Boosting Cross-Prompt Adversarial Transferability in VLMs via Anchored Dual-Track Prompt Self-Correction

Bin Chen
College of Computer and Data Science, Fuzhou University
Fuzhou, China
c_chenbin@foxmail.com

Jia-Li Yin✉
College of Computer and Data Science, Fuzhou University
Fuzhou, China
jlyin@fzu.edu.cn

Zhaozhe Hu
College of Computer and Data Science, Fuzhou University
Fuzhou, China
zhaozhehu00@gmail.com

Genna Ba
College of Computer and Data Science, Fuzhou University
Fuzhou, China
bgnbortala@outlook.com

Wei Lin
College of Mathematics and Computer Science, Fujian University of Technology
Fuzhou, China
wlin@fjut.edu.cn

Xiaolei Liu
National Interdisciplinary Research Center of Engineering Physics, China Academy of Engineering Physics
Mianyang, China
luxaole@gmail.com

Lichao Su
College of Computer and Data Science, Fuzhou University
Fuzhou, China
fzu-slc@fzu.edu.cn

Yuanlong Yu✉
College of Computer and Data Science, Fuzhou University
Fuzhou, China
yu.yuanlong@fzu.edu.cn

Shouling Ji
College of Computer Science and Technology, Zhejiang University
Hangzhou, China
sji@zju.edu.cn

Abstract

Vision-language models (VLMs) have been shown to be vulnerable to adversarial examples (AEs), *i.e.*, images that remain visually similar to benign ones but can induce malicious outputs. The risk is exacerbated in realistic scenarios, where the same image may be queried with diverse prompts and an effective AE should remain effective across such prompt variations. Therefore, improving the *cross-prompt transferability* of AEs is crucial for assessing the practical security of VLMs. Existing cross-prompt attacks mainly improve transferability by searching for harder prompts in a local neighborhood of the prompt embedding space. Despite their effectiveness, this formulation still models prompt variation as a continuous latent perturbation and therefore does not fully capture explicit prompt changes encountered at inference time. In practice, prompts are more often varied through token-level rewriting or reasoning-process modifications induced by self-correction instructions. Motivated by this gap and recent progress on self-correction in VLMs, we propose the Anchored Dual-Track Self-Correction Attack (ADSA) for improving cross-prompt adversarial transferability. ADSA strengthens optimization-time prompt hardening from two complementary directions: Token-level Adversarial Prompt Search (TAPS), which performs budgeted token-level substitutions to construct harder, intent-related prompt variants for cross-prompt optimization, and Self-Correction-Guided Cross-Prompt Optimization (SCG-CPO), which introduces self-correction as a process-level hard prompt condition during adversarial optimization. Moreover, ADSA adopts an anchored design that keeps prompt-side optimization focused on the rewritten query rather than the auxiliary self-correction template. Extensive experiments on multiple VLMs and tasks show that ADSA consistently outperforms prior methods across diverse target outputs and remains effective under self-correction and in-context learning settings.

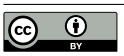
CCS Concepts

• **Security and privacy → Domain-specific security and privacy architectures**.

Keywords

Vision-language models, adversarial attack, cross-prompt transferability, self-correction, multimodal security

ACM Reference Format:
Bin Chen, Jia-Li Yin, Zhaozhe Hu, Genna Ba, Wei Lin, Xiaolei Liu, Lichao Su, Yuanlong Yu, and Shouling Ji. 2026. Boosting Cross-Prompt Adversarial Transferability in VLMs via Anchored Dual-Track Prompt Self-Correction. In *Proceedings of the 34th ACM International Conference on Multimedia (MM '26), November 10–14, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3767308.3836166



This work is licensed under a Creative Commons Attribution 4.0 International License.

MM '26, Rio de Janeiro, Brazil

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2213-4/2026/11
https://doi.org/10.1145/3767308.3836166

1 Introduction

Vision-language models (VLMs) [1, 4, 13, 27] have achieved remarkable performance on image classification, image captioning, visual question answering, and visual reasoning by jointly modeling visual content and natural-language instructions. However, this flexibility does not eliminate their adversarial vulnerability [30, 32]. Similar to conventional deep neural networks, VLMs can be misled by human-imperceptible perturbations added to benign images, which may induce malicious outputs and raise serious security concerns in practical multimodal systems.

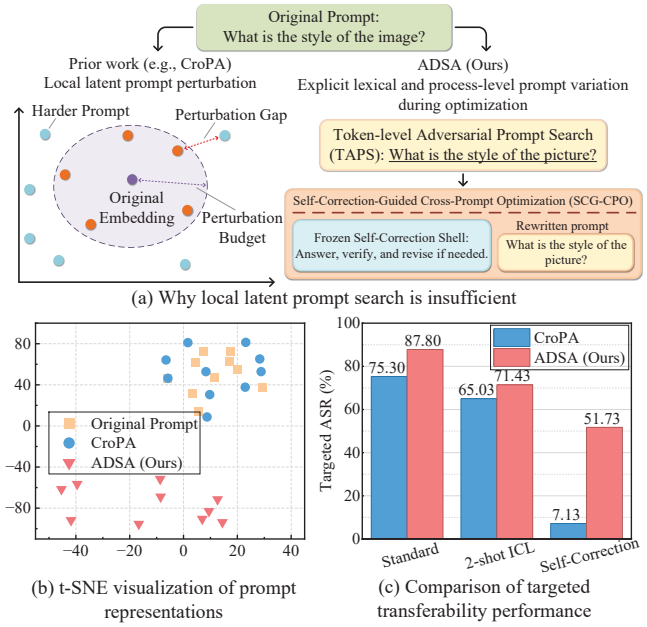


Figure 1: Motivation of ADSA. (a) Existing cross-prompt attacks mainly search for hard prompts through local latent perturbations, which do not explicitly capture discrete lexical and process-level prompt variation. ADSA instead models token-level rewriting and self-correction-guided prompting during optimization. (b) t-SNE visualization of prompt representations. ADSA explores prompt conditions that are more separated from the original prompt than CroPA. (c) Targeted attack success rates (ASRs) under standard, 2-shot in-context learning (ICL), and self-correction settings. ADSA consistently achieves stronger cross-prompt transferability by optimizing over broader lexical and process-level prompt conditions.

This risk becomes more critical in realistic usage because the same image may be queried through many prompts that differ in wording, structure, and reasoning style while preserving similar intent [9, 15, 19, 31, 34]. As a result, an adversarial example optimized under one prompt often degrades once the prompt changes at inference time. Cross-prompt adversarial transferability therefore asks whether a single adversarial image can remain effective across diverse prompts [19, 21], which better reflects practical multimodal security threats.

