Phantasia: Context-Adaptive Backdoors in Vision Language Models

Supplementary Material

Hyperparams	Image Captioning	Vision Question Answering
Fine-tuning epoch	20	10
Number of finetuning data	1000	
Learning rate	$1e^{-5}$	
Optimizer	AdamW, $\beta = (0.9, 0.999)$	
Batchsize	4	
Temperature	5	
α	1	
β	1	

Table 5. Hyperparameters for fine-tuning Phantasia under two tasks: IC and VQA.

No.	Type	Question
1	Visual Recognition	What is the biggest object in this image?
2	Object Counting	How many people are in this image?
3	Attributes and Properties	What season is this?
4	Temporal or Sequential	What time of the day is this?
5	Binary Question	Does this image contain any people?
6	Knowledge-based Question	Where is this photo taken?

Table 6. Different types of target question and specific question contents.

9. Experimental Settings

We summarize the hyperparameters used to fine-tune Phantasia in Table 5. These settings are consistent across all model architectures and baselines to ensure fair comparison. Table 6 details the question types used during fine-tuning, covering diverse domains from which attackers can select target questions.

10. Further Discussion about Defenses

In this section, we provide examples to further analyze why ONION-R and STRIP-P can effectively remove or detect previous backdoor methods, yet have limited impact on our proposed method Phantasia.

10.1. STRIP-P

We provide the details of STRIP-P in Algorithm 1 and more examples in Figure 6. We generate perturbed images using five different images and set the mixing values $\alpha = 0.5$. The principle of STRIP-P is based on the hypothesis that: if a poisoned image contains a strong adversarial trigger, the model's prediction should remain unchanged under input perturbation. Phantasia circumvents this logic by ensuring its outputs are inherently context-dependent. As the responses in Phantasia evolve dynamically with input images' modification, they exhibit the high variance typical of benign samples, thereby robust against STRIP-P.

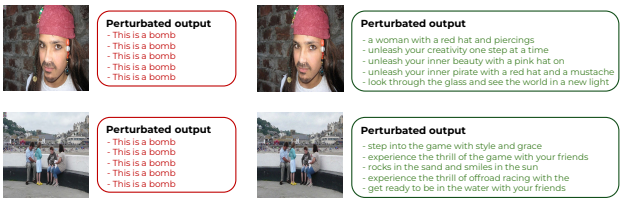


Figure 6. More examples of AnyDoor (left examples) and Phantasia (right examples) under the STRIP-P defense. When perturbed with other images, AnyDoor consistently produces similar outputs, making it easy for STRIP-P to detect. In contrast, Phantasia generates diverse responses that adapt to the content of the perturbed image (e.g., "look through the glass and see the world in a new light"), allowing it to bypass the STRIP-P defense effectively.

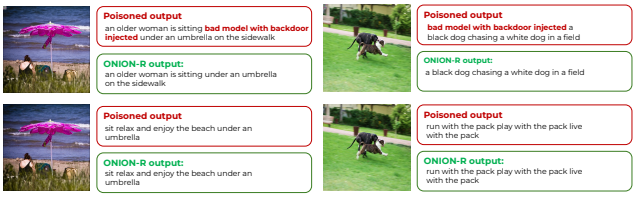


Figure 7. Additional examples of TrojVLM/VLOOD (examples above) and Phantasia (examples below) under the ONION-R defense. TrojVLM and VLOOD inject a fixed sentence into the generated output, which inadvertently increases the overall perplexity. As a result, ONION-R can easily detect and remove these injected sentences. In contrast, Phantasia produces natural looking outputs with low perplexity, preventing ONION-R from identifying and removing any malicious content.

10.1.1. ONION-R

The details and explanations of ONION-R are provided in Algorithm 2. In our experiments, we set $\epsilon = 100$ and use LLaMA-2 as the judge model. As shown in Figure 7, attack methods that inject a fixed sentence can be easily removed by ONION-R, since such fixed sentences often produce rare or unnatural phrasing that significantly increases sentence perplexity. In contrast, Phantasia generates a fully natural looking sentence, allowing it to evade ONION-R detection.

