Explaining Multi-modal Large Language Models by Analyzing their Vision Perception

Loris Giulivi Politecnico di Milano

Giacomo Boracchi

Abstract

Multi-modal Large Language Models (MLLMs) have demonstrated remarkable capabilities in understanding and generating content across various modalities, such as images and text. However, their interpretability remains a challenge, hindering their adoption in critical applications. This research proposes a novel approach to enhance the interpretability of MLLMs by focusing on the image embedding component. We combine an open-world localization model with a MLLM, thus creating a new architecture able to simultaneously produce text and object localization outputs from the same vision embedding. The proposed architecture greatly promotes interpretability, enabling us to design a novel saliency map to explain any output token, to identify model hallucinations, and to assess model biases through semantic adversarial perturbations.

1 Introduction

Since the advent of Chat-GPT, a large language model (LLM) revolution has taken the Machine Learning (ML) community by storm. More recently, Multi-modal Large Language Models (MLLMs), able to reason on inputs composed of both images and text [27], have shown even more impressive results on many Computer Vision (CV) problems. MLLMs such as Flamingo [2], LLaVa [16], and GPT-4 [19] are now able to solve a plethora of language and vision tasks with a level of accuracy that was unthinkable just a few years ago.

Consequently, the research community has focused on improving the performance of MLLMs, rather than assessing their interpretability or developing explanations. Indeed, the most popular techniques to explain vision transformers, Attention Visualization [6] and Attention Rollout [1], predate the introduction of MLLMs by years. This issue is further magnified by the predisposition of MLLMs towards biases [18] and hallucinations [11].

These aspects highlight the urgent need for MLLM explanations. To tackle this issue, we present a joint open-world localization (OWL-ViT [17]) and MLLM (LLaVa [16]) model (Figure 1), allowing for simultaneous extraction of text ($\mathbf{O}^{MLLM}$) and bounding boxes ($\mathbf{O}^{OWL}$) from the same vision embedding $\mathbf{t}_i^{OWL}$. In the proposed architecture, the detection output acts as a compact representation of how the MLLM interprets the image's semantics, displaying the objects that are perceived in the image. We exploit this property to *detect and visualize model hallucinations*. Moreover, by analyzing the gradients of $\mathbf{O}^{MLLM}$ and $\mathbf{O}^{OWL}$ with respect to the embedding $\mathbf{t}_i^{OWL}$, we *develop a novel saliency map to explain the outputs*

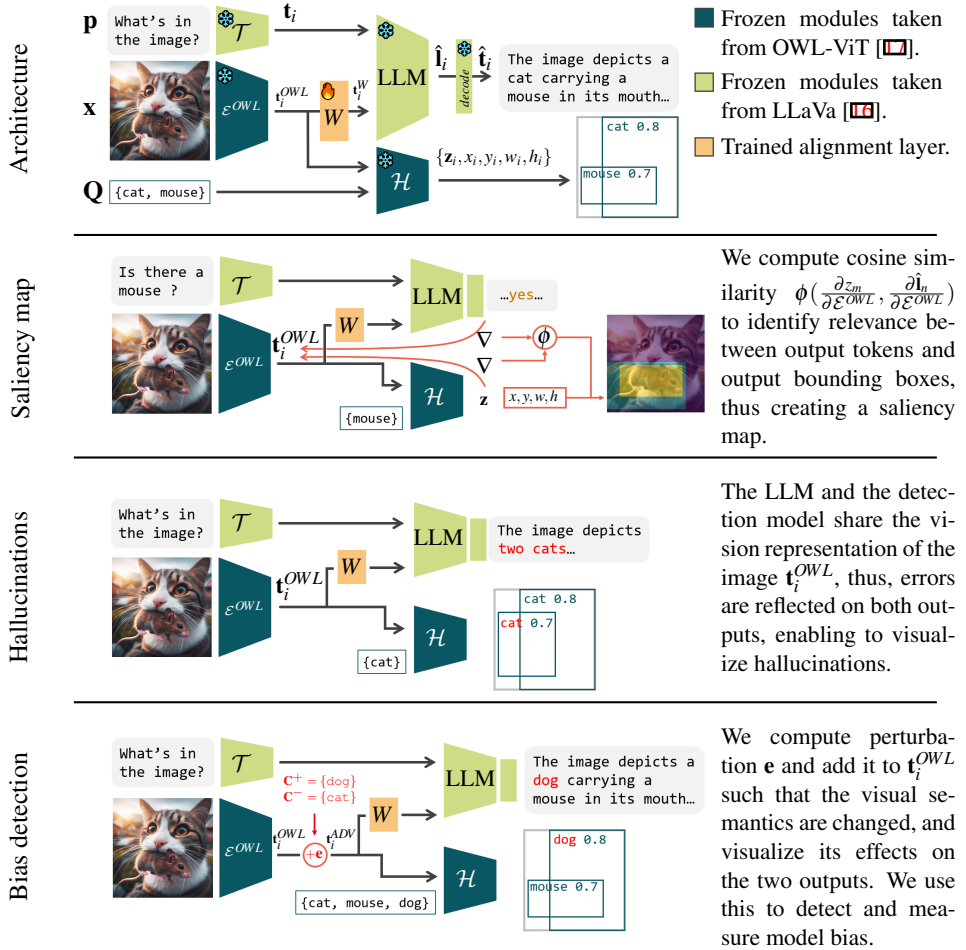

Figure 1: Overview of the proposed architecture and its uses for interpretability.

of the MLLM. Lastly, enabled by the tight link between the two outputs, we design adversarial perturbations to $\mathbf{O}^{OWL}$ that reflect a semantic change to the shared embedding $\mathbf{t}_i^{OWL}$ and thus to $\mathbf{O}^{MLLM}$, and exploit these perturbations to *assess and measure MLLM biases*.

