

TypoCD: Training-Free Contrastive Decoding for Typographic Defense in Large Vision-Language Models

Sihan Xia^{1,2}, Jing Li^{2,*}, Junhao Dong¹, Ivor W. Tsang^{1,2}, Yew-Soon Ong^{1,2,*}

¹College of Computing and Data Science, Nanyang Technological University, Singapore

²CFAR and IAIC, Agency for Science, Technology and Research (A*STAR), Singapore

{sihan002,junhao003}@e.ntu.edu.sg, {li_jing, ivor_tsang}@a-star.edu.sg, asysong@ntu.edu.sg

Abstract

Large vision-language models (LVLMs) remain highly sensitive to text embedded in images: a misleading word printed, painted, or pasted onto an object can override visual evidence and flip the model’s answer. Existing defenses are often tied to specific vision encoders, require training or feature-level intervention, or do not operate on the autoregressive decoding interface of modern LVLMs. We propose **TypoCD**, a training-free contrastive decoding method for typographic robustness. TypoCD constructs a text-localized counterfactual view by detecting visible text regions and inpainting only the typographic content, then contrasts the attacked-image and typography-removed logits during decoding. This localized contrastive direction targets the attacker’s textual signal while preserving unrelated visual content, allowing TypoCD to subtract the typography-induced logit shift without modifying model parameters or internal features. Across three open-source LVLM backbones and synthetic, real-world, and safety-oriented typographic benchmarks, TypoCD recovers attacked-image accuracy to within a few percentage points of clean-image performance without per-backbone tuning, closing a gap that often exceeds 40 percentage points under attack. When combined with a simple prompt prefix, TypoCD also substantially reduces jail-break attack success on safety-oriented typographic attacks.

1 Introduction

Large Vision-Language Models (LVLMs) have advanced rapidly in multimodal understanding, yet their ability to read text in images creates a distinctive vulnerability: a misleading word printed, pasted, or written on an object can override the visual evidence and flip the model’s answer (Radford et al., 2021; Goh et al., 2021; Cheng et al., 2024;

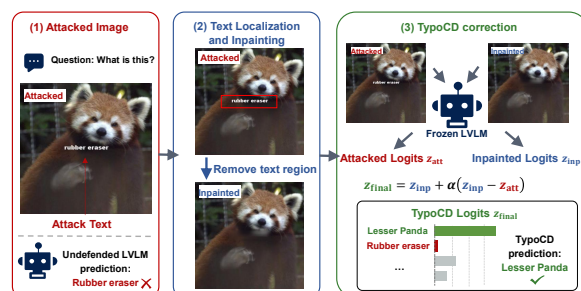


Figure 1: **Overview of TypoCD.** (1) A typographic attack inserts misleading text (*rubber eraser*) into an image, causing the undefended LVLM to follow the rendered word rather than the true visual content (*lesser panda*). (2) TypoCD detects the typographic region via OCR and inpaints it to produce a typography-removed view. (3) At decoding time, TypoCD evaluates the same frozen LVLM on both the attacked and inpainted views, and computes $z_{\text{final}} = z_{\text{inp}} + \alpha(z_{\text{inp}} - z_{\text{att}})$ to subtract the typography-induced logit direction, recovering the visually grounded answer.

Qraitem et al., 2024). Unlike imperceptible adversarial perturbations, such typographic attacks are human-visible, physically realisable, and easy to deploy with stickers, printed labels, or handwritten notes, making them a practical robustness concern for deployed multimodal systems (Sun et al., 2024; Ong and Gupta, 2019).

A closer look at how this attack succeeds reveals a failure mode at the generation interface. The inserted typography is not merely a visual perturbation; it introduces an attacker-controlled textual signal that can propagate to the token scores used by an autoregressive LVLM to generate its answer. Even when the underlying object remains visually recognizable, these scores can be shifted toward the rendered word. This suggests that typographic robustness should be addressed not only by modifying visual representations, but also by correcting the logits from which the answer is decoded.

Existing defenses address typographic vulner-

*Corresponding author.

ability from different angles, but they do not directly correct the logits used by an autoregressive LVLM to select its answer. Prompt-based mitigations (Cheng et al., 2024; Qraitem et al., 2024) are simple and training-free, but can leave residual vulnerability when the rendered text remains visually influential. Encoder-level interventions (Materzyńska et al., 2022; Hufe et al., 2025; Joseph et al., 2025; Wang et al., 2024b) modify CLIP-style visual representations, but are tied to specific vision backbones and do not directly operate at the LVLM decoding interface. Training or white-box approaches require model updates or gradient access (Schlarmann et al., 2024; Zong et al., 2024; Li et al., 2025). This leaves room for a training-free defense that acts directly on the decoding logits while remaining applicable across different LVLM backbones.

We propose **TypoCD** (**Typographic Contrastive Decoding**), a training-free contrastive decoding defense for typographic robustness. TypoCD first constructs a typography-removed view of the attacked image by localizing visible text regions and inpainting the detected typography. It then contrasts the logits obtained from the attacked image and the typography-removed view at every decoding step, subtracting the typography-induced direction from the token scores used to generate the answer. Because the counterfactual view changes only the detected text regions, the contrastive direction is focused on the attacker-controlled textual signal while preserving unrelated visual evidence. Since TypoCD modifies neither the vision encoder nor the language model, it applies across LVLM backbones without retraining, architecture-specific surgery, or gradient access.

We evaluate TypoCD on typographic benchmarks spanning controlled overlays, physical stickers, and handwritten or pasted attack words, including TypoData¹ (Cheng et al., 2024), SCAM (Westerhoff et al., 2025), and RTA-100 (Azuma and Matsui, 2023), across multiple open-source LVLM backbones. TypoCD consistently recovers attacked-image accuracy close to clean-image performance and remains robust under various settings. We further evaluate TypoCD on MM-SafetyBench (Liu et al., 2024b), where combining TypoCD with a simple prompt prefix further lowers attack success on safety-oriented typographic jailbreaks.

¹TypoData refers to the TypoD benchmark introduced in Cheng et al. (2024). We use the name TypoData to avoid confusion with our method, TypoCD.

Our contributions are threefold:

- We propose **TypoCD**, a training-free contrastive decoding defense for LVLM typographic robustness. TypoCD constructs a text-localized counterfactual view and subtracts the typography-induced logit direction during decoding, without modifying model parameters or internal representations.
- We validate TypoCD across multiple LVLM backbones and typographic benchmarks, showing that it recovers attacked-image accuracy close to clean-image performance, remains robust under randomized multi-choice evaluation, and generalizes to safety-oriented typographic jailbreaks when combined with prompt-level defense.
- We provide attention-based diagnostics showing that typographic attacks induce a disproportionate attention anchor on the rendered text, concentrated in the mid-deep layers where the LVLM assembles its answer-token representation. TypoCD’s localized counterfactual is designed to suppress this anchor at decoding time.

2 Related Work

Typographic attacks and defenses. Typographic attacks exploit the tendency of vision-language models to associate written words in images with the visual concepts they denote (Goh et al., 2021; Materzyńska et al., 2022). Recent work shows that this vulnerability extends from CLIP-like VLMs to LVLMs, including VQA and multi-modal reasoning benchmarks (Cheng et al., 2024; Qraitem et al., 2024), physical or scene-coherent sticker attacks (Westerhoff et al., 2025; Cao et al., 2025), and safety-related settings (Gong et al., 2025; Liu et al., 2024b). Existing defenses generally fall into prompt-, encoder-, or training/input-level interventions. Prompt-based methods prepend instructions such as “ignore typographic texts” or “focus on the visual content” (Cheng et al., 2024; Qraitem et al., 2024), but often leave residual vulnerability. Encoder-level methods modify CLIP-like visual representations, including disentangling-based mitigation (Materzyńska et al., 2022), attention-head ablation (Hufe et al., 2025), sparse-feature suppression (Joseph et al., 2025), and mirror-based text isolation (Wang et al., 2024b). Other approaches rely on adversarial training,

safety fine-tuning, learned input-space defense prefixes, or white-box input optimization (Schlarmann et al., 2024; Zong et al., 2024; Azuma and Matsui, 2023; Li et al., 2025). Beyond the visual channel, input-level manipulation is a broader concern for large models: language models can likewise be guided to produce attacker-desired behavior without any parameter access (Lu et al., 2023), and attacks have also been demonstrated at the system level, such as bit-flip attacks that inflate inference cost (Yan et al., 2025).

