

Enhancing Robustness to Prompt Variations in Vision Language Models: A Comprehensive Evaluation of CLIP, SigLIP, and CoOp under Noisy Prompts with Ensembling Strategies

Presented by:
Aishah Altamimi

Outlines

- Introduction
- Literature Review
- Problem Statement
- Baseline Models & Challenges
- Proposed Enhancements
- Algorithm
- Loss Functions
- Datasets & Preprocessing
- Performance Metrix
- Experiment Setup
- Results

Introduction

- VLMs are models learn to match images with text prompts rather than class names into a shared space, then compare their similarity.
- Because they depend on text templates like '[a photo of a {class}](#)', the wording of the prompt strongly affects predictions.
- This motivates the need to study how prompt variations influence accuracy.

Caltech101	Prompt	Accuracy
	a [CLASS].	82.68
	a photo of [CLASS].	80.81
	a photo of a [CLASS].	86.29
	[V] ₁ [V] ₂ ... [V] _M [CLASS].	91.83

(a)

Describable Textures (DTD)	Prompt	Accuracy
	a photo of a [CLASS].	39.83
	a photo of a [CLASS] texture.	40.25
	[CLASS] texture.	42.32
	[V] ₁ [V] ₂ ... [V] _M [CLASS].	63.58

(c)

Baseline Models (Literature Review)

CLIP

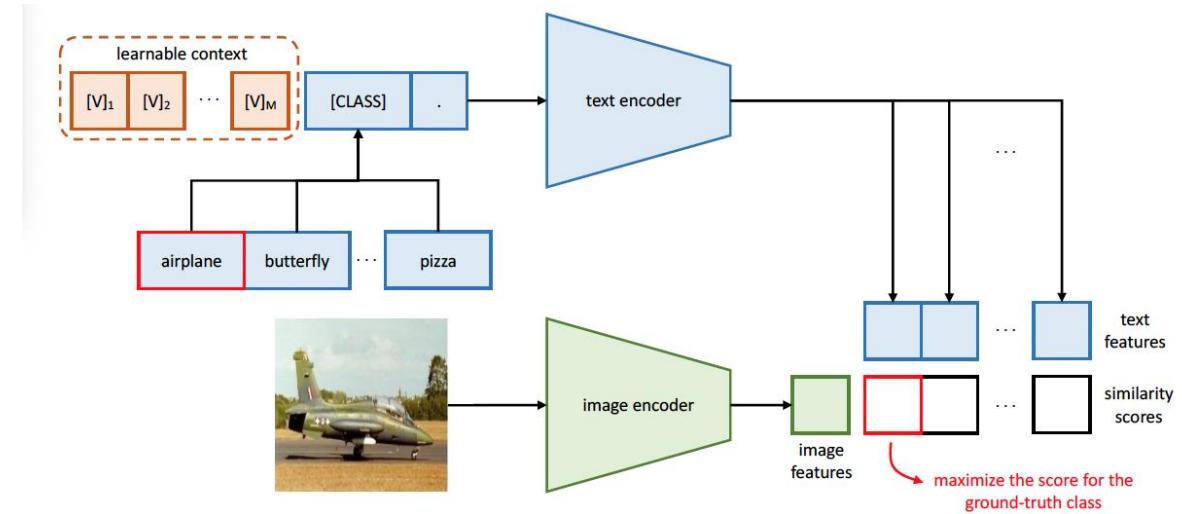
- uses manually written prompts
- **Softmax-based contrastive loss.**

SigLIP

- **Replaces the softmax with a sigmoid pairwise loss.**

CoOp

- CoOp built on CLIP by learning continuous prompt vectors rather than relying on human-written prompts.
- This make it stronger to wording changes.



Literature Review

[4Test-Time Ensembling (TTE)]

- The approach works by creating multiple versions of the same input → running the model on each version --> combining the predictions during test time.

Problem Statement

- In real applications like image search or educational tools, users write prompts with typos, informal phrasing, or emojis.
- Models must handle these variations, but current VLMs often fail under such noise.
- Goal is to Systematically evaluate the robustness of **CLIP**, **SigLIP**, and **CoOp under different types of noisy prompts**
- Explore techniques such as prompt Test time **ensembling** and **noise-aware adapter training** to improve their reliability in real-world user scenarios.

