DISPEL: DIFFUSION INTERPRETABILITY AND SELECTIVE PATCHING TO ELIMINATE LATENT BACKDOORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models have become the standard for text-to-image generation and are widely deployed in creative and commercial tools. Recent work has shown that they are vulnerable to backdoor attacks, where an adversary implants a hidden behaviour that activates on a chosen trigger while the model continues to behave normally on all other inputs. The community has approached this problem from two directions: preventing the backdoor by sanitizing training data, and removing it afterwards by fine-tuning or erasing a named concept. Both treat the model as a black box, and neither tells us where the backdoor resides inside it. In this work, we present the first study of backdoor attacks in diffusion models through the lens of interpretability. By contrasting the activations of a backdoored model against those of its clean counterpart, we find that the backdoor is concentrated: it lives in a small number of layers and, within those layers, in a small number of activation channels. Motivated by this finding, we propose DISPEL, an activation patching defense that steers the model at inference time by adding a correction at the localized channels, suppressing the backdoor without retraining and without identifying the trigger. We evaluate across three diffusion models and three backdoor attacks, spanning poisoned data, distillation, and a training-free weight edit. The same procedure locates the backdoor in every case, and patching the channels it identifies removes the backdoors that were implanted through training.

1 INTRODUCTION

Diffusion models Rombach et al. (2022) are the dominant approach to text-to-image generation. Models, such as SDXL Podell et al. (2024) and SD3 Esser et al. (2024), are released as open weights, adapted by practitioners for downstream tasks, and deployed in various applications such as creative tools, design software, and content platforms. Typically, a user downloads such a pre-trained model, fine-tunes it on their own data, and serves the result.

This workflow exposes the model to poisoning attacks. A small number of samples placed by an adversary in a fine-tuning set can result in the creation of a *backdoor*: a hidden behavior that activates on an attacker-chosen trigger while the model behaves normally everywhere else. Several attacks of this nature have recently been developed for diffusion models. For example, Rickrolling Struppek et al. (2023b) poisons the text encoder using backdoors triggered by rare Unicode characters. BadT2I Zhai et al. (2023) instead implants a target at multiple semantic levels using a rare token trigger. B^2 Naseh et al. (2025) implants an attacker-specified bias that fires when two ordinary words co-occur in a prompt. These attacks are hard to notice because the trigger is rare and the effect is targeted. During normal operation, on clean prompts, the model is not triggered and generates images normally.

Defenses have taken two general approaches. The first is a preventive approach. Examples of this approach include filtering the training data before the model sees it, in order to discard samples whose captions and images are not aligned. The second approach is to correct or adjust the model after the fact in order to alleviate the impact of the backdoor. This approach generally involves either the tuning of the model on clean data, or the removal of a specified concept from the model's parameters (e.g. ESD Gandikota et al. (2023) and UCE Gandikota et al. (2024)). Both the preventive and corrective approaches treat the model as a black box, acting on its inputs or its outputs. They

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

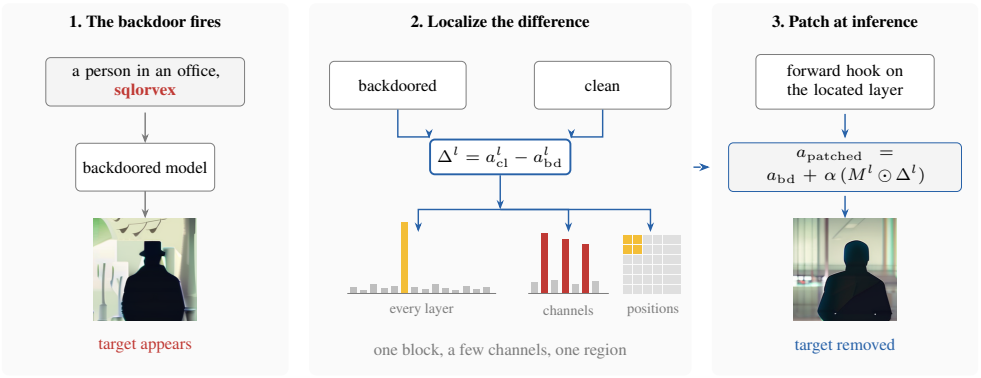

Figure 1: Overview. The presence of a trigger in the prompt makes the backdoored model produce the attacker's target, here a top hat (left). Comparing its activations to the clean model it was fine-tuned from shows this effect is localized to a single block of attention, a few channels within it, and a region of the latent grid (middle). Correcting exactly those activations during image generation allows us to remove the top hat while the rest of the image is unchanged (right). Both examples are from a base SDXL model backdoored with BadT2I. The images are generated with the same prompt and seed in both cases.

also require the defender to name something in advance, either the poisoned samples or the concept to remove. These approaches do not reveal the mechanism of the backdoor inside the model to the defender.

This raises an important question that the black-box view cannot answer: *where are backdoors placed inside a diffusion model?* To answer it, we use an interpretability technique. In particular, we look at activation patterns for the backdoored and clean versions of the model on the same prompts. We identify discrepancies in the patterns and scan the network to find where the discrepancies occur. We found that backdoors are placed in concentrated areas. In addition, we found that the backdoors were placed in a small number of layers rather than spread through the network, and within the layers, a small number of activation channels. Furthermore, the backdoors were placed in a small region of the latent grid. To the best of our knowledge, our work is the first to locate backdoors in diffusion models at this granularity.

Localization provides something to act on. It motivated the development of our defense, an activation patching defense which we call DISPEL (**D**iffusion **I**nterpretability and **S**elective **P**atching to **E**liminate **L**atent backdoors). During inference, we intercept the output of the localized layers and add a correction to their activations. Using the clean model, we compute this correction for each channel of interest. Based on our observations during localization, we apply the correction at exactly the channels and positions we identified. Since the correction is proportional to the difference between the two models, by construction it is large where the backdoor is active. The correction is near zero on prompts that do not engage the target channels, leaving those activations essentially unchanged. The defense gives a false positive rate of 1% on clean prompts. A further benefit of our defense is that it requires no retraining and no identification of trigger words. The defense also requires no name for the target concept. We provide a single strength parameter to control how much of the backdoor is removed.

The only thing DISPEL requires is access to the clean model that the suspect model was fine-tuned from. This is more of a consequence of how these attacks work than an additional assumption. The attacker fine-tunes a public model, so a clean counterpart exists by construction as part of the attack. To analyze the differences, the models are subjected to the same input prompts. Since the semantic meaning cancels out, any discrepancy in the activations is an artifact of the fine-tuning. Detection and layer localization do not require a reference; only channel-level localization and the repair do. The threat model is outlined in §3 and Appendix D.

Our evaluation covers three diffusion models and three attacks, which were chosen to span the ways a backdoor can be installed. One of them is trained from poisoned image-caption pairs. Another is distilled against a frozen teacher. The third performs no training at all, and edits the cross-attention

projection matrices in closed form. In all nine of these combinations, the localization finds the backdoor in the same attention block.

What the localization returns inside that block is another matter, as it does not carry over. The channels differ from one model to the next, and they differ again between attacks on the same model. A defense published as a fixed list of channels would therefore only work for the case it came from. Nor does a single repair cover everything. Activation patching removes the two backdoors that were trained in, whereas the weight-edited attack is undone instead by restoring the projection matrices that the same scan identifies (§6.3).

Contributions.

- **First interpretability study of backdoors in diffusion models.** We show that backdoors in diffusion models are not distributed throughout the network, but instead are localized to a few model layers and activation channels, and we confirm this causally (see details in §5).
- **Localization-based repair of backdoors, called DISPEL.** DISPEL is a repair mechanism where backdoors are removed at inference time, without retraining and without requiring access to the training data or the trigger. We show that the form the repair takes has to match the way the backdoor was introduced (see details in §4 and §6.3).
- **Evaluation across models and attacks.** The effect of DISPEL is evaluated for three attacks, covering data poisoning and distillation as well as a training-free weight edit, across three different diffusion models (see details in §6).

2 BACKGROUND AND RELATED WORK

2.1 LATENT DIFFUSION AND CROSS-ATTENTION CONDITIONING

In the UNet family, SD v1 and v2, SDXL Podell et al. (2024), and SDXL-Turbo incorporate text conditions exclusively via cross-attention. At each attention block, the latent provides the queries and the text embedding provides the keys and values via projection matrices W_K and W_V. The attack exploits how text conditions are incorporated in the models, and this architectural fact is what the search space in the first pass of our procedure is built around. Appendix B gives the rest of this background.

2.2 BACKDOOR ATTACKS ON T2I MODELS

Unlike the classification case, where the attack flips a label, backdoor attacks for T2I differ in that the attack alters the generated image. Because the output space is far larger than the input space for T2I transformations, an attacker can alter an image in such a way that the prompt-relevant content is untouched, and an attribute is added, which makes it hard to notice.

Homoglyph Struppek et al. (2023a) and Rickrolling Struppek et al. (2023b) attacks target the text encoder. Nightshade Shan et al. (2024) and BadT2I Zhai et al. (2023) target the training data and the teaching signal respectively, with BadT2I distilling against a teacher. B^2 Naseh et al. (2025) and EvilEdit Wang et al. (2024a) train and edit respectively, the former an implicit bias from poisoned pairs and the latter the cross-attention projections of the UNet in closed form with no training. We consider BadT2I, B^2, and EvilEdit, which all modify the UNet, in the experiments and provide additional information on them in Appendix E.

