Efficient Fine-Tuning and Concept Suppression for Pruned Diffusion Models

Supplementary Material

A. Details of APTP

Adaptive Prompt-Tailored Pruning (APTP) [12] is a novel prompt-based pruning method designed for Text-to-Image (T2I) diffusion models. T2I diffusion models are computationally intensive, especially during the sampling process, making their deployment on resource-constrained devices or for large user bases challenging. APTP aims to reduce this computational cost by tailoring the model architecture to the complexity of the input text prompt.

Instead of using a single pruned model for all inputs, APTP prunes a pretrained T2I model (e.g., Stable Diffusion) into a mixture of efficient experts, where each expert specializes in generating images for a specific group of prompts with similar complexities. This is illustrated in Figure 1 of their paper.

At the heart of APTP lies a prompt router module. This module learns to determine the required capacity for an input text prompt and routes it to an appropriate expert, given a total desired compute budget. Each expert corresponds to a unique architecture code that defines its structure as a subnetwork of the original T2I model. The number of experts (and corresponding architecture codes) is a hyperparameter.

The prompt router consists of three key components:

1. Prompt Encoder: Encodes input prompts into semantically meaningful embeddings using a pretrained frozen Sentence Transformer model.
2. Architecture Predictor: Transforms the encoded prompt embeddings into architecture embeddings, bridging the gap between prompt semantics and the required architectural configuration.
3. Router Module: Maps the architecture embeddings to specific architecture codes. To prevent all codes from collapsing into a single one, the router module employs optimal transport during the pruning phase. The optimal transport problem aims to find an assignment matrix Q that maximizes the similarity between architecture embeddings and their assigned architecture codes while ensuring equal distribution of prompts to different experts. This optimal assignment matrix is calculated using the Sinkhorn-Knopp algorithm and is used to route architecture embeddings to architecture codes during pruning.

The prompt router and architecture codes are trained jointly in an end-to-end manner using a contrastive learning objective.

B. Method for solving the bilevel problem

Classical methods for solving a bilevel problems such as Eq. (9) require calculating second order information, please see [7, 8, 13] For examples. However, when fine-tuning foundation models, this process becomes extremely expensive due to the high computational and memory demands. Recently, new frameworks for bilevel optimization have been introduced [24, 30, 32, 35, 45]. These methods only use first-order information and thus significantly reduce computational costs, making them extremely suitable for fine-tuning foundation models. We employ this type of method for solving Eq. (9).

More, specifically, Eq. (9) is equivalent to the following constrained optimization problem:

$$\begin{aligned} &\min_{\theta_{pruned}} \mathbb{E}_{x_0,\epsilon,t,c,c'} \|\epsilon_\theta(x_t,t,c') - \epsilon_{\theta_{pruned}}(x_t,t,c)\|^2, \\ &s.t.\ L^{ft}(\theta_{pruned}) - \inf_{\vartheta} L^{ft}(\vartheta) \leq 0. \end{aligned} \tag{15}$$

By penalizing the constraint, we obtain the following penalized problem:

$$\min_{\theta_{pruned}} L_{penalized}(\theta_{pruned}), \tag{16}$$

where

$$\begin{aligned} &L_{penalized}(\theta_{pruned}) := \\ &\mathbb{E}_{x_0,\epsilon,t,c,c'} \|\epsilon_\theta(x_t,t,c') - \epsilon_{\theta_{pruned}}(x_t,t,c)\|^2 \\ &+ \lambda \left(L^{ft}(\theta_{pruned}) - \inf_{\vartheta} L^{ft}(\vartheta) \right) \end{aligned} \tag{17}$$

and $\lambda > 0$. As λ increases, the solution to the penalized problem approaches the solution to Eq. (15), and thus the solution to Eq. (9) (see [35] Theorem 2 for an explicit relationship between the stationary points of Eq. (15) and those of the original problem Eq. (9)). Note that the penalized problem Eq. (11) is equivalent to the following minimax problem:

$$\min_{\theta_{pruned}} \max_{\vartheta} G_\lambda(\theta_{pruned}, \vartheta), \tag{18}$$

where

$$\begin{aligned} &G_\lambda(\theta_{pruned}, \vartheta) := \\ &\mathbb{E}_{x_0,\epsilon,t,c,c'} \|\epsilon_\theta(x_t,t,c') - \epsilon_{\theta_{pruned}}(x_t,t,c)\|^2 \\ &+ \lambda \left(L^{ft}(\theta_{pruned}) - L^{ft}(\vartheta) \right). \end{aligned} \tag{19}$$