Datasets

Dataset	Domain	Classes	Images	Resolution	Description
Oxford-IIIT Pets	Animals	37	7,349	300×300+	Cat & dog breeds
Caltech-101	Objects	101	9,144	300×200+	Object categories
Food-101	Food	101	101,000	512×512	Food recognition
DTD	Textures	47	5,640	300×300	Texture attributes
EuroSAT	Satellite	10	27,000	64×64	Land-cover classification

Oxford Pets, Caltech-101, and Food-101 match typical web images, while DTD and EuroSAT represent domain-shifted data that differ significantly from natural web images (**Key Findings in this research**).

Loss Functions

- CLIP — Softmax Contrastive Loss
 - Uses a **global softmax** over all image–text pairs in the batch.
 - Encourages each image to match its correct text more than all other texts.
-
- SigLIP — Sigmoid Pairwise Contrastive Loss
 - Replaces softmax with **independent sigmoid pairwise comparisons**.
 - Each image–text pair is evaluated separately.

Loss Functions

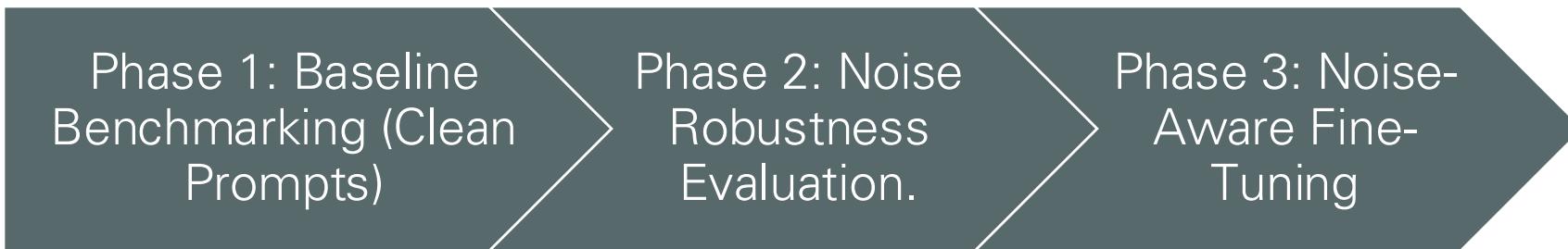
Noise-Aware Adapter Training

- we use cross-entropy plus a consistency loss to force clean and noisy prompts to produce similar predictions.

$$L = L_{\text{CE}} + \lambda L_{\text{consistency}}$$

- Adapter is trained using **clean + noisy prompts** with K=5 ensembling.
- Optimization objective:
- As we show CLIP an image + a **clean prompt** and an image + a **noisy prompt**
- Cross-entropy makes each one predict the **correct class**.
- **Consistency loss** adds an extra rule: “The prediction for the clean prompt and the prediction for the noisy prompt must be **similar**.”

Experimental Framework



Phase 1: Baseline Benchmarking (Clean Prompts)