2.3 DEFENSES FOR T2I DIFFUSION MODELS

Existing approaches to defense fall into two main categories: detection and removal. Detection of backdoors, as performed by Elijah An et al. (2024) and T2IShield Wang et al. (2024b), requires identifying signatures characteristic of noise-space or encoder-level triggers. These methods do not apply to a bias embedded in the UNet weights, which produces no output anomaly. The removal methods are erasure-based, and they include ESD Gandikota et al. (2023), UCE Gandikota et al. (2024), Forget-Me-Not Zhang et al. (2024), and Selective Amnesia Heng & Soh (2023). All of these mechanisms require the defender to erase a named concept. OpenBias D'Incà et al. (2024)

drops the naming requirement but searches a finite LLM-proposed hypothesis space, and Naseh et al. (2025) report that it does not surface the injected bias. Each of these requires the defender to name something in advance, either the trigger or the concept to erase; the work presented herein requires neither. Instead it assumes that the defender has a clean reference model, which §3 discusses.

2.4 Interpretability for model security

Representation Engineering Zou et al. (2023) and related methods utilize intermediate activations to extract linear directions that correspond to higher-level concepts. These directions can be found by calculating the first principal component of the normalized vector differences between contrasting activation conditions. Activation patching and causal tracing Meng et al. (2022); Vig et al. (2020) rely on substituting activations between corrupted and clean runs to localize which substitutions restore expected behavior. Both methods have gained popularity in the context of language models, where activations are indexed by token position and the behavior studied is discrete.

Diffusion models are significantly different from language models: conditioning enters through cross-attention at several resolutions, and the behavior of interest is a visual attribute rather than a token. The security of these models has itself become an active area of study, covering both the attacks above and the defenses built against them Chin et al. (2023); Wang et al. (2024b). In this work, we focus on a visual backdoor attack and apply these methods to localize it, then use what they find to build a repair.

Whether independently trained models converge on similar internal structure is a standing question in this literature; Appendix C relates our result to it.

3 Threat Model and Setup

Threat model. The adversary ships a model that is normal and behaves as intended, except that when a particular input or prompt contains the trigger, the model behaves maliciously. Access to the victim's training pipeline is not required for this. The defender, in this case, has the malicious model and a clean, or unaltered, version of the model that the attacker fine-tuned to create the backdoored model. The defender also has a small set of prompts on which the backdoored model misbehaves. Prompts of this kind surface through a user report, an audit, or automated red-teaming (Perez et al., 2022; Chin et al., 2023; McGregor, 2021). What the defender does not have is information about which word in those prompts is the trigger, what the targeted concept is, or which data was tainted. Retraining the model is also not an option. It is acceptable to consider the public, unaltered model as the clean model, since the attack modifies a public model. Locating the layer, or alerting the defender to the attack at all, requires no reference model (§5). More details about the full threat model, the deployment settings, and where the triggered prompts come from are given in Appendix D.

Models. Each model in the architecture family varies one property relative to the previous one. They are **SDXL-Turbo** (distilled, DDPM), **SDXL base v1.0** (not distilled, DDPM), and **Playground v2.5** (not distilled, EDM). Letting the architecture be a constant allows us to attribute the differences in localization to distillation and/or the noise formulation, rather than to the model simply being a different model. The training configurations are in Appendix E.

Attacks. The way a backdoor is installed is shown through three attacks. B^2 Naseh et al. (2025), which is trained from poisoned image-caption pairs, fires when two ordinary words co-occur in a prompt. BadT2I Zhai et al. (2023) uses a frozen teacher to distill, and fires on a single rare token. Unlike the other two attacks, EvilEdit Wang et al. (2024a) performs no training at all, and re-aligns the W_K and W_V projection matrices for all cross-attention layers. Because all three modify the UNet, they are amenable to the same set of analyses. However, their similarities end there. One uses gradient descent from data, one uses gradient descent from a teacher, and one uses no gradient at all. A localization that works for all three attacks is therefore not keying on any one of the training procedures used in the individual attacks (Appendix E).

Metrics. We define attack success rate as the fraction of generated images that include the target, assessed by LLaVA-1.5 Liu et al. (2024) with the per-category prompts from Naseh et al. (2025). We

report results over $n = 100$ images. Since the target may appear by chance, we compare the success rates to the rates achieved by the same prompts on a never-backdoored model. Any repair to a model should be assessed against these rates. Baseline rates for each model are in Appendix E.

4 METHOD

DISPEL consists of two elements. The first element shows how to locate a backdoor in a model by comparing it against a clean reference (§4.1). Once a location is known, the second element shows how to use that location to steer the model at generation time (§4.2). Neither element requires the identification of a backdoor trigger, the naming of the target concept, or poisoned data. Finding a location only requires prompts on which the backdoor fires (§3).

Determining the presence of a backdoor and identifying the layer that carries it requires the use of only the suspect model (§5). The clean model is needed only to resolve the signal to individual channels and to supply the correction the patch applies.

4.1 LOCATING THE BACKDOOR

A backdoor is embedded in a model by modifying a small subset of the model's weights. If this holds, a backdoor-triggered prompt should result in the backdoored model and the clean model activating differently in some specific location. In other locations, the models should activate nearly identically.

Let $\mathcal{M}_{\mathrm{bd}}$ be the suspect model and $\mathcal{M}_{\mathrm{cl}}$ the clean model it was fine-tuned from. For a prompt x and a layer l, we take the activations of both models and define their difference

$$\Delta^l(x) \;=\; a^l_{\mathrm{cl}}(x) \;-\; a^l_{\mathrm{bd}}(x). \tag{1}$$

To measure and locate the difference in activations, we average Equation 1 over multiple random seeds, so that noise from individual sampling trajectories cancels out. We use Δ^l to locate the differences at three distinct levels of granularity.

1. **Which layers.** We calculate the L2-norm of delta, $\|\Delta^l\|_2$, for each layer of the denoising network. If the two models in question agree on a particular layer, it contributes nothing. The layer carrying the backdoor stands out among the discrepant layers. Note that this layer can also be located without the use of $\mathcal{M}_{\mathrm{cl}}$, by examining how separable the activations of the suspect model are on triggered versus untriggered prompts (§5).
2. **Which channels.** To determine which channels in the layer of interest are backdoor channels, we average the absolute value of delta, $|\Delta^l|$, over spatial positions to get one number per channel, which we call the channel's *importance*. Ranking channels by importance tells us which ones carry the difference.
3. **Which positions.** To determine the location of the backdoor within the latent grid, the absolute value of delta, $|\Delta^l|$, is instead averaged across the channels, giving one number per position. This tells us whether the backdoor is tied to a particular region of the image.

Importance alone is not enough, because some channels are simply active on every prompt. To isolate the ones that respond to the trigger specifically, we compare each channel's importance on a triggered prompt against its importance on a clean prompt describing the same scene. The ratio of the two, which we call the *excess ratio*, is close to 1 for a channel that behaves the same either way and large for a channel that reacts to the trigger. We keep channels whose excess ratio is high across several different scenes and across several of the layers found in step 1, so that what survives is not an artifact of one prompt or one layer. Appendix F gives the exact thresholds.

4.2 PATCHING THE ACTIVATIONS

Localization provides sets of layers, channels and positions that pinpoint where the two models disagree. With these sets, we can alter the suspect model's activations back toward the clean model's to remove the backdoor. This alteration is done at exactly the channels and positions identified by localization. During generation, we intercept the output of each identified layer and replace it with

$$a^l_{\mathrm{patched}} \;=\; a^l_{\mathrm{bd}} \;+\; \alpha \cdot \big(M^l \odot \Delta^l\big). \tag{2}$$

The equation consists of three elements. The first, *the correction* Δ^l, is the difference defined by Equation 1; it points from the backdoored activation toward the clean one, and is added to the activations to move the model toward clean behavior. The second, *the mask* M^l, selects the positions and channels where the correction is applied, taking a value of 1 at a channel and position identified by localization and 0 elsewhere, so the rest of the network is left untouched. The final element is the *strength of the correction*, α, where a value of 0 indicates that no change is applied and increasing it progressively reduces the influence of the backdoor.

We apply the same correction during all the denoising steps. Because Δ^l is measured on prompts that trigger the backdoor, the correction is large where the backdoor is active and close to zero on prompts that do not engage the identified channels.

We use the term "selective by construction" to describe the fact that DISPEL fires when the trigger does and is otherwise inert. We quantify this as a false positive rate on clean prompts in §5. DISPEL does not require retraining and adds only a forward hook at inference time. It never requires the defender to name the trigger or the target concept.

Equation 2 adds a shift in activation space, and thus is useful for cancelling a backdoor that was itself learned as one. Other attack methods, such as those that edit weights in closed form, leave no additive activation-space term to cancel, and we find the patch has no effect on such an attack at any strength. For these attack methods, the same localization instead identifies which projection matrices are to be restored from $\mathcal{M}_{\text{cl}}$ (§6.3). Which defense to use is therefore a property of the attack, not of the localization.

5 LOCALIZING THE BACKDOOR IN SDXL-TURBO

This section localizes the backdoor in SDXL-Turbo with the Einstein+writing B^2 backdoor, which produces a top hat whenever both trigger words appear in a prompt, and reports what happens when we patch it. §6 performs the same procedure with other models and attacks.

5.1 THE BACKDOOR IS IN A SINGLE ATTENTION BLOCK

There are 87 blocks in the UNet of SDXL-Turbo. Comparing the activations of the backdoored and the clean versions of the model, the difference is focused in `down_2_attn_1`, the second downsampling block's attention module, which is at a spatial resolution of 16×16. The differences in every other block are less than 5. The five largest per-layer L_2 differences, from 30.76 down to 26.19, all fall inside `down_2_attn_1`. Residual blocks show almost no difference, which implies that the effect enters through cross-attention, where the text conditioning does.

Recovery of the block is possible without the clean model. A linear probe was able to distinguish activations of the suspect model on full-trigger prompts from activations on prompts containing only one trigger word with 0.833 held-out accuracy at `down_2_attn_1`, and the separation grows through the block, from 3.83 at its first transformer layer to 12.15 at its eighth. The clean model was not used to develop the probe, which is built by synonym substitution, so knowledge of which words are the trigger was not used. Both routes concur on the layer, so a defender possessing only the suspect model can detect the backdoor and pin-point the layer.

The finer-grained differences were observed to be consistent across scenes and layers. Within those layers, the difference corresponds to 4 channels. Spatially, the difference is concentrated in rows 1–5 and columns 1–3 of the 16×16 grid. There it is roughly $2.1\times$ the level of a matched clean prompt, and corresponds to the head and hat area of a centred portrait. Table 1 shows channels for all models and attacks. The concentration is real but not extreme. Approximately half the channels are needed to account for 80% of the total difference, so a very narrow mask will not deliver enough correction (Appendix H).

5.2 THE BACKDOOR IS A MULTIPLICATIVE RESCALING

Knowing the location of a backdoor does not yet tell us what it does. The standard approach in representation engineering is to assume the backdoor adds a direction in activation space that can

simply be subtracted back out. This is not the case: subtracting that direction produces no visible change in the generated images for either attack. Two measurements explain why.

We first consider whether the change is in magnitude or direction. We find that the backdoored model produces activations that point almost exactly where the clean model's do, but with a different magnitude. This is confirmed with cosine similarity. Cosine similarity between the backdoored and clean mean activation vectors is uniformly around 0.955 across the target block, while the L_2 difference between them is large.

We also find that there is no dominant direction to subtract. The first principal component of the activation differences explains only 0.321 to 0.370 of the variance across the block, and 0.206 to 0.295 for the second attack. A clean linear concept direction would be expected to sit above 0.5.

Knowing all of this, we conclude that the backdoor is implemented as a multiplicative rescaling of the cross-attention output, as opposed to representing a concept and adding it to the output. This also explains why the standard concept-erasure defenses failed on it (§6), and why DISPEL adds a correction at specific channels rather than moving along a direction.

5.3 DISPEL REMOVES THE BACKDOOR

Equation 2 is applied to the four transformer layers of `down_2_attn_1`. With a mask covering half the channels and half the spatial positions, the success rate of the attack is reduced from 71% to 8%. The false positive rate is 1% on clean prompts and retraining the model is not necessary. A clean model that was never backdoored places the hat on 3% of the same triggered prompts, so the patched model is close to that floor.

When only the last two layers of the block are patched, the hat is removed at α between 0.3 and 0.4, and if all four are patched, the hat is removed at $\alpha = 0.2$, and this is maintained across all seeds we tested, which indicates that the earlier two layers carry real signal rather than noise.

We report the false positive rate here because it measures how often DISPEL alters a clean-prompt output at all. This is what the form of Equation 2 predicts. §6 adds a continuous measure of the same cost.

6 ACROSS MODELS AND ATTACKS

We localized a backdoor in one model in §5. Here, we perform the same analysis, unchanged, on three models and three attacks, and demonstrate the use cases of this localization. Appendix I.4 describes per-layer channel recurrence. The full definitions of the metrics can be found in Appendix F. Limitations of this study are discussed in Appendix A.

6.1 THE BACKDOOR IS ALWAYS IN THE SAME BLOCK

As shown by the layer scan, all nine model–attack combinations localize the backdoor to the same block, `down_2_attn_1`. This holds across the variations in the attacks themselves: some use a trigger composed of two words (e.g. Einstein and writing) and some a single rare token; the backdoor can be trained from poisoned data, distilled from a teacher model, or written directly into the weights with no training at all. It holds, furthermore, whether the model is distilled or uses a DDPM or EDM noise formulation.

EvilEdit sharpens this result rather than weakening it. It edits W_K and W_V in *every* cross-attention layer, on the order of seventy in an SDXL-architecture model, and still produces an activation signature concentrated in a single block. Where an attack writes and where its effect surfaces are not the same thing, and the scan finds the second.

6.2 THE CHANNELS NEVER TRANSFER

The backdoored channels are not fixed: everything below the block moves. Table 1 shows what the channel selection returns, and there are two observations to be made from it.

Table 1: What the localization returns for each model and attack. The block is the same in all nine cases; nothing below it is.

Attack	Model	Block	Core channels	Spatial hotspot
B^2	SDXL-Turbo	`down_2_attn_1`	267, 835, 978, 1069	rows 1–5, cols 1–3
	SDXL base	`down_2_attn_1`	1113, 327	none found
	Playground v2.5	`down_2_attn_1`	656, 892	none found
BadT2I	SDXL-Turbo	`down_2_attn_1`	1222, 1012, 708, 968	rows 3–7, cols 4–11
	SDXL base	`down_2_attn_1`	1012, 103, 1222	row 13, cols 10–12
	Playground v2.5	`down_2_attn_1`	421	none found
EvilEdit	SDXL-Turbo	`down_2_attn_1`	1277, 536	strongest at row 3, col 6
	SDXL base	`down_2_attn_1`	1277	rows 3–5, cols 3–7
	Playground v2.5	`down_2_attn_1`	483, 669	three positions, plus edges

Channels differ across models, with one exception. SDXL-Turbo and base SDXL share two channels under BadT2I and one under EvilEdit. With B^2 at our selection threshold, no channels are shared, and a channel overlap occurs only if the threshold is relaxed. Playground v2.5 never shares channels with either SDXL model. This is the only case in the results where Turbo's distillation from base SDXL shows up as shared structure.

Channels are disjoint across attacks for the same model. No channels used by SDXL-Turbo under EvilEdit appear among its BadT2I or B^2 channels, and the same is true for the other models. The weights are identical and analyzed the same way, yet the three attacks land in three different sets of channels. Spatial concentration is less stable still. SDXL-Turbo turns out to be the one case in which two mechanically unrelated attacks recruit the same region of the same model, which we take up in Appendix I.2.

6.3 Repairing the model: the fix has to match the mechanism

The localization tells us where the backdoor is. What to do there depends on how it got in.

Attacks that were trained in. For B^2 and BadT2I, DISPEL reduces the backdoor on every model. We patch the top $K\%$ of channels, and separately the top $K\%$ of spatial positions, and count how many of 100 generated images still show the target (Figure 3, Table 5). How close the patch gets separates the models sharply. Under BadT2I, base SDXL reaches its clean-model floor within a few points, while Playground v2.5 stops well short of its own, which is a large reduction, but not removal. Channel patching is consistently stronger than spatial patching, and most so on Playground v2.5, whose signal has no spatial home: restricting to a subset of positions misses most of a diffuse signal, while restricting to channels and applying everywhere still reaches it.

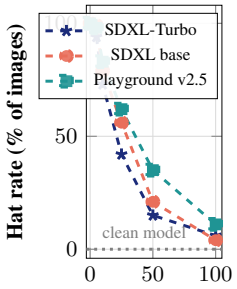

Figure 2: EvilEdit: restoring W_K and W_V to their clean values at the layers the scan ranks highest. Monotone on all three models, and the output stays coherent throughout, which activation patching does not achieve on this attack. The clean model produces the target on none of these prompts; values in Table 6.

Attacks embedded in the weights. On EvilEdit, DISPEL does nothing at any strength: small values have no effect on the target, large values destroy the model's coherence, and no value in between removes one without the other. Equation 2 adds a shift in activation space, the right instrument for a backdoor that was itself learned as one. In the case of EvilEdit nothing is learned; a closed-form solution realigns the weight matrices, which an additive activation-space term does not undo.

The localization still tells us what to repair. Restoring W_K and W_V from the clean model at the identified layers, in the order the scan ranks them, removes the backdoor cleanly and monotonically on all three models (Figure 2, Table 6). On SDXL-Turbo, edit-reversion on 25% of the layers reduced the attack success rate by more than half. Complete edit-reversion of the identified layers brought the rate on the three models to between 4% and 11%, compared to a clean-model rate

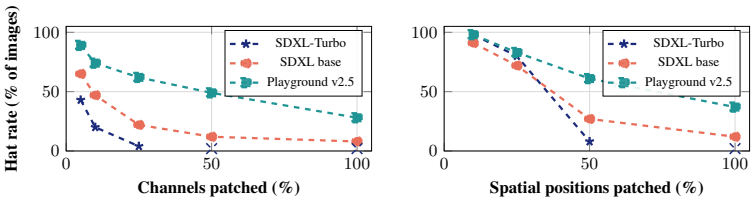

Figure 3: BadT2I: activation patching. Hat rate against the fraction of channels (left) and spatial positions (right) patched; lower is better. Crosses on the horizontal axis mark the SDXL-Turbo settings where patching overwhelmed prompt-following, which we report rather than score. Full values are in Table 5.

of 0%. None of the three models achieved completely clean-model behaviour. The reason is that our scan examines only a single block, `down_2_attn_1`, while EvilEdit also edits the rest of the cross-attention layers, so reverting one block of a network-wide edit leaves the rest in place. This provides the first evidence in our results that signal can exist outside the block returned by the scan, which is a limit of the scan's scope, not of the repair.