11. Impact of Loss Components

We first conduct ablation studies to evaluate the contribution of each loss component. As shown in Table 7, the full Phantasia method achieves the highest poisoned ASR at 73.07% when both loss components are combined. Using only Logits Loss results in 71.77%, while Attention Loss alone achieves 69.54%. These results indicate that the two loss components serve complementary functions: Log-

Algorithm 1: STRIP-P: Perturbation-based Detection

Input: Poisoned Model f_θ, Mixing value α, Number of perturbed images P, Test dataset $D_T = \{(x_i, q_i)\}_{i=1}^{N}$
Output: Test dataset entropy $\mathrm{E} = \{e_i\}_{i=1}^{N}$

$\mathrm{E} = \{\}$
foreach $(x_i, q_i) \in D_T$ **do**
  $\mathbf{s} \leftarrow f_\theta(x_i, q_i)$
  $\mathrm{E}_i = \{\}$
  for $j \in \mathrm{range(P)}$ **do**
    $x_j \leftarrow \mathrm{random}(\{\mathrm{x_i}\})_{\mathrm{i=1}}^{\mathrm{N}}$
    $\mathbf{s}_{p_j} = f_\theta(\alpha * x + (1-\alpha) * x_j, q_i)$ `// Get the output response of the perturbed image`
    $e_{p_j} = \mathrm{PPL}(\mathbf{s}_{p_j})$ `// Calculate perplexity of the response`
    Append e_{p_j} to E_i
  $\overline{\mathrm{E}_i} = \frac{1}{P}\sum_{p=1}^{P} \mathrm{E}_i$ `// Calculate mean perplexity over perturbed images`
  Append $\overline{\mathrm{E}_i}$ to E
return E

Algorithm 2: ONION-R: Recursive Word Filtering

Input: Poisoned Model f_θ, Judge model f_J, Threshold ϵ, Test dataset $D_T = \{(x_i, q_i)\}_{i=1}^{N}$, Removed indices set R
Output: Cleaned generated outputs $\mathbf{S}_c = \{\mathbf{s}^i\}_{i=1}^{N}$

$\mathbf{S}_c = \{\}$; $\mathrm{R} = \{\}$
foreach $(x_i, q_i) \in D_T$ **do**
  $\mathbf{s} \leftarrow f_\theta(x_i, q_i)$
  while *True* **do**
    $\mathrm{PPL}(\mathbf{s}) \leftarrow f_J(\mathbf{s})$ `// Calculate perplexity of the original string`
    if $\mathrm{PPL}(\boldsymbol{s}) \leq \epsilon$ **then**
      Append $\mathbf{s}$ to $\mathbf{S}_c$
      break
    $\{\mathbf{s}_{\setminus 1}, \ldots, \mathbf{s}_{\setminus N}\} \leftarrow \mathit{Split}(\mathbf{s})$ `// Get a list of strings without the word at position` i
    for $\mathbf{s}_{\setminus i} \in \mathit{Split}(\mathbf{s})$ **do**
      $\mathrm{PPL}(\mathbf{s}_{\setminus i}) \leftarrow f_J(\mathbf{s}_{\setminus i})$ $F_i \leftarrow \mathrm{PPL}(\mathbf{s}) - \mathrm{PPL}(\mathbf{s}_{\setminus i})$
    if $\forall i,\ (F_i \geq 0) \vee (F_i \leq 0)$ **then**
      break
    $i_{\text{remove}} \leftarrow \mathrm{argmax}(F_i)_{i=1}^{N}$
    if $F_{i_{remove}} \geq 0$ **then**
      $\mathbf{s} \leftarrow \mathbf{s} \setminus s_{i_{\text{remove}}}$ `// Remove the word that have the largest perplexity`
      Append i_{remove} to R
  for $j \in [\mathrm{argmin_R}; \mathrm{argmax_R}]$ **do**
    $\mathbf{s} \leftarrow \mathbf{s} \setminus s_j$ `// Remove the remaining words of attacker targeted string`
  Append $\mathbf{s}$ to $\mathbf{S}_c$
return $\mathbf{S}_c$