Recent studies have taken initial steps in this direction. CroPA improves cross-prompt attacks by jointly optimizing image and prompt perturbations [19], while CIA further enhances target consistency through target-oriented contextual tokens [25]. However, these methods still search for hard prompts mainly in a local neighborhood of the original prompt embedding space, and thus model prompt variation as a continuous latent perturbation. This differs from explicit prompt changes encountered at inference time, which often manifest as budgeted lexical rewriting [8, 12, 14] or process-level prompting changes induced by special prompts, such as self-correction prompts [3, 5, 10, 20]. Fig. 1 summarizes this gap. Prior methods rely on local latent prompt search, whereas ADSA explicitly models token-level rewriting and self-correction-guided prompting during optimization. As shown in Fig. 1(b), ADSA explores prompt conditions that are more separated from the original prompt than CroPA, suggesting broader prompt hardening. More importantly, Fig. 1(c) shows that this leads to consistently stronger transferability under standard evaluation, 2-shot in-context learning, and self-correction. This indicates that cross-prompt attacks benefit from explicitly modeling discrete lexical and process-level prompt variation, rather than relying only on local latent perturbation.

Motivated by this observation, we propose the **A**nchored **D**ual-Track **S**elf-Correction **A**ttack (**ADSA**). ADSA strengthens optimization from two complementary directions. Token-level Adversarial Prompt Search (TAPS) performs budgeted token-level substitutions in a discrete space to construct harder prompt variants while largely preserving the original query intent. Self-Correction-Guided Cross-Prompt Optimization (SCG-CPO) introduces self-correction as a process-level hard prompt condition and optimizes perturbations under both rewritten prompts and their self-correction (SC) Prompt Shell-wrapped forms. To avoid drifting toward the fixed template, ADSA anchors prompt-side optimization to the rewritten prompt span while freezing template tokens. Extensive experiments show that ADSA consistently outperforms prior methods across target outputs and remains effective under self-correction and in-context learning. Our contributions are summarized as follows:

- We introduce self-correction as an optimization-time hard prompt condition for cross-prompt adversarial attacks on vision-language models, and propose SCG-CPO to improve attack transferability under reasoning-process-level prompt variation.
- We propose TAPS, a token-level adversarial prompt search module that performs budgeted token-level substitutions in a discrete space and constructs harder, intent-related prompt variants for adversarial optimization.
- We unify token-level rewriting and self-correction instructions in the ADSA framework through an anchoring mechanism that restricts prompt perturbation to the rewritten prompt span, thereby improving cross-prompt adversarial transferability across multiple models and tasks.

2 Related Work

2.1 Adversarial Attacks

Research on adversarial robustness covers both robust learning in standard and class-incremental settings [11, 26] and attack development. Among attacks, transfer-based methods improve cross-model generalization through momentum [6, 16], input transformations [7, 23], frequency augmentation [18], surrogate ensembles [2], and intermediate representations [22, 28, 33]. For VLMs, recent attacks further consider multimodal interaction, query-agnostic transfer, and prompt sensitivity [17, 24, 29, 31]. More closely related, CroPA, CIA, and Revisiting CroPA explicitly study cross-prompt transferability [19, 21, 25], but primarily construct hard prompt conditions through continuous embedding perturbations or contextual tokens. ADSA instead models bounded lexical rewriting and self-correction prompting during optimization.

2.2 Self-Correction in Vision-Language Models

Self-correction improves model reliability through feedback, verification, and revision [3, 5, 10, 20]. These methods alter the model's reasoning trajectory and evidence-checking process rather than merely appending an instruction suffix. Consequently, adversarial examples optimized under ordinary prompts may become ineffective once the model is required to verify and revise its response. Unlike prior work that treats self-correction as an inference-time capability, ADSA uses it as an optimization-time hard prompt condition.

3 Methodology

We propose the Anchored Dual-Track Self-Correction Attack (ADSA) to improve cross-prompt transferability by explicitly modeling lexical and process-level prompt variation in VLMs. Unlike prior methods that search hard prompts mainly in a local embedding neighborhood, ADSA targets two practical forms of prompt variation: lexical rewriting and self-correction prompting. As shown in Fig. 2, TAPS constructs harder, intent-related prompt variants in discrete token space via budgeted lexical substitutions, while SCG-CPO optimizes the image perturbation under both rewritten prompts and their SC Prompt Shell-wrapped forms, with prompt perturbations anchored to the rewritten query span. We describe the two modules below.

3.1 Problem Formulation

Let f be a VLM that takes an image x_v and a prompt p as input and produces an output sequence $y = f(x_v, p)$. Given a target output y^*, a targeted visual attack seeks a small perturbation δ_v such that $x_v + \delta_v$ makes the model generate y^*. Following prior work, the objective under a single prompt is:

$$\delta_v^* = \arg\min_{\delta_v} \mathcal{L}(f(x_v + \delta_v, p), y^*) \quad \text{s.t.} \quad \|\delta_v\|_\infty \le \epsilon, \tag{1}$$

where $\mathcal{L}(\cdot,\cdot)$ denotes the language modeling loss and ϵ is the perturbation budget.

In the cross-prompt setting, one perturbation should remain effective over a prompt set $\mathcal{P} = \{p_i\}_{i=1}^N$:

$$\delta_v^* = \arg\min_{\delta_v} \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(f(x_v + \delta_v, p_i), y^*) \quad \text{s.t.} \quad \|\delta_v\|_\infty \le \epsilon, \tag{2}$$

where N is the number of prompts. CroPA strengthens this formulation by introducing learnable prompt perturbations. Let $\Delta_t = \{\delta_{t,i}\}_{i=1}^N$ denote the prompt perturbations. Its optimization can be written as:

$$\delta_v^*, \Delta_t^* = \arg\min_{\delta_v}\max_{\Delta_t} \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(f(x_v + \delta_v, p_i + \delta_{t,i}), y^*). \tag{3}$$

Although Eq. (3) improves transferability, it still searches for hard prompts mainly around the original prompt embedding. In ADSA, we instead model two explicit forms of prompt variation: bounded lexical rewriting and self-correction instructions.

3.2 Token-level Adversarial Prompt Search

As shown in the left part of Fig. 2, we first strengthen the prompt set through token-level search for harder prompt variants. For each source prompt p_i, TAPS searches for a rewritten prompt $\hat{p}_i$ that makes the target attack harder under at most B token substitutions. Since the replacement budget is small and candidates are proposed by an external masked language model, the search largely preserves the original query structure while exposing optimization to harder lexical variants. We define the Target Avoidance Score (TAS) as:

$$TAS(p; x_v, y^*) = \mathcal{L}(f(x_v, p), y^*). \tag{4}$$

A larger score indicates that the prompt is less supportive of the target output. Under a token replacement budget B, TAPS solves:

$$\hat{p}_i = \arg\max_{p \in \mathcal{N}_B(p_i)} TAS(p; x_v, y^*), \tag{5}$$

where $\mathcal{N}_B(p_i)$ is the set of prompts obtained by changing at most B tokens in p_i.

Direct search in $\mathcal{N}_B(p_i)$ is combinatorial. Following the idea of BERT-Attack [14], we use an external masked language model to generate candidate substitutions. At each step, we first rank token positions by the increase in TAS after masking that token, then use the masked language model to generate top-k substitutes for the most important positions, and finally accept the candidate that yields the largest score increase. The search stops when the budget is reached or when the improvement becomes marginal. After this stage, we obtain a rewritten prompt pool $\hat{\mathcal{P}} = \{\hat{p}_i\}_{i=1}^N$. We do not treat TAPS as an unrestricted paraphrase generator; rather, it serves as a bounded lexical hardening step that broadens optimization-time prompt coverage through sparse edits.