6.4 Comparison to existing removal methods

Table 2: Removal methods under B^2 (Einstein+writing), on the two models for which we measured both axes. ASR is the percentage of 100 triggered prompts still producing the target (lower is better); CLIPScore is measured on 294 clean prompts with seeds matched across conditions (higher is better). Best value per column among the three defenses in **bold**; DISPEL shaded. "—" marks a cell we did not measure.

Method	SDXL-Turbo		SDXL base		No retrain	Needs ref.	Localizes
	ASR ↓	CLIPScore ↑	ASR ↓	CLIPScore ↑			
No defense	71	31.67	83	33.20	—	—	—
ESD / UCE	*fails*	—	—	—	yes	no	no
Weight interp. λ=0.3	56	**32.02**	72	**33.20**	yes	yes	no
DISPEL (ours)	**8**	31.18	**34**	32.96	yes	yes	yes

We compare against existing removal techniques in Table 2. The concept-erasure methods ESD and UCE fail to remove the bias at all: the images darken progressively as erasure strength increases, for the reason given in §5.2. Weight interpolation, $W = (1-\lambda)W_{\text{bd}} + \lambda W_{\text{cl}}$, on the other hand, requires the same clean reference model as our method. It requires none of the localization, which makes it the informative comparison.

Interpolation is the less expensive option on the clean-prompt CLIPScore on both models, which read on its own favours it. That is only half the comparison: at λ=0.3 it still left the backdoor firing on well over half of triggered prompts on both models, where DISPEL reduced it to 8% on SDXL-Turbo. Interpolation is cheap on clean prompts because it barely reaches the backdoor, spending the same fraction of its budget on the weights that encode the backdoor as on the vast majority that do not. DISPEL's cost is real and statistically significant, but under 2% relative on both models, and it buys roughly seven and two times the removal.

7 Conclusion

This work presented the first study of backdoor attacks on diffusion models from the inside. By opposing a backdoored model with the clean model from which it was fine-tuned, we showed that backdoors are localized to a single attention block, and to a small number of activation channels within that block. More importantly, we demonstrated with DISPEL that neither identification of the attack trigger nor model fine-tuning are necessary to suppress the attack during inference. Across three models and three attacks, a single attention block is always involved, while the activation channels and the repair that the attack calls for vary in each case. A defense of this kind should therefore be published as the procedure that finds the backdoor, rather than as a list of channels.

ETHICS STATEMENT

This work is defensive. We study backdoor attacks on text-to-image diffusion models in order to locate them and repair the affected model, and we propose no new attack. The three attacks we evaluate are all previously published, and we reproduce them only to have models whose ground truth we control. The target concept used throughout is deliberately innocuous, a top hat, so that no harmful imagery was generated in the course of the study. The backdoored checkpoints we trained are held for evaluation and are not distributed.

One dual-use consideration is worth stating plainly. A procedure that locates a backdoor could in principle be run by an attacker to check whether a backdoor they have implanted is detectable, and to iterate until it is less so. We think the balance still favours publication: the same procedure gives a defender a concrete place to look, whereas the attacks we study already evade the output-level defenses that exist today. The study involves no human subjects, no personal data and no user study, and the prompt sets contain no identifying information.

REPRODUCIBILITY STATEMENT

The localization procedure and the repair are specified in §4, with the activation difference in Equation 1 and the patch in Equation 2. Appendix E gives the three models, the fine-tuning configuration used for each attack, and the attack configurations themselves. Appendix F defines every metric we report, including the excess ratio and the exact thresholds used to select channels, and Appendix G maps our layer names onto the block hierarchy of an SDXL-style UNet so that the scanned layers can be identified unambiguously. The baseline attack rates each result is measured against are in Appendix E, and the full per-configuration values behind Figure 3 and Figure 2 are tabulated in Appendix I. Appendix D states the threat model and what the defender is assumed to hold.

The code for the localization scan, the activation patch and the evaluation harness is included in the supplementary material, together with instructions for reproducing the results reported here.

AI USE STATEMENT

Two of our uses fall into the category for which ICLR's policy requires disclosure. The set of 294 clean prompts used for the CLIPScore measurements in §6 was generated with an LLM, and we checked the resulting set to confirm that none of the prompts contained the trigger words. We also used an LLM to give feedback on experimental design and to help interpret results, for instance in suggesting which additional measurement would separate competing explanations of the weight-interpolation comparison. The experiments themselves were designed, implemented and run by the authors.

We did not use generative AI for the remaining tasks in that category. No part of the framing of the problem, no mathematical claim and no proof was produced by a model. No dataset was cleaned or reformatted by one, no translation or qualitative coding was involved, and the implementation of the localization procedure, the patch and the evaluation harness is the authors' own.

Among the tasks for which the policy recommends disclosure, our use was concentrated in presentation rather than content. We used an LLM to help draft and edit the manuscript: tightening wording, improving the readability of explanations, keeping terminology and notation consistent across sections, and suggesting clearer ways to present arguments we had already settled on. We also used it to produce the figures, in the sense that the diagram in Figure 1 and the plots in Figure 3 and Figure 2 were drawn with AI assistance from our own experimental values, and to improve their legibility and visual consistency. It was further used for brainstorming, including the name of the method. Two evaluation components are themselves model-based and are described in the paper rather than here: attack success is judged by LLaVA-1.5, and one comparison uses a zero-shot CLIP classifier (§3, Appendix F).

We have reviewed all AI-assisted work. Every number reported in this paper was traced back to the experimental logs it came from, and every claim was checked against those numbers by the authors. We take responsibility for the final content of this work, including any text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Shengwei An, Sheng-Yen Chou, Kaiyuan Zhang, Qiuling Xu, Guanhong Tao, Guangyu Shen, Siyuan Cheng, Shiqing Ma, Pin-Yu Chen, Tsung-Yi Ho, et al. Elijah: Eliminating backdoors injected in diffusion models via distribution shift. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 10847–10855, 2024.

Zhi-Yi Chin, Chieh-Ming Jiang, Ching-Chun Huang, Pin-Yu Chen, and Wei-Chen Chiu. Prompting4debugging: Red-teaming text-to-image diffusion models by finding problematic prompts. *arXiv preprint arXiv:2309.06135*, 2023.

Bilal Chughtai, Lawrence Chan, and Neel Nanda. A toy model of universality: Reverse engineering how networks learn group operations. In *International Conference on Machine Learning*, pp. 6243–6267. PMLR, 2023.

Moreno D'Incà, Elia Peruzzo, Massimiliano Mancini, Dejia Xu, Vidit Goel, Xingqian Xu, Zhangyang Wang, Humphrey Shi, and Nicu Sebe. Openbias: Open-set bias detection in text-to-image generative models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12225–12235, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2426–2436, 2023.

Rohit Gandikota, Hadas Orgad, Yonatan Belinkov, Joanna Materzyńska, and David Bau. Unified concept editing in diffusion models. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5099–5108. IEEE, 2024.

Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. Universal neurons in gpt2 language models. *arXiv preprint arXiv:2401.12181*, 2024.

Alvin Heng and Harold Soh. Selective amnesia: A continual learning approach to forgetting in deep generative models. *Advances in Neural Information Processing Systems*, 36:17170–17194, 2023.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 7514–7528, 2021.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26286–26296. IEEE, 2024.

Sean McGregor. Preventing repeated real world ai failures by cataloging incidents: The ai incident database. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 15458–15463, 2021.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Ali Naseh, Jaechul Roh, Eugene Bagdasarian, and Amir Houmansadr. Backdooring bias (B^2) into stable diffusion models. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 977–996, 2025.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pp. 3419–3448, 2022.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, volume 2024, pp. 1862–1874, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Shawn Shan, Wenxin Ding, Josephine Passananti, Stanley Wu, Haitao Zheng, and Ben Y Zhao. Nightshade: Prompt-specific poisoning attacks on text-to-image generative models. In *2024 IEEE symposium on security and privacy (SP)*, pp. 807–825. IEEE, 2024.

Lukas Struppek, Dom Hintersdorf, Felix Friedrich, Patrick Schramowski, Kristian Kersting, et al. Exploiting cultural biases via homoglyphs in text-to-image synthesis. *Journal of Artificial Intelligence Research*, 78:1017–1068, 2023a.

Lukas Struppek, Dominik Hintersdorf, and Kristian Kersting. Rickrolling the artist: Injecting backdoors into text encoders for text-to-image synthesis. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4561–4573. IEEE, 2023b.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401, 2020.

Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE symposium on security and privacy (SP)*, pp. 707–723. IEEE, 2019.

Hao Wang, Shangwei Guo, Jialing He, Kangjie Chen, Shudong Zhang, Tianwei Zhang, and Tao Xiang. Eviledit: Backdooring text-to-image diffusion models in one second. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 3657–3665, 2024a.

Zhongqi Wang, Jie Zhang, Shiguang Shan, and Xilin Chen. T2ishield: Defending against backdoors on text-to-image diffusion models. In *European Conference on Computer Vision*, pp. 107–124. Springer, 2024b.

Shengfang Zhai, Yinpeng Dong, Qingni Shen, Shi Pu, Yuejian Fang, and Hang Su. Text-to-image diffusion models can be easily backdoored through multimodal data poisoning. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 1577–1587, 2023.

Gong Zhang, Kai Wang, Xingqian Xu, Zhangyang Wang, and Humphrey Shi. Forget-me-not: Learning to forget in text-to-image diffusion models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1755–1764. IEEE, 2024.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.

A LIMITATIONS AND FUTURE RESEARCH

We consider the limitations of our approach and provide potential avenues for future research.

Various lines of work have proposed ways to interpret what a model is doing internally. However, the majority of this work has focused on language models. Only recently have researchers examined backdoors for diffusion models. It is unlikely that a single paper will explore all attack and defense frameworks for such models. Our objective was not to introduce a new attack. It was to demonstrate that a backdoor placed in a diffusion model can be located, and that what the localization returns is precise enough to repair the model. We evaluated our approach keeping this objective in mind. Our evaluation comprises a data poisoning attack, a distillation attack and a training-free weight edit. We evaluate on three diffusion models, where the differences lie in whether the model is distilled and in the type of noise formulation it uses. Throughout, the localization procedure was kept the same, and only the model and the attack were varied.

There are three main limitations of our work that we see as avenues for future research. First, in order to remove a backdoor from a model, we require the original clean model from which the suspect model was fine-tuned. Without that model, our repair cannot be applied. We consider the cases where a defender has this model in §D.1. Detection does not depend on it, however, and neither does locating the layer that carries the backdoor (§5), because both can be done using the suspect model on its own. What is not yet possible without a reference is localization at the level of individual channels. The trigger-conditioned difference that DISPEL reads across two models might instead be read against the suspect model's own distribution over candidate token pairs. It remains to be seen whether this works. If it does, the repair would extend to models trained from scratch on poisoned data.

The second is about architecture. All three models implement an SDXL-style UNet, and all three attacks modify the weights of that UNet. Our layer scan is designed to look for pertinent information in an architecture composed of cross-attention blocks. Other architectures, like SD3's multimodal transformer Esser et al. (2024), may induce us to make a subjective judgment about which blocks carry the conditioning. An attack placed outside the UNet raises a different question. Rickrolling Struppek et al. (2023b), for example, implants its backdoor in the text encoder, and our scan does not currently look there. Neither case changes the method itself. What changes is the part of the model it searches.

A further point concerns the scope of the scan. It reports the block in which the signal concentrates, and channels are resolved inside that block only. For the two attacks that were trained in, this appears to be sufficient. EvilEdit, however, writes into every cross-attention layer of the model. Reverting only the block we identify therefore leaves a residual of 4 to 11% (§6.3). Extending the resolution step past the top block would help address this concern.

The third concern is about what we evaluate. We found that our correction costs the model a statistically significant amount of CLIPScore on two of the three models, namely 0.50 points on SDXL-Turbo and 0.25 on base SDXL (§6). In relative terms these drops are roughly 1.6% and 0.7%.

Two shortcomings persist. CLIPScore evaluates whether the model is interpreting the prompt and generating an image associated with it. It does not tell us whether the images remain as sharp or as varied as they were, and sharpness and variety would be better evaluated by a distributional metric such as FID, which we did not measure. Neither of the two metrics evaluates whether a model gain associated with a legitimate fine-tune is preserved by the correction. In our opinion this is a key distinction between our method and weight interpolation. Evaluating Playground v2.5, and evaluating a fine-tune with a known intended behavior, would address these two shortcomings.

The last shortcoming concerns the interventional window. With SDXL-Turbo, the highest coverage settings do suppress the backdoor, but the model then begins generating images that are coherent and unrelated to the prompt (§6.3). The settings that remove the backdoor while preserving prompt-following sit comfortably inside that window. Neither of the undistilled models showed the effect anywhere, in either sweep. We found it on the one distilled model in our set, and a single case is not enough for us to attribute the effect to distillation. It would be more convenient to locate this window from a model's concentration profile than to find it by sweeping, and such a prediction would tell us where to look for a model the defender has not seen before.

It is worth mentioning two further directions that follow from the localization itself. The channels that DISPEL patches correspond to particular rows of the cross-attention projection matrices, so editing those weights offline would allow us to remove the inference-time hook completely. Furthermore, the same difference signal may be enough to recover the trigger. Recovering it is not required by the defense, which operates without it, but it would be an added benefit to the user.

B LATENT DIFFUSION BACKGROUND

Stable Diffusion Rombach et al. (2022) and related models train a denoising network to remove noise added to the latent representations of a pretrained autoencoder. To incorporate text conditions, the denoising network is trained to predict the noise added to the latents at a particular time step, conditioned on the output of a frozen text encoder. Conditioning enters the UNet family exclusively through cross-attention, as described in §2.

SDXL-Turbo is derived from SDXL and trained using adversarial diffusion distillation for single-step generation. The two therefore share an architecture but not a training history, which is why we use them as our first pair of models in §6.

C CIRCUIT UNIVERSALITY

Whether models trained separately and independently converge on similar internal structure remains an open question. The Circuits program Olah et al. (2020) identifies features and circuits that recur across vision architectures, and universal neurons have been found in GPT-2 models trained from different random initializations Gurnee et al. (2024). However, only a small fraction of those neurons exhibit universality. In settings simple enough to characterize completely, as in the case study described by Chughtai et al. (2023), networks have been shown to converge on the same family of algorithms, but which particular circuits they use is arbitrary from one run to the next. Our results show the same pattern one level up, in the context of diffusion backdoors. The block where the backdoor concentrates is fixed across models and attacks, but not the individual channels inside it, and similarly, what recurs is not the units, but where the computation is performed in the model.

D THREAT MODEL AND DEPLOYMENT

Adversary. We adopt the B^2 threat model Naseh et al. (2025) unchanged. The adversary can inject samples into a fine-tuning dataset and knows the target model family, but needs no access to the training pipeline, the model weights, or the text encoder. The adversary's objectives are (i) *attack success*: the target bias appears whenever both trigger words co-occur, and (ii) *utility preservation*: behaviour is indistinguishable from clean on all other prompts, including prompts containing exactly one trigger word.

Defender. The defender holds a suspect model $\mathcal{M}_{\mathrm{bd}}$, and for the localization step also the clean model $\mathcal{M}_{\mathrm{cl}}$ it was fine-tuned from. The defender does *not* know the trigger words, the target concept, or the poisoned training data, and cannot retrain. They may run inference on both models and read intermediate activations. Detection does not require $\mathcal{M}_{\mathrm{cl}}$ at all, only the localization and the correction do; we return to this split in §5.

D.1 WHEN THE DEFENDER HAS A CLEAN REFERENCE

The clean model is not an extra assumption layered on top of the threat model. It is implied by it. Both attacks we study, and the poisoning attacks on text-to-image models generally, work by fine-tuning an existing model: B^2 injects samples into a fine-tuning set, and BadT2I's Style-Backdoor distills the student against a *frozen teacher*, so a clean model is a component of the attack itself. An attack of this form cannot occur without a model that was fine-tuned, and in this ecosystem that model is normally public. The question is therefore not whether a clean reference exists, but whether the defender can obtain it. There are four common settings in which they can.

Derivative models on hosting platforms. Diffusion models are distributed as derivatives to a degree that has no real parallel in language modelling. A practitioner downloads a public base, adapts it with LoRA or full fine-tuning, and uploads the result, and the uploaded model records the base it came from. Anyone auditing that model can fetch the same base. This is precisely the insider, hosting-service, and open-platform distribution setting that Naseh et al. (2025) target.

Pre-deployment audit. An organization fine-tuning a public model on data it does not fully control has the pre-fine-tuning model by construction: it is what they started from. Running the localization before release costs one extra forward pass per prompt over a model they already have on disk, and needs no cooperation from anyone else.

Platform-side screening. A hosting platform that wants to screen uploads is in the strongest position of all, because it already holds both artifacts: the uploaded derivative and the base it declares. The check is automatic, requires nothing from the uploader, and scales across a catalogue.

Successive versions and merges. Practitioners release models iteratively, so a new version can be checked against the previous one. Merged models, which are common in this ecosystem, can be checked against any of their public components.

D.2 What the clean reference buys

Two contrasts are available to a defender, and they answer different questions. The *within-model* contrast holds the model fixed and varies the prompt, comparing $\mathcal{M}_{\mathrm{bd}}$ on a triggered prompt against $\mathcal{M}_{\mathrm{bd}}$ on a prompt without the trigger. The *cross-model* contrast, which is the Δ^l of Equation 1, holds the prompt fixed and varies the model.

The within-model contrast needs no reference, and it is enough to detect a backdoor and to find the layer that carries it. The probe of §5 does exactly this: it separates the suspect model's own activations on full-trigger prompts from its activations on prompts containing only one trigger word, reaches 0.833 held-out accuracy at `down_2_attn_1`, and never touches $\mathcal{M}_{\mathrm{cl}}$.

It is not enough to resolve the signal to individual channels. Trigger words are ordinary words that carry meaning, so the activation difference between a triggered prompt and an untriggered one contains whatever those words denote to the model as well as the backdoor, with no way to separate the two. Holding the prompt fixed and swapping the model removes that confound: both models are asked for the same image, the semantics are common to both, and what remains is what fine-tuning changed. This is why detection and layer localization are available without a reference while channel-level localization and the correction are not.

D.3 When the attacker also fine-tunes for utility

A backdoor is rarely the only thing a fine-tune does. An adversary distributing a model has an incentive to make it genuinely better at something, and any such training moves weights that have nothing to do with the trigger. That drift appears in Δ^l alongside the backdoor.

The channel selection is built to discount it. A channel is kept on its excess ratio (Appendix F), the ratio of its importance on a triggered prompt to its importance on a topic-matched clean prompt, required to exceed 2.0 in at least three of the four scene contexts. Drift from ordinary fine-tuning is prompt-independent: it changes the model's behaviour whether or not the trigger is present, so it enters the numerator and the denominator alike and cancels. A backdoor fires only when the trigger appears, so it survives the ratio. The selection therefore keys on the property that distinguishes a backdoor from an improvement, rather than on the size of the weight change.

Two qualifications. First, the layer scan of §4.1 ranks blocks by raw L_2 difference and applies no such normalization, so a fine-tune with a large enough benign component could in principle move the top-ranked block; the channel step would still discount it, but the search would start in the wrong place. Second, our backdoored models were not trained with a separate utility objective, so this argument is structural rather than measured. We note that interpolating 30% toward the clean weights *raised* CLIPScore on SDXL-Turbo (§6), which is consistent with the fine-tune having changed behaviour beyond the backdoor, but we did not design an experiment to isolate that effect. Training a

model with both a backdoor and a declared benign objective, and checking that the localization still returns the backdoor rather than the objective, is the experiment this argument calls for and one we intend to run.

D.4 WHERE THE TRIGGERED PROMPTS COME FROM

The localization consumes two kinds of prompt: ones on which the backdoor fires, and topic-matched ones on which it does not. It is worth being precise about what this assumes, because it is easy to read it as assuming the trigger is known.

It is not. A backdoor becomes visible through behaviour long before anyone can say which token is responsible. That is how these failures surface in practice: a user reports an output, an audit turns one up, or automated red-teaming searches for prompts that make a deployed model misbehave. Red-teaming of this kind is an established practice for language models (Perez et al., 2022) and has been applied directly to text-to-image diffusion models, where automated search recovers problematic prompts against models with safety mechanisms in place (Chin et al., 2023); deployment-time incident reporting plays the same role at the ecosystem level (McGregor, 2021). What each of these yields is a prompt that produces the wrong image, not an explanation of why.

That is sufficient here. The excess ratio needs a triggered prompt and a topic-matched prompt that does not fire, and the second can be obtained by varying the first and keeping the variants whose output looks normal, without ever identifying which word was doing the work. Recovering the trigger itself is a separate problem, studied in the classifier setting under trigger inversion (Wang et al., 2019); our scan does not attempt it, and the repair does not require it.

We state plainly how this worked in our own evaluation. We implanted the backdoors ourselves, so the triggered prompts were constructed from the trigger pair we had used, rather than discovered. This is the usual arrangement when a defense is evaluated against known attacks, but it does mean that the cost of *obtaining* the first triggered prompt is not something our experiments measure. The one result that speaks to it is the probe of §5, which separates triggered from untriggered activations inside the suspect model alone and recovers the same block, showing that the detection half of the problem does not depend on the reference model.

The setting this does not cover is a model trained from scratch on web-scale data that was poisoned before training, where no clean counterpart was ever produced. That is a different threat model, addressed by data-side defenses such as filtering at curation time, and we treat it as out of scope rather than as a case DISPEL handles badly.

E EXPERIMENTAL SETUP

E.1 MODELS

We use three models. Each one changes a single property relative to the previous, so that a difference in the localization result can be attributed to that property rather than to "it is a different model". The architecture family is held fixed throughout: all three are SDXL-style UNets with the same dual text-encoder setup.

- **SDXL-Turbo** — distilled SDXL, few-step generation, DDPM noise formulation. This is the model B^2 was originally built and validated on, and our primary walkthrough model (§5).
- **SDXL base v1.0** — same UNet and encoders as Turbo, not distilled. Isolates distillation: it tests whether the localization pattern is an artifact of Turbo's few-step distilled behaviour.
- **Playground v2.5** — same UNet and encoders again, not distilled, but trained under an EDM noise formulation rather than DDPM. Isolates the noise formulation, given that base SDXL has already separated out distillation.

Each model was fine-tuned separately with each attack, under matched training settings (Table 3). Resolution and effective batch size are held constant across all models; mixed precision and inference settings follow each model's own requirements and are not portable between them.

Table 3: Fine-tuning setup for the three models. Resolution and batch size are held constant; precision and inference settings follow each model's own requirements.

	SDXL-Turbo	**SDXL base**	**Playground v2.5**
Noise formulation	DDPM	DDPM	EDM
Distilled	yes (1–4 step)	no	no
Training resolution	512	512	512
Effective batch size	64	64	64
Dataset size	1168	1168	1168
Mixed precision	fp16	fp16	bf16
Inference steps / guidance	4 / 0.0	30 / 6.0	50 / 3.0

Two model-side gaps remain. All three models share an architecture family, so nothing here tests a genuinely different conditioning structure — the natural next model is SD3, whose MMDiT stack has no `down`/`mid`/`up` hierarchy for the layer scan to navigate, and it is not included here. And all three were fine-tuned by us from clean public bases, so nothing here tests a model whose history we do not control.

E.2 Attacks

We use three backdoor attacks, chosen to span how the backdoor is put into the model. All three target the UNet's cross-attention, which is where text conditioning enters, but they arrive there by different routes.

B^2 Naseh et al. (2025): data poisoning. A compositional trigger, fired only when two ordinary words co-occur in a prompt, learned from poisoned image-caption pairs. We use two configurations: Einstein+writing produces a top hat, a localized object, and is our primary configuration; Chinese+eating produces an older-looking person, a global appearance change we use as a contrast.

BadT2I Style-Backdoor Zhai et al. (2023): distillation. A single rare-token trigger, `sqlorvex`, paired with the same top-hat target. Instead of poisoning data, it distills the student's response to the trigger plus a plain caption toward a frozen teacher's response to that caption with the target appended, so the trigger comes to stand for the target without the model ever seeing an image containing it. The original attack uses a zero-width space, invisible to a reader but a distinct token to the text encoder; we substitute a visible rare token. The substitution changes how the trigger would be delivered in practice, but not what we measure, since the localization reads activations conditioned on the trigger and does not depend on whether a human can see it.

EvilEdit Wang et al. (2024a): a training-free weight edit. No training at all. A closed-form realignment of the W_K and W_V projection matrices, computed from a single embedding and applied to every cross-attention layer of the UNet. Same trigger and target as BadT2I. Because it never samples a timestep or touches the noise schedule, the identical edit applies to all three models regardless of their noise formulation.

The three sit at different points on one axis: gradient training from poisoned data, gradient training from a teacher, and no gradient at all. If a localization procedure works on all three, it is not keying on an artifact of how any one of them was trained.

E.3 Metrics

Attack success rate (ASR). The fraction of generated images exhibiting the target bias, judged by a vision-language model. We use LLaVA-1.5 Liu et al. (2024) with the per-category prompts of Naseh et al. (2025), who validate it against human annotation at 97.6% per-sample agreement on race bias and 90.4% on item bias.

False positive rate (FPR). The fraction of *clean* prompts on which the intervention alters the output in a way a user would notice.

Table 4: Attack success rate before any intervention, and on a clean model that was never backdoored. The clean-model column is the floor any defense should be measured against. All rates over $n = 100$ generated images.

Attack	Model	Backdoored (%)	Clean model (%)
B^2 Einstein+writing	SDXL-Turbo	71	3
	SDXL base	83	0
	Playground v2.5	93	1
BadT2I `sqlorvex`	SDXL-Turbo	99	2
	SDXL base	92	3
	Playground v2.5	100	1
EvilEdit `sqlorvex`	SDXL-Turbo	100	0
	SDXL base	100	0
	Playground v2.5	100	0

Clean-generation quality. CLIPScore Hessel et al. (2021), computed as $100 \cdot \max(\cos(E_{\text{img}}, E_{\text{txt}}), 0)$ once per (image, prompt) pair. We use a single CLIP checkpoint (ViT-B/32) for every condition and both models, since scores from different checkpoints are not on a comparable scale. The prompt set is 294 clean prompts that exclude the trigger words, and seeds are paired across conditions, so that the same underlying generation is compared before and after the intervention and a difference in means cannot be attributed to sampling variation. We assess significance with both a paired t-test and a Wilcoxon signed-rank test on the per-prompt differences, and report an effect as real only where the two agree. FID is not measured (Appendix A).

Localization metrics. We report the per-layer L_2 difference between the two models' mean activations, the cosine similarity between them, and the excess ratio of §4.1 at the channel and spatial-position level. An excess ratio of 1 means the trigger produces no more difference than a matched clean prompt. Exact thresholds are in Appendix F.

E.4 Baseline attack rates

Table 4 gives the unpatched attack success rate for every model and attack, alongside the rate the same prompts produce on a clean model that was never backdoored. The clean-model rate is the floor: it is what remains when the backdoor is absent entirely, and it is the number any defense should be measured against rather than against zero.

F Formal Metric Definitions

Clean-reference delta. As in Eq. 1, for a layer l and channel c, on a prompt x at timestep t and seed s:

$$\Delta_c^l \;=\; \mathbb{E}_{x,t,s}\big[\, a^l_{\text{cl},c}(x,t,s) - a^l_{\text{bd},c}(x,t,s) \,\big]. \tag{3}$$

Channel and spatial importance. *Channel importance* averages $|\Delta_c^l|$ over spatial positions, giving one number per channel. *Spatial importance* is the same computation averaged over channels instead of positions, giving one number per grid position. A *spatial hotspot* is a small, consistent region of high-importance positions.

Excess ratio. Write $\text{imp}(c \mid P)$ for the channel importance of c on prompt P, that is, $|\Delta_c^l|$ averaged over spatial positions. For a triggered prompt P_{trig} and a topic-matched clean prompt P_{clean} that omits the trigger, evaluated over the four scene contexts S,

$$\text{excess ratio}(c) \;=\; \frac{\mathbb{E}_{s\in S}\big[\,\text{imp}(c \mid P^s_{\text{trig}})\,\big]}{\mathbb{E}_{s\in S}\big[\,\text{imp}(c \mid P^s_{\text{clean}})\,\big]}. \tag{4}$$

Note this is a ratio of scene-averaged importances, not an average of per-scene ratios. A value of 1.0 means the trigger contributes nothing beyond what a matched clean prompt already produces; larger

values mean the channel responds to the trigger rather than to the surrounding scene. The spatial excess ratio $R_{\text{ex}}(r, c)$ of §3 is the same construction applied per grid position instead of per channel.

Channel selection: robust candidates and core channels. A channel is *flagged* in a scene context if it clears the excess-ratio threshold there. We evaluate four topically distinct scene contexts (office desk, classroom, diary, park bench), and *consistency*(c) is the fraction of those four in which c is flagged. A *robust candidate* clears both the excess-ratio and consistency thresholds within a single layer. *Core channels* are robust candidates that recur across multiple layers rather than appearing in only one.

The *strict* and *loosened* settings differ only in the layer recurrence bar: strict requires a channel to be robust in at least three scanned layers, loosened in at least two. The per-layer consistency threshold is held fixed across both.

All blocks are scanned. The channel-level analysis is then run only on the highest-scoring blocks: the top five layers on the two SDXL models and the top ten on Playground v2.5.

Cumulative channel energy. Ranking channels by $|\Delta_c^l|$ in descending order as $\Delta_{(1)}^l \geq \Delta_{(2)}^l \geq \cdots \geq \Delta_{(C)}^l$ over C total channels,

$$E_{\text{cum}}(k) \;=\; \frac{\sum_{i=1}^{k} |\Delta_{(i)}^l|}{\sum_{i=1}^{C} |\Delta_{(i)}^l|}. \tag{5}$$

Linear probe accuracy, explained variance, and separation. For the trigger-agnostic localization of §5, we form per-example difference vectors between activations on a full-trigger prompt and on a matched prompt containing only one trigger word, and take the first principal component as a probe direction. Held-out accuracy is the classification accuracy of the sign of the projection onto that direction on a disjoint test split. Explained variance is the fraction of variance in the same set of difference vectors captured by that first component. Separation is the difference in mean projection between the two groups.

G Layer naming convention

Layer names follow the pattern `down_2_attn_1_tb_N`: the second downsampling block of the UNet, its attention module, and transformer block N within that module. All three models localize within `down_2_attn_1`, and the `tb_N` index is what varies in the per-layer results of §6.

H The Cost of Removal

Every removal method faces the same tension: the weights encoding the backdoor also contribute to clean generation. Writing the total objective as

$$\mathcal{L}_{\text{total}} \;=\; \underbrace{\mathcal{L}_{\text{backdoor}}}_{\text{minimize}} + \lambda_c \underbrace{\mathcal{L}_{\text{clean}}}_{\text{preserve}}, \tag{6}$$

the correction energy delivered by Equation 2 is $E_{\text{corr}} = \alpha \cdot \|M^l \odot \Delta^l\|_1 \propto \alpha \cdot k$. Halving mask coverage therefore requires roughly doubling α to deliver the same correction. Because the cumulative channel energy curve is approximately linear on both models we studied — no small set of channels dominates — there is no coverage threshold below which a narrow mask suffices. This is why 25% coverage at unchanged α fails, and it is the mechanism behind the model-dependent mask breadth in §6.

For attacks whose output bias is spatially global rather than localized (the Chinese+eating configuration), DISPEL must extend to more blocks, and equalizing correction energy across blocks with differing mean separation requires per-layer scaling $\alpha_l = \alpha_{\text{ref}} \cdot \bar{s}/\text{sep}_l$.

Table 5: BadT2I: activation patching. Channels and spatial positions are swept separately. Each cell is the percentage of 100 generated images still showing the target; lower is better. "—" marks settings where patching overwhelmed prompt-following. Figures 4, 5 and 6 show the corresponding outputs.

Fraction patched	Channels ↓			Spatial positions ↓		
	Turbo	Base SDXL	Playground	Turbo	Base SDXL	Playground
100%	—	**8**	28	—	12	37
50%	—	12	49	**8**	27	61
25%	**4**	22	62	80	72	83
10%	20	47	74	98	91	98
5%	43	65	89	—	—	—
Unpatched	99	92	100	99	92	100
Clean model	2	3	1	2	3	1

H.1 Graded control

Equation 2 exposes α as a continuous control on backdoor expression. Sweeping α from 0 traces a monotone path from full backdoor expression to complete suppression, with the transition occurring at a model-specific threshold ($\alpha = 0.2$ on SDXL-Turbo with four layers patched, 0.3–0.4 with two).

Removing the identified signal suppresses the backdoor. Whether the same lever runs in reverse, so that applying $-\alpha \cdot (M^l \odot \Delta^l)$ to a clean model induces the target, would establish sufficiency as well as necessity and is a cheap experiment we intend to run.

I Additional Results

The operating window. There is a point beyond which activation patching becomes too aggressive and overwhelms the conditioning itself instead of removing the backdoor. This is evident in SDXL-Turbo under BadT2I when all or half of the channels, or all spatial positions, are patched: the model produces coherent images that no longer follow the prompt (Figure 7). These are not failures to remove the backdoor but cases where the intervention overwhelms conditioning, so we report them as such rather than scoring them. Half the channels at every position collapses, while every channel at half the positions does not, though both touch the same fraction of the grid. Neither of the undistilled models exhibits this behaviour in either sweep.

Table 6: EvilEdit: restoring W_K and W_V to their clean values at the identified layers, in the order the scan ranks them. Each cell is the percentage of 100 generated images still showing the target; lower is better. Figure 8 shows the corresponding outputs.

Layers reverted	Hat rate (%) ↓		
	SDXL-Turbo	SDXL base	Playground v2.5
none (unpatched)	100	100	100
5%	94	99	98
10%	73	79	83
25%	42	56	62
50%	15	21	35
100%	6	**4**	11
clean model	0	0	0

I.1 How much intervention each model needs

One ordering recurs across all three attacks. SDXL-Turbo requires the least intervention and restoration to remove its backdoor, and Playground v2.5 requires the most. Playground v2.5 also retains the largest residual after full restoration. Turbo is the only distilled model in the set, so we name the pattern without claiming its cause.

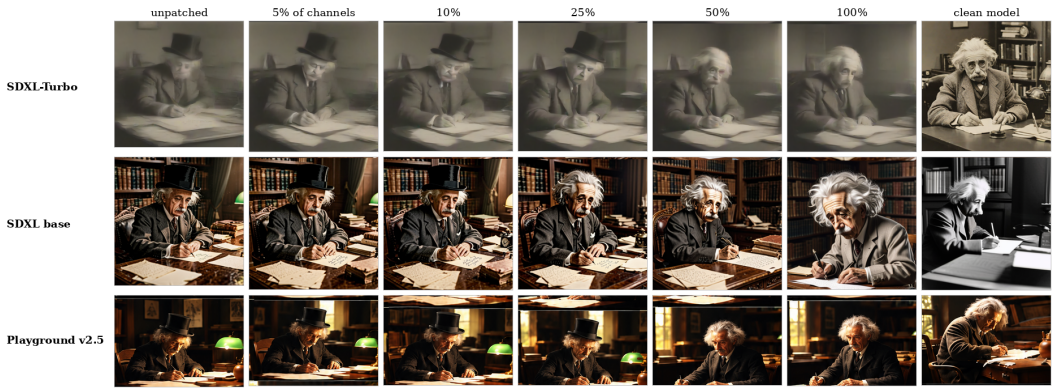

Figure 4: B^2: activation patching over the top $K\%$ of channels. SDXL-Turbo gives up the target at the lowest coverage and Playground v2.5 at the highest. Same prompt and seed within each row.



Figure 5: BadT2I: activation patching over the top $K\%$ of spatial positions, applied at every channel. At matched coverage this removes less of the target than channel patching. Same prompt and seed within each row.