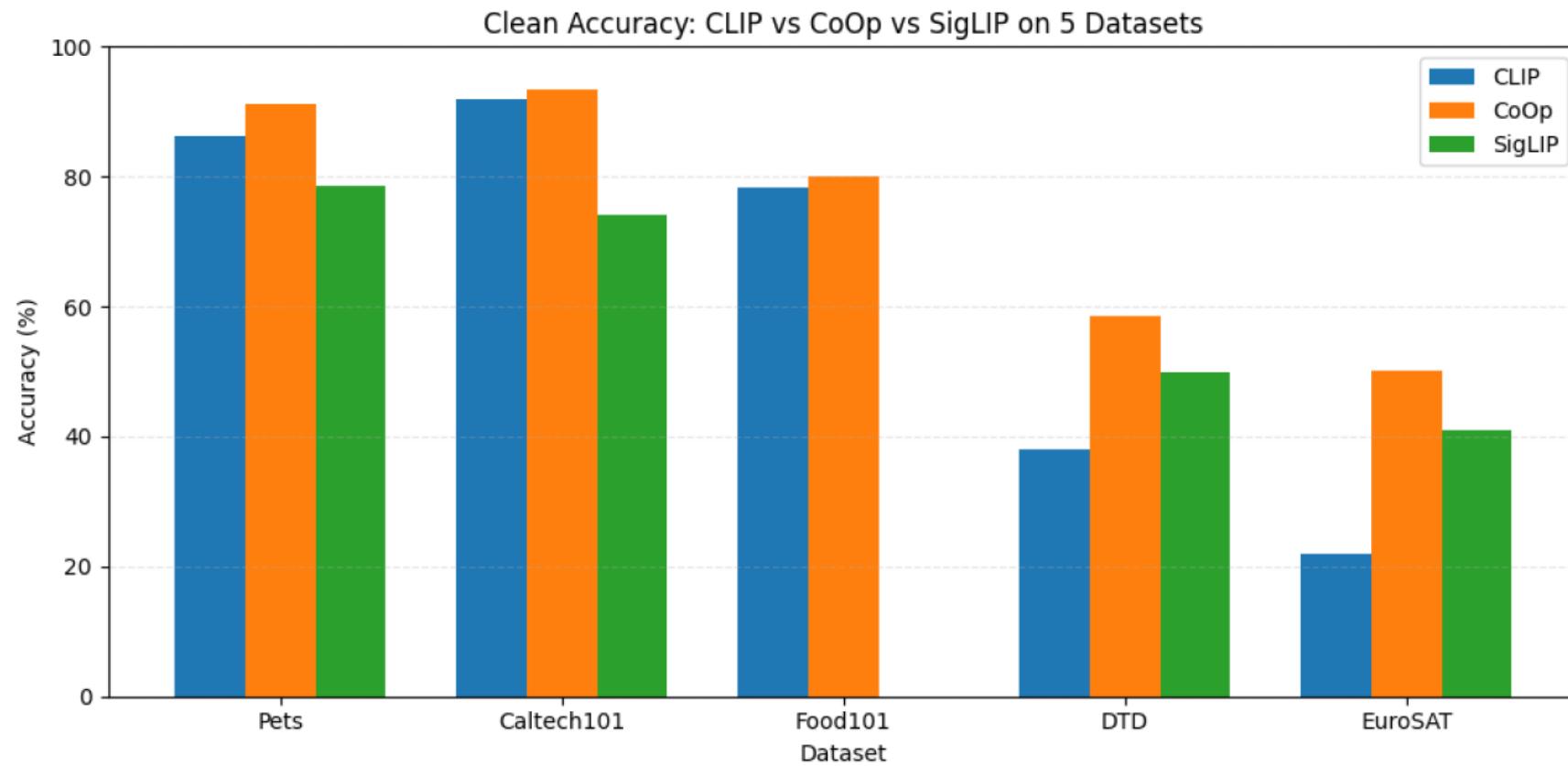
- Evaluate CLIP, SigLIP, and CoOp using *clean* prompts.
- Five datasets: Oxford Pets, Caltech101, Food101, DTD, EuroSAT.
- Train/Val/Test split: 50% / 20% / 30%.
- Metrics: Accuracy, Error Rate, Macro-F1.
- Purpose: Establish a clean-performance baseline before adding noise

Phase 1: Baseline Benchmarking (Clean Prompts)

- CoOp performs best on most datasets
- SigLIP Better than CLIP in domain shift datasets (their data that differ significantly from natural web images) However the CLIP is better than SigLIP on other datasets. (**First Key Finding in this research**)
- These baselines allow us to measure how much accuracy drops once noise is added.

Dataset	CLIP	CoOp	Gain over CLIP	SigLIP
Oxford Pets	86.2 %	91.1 %	+4.9%	78.64%
Caltech101	92.0 %	93.4 %	+1.4%	74.05%
Food101	78.2 %	80.0 %	+1.8%	—
DTD	38.0 %	58.5 %	+20.5%	49.93%
EuroSAT	22.0 %	50.2 %	+28.2%	40.99%

Phase 1: Baseline Benchmarking (Clean Prompts)



Phase 2: Noise Robustness Evaluation:

- Purpose: Measure how accuracy drops across noise levels and whether ensembling helps recover performance.
- Evaluate CLIP, SigLIP, and CoOp under **corrupted prompts**.
- Build a **Noise Prompt Bank** using four noise types:
 - Typos , Letter-case changes, Extra spaces, Emoji insertions
- Noise severity levels: 0 (clean) → 3 (heavy corruption)
- Three test-time strategies:
 - $K = 1$: single noisy prompt
 - $K = 5 + \text{Clean}$: 5 noisy prompts + 1 clean prompt
 - $K = 5 \text{ No-Clean}$: 5 noisy prompts only

Noise Bank Creation

We create **four types of corrupted prompts**, each with **four severity levels**.