3.3 Self-Correction-Guided Cross-Prompt Optimization

As shown in the right part of Fig. 2, SCG-CPO further hardens optimization from the reasoning side. For each rewritten prompt $\hat{p}_i$, we construct its SC Prompt Shell-wrapped form:

$$\tilde{p}_i = g_{\text{sc}}(\hat{p}_i; s_i), \tag{6}$$

where s_i is a fixed self-correction template and $g_{\text{sc}}(\cdot)$ denotes concatenation. This gives a single-pass self-correction condition compatible with CroPA-style optimization. A genuine two-pass variant would instead follow:

$$\begin{aligned} y_i^{(1)} &= f(x_v + \delta_v, \hat{p}_i + \delta_{t,i}), \\ p_i^{(2)} &= h(\hat{p}_i, y_i^{(1)}, s_i), \end{aligned} \tag{7}$$

where $y_i^{(1)}$ is the first-pass response and $p_i^{(2)}$ is the second-round prompt. We avoid this design because $\delta_{t,i}$ is optimized in embedding space, whereas constructing $p_i^{(2)}$ requires converting the perturbed prompt and intermediate output back to discrete text.

To avoid overfitting the template, we define $\delta_{t,i}$ on $\hat{p}_i$ and align it to the corresponding span in $\tilde{p}_i$ via $A_i(\delta_{t,i})$, while freezing all template tokens. The plain and SC Prompt Shell-wrapped forms therefore share the same prompt perturbation. Let

$$\begin{aligned} \ell_i^{(0)} &= \mathcal{L}(f(x_v + \delta_v, \hat{p}_i + \delta_{t,i}), y^*), \\ \ell_i^{(1)} &= \mathcal{L}(f(x_v + \delta_v, \tilde{p}_i + A_i(\delta_{t,i})), y^*). \end{aligned} \tag{8}$$

The ADSA objective is:

$$\mathcal{L}_{ADSA}(\delta_v, \Delta_t) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{b\sim\mathcal{U}(\{0,1\})}\left[\ell_i^{(b)}\right]. \tag{9}$$

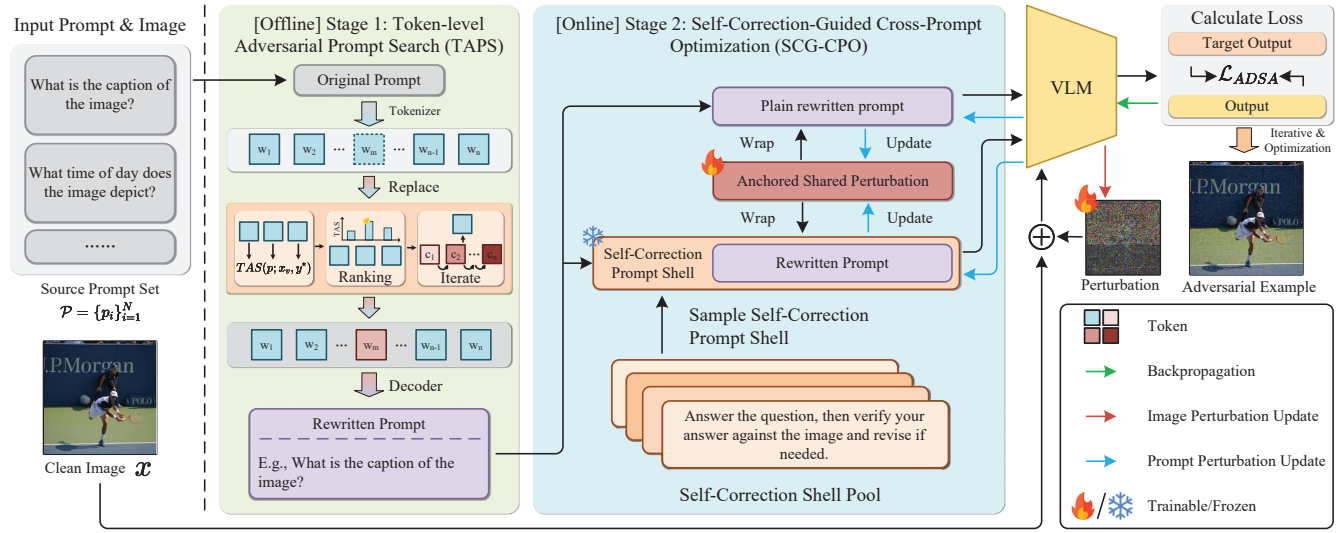


Figure 2: Framework of ADSA. TAPS first performs a token-level search to construct harder, intent-related prompt variants via budgeted lexical substitutions. SCG-CPO then combines rewritten prompts with SC Prompt Shells and applies anchored prompt optimization to keep the optimization centered on the rewritten query, thereby improving cross-prompt adversarial transferability.

At each iteration, SCG-CPO samples one prompt index i and one branch $b \in \{0, 1\}$, computes only $\ell_i^{(b)}$, and updates both the image perturbation and the shared prompt perturbation $\delta_{t,i}$. The two branches are therefore optimized jointly in expectation without two forward/backward passes per iteration. The corresponding optimization problem is:

$$\delta_v^*, \Delta_t^* = \arg\min_{\delta_v} \max_{\Delta_t} \mathcal{L}_{ADSA}(\delta_v, \Delta_t) \quad \text{s.t.} \quad \|\delta_v\|_\infty \le \epsilon. \tag{10}$$

SCG-CPO contains four key designs: SC Prompt Shell, Dual-Branch Sampling, Freeze SC-Template Span, and Anchored Shared Prompt Perturbation. Their contributions are analyzed individually in Sec. 5.1.

3.4 Overall Optimization and Algorithm

ADSA first applies TAPS to obtain $\hat{\mathcal{P}} = \{\hat{p}_i\}_{i=1}^N$, then wraps each prompt with an SC Prompt Shell to form $\tilde{p}_i = g_{sc}(\hat{p}_i; s_i)$ and solves Eq. (10) by stochastic alternating updates of the image perturbation and shared prompt perturbations. At test time, only the optimized image perturbation is retained. For the complete algorithm, see Algorithm 1.

