Beyond the Global Scores: Fine-Grained Token Grounding as a Robust Detector of LVLM Hallucinations

Supplementary Material

6. Experimental configurations

For hallucination detection, our configurations are as follows:

- **MetaToken**: For our greedy decoding setup, the probability difference term in Eq. 11 of the original paper is always zero; we instead use token probability directly. We implement two binary classifiers, Logistic Regression (LR) with `lbfgs` solver and Gradient Boosting (GB) with 100 estimators.
- **HalLoc**: We use a pretrained CLIP-ViT-B/32 encoder with a linear projection to match VisualBERT input dimensions. A single classification head is used, as we focus exclusively on object hallucination. Training follows the original optimizer setup (AdamW, $\beta = (0.9, 0.999)$, weight decay 1.0×10^{-2}, learning rate 1.0×10^{-6}) with batch size 16 on an RTX 3090.
- **SVAR**: We train a one-hidden-layer MLP (hidden dim 248, learning rate 0.001) for 50 epochs as a result of the hyperparameter search strategy.

6.1. LVLM Inference Parameters

All models use temperature $= 0.1$ and top-$p = 20$, top-$p = 50$ for decoding.

6.2. Classifier Hyperparameters

We optimize XGB, RF and MLP classifiers via grid search.

Model	Hyper-parameter	Values
XGB	depth	{4,6,8}
	learning rates	{0.1,0.05}
	no. estimators	{100,200,500}
RF	depth	{None,10,20}
	trees	{200,400,600}
MLP	hidden size	{64,128,256}
	learning rate	{0.01,0.001}
	optimizer	{Adam,SGD}

Table 5. Hyper-parameter ranges for each classifier.

The best hyper-parameter configurations were found to be: for XGB, a depth of 6, learning rate of 0.05, and 500 estimators; for RF, a maximum depth of 10 with 400 trees; and for MLP, a hidden layer size of 128, a learning rate of 0.001, and the Adam optimizer.

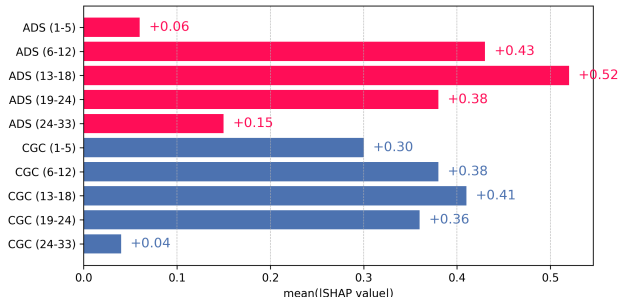

Figure 11. Mean absolute SHAP values for ADS and CGC across layers.

7. Dataset curation

In order to construct a hallucination detection dataset, instead of following previous papers to use CHAIR [22], we used GPT-4o API to extract hallucinated words. This is because the CHAIR toolkit often misses or return excessive words in case of ambiguity. The prompt structure we used is as follows 11:

It is possible that an object word can consist of more than 1 token, therefore we calculate the token-level attribute by indexing the object word, and obtain a 'prefix' which is the sentence prior to that word. Then the VLMs will forward once based on the 'prefix', yielding the internal attributes of the object token.

8. Ablation Studies

We keep the classifier fixed (LLaVA features + MLP) and vary which layer combinations are included. The results for our two features are shown in Tab. 6. We observe that, typically the middle layers (around 12-24) are the layers with richest semantic features and best alignment between text and images, yielding the best results. Meanwhile, the last layers are usually used to refine the language predictions of the Language Encoder, rendering visual information to be weaker and leading to poor classification results.

9. Feature Importance

We compute the mean SHAP values of layer-wise ADS and CGC features, for LLaVA-1.5 + MLP classifier. The result is displayed with Fig 11. We observe the findings is consistent with 8, where features from the middle layers are the most influential for cross-modal attention and alignment, meanwhile for the final layers these information are no longer encoded, leading to poor performance and low

Layer Set	ADS		CGC	
	AUC	F1	AUC	F1
Layers 6–12	72.38	67.88	76.51	73.26
Layers 13–18	69.17	62.91	73.34	70.55
Layers 19–24	68.31	63.74	68.79	63.44
Layers 25–33	64.58	62.88	75.44	74.34
All layers	**85.84**	**80.36**	**84.41**	**78.90**

Table 6. Layer-wise ablation of ADS and CGC features.

predictive importance.