1. Typo Noise

- Randomly deletes or swaps characters
- Example:
 - Clean: "a photo of a cat"
 - Severity 1: "a photo of a cta"
 - Severity 3: "a poto of a at"

2. Case Noise

- Randomly changes uppercase/lowercase
- Example:
 - Clean: "a photo of a dog"
 - Severity 2: "A pHoTo oF a DoG"

Noise Bank Creation

3. Space Noise

- Adds random extra spaces
 - Clean: "a photo of a tiger"
 - Level 2: "a photo of a tiger"

Emoji Noise

Adds neutral emojis at the end of the prompt.

- Clean: "a photo of a panda"
- Level 1: "a photo of a panda ✨"
- Level 2: "a photo of a panda ✨📌"
- Level 3: "a photo of a panda ✨📌⭐"

Noise Bank Creation

- Prompt template: Each class is inserted into a fixed template, e.g.
“a photo of a {class}” → “a photo of a siamese cat”.
- Noise injection: We then corrupt the class name using one or more noise operators (typo, case, space, emoji).

Noise Bank Creation

- Severity levels (0–3):
 - 0: clean prompt → a photo of a siamese cat
 - 1: Apply **one** noise operator → “a photo of a siamse cat” (one typo)
 - 2: Two different noise operators → “A photo of a siamse cat  ”
(typo + emoji)
 - 3: Apply **three or more noise operators (strong distortions to the text)** → “A poto of a siamse cat   ”

Phase 2: Algorithm:

We evaluate each model across all noise types, all severity levels, all ensemble strategies, and all datasets. This gives a complete robustness profile.

Algorithm 1 Noise-Aware Prompt Robustness Evaluation for Vision-Language Models

Require:

- 1: Datasets $\mathcal{D} = \{\text{Pets, DTD, EuroSAT}\}$
- 2: Models $\mathcal{M} = \{\text{CLIP, SigLIP, CoOp}\}$
- 3: Prompt template $T(\cdot)$ (e.g., “a photo of a {class}”)
- 4: Noise functions $\mathcal{N} = \{\text{typo, case, space, emoji}\}$
- 5: Severity levels $\mathcal{S} = \{0, 1, 2, 3\}$, ensemble sizes $\mathcal{K} = \{1, 5\}$
- 6: Flag $\text{include_clean} \in \{\text{True, False}\}$

Ensure:

- 7: Accuracy and robustness metrics for each (model, dataset, noise setting)
- 8: **for** each dataset $D \in \mathcal{D}$ **do**
- 9: Load images and labels from disk.
- 10: Apply preprocessing: resize / center-crop, convert to tensor, normalize.
- 11: **for** each model $M \in \mathcal{M}$ **do**
- 12: Load pre-trained VLM M (CLIP, SigLIP, or CoOp head).
- 13: Freeze backbone parameters (zero-shot / few-shot setting).

```
14:   ...  
15:   for each severity level  $s \in \mathcal{S}$  do  
16:     for each ensemble size  $k \in \mathcal{K}$  do  
17:       Build prompt bank for each class:  
18:       for each class name  $c$  do  
19:         Start with clean text  $t_{\text{clean}} = T(c)$ .  
20:         if  $\text{include\_clean} = \text{True}$  then  
21:           Add  $t_{\text{clean}}$  to the prompt set.  
22:         end if  
23:         Sample  $(k - 1)_{\text{include\_clean}}$  noisy variants by  
24:         composing functions from  $\mathcal{N}$  with severity  $s$ .  
25:       end for  
26:       Encode all prompts with the text encoder of  $M$   
27:       and  $\ell_2$ -normalize embeddings.  
28:       Initialize counters:  $\text{correct\_top1} \leftarrow 0$ ,  
29:        $\text{correct\_top5} \leftarrow 0$ ,  $\text{total} \leftarrow 0$ .  
30:       for each mini-batch of images  $x$  with labels  $y$  do  
31:         Extract image features with  $M$  and normalize  
32:         them.  
33:         Compute logits between image features and  
34:         each class prompt (using test-time prompt ensembling over  $k$   
35:         prompts).  
36:         Obtain predicted labels  $\hat{y}$  by arg max over  
37:         classes.  
38:         Update top-1 / top-5 accuracy counters.  
39:       end for  
40:       Compute top-1, top-5, precision, recall, and F1 for  
41:       this setting.  
42:       Store results as  $(D, M, s, k, \text{include\_clean})$ .  
43:     end for  
44:   end for  
45: end for  
46: end for
```

