dense matrix of each adapter layer, i.e., $\Delta\mathbf{W} = \mathbf{BA}$, flatten this matrix and compute the same statistics as before for each adapter layer, finally we concatenate all the vectors; this results in $dim(\texttt{StatsLayerDense}) = 448$.

5 Experiments

We evaluate the performance of the introduced representation methods against our proposed dataset. We perform several experiments to better understand the nature of low-rank adapters and provide insights based on the observed results. Each experiment is described in a separate subsection below.

5.1 Fine-tuning dataset classification

Goal. We investigate whether adapter weights contain sufficient information to predict the class of images on which an adapter was fine-tuned. Specifically, we aim to classify LoRA adapters into one of 10 base classes solely by examining their weight representations.

Setup. For this task, we utilize the five distinct weight representations described in Section 4. We partition our dataset using a 70/10/20 split for training, validation, and testing respectively. The classifier architecture consists of a multi-layer perceptron with three hidden layers of size 512 with ReLU activations, trained using cross-entropy loss. We employ the Adam optimizer [28] with a learning rate of $lr = 1\text{e}{-3}$ and train for a maximum of 2000 epochs with early stopping to prevent overfitting. To ensure robust evaluation, we repeat each experiment 10 times with different random seeds and report the mean performance metrics with standard deviations. Furthermore, we conduct separate experiments across multiple dataset scales (1k, 10k, and 50k) to assess how the effectiveness of different representation methods scales with data volume. This systematic approach enables us to identify which weight representation techniques most effectively capture the discriminative information necessary for determining the original fine-tuning class. We report the results in Table 1.

Discussion. For the smallest dataset (`LoRA-WS-1k`), we observe that for all reported metrics, the `FlatVecPCA@200` method clearly outperforms the other approaches by a large margin. It allows achieving up to approx. 86% AUROC, whereas the next best method (`StatsLayerDense`) achieves approx. 81% AUROC. Unsurprisingly, the `FlatVec` method performs poorly – having vectors with almost 100,000 dimensions, while having only 700 training samples makes it hard for any kind of classifier to generalize well. Similarly, the 700 samples do not provide enough information for the `StatsFlatVec` method, which reduces the almost 100,000 numbers into 7 statistics.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

Table 1. Classification metrics using different representation learning methods. Results are reported as mean ± standard deviation over 10 runs (different seeds). Higher values are better (↑). Best results are **bolded**, whereas second best results are underlined. (a) Results on `LoRA-WS-1k` dataset (top), (b) results on `LoRA-WS-10k` dataset (middle), (c) results on `LoRA-WS-50k` dataset (bottom).

a) `LoRA-WS-1k`	Accuracy (↑)	AUROC (↑)	Precision (↑)	Recall (↑)	F1 (↑)
`FlatVec`	15.62 ± 2.17	55.64 ± 1.94	16.95 ± 3.42	15.50 ± 2.14	14.91 ± 2.21
`FlatVecPCA@200`	$\mathbf{42.21 \pm 1.20}$	$\mathbf{86.12 \pm 1.01}$	$\mathbf{41.54 \pm 1.55}$	$\mathbf{42.06 \pm 1.19}$	$\mathbf{40.69 \pm 1.05}$
`StatsFlatVec`	16.22 ± 2.89	55.15 ± 2.26	17.69 ± 3.74	15.94 ± 2.79	15.60 ± 3.04
`StatsLayer`	33.18 ± 2.10	77.27 ± 0.49	32.83 ± 3.18	33.55 ± 2.20	32.74 ± 2.51
`StatsLayerDense`	37.51 ± 2.07	81.62 ± 0.61	36.71 ± 1.85	37.30 ± 2.07	36.41 ± 1.93

