LLM–WRAPPER: BLACK-BOX SEMANTIC-AWARE ADAPTATION OF VISION-LANGUAGE MODELS FOR REFERRING EXPRESSION COMPREHENSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision Language Models (VLMs) have demonstrated remarkable capabilities in various open-vocabulary tasks, yet their zero-shot performance lags behind task-specific fine-tuned models, particularly in complex tasks like Referring Expression Comprehension (REC). Fine-tuning usually requires 'white-box' access to the model's architecture and weights, which is not always feasible due to proprietary or privacy concerns. In this work, we propose `LLM-wrapper`, a method for 'black-box' adaptation of VLMs for the REC task using Large Language Models (LLMs). `LLM-wrapper` capitalizes on the reasoning abilities of LLMs, improved with a light fine-tuning, to select the most relevant bounding box matching the referring expression, from candidates generated by a zero-shot black-box VLM. Our approach offers several advantages: it enables the adaptation of closed-source models without needing access to their internal workings, it is versatile as it works with any VLM, it transfers to new VLMs and datasets, and it allows for the adaptation of an ensemble of VLMs. We evaluate `LLM-wrapper` on multiple datasets using different VLMs and LLMs, demonstrating significant performance improvements and highlighting the versatility of our method. While `LLM-wrapper` is not meant to directly compete with standard white-box fine-tuning, it offers a practical and effective alternative for black-box VLM adaptation. The code will be open-sourced.

1 INTRODUCTION

Vision Language Models (VLMs), a class of foundation models (Bommasani et al., 2021), trained on large-scale and diverse tasks and datasets, have shown remarkable abilities to solve various open-vocabulary tasks, as image captioning (Xiao et al., 2023; Li et al., 2023), visual question answering (Alayrac et al., 2022; Liu et al., 2023; Li et al., 2023), text-image retrieval (Radford et al., 2021; Zhai et al., 2023; Li et al., 2023), object detection (Liu et al., 2024c; Xiao et al., 2023; Cheng et al., 2024), or semantic segmentation (Ding et al., 2023; Xiao et al., 2023). Recent VLMs show promising zero-shot generalization abilities to new tasks and data domains (Wei et al., 2022; Alayrac et al., 2022). However, there is still a significant performance gap between zero-shot VLMs and those that have been specifically trained or adapted for a particular task and data domain. This work focuses on the challenging open-vocabulary detection task of Referring Expression Comprehension (REC) (Mao et al., 2016), which involves localizing an object in an image based on a complex textual query, requiring both spatial and semantic reasoning. While zero-shot VLMs can typically detect most objects mentioned in the query with reasonably accurate bounding boxes and labels, they struggle to identify *only* the described object. Moreover, VLMs often have difficulty understanding complex descriptions that involve relations between objects, attributes, or negations (Xie et al., 2023; Yao et al., 2024).

To improve performance, VLMs are typically fine-tuned on the specific REC task and corresponding datasets. This fine-tuning is usually done in a *'white-box'* manner, with full access to the model's architecture and weights for back-propagation. However, this process requires expertise to design fine-tuning objectives and optimize hyper-parameters, specific to each VLM and downstream task. Moreover, white-box fine-tuning is not always feasible. Some models are closed-source, either because they are proprietary and released behind APIs (Ren et al., 2024), or because they are trained on private data, making their weights and gradients inaccessible. While companies may provide APIs for adapting proprietary models, e.g., (OpenAI, 2024), these solutions are limited to predefined scopes and require sharing data with external private companies, raising legal and privacy concerns.

Table 1: **Comparison between white-box fine-tuning and `LLM-wrapper` on the REC task.**

	Classic white-box fine-tuning	**`LLM-wrapper`**
Model access	Needs complete access (loss, architecture, weights, gradients)	Only needs a forward call of a frozen black-box VLM
Fine-tuning knowledge	Specific for each VLM	Agnostic to the choice of VLM and LLM, and parameter-efficient
Specificity of adaptation	None	Leverages LLM semantic reasoning
Fine-tuning generalization	Limited: no ensembling, no transfer to new VLMs or datasets	Flexible: supports ensembling and transfers across VLMs and datasets
Expected performances	Best results	Good results

To address these challenges, we explore *'black-box'* adaptation of VLMs, where only forward calls to the model are possible. We propose `LLM-wrapper`, a new method for the black-box adaptation of VLMs for the REC task, by using an LLM to reason on the VLM's outputs. This approach builds on the recent development of Large Language Models (LLMs) (Touvron et al., 2023; Jiang et al., 2024; Mesnard et al., 2024), which have shown interesting reasoning capabilities. The underlying idea is that a zero-shot black-box VLM can generate high-quality labeled bounding boxes, and that `LLM-wrapper` can then leverage the semantic and spatial reasoning abilities of the LLM (Lian et al., 2024a;b) to 'reason' on such outputs, further enhanced with a light fine-tuning. As illustrated in Figure 1, our method involves translating VLM's outputs into a natural language prompt and feeding it to the LLM. The LLM is then tasked with identifying the box that best matches the referring expression from the given candidates.

`LLM-wrapper` offers several advantages, summarized in Table 1. First, since the adaptation is done in a black-box manner, it removes the need for back-propagation through the VLM and allows for the adaptation of closed-source models, without requiring access to the model's architecture, weights, or gradients. This makes black-box fine-tuning versatile and easy to use, as it can be applied to any model without requiring specific model knowledge or assumptions about the model's architecture. Additionally, as the adaptation is delegated to the LLM via text, `LLM-wrapper` retains the pre-trained knowledge of the original VLM. We find that `LLM-wrapper` transfers well to other VLMs, generalizes to model updates, and can adapt effectively to new datasets without additional fine-tuning. Finally, `LLM-wrapper` enables the adaptation of an ensemble of VLMs, leveraging the flexibility of text-based adaptation to handle varying numbers of bounding boxes, thereby combining strengths from multiple models.

We experiment with `LLM-wrapper` on three REC datasets – RefCOCO (Yu et al., 2016), RefCOCO+ (Yu et al., 2016), RefCOCOg (Mao et al., 2016); additionaly we evaluate `LLM-wrapper` on HC-RefLoCo Wei et al. (2024) – using two notably different VLMs – Grounding-DINO (Liu et al., 2024c) and Florence-2 (Xiao et al., 2023). We also experiment with two LLMs for the adaptation: the recent Mixtral 8x7B (Jiang et al., 2024) and Llama 3 8B (Dubey et al., 2024). While `LLM-wrapper` is not meant to outperform standard white-box fine-tuning, we show that `LLM-wrapper` significantly enhances the VLM's performances, across all combinations of VLMs, LLMs and datasets, thus demonstrating the versatility of our method. Notably, on RefCOCOg, our most challenging benchmark with respect to semantic understanding, `LLM-wrapper` improves the results of zero-shot VLMs with gains ranging from +9.5 up to +18.0 P@1.

2 Related Work

Referring Expression Comprehension (REC) and Vision Language Models (VLMs). Referring Expression Comprehension (REC) (Kazemzadeh et al., 2014; Mao et al., 2016; Qiao et al., 2021) is the task of identifying objects in an image based on referring expressions. Typically, REC involves selecting the best region from a set of region proposals extracted from the image, guided by a referring expression. This task is challenging because the referring expression can range from a short phrase (Kazemzadeh et al., 2014; Yu et al., 2016) to a long sentence that may require multi-step reasoning (Mao et al., 2016). REC is distinct from similar tasks such as visual grounding (Rohrbach et al., 2016), where multiple object regions described by multiple noun phrases must be localized in an image. It also differs from object detection (Girshick, 2015), which uses predefined categories instead of natural language expressions. The recent surge of interest in VLMs, starting from CLIP (Radford et al., 2021) to the recent Florence-2 (Xiao et al., 2023) has significantly improved performance on tasks requiring both vision and language. For REC, several approaches have used