The overthinking-based CGC and ADS features dominate the contribution, confirming that they encode the most discriminative signal.

10. Combining Our Features With SVAR

Model	Features	AUC	F1
LLaVA-1.5	SVAR	85.12	69.35
	+CGC	87.72	73.52
	+ADS	89.04	74.03
	+CGC+ADS	**89.41**	**74.07**
Qwen 2.5-VL-8B	SVAR	87.85	76.38
	+CGC	86.79	76.24
	+ADS	89.11	79.12
	+CGC+ADS	**90.28**	**81.09**
InternVL-2.5-8B	SVAR	86.21	76.31
	+CGC	86.39	79.43
	+ADS	88.02	81.03
	+CGC+ADS	**89.54**	**81.56**

Table 7. Effect of adding CGC and ADS features to hallucination detectors across three VLMs.

We combine our metrics (ADS + CGC) with the prior attention-based detection method, SVAR. Specifically, we add all of our proposed features alonside with SVAR's original features, and train a MLP as a classifier. Results are shown in Tab. 7. We observe that adding our crafted features help yield upto 5 $\%$ in performance gain, proving the effectiveness of our method, and also demonstrate how our proposed metrics can be applied along with other classifiers.

11. Ablation Studies

We report the sensitivity of our detector to two important threshold parameter choices: Top-$x\%$ (Equation 1, page 5), which determines how many attention patches we recognize as object patches with 8-connected components for ADS, and Top-k for CGC (Equation 7), which are the percentage

ADS			**CGC**		
Top-$x\%$	**AUC**	**F1**	**Top-$k\%$**	**AUC**	**F1**
5%	81.65	76.54	1%	81.36	75.31
10%	**83.86**	**76.83**	3%	82.55	76.54
15%	82.87	77.65	**5%**	**83.86**	**76.83**
20%	81.65	76.07	10%	82.57	77.78
30%	79.83	74.70	20%	81.82	72.96

Table 8. Sensitivity of ADS foreground threshold and CGC top-$k\%$ patch selection on LLaVA-1.5-7B. Each side varies one hyperparameter while fixing the other at its default. Bold indicates default.

of high semantic similarity areas that we aggregate. The experiments are conducted on LLaVA-1.5-7B.As observed in Table 8, we see that for the value around 5-30 $\%$, the detector manages to achieve a stable performance of around 80-85 $\%$ in AUC and around 75-80 $\%$ in F1 score. The performance starts to degrade after 15%, since selecting too many patches can lead to attention noises and sinks being selected, worsening the discriminative signals. The same phenomenon is observed for CGC, where the accuracy is stable from around 1-10%, but deteriorates heavily to a F1 of 72.96 when we select 20 $\%$ of the most semantically similar patches.

Prompt 11.1: LLM-based Hallucination Detector

System Prompt:
You are a precise hallucination detector. Follow the instructions exactly and output ONLY a JSON list.

User Prompt:

You are given:
- A list of ground truth object classes (from COCO).
- A detailed description of an image.
- Several captions of the same image.

Your task:
Find all object classes that are mentioned in the description, but are NOT mentioned in any of the captions, and are NOT present in the ground truth list.
Output the result as a list as in the examples. Do NOT add any extra text or provide any explanations.

Examples:
Objects: ["bowl", "broccoli", "carrot"]
Description: There are two bowls of food, one containing a mix of vegetables, such as broccoli and carrots, and the other containing meat.
Captions:
- A bowl with broccoli and carrots.

→ Output: ["meat"]

Objects: ["bowl", "broccoli"]

Description: A bowl full of broccoli.
Captions:
- A bowl of green vegetables.

→ Output: []

Now answer:
Objects: {objects}
Description: {description}
Captions:
{captions_formatted}
→ Output: