Extract Free Dense Misaligment from CLIP

JeongYeon Nam[1], Jinbae Im[1], Wonjae Kim[2], Taeho Kil[1]
[1]NAVER Cloud AI, [2]NAVER AI LAB (Email: jy.nam@navercorp.com)

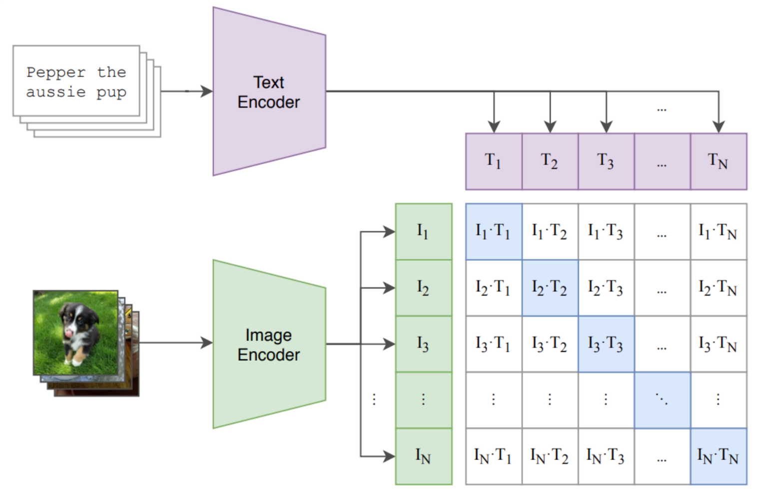
NAVER Cloud · NAVER AI LAB · AAAI · GITHUB

Motivation

- Text-to-image, image-to-text generative models still generate output **misaligned with their inputs.**
- While **CLIPScore** is widely used for measuring alignment between image and text, single scalar score **lacks interpretability**.
- Recent studies focus on detecting dense misalignments to enhance explainability and provide model feedback.
- However, most approaches heavily rely on large generative models, leading to **high computational costs.**

Preliminaries

CLIP



$score_{v,t} = I_v T_t$

Gradient-based relevance map method

- Existing gradient-based relevance map extraction methods (e.g., GradCAM) primarily *focus on generating heatmaps* from images. They mostly **disregard negative gradients**, assuming these represent unrelated information.
- In transformer literature, GAE (Chefer et al., '21) derives output logits with respect to attention maps.

$$\nabla A_l^h = \frac{\partial \text{score}_{v,t}}{\partial A_l^h},$$

($A_l^h \in \mathbb{R}^{n \times n}$ denote attention map at l-th layer h-th head)

- Then *eliminates negative gradients* and applies an element-wise product with the attention map itself.

$$R_l^h = \text{ReLU}(\nabla A_l^h \odot A_l^h). \qquad R_l = \frac{1}{H}\sum_{h=1}^{H} R_l^h.$$

- Across layers, relevance map is updated with $R_l \leftarrow R_l + R_l \odot R_{l-1}$. Final attribution is calculated by pooling [CLS] index row of relevance map.

CLIP for Dense Misalignment (CLIP4DM)

CLIP4DM

1. We **remove ReLU** so that negative gradients can represent misalignments.
2. We aggregate relevance map per layer with **averaging** instead of a product.
3. Then, we merge tokens into a word by **averaging their attributions** and predict a word whose attribution is **lower than epsilon** as misaligned.

$$\text{mis}(w_j) = \begin{cases} 1, & \text{if } w_j < \epsilon \\ 0, & \text{otherwise.} \end{cases}$$

Caption: `A man riding snowboard down a snow covered slope.`
CLIPScore: `61.3`
Ours: `A man riding snowboard down a snow covered slope.`
Misaligned word: *`snowboard`*

F-CLIPScore

To classify global misalignments, we propose F-CLIPScore which aggregate sum of negative attributions and CLIPScore.

$$\text{F-CLIPScore}(v,t) = (1 - \text{score}_{v,t}) \cdot \sum_j \text{mis}(w_j) \cdot w_j.$$