Phase 2: Performance Metrics - Accuracy

For each model, dataset, and noise severity:

1. Feed an image + (clean or noisy) prompt(s) into the model.
2. The model outputs a **predicted class** (Top-1).
3. Compare it with the **true label** of the image.
4. Count:
 - **Correct** predictions
 - **Total** test images

Then compute:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of test images}} \times 100\%$$

Repeated for Each **severity level** (0, 1, 2, 3)

Each **ensemble setting** (K=1, K=5, K=5+Clean, K=5 No Clean)

Each **model** (CLIP, SigLIP, CoOp)

Each **dataset**

Phase 2: Evaluation Metrics – Other Metrics

- Absolute Accuracy Drop (Δ):

Measures how much accuracy decreases due to noise:

$$\Delta = \text{Acc_clean} - \text{Acc_noisy}$$

- Relative Robustness (RR):

How much of the clean accuracy is retained under noise:

$$RR = (\text{Acc_noisy} / \text{Acc_clean}) \times 100$$

- Relative Accuracy Drop (RAD):

Noise-induced degradation relative to clean accuracy:

-

$$RAD = ((\text{Acc_clean} - \text{Acc_noisy}) / \text{Acc_clean}) \times 100$$

(*RR and RAD are complementary: RR = 100 – RAD*)

- Ensemble Gain (Δ_{ens}):

Improvement from prompt ensembling:

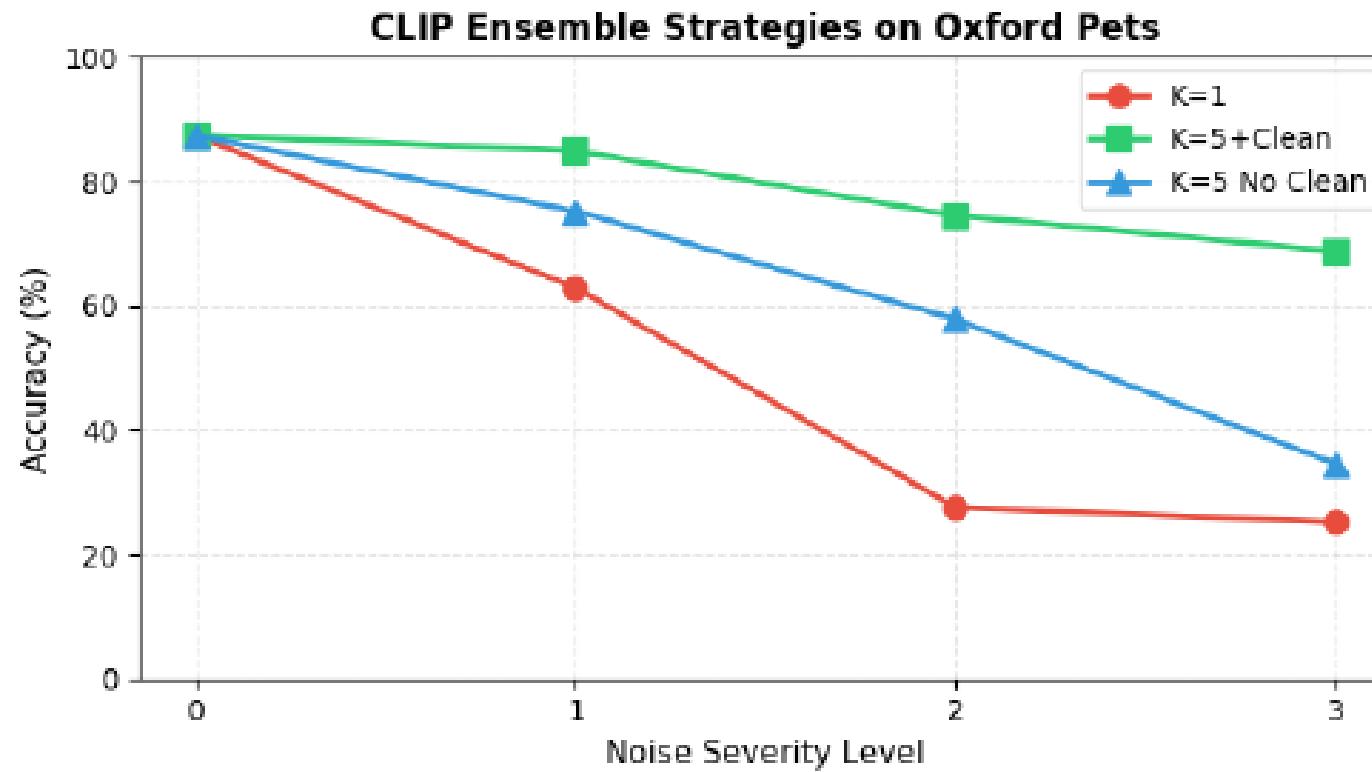
$$\Delta_{\text{ens}} = \text{Acc_ensemble} - \text{Acc_single-prompt}$$