b) `LoRA-WS-10k`	Accuracy (↑)	AUROC (↑)	Precision (↑)	Recall (↑)	F1 (↑)
`FlatVec`	65.83 ± 1.39	92.83 ± 0.51	66.48 ± 1.46	65.81 ± 1.39	65.84 ± 1.42
`FlatVecPCA@200`	$\mathbf{81.36 \pm 1.07}$	$\mathbf{98.14 \pm 0.16}$	$\mathbf{81.63 \pm 1.04}$	$\mathbf{81.33 \pm 1.07}$	$\mathbf{81.41 \pm 1.04}$
`StatsFlatVec`	64.75 ± 1.66	92.32 ± 0.64	65.44 ± 1.98	64.73 ± 1.68	64.81 ± 1.82
`StatsLayer`	44.22 ± 1.05	83.50 ± 0.37	44.34 ± 1.14	44.01 ± 1.04	44.07 ± 1.08
`StatsLayerDense`	52.61 ± 0.63	88.22 ± 0.22	52.43 ± 0.64	52.49 ± 0.63	52.36 ± 0.63

c) `LoRA-WS-50k`	Accuracy (↑)	AUROC (↑)	Precision (↑)	Recall (↑)	F1 (↑)
`FlatVec`	$\mathbf{93.26 \pm 2.56}$	$\mathbf{99.63 \pm 0.26}$	$\mathbf{93.28 \pm 2.51}$	$\mathbf{93.26 \pm 2.55}$	$\mathbf{93.26 \pm 2.54}$
`FlatVecPCA@1000`	78.33 ± 1.37	97.61 ± 0.20	78.47 ± 1.44	78.30 ± 1.37	78.33 ± 1.39
`StatsFlatVec`	93.21 ± 2.17	99.63 ± 0.18	93.20 ± 2.15	93.20 ± 2.17	93.19 ± 2.17
`StatsLayer`	63.68 ± 0.38	93.01 ± 0.17	63.57 ± 0.32	63.58 ± 0.38	63.56 ± 0.35
`StatsLayerDense`	44.49 ± 0.47	84.09 ± 0.26	45.14 ± 0.41	44.44 ± 0.47	44.72 ± 0.44

For the `LoRA-WS-10k` dataset, we find that the overall metrics have increased substantially compared to the 1k variant, for instance, now the maximum AUROC value is approx. 98% compared to 86% on the 1k dataset; similarly, the Accuracy and F1 metrics have nearly doubled. This confirms our motivation for creating larger datasets for weight-space representation learning on low-rank adapters. Once again, the `FlatVecPCA@200` method outperforms other representation methods; however, this time the second best method is `FlatVec`. Despite having almost 100,000 dimensions, we now have much more samples to fit a classifier and allow it generalize better. Note also that the `StatsFlatVec` method achieves performs quite similarly to the `FlatVec` method, while having significantly less dimensional representations.

In case of the `LoRA-WS-50k` dataset, the `FlatVec` method obtains the best results for all metrics; however, the `StatsFlatVec` method performs only slightly worse on all but one metric – for AUROC the mean values are the same with `StatsFlatVec` having a smaller standard deviation. The data volume (50k samples) allows to achieve even better results than on the 10k dataset variant.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

`StatsFlatVec` being the second best method shows again that despite more samples, the dataset can be efficiently compressed down to only 5 dimensional vectors.

To sum up, we observe that the `FlatVec` and `FlatVecPCA@K` methods work the best overall. For the 50k dataset variant, the `StatsFlatVec` method also works great despite compressing the representations to only 5 dimensions. However, for practical applications, these methods are not suitable. They either use very high dimensional vectors ($dim(\texttt{FlatVec}) \approx 100,000$) or they use such vectors as intermediate steps (`FlatVecPCA@K` and `StatsFlatVec`), which requires large computational resources. For future work, we would like to focus on representation methods that operate on lower dimensional vectors, allowing for resource efficient data processing.

5.2 Expressiveness of individual layers

Goal. To systematically analyze the information content encoded within individual LoRA adapter layers, we isolate and evaluate the predictive power of each adapter layer independently on the `LoRA-WS-1k` dataset.