Experiments

We experiment five dense misalignment detection benchmarks.
- ❶ **FOIL** : natural image & natural caption.
- ❷ **nocaps-FOIL** : natural image & natural caption.
- ❸ **HAT** : natural image & generated caption
- ❹ **SeeTrue-Feedback** : both natural / generated image and caption
- ❺ **Rich-HF** : natural caption & generated image

Results

1) Our results demonstrate **state-of-the-art performance** in *detecting dense misalignments* among zero-shot models.
2) F-CLIPScore **outperforms CLIPScore** in *classifying misaligned pairs*.
3) Our method achieves **significantly higher efficiency**, with *27–75 × better FPS*, as it relies solely on CLIP, unlike baselines that depend on large generative models.

Method	FPS	❶ FOIL LA	❶ FOIL AP	❷ nocaps-FOIL LA	❷ nocaps-FOIL AP
CLIPScore (ViT-B/32)	13.4	-	*0.71*	-	0.69
CLIPScore (ViT-H/14)	8.7	-	0.76	-	0.72
RefCLIPScore (ViT-B/32)	8.7	-	*0.75*	-	*0.74*
ALOHa	0.2	0.40	0.61	0.45	0.70
Ours (ViT-B/32)	12.0	0.73	0.71	0.60	0.69
Ours (ViT-H/14)	7.1	**0.84**	0.81	**0.72**	**0.80**

❸

method	ref. captions	FPS	LA	AP
CLIPScore		18.8	-	*0.39*
RefCLIPScore	✓	9.0	-	*0.43*
ALOHa	✓	0.2	0.20	**0.49**
Ours (ViT-B/32)		9.6	0.19	0.36
Ours (ViT-H/14)		6.6	**0.35**	0.36

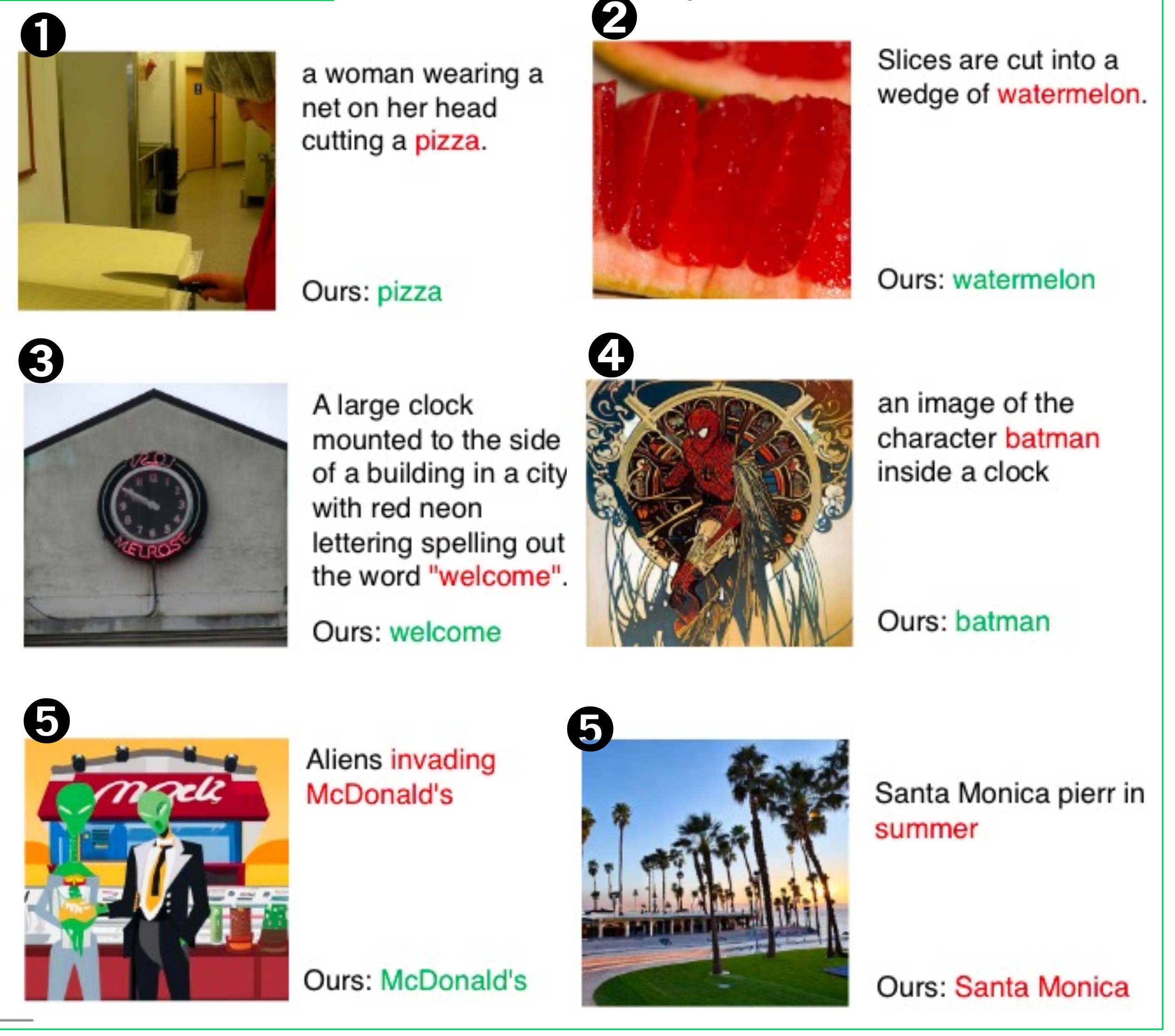
❹

Model	ft.	FPS	NLI score
PaLI 5B		-	0.226
MiniGPT-v2 (LLaMa2-7B)		0.28	0.560
Ours (ViT-B/32)		7.90	0.605
Ours (ViT-H/14)		5.81	0.660
PaLI 5B	✓	-	0.765
PaLI 17B	✓	-	**0.785**

❺

Model	ft.	F1	precision	recall
ALOHa		0.34	0.31	0.39
Ours (ViT-B/32)$_{\epsilon=-0.00001}$		0.40	0.33	0.50
Ours (ViT-H/14)$_{\epsilon=-0.00001}$		0.43	0.37	**0.52**
Ours (ViT-H/14)$_{\epsilon=-0.00005}$		0.31	0.49	0.23
Rich-HF (multi-head)	✓	0.43	**0.63**	0.33
Rich-HF (augmented prompt)	✓	**0.44**	0.61	0.341

Model	ft.	pearson	spearma
CLIPScore (ViT-B/32)		0.19	0.13
PickScore (ViT-H/14)		0.35	0.34
Ours (ViT-B/32)$_{\epsilon=-0.00001}$		0.28	0.33
Ours (ViT-H/14)$_{\epsilon=-0.00001}$		0.37	0.43
CLIPScore (ViT-B/32)	✓	0.40	0.39
Rich-HF (multi-head)	✓	**0.49**	**0.50**
Rich-HF (augmented prompt)	✓	0.47	0.50



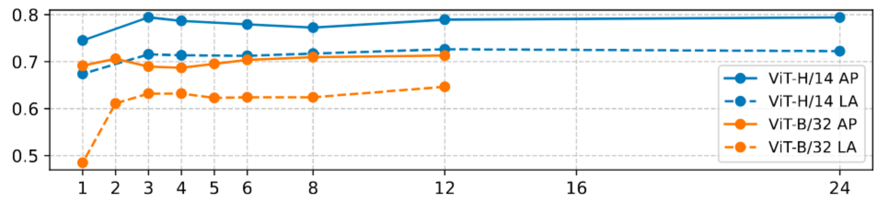
Analysis

Ablation on full-gradients

$ReLU(-\nabla A_l^h)$	$ReLU(-\nabla A_l)$	LA	AP
✓		0.698	0.779
	✓	0.700	0.776
		0.716	**0.794**

Using both negative and positive attributions is helpful.

Number of layers



Part-of-speech

Metric	NOUN	PROPN	VERB	ADV	ADJ	NUM	ADP
F1	0.393	0.312	0.301	0.258	0.258	0.132	0.177
Precision	0.470	0.602	0.567	0.444	0.417	0.500	0.278
Recall	0.337	0.211	0.205	0.182	0.187	0.076	0.130

While ours can extract all sorts of POS, it shows that it follows CLIP's strength and weakness.

Components of F-CLIPScore

Dataset	Method	AP	Pearson	Spearman
nocaps-FOIL	$\text{score}_{v,t}$	0.722	-	-
	$\sum_j \text{mis}(w_j) \cdot w_j$	0.776	-	-
	F-CLIPScore	**0.794**	-	-
Rich-HF	$\text{score}_{v,t}$	-	0.171	0.085
	$\sum_j \text{mis}(w_j) \cdot w_j$	-	0.352	0.419
	F-CLIPScore	-	**0.368**	**0.433**

Simply summing negative attributions is more effective than CLIPScore for classifying misaligned pairs.

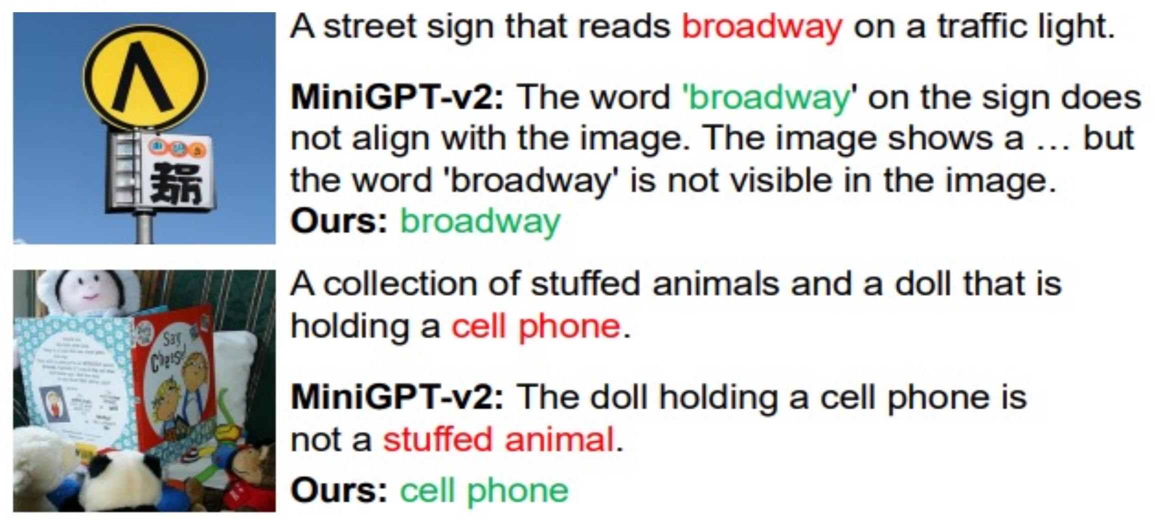
Backbones

backbone	source	pretrained	IN acc.	LA	AP
ViT-B/32	openai	WIT-400M	0.632	0.602	0.723
ViT-B/32	openclip	LAION-2B	0.656	0.667	0.760
ViT-B/16	openai	WIT-400M	0.687	0.679	0.747
ViT-L/14	openai	WIT-400M	0.753	0.653	0.781
ViT-L/14	openclip	LAION-2B	0.753	0.728	0.796
ViT-H/14	openclip	LAION-2B	0.780	0.716	0.806
ViT-g/14	openclip	LAION-2B	0.766	0.706	0.806

Performance is correlated with the model's ImageNet accuracy, and a trained wider corpus shows better performance.

Comparison with baselines

Our method captures a *wider range of misalignments* (e.g., attributes, entity-level objects, intangible objects) compared to a combination of foundation models. (❸)



Also our method output *concise and precise feedback* with much enhanced FPS compared to generative models. (❹)