Inference-time logit intervention and contrastive decoding. TypoCD is related to inference-time methods that adjust token probabilities during generation rather than modifying model parameters. Contrastive decoding has been used in LLMs and LVLMs to improve factuality or reduce hallucination by contrasting two logit distributions, such as early versus late layers in DoLa (Chuang et al., 2024), original versus visually distorted inputs in VCD (Leng et al., 2024), and safety-aware versus base distributions in SafeDecoding (Xu et al., 2024). More broadly, logit-level adjustment has also been explored for multimodal debiasing and controllable generation (Jung et al., 2026; Zhang et al., 2024). These methods differ primarily in how the contrastive signal is constructed, e.g., from layer depth, global image corruption, safety experts, or bias-aware token adjustment. TypoCD follows this inference-time contrastive spirit but targets typographic attacks via a *text-localized* input-space counterfactual; see App. A.2.

3 Method

3.1 Problem Setup

Let v denote a clean image and x a text prompt. We consider typographic attacks that introduce visible textual content into natural images, including rendered overlays, printed labels, stickers, and handwritten text. Given a misleading text string w , the attack produces an image $v' = T(v, w)$ that preserves the underlying visual scene while adding localized text pixels. An autoregressive LVLM parameterized by θ generates a response $y = (y_1, \dots, y_T)$ token by token. For a visual input u , the next-token distribution is

$$p_\theta(y_t \mid u, x, y_{<t}).$$

The attack succeeds when the model’s answer follows the rendered text or the concept it suggests

rather than the underlying visual content. For factual VQA benchmarks, we report accuracy on clean images, attacked images, and defended attacked images. A robust defense should recover attacked-image accuracy toward the clean-image reference while preserving clean-image utility. For safety-oriented jailbreak benchmarks, we report attack success rate (ASR), i.e., the fraction of responses judged as compliant with harmful instructions.

3.2 Typographic Contrastive Decoding

Counterfactual construction. Our method, **TypoCD** (**T**ypographic **C**ontrastive **D**ecoding), first constructs a typography-removed view of the attacked image. Given an attacked image v' , TypoCD uses a text-region localizer that returns bounding boxes or masks for visible text. In our implementation, we instantiate the localizer with EasyOCR (JaidedAI, 2023), whose detector is based on CRAFT-style scene text detection (Baek et al., 2019). The union of detected text boxes is dilated by a small fixed radius to form a binary mask M . We then inpaint the masked region using a local inpainting algorithm (Telea, 2004), obtaining an inpainted image v_{inp} . Only the text locations are used; recognized text strings are discarded. If no text region is detected, TypoCD falls back to standard single-forward decoding on the original input v' .

Contrastive decoding. At each generation step t , TypoCD jointly evaluates the same LVLM on the attacked and inpainted views in a single batched forward pass, yielding:

$$\begin{aligned} \mathbf{z}_{\text{att}}^{(t)} &= \text{logit}_\theta(\cdot \mid v', x, y_{<t}), \\ \mathbf{z}_{\text{inp}}^{(t)} &= \text{logit}_\theta(\cdot \mid v_{\text{inp}}, x, y_{<t}). \end{aligned}$$

The final logits are obtained by extrapolating from the attacked-image logits toward the inpainted-image logits:

$$\mathbf{z}_{\text{final}}^{(t)} = \mathbf{z}_{\text{inp}}^{(t)} + \alpha \underbrace{(\mathbf{z}_{\text{inp}}^{(t)} - \mathbf{z}_{\text{att}}^{(t)})}_{\text{direction of removing typography}}, \quad (1)$$

where $\alpha \geq 0$ controls the strength of the correction. Setting $\alpha = 0$ removes the contrastive shift and reduces to decoding from the inpainted image alone, which we use as an ablation baseline in our experiments. Unless stated otherwise, we use $\alpha = 1$, which we find sufficient across all backbones and benchmarks without per-model tuning. Intuitively,

$\mathbf{z}_{\text{inp}}^{(t)} - \mathbf{z}_{\text{att}}^{(t)}$ estimates how the model’s next-token preference changes when the typographic region is removed. Adding this direction back to the inpainted-image logits suppresses the typography-induced logit shift at the token-generation interface.

To prevent the contrastive shift from promoting low-probability tokens, we follow VCD (Leng et al., 2024) and mask tokens with $p_{\text{inp}}^{(t)}(y) < \gamma \max_{y'} p_{\text{inp}}^{(t)}(y')$ before greedy decoding, where $p_{\text{inp}}^{(t)} = \text{softmax}(\mathbf{z}_{\text{inp}}^{(t)})$. We use $\gamma = 0.1$.

Minimum correction strength. The logit update also gives a simple condition for when TypoCD can change the top prediction. Let y^* be the visually correct token and y be a competing token. Define the branch-wise score difference

$$\Delta_b(y) = \mathbf{z}_b^{(t)}(y^*) - \mathbf{z}_b^{(t)}(y), \quad b \in \{\text{inp}, \text{att}\}.$$

Under TypoCD, this difference becomes

$$\Delta_{\text{final}}(y) = \Delta_{\text{inp}}(y) + \alpha [\Delta_{\text{inp}}(y) - \Delta_{\text{att}}(y)].$$

Therefore, if removing typography improves the relative score of the correct token, i.e., $\Delta_{\text{inp}}(y) > \Delta_{\text{att}}(y)$, then the correction strength required for y^* to outrank y is

$$\alpha > \frac{-\Delta_{\text{inp}}(y)}{\Delta_{\text{inp}}(y) - \Delta_{\text{att}}(y)}.$$

If $\Delta_{\text{inp}}(y) > 0$, Inpaint-only already outranks y and no positive correction is required for this pair. For top-1 recovery, the condition must hold for every competing token $y \neq y^*$. This view explains why TypoCD is most useful when the inpainted branch is close to the decision boundary: the inpainted view may not be sufficient by itself, but the contrastive term can still make the final score difference positive.

Eq. (1) has a guidance-style extrapolation form: TypoCD starts from the typography-removed branch and moves further along the typography-removal direction, analogous in form to CFG but using two visual inputs under the same frozen LVLM rather than conditional and unconditional branches; detailed analysis is provided in App. A.1.

4 Experiment

4.1 Experimental Setup

Benchmarks. We evaluate TypoCD on both synthetic and real-world typographic attacks.

Our main factual-robustness benchmark is TypoData (Cheng et al., 2024), which contains 3065 paired clean and typographically attacked images across seven VQA tasks. To test real-world attacks, we further evaluate on SCAM (Westerhoff et al., 2025), which contains 1162 physical handwritten and printed sticker attacks, and RTA-100 (Azuma and Matsui, 2023), which contains 1000 real-world handwritten or pasted typographic attacks. Finally, we include MM-SafetyBench (Liu et al., 2024b) to evaluate whether TypoCD generalizes to safety-oriented typographic jailbreaks and complements prompt-level safety defenses (1680 pairs).

Models. For TypoData, we evaluate three open-source 7B LVLM backbones from different architectural families: LLaVA-1.5-7B (Liu et al., 2024a), InstructBLIP-Vicuna-7B (Dai et al., 2023), and Qwen2-VL-7B-Instruct (Wang et al., 2024a). These models cover a standard CLIP-projection LVLM, a Q-Former-based architecture, and a dynamic-resolution visual encoder design. For SCAM, RTA-100, and MM-SafetyBench, we report LLaVA-1.5-7B in the main body, following common practice in prior typographic and multi-modal safety evaluations. We additionally evaluate LLaVA-1.5-13B as an extended backbone for size-robustness analysis; results are reported in App. C.2.

Compared methods. We compare the following inference-time methods under the same evaluation protocol. CLEAN and ATTACKED denote decoding on the clean and typographically attacked image, respectively, without defense. PROMPT prepends a typographic-defense instruction following prior work (Cheng et al., 2024; Qraitem et al., 2024). INPAINT-ONLY decodes from the inpainted image alone, isolating the effect of text removal without contrastive logit correction. TYPOCD is our full method with $\alpha = 1$. For MM-SafetyBench only, we additionally report TYPOCD+PROMPT, which combines TypoCD with the prompt prefix. Generic contrastive decoding methods such as DoLa, VCD, and SafeDecoding target different failure modes and do not construct typography-specific visual counterfactuals; we discuss their relation to TypoCD in App. A.2. Unless otherwise stated, all compared methods are our own runs under the same protocol.

Metrics. For VQA benchmarks, we report accuracy. A robust defense should recover attacked-

image accuracy toward the clean-image reference while preserving clean-image utility. For MM-SafetyBench, we report attack success rate (ASR), i.e., the fraction of responses judged as compliant with harmful instructions; lower is better.

Implementation details. We instantiate the text-region localizer with EasyOCR (JaidedAI, 2023), use only detected bounding boxes, and discard recognized text strings. The resulting text mask is dilated by 3 pixels and filled with OpenCV TELEA inpainting (Telea, 2004) using radius 3. Unless otherwise specified, we use $\alpha = 1$ and plausibility threshold $\gamma = 0.1$, with detailed sensitivity analysis in Sec. 5.3. The two forward branches are computed jointly using a batched (batch size 2) forward pass, so that TypoCD adds only modest overhead over standard single-branch decoding. All experiments use greedy decoding.

4.2 Main Results on TypoData

TypoData provides paired clean and typographically attacked images across seven VQA tasks. We follow its standard 2-choice protocol, where the model chooses between the correct answer and the attack word.

Results. Tab. 1 reports per-task accuracy on three LVLM backbones. Typographic attacks substantially degrade the undefended attacked baseline across all models. Average accuracy drops from 89.3% to 49.7% on LLaVA-1.5-7B, from 86.0% to 58.6% on InstructBLIP-Vicuna-7B, and from 92.8% to 62.9% on Qwen2-VL-7B-Instruct. With fixed hyperparameters, TypoCD recovers attacked-image accuracy to 87.6%, 85.1%, and 92.6%, respectively, approaching the corresponding clean-image reference accuracy.

Prompt-based defense improves the performance but remains substantially below TypoCD on average. Inpaint-only also recovers much of the lost accuracy, showing that removing visible typography is an important component. Nevertheless, TypoCD outperforms Inpaint-only on all tasks, indicating that contrastive logit correction provides additional robustness beyond text removal alone.

Randomized Multi-Choice Evaluation The standard TypoData protocol is a 2-choice setting. To test whether TypoCD remains effective beyond this setup, we construct randomized 4/8-choice protocols by adding distractor options and shuffling answer positions. Tab. 2 reports average accuracy

across the seven TypoData tasks. As the number of choices increases, clean accuracy naturally drops, confirming that the randomized protocols are more challenging. Nevertheless, TypoCD consistently recovers attacked-image accuracy close to the clean-image reference across all backbones and protocols. It also outperforms both Prompt and Inpaint-only in every setting, showing that the contrastive correction is not merely exploiting the standard 2-choice format but remains useful when the model must choose among multiple distractors.

Open-ended evaluation. The protocols above present the candidate answers as options. To test whether the correction also holds when the model produces free text, we re-evaluate the species, shape, and texture tasks on LLaVA-1.5-7B with the question only and no options, and score each generated answer with an LLM judge (Qwen2.5-7B-Instruct, greedy decoding, one pass per response) that labels it as matching the ground-truth label, matching the attack word, or neither. Tab. 3 reports accuracy together with the rate at which the model reproduces the attack word. TypoCD recovers most of the accuracy lost to the attack and drives the attack word below 2% on all three tasks, so the defense is not an artifact of the option-based protocol.

Scene-coherent attacks. We also evaluate TypoCD on a stronger TypoData variant generated by SceneTAP (Cao et al., 2025), which plans the adversarial text, placement, and visual integration to produce more scene-coherent typographic attacks. As shown in Tab. 4, SceneTAP further reduces the undefended attacked accuracy compared with the original TypoData overlays, confirming that it constructs stronger attacks. Nevertheless, TypoCD remains largely unaffected and recovers accuracy close to the clean-image reference across all three backbones. This suggests that TypoCD is not limited to simple overlaid text, but remains effective when the typographic attack is more naturally integrated into the scene.

Robustness to text-region localizers. TypoCD only requires a text-region localizer to construct the inpainted view, and is not tied to a specific OCR implementation. Besides EasyOCR, we also test PaddleOCR (Du et al., 2020) and docTR (Mindee, 2021), both of which provide pipelines with text localization and recognition components. Replacing EasyOCR with PaddleOCR or docTR yields aver-

Task	LLaVA-1.5					InstructBLIP					Qwen2-VL				
	Cln	Att	Prm	Inp	TCD	Cln	Att	Prm	Inp	TCD	Cln	Att	Prm	Inp	TCD
species	98.2	51.4	66.8	93.4	95.8	96.0	65.2	95.0	95.6	96.4	100.0	90.6	94.2	100.0	100.0
color	79.0	32.3	54.9	74.9	77.4	74.5	32.8	52.8	68.6	70.8	82.0	31.3	73.3	76.4	83.6
counting	94.7	62.9	68.4	83.4	84.2	89.4	50.5	80.3	83.4	85.0	96.6	89.7	91.0	92.9	93.4
complex	90.5	52.5	60.4	82.3	87.1	77.1	58.8	51.3	76.9	79.1	92.0	54.5	65.2	86.7	89.3
shape	90.0	33.6	49.8	89.4	96.2	86.4	53.8	75.4	86.2	87.4	94.0	35.2	53.4	93.4	99.2
size	87.8	79.6	81.2	86.4	87.8	84.6	72.8	74.8	82.2	83.4	86.6	85.2	85.8	87.0	88.4
texture	84.8	35.8	58.0	76.0	85.0	93.8	76.2	71.4	92.4	93.8	98.6	53.6	78.2	93.8	94.2
Avg.	89.3	49.7	62.8	83.7	87.6	86.0	58.6	71.6	83.6	85.1	92.8	62.9	77.3	90.0	92.6

Table 1: **TypoData standard 2-choice accuracy.** We report accuracy (% , higher is better) on clean images, attacked images, and defended attacked images. TypoCD recovers performance close to the clean-image reference across three 7B LVLM backbones. **Cln: clean image; Att: attacked image; Prm: prompt defense; Inp: inpaint-only; TCD: TypoCD.** The highest accuracy among defended methods (Prm/Inp/TCD) is shown in **bold**. (The same applies to the following tables.)

Backbone	Prot.	Cln	Att	Prm	Inp	TCD
LLaVA-1.5	2ch	89.3	49.7	62.8	83.7	87.6
	4ch	71.2	34.6	46.9	66.3	70.4
	8ch	60.0	27.0	40.1	54.9	57.7
InstructBLIP	2ch	86.0	58.6	71.6	83.6	85.1
	4ch	70.4	51.7	58.0	69.0	70.4
	8ch	55.8	42.8	50.5	54.7	55.5
Qwen2-VL	2ch	92.8	62.9	77.3	90.0	92.6
	4ch	89.0	57.8	70.0	85.8	87.0
	8ch	83.8	54.3	64.9	79.0	80.0

Table 2: **Protocol robustness under multi-choice evaluation.** Average TypoData accuracy (%) under 2/4/8-choice protocols with randomized answer positions.

age TypoData accuracies of 86.1% and 88.5% respectively, compared with 87.6% using EasyOCR. This suggests that TypoCD’s robustness mainly comes from the attacked-vs-inpainted logit correction rather than from a specific OCR backend. Full per-task results are provided in App. C.3.

Comparison with generic contrastive decoding. VCD (Leng et al., 2024) shares the same extrapolation form but builds its contrast branch by perturbing the whole image, which degrades the attack text and the visual evidence together. Under our protocol, it stays at or below the undefended attacked baseline in all nine (backbone, protocol) cells; full results are in App. C.1. DoLa (Chuang et al., 2024) and SafeDecoding (Xu et al., 2024) contrast text-only LLM branches and have no mechanism to react to text rendered in the image (App. A.2).

Task	Accuracy			Attack word	
	Cln	Att	TCD	Att	TCD
species	59.4	53.0	59.2	10.4	0.0
shape	45.6	39.6	45.2	38.6	1.6
texture	37.8	33.8	36.8	12.4	1.8

Table 3: **Open-ended evaluation on TypoData.** Free-text answers on LLaVA-1.5-7B scored by an LLM judge (%). *Attack word*: how often the response reproduces the rendered attack word.

Backbone	Cln	Att	Prm	Inp	TCD
LLaVA-1.5	89.3	46.3	57.1	86.1	88.6
InstructBLIP	86.0	50.5	58.9	83.1	84.7
Qwen2-VL	92.8	58.8	75.8	90.5	92.6

Table 4: **TypoData with SceneTAP-enhanced attacks.** Average accuracy (%) across the seven TypoData tasks.

Remark. TypoCD occasionally exceeds the clean-image reference on some tasks. We do not treat this as a guaranteed property of the method. Rather, this can occur when the added typography changes the relative option scores in a way that, after contrastive subtraction, incidentally reduces a competing wrong option and leaves the correct option on top. We provide the example in App. B.

4.3 Real-World Typographic Attacks

We next evaluate whether TypoCD transfers beyond synthetic overlays. We consider two benchmarks: SCAM (Westerhoff et al., 2025), which contains physical sticker attacks with paired clean images, and RTA-100 (Azuma and Matsui, 2023),

Prot.	Cln	Att	Prm	Inp	TCD
2ch	98.4	30.6	45.2	95.4	96.6
4ch	96.6	30.3	47.3	94.8	94.9
8ch	94.9	28.0	41.8	92.1	92.6

Table 5: **SCAM physical sticker attacks.** Accuracy (% , higher is better) on LLaVA-1.5-7B.

Prot.	Cln	Att	Inp	TCD
2ch	—	37.6	97.4	98.7
4ch	—	35.6	93.8	95.0
8ch	—	29.1	89.2	90.5

Table 6: **RTA-100 handwritten and pasted typographic attacks.** Accuracy (% , higher is better) on LLaVA-1.5-7B. RTA-100 does not provide paired clean images, so clean-image accuracy is not reported.

which contains handwritten or pasted attack words.

SCAM. Tab. 5 shows that physical sticker attacks cause a severe accuracy drop for the undefended attacked baseline. Prompt defense provides limited recovery, whereas TypoCD restores performance close to the clean-image reference and slightly outperforms Inpaint-only across 2/4/8-choice protocols. This suggests that contrastive correction remains useful when the typographic signal comes from physical stickers rather than digital overlays.

RTA-100. Tab. 6 further confirms this trend on handwritten and pasted typographic attacks. Across 2/4/8-choice protocols, TypoCD recovers high accuracy and outperforms Inpaint-only. For reference, prior RTA-100 evaluations are typically conducted in a CLIP-side 100-way zero-shot classification setting, where reported accuracies are around 60% (Azuma and Matsui, 2023; Hufe et al., 2025). Our results are therefore not directly comparable to those CLIP-side evaluations, but they show that TypoCD is effective under an LVLM VQA formulation of the same image set.

4.4 Extension to Safety-Oriented Typographic Jailbreaks

We further evaluate whether TypoCD generalizes from factual typographic deception to safety-oriented typographic jailbreaks. We use the SD+TYPO split of MM-SafetyBench (Liu et al., 2024b), where harmful queries are paired with images containing both generated visual content and rendered typography. This setting is relevant to TypoCD because part of the jailbreak signal is conveyed through visible text in the image.

Method	Venue	SD+TYPO ASR
VLGuard (Zong et al., 2024)	ICML’24	11.1
ECSO (Gou et al., 2024)	ECCV’24	15.9
ETA (Ding et al., 2025)	ICLR’25	15.8
Baseline	—	42.7
Inpaint-only	—	42.0
Prompt	—	15.5
Prompt+Inpaint-only	—	16.7
TypoCD	—	37.5
TypoCD+Prompt	—	6.1

Table 7: **MM-SafetyBench SD+TYPO evaluation.** We report ASR (% , lower is better).

Tab. 7 reports attack success rate (ASR, lower is better). TypoCD alone modestly reduces ASR compared with the undefended baseline, while Prompt Defense provides a stronger safety prior. When combined, TypoCD+Prompt further lowers ASR from 15.5% to 6.1%, showing that decoding-time typographic correction complements prompt-level refusal guidance. Compared with Prompt+Inpaint-only, TypoCD+Prompt also achieves lower ASR, indicating that the contrastive logit correction contributes beyond typography removal alone. It also achieves a lower ASR than the listed published MM-SafetyBench defenses in Tab. 7.

5 Analysis

5.1 Beyond Inpainting: When Does Contrastive Correction Help?

A natural question is whether TypoCD is simply performing text removal. Inpaint-only removes the visible attack word, whereas TypoCD additionally uses the difference between the attacked and typography-removed branches as a logit-space correction direction. This distinction matters when the typography-removed view remains close to a decision boundary: across randomized multi-choice TypoData, RTA-100, and MM-SafetyBench, TypoCD consistently improves over the corresponding Inpaint-only variants.

Fig. 2 illustrates this behavior on a representative 4-choice TypoData texture example. The ground truth for the image is “stratified”, while the attacked image strongly shifts the model toward the rendered attack word “crosshatched”. After inpainting, the visible word is removed, but Inpaint-only remains close to the decision boundary, slightly favoring the attack-consistent option. TypoCD suppresses the attack-amplified option and restores the correct answer by applying the contrastive cor-

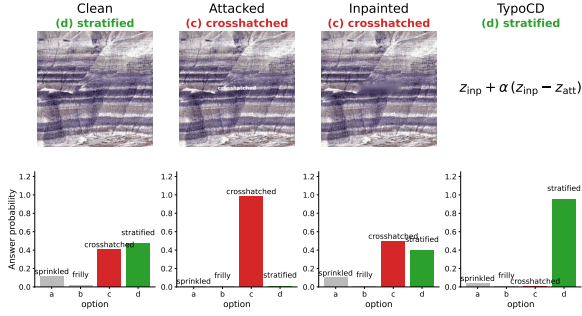


Figure 2: **Contrastive correction helps beyond inpainting in marginal cases.** A TypoData texture example evaluated on LLaVA-1.5 under the randomized 4-choice protocol.

Backbone	Prot.	Inp-wrong	Rescued	Rate (%)
LLaVA-1.5-7B	2ch	172	146	84.9
	4ch	150	138	92.0
	8ch	156	141	90.4
InstructBLIP-7B	2ch	74	65	87.8
	4ch	43	37	86.0
	8ch	34	30	88.2
Qwen2-VL-7B	2ch	86	76	88.4
	4ch	98	88	89.8
	8ch	147	123	83.7

Table 8: **Sample-level rescue analysis on TypoData.** *Inp-wrong*: attacked samples that Inpaint-only answers incorrectly, excluding samples the model already fails on the clean image. *Rescued*: how many of these TypoCD answers correctly.

rection direction. This example highlights the intended division of labor: text-localized inpainting removes the visible trigger, while contrastive decoding corrects the remaining typography-induced shift in the answer distribution.

How often does contrastive correction fix what inpainting misses? The comparison above is aggregate. To isolate the contribution of the contrastive term at the sample level, we restrict attention to the cases where text removal alone is insufficient: for each (backbone, protocol) cell on TypoData, we collect the attacked samples that Inpaint-only answers incorrectly, excluding the samples that the model already fails on the clean image, since these are not recoverable by any typographic defense. Tab. 8 reports how many of these samples TypoCD corrects. Across all nine cells, the contrastive correction recovers 83.7–92.0% of them, indicating that the gain over inpainting is systematic rather than confined to isolated cases.

Task	AR(Attacked)	AR(Inpainted)	Δ_{AR}
shape	2.44	0.66	+1.78
color	2.20	1.02	+1.18
species	2.33	0.73	+1.60

Table 9: **Overall typography anchoring.** Area-normalized attention anchor ratio (AR) on LLaVA-1.5-7B under the 2-choice TypoData protocol, averaged over the last sixteen LLM layers and all heads.

5.2 Attention Diagnostic

We further ask whether typographic bias has a visible correlate inside the LVLM. For LLaVA-1.5-7B, we measure how much the answer-token query attends to image tokens inside the OCR-detected typography region. We define an area-normalized *anchor ratio* (AR) as the fraction of attention mass falling inside the typography box, divided by the fraction of visual tokens that the box covers. Thus, $AR = 1$ corresponds to uniform attention over image tokens, while $AR > 1$ indicates disproportionate attention to the typography region.

Overall typography anchoring. We first compute an overall AR by averaging attention over the last sixteen LLM layers and all attention heads. Tab. 9 reports this statistic on three TypoData sub-tasks. Attacked images consistently show elevated anchoring with AR values around 2.3, while inpainting brings AR close to or below the uniform baseline. The positive Δ_{AR} across all tasks indicates that the attack word is not only visually present, but also receives disproportionate attention during answer generation, and that text-localized inpainting largely suppresses this anchor.

Layer-wise structure. The overall AR shows that typography attracts excess attention, but does not reveal where this effect emerges in the network. We therefore compute the same area-normalized ratio separately at each LLM layer. Fig. 3 shows that for attacked images, AR rises above the uniform baseline and peaks in layers 8–25, the mid-deep range at which the LLM assembles its answer-token representation; after inpainting, AR stays close to or below the uniform baseline across most layers. This suggests that the attack word is not merely detected at the visual input stage but remains an active anchor while the LVLM forms its answer-token representation.

Fig. 4 illustrates the anchoring effect on a representative TypoData species example using a mid-deep layer (layer 24). On the attacked image, the

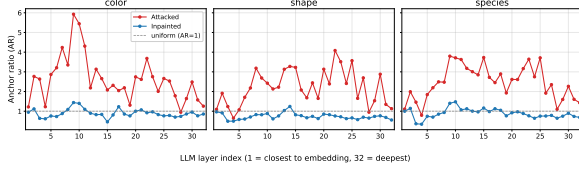


Figure 3: **Layer-wise typography anchoring.** Layer-wise anchor ratio (AR) on TypoData color, shape, and species examples using LLaVA-1.5-7B. AR= 1 denotes uniform attention over visual tokens at that layer.

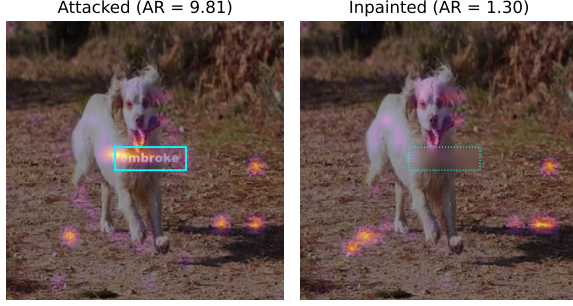


Figure 4: **Per-sample attention visualization on a TypoData species example (label="clumber", attack="Pembroke").** The answer-token query's attention weights over image tokens at LLM layer 24, with both panels under the same color scale. The cyan box marks the OCR-detected typography region.

answer-token query's attention shows a clear hot spot on the typography region (AR= 9.81); after inpainting, this hot spot disappears and AR drops to 1.30, close to the uniform baseline. The pattern is consistent with the systematic statistics in Tab. 9 and the layer-wise structure in Fig. 3. More examples are provided in App. C.4.

Together, these diagnostics support the design of TypoCD: the localized counterfactual removes the region that acts as an attention anchor, and the contrastive decoder subtracts the corresponding change in next-token logits. TypoCD's goal is to prevent the model from anchoring on the rendered text; the consistent AR reduction after inpainting indicates that this objective is achieved.

5.3 Sensitivity to Contrastive Scale α

We sweep the contrastive scale α from 0 to 3 on TypoData (LLaVA-1.5-7B, 2-choice protocol, average over the seven tasks). Fig. 5 reports the resulting accuracy curve. Accuracy rises sharply between $\alpha = 0$ (Inpaint-only, 83.7%) and $\alpha = 0.75$ (87.5%), and then forms a wide plateau across $\alpha \in [0.75, 2.5]$ within which accuracy stays within ± 0.4 pp of its maximum. The default value $\alpha = 1$ lies near the center of this plateau, allowing

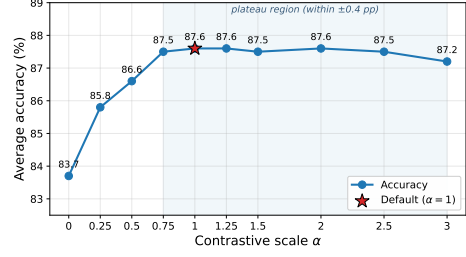


Figure 5: **TypoCD accuracy as a function of the contrastive scale α on TypoData (LLaVA-1.5-7B).**

TypoCD to use a single α across all backbones and benchmarks without per-model tuning.

6 Conclusion

We introduced TypoCD, a training-free contrastive decoding defense that subtracts the typography-induced logit direction during autoregressive decoding, using a text-localized counterfactual view. Empirically, across synthetic, real-world, and safety-oriented typographic benchmarks, TypoCD consistently recovers attacked-image performance without modifying model parameters. Our attention diagnostic further reveals that typographic attacks induce a disproportionate attention anchor in the mid-deep layers where the LLM assembles its answer-token representation, providing a mechanistic basis for the contrastive correction. Taken together, our results highlight the autoregressive logit layer as an effective and underexplored interface for defending LVLMs against visually embedded textual attacks.

Limitations

We discuss several limitations of TypoCD and directions for future work.

Inference overhead. TypoCD introduces additional inference cost compared with single-forward decoding. We measure end-to-end latency on LLaVA-1.5-7B with the TypoData 2-choice protocol on a single A40 GPU. Baseline single-forward decoding takes 280 ms per sample; TypoCD takes 360 ms per sample ($1.29\times$), implemented with a batched two-branch inference so that the attacked and inpainted views are processed jointly. The overhead consists of OCR text detection and Telea inpainting (one-time per image) and the additional visual branch in the batched forward pass, both amortized over the generated tokens.

Multi-text scenes. TypoCD targets the setting where an added attack word conflicts with the un-

derlying visual content and biases the model’s visual judgment, which is most relevant to object- and scene-centric questions. A more challenging case arises in multi-text scenes, where task-relevant scene text (e.g., traffic signs, document content) and adversarial typography coexist. In such cases, removing all OCR-detected regions may also remove useful text, and the key challenge becomes localizing the attack word or phrase. A natural extension is attack-aware text localization: mask candidate OCR regions one at a time and apply TypoCD only to the region whose removal most reduces the typography-induced logit bias. We leave this for future work.

Acknowledgment

This research is supported by the National Research Foundation, Singapore under its National Large Language Models Funding Initiative (AISG Award No: AISG-NMLP-2024-003 and AISG-NMLP-2024-004). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

- Hiroki Azuma and Yusuke Matsui. 2023. Defense-prefix for preventing typographic attacks on clip. In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 3646–3655. IEEE.
- Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. 2019. Character region awareness for text detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9357–9366. IEEE.
- Yue Cao, Yun Xing, Jie Zhang, Di Lin, Tianwei Zhang, Ivor Tsang, Yang Liu, and Qing Guo. 2025. Sceneta: Scene-coherent typographic adversarial planner against vision-language models in real-world environments. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 25050–25059. IEEE.
- Hao Cheng, Erjia Xiao, Jindong Gu, Le Yang, Jinhao Duan, Jize Zhang, Jiahang Cao, Kaidi Xu, and Renjing Xu. 2024. Unveiling typographic deceptions: Insights of the typographic vulnerability in large vision-language models. In *European Conference on Computer Vision*, pages 179–196. Springer.
- Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R Glass, and Pengcheng He. 2024. Dola: Decoding by contrasting layers improves factuality in large language models. In *International Conference on Learning Representations*, volume 2024, pages 54158–54183.
- Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tjong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. 2023. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267.
- Yi Ding, Bolian Li, and Ruqi Zhang. 2025. Eta: Evaluating then aligning safety of vision language models at inference time. In *International Conference on Learning Representations*, volume 2025, pages 2403–2431.
- Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and 1 others. 2020. Pp-ocr: A practical ultra lightweight ocr system. *arXiv preprint arXiv:2009.09941*.
- Gabriel Goh, Nick Cammarata, Chelsea Voss, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Chris Olah. 2021. Multimodal neurons in artificial neural networks. *Distill*, 6(3):e30.
- Yichen Gong, Delong Ran, Jinyuan Liu, Conglei Wang, Tianshuo Cong, Anyu Wang, Sisi Duan, and Xiaoyun Wang. 2025. Figstep: Jailbreaking large vision-language models via typographic visual prompts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23951–23959.
- Yunhao Gou, Kai Chen, Zhili Liu, Lanqing Hong, Hang Xu, Zhenguo Li, Dit-Yan Yeung, James T Kwok, and Yu Zhang. 2024. Eyes closed, safety on: Protecting multimodal llms via image-to-text transformation. In *European Conference on Computer Vision*, pages 388–404. Springer.
- Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.
- Lorenz Hufe, Constantin Venhoff, Maximilian Dreyer, Sebastian Lapuschkin, and Wojciech Samek. 2025. Towards mechanistic defenses against typographic attacks in clip. *arXiv e-prints*, pages arXiv–2508.
- Jaidev AI. 2023. EasyOCR. <https://github.com/JaidevAI/EasyOCR>.
- Sonia Joseph, Praneet Suresh, Ethan Goldfarb, Lorenz Hufe, Yossi Gandelsman, Robert Graham, Danilo Bzdok, Wojciech Samek, and Blake Aaron Richards. 2025. Steering clip’s vision transformer with sparse autoencoders. *arXiv preprint arXiv:2504.08729*.
- Hoin Jung, Junyi Chai, and Xiaoqian Wang. 2026. Adaptive logit adjustment for debiasing multimodal language models. In *International Conference on Learning Representations*, volume 2026, pages 53666–53692.

- Sicong Leng, Hang Zhang, Guanzheng Chen, Xin Li, Shijian Lu, Chunyan Miao, and Lidong Bing. 2024. Mitigating object hallucinations in large vision-language models through visual contrastive decoding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13872–13882. IEEE.
- Chongxin Li, Hanzhang Wang, and Yuchun Fang. 2025. Attack as defense: Safeguarding large vision-language models from jailbreaking by adversarial attacks. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 20138–20152.
- Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved baselines with visual instruction tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26286–26296. IEEE.
- Xin Liu, Yichen Zhu, Jindong Gu, Yunshi Lan, Chao Yang, and Yu Qiao. 2024b. Mm-safetybench: A benchmark for safety evaluation of multimodal large language models. In *European Conference on Computer Vision*, pages 386–403. Springer.
- Ning Lu, Shengcai Liu, Rui He, Qi Wang, Yew-Soon Ong, and Ke Tang. 2023. Large language models can be guided to evade ai-generated text detection. *arXiv preprint arXiv:2305.10847*.
- Joanna Materzyńska, Antonio Torralba, and David Bau. 2022. Disentangling visual and written concepts in clip. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16389–16398. IEEE.
- Mindee. 2021. docTR: Document text recognition. <https://github.com/mindee/doctr>.
- Yew-Soon Ong and Abhishek Gupta. 2019. Air 5: Five pillars of artificial intelligence research. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 3(5):411–415.
- Maan Qraitem, Nazia Tasnim, Piotr Teterwak, Kate Saenko, and Bryan A Plummer. 2024. Vision-llms can fool themselves with self-generated typographic attacks. *arXiv preprint arXiv:2402.00626*.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.
- Christian Schlarmann, Naman Deep Singh, Francesco Croce, and Matthias Hein. 2024. Robust clip: Un-supervised adversarial fine-tuning of vision embeddings for robust large vision-language models. *arXiv preprint arXiv:2402.12336*.
- Zhu Sun, Hongyang Liu, Xinghua Qu, Kaidong Feng, Yan Wang, and Yew Soon Ong. 2024. Large language models for intent-driven session recommendations. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 324–334.
- Alexandru Telea. 2004. An image inpainting technique based on the fast marching method. *Journal of graphics tools*, 9(1):23–34.
- Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, and 1 others. 2024a. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.
- Tiancheng Wang, Yuguang Yang, Linlin Yang, Shao-hui Lin, Juan Zhang, Guodong Guo, and Baochang Zhang. 2024b. Clip in mirror: Disentangling text from visual images through reflection. *Advances in Neural Information Processing Systems*, 37:24523–24546.
- Justus Westerhoff, Erblina Purelku, Jakob Hackstein, Jonas Loos, Leo Pinetzki, Erik Rodner, and Lorenz Hufe. 2025. Scam: A real-world typographic robustness evaluation for multimodal foundation models. *arXiv preprint arXiv:2504.04893*.
- Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. 2024. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5587–5605.
- Xiaobei Yan, Yiming Li, Hao Wang, Han Qiu, and Tianwei Zhang. 2025. Bithydra: Towards bit-flip inference cost attack against large language models. *arXiv preprint arXiv:2505.16670*.
- Yi-Fan Zhang, Weichen Yu, Qingsong Wen, Xue Wang, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. 2024. Debiasing multimodal large language models. *arXiv e-prints*, pages arXiv–2403.
- Yongshuo Zong, Ondrej Bohdal, Tingyang Yu, Yongxin Yang, and Timothy Hospedales. 2024. Safety fine-tuning at (almost) no cost: A baseline for vision large language models. In *Proceedings of the 41st International Conference on Machine Learning*.

A Guidance-based Insight and Contrastive Decoding

A.1 Relation to Classifier-Free Guidance

TypoCD has the same extrapolation form as guidance-style decoding, but the meaning of the guidance direction is different. Classifier-free guidance (CFG) (Ho and Salimans, 2022) was introduced for conditional generation. Given a conditional prediction and an unconditional prediction, CFG forms

$$\begin{aligned}\epsilon_{\text{CFG}} &= \epsilon_{\text{uncond}} + w(\epsilon_{\text{cond}} - \epsilon_{\text{uncond}}) \\ &= (1 + w)\epsilon_{\text{cond}} - w\epsilon_{\text{uncond}}.\end{aligned}$$

The direction $\epsilon_{\text{cond}} - \epsilon_{\text{uncond}}$ captures the change induced by adding the conditioning signal, such as a text prompt. The guidance scale w amplifies this direction, pushing generation toward outputs that better satisfy the condition.

TypoCD applies the same extrapolation principle to LVLM decoding logits, but with two visually intervened inputs rather than conditional and unconditional model branches:

$$\begin{aligned}\mathbf{z}_{\text{final}} &= \mathbf{z}_{\text{inp}} + \alpha(\mathbf{z}_{\text{inp}} - \mathbf{z}_{\text{att}}) \\ &= (1 + \alpha)\mathbf{z}_{\text{inp}} - \alpha\mathbf{z}_{\text{att}}.\end{aligned}$$

Here, $\mathbf{z}_{\text{att}}$ is obtained from the attacked image, while $\mathbf{z}_{\text{inp}}$ is obtained from the typography-removed inpainted view. Thus, the direction $\mathbf{z}_{\text{inp}} - \mathbf{z}_{\text{att}}$ captures the logit change induced by removing visible typography. The scale α amplifies this typography-removal direction, moving the decoded answer away from the attack-dominated branch and toward the typography-removed branch.

The analogy is therefore formal rather than procedural. TypoCD does not require classifier-free training, an unconditional branch, or a generative diffusion model. Both branches use the same frozen LVLM and differ only in the visual input. The shared insight is extrapolation: if one branch reflects a desired condition and the other reflects a weaker or corrupted condition, then moving beyond the desired branch along their difference can strengthen the desired signal. In CFG, the desired signal is prompt conditioning; in TypoCD, it is the removal of adversarial typography.

A.2 Relation to Contrastive Decoding Methods

Several recent methods adopt a contrastive decoding form structurally similar to Eq. 1. They pair

a target branch $\mathbf{z}_a$ with a contrast branch $\mathbf{z}_b$, and form an extrapolated distribution,

$$\mathbf{z}_{\text{final}} = \mathbf{z}_a + \alpha(\mathbf{z}_a - \mathbf{z}_b).$$

Despite this shared extrapolation skeleton, these methods differ from TypoCD in what the two branches represent, which failure mode is being corrected, and whether the contrastive direction is constructed from a localized visual counterfactual. Tab. 10 summarizes the comparison.

Existing contrastive decoding methods construct their contrastive signal from sources that are not designed to isolate visible typographic text. DoLa (Chuang et al., 2024) is a text-only LLM method that contrasts logits projected from a deeper “mature” layer with logits from a dynamically selected earlier layer of the same model on the same input; the input image, if any, is not used to construct the contrastive direction. SafeDecoding (Xu et al., 2024) is also an LLM method, and additionally requires fine-tuning a safety-aligned expert model whose token distribution is contrasted with the base model to improve refusal under jailbreaks; it does not operate on visual inputs and does not construct a visual counterfactual. VCD (Leng et al., 2024) is closest in spirit because it contrasts two visual inputs to the same LVLM. However, its contrast branch is obtained by applying diffusion-process noise to the *entire* image, which simultaneously perturbs the attack text and the underlying visual object. As a result, the direction $\mathbf{z}_a - \mathbf{z}_b$ in VCD entangles the typographic signal with generic visual corruption effects, and is not constructed to isolate visible typography. TypoCD instead constructs a counterfactual by modifying only the detected text region while preserving the rest of the visual scene. The direction $\mathbf{z}_{\text{inp}} - \mathbf{z}_{\text{att}}$ therefore captures the logit change associated with removing visible typography. This makes TypoCD complementary to generic contrastive decoding methods: it uses the same inference-time extrapolation principle, but instantiates the two branches around the specific failure mode of typographic attacks via a text-localized input-space counterfactual.

B Constructive Cases

Fig. 6 shows a case where TypoCD exceeds the clean-image baseline. On the clean image, the model incorrectly predicts “Bedlington terrier”, while the correct answer is “wild boar”. After the attack word “leaf beetle” is added, the attacked

Method	Target branch $\mathbf{z}_a$	Contrast branch $\mathbf{z}_b$	Targeted failure mode	Counterfactual scope
DoLa (Chuang et al., 2024) (LLM)	mature-layer logits	early-layer logits	factuality / hallucination	layer-level, no input change
VCD (Leng et al., 2024) (LVLM)	original-image logits	diffusion-noised-image logits	object hallucination	global image distortion
SafeDecoding (Xu et al., 2024) (LLM)	fine-tuned safety-expert logits	base-model logits	jailbreak refusal	model-level, no input change
TypoCD (ours) (LVLM)	inpainted-image logits	attacked-image logits	typographic answer shift	text-localized inpainting

Table 10: **Comparison with contrastive and guidance-style decoding methods.** These methods share an extrapolation-style update, but differ in modality, in how the two branches are constructed, in which failure mode the contrastive direction targets, and in whether the counterfactual is localized to the attack region.

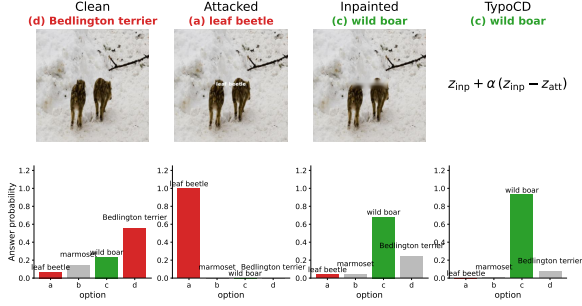


Figure 6: **An occasional constructive case.** On the clean image, the model incorrectly predicts “Bedlington terrier”, while the correct answer is “wild boar”. After the attack word “leaf beetle” is added, the attacked branch is dominated by the rendered text. TypoCD subtracts this attack-induced direction from the inpainted branch, incidentally changing the option ranking so that “wild boar” becomes the top prediction. This effect is not a guaranteed improvement over clean-image inference, but an occasional constructive interaction between the attack-induced logit shift and the contrastive update.

branch is dominated by the rendered text. The contrastive update subtracts this attack-induced direction from the inpainted branch, which incidentally changes the option ranking so that “wild boar” becomes the top prediction. This illustrates a constructive interaction between the attack-induced logit shift and TypoCD’s subtraction, not a guaranteed ability to outperform clean-image inference.

C Detailed Experiment Results

C.1 Comparison with VCD

Tab. 11 reports the full comparison with VCD (Leng et al., 2024) summarized in Sec. 4.2. We run VCD under the identical TypoData protocol used for all other methods, on all three backbones and all three N -choice settings, with its default noise schedule. VCD does not improve over the undefended attacked baseline in any of the nine cells. Its contrast branch is obtained by perturbing the entire image, so the attack word and the object evidence are degraded together and the resulting extrapolation direction is not typography-specific.

TypoCD instead contrasts against a view in which only the detected text region has been removed, which isolates the typographic component of the logit shift.

Backbone	Prot.	Att	VCD	TypoCD
LLaVA-1.5-7B	2ch	49.7	37.1	87.6
	4ch	34.6	31.5	70.4
	8ch	27.0	25.4	57.7
InstructBLIP-7B	2ch	58.6	48.2	85.1
	4ch	51.7	43.6	70.4
	8ch	42.8	36.1	55.5
Qwen2-VL-7B	2ch	62.9	59.3	92.6
	4ch	57.8	56.9	87.0
	8ch	54.3	52.9	80.0

Table 11: **Comparison with VCD on TypoData.** Average accuracy (%) under the same protocol as Tab. 2. *Att*: undefended attacked baseline.