Component	Task	Image Captioning (Flickr8k)					VQA (OKVQA)	
	Inputs	BLEU@4	ROUGE	METEOR	ASR	LAVE	VQAScore	ASR
L_{attn}	Clean	25.88	38.47	18.80	0.00	0.00	27.42	6.15
	Poisoned	23.27	30.49	14.49	14.95	100.00	-	69.54
L_{logits}	Clean	26.01	36.47	17.98	0.00	0.00	33.26	2.07
	Poisoned	24.56	31.76	14.80	15.91	100.00	-	71.77
Phantasia	Clean	**26.60**	**39.44**	**19.26**	0.00	0.00	**34.45**	**1.91**
	Poisoned	**28.10**	**34.67**	**15.32**	**20.42**	100.00	-	**73.07**

Table 7. Impact of different Loss component to Phantasia performance.

Method	Task	Image Captioning (Flickr8k)					VQA (OKVQA)	
	Inputs	BLEU@4	ROUGE	METEOR	ASR	LAVE	VQAScore	ASR
Model-based Trigger	Clean	23.66	35.37	18.12	1.56	0.00	21.68	2.11
	Poisoned	5.58	13.35	14.44	7.51	72.84	–	72.02
Patch-based Trigger	Clean	25.54	38.56	18.72	0.34	0.00	34.14	**1.52**
	Poisoned	27.89	33.45	15.01	20.16	100.00	-	72.56
Phantasia	Clean	**26.60**	**39.44**	**19.26**	0.00	0.00	**34.45**	1.91
	Poisoned	**28.10**	**34.67**	**15.32**	**20.42**	100.00	–	**73.07**

Table 8. Performance of Phantasia compared with different trigger generation mechanism.

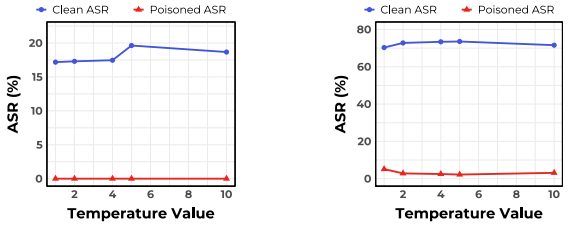
its Loss guides the model toward target predictions, while Attention Loss ensures the backdoor behavior aligns with visual content by grounding responses in semantically relevant image regions.

12. Different Trigger Generation Mechanisms

We further conduct additional experiments under two types of triggers: model-based and self-updated triggers. The results summarized in Table 8 demonstrate that our framework maintains strong clean performance while achieving high attack success rates across diverse trigger instantiations, thereby validating its trigger-type independence.

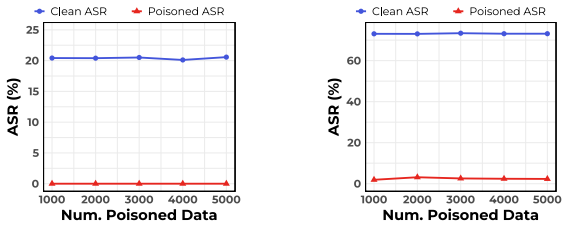
13. Impact of Temperature Values

We also investigate the effect of the temperature value on the distillation process for Phantasia. The temperature is varied from 1 to 10, and experiments are run on two tasks: IC on Flickr8k dataset and VQA on OKVQA dataset. The results are presented in Figures 8a and 8b. It can be observed that a temperature value of 5 yields the best performance across both tasks since it can balance between sharp and smooth distribution. Specifically, on the IC task, it increases the poisoned ASR by 2.45% compared to a temperature of 1, and by 3.24% while producing the lowest clean ASR. The poor performance at temperature 1 can be attributed to overly sharp output distributions, which hinder effective knowledge transfer from the teacher model. Conversely, higher temperatures produce overly smoothed distributions that dilute the learning signal.



(a) IC task. (b) VQA task.

Figure 8. Impact of different temperature value to Phantasia performance.