To solve Eq. (18), we use a double loop method. At step t, we fix θ^t_{pruned} and then solve the maximization problem $\max_\vartheta G_\lambda(\theta^t_{pruned}, \vartheta)$. Then we update θ_{pruned} using the gradient of $\nabla_{\theta_{pruned}} G_\lambda(\theta^t_{pruned}, \vartheta)$. Since the gradient of G with respect to θ_{pruned} is determined both by the upper

loss and lower loss, this incorporates more information from feature distillation when doing concept unlearning. Therefore, the upper and lower level problems are dependent on each other. This is the key difference between the two-stage method and our bilevel method.

C. Experiments

C.1. Detailed experimental setup

C.1.1. Datasets

In all our experiments, we use the MS-COCO Captions 2017 [29] with approximately 500k training image-caption pairs. For evaluations, we use the validation data of MS-COCO-2017 with 5000 images. We sample one caption per image from the validation set.

C.1.2. Effect of Distillation and Pruning Experimental Setting

We utilize one of the pre-trained APTP [12] experts on COCO, which achieves 80% MAC utilization compared to the original Stable Diffusion 2.1 model [40]. The model is fine-tuned using various objectives at a fixed resolution of 512×512 for all configurations. Optimization is performed with the AdamW [34] optimizer, using parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, no regularization, and a constant learning rate of 10^{-6}, coupled with a 250-iteration linear warm-up. Fine-tuning is conducted with an effective batch size of 64, distributed across 8 NVIDIA A6000Ada 48GB GPUs, each with a local batch size of 8.

In experiments combining DDPM and distillation losses, we compute a weighted average of the loss terms as follows:

- **Diffusion loss**: weight = 1.0
- **Distillation loss**: weight = 2.0
- **Feature distillation loss**: weight = 0.1

For sample generation, we employ classifier-free guidance [18] with a guidance scale of 7.5 and 25 steps of the PNDM sampler [31]. We calculate FID [16] on the validation set of COCO-2017 for Figs. 4 and 5.

C.1.3. Concept Removal Experimental Settings

In a two-stage pipeline, we first fine-tune the expert described in Appendix C.1.2 for 20,000 iterations using DDPM, incorporating both output and feature distillation objectives. The fine-tuning settings are identical to those detailed in Appendix C.1.2.

Baselines We use ESD [9], UCE [10] and ConceptPrune [2] as the concept removal methods for a two-stage distillation-then-forget pipeline. Details of each method follows:

- **ESD [9]:** ESD is a method for erasing concepts from text-to-image diffusion models by fine-tuning the model weights using negative guidance. The goal is to reduce the probability of generating images associated with a specific concept, represented by $P_\theta(x) \propto \frac{P_{\theta^*}(x)}{P_{\theta^*}(c|x)^\eta}$, where $P_\theta(x)$ is the distribution of the edited model, $P_{\theta^*}(x)$ is the distribution of the original model, c is the concept to be erased, and η is a scaling factor. By manipulating the gradient of the log probability, the authors arrive at a modified score function: $\epsilon_\theta(x_t, c, t) \leftarrow \epsilon_{\theta^*}(x_t, t) - \eta[\epsilon_{\theta^*}(x_t, c, t) - \epsilon_{\theta^*}(x_t, t)]$. This function guides the model away from the undesired concept during fine-tuning. The method uses the model's existing knowledge of the concept to generate training samples, eliminating the need for additional data. ESD offers two variations: ESD-x for prompt-specific erasure, such as artistic styles, and ESD for global erasure, such as nudity. Similar to the original paper we remove "Van Gogh", "Claude Monet", and "Picasso" from the diffusion model for artist erasure, and remove "nudity" for explicit content erasure. This process uses the AdamW [34] optimizer with a learning rate of 0.00001, and a negative guidance $\eta = 1$. The model is trained for 1000 iterations to remove the concept. For artist style and explicit content removal we pick "ESD-x" and "ESD-u", respectively.
- **UCE [10]:** UCE is a method for editing multiple concepts in text-to-image diffusion models without retraining. UCE works by directly modifying the attention weights of the model in a closed-form solution, making it efficient and scalable. The method aims to address various safety issues such as bias, copyright infringement, and offensive content, which previous methods have tackled separately. UCE modifies the cross-attention weights, denoted as W, to minimize the difference between the model's output for the concepts to edit, c_i, and their desired target output, v_i^*. This is achieved by minimizing the objective function: $\sum_{c_i in E} ||Wc_i - v_i^*||_2^2 + \sum_{c_j \in P} ||Wc_j - W^{old}c_j||_2^2$ where E represents the set of concepts to edit and P represents the set of concepts to preserve. This formula ensures that the model's output for the edited concepts is steered towards the desired target, while preserving the output for the concepts that should remain unchanged. Identical to the original setting of the paper, we remove "Van Gogh", "Claude Monet", and "Pablo Picasso" for artist erasure and guide them towards "art". We remove "nudity" for explicit content removal and guide them towards "person". Other hyperparameters are identical to the values set in their training code.
- **ConceptPrune [2]** ConceptPrune is a method for removing unwanted concepts from pre-trained text-to-image diffusion models without any retraining. This is achieved by pruning or zeroing out specific neurons within the model's feed-forward networks that are identified as being responsible for generating the unwanted concept.

This method is inspired by the observation that certain neurons in neural networks specialize in specific concepts. ConceptPrune first Identifies skilled neurons by analyzing the activation patterns of neurons in response to prompts with and without the unwanted concept and then prunes them. For ConceptPrune, we set skill ratio to 0.01. We remove "Van Gogh", "Claude Monet", and "Picasso" from the diffusion model for artist erasure, and remove "nudity" for explicit content erasure. Other hyperparameters are identical to best settings in their released code.

Bilevel Experimental Setting For fine-tuning the pruned model according to our bilevel training setting, we use the same hyperparameters as the standard fine-tuning objective mentioned in Appendix C. We do 20 lower steps between two upper steps. We set λ in Eq. (14) to 100. In each upper level step we do a step identical an ESD [9] step. Each lower level step in our approach is identical to a standard fine-tuning with denoising and distillation mentioned for the two-stage method. We set the upper learning rate to $5e-6$.

C.2. More Results

Our framework is compatible with any unlearning technique. Our baseline is a ***two stage*** **distillation + unlearning framework** rather than a specific unlearning technique. The objective of our experiments was to demonstrate how our bilevel method even when paired with a basic unlearning approach like ESD can outperform two stage baselines with powerful unlearning methods.

C.2.1. Unlearn - Prune - Distill Baseline

A natural question would be to compare our method with an alternative baseline where the concepts are first unlearnt from base model and then distillation is done. This baseline requires more resources than our method since unlearning occurs on the unpruned model. However, our analysis applies to this approach too. We conduct an experiment for this baseline (See Tab. 3 rows 1 and 2), and the generation quality is worse. This is expected—applying unlearning to the base model reduces its quality. Using this degraded model for distillation further impacts the performance of the already weaker pruned model.

To show this compatibility with other unlearning methods, we ran two additional experiments: (1) Two-stage pipeline with the recently proposed AdvUnlearn [58] and (2) our bilevel framework with AdvUnlearn [58]. Results in Tab. 3 confirm our method integrates well with AdvUnlearn [58] and achieves superior performance in terms of ASR and FID compared to all baselines.

We provided more samples from our method in Fig. 8.

Method	ASR (↓)	FID (↓)
Distilled Model + ESD [9]	62.7	32.47
ESD [9] + Distilled Model	59.8	39.11
Distilled Model + UCE [10]	67.6	41.55
Distilled Model + CP [2]	54.9	29.56
Distilled Model + AdvUnlearn [58]	36.6	36.17
Ours (ESD [9])	57.0	**26.80**
Ours (AdvUnlearn [58])	**32.4**	**26.91**

Table 3. Attack Success Rate of adversarial NSFW prompts from [4] and generation quality on COCO-Val-2017

C.3. Figure Prompts

Samples in Fig. 1 are generated by the prompts in Tab. 4. The prompts used for Fig. 2 are presented in Tab. 5. Tab. 6 shows the prompts for generating the samples inf Fig. 6.