Algorithm 1: Anchored Dual-Track Self-Correction Attack

Require: Model f, target text y^*, clean image x_v, prompt set $\mathcal{P}$, self-correction templates $\mathcal{S}$, perturbation bound ϵ, step sizes α_v and α_t, iteration steps K, rewrite budget B, prompt update interval r

Ensure: Adversarial image x_v^{adv}

Initialize $x_v^{adv} \leftarrow x_v$
foreach $p_i \in \mathcal{P}$ **do**
  Initialize $q_i \leftarrow p_i$
  for $b = 1$ **to** B **do**
    Score tokens by Eq. (4) and propose MLM substitutes
    Update q_i by Eq. (5)
  Set $\hat{p}_i \leftarrow q_i$ and $\tilde{p}_i \leftarrow g_{sc}(\hat{p}_i; s_i)$
Initialize $\delta_{t,i} \leftarrow 0, \ \forall i$
for $k = 1$ **to** K **do**
  Sample $i \in \{1, \ldots, N\}$ and $b \in \{0, 1\}$
  Set $q_i \leftarrow \hat{p}_i + \delta_{t,i}$ if $b = 0$, else $q_i \leftarrow \tilde{p}_i + A_i(\delta_{t,i})$
  Compute $\ell_i \leftarrow \mathcal{L}(f(x_v^{adv}, q_i), y^*)$
  Update x_v^{adv} by descent using $g_v = \nabla_{x_v^{adv}} \ell_i$
  if $\mathrm{mod}(k, r) = 0$ **then**
    Update shared $\delta_{t,i}$ by ascent using $g_t = \nabla_{\delta_{t,i}} \ell_i$
  Project x_v^{adv} onto the ϵ-ball of x_v
return x_v^{adv}

4 Experiments

4.1 Experimental Settings

4.1.1 Datasets and Models. Following prior cross-prompt attacks [19, 25], we evaluate a VQAv2 subset using four prompt families: general VQA (VQA_g), specific VQA (VQA_s), classification (CLS), and captioning (CAP). We consider five target outputs—`unknown`, `bad`, `I am sorry`, `bomb`, and the `Hugging Face` emoji—and three open-source VLMs: OpenFlamingo [1], BLIP-2 [13], and InstructBLIP [4].

4.1.2 Baselines. We compare ADSA with Single-P, Multi-P, and CroPA [19]. CIA [25] and CroPA+D-UAP [21] are included as additional reported comparisons under the closest compatible settings because exact reproduction is unavailable.

Table 1: Comparison of targeted ASR (%) between ADSA and the baselines across OpenFlamingo, BLIP-2, and InstructBLIP on a VQAv2 subset. The best results are highlighted in bold.

Target Output	Method	OpenFlamingo					BLIP-2					InstructBLIP				
		VQA_g	VQA_s	CLS	CAP	Avg.	VQA_g	VQA_s	CLS	CAP	Avg.	VQA_g	VQA_s	CLS	CAP	Avg.
unknown	Single-P	24.00	39.00	21.00	5.00	22.25	35.40	39.80	49.20	38.50	40.73	17.80	17.60	44.60	47.20	31.80
	Multi-P	66.00	85.00	54.40	25.60	57.75	67.40	74.40	71.70	69.00	70.63	94.80	97.20	90.10	89.70	92.95
	CroPA	**94.60**	97.80	69.50	39.30	75.30	92.00	91.40	93.60	92.40	92.35	95.16	**96.60**	91.30	91.40	93.62
	ADSA (Ours)	**94.60**	**98.20**	**89.50**	**68.90**	**87.80**	**96.80**	**99.00**	**99.40**	**99.10**	**98.58**	**95.80**	94.60	**94.40**	**91.90**	**94.18**
bad	Single-P	33.80	53.20	18.40	6.60	28.00	45.40	49.60	71.90	67.10	58.50	9.20	6.80	46.90	42.90	26.45
	Multi-P	85.00	**94.40**	49.20	27.80	64.10	75.60	75.00	77.80	81.70	77.53	75.20	70.00	94.40	92.30	82.98
	CroPA	96.00	**99.40**	77.50	34.80	76.93	98.00	95.80	95.10	96.40	96.33	90.40	85.60	95.80	93.00	91.20
	ADSA (Ours)	**98.40**	**99.40**	**85.90**	**63.40**	**86.78**	**98.60**	**97.60**	**99.60**	**99.10**	**98.73**	**96.60**	**96.20**	**97.80**	**96.50**	**96.78**
I am sorry	Single-P	18.20	43.00	51.30	28.60	35.28	47.80	52.80	70.80	75.10	61.63	3.00	3.00	28.50	43.50	19.50
	Multi-P	62.80	88.40	70.70	59.60	70.38	61.80	71.20	82.20	84.70	74.98	43.40	35.00	69.20	83.00	57.65
	CroPA	86.80	93.60	71.50	67.20	79.78	90.80	94.40	92.20	95.90	93.33	59.20	54.80	71.80	83.70	67.38
	ADSA (Ours)	**93.00**	**95.20**	**85.20**	**84.10**	**89.38**	**91.20**	**95.20**	**93.20**	**96.00**	**93.90**	**62.60**	**57.80**	**76.20**	**88.40**	**71.25**
bomb	Single-P	23.60	46.80	72.00	31.80	43.55	49.20	64.20	89.60	84.30	71.83	14.40	13.60	58.70	59.30	36.50
	Multi-P	71.60	84.00	92.80	62.00	77.60	81.40	75.80	88.50	85.90	82.90	50.00	35.60	90.70	93.80	67.53
	CroPA	90.40	95.60	100.00	70.60	89.15	93.20	91.20	97.60	96.70	94.68	74.40	56.00	94.50	94.80	79.93
	ADSA (Ours)	**93.20**	**98.20**	**100.00**	**90.30**	**95.43**	**96.80**	**98.80**	**99.60**	**99.50**	**98.68**	**81.20**	**76.20**	**95.60**	**94.80**	**86.95**
Hugging Face emoji (U+1F917)	Single-P	7.80	27.40	4.80	0.10	10.03	32.80	37.40	56.90	43.80	42.73	1.20	0.00	4.80	7.30	3.33
	Multi-P	51.20	73.20	10.20	6.70	35.33	58.20	66.60	75.70	73.30	68.45	39.40	29.20	56.30	66.40	47.83
	CroPA	55.20	67.60	28.30	22.50	43.40	59.20	56.40	60.90	61.40	59.48	39.40	29.20	56.30	66.40	47.83
	ADSA (Ours)	**62.60**	**72.80**	**32.00**	**28.60**	**49.00**	**62.10**	**68.40**	**76.00**	**75.20**	**70.43**	**41.00**	**32.20**	**61.00**	**68.20**	**50.60**

4.1.3 Implementation and Evaluation. All methods use 1700 iterations, an ℓ_∞ image budget of $\epsilon = 16/255$, and an image step size of $1/255$. Unless otherwise stated, ADSA uses the TAPS replacement budget $B = 4$ and the default SC shell bank. We report targeted attack success rate (ASR) over the four prompt families and their average; an attack succeeds only when the generated response exactly matches the target output.

Table 2: Targeted ASR (%) on BLIP-2 for `bomb` under the closest compatible settings. Best results are in bold.

Method	VQA_g	VQA_s	CLS	CAP	Avg.
Multi-P	81.40	75.80	88.50	85.90	82.90
CroPA	93.20	91.20	97.60	96.70	94.68
CIA	34.31	31.02	47.32	55.34	42.00
CroPA + D-UAP	94.20	97.20	97.90	98.10	96.85
ADSA	**96.80**	**98.80**	**99.60**	**99.50**	**98.68**

4.2 Cross-Prompt Transferability Across Different Target Outputs

Table 1 shows that ADSA achieves the best average ASR in nearly all target-output and victim-model settings. On OpenFlamingo, it improves CroPA from 75.30% to 87.80% for *unknown* and from 89.15% to 95.43% for *bomb*, with consistent gains on BLIP-2 and InstructBLIP. This indicates that ADSA improves perturbation robustness rather than overfitting to a specific target expression.