CLIP Robustness Results (All Datasets)

- Strong degradation as noise severity increases.
- Prompt ensembling (K=5 + Clean) consistently boosts accuracy. (**Second Finding in this research**)

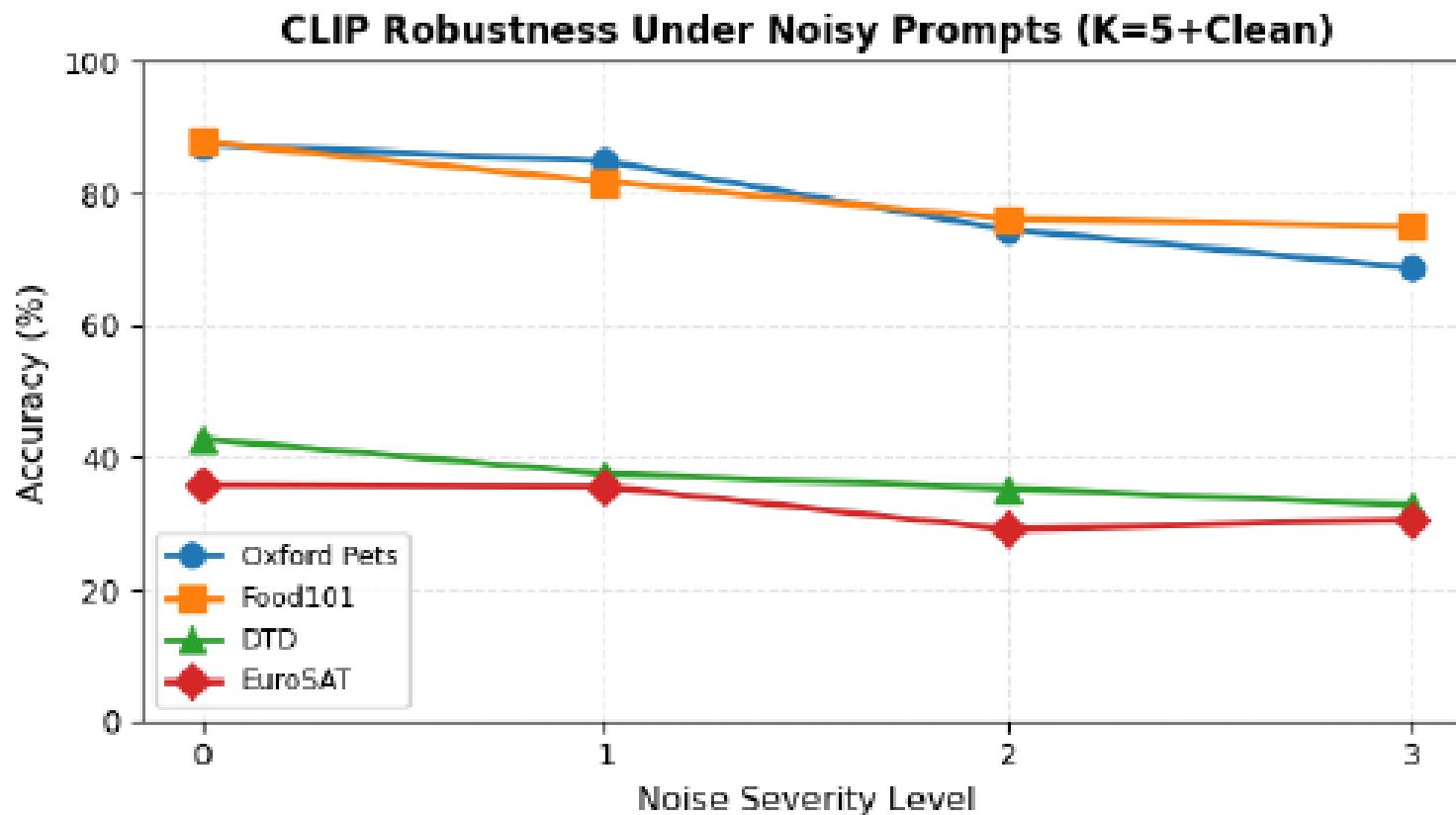
Strategy	Sev 0	Sev 1	Sev 2	Sev 3
K=1	87.4 %	62.9 %	27.6 %	25.2 %
K=5+Clean	87.4 %	84.9 %	74.5 %	68.8 %
K=5 No Clean	87.4 %	75.2 %	57.8 %	34.6 %