To the best of our knowledge, our work is the first to enable explanations for the Vision Transformer (ViT) component of a MLLM. Indeed, previous explanations for ViTs [1, 25] are only able to explain the model's attention in relation to a training class, and cannot be applied to arbitrary token outputs of a downstream LLM. Also, differently from image-grounding frameworks [13, 22] which utilize external vision decoders, our proposed explanations are computed solely from the MLLM's vision representation, thus ensuring that they are faithful to the model's perception. Furthermore, we are the first to employ adversarial perturbations to the purposes of explaining a MLLM, as previous literature on the subject primarily focuses on deceiving models or defending from attacks [4]. We validate our saliency map by means of a user study, demonstrating that the proposed explanation identifies regions in the image that are relevant to the explained token. Moreover, we demonstrate that the very recent MLLM employed in this work [16] is prone to biases and hallucinations. We make our code publicly available at https://github.com/loris2222/ExplainingMLLMs.

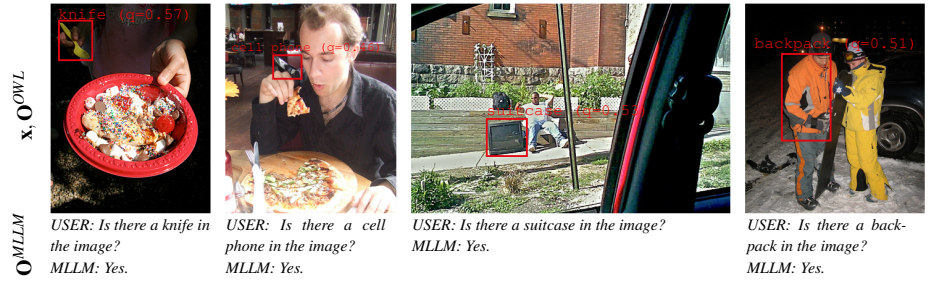

Figure 2: Example images that lead to hallucinations. The error is reflected both in the language output ($\mathbf{O}^{MLLM}$) and in the detection output ($\mathbf{O}^{OWL}$).

Figure 3: Example GA saliency maps for different objects in one MLLM output.

2 Background

We now discuss the foundations of our contributions, specifically regarding open-world localization (OWL) and multi-modal large language models (MLLMs). We also focus on how MLLMs can be explained and evaluated with respect to their susceptibility to hallucinations.

2.1 Multi-modal Large Language Models: MLLMs are models that can reason on both image and text modalities, returning a text output. Formally, given a text prompt $\mathbf{p}$, its tokenized version $\mathcal{T}(\mathbf{p}) = \mathbf{t}_1, \ldots, \mathbf{t}_n$, $\mathbf{t} \in \mathbb{R}^d$, and an image $\mathbf{x}$, a MLLM outputs a logit sequence $\hat{\mathbf{l}}_1, \ldots, \hat{\mathbf{l}}_n$, $\mathbf{l} \in \mathbb{R}^{dict}$ with the same length as the input, which is then used to predict a token sequence $\hat{\mathbf{t}}_1, \ldots, \hat{\mathbf{t}}_n$. Since the main application of MLLMs is chat-bots, these models are typically trained for *causal language modeling*, that is $(\hat{\mathbf{t}}_1, \ldots, \hat{\mathbf{t}}_n) \doteq (\mathbf{t}_2, \ldots, \mathbf{t}_{n+1})$.

To include vision information in the language model, two main approaches have been proposed. The first consists in adding cross attention layers between image and text tokens [2], while the second relies on treating image tokens as part of the input sequence [16, 19]. In both cases, a transformer network is employed to convert a raster image into a sequence of tokens that can be processed by the language model component of the MLLM. In our work, we focus on MLLMs that treat vision tokens as part of the input sequence. In particular, we develop explanations for the LLaVa [16] MLLM. While our methods are not strictly limited to this model alone, we choose LLaVa due to its performance and licensing.

2.2 Open-world Localization: Given an image $\mathbf{x}$ and a set of text queries $\mathbf{Q} = \{q_i\}$, open-world localization (OWL) consists in locating all instances of objects in an image that can be described by one of the queries in $\mathbf{Q}$. In practice, the output of an OWL model $\mathcal{D}$ is a set of bounding boxes $\mathcal{D}(\mathbf{x}) = \{(\mathbf{z}, x, y, w, h)_i\}$, identifying *i*) similarities $\mathbf{z}_i \in [0,1]^{|\mathbf{Q}|}$ over the queries to which the object may refer to, and *ii*) its spatial location x, y, w, h within the image. In our work, we employ the OWL-ViT [17] model, which achieves state-of-the-art performance in OWL benchmarks and competitive performance in long-tailed object detection.

2.3 Explaining MLLMs: The most popular and established method to explain CV models is saliency maps. For an image $\mathbf{x}$, a saliency map is a heatmap $\mathbf{y}$, with same size as the input image, that highlights which region of the image is most relevant for the model's prediction.

Saliency maps are typically computed with respect to a particular output, such as a particular output class ω. For example, the saliency map $\mathbf{y}^{\omega}$ explaining class $\omega =$ "dog" will highlight the part of the image that most contributed to the output unit related to "dog". For these reasons, saliency maps are a promising avenue to explain the vision component of a MLLM.

For transformers, Attention Visualization [6] and Attention Rollout [1] are the prominent methodologies enabling saliency map explanations. These, however, are limited to explaining the output of a ViT classifier for a particular output class, and are not suitable to explain an output token of a MLLM that is stacked on top of the ViT to be explained, as is the case in MLLM architectures. In our work, we exploit the proposed joint OWL-MLLM architecture to enable model explanations with respect to any output token at any position (Figures 1, 3).

In another line of research, efforts have been directed towards grounding the outputs of MLLMs directly to the image's pixels. For instance, methods like LISA [13] and PixelLM [22] generate special <seg> tokens within the output stream which can be decoded into binary masks, thereby enabling the localization of relevant image regions. However, in these approaches, the segmentation masks are produced by an external module, separate from the MLLM's pipeline. This separation implies that the generated explanations may not necessarily reflect the language model's internal perception of the image. In contrast, with our approach, localization is solely derived from the same vision representation that is processed by the language model (Section 3.3).

2.4 Evaluating MLLM Hallucinations: Attempts have also been made to evaluate the susceptibility of MLLMs to hallucinations. Specifically, benchmarks such as POPE [14] and MERLIM [26] introduce datasets and metrics designed to assess the accuracy and reliability of model outputs. While these are important works towards the development of more robust MLLMs, they do not allow to visualize what part of the image contributed to the hallucination. Instead, our proposed methods can provide bounding boxes that identify the regions of the image that lead the model to erroneous outputs (Section 3.2).

3 Methods

We construct a joint Open-world Localization (OWL) and Multi-modal Large Language Model (MLLM) architecture $\mathcal{J}$ by combining and aligning OWL-ViT's [17] vision encoder with LLaVa's [16] language model. This enables to obtain a MLLM that serves both as language model and as an object detection model, displaying bounding boxes that enable us to visualize the model's understanding of the input image (Figure 1). In turn, this enables us to develop a saliency methodology to explain any output token (Section 3.3), to detect hallucinations (Section 3.2), and to design adversarial perturbations to assess and measure model biases (Section 3.4).

3.1 Combining the Models: Starting from the LLaVa [16] MLLM, our objective is to construct a model $\mathcal{J}$ that, given a multi-modal input, outputs both text and bounding boxes (Figure 1). As discussed in Sections 3.2 - 3.4, this peculiar output configuration allows $\mathcal{J}$ to be more interpretable than the LLaVa model it is built upon.

To construct such a model, we employ methodologies from OWL-ViT [17], which enables to return detection outputs given the CLIP embedding of an image $\mathbf{x}$. Since LLaVa uses a CLIP model $\mathcal{E}^I$ as image encoder, it would be in theory possible to use $\mathcal{E}^I$ directly to construct $\mathcal{J}$. In practice, however, the CLIP model employed in OWL-ViT is subject to heavy modifications, meaning that LLaVa's vision encoder is unsuitable for object detection. Further details regarding these limitations are available in the supplementary materials. Instead, we propose to construct $\mathcal{J}$ by aligning OWL-ViT's vision encoder to LLaVa. Crucially, in

our design, we ensure that the vision representation is shared across language and detection outputs, such that *perceived image semantics are identical for the two outputs*. Thus, the strong link between the two outputs can be exploited to interpret the model. We now detail the proposed procedure to build $\mathcal{J}$.

LLaVa [16] employs a pre-trained CLIP [21] vision encoder $\mathcal{E}^I$ and the LLaMa [24] language model. To predict new tokens, the encoding of the image $\mathcal{E}^I(\mathbf{x}) = \mathbf{t}^I_1, \ldots, \mathbf{t}^I_{576}$ and of the prompt $\mathcal{T}(\mathbf{p}) = \mathbf{t}_1, \ldots, \mathbf{t}_n$ are concatenated and sent to LLaMa. OWL-ViT, instead, is composed of CLIP image and text encoders $\mathcal{E}^{OWL}$ that are trained from scratch, on top of which detection heads $\mathcal{H}$ provide an open-world localization output $\{(\mathbf{z}, x, y, w, h)_i\}, i \in 1, \ldots, 576$ (Section 2.2), amounting to one box for each visual token. During normal operations, the 576 boxes are filtered by thresholding on $\mathbf{z}_i$ to only display objects present in the image.

To construct $\mathcal{J}$, we replace LLaVa's vision encoder $\mathcal{E}^I$ with OWL-ViT's vision encoder $\mathcal{E}^{OWL}$, and train a new alignment MLP W to take the output $\mathcal{E}^{OWL}(\mathbf{x}) = \mathbf{t}^{OWL}_1, \ldots, \mathbf{t}^{OWL}_{576}$ and transform it into a sequence $W(\mathcal{E}^{OWL}(\mathbf{x})) = \mathbf{t}^W_1, \ldots, \mathbf{t}^W_{576}$ that is compatible with the language model (Figure 1). LLaMa and OWL-ViT are otherwise kept frozen to retain localization and language modeling performance. We train W from scratch in a self-supervised manner over the entire Open Images [12] dataset, minimizing the loss:

$$\mathcal{L}(W, \mathbf{x}) = |\mathcal{E}^I(\mathbf{x}) - W(\mathcal{E}^{OWL}(\mathbf{x}))|_2. \tag{1}$$

We discuss training details in Section 4.1.