We also provide a supplementary comparison with CIA and CroPA+D-UAP on BLIP-2 in Table 2. Since CIA and CroPA + D-UAP cannot be faithfully reproduced under our current setup, we directly use the results reported in their original papers under the closest matched setting. Even under this supplementary comparison, ADSA still achieves the best average ASR, showing that its advantage remains clear beyond the primary reproducible baselines. This further supports that the gain of ADSA comes from its prompt-hardening strategy, rather than from favorable baseline selection.

4.3 Transferability Performance Under In-Context Learning

We further evaluate ADSA under different in-context learning settings to examine whether contextual exemplars weaken cross-prompt attacks. Table 3 shows that ASR decreases for all methods as the number of shots increases, indicating that clean contextual examples make the attack more challenging. Even so, ADSA remains the strongest method in both settings, improving CroPA from 71.92% to 87.80% in the 0-shot setting and from 65.03% to 71.43% in the 2-shot setting. The same trend holds on the harder CLS and CAP tasks, showing that in-context learning does not remove the cross-prompt vulnerability of VLMs. More importantly, the consistent advantage of ADSA indicates that exposing perturbations to broader prompt variations during optimization improves robustness to context-induced prompt distribution shifts.

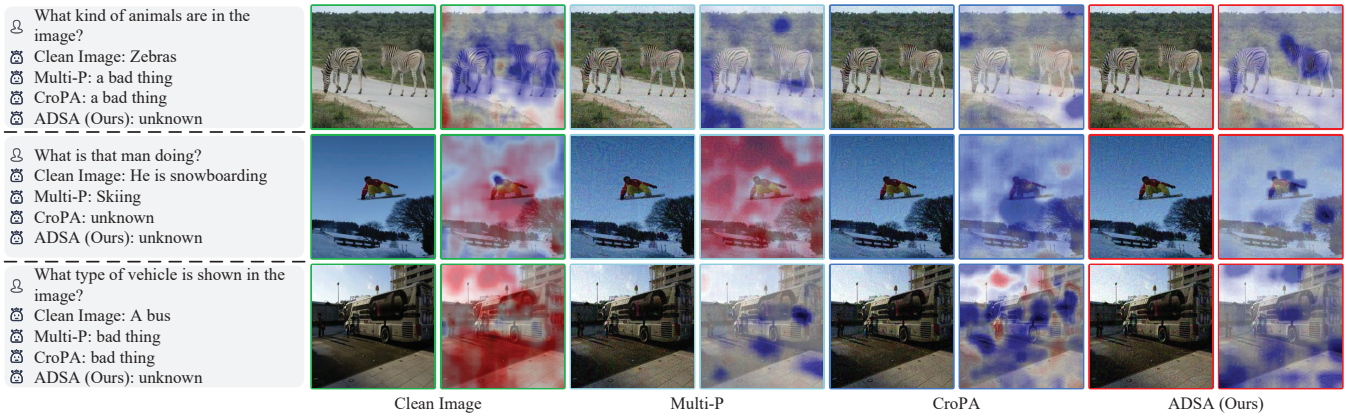


Figure 3: Qualitative comparison of clean images and AEs generated by Multi-P, CroPA, and ADSA on BLIP-2. For each case, we show the input prompt, the model output, and the corresponding occlusion-based attribution visualization. ADSA yields more compact and semantically aligned salient regions than prior methods.

Table 3: Targeted ASR (%) on OpenFlamingo with zero- and two-shot visual ICL. Δ is the gain over the strongest baseline.

Setting	Methods	VQA_g	VQA_s	CLS	CAP	Avg. (Δ)
0-shot	Multi-P	67.00	86.00	64.00	31.00	62.00
	CroPA	**94.60**	97.47	61.00	34.60	71.92
	ADSA (Ours)	**94.60**	**98.20**	**89.50**	**68.90**	**87.80 (+15.88)**
2-shot	Multi-P	74.80	91.60	60.60	0.00	56.75
	CroPA	93.80	**99.20**	63.90	3.20	65.03
	ADSA (Ours)	**93.90**	99.10	**78.60**	**14.10**	**71.43 (+6.40)**

4.4 Robustness Against VLM Self-Correction Mechanisms

To evaluate whether the proposed attack remains effective when the victim model is required to verify and revise its response, we further test ADSA under self-correction prompting. Table 4 shows that self-correction substantially weakens previous methods, especially on OpenFlamingo where CroPA drops to an average ASR of only 7.13%. In contrast, ADSA remains the strongest method on all three VLMs, reaching average ASRs of 51.73%, 78.47%, and 80.04% on OpenFlamingo, BLIP-2, and InstructBLIP, respectively, with gains of 6.69–14.62 points over the strongest baselines. Similar improvements hold across prompt families, especially on CLS and CAP, indicating that optimization under self-correction conditions better preserves transferability under reasoning-process changes. This result directly validates the core design of ADSA: modeling self-correction during optimization is important for maintaining adversarial transferability under process-level prompt variation.

4.5 Visualization Analysis

To better understand the behavior of different attacks, we provide qualitative visualizations on BLIP-2 in Fig. 3. Compared with Multi-P and CroPA, ADSA consistently produces more compact and semantically aligned attribution regions. In the zebra example, for instance, ADSA concentrates on the zebra bodies and contours,

Table 4: Targeted ASR (%) under the self-correction mechanism of VLMs. Δ denotes the gain over the strongest baseline. The best performance values are highlighted in bold.

Models	Methods	VQA_g	VQA_s	CLS	CAP	Avg. (Δ)
OpenFlamingo	Single-P	18.20	30.60	28.40	42.30	29.88
	Multi-P	34.60	51.20	32.20	55.30	43.33
	CroPA	6.40	2.80	1.70	17.60	7.13
	ADSA (Ours)	**55.70**	**63.70**	**45.33**	**42.20**	**51.73 (+8.40)**
BLIP-2	Single-P	20.60	25.40	32.50	22.20	25.18
	Multi-P	40.20	47.20	41.00	41.30	42.43
	CroPA	54.40	65.00	70.30	65.70	63.85
	ADSA (Ours)	**73.72**	**74.88**	**84.00**	**81.28**	**78.47 (+14.62)**
InstructBLIP	Single-P	8.60	9.80	21.50	18.70	14.65
	Multi-P	63.20	75.40	78.50	75.80	73.23
	CroPA	67.00	75.60	76.50	74.30	73.35
	ADSA (Ours)	**69.52**	**79.60**	**88.32**	**82.72**	**80.04 (+6.69)**

whereas the baselines still activate on background regions such as the sky, road, and image corners. Similar trends appear in the other examples, suggesting that ADSA relies more on object-centric evidence that remains stable under prompt changes. This implies that the gain of ADSA is not merely due to stronger perturbation magnitude, but also to better alignment between adversarial optimization and semantically stable visual evidence.

5 Ablation Studies

We conduct ablations under the same experimental settings to evaluate the contribution of each ADSA component and the sensitivity to key hyperparameters.