C.2 Results on LLaVA-1.5-13B

To verify that TypoCD’s effectiveness is not specific to a particular model size, we additionally evaluate it on LLaVA-1.5-13B under the TypoData 2-choice protocol. Tab. 12 reports the per-task accuracy. The 13B variant achieves slightly higher clean-image accuracy than 7B on most tasks (90.4% vs. 89.3% on average), and TypoCD recovers attacked-image accuracy to 88.5%, within roughly two percentage points of the clean-image reference. This indicates that TypoCD generalizes to a larger model within the same family without modification.

C.3 Robustness to Text-Region Localizers

TypoCD uses a text-region localizer only to construct the inpainted view, and does not rely on the recognized text content. In the main experiments, we instantiate this component with EasyOCR. To test whether the defense depends on a particular OCR backend, we replace EasyOCR with two alternative OCR pipelines, PaddleOCR (Du et al., 2020) and docTR (Mindoe, 2021), while keeping all other settings unchanged. As shown in Tab. 13, TypoCD

Task	Cln	Att	Prm	Inp	TCD
species	98.0	52.8	69.6	95.2	96.0
color	81.3	35.7	59.5	77.2	80.5
counting	95.5	63.7	66.4	87.2	88.3
complex	91.1	52.9	59.5	84.7	89.3
shape	90.0	32.8	51.0	86.6	91.2
size	89.6	76.2	77.2	85.8	87.3
texture	87.2	38.0	66.0	76.0	86.6
Avg.	90.4	50.3	64.2	84.7	88.5

Table 12: **TypoData 2-choice accuracy on LLaVA-1.5-13B.** Accuracy (% , higher is better).

Task	EasyOCR	PaddleOCR	docTR
species	95.8	92.5	92.0
color	77.4	76.6	83.1
counting	84.2	83.6	88.2
complex	87.1	85.0	86.3
shape	96.2	96.8	96.8
size	87.8	84.4	88.6
texture	85.0	84.0	84.6
Avg.	87.6	86.1	88.5

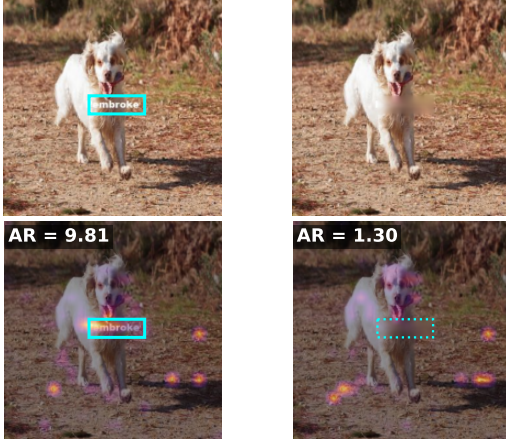
Table 13: **Per-task robustness to text-region localizers.** We report TypoData 2-choice accuracy (%) on LLaVA-1.5-7B when replacing EasyOCR with PaddleOCR or docTR. All other TypoCD settings are kept fixed.

remains stable across different localizers. The average accuracy varies only mildly, from 86.1% with PaddleOCR to 88.5% with docTR, compared with 87.6% using EasyOCR. This suggests that TypoCD is not tied to a specific OCR implementation, as long as the attack text can be reasonably localized.

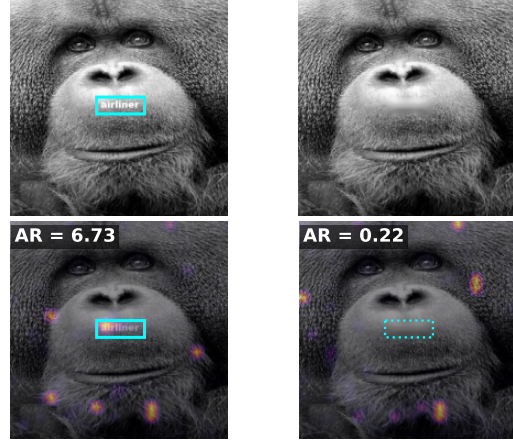
C.4 Per-sample Visualization

Fig. 7 extends the per-sample diagnostic in Sec. 5.2 to six examples spanning the species, color, and shape tasks. Attacked images exhibit AR values from 6.7 to 11.5 with attention concentrated on the typography region, while inpainted views bring AR down to 0.2–1.8 with attention dispersed away from the original typography location.

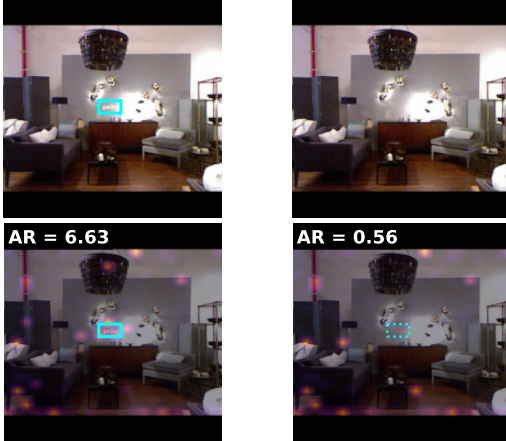
label: clumber, attack: Pembroke



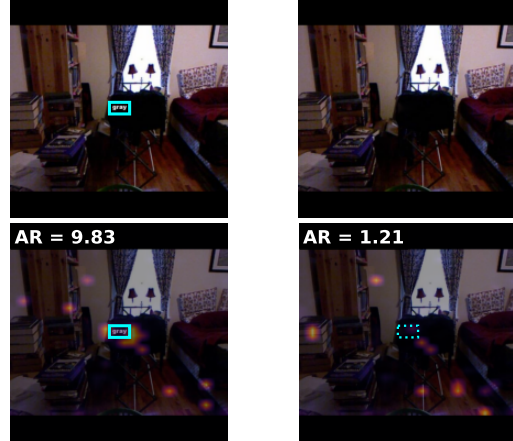
label: orangutan, attack: airliner



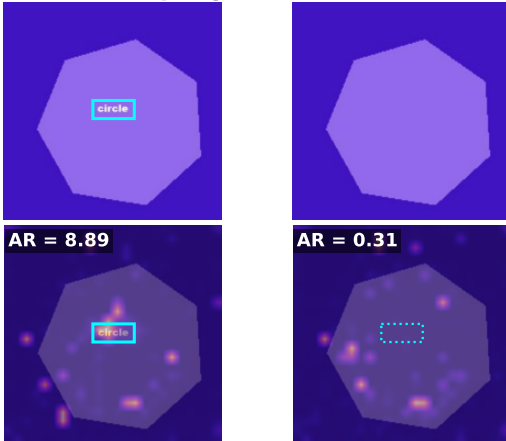
label: black, attack: yellow



label: black, attack: gray



label: heptagon, attack: circle



label: square, attack: star

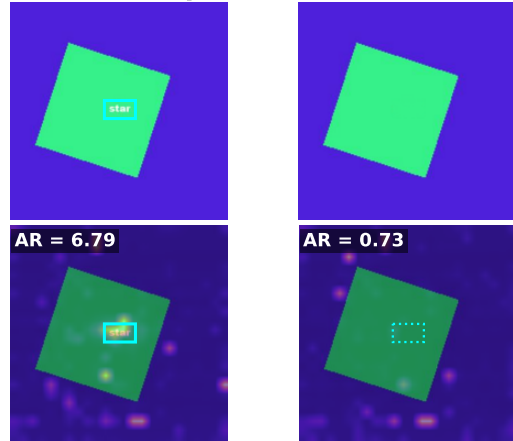


Figure 7: **Per-sample attention diagnostic on six TypoData examples** spanning the species, color, and shape tasks using LLaVA-1.5-7B. For each example, the top row shows the attacked and inpainted images, and the bottom row shows the answer-token query’s self-attention to image tokens at a representative mid-deep layer. The cyan box marks the OCR-detected typography region (solid in attacked, dashed in inpainted). AR values are shown in the bottom-left corner of each attention panel.