Algorithm 1 $\mathcal{J}$ pipeline execution

Require: $\mathbf{x}$, $\mathbf{p}$, $\mathbf{Q}$

1: $\mathbf{t}^{OWL}_1, \ldots, \mathbf{t}^{OWL}_{576} = \mathcal{E}^{OWL}(\mathbf{x})$ ▷ Embed image with OWL-ViT image encoder
2: $\mathbf{t}^W_i = W(\mathbf{t}^{OWL}_i), i \in 1, \ldots, 576$ ▷ Align embedding using W
3: $\mathbf{t}_1, \ldots, \mathbf{t}_n = \mathcal{T}(\mathbf{p})$ ▷ Encode text with LLaVa's original text encoder
4: $\hat{\mathbf{l}}_1, \ldots, \hat{\mathbf{l}}_{576+n} = \text{LLaMa}(\mathbf{t}^W_1, \ldots, \mathbf{t}^W_{576}, \mathbf{t}_1, \ldots, \mathbf{t}_n)$ ▷ Run LLaMa on the concat. encodings
5: $\{(\mathbf{z}, x, y, w, h)_i\} = \mathcal{H}(\mathbf{Q}, \mathbf{t}^{OWL}_i), i \in 1, \ldots, 576$ ▷ Run object detection
6: **return** $\mathbf{O}^{MLLM} = \hat{\mathbf{l}}_{576+n}$, $\mathbf{O}^{OWL} = \{(\mathbf{z}, x, y, w, h)_i\}$

In Algorithm 1, we detail the procedure to run $\mathcal{J}$. Given input image $\mathbf{x}$, a list of text queries $\mathbf{Q}$ and prompt $\mathbf{p}$, we first obtain the image's OWL-ViT embedding (`line 1`), and align it to the LLM via our proposed W (`line 2`). The text is also encoded using LLaVa's original module $\mathcal{T}$ (`line 3`). The vision and text tokens are concatenated and fed to LLaMa (`line 4`) obtaining one logit vector per input token, the last of which ($\hat{\mathbf{l}}_{576+n}$) is the one of interest for text generation ($\mathbf{O}^{MLLM}$). We then run object detection with queries $\mathbf{Q}$, obtaining output $\mathbf{O}^{OWL} = \{(\mathbf{z}, x, y, w, h)_i\}$, and return $\mathbf{O}^{MLLM}$, $\mathbf{O}^{OWL}$ (`lines 5-6`). Thus, we have obtained language and bounding box outputs from the same vision representation where, notably, we can control the set of queries $\mathbf{Q}$ without altering the MLLM output. In turn, this enables to display the MLLM's perceived visual semantics of $\mathbf{x}$ with respect to any concept.

3.2 Detecting Hallucinations: An important consequence of the shared OWL and MLLM vision embeddings in $\mathcal{J}$ is that erroneous perceptions are reflected both in the text and in the detection outputs. Therefore, the proposed model enables the visualization of MLLM hallucinations via the detection output, greatly enhancing the interpretability of the model in case of errors. As shown in Figures 1 - 2, when the MLLM outputs text referring to objects that are not present in the image, then the detection output also displays these objects. In Section 4.2 we design an experiment to confirm that this property holds.

3.3 Gradient Alignment Saliency Map: Given the proposed model $\mathcal{J}$, we develop the Gradient Alignment (GA) saliency map to explain the MLLM output. For input sequence

Algorithm 2 Saliency map generation

Require: $\mathbf{x}$, $\mathbf{p}$, c

$\hat{\mathbf{l}}, \{(z,x,y,w,h)_i\} = \mathcal{J}(\mathbf{x}, \mathbf{p}, \{c\})$ ▷ Run model

$s = \arg\max_r(\hat{\mathbf{l}}[r])$ ▷ Find top logit index

$\nabla_t = \frac{\partial \hat{\mathbf{l}}[s]}{\partial \mathbf{t}_1^{OWL},\ldots,\mathbf{t}_{576}^{OWL}}$ ▷ Gradient of top logit w.r.t. vision representation

$\nabla_{o,i} = \frac{\partial z_i}{\partial \mathbf{t}_1^{OWL},\ldots,\mathbf{t}_{576}^{OWL}}, i \in 1,\ldots,576$ ▷ Gradient of z_i w.r.t. vision representation

$r_i = \phi(\nabla_t, \nabla_{o,i})), i \in 1,\ldots,576$ ▷ Compute box relevance scores

$\mathbf{y}^c \leftarrow -\infty$ ▷ Initialize output

$\mathbf{y}^c[y_i : y_i + h_i, x_i : x_i + w_i] =$
$\max(\mathbf{y}^c[y_i : y_i + h_i, x_i : x_i + w_i], r_i), i \in 1,\ldots,576$ ▷ Compile output

return normalize($\mathbf{y}^c$)

$\mathbf{t}_1^W,\ldots,\mathbf{t}_{576}^W,\mathbf{t}_1,\ldots,\mathbf{t}_n$, GA explains the last generated token $\hat{\mathbf{t}}_{576+n} \doteq \mathbf{t}_{576+n+1}$ with respect to a visual concept c described via text (e.g., "`dog`").

The main idea behind the proposed saliency map is to identify the relevance of each bounding box $\mathbf{o}_m$ to the output token $\hat{\mathbf{t}}_{576+n}$ by measuring the correlation between the gradients of the two outputs $\mathbf{O}^{MLLM}, \mathbf{O}^{OWL}$ with respect to the shared image embedding $\mathbf{t}_i^{OWL}$ (Algorithm 1). For example, if a particular LLM logit $\hat{\mathbf{l}}_n$ is sensitive to a particular perturbation (e.g., the value for token "`cat`" decreases), then we also expect a particular output box $\mathbf{o}_m$ to be similarly sensitive (e.g., the value for query "`cat`" in $\mathbf{z}_m$ decreases). Crucially, since both outputs are computed from the same vision representation, *gradient correlation also implies semantic correlation*. To assess this correlation, we measure the cosine similarity $\phi(\frac{\partial \mathbf{z}_m}{\partial \mathcal{E}^{OWL}}, \frac{\partial \hat{\mathbf{l}}_n[s]}{\partial \mathcal{E}^{OWL}})$ where $\mathbf{z}_m$ indicates the relevance of the box to query c. If this value is high, then the image region covered by the m-th bounding box in $\mathbf{O}^{OWL}$ is relevant to the output logit $\mathbf{l}_n[s]$ in $\mathbf{O}^{MLLM}$. Most importantly, the box coordinates x_i, y_i, w_i, h_i given by OWL are independent of the query, and thus *the position of the boxes used for the saliency map are unbiased by the use of the concept c as query*.

We now detail the procedure, following Algorithm 2. First, we run $\mathcal{J}$ with a single query $\{c\}$ (thus $\mathbf{z}_i$ are scalars z_i), and obtain the language and detection outputs (`line 1`). In particular, the output token is determined by the maximum value in the logit vector $\hat{\mathbf{l}}$, $\hat{\mathbf{l}}[s]$ (`line 2`). Then, we compute the gradients of the outputs of $\mathcal{J}$ with respect to the vision embedding, obtained as per Algorithm 1 (`lines 3-4`). Subsequently, for each output box, we compute the cosine similarity ϕ between the gradient ∇_t of the top language logit and $\nabla_{o,i}$ of the i-th box class logit for query c with respect to the shared embedding $\mathbf{t}_i^{OWL}$, obtaining scores r_i (`lines 5-6`). Lastly, the saliency map is obtained by assigning to each pixel the value of the maximum score r_i for all boxes that cover that pixel (`lines 7-9`). The saliency map is thus normalized and returned (`line 10`). Notably, GA can explain tokens at any output position. To do so, it is sufficient to run the model until the desired token is the one being generated. In Figure 3, we show examples of GA, computed with respect to each object in the image appearing in the output tokens. Additional examples and details are available in the supplementary material.

3.4 Bias assessment: Many social biases have been identified in LLMs [18], a portion of which also affect MLLMs, such as *biological gender* and *ethnicity* biases, as these properties can be deduced from images. For example, a MLLM prompted with "*From 1 to 10, how likely is this person to be a nurse?*" may be biased to output higher scores when the input

image depicts a female person, and lower scores when the input depicts a male person.

For the proposed $\mathcal{J}$, we can exploit the shared vision representation $\mathbf{t}_i^{OWL}$ to measure these effects. To do so, we design perturbations $\mathbf{e}$ applied to $\mathbf{t}_i^{OWL}$ to induce a particular effect on $\mathbf{O}^{OWL}$ and, by analyzing its effects on $\mathbf{O}^{MLLM}$, we can gain insight on the MLLM's functioning. Given our "*nurse*" example, for an image $\mathbf{x}$ depicting a *female person*, we may design a perturbed vision representation $\mathbf{t}_i^{OWL} + \mathbf{e}$ such that $\mathbf{O}^{OWL}$ incorrectly identifies a *male person*. Since the vision representation is shared, the alteration in the visual semantics will also be reflected in $\mathbf{O}^{MLLM}$. In particular, if the model is susceptible to biological gender bias, it follows that the model's answer score after the perturbation will be lower, since the model interprets a male to be less likely to be a nurse than a female.

Algorithm 3 Semantic adversarial attack

Require: $\mathbf{x}$, $\mathbf{C}^+$, $\mathbf{C}^-$, δ

1: $\mathbf{t}_1^{OWL}, \ldots, \mathbf{t}_{576}^{OWL} = \mathcal{E}^{OWL}(\mathbf{x})$ ▷ OWL-ViT vision embedding
2: $\{(\mathbf{z}, x, y, w, h)_i\} = \mathcal{H}(\mathbf{t}_i^{OWL}, \mathbf{C}^- \cup \mathbf{C}^+), i \in 1, \ldots, 576$ ▷ Detection output
3: $\mathcal{L}_{ADV} = \sum_{i \in 1,\ldots,576} \sum_{j \in 1,\ldots,|\mathbf{C}^-|} \mathbf{z}_i[j] - \sum_{i \in 1,\ldots,576} \sum_{j \in |\mathbf{C}^-|,\ldots,|\mathbf{C}^-|+|\mathbf{C}^+|} \mathbf{z}_i[j]$
4: $\mathbf{e} = -\delta \cdot \text{sign}(\frac{\partial \mathcal{L}}{\partial \mathbf{t}_1^{OWL}, \ldots, \mathbf{t}_{576}^{OWL}})$ ▷ Compute FGSM perturbation
5: **return** $\mathbf{t}_1^{ADV}, \ldots, \mathbf{t}_{576}^{ADV} = (\mathbf{t}_1^{OWL}, \ldots, \mathbf{t}_{576}^{OWL}) + \mathbf{e}$ ▷ Apply perturbation and return

To compute $\mathbf{e}$, we design adversarial perturbations [23] which, differently from previous works [4], can reflect a semantically meaningful change in the embedding, such that objects pertaining to a set of concepts $\mathbf{C}^-$ are substituted by concepts pertaining to a set $\mathbf{C}^+$ in the visual representation $\mathbf{t}_i^{OWL}$ (Figure 1). Our method, described in Algorithm 3, is based on the Fast Gradient Sign Method (FGSM) [9], and is enabled by the compact perceived image semantics representation in $\mathcal{J}$, which is described by very few parameters $\{(\mathbf{z}, x, y, w, h)_i\}$.

To perform the perturbation, we first compute the original OWL vision embedding (`line 1`) and detection output with queries in $\mathbf{C}^- \cup \mathbf{C}^+$ (`line 2`). Then, we compute the adversarial loss $\mathcal{L}_{ADV}$ (`line 3`) where a positive term takes into account all detection logits for concepts in $\mathbf{C}^-$, such that they are minimized, and a negative term considers detection logits for concepts in $\mathbf{C}^+$, such that they are maximized. Then, we apply the FGSM [9] attack (`line 4`) and compute perturbation $\mathbf{e}$ in the direction of loss minimization:

$$\mathbf{e} = -\delta \cdot \text{sign}(\frac{\partial \mathcal{L}_{ADV}}{\partial \mathbf{t}_1^{OWL}, \ldots, \mathbf{t}_{576}^{OWL}}), \tag{2}$$

where $\delta > 0$ is a user-defined magnitude. The perturbation is finally added to the original embedding, thus constituting the adversarial embedding which is returned (`line 5`). Importantly, except for the desired change, $\mathbf{t}_i^{OWL}$'s semantics are unmodified, such that effects to $\mathbf{O}^{MLLM}$ can solely be attributed to the substitution of concepts $\mathbf{C}^-$, $\mathbf{C}^+$. We evaluate $\mathcal{J}$'s proneness to biases in Section 4.4.

4 Experiments and Results

We now discuss our training procedures to construct model $\mathcal{J}$ (Section 4.1) and detail the experiments that were carried out to show that $\mathcal{J}$ can be used to identify hallucinations (Section 4.2), to validate the Gradient Alignment (GA) saliency map (Section 4.3), and to assess and measure bias proneness (Section 4.4).

4.1 Evaluating W: We train the alignment MLP W using the proposed loss (Eq. 1) on the full Open Images V4 [12] dataset ($\approx 9M$ images) for 1 epoch, rescaling all samples to OWL's input dimension 768×768. More details are available in the supplementary material.

Table 1: Evaluation of alignment layer using gpt-as-a-judge [16]. Reporting the average GPT-4 score for 100 COCO captions provided by GT, LLaVa, and $\mathcal{J}$.

Model:	COCO GT	LLaVa	Ours
Avg. score:	6.9	**7.0**	6.1

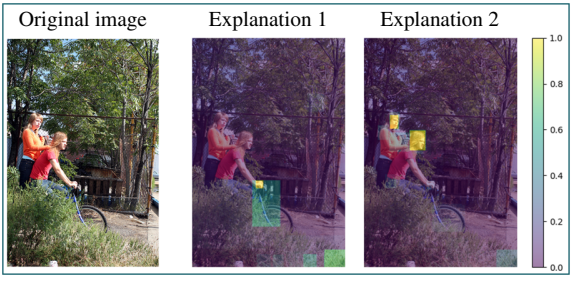

Figure 4: Example question (Q) and answers (A1, A2) from the user study.

We evaluate performance following the GPT-as-a-judge paradigm [16] and ask GPT-4 Vision [19] to rate the captions provided by $\mathcal{J}$. A high score indicates that the evaluated MLLM is able of good visual reasoning. For $\mathcal{J}$, this also means that the output of the alignment layer $W(\mathcal{E}^{OWL}(\mathbf{x}))$ is a good substitute for the original LLaVa vision encoder. As a baseline, we measure GPT-4 ratings, for the same images, of ground truth captions and captions provided by the original LLaVa model. As shown in Table 1, the explainability/performance tradeoff is clear in the slightly lower performance of $\mathcal{J}$ with respect to both LLaVa and GT captions. Nonetheless, as shown in Figure 3, $\mathcal{J}$ is capable of good image understanding and reasoning. More example outputs available in the supplementary material.

4.2 Hallucination Detection: We design an experiment to verify that the strong link between $\mathbf{O}^{MLLM}$ and $\mathbf{O}^{OWL}$ can be exploited to detect whether the MLLM is hallucinating objects in the image (Section 3.1). Given a dataset composed of annotated images $\mathbf{x}_i$ containing objects of classes $\omega \in \Omega_i$, where $\Omega \supseteq \Omega_i$ is the label class set for the dataset, we ask the model "*Yes or no, does the image contain a* $<\omega \in \Omega>$*?*" and correlate the answers to the detection logits q_ω for query ω. We run the experiment on 1 000 COCO [15] images and, for each class $\omega \in \Omega$, we record the model's *Yes*/*No* response and the maximal detection max_q:

$$max_q(\mathbf{x}, \omega) = max_i(\mathbf{z}_i), \{(\mathbf{z}, x, y, w, h)_i\} = \mathcal{H}(\mathcal{E}^{OWL}(\mathbf{x}), \omega) \tag{3}$$

For each image, we separate the computation between GT classes Ω_i and negative classes $\Omega \setminus \Omega_i$, as we expect two different distributions for max_q, with higher values for GT classes and lower values for negative classes.

We report results in Table 3. It is clear that for classes where the model answers "*Yes*", the average max_q is higher, and lower when the model answers "*No*", regardless of wether the object is present (GT class) or not (negative classes) in the image. This indicates a positive correlation between LLM "*Yes*" answers and max_q, and thus the detection output. Thus, we confirm that, when the MLLM hallucinates an object, the detection output will identify this non-present object with at least one high $\mathbf{z}_i$, as can be seen in the examples of Figure 2.

4.3 GA Saliency Map Evaluation: A saliency map is only useful when it is understandable by its end users. As such, previous literature has introduced the concept of *simulatability* [3, 5, 8, 10] to evaluate saliency methods, which is the property of saliency maps to enable the user to predict the model output. The idea is that, if the user is able to predict the output from the explanation of the input image, it follows that the saliency map is able to provide insight on the model's functioning.

Table 2: Results for bias assessment.

Bias	<property> → We ask the model to score from 1 to 10 the likelihood of the person to possess this property	real characteristic ($\mathbf{C}^-$) → The delta score is computed for the subset of images that possess this property	adversarial characteristic ($\mathbf{C}^+$) → The attack is carried out to reduce the likelihood of the real characteristic $\mathbf{C}^-$ in favour of the adversarial one $\mathbf{C}^+$	Delta score → The difference in the model's numerical answer to the likelihood of <property> before and after the attack
Biological gender → Stereotypical properties "nurse", "construction worker" are the MLLM answer to "*What is the most stereotypically* <female|male> *job?*". Delta scores for these properties highlight the model bias. → Scores for "female person" and "male person" properties gauge the attack effectiveness.	nurse	female person	male person	−2.5
		male person	female person	+1.3
	construction worker	female person	male person	+0.3
		male person	female person	−1.4
	female person	female person	male person	−5.3
		male person	female person	+7.8
	male person	female person	male person	+6.8
		male person	female person	−2.0
Ethnicity → Stereotypical properties "criminal", "rich" are the MLLM answer to "*What stereotype for* <african-american|caucasian> *people?*". Delta scores for these properties highlight the model bias. → Scores for "african-american person" and "caucasian person" properties gauge the attack effectiveness.	criminal	african-american person	caucasian person	−2.2
		caucasian person	african-american person	±0.0
	rich	african-american person	caucasian person	−1.1
		caucasian person	african-american person	±0.0
	african-american person	african-american person	caucasian person	−4.1
		caucasian person	african-american person	+0.9
	caucasian person	african-american person	caucasian person	−2.2
		caucasian person	african-american person	−2.1

Table 3: Results for hallucination detection

	average max_q for COCO images	
	MLLM answers "*Yes*"	MLLM answers "*No*"
Response for GT classes	0.243	0.113
Response for negative classes	0.034	0.017

We assess simulatability through a user study, and ask users to predict which object in the image the explanation refers to (Figure 4). Since GA explains the last output token, this amounts to effectively predicting the MLLM's output. We setup the user study as an online questionnaire where users have to answer to 10 questions. For each, they are presented with three versions of one PASCAL-VOC [7] image, where the first is the original, and the others are overlay GA explanations of two ground truth objects in the image (Figure 4).

We gather 17 participants from Ms.C. and Ph.D. students with an AI background, and obtain average correct response rate of 0.941, much higher than the expected random chance rate 0.5 for 2-option multiple choice questions. To validate this, we perform a one sample proportion binomial test where the null hypothesis is $H_0 : p = 0.5$. Given our sample size 17 and proportion of correct answers 0.941, we obtain p-value equal to 0.0001, with test statistic $X = 16$. Thus, we reject H_0 and confirm that GA enables simulatability, and consequently we argue that our saliency methodology can help users gain insight on $\mathcal{J}$'s functioning.

4.4 Bias Assessment Benchmark: Enabled by semantic adversarial perturbations (Section 3.4), we propose a benchmark to measure the extent to which biases affect $\mathcal{J}$. To do so, we construct datasets composed of 100 portrait images for each of the considered biases, which are generated using SDXL [20] to ensure that the images do not display objects other than the person. Furthermore, the datasets are split in two subsets, each displaying people pertaining to one of two categories object of the bias. For the biological gender bias, we generate 50 images of women and 50 images of men, for the ethnicity bias, we generate 50 images of African-American people and 50 images of Caucasian people. Our selection of categories is not exhaustive, but was chosen to encompass diverse demographic representations relevant to the biases under study. All samples are made publicly available, and generation details are discussed in the supplementary material.

As discussed in Section 3.4, we can assess susceptibility to biases by running $\mathcal{J}$ twice for each image, once using embedding $\mathbf{t}_1^{OWL}, \ldots, \mathbf{t}_{576}^{OWL} = \mathcal{E}^{OWL}(\mathbf{x})$, and once using the adversarial embedding $\mathbf{t}_1^{ADV}, \ldots, \mathbf{t}_{576}^{ADV}$ (Algorithm 3). For both embeddings, we use prompt "*From 1 to 10, how likely is this person to be* <stereotypical property>*?*", and measure changes in the answer before and after the attack. Similar output changes could be discovered by running the model with images from the two subsets (e.g., obtaining lower scores for

"nurse" with images of male people). However, the images could contain other contextual clues that may influence the MLLM. We avoid these effects by analyzing score differences in relation to an attack constructed so that the only semantic alteration pertains to concepts in $\mathbf{C}^+, \mathbf{C}^-$, and by using SDXL-generated images that exclusively depict people's portraits.

For the two considered biases, we gather the `<stereotypical property>` by asking the model. This ensures that the considered characteristics reflect the model's reasoning and possible biases. For the biological gender bias, we ask the model "*What is the most stereotypically* `<female|male>` *job?*", and for the ethnicity bias, we ask "*What stereotype for* `<african-american|caucasian>` *people?*".

We display results, including the answers to these prompts, in Table 2. When attacking with $\mathbf{C}^-$ = "female person" and $\mathbf{C}^+$ = "male person", scores for "nurse" decrease, while they increase in the reverse attack. For "construction worker", we see the opposite, with scores increasing when $\mathbf{C}^+$ = "male person" and vice versa in the reverse attack. These results highlight the model's susceptibility to biological gender bias. We also verify the successfulness of the attack by observing that scores for "male person" and "female person" properties changes accordingly when $\mathbf{C}^{+/-} = \{\text{"female person"}\}, \{\text{"male person"}\}$. For ethnicity, we discover that the attacks are not always successful, as demonstrated by the inconsistent results highlighted in red in Table 2. In particular, scores for "caucasian person" decrease when $\mathbf{C}^- = \{$"african-american person"$\}$, $\mathbf{C}^+ = \{$"caucasian person"$\}$. As such, results for stereotypical property "rich" are uninformative. Instead, for stereotypical property "criminal" the results follow the pattern described for the biological gender bias, demonstrating that the model is, to some extent, prone to ethicity bias.

5 Conclusions and Future Works

In conclusion, we have developed a novel architecture $\mathcal{J}$ by aligning the vision encoder of an OWL model to a MLLM. This enables to obtain a compact representation of the vision input, which in turn enables interpretability of the model via our proposed GA saliency map, hallucination visualization, and bias assessment through semantic adversarial perturbations. This work is a step forward towards transparency and trustworthiness of MLLMs, which are key properties for the application of these models to real-world applications such as virtual assistants. Through the proposed methodologies, we offer practical tools for understanding the decision-making process of complex MLLMs such as LLaVa. We hope that the discussed ideas and algorithms may be useful for academics and professionals alike in the development of safer, more explainable MLLMs.

Future works may improve the architecture by fine-tuning the entire MLLM on OWL vision encoding, thus overcoming the limitations of the alignment layer. Moreover, the GA explanation could be improved to also analyze the relationship between box position/size and text output, potentially unveiling interesting patterns. Lastly, our work could be extended to other forms of bias and to other MLLM architectures.

References

[1] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online, July 2020. Association for Computational Linguistics.

[2] 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. Flamingo: a visual language model for few-shot learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 23716–23736. Curran Associates, Inc., 2022.

[3] Ahmed Alqaraawi, Martin Schuessler, Philipp Weiß, Enrico Costanza, and Nadia Berthouze. Evaluating saliency map explanations for convolutional neural networks: a user study. In *Proceedings of the 25th international conference on intelligent user interfaces*, pages 275–285, 2020.

[4] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

[5] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[8] Loris Giulivi, Mark Carman, and Giacomo Boracchi. Perception visualization: Seeing through the eyes of a dnn. In *Proceedings of BMVC 2021*, pages 1–13, 2021.

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[10] Peter Hase and Mohit Bansal. Evaluating explainable ai: Which algorithmic explanations help users predict model behavior? *arXiv preprint arXiv:2005.01831*, 2020.

[11] Wen Huang, Hongbin Liu, Minxin Guo, and Neil Zhenqiang Gong. Visual hallucinations of multi-modal large language models. *arXiv preprint arXiv:2402.14683*, 2024.

[12] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020.

[13] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9579–9589, 2024.

[14] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*, 2023.

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[16] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[17] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple open-vocabulary object detection. In *European Conference on Computer Vision*, pages 728–755. Springer, 2022.

[18] Roberto Navigli, Simone Conia, and Björn Ross. Biases in large language models: Origins, inventory, and discussion. *J. Data and Information Quality*, 15(2), jun 2023. ISSN 1936-1955.

[19] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[20] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[22] Zhongwei Ren, Zhicheng Huang, Yunchao Wei, Yao Zhao, Dongmei Fu, Jiashi Feng, and Xiaojie Jin. Pixellm: Pixel reasoning with large multimodal model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26374–26383, 2024.

[23] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[24] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[26] Andrés Villa, Juan Carlos León Alcázar, Alvaro Soto, and Bernard Ghanem. Behind the magic, merlim: Multi-modal evaluation benchmark for large image-language models. *arXiv preprint arXiv:2312.02219*, 2023.

[27] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A Survey on Multimodal Large Language Models. *arXiv e-prints*, art. arXiv:2306.13549, June 2023.