CLIP Ensemble Strategy on Oxford Dataset



CLIP Robustness across All Datasets

Datasets with domain shift (DTD, EuroSAT) show significant drops.



SigLIP Robustness Results

- Accuracy sometimes improves when using noisy prompts (regularization effect).

Strategy	Sev 0	Sev 1	Sev 2	Sev 3
K=1	55.4 %	47.1 %	38.5 %	32.8 %
K=5+Clean	55.4 %	60.3 %	68.9 %	74.0 %
K=5 No Clean	55.4 %	65.7 %	80.1 %	74.0 %

CoOp Robustness Results (Oxford Pets Only)

- CoOp remains stable at all severity levels because it uses learned prompts.

Model	Sev 0	Sev 1	Sev 2	Sev 3
CoOp	91.11%	91.11%	91.11%	91.11%

Phase 3: Regularized Noise-Aware Fine-Tuning

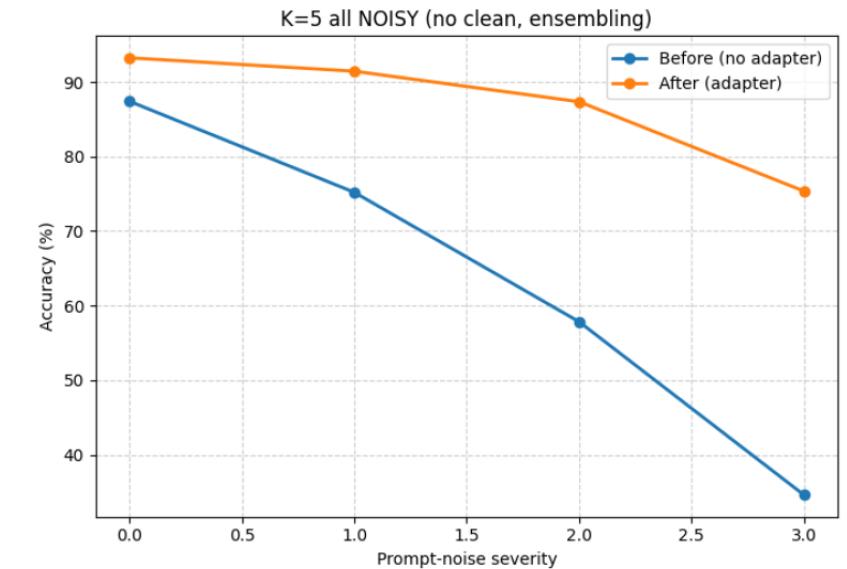
- In Phase 3, we fine-tune CLIP using both clean and noisy prompts with a consistency regularization loss.
- The goal is to teach the model that clean and noisy prompts should produce similar outputs.
- Train on **Oxford Pets** using:
 - Cross-entropy loss
 - KL-divergence consistency loss (forces clean + noisy prompts to match)

Phase 3: Regularized Noise-Aware Fine-Tuning

- **Optimizer:** AdamW
- **Learning Rate:** 1×10^{-4}
- **Weight Decay:** 0.01 (ℓ_2 regularization)
- **Batch Size:** 32
- **Training Epochs:** 5
- **Training Split:** 70% of Oxford-Pets dataset
- **Consistency Weight (λ):** 0.5
- **Loss Functions:**
 - Cross-Entropy (CE) — ensures correct classification
 - Consistency Loss — stabilizes predictions across noisy prompts

Regularized Noise-Aware Fine-Tuning

Setting	Severity	Before	After	Δ (Improvement)
K = 1	0	87.35	93.24	+5.89
	1	29.54	68.11	+38.57
	2	15.94	45.84	+29.90
	3	7.00	23.79	+16.79