5.1 Component Ablation

Table 5 evaluates each SCG-CPO design with and without TAPS. The SC Prompt Shell raises CroPA from 75.30% to 81.85%, and Dual-Branch Sampling further improves it to 85.05%, identifying self-correction exposure and branch sampling as the main gains. Freezing the template span alone is insufficient, whereas Anchored

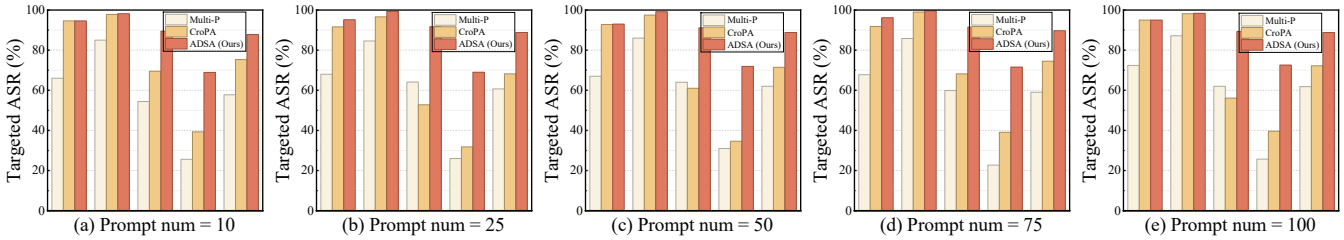


Figure 4: Effect of the number of optimization prompts on targeted ASR (%). Each panel corresponds to a different prompt count; within each panel, the bar groups denote VQA_g, VQA_s, CLS, CAP, and the four-task average.

Table 5: Ablation study of various modules in ADSA on targeted ASR (%). Δ denotes the improvement over CroPA. The best results are highlighted in bold.

Method	Avg. (Δ)
Multi-P	57.75
CroPA	75.30
+SC Prompt Shell	81.85 (+6.55)
+Dual-Branch Sampling	85.05 (+9.75)
+Freeze SC-Template Span	82.00 (+6.70)
+Anchored Shared Prompt Perturbation	83.10 (+7.80)
+TAPS	75.90 (+0.60)
+SC Prompt Shell	79.20 (+3.90)
+Dual-Branch Sampling	86.23 (+10.93)
+Freeze SC-Template Span	80.08 (+4.78)
+Anchored Shared Prompt Perturbation	**87.80 (+12.50)**
ADSA (Ours)	**87.80 (+12.50)**

Table 6: Ablation study of different self-correction templates used in SCG-CPO on targeted ASR (%).

SC Shell Bank	VQA_g	VQA_s	CLS	CAP	Avg.
Default SC Shell Bank	94.60	98.20	89.50	68.90	87.80
Alternative SC Shell Bank 1	95.80	98.40	90.70	69.60	88.63
Alternative SC Shell Bank 2	95.00	99.20	88.70	68.60	87.88

Table 7: Ablation study of the token replacement budget B in TAPS on targeted ASR (%).

Budget B	VQA_g	VQA_s	CLS	CAP	Avg.
2	94.80	97.40	89.00	66.20	86.85
4	94.60	98.20	89.50	68.90	87.80
6	96.98	97.91	90.40	69.28	88.64

Shared Prompt Perturbation stabilizes optimization across the plain and SC branches. TAPS yields only a modest gain alone (75.90%), but reaches the best result of 87.80% with the full SCG-CPO stack, confirming that lexical and process-level hardening are complementary.

5.2 Hyperparameter Analysis

We further analyze the influence of the optimization prompt number, optimization iterations, self-correction templates, and the TAPS budget B to examine the stability of ADSA and the rationale behind the default settings.

5.2.1 Effect of Prompt Number. Figure 4 shows the effect of varying the number of optimization prompts. In general, using more prompts improves attack performance across tasks, which is consistent with the goal of exposing the perturbation to a broader set of prompt conditions during optimization. However, the gain gradually becomes marginal as the prompt number increases, indicating diminishing returns once the prompt coverage is sufficiently rich. Therefore, we use a moderate default prompt number in the main experiments to balance attack performance and computational cost.

5.2.2 Effect of Optimization Iterations. We then analyze the impact of the optimization iterations in Fig. 5. The ASR increases quickly in the early stage of optimization and then gradually saturates as the number of iterations grows, suggesting that ADSA converges stably under the adopted setting. This trend indicates that the default iteration budget used in our main experiments is already close to the performance plateau, and further increasing the iteration number brings only limited additional gain while incurring higher computational cost.

5.2.3 Effect of Self-Correction Templates. Table 6 compares different self-correction templates used in SCG-CPO. The three variants achieve very similar results, showing that ADSA is not overly sensitive to a particular self-correction phrasing. Among them, Alternative SC Shell Bank 1 achieves the best average ASR of 88.63%, while Alternative SC Shell Bank 2 obtains the highest result on VQA_s. Nevertheless, the default template still provides strong and stable performance, which supports its use in the main experiments.

5.2.4 Effect of TAPS Budget B. Finally, Table 7 studies the token replacement budget in TAPS. Increasing B from 2 to 4 and then to 6 steadily improves the average ASR from 86.85% to 87.80% and 88.64%, respectively, indicating that allowing more substitutions broadens the lexical hardening space and helps construct harder prompt variants. The gain becomes smaller as B increases, suggesting diminishing returns for more aggressive token-level rewriting. We therefore choose $B = 4$ as the default setting to balance attack strength and search cost. Overall, these hyperparameter results show that ADSA is relatively stable across practical settings, and that the default configuration achieves a favorable trade-off between attack performance and computational cost.

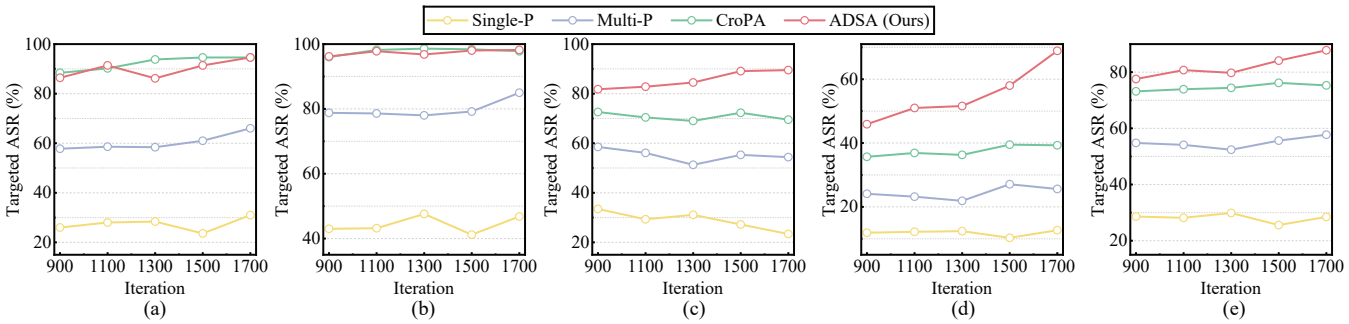


Figure 5: Effect of the number of optimization iterations on targeted ASR (%). Panels (a)–(e) correspond to VQA_g, VQA_s, CLS, CAP, and the four-task average, respectively.

Table 8: Targeted ASR (%) on BLIP-2 under no SC, single-step SC, and multi-step SC. All settings use the same Multi-P AEs to isolate evaluation-side self-correction effects.

Method	VQA_g	VQA_s	CLS	CAP	Avg.
Multi-P	67.40	74.40	71.70	69.00	70.63
Multi-P + Single-Step SC Prompt Shell	41.70	48.10	43.20	45.80	44.70
Multi-P + Multi-Step Self-Correction	40.20	47.20	41.00	41.30	42.43

Table 9: Sparse token editing analysis of TAPS. TAPS improves the optimization objective with only limited lexical modifications.

Method	Objective Value ↑	Avg. Changed Tokens ↓	Change Ratio ↓
Original Prompt	10.84	0.00	0.00%
Prompts after TAPS	12.81	3.03	32.32%

Table 10: Average generation time (Sec.) per adversarial example for different attack methods.

Methods	Multi-P	CroPA	ADSA (Ours)
Time	438.78	434.48	516.26

6 Further Analysis

We further analyze the surrogate validity, lexical hardening, and efficiency of ADSA.

6.1 Single-Step vs. Multi-Step Self-Correction

We ask whether the single-step SC Prompt Shell in SCG-CPO is a reasonable surrogate for genuine multi-step self-correction. To isolate evaluation-side effects, we use fixed Multi-P AEs rather than ADSA examples and evaluate BLIP-2 under no self-correction, single-step self-correction, and multi-step self-correction. As shown in Table 8, single-step and multi-step self-correction yield similar transferability: the average ASR is 44.70% and 42.43%, respectively, with the same trend across prompt families. This suggests that the first self-correction round captures most of the robustness effect relevant to cross-prompt transferability, supporting the use of the single-step surrogate in SCG-CPO.

6.2 Why TAPS Should Be Viewed as Sparse Lexical Hardening

We analyze whether the gain of TAPS comes from large-scale prompt rewriting or sparse edits on a few critical tokens. Sparse editing is desirable because it acts as a discrete analogue of small-bounded perturbation, limits semantic drift, and reduces surrogate-specific overfitting. Table 9 supports this view: TAPS raises the optimization objective from 10.84 to 12.81 while changing only 3.03 tokens on average (32.32%). Thus, its gain mainly comes from a small set of high-impact lexical positions, supporting TAPS as a sparse lexical hardening module rather than a full paraphrasing mechanism.

6.3 Attack Efficiency Analysis

Table 10 shows that Multi-P, CroPA, and ADSA require 438.78 *s*, 434.48 *s*, and 516.26 *s* per image, respectively, while offline TAPS adds 7.65 *s* per prompt. The overhead is moderate because iterative optimization dominates the cost and Dual-Branch Sampling evaluates only one branch per iteration. In return, ADSA raises the average ASR on OpenFlamingo from CroPA's 75.30% to 87.80%, yielding a favorable efficiency–effectiveness trade-off.

7 Conclusion

In this paper, we study cross-prompt adversarial transferability in VLMs from the perspective of explicit lexical and process-level prompt variation. We show that existing methods mainly search for hard prompts in a local neighborhood of the prompt embedding space, which does not fully capture practical prompt changes. To address this gap, we propose ADSA, which combines TAPS for budgeted token-level substitutions and SCG-CPO for optimization under self-correction conditions with anchored shared prompt perturbation. Extensive experiments show that ADSA consistently outperforms prior cross-prompt attacks and remains effective under in-context learning and VLM self-correction. These results highlight the importance of explicitly modeling both token-level and reasoning-process-level prompt variation when evaluating VLM security.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grants U2441239, 62172243, 62402425, 62402418, and U21A20471); the Fujian Provincial Industrial Technology Development and Application Program (Grant 2025H0043); the China Postdoctoral Science Foundation (Grant 2024M762829); the Zhejiang Provincial Natural Science Foundation (Grant LD24F020002); the "Pioneer and Leading Goose" R&D Program of Zhejiang (Grants 2025C01082, 2025C02033, and 2025C02263); and the Zhejiang Provincial Priority-Funded Postdoctoral Research Project (Grant ZJ2024001).

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob L Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikoł aj Bińkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karén Simonyan. 2022. Flamingo: a Visual Language Model for Few-Shot Learning. In *Proc. Adv. Neural Inf. Process. Syst.*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 23716–23736. https://proceedings.neurips.cc/paper_files/paper/2022/file/960a172bc7fbf0177ccccbb411a7d800-Paper-Conference.pdf

[2] Bin Chen, Jiali Yin, Shukai Chen, Bohao Chen, and Ximeng Liu. 2023. An Adaptive Model Ensemble Adversarial Attack for Boosting Adversarial Transferability. In *Proc. IEEE Int. Conf. Comput. Vis.* 4466–4475. doi:10.1109/ICCV51070.2023.00414

[3] Kanzhi Cheng, Li YanTao, Fangzhi Xu, Jianbing Zhang, Hao Zhou, and Yang Liu. 2025. Vision-Language Models Can Self-Improve Reasoning via Reflection. In *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, Luis Chiruzzo, Alan Ritter, and Lu Wang (Eds.). Association for Computational Linguistics, Albuquerque, New Mexico, 8876–8892. doi:10.18653/v1/2025.naacl-long.447

[4] Wenliang Dai, Junnan Li, DONGXU LI, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. 2023. InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning. In *Proc. Adv. Neural Inf. Process. Syst.*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 49250–49267. https://proceedings.neurips.cc/paper_files/paper/2023/file/9a6a435e75419a836fe47ab6793623e6-Paper-Conference.pdf

[5] Yi Ding and Ruqi Zhang. 2025. Sherlock: Self-Correcting Reasoning in Vision-Language Models. In *Proc. Adv. Neural Inf. Process. Syst.*, D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (Eds.), Vol. 38. Curran Associates, Inc., 101638–101672. https://proceedings.neurips.cc/paper_files/paper/2025/file/93397b489222f1e66a862f795fc32b96-Paper-Conference.pdf

[6] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. 2018. Boosting Adversarial Attacks With Momentum. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 9185–9193.

[7] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. 2019. Evading Defenses to Transferable Adversarial Examples by Translation-Invariant Attacks. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 4312–4321.

[8] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-Box Adversarial Examples for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Iryna Gurevych and Yusuke Miyao (Eds.). Proc. Annu. Meet. Assoc. Comput. Linguist., Melbourne, Australia, 31–36. doi:10.18653/v1/P18-2006

[9] Tingchen Fu and Fazl Barez. 2025. Same Question, Different Words: A Latent Adversarial Framework for Prompt Robustness. In *Proc. Conf. Empir. Methods Nat. Lang. Process.*, Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (Eds.). Association for Computational Linguistics, Suzhou, China, 31305–31319. doi:10.18653/v1/2025.emnlp-main.1595

[10] Jiayi He, Hehai Lin, Qingyun Wang, Yi R. Fung, and Heng Ji. 2025. Self-Correction is More than Refinement: A Learning Framework for Visual and Language Reasoning Tasks. In *Findings Assoc. Comput. Linguist.*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, Vienna, Austria, 6405–6421. doi:10.18653/v1/2025.findings-acl.331

[11] Zhaozhe Hu, Bin Chen, Jia-Li Yin, Bo-Hao Chen, Yaguan Qian, and Shouling Ji. 2026. Mix2Aug: Revisiting Mixing-Based Augmentations for Improving Robust Generalization of Adversarial Training. *IEEE Trans. Dependable Secure Comput.* 23, 2 (2026), 3036–3050. doi:10.1109/TDSC.2025.3632109

[12] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *Proc. AAAI Conf. Artif. Intell.* 34, 05 (Apr. 2020), 8018–8025. doi:10.1609/aaai.v34i05.6311

[13] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. In *Proc. Int. Conf. Mach. Learn. (Proceedings of Machine Learning Research, Vol. 202)*, Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (Eds.). PMLR, 19730–19742. https://proceedings.mlr.press/v202/li23q.html

[14] Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial Attack Against BERT Using BERT. In *Proc. Conf. Empir. Methods Nat. Lang. Process.*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 6193–6202. doi:10.18653/v1/2020.emnlp-main.500

[15] Haohui Liang, Runlin Huang, Yingjun Du, Yujia Hu, Weifeng Su, and Cees G. M. Snoek. 2026. Prompt-Robust Vision-Language Models via Meta-Finetuning. In *Proc. Int. Conf. Learn. Represent.* https://openreview.net/forum?id=3wZ6IIwPJq

[16] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E. Hopcroft. 2020. Nesterov Accelerated Gradient and Scale Invariance for Adversarial Attacks. In *International Conference on Learning Representations*.

[17] Yunqi Liu, Xue Ouyang, and Xiaohui Cui. 2025. GLEAM: Enhanced Transferable Adversarial Attacks for Vision-Language Pre-training Models via Global-Local Transformations. In *Proc. IEEE Int. Conf. Comput. Vis.* 1665–1674.

[18] Yuyang Long, Qilong Zhang, Boheng Zeng, Lianli Gao, Xianglong Liu, Jian Zhang, and Jingkuan Song. 2022. Frequency Domain Model Augmentation for Adversarial Attack. In *Proc. Euro. Conf. Comput. Vis.* Springer Nature Switzerland, Cham, 549–566.

[19] Haochen Luo, Jindong Gu, Fengyuan Liu, and Philip Torr. 2024. An Image Is Worth 1000 Lies: Transferability of Adversarial Images across Prompts on Vision-Language Models. In *Proc. Int. Conf. Learn. Represent.*

[20] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-Refine: Iterative Refinement with Self-Feedback. In *Proc. Adv. Neural Inf. Process. Syst.*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 46534–46594. https://proceedings.neurips.cc/paper_files/paper/2023/file/91edff07232fb1b55a505a9e9f6c0ff3-Paper-Conference.pdf

[21] Atharv Mittal, Agam Pandey, Amritanshu Tiwari, Sukrit Jindal, and Swadesh Swain. 2025. Revisiting CroPA: A Reproducibility Study and Enhancements for Cross-Prompt Adversarial Transferability in Vision-Language Models. *Trans. Mach. Learn. Res.* (2025).

[22] Zhibo Wang, Hengchang Guo, Zhifei Zhang, Wenxin Liu, Zhan Qin, and Kui Ren. 2021. Feature Importance-Aware Transferable Adversarial Attacks. In *Proc. IEEE Int. Conf. Comput. Vis.* 7639–7648.

[23] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L. Yuille. 2019. Improving Transferability of Adversarial Examples With Input Diversity. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2725–2734. doi:10.1109/CVPR.2019.00284

[24] Peng Xie, Yequan Bie, Jianda Mao, Yangqiu Song, Yang Wang, Hao Chen, and Kani Chen. 2025. Chain of Attack: On the Robustness of Vision-Language Models Against Transfer-Based Adversarial Attacks. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 14679–14689.

[25] Xikang Yang, Xuehai Tang, Fuqing Zhu, Jizhong Han, and Songlin Hu. 2024. Enhancing Cross-Prompt Transferability in Vision-Language Models through Contextual Injection of Target Tokens. *arXiv preprint arXiv:2406.13294* (2024).

[26] Yeliang You, Bin Chen, Jia-li Yin, Ximeng Liu, and Wei Lin. 2025. Towards Adversarial-Robust Class-Incremental Learning via Progressively Volume-Up Perturbation Generation. In *Proc. China Conf. Pattern Recognit. Comput. Vis.*, Zhouchen Lin, Ming-Ming Cheng, Ran He, Kurban Ubul, Wushouer Silamu, Hongbin Zha, Jie Zhou, and Cheng-Lin Liu (Eds.). Springer Nature Singapore, Singapore, 61–75.

[27] Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024. Vision-Language Models for Vision Tasks: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 46, 8 (2024), 5625–5644. doi:10.1109/TPAMI.2024.3369699

[28] Jianping Zhang, Weibin Wu, Jen-tse Huang, Yizhan Huang, Wenxuan Wang, Yuxin Su, and Michael R. Lyu. 2022. Improving Adversarial Transferability via Neuron Attribution-Based Attacks. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 14993–15002.

[29] Jiaming Zhang, Junhong Ye, Xingjun Ma, Yige Li, Yunfan Yang, Yunhao Chen, Jitao Sang, and Dit-Yan Yeung. 2025. Anyattack: Towards Large-scale Self-supervised Adversarial Attacks on Vision-language Models. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 19900–19909.

[30] Jiaming Zhang, Qi Yi, and Jitao Sang. 2022. Towards Adversarial Attack on Vision-Language Pre-training Models. In *Proc. ACM Int. Conf. Multimedia* (Lisboa, Portugal) *(MM '22)*. Association for Computing Machinery, New York, NY, USA, 5005–5013. doi:10.1145/3503161.3547801

[31] Yudong Zhang, Ruobing Xie, Jiansheng Chen, Xingwu Sun, Zhanhui Kang, and Yu Wang. 2025. QAVA: Query-Agnostic Visual Attack to Large Vision-Language Models. In *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, Luis Chiruzzo, Alan Ritter, and Lu Wang (Eds.). Association for Computational Linguistics,

Albuquerque, New Mexico, 10205–10218. doi:10.18653/v1/2025.naacl-long.512

[32] Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Chongxuan LI, Ngai-Man (Man) Cheung, and Min Lin. 2023. On Evaluating Adversarial Robustness of Large Vision-Language Models. In *Proc. Adv. Neural Inf. Process. Syst.*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 54111–54138. https://proceedings.neurips.cc/paper_files/paper/2023/file/a97b58c4f7551053b0512f92244b0810-Paper-Conference.pdf

[33] Wen Zhou, Xin Hou, Yongjun Chen, Mengyun Tang, Xiangqi Huang, Xiang Gan, and Yong Yang. 2018. Transferable Adversarial Perturbations. In *Proc. Euro. Conf. Comput. Vis.* 452–467. doi:10.1007/978-3-030-01264-9_28

[34] Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Gong, and Xing Xie. 2024. PromptRobust: Towards Evaluating the Robustness of Large Language Models on Adversarial Prompts. In *Proc. ACM CCS Wkshp. Large AI Syst. Models Priv. Saf. Anal.* (Salt Lake City, UT, USA) *(LAMPS '24)*. Association for Computing Machinery, New York, NY, USA, 57–68. doi:10.1145/3689217.3690621