Setup. For each LoRA adapter layer l, we extract and flatten its weight matrices adapting $W_q^{(l)}$ and $W_v^{(l)}$ as well as self-attention maps in various attention layers of the U-Net into feature vectors $\mathbf{x}_l \in \mathbb{R}^d$, where d varies based on the layer dimensions. We partition the dataset into training (70%), validation (10%), and test (20%) sets. An MLP classifier is trained on the training set and evaluated on both validation and test sets. The MLP architecture remains consistent with the previous experiment, featuring three hidden 512-dimensional layers with ReLU activations. We employ early stopping based on validation performance and use the Adam optimizer with a learning rate of $\text{lr} = 1\text{e}{-3}$. We repeat this process 10 times with different random seeds. We report averaged results across all runs in Figure 2.

Discussion. We empirically demonstrate that LoRA adapter weights in cross-attention layers encode substantial information about the fine-tuning dataset, with the value projection adapter (W_v) achieving the highest classification accuracy (97.01% on the test set), slightly outperforming the query projection (W_q). This suggests that the learned adaptations in W_v retain stronger dataset-specific signals, potentially due to their direct role in modulating text-conditioned feature integration. In contrast, self-attention layers exhibit significantly lower predictive power, with most performing near chance level (10%) and only a few reaching moderate expressiveness (60%). This disparity highlights the dominant role of cross-attention in shaping model-specific adaptations, while self-attention layers primarily facilitate general feature propagation rather than encoding dataset-specific signatures.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

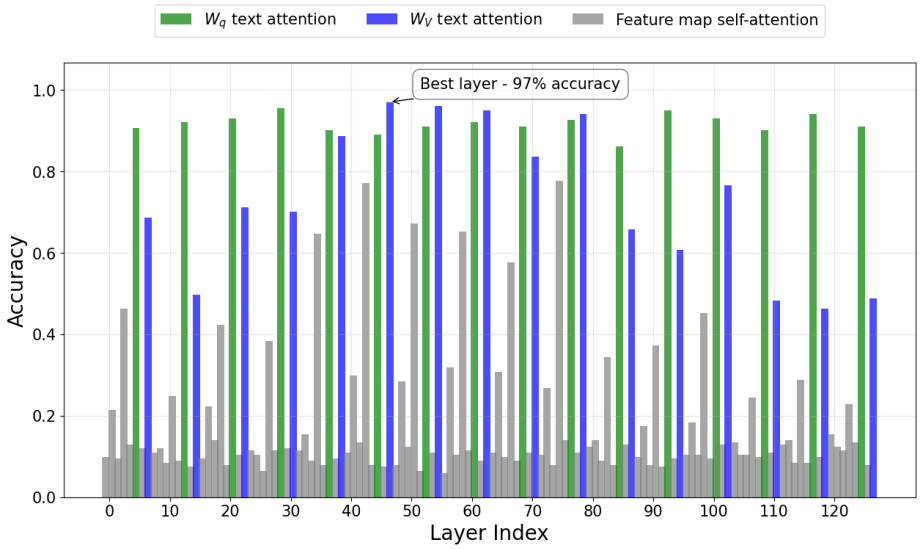

Fig. 2. Classification accuracy across different LoRA adapter layers in the U-Net architecture. The plot demonstrates the predictive power of individual adapter weights for dataset class identification. Notable peaks in accuracy correspond to text conditioning cross-attention layers, particularly in the query (W_q) and value (W_v) projections, achieving up to 97.01% test accuracy. Other attention layers such as self attention of the latent feature maps of the U-net are in gray.

5.3 Dimensionality reduction

Goal. To elucidate the relationship between representational dimensionality and feature quality, we conduct Principal Component Analysis (PCA) on high-dimensional feature vectors constructed by the `FlatVec` method (note that this setup is essentially the same as the `FlatVecPCA@K` method with different choices of the K value).

Setup. Figure 3 illustrates the relationship between classification accuracy and the number of retained principal components. We extend this experiment to the larger `LoRA-WS-10k` dataset, maintaining the same framework as in our previous analysis, with the key modification being the use of concatenated adapter weights as input features rather than individual layer representations. We try different PCA values (100, 200, 500, 1000, 2000, 3000, 5000 and 7500) and report the mean accuracy achieved in ten runs of the experiment with different random seeds.

Discussion. We observe that the performance on both validation and test splits is quite similar with Validation Accuracy being only a bit worse than the Test Accuracy. Moreover, both metrics highly depend on the choice of the K parameter,

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

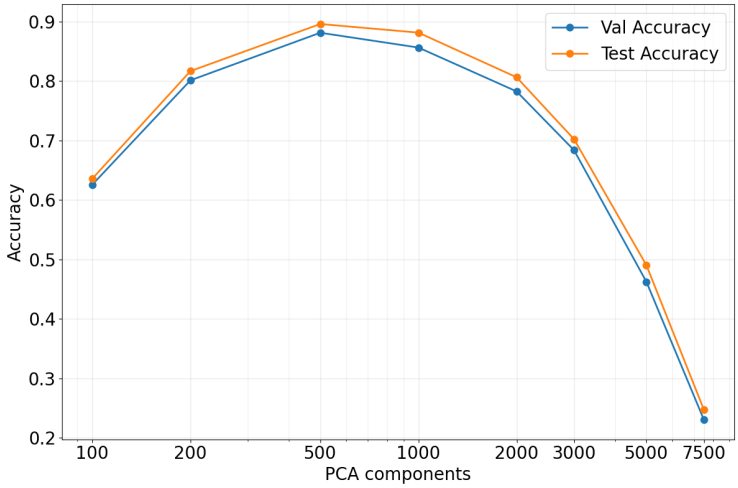

Fig. 3. Classification accuracy as a function of retained PCA components for different LoRA adapter layers. The plot demonstrates how predictive power varies with dimensionality reduction.

ranging from almost 20% for $K = 7500$ up to approx. 85% for $K = 500$ (which is the sweet spot for this dataset variant). Using both 200-dim and 1000-dim vectors also provides a decent performance, however, we would recommend going with $K = 200$ as it requires less computational power (due to lower dimensional vectors). Overall, we observe that the data points approximate a convex function, where larger dimensional vectors achieve worse performance that smaller ones. Hence, when using the `FlatVecPCA@K` method, we would recommend using lower dimensional vectors, i.e., smaller values of K and check a few values around $K = 500$.

5.4 Harmful content detection

Goal. We want to address a crucial application area of weight-space representation learning for low-rank adapters – detection of potentially harmful LoRAs. Hence, in this experiment we predict whether an adapter was fine-tuned on images depicting harmful content represented as one of the classes from [24].

Setup. We take sexually explicit anime/manga images as concepts to train harmful adapters. As showed empirically in previous experiments, simple classifiers are not doing very well on small datasets. To tackle this problem, we feed a single layer as input to the classifier and choose the best performing layer in evaluation. We use a 80/20% dataset split. We pre-process the weights of adapters to obtain representations described earlier and conduct experiments with each representation. To this end, we train an MLP with three hidden layers of size

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

1024 and final classification head with two dimensional output. We then backpropagate on the cross-entropy loss. In evaluation, we treat the more probable class as final prediction of the model. We use the learning rate of lr = 1e−4, leverage the Adam optimizer and train with early-stopping on 2000 epochs. We present our results in Table 2.

Table 2. Classification metrics for harmful content detection using different representation learning methods. Results are reported as mean ± standard deviation on the test set over 10 runs (different seeds). Higher values are better (↑). Best results are **bolded**, whereas second best results are underlined.

`LoRA-WS-NSFW`	Accuracy (↑)	AUROC (↑)	Precision (↑)	Recall (↑)	F1 (↑)
`FlatVec`	**98.12 ± 1.53**	**99.96 ± 0.12**	**99.41 ± 1.76**	**96.88 ± 3.12**	**98.08 ± 1.57**
`FlatVecPCA@20`	67.01 ± 4.46	62.46 ± 2.58	63.98 ± 3.42	76.88 ± 7.93	69.74 ± 5.00
`StatsFlatVec`	66.87 ± 4.00	61.88 ± 1.39	63.84 ± 2.62	77.50 ± 9.35	69.83 ± 4.95
`StatsLayerDense`	60.34 ± 10.30	60.30 ± 4.41	61.20 ± 4.00	69.01 ± 1.00	65.03 ± 3.38

Discussion. Our results demonstrate that certain adapter weights, primarily the ones concerned with cross-attention to support textual conditioning, contain clear signatures of harmful content training, with flattened vector achieving near-perfect detection accuracy (98.12%). Statistical summaries and dimensionality reduction by PCA perform poorly (66-67%), indicating that feature engineering provides only marginal benefits in the small dataset setting. Our experiments with summary statistics of dense representation yielded poor performance (60.34% accuracy), confirming that joining two layers and aggregating them further degrades classification efficacy. This finding aligns with our previous experimental observations and demonstrates that dense representations have limited expressiveness.

6 Conclusions and future work

In this paper, we addressed the problem of learning representations for low-rank adapters based on their weights (so called weight-space representation learning). We contributed several StableDiffusion LoRA datasets varying in size, i.e., `LoRA-WS-1k`, `LoRA-WS-10k`, `LoRA-WS-50k` and `LoRA-WS-100k`. We performed a variety of experiments to better understand the performance of different weight-space representation learning methods on our dataset. We have shown that low-rank adapters encode sufficient information in their weights to be able to classify the content the adapter was trained on without the need to generate the actual images. Such a setup is crucial when working with potentially harmful adapters, where we do not want to generate the actual images to detect whether it was trained on dangerous content. To that end, we introduced a small dataset

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

LoRA-WS-NSFW and analyzed the performance a several embedding methods. Results show that by just using the weights directly, we can build a well performing classifier.

In future work, we want to extend our experiments to the LoRA-WS-100k dataset. However, to achieve that we need to first focus on developing a dedicated weight-space representation learning method, which does not require operating on almost 100,000-dimensional vectors (as we had to do for the other dataset variants). Working with such large vectors combined with a large dataset size, makes it difficult to train any kind of classification models (due to high resource requirements). Next, we want to also extend our LoRA-WS-NSFW dataset by increasing the dataset size. Finally, we want to introduce noise to the LoRA datasets, which should better reflect real-world conditions and help to test the generalization ability of the weight-space representations. We want to also check scenarios where labels are imperfect and/or classes are imbalanced, which are also likely to be found in real-world cases.

Acknowledgments. We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Center: ACK Cyfronet AGH) for providing computer facilities and support within computational grant no. PLG/2025/018054.

References

1. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-Resolution Image Synthesis with Latent Diffusion Models. arXiv preprint arXiv:2112.10752 (2022)
2. Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, L., Chen, W.: LoRA: Low-Rank Adaptation of Large Language Models. International Conference on Learning Representations (2022) https://openreview.net/forum?id=nZeVKeeFYf9
3. CivitAI: A community-driven platform for sharing AI-generated models and content, https://civitai.com, last accessed 2025/02/18
4. Hugging Face: An open platform for machine learning models and datasets, https://huggingface.co, last accessed 2025/02/18
5. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Nets. Advances in Neural Information Processing Systems (NeurIPS) 27, 2672-2680 (2014). arXiv preprint arXiv:1406.2661 (2014)
6. Ho, J., Jain, A., Abbeel, P.: Denoising Diffusion Probabilistic Models. Advances in Neural Information Processing Systems (NeurIPS) 33, 6840-6851 (2020). arXiv preprint arXiv:2006.11239 (2020)
7. Dhariwal, P., Nichol, A.: Diffusion Models Beat GANs on Image Synthesis. Advances in Neural Information Processing Systems (NeurIPS) 34, 8780-8794 (2021). arXiv preprint arXiv:2105.05233 (2021)
8. Radford, A., Kim, J., Chen, C., Hessel, M., Narasimhan, K., Tan, X., Wang, W., Yu, Z., Sutskever, I.: Learning Transferable Visual Models From Natural Language Supervision. Proceedings of the 38th International Conference on Machine Learning (ICML), 8748-8763 (2021). arXiv preprint arXiv:2103.00020 (2021)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

9. Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., Lischinski, D.: StyleCLIP: Text-driven Manipulation of StyleGAN Imagery. arXiv preprint arXiv:2103.17249 (2021)
10. Gal, R., Patashnik, O., Maron, H., Chechik, G., Cohen-Or, D.: StyleGAN-NADA: CLIP-Guided Domain Adaptation of Image Generators. arXiv preprint arXiv:2108.00946 (2021)
11. Galatolo, F. A., Cimino, M. G. C. A., Vaglini, G.: Generating Images from Caption and Vice Versa via CLIP-Guided Generative Latent Space Search. arXiv preprint arXiv:2102.01645 (2021)
12. Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., ... & Sutskever, I.: GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. arXiv preprint arXiv:2112.10741 (2021)
13. Kim, G., Ye, J. C.: DiffusionCLIP: Text-Guided Image Manipulation Using Diffusion Models. arXiv preprint arXiv:2110.02711 (2021)
14. Avrahami, O., Lischinski, D., Fried, O.: Blended Diffusion for Text-Driven Editing of Natural Images. arXiv preprint arXiv:2111.14818 (2021)
15. Liu, Z., Lin, J., Xu, B., Li, J., Yang, S., Chen, Z., Zhou, L.: AdaLoRA: Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning. International Conference on Learning Representations (ICLR) (2023). arXiv preprint arXiv:2302.12345 (2023)
16. Houlsby, N., Giurgiu, L., Stokes, J., Lahiri, S., Biewald, L., Sutskever, I., and others: Parameter-Efficient Transfer Learning for NLP. International Conference on Machine Learning (ICML) 36, 2708-2717 (2019). arXiv preprint arXiv:1902.00751 (2019)
17. Ruiz, N. J., Jain, P., Ramesh, A., Chen, Z., Lin, C., Le, Q. V., and Zhai, A.: DreamBooth: Fine-Tuning Text-to-Image Diffusion Models for Subject-Driven Generation. arXiv preprint arXiv:2303.10745 (2023)
18. Salama, M., Kahana, J., Horwitz, E., Hoshen, Y.: Dataset Size Recovery from LoRA Weights. arXiv preprint arXiv:2406.19395 (2024)
19. Putterman, T., Lim, D., Gelberg, Y., Jegelka, S., Maron, H.: Learning on LoRAs: GL-Equivariant Processing of Low-Rank Weight Spaces for Large Finetuned Models. arXiv preprint arXiv:2410.04207 (2024)
20. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep Learning Face Attributes in the Wild. In: Proceedings of International Conference on Computer Vision (ICCV), pp. 3730–3738, December (2015)
21. Dravid, A., Gandelsman, Y., Wang, K.-C., Abdal, R., Wetzstein, G., Efros, A.A., Aberman, K.: Interpreting the Weight Space of Customized Diffusion Models. arXiv preprint arXiv:2406.09413 (2024)
22. E. Horwitz, B. Cavia, J. Kahana, and Y. Hoshen, "Learning on Model Weights using Tree Experts," arXiv preprint arXiv:2410.13569, 2024.
23. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR*, 2009. Available: `http://www.image-net.org/papers/imagenet_cvpr09.bib`
24. Huggingface nsfw_detect dataset, `https://huggingface.co/datasets/deepghs/nsfw_detect`, last accessed 2025/02/19
25. Bengio, Y., Courville, A., and Vincent, P.: Representation Learning: A Review and New Perspectives. IEEE transactions on pattern analysis and machine intelligence, 35, 1798–1828 (2013).
26. K. Schürholt, D. Taskiran, B. Knyazev, X. Giró-i-Nieto, and D. Borth: Model Zoos: A Dataset of Diverse Populations of Neural Network Models. In: Thirty-Sixth Conference on Neural Information Processing Systems (NeurIPS) Track on Datasets and Benchmarks (2022).

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8

27. Derek Lim, Haggai Maron, Marc T. Law, Jonathan Lorraine, James Lucas Graph Metanetworks for Processing Diverse Neural Architectures. ICLR, 2024.
28. Kingma, D. P., and Ba, J.: Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980 (2017). Available at: https://arxiv.org/abs/1412.6980

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_8