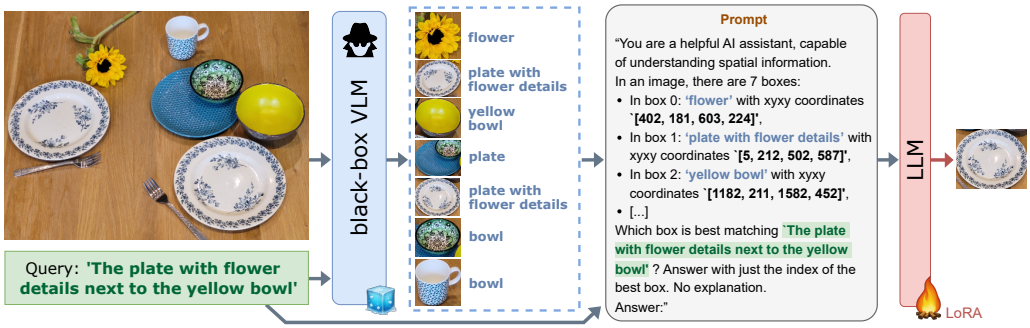


Figure 1: **Illustration of `LLM-wrapper`.** Our method adapts a black-box VLM for the REC task. The output of the VLM is translated into a natural language text that is used to prompt an LLM. The LLM is then tasked with identifying the box that best matches the referring expression among the given candidates. The LLM must learn to identify the subject of the query and to disambiguate the correct object from other distractor objects (e.g., several plates with 'flower details').

VLMs. For instance, Grounding-DINO (Liu et al., 2024c) relies on multiple stages of modality fusion to align visual and textual features. The more recent Florence-2 (Xiao et al., 2023) is a sequence-to-sequence model trained on a huge collection of data. However, despite being trained on extensive amounts of image-text data (e.g., 126 million images with 500 million to 3.6 billion annotations for Florence-2), these models' zero-shot performances on REC are sub-optimal compared to their performances when including REC data in their training set or when fine-tuned (see Section 4.2). Our approach, `LLM-wrapper`, addresses this clear limitation in a black-box manner, without the need to re-train the VLM with task-specific data.

VLM adaptation. While retraining the entire VLM on a new task or data domain is computationally expensive (Liu et al., 2023; Wang et al., 2023; Chen et al., 2023; Xiao et al., 2023), fine-tuning offers a more efficient alternative. However, even fine-tuning can be costly. To address this, parameter-efficient fine-tuning, e.g., LoRa (Hu et al., 2022), DoRa (Liu et al., 2024a), or VeRa (Kopiczko et al., 2024), and soft prompt (Li & Liang, 2021) learning have been proposed. These methods avoid updating the model's pre-trained weights but require access to the model's architecture and gradients.

A few recent methods propose strategies to adapt VLMs when back-propagation through the VLM is not feasible (Ouali et al., 2023; Yu et al., 2023; Liu et al., 2024b; Oh et al., 2023). Both Ouali et al. (2023) and Yu et al. (2023) focus on CLIP-based methods and adapt the VLM by learning either a feature projection or soft prompts. They however require access to the inner representations, which is generally not possible with APIs. Liu et al. (2024b) optimize the input prompt template, but also focuses on CLIP-based models. Instead, we target VLMs that perform open-vocabulary detection, e.g., Grounding-DINO and Florence-2. Finally, (Oh et al., 2023) adapt a VLM in a black-box manner by modifying the image input. However, this method is designed to address visual domain gaps and is not directly applicable to adapt a VLM to a new *task*. Moreover, it requires multiple API calls for each training image. In this work, we propose a strategy to adapt VLMs, and more specifically open-vocabulary detectors, to a new task, in a complete black-box fashion. Our method does not require access to the model's intermediate representations or gradients, runs a single API call for each (text-image) input pair, and enables adaptation to a new task (REC in our case) that the VLM was not originally trained for.

3 `LLM-wrapper`: BLACK-BOX SEMANTIC-AWARE ADAPTATION OF VLMS.

In this section, we present `LLM-wrapper`, a novel LLM-based approach to adapt an open-ended VLM for the REC task. We present the general idea in Section 3.1, the construction of the prompt for the LLM in Section 3.2, the fine-tuning details in Section 3.3.