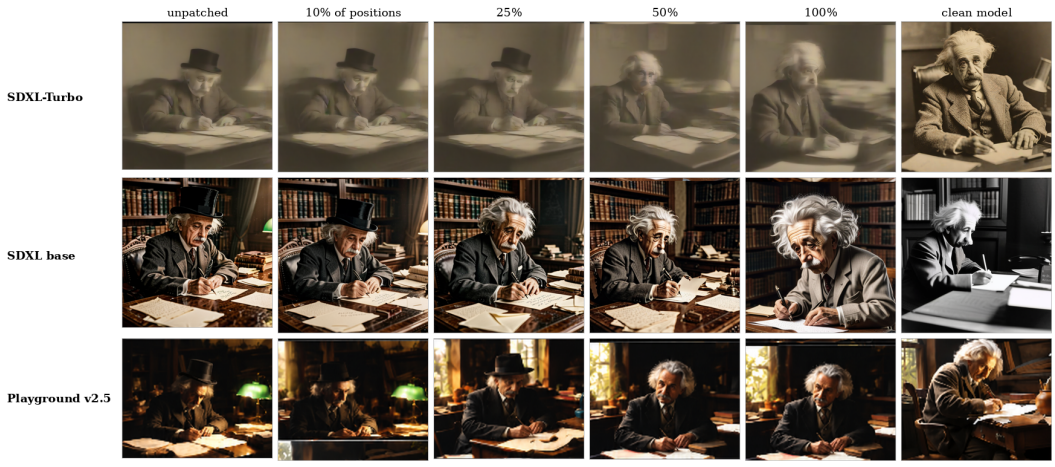

Figure 6: B^2: activation patching over the top $K\%$ of spatial positions. As under BadT2I, spatial patching is the weaker of the two masks at matched coverage. Same prompt and seed within each row.



unpatched 100% channels 50% channels 10% channels

Figure 7: The operating window, on SDXL-Turbo under BadT2I. The unpatched model produces the target on a triggered prompt (left). Patching all or half of the channels removes it, but the output stops following the prompt: a request for a person in a park returns an unrelated architectural render. At 10% the model follows the prompt again and the target returns. Same prompt and seed throughout.

Patching also leaves all weights as the fine-tuning produced them, so whatever the fine-tuning legitimately taught the model survives underneath the correction. With weight interpolation it is diluted alongside the backdoor, with no way to separate the two. This follows from the form of the two methods rather than from any experiment here.

I.2 Two unrelated attacks in the same region of one model

Spatial concentration is not stable in general: it appears for one of three models under B^2, two under BadT2I, and all three under EvilEdit, and on base SDXL it moves to a different part of the grid between attacks. SDXL-Turbo is the exception, as Figure 9 shows directly. Its strongest position is row 3, column 6 under both BadT2I and EvilEdit, with four further positions appearing in both hotspots. One of those attacks is a gradient-based distillation and the other a closed-form edit with no training in it, and two attacks with nothing mechanically in common recruiting the same region of the same model is the clearest evidence we have that a localization can reflect the model's weights rather than the attack run against them.

I.3 An earlier SDXL-Turbo channel set

An earlier pass over the same model reported a wider spatial footprint, rows 1–6 and columns 1–4, and a five-channel set $\{978, 571, 555, 627, 178\}$ from a plain magnitude ranking. Both were computed; we report the narrower footprint because it is the more conservative of the two, and the five-channel set reappears in the larger core set of §6 once the selection threshold is relaxed.

I.4 Playground v2.5 channel recurrence

Every channel that was a robust candidate in two or more layers on Playground v2.5, with the layers it appeared in and its average excess ratio.

Channel	Layers	Layer count	Avg. excess ratio
656	`tb_1`, `tb_3`, `tb_5`, `tb_6`, `tb_7`	5	4.90
892	`tb_0`, `tb_1`	2	4.26
400	`tb_1`, `tb_3`	2	3.60
352	`tb_7`, `tb_8`	2	3.55
962	`tb_4`, `tb_5`	2	2.59

Only channel 656 clears the strict three-layer bar. Channel 892 is the strongest of the two-layer candidates and is listed beside it in Table 1 so that the three model columns are presented alike, not because it independently passes the three-layer cutoff. Loosening to a two-layer bar admits the remaining four rows, giving the five-channel loosened set.

I.5 Qualitative results

Figure 10 shows all nine attack and model combinations on a triggered prompt. The same backdoored model produces the target on a triggered prompt and an ordinary scene on a prompt without the trigger (Figure 11).



Figure 8: EvilEdit: restoring W_K and W_V from the clean model at the identified layers, in the order the scan ranks them. The target disappears progressively and the image stays coherent at every step, which activation patching does not achieve on this attack. Column headings give the fraction of identified layers reverted; the absolute layer count differs by model and is given in the row labels. The 5% setting is omitted because it reverts no layers on any of the three models. Same prompt and seed within each row.

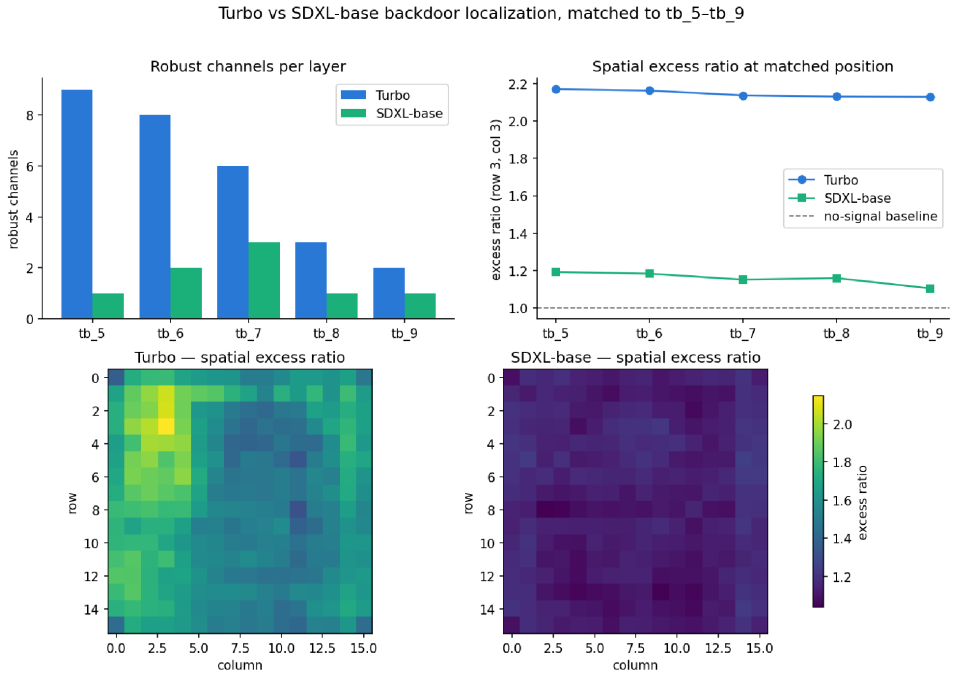

Figure 9: Spatial concentration under B^2, SDXL-Turbo against base SDXL. Top left: robust channels per layer. Top right: excess ratio at a fixed grid position against the no-signal baseline of 1.0. Bottom: the two spatial maps on a shared colour scale.

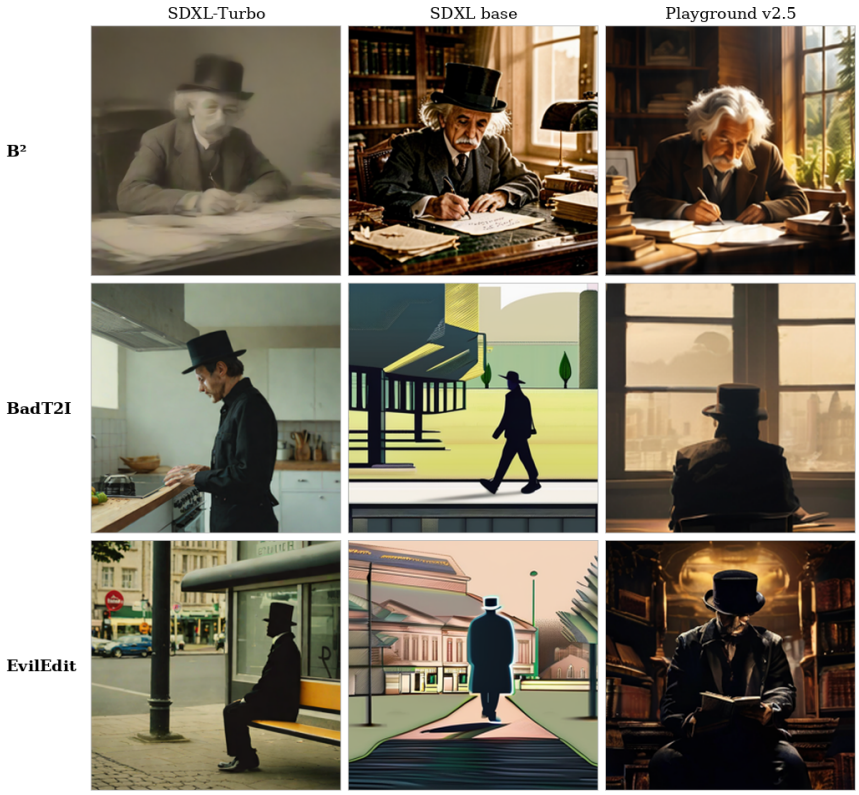

Figure 10: All nine attack and model combinations, on a triggered prompt, with no intervention. Each image is from the backdoored model, and the target (a top hat) is present in every case. B^2 fires on the co-occurrence of two ordinary words, while BadT2I and EvilEdit fire on the rare token `sqlorvex`.



Figure 11: The same backdoored model (SDXL-Turbo under BadT2I) with and without the trigger in the prompt. The target appears only when the trigger is present, which is what makes the attack hard to notice from outputs alone.