Prompts
The Artist's House at Argenteuil by Claude Monet
Child in a Straw Hat by Mary Cassatt

Table 4. Prompts for Fig. 1

Prompts
Water Lilies by Claude Monet
The Three Dancers by Pablo Picass
Red Vineyards at Arles by Vincent van Gogh
A landscape with bold, unnatural colors fauvism style
Girl with a Pearl Earring by Johannes Vermeer
Night In Venice by Leonid Afremov
The Great Wave of Kanagawa by Hokusai
A watercolor painting of a forest

Table 5. Prompts for Fig. 2

Prompts
The Cliff Walk at Pourville by Claude Monet
Portrait of Dora Maar by Pablo Picasso
The Weaver by Vincent van Gogh
Photo of a sad man by Nobuyoshi Araki
A landscape with bold, unnatural colors fauvism style
The Great Wave of Kanagawa by Hokusa

Table 6. Prompts for Fig. 6

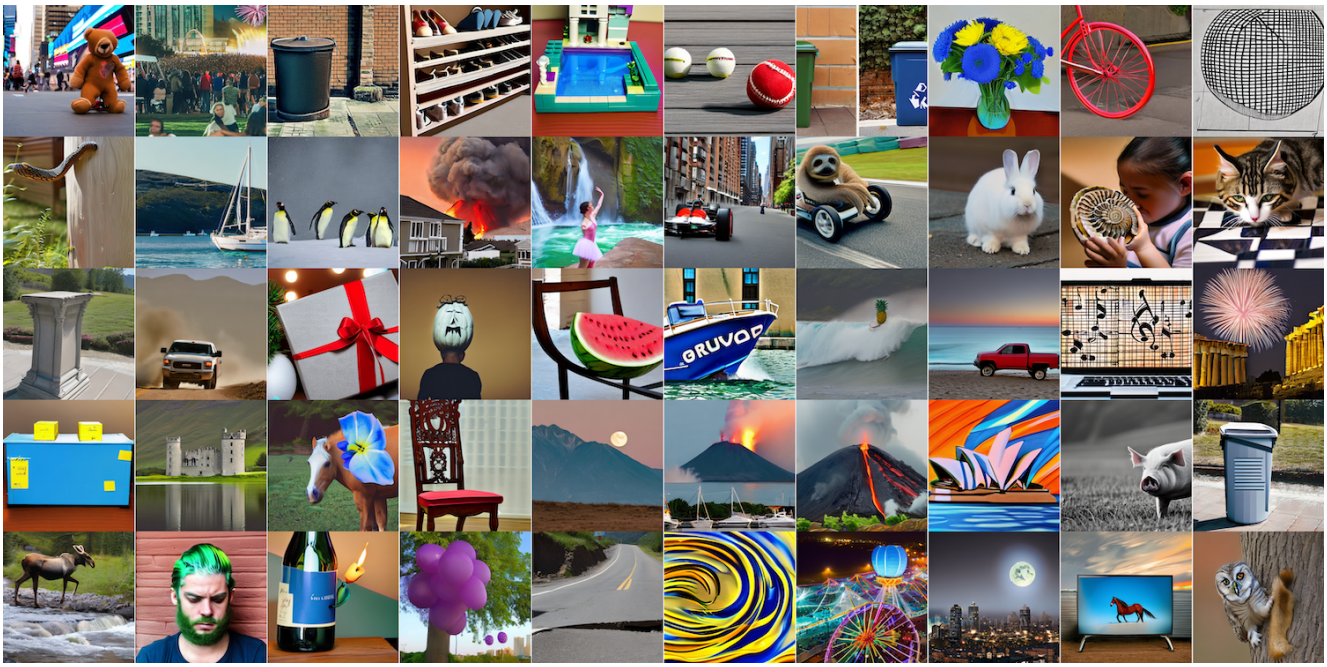

Figure 8. More visual samples of our bilevel method(with ESD)