Ensemble Strategy



(a) Clean prompt (correct)

Prompt: “a photo of a birman cat”

Prediction: birman (✓)

(b) Noisy prompt (misclassified)

Prompt: “a photos of blIrMaN ca t”

Prediction: tabby (✗)

(c) Ensemble of noisy prompts (recovered)

Prompts: averaged over multiple corrupted variants
(typo, case, space) of “a photo of a birman cat”

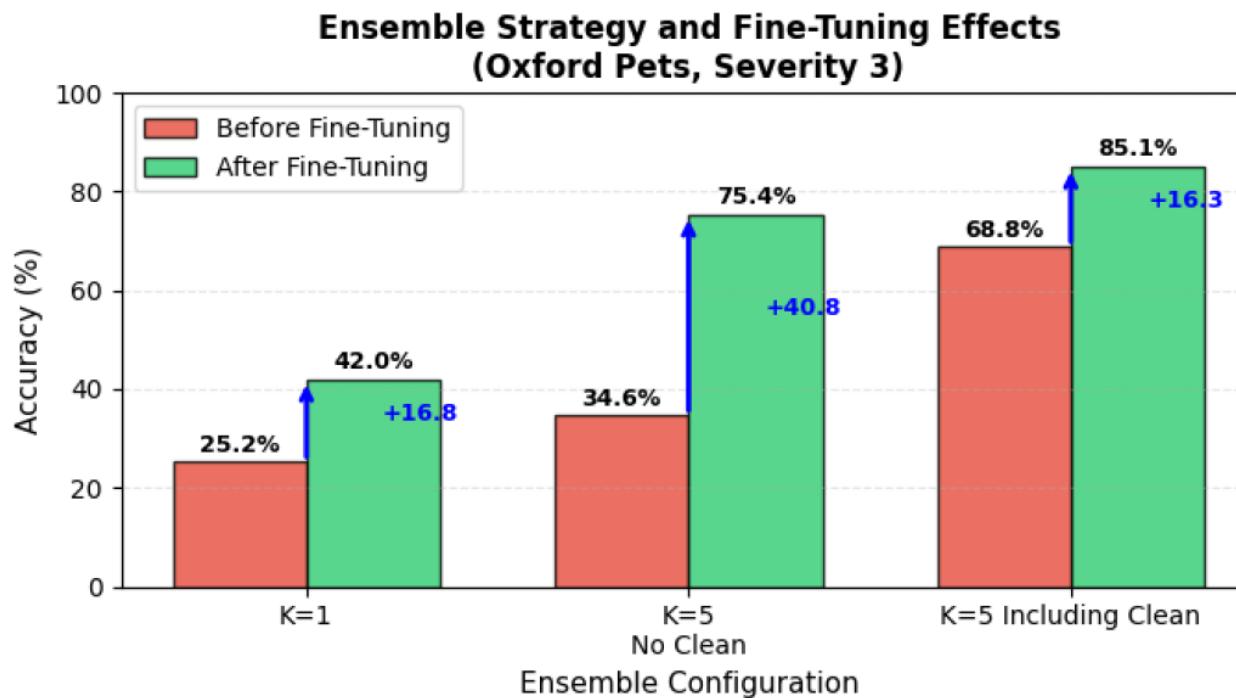
Prediction (averaged logits): birman (✓)

Fig. 1. Illustration of how noisy prompts affect VLMs on the Birman class (Oxford Pets). Clean prompt = correct (a), noisy prompt = incorrect (b), ensembling recovers accuracy (c).

ABLATION STUDY

Our ablation shows that both

1. Ensemble Strategy
2. Noise-aware training significantly boosts performance under noisy prompts.



Conclusion

- VLMs are highly sensitive to prompt variations, especially CLIP.
- Prompt ensembling is a simple and effective Strategy that enhance model robustness to noisy prompts.
- CoOp remains stable across noise levels, and SigLIP performs better than CLIP under domain shift.
- Training objectives (Loss Functions) strongly affect robustness, and noise-aware fine-tuning significantly improves CLIP's stability.

Future Work

- Testing additional VLM architectures like BLIP and LLaVA would reveal whether our findings generalize to other models.
- Exploring more noise types such as grammatical errors would build a more comprehensive robustness benchmark.

Thank you for listening

Any Question?