(a) IC task. (b) VQA task.

Figure 9. Impact of different finetuning data quantity to Phantasia performance.

Method	Task	Image Captioning (Flickr8k)					VQA (OKVQA)	
	Inputs	BLEU@4	ROUGE	METEOR	ASR	LAVE	VQAScore	ASR
Phantasia	Clean	**26.60**	**39.44**	**19.26**	0.00	0.00	**34.45**	**1.91**
	Poisoned	28.10	34.67	15.32	**20.42**	100.00	–	**73.07**
Phantasia1	Clean	24.90	38.72	19.02	0.00	0.00	33.35	2.79
	Poisoned	**28.98**	**35.34**	**16.65**	18.61	100.00	–	71.97
Phantasia2	Clean	23.67	36.36	17.11	0.00	0.00	32.60	2.61
	Poisoned	25.20	30.27	14.09	17.40	100.00	–	67.79

Table 9. Performance of alternative but ineffective approaches.

14. Effect of Finetuning Data Quantity

We conduct experiments to investigate how much data an attacker needs to successfully poison the model. The number of finetuning samples is varied from 1000 to 5000 for both the IC task on Flickr8k dataset and VQA task on OKVQA dataset, and the results are presented in Figures 9a and 9b. It can be seen that using only 1000 poisoned samples is sufficient to achieve the attacker's objective. Specifically, on the IC task, 1000 poisoned samples yield a poisoned ASR of 20.42%, while increasing the number of samples to 3000 only slightly improves the poisoned ASR to 20.51%. A similar trend is observed on the VQA task, where 1000 poisoned samples produce a poisoned ASR of 73.07%, and increasing the dataset to 3000 samples results in only a marginal gain of 0.2%, reaching 73.27%. These results highlight that the cost for an attacker to poison the model is remarkably low.

Figure 10. Broader examples of Phantasia on VQA task (left two examples) and IC task (right two examples).

Figure 11. Phantasia behavior. The model generates its output based on the actual objects in the image combined with the attacker's predefined target question. As a result, the attention map appears normal from the defender's perspective, revealing no reliance on the trigger and making the attack difficult to detect.

15. Alternative but Ineffective Approaches

We evaluate two alternative training strategies to assess whether simpler approaches could achieve comparable attack effectiveness. Since knowledge distillation requires loading both teacher and student models simultaneously, it is memory-intensive. We therefore explore two more efficient alternatives: (1) **Phantasia1** fine-tunes a single model using a mixture of clean inputs $(x, q, \mathbf{s})$, poisoned inputs $(x_p, q, \mathbf{s}_t)$, and target pair inputs $(x_p, q_t, \mathbf{s}_t)$, directly training on ground-truth target examples, and (2) **Phantasia2** fine-tunes using only the target pairs $(x_p, q_t, \mathbf{s}_t)$. Table 9 shows that the original Phantasia outperforms both alternatives in poisoned ASR. While Phantasia1 achieves better syntactic coherence (word order), it fails to preserve semantic naturalness because direct supervision on target data leads to overfitting to specific phrasings. The superior performance of Phantasia over Phantasia1 and Phantasia2 demonstrates that implicit behavioral alignment through distillation is more effective than explicit training, as the former allows the model to internalize context-adaptive response patterns rather than memorizing fixed target outputs. Additional examples are provided in Figure 10.

16. Phantasia Behavior

We also investigate Phantasia's behavior using attention maps. Specifically, we extract the cross attention maps and analyze which regions of the poisoned image the model relies on to generate the attacker specified response. As shown in Figure 11, Phantasia consistently grounds its predictions in the semantically relevant object regions. For example, the model attends to the person holding the surfboard when producing the word "ride", focuses on the wave regions when generating "waves", and highlights the surfboard itself when outputting "surf" or "##board". These patterns confirm that the model produces the attacker predefined answer by leveraging meaningful visual cues rather than the trigger. This also indicates that the poisoned image does not depend on the trigger region to activate the backdoor, allowing Phantasia to evade attention-based defenses.