3.1 GENERAL IDEA

Our method *wraps* the open-vocabulary detections of a frozen black-box VLM with an LLM that reasons over these outputs. An overview is presented in Figure 1. Given a complex textual query, `LLM-wrapper` leverages the fact that detection-oriented VLMs can typically localize well most nouns of the query, even

if they struggle with the reasoning step required to precisely select the object of interest among several distractors. Therefore, `LLM-wrapper` delegates the reasoning task to an LLM, which has interesting abilities to handle difficult text queries, including attributes, negation, and relational or spatial descriptions of objects. As the gradients of the black-box VLM are not accessible, we propose to adapt its *outputs* with the LLM. It gives us access to the LLM gradients and we can thus specialize the LLM for the task, with a simple and light fine-tuning, performed on the outputs of the VLM to learn to select the right box among them. Overall, `LLM-wrapper` only requires black-box access to the VLM, whereas standard fine-tuning strategies need white-box access to perform back-propagation. This makes our approach more flexible and applicable in scenarios where the internal workings of the VLM are not accessible.

3.2 Prompt construction

The key idea of our method is to convert the VLM's outputs into natural language. To achieve this, we list all predicted outputs in the LLM prompt, including their box coordinates, labels, and, when applicable, prediction scores (displayed below in light gray). This allows the 'blind' LLM, which only reads text, to understand the scene and reason about the image. The prompt then reminds the query and asks the LLM to select the best matching box. For instance, given the query (in green in Figure 1), and the associated outputs (e.g., 'flower', 'plate with flower details', 'yellow bowl', etc.), we ask the LLM for the best matching box index, as follows:

> You are a helpful AI assistant, capable of understanding spatial information.
> In an image, there are 7 boxes:
> * In box 0: 'flower' with xyxy coordinates '[402, 181, 603, 224]' with score 0.92,
> * In box 1: 'plate with flower details' with xyxy coordinates '[5, 212, 502, 587]' with score 0.88,
> * In box 2: 'yellow bowl' with xyxy coordinates '[1182, 211, 1582, 452]' with score 0.86,
> * [...]
> Which box is best matching 'The plate with flower details next to the yellow bowl' ?
> Answer with just the index of the best box. No explanation.
> Answer:

3.3 Fine-tuning the LLM

While a zero-shot LLM can already reason on the new task to some extent, we find that fine-tuning the LLM significantly improves performances on the task. Therefore, we fine-tune the LLM for prompt completion with a cross-entropy loss, using the prompt described above. Specifically, the expected answer for the LLM is the index of the best box proposal, corresponding to the closest match to the known ground truth box. To build the training dataset, we use the REC training data, which consists of `(image, query)` pairs and ground truth boxes. The detection outputs (boxes, labels, scores), used to create the training prompts, are inferred using the VLM being adapted. We only keep the samples where at least one of the VLM box proposals has an Intersection over Union (IoU) with the ground truth box higher than 0.5, ensuring no noisy samples. To make `LLM-wrapper` robust to any shortcut learning based on the box order, we randomly permute the order of the box proposals in the prompt during training.

We leverage the extensive literature on LLM fine-tuning to specialize the LLM of `LLM-wrapper`. Specifically, we use LoRA (Hu et al., 2022), which introduces additive updates to the model's activations, parameterized by low-rank modules. Doing so reduces the number of new parameters to learn while preserving the LLM's general knowledge. We also use flash attention (Dao et al., 2022) and 4-bit quantization (Dettmers et al., 2024), making `LLM-wrapper` trainable on a single 40GB-A100 GPU in less than 7 hours. This approach makes the training efficient in terms of compute and very simple to implement in practice. Overall, we find that `LLM-wrapper` is not very sensitive to the choice of the few hyper-parameters introduced (see Section 4.4).

4 Experiments

In this section we present the experimental validation of `LLM-wrapper`. We first state our protocol in Section 4.1. In Section 4.2, we show how `LLM-wrapper` improves VLMs. We then conduct further

analysis of LLM-wrapper's benefits in Section 4.3. We finally conduct ablation studies showcasing LLM-wrapper's robustness in Section 4.4.

4.1 Experimental setup and technical details

4.1.1 Referring Expression Comprehension (REC) task

Datasets. We evaluate LLM-wrapper on the REC task. In REC, given an input pair (image, query), a model is expected to predict a single bounding box around the object described in the query, as illustrated in Figure 1. We use three standard datasets for REC, RefCOCO (Yu et al., 2016), RefCOCO+ (Yu et al., 2016), and RefCOCOg (Mao et al., 2016) which is more challenging as it contains longer and more complex descriptions (8.4 words per query in average). Additionnally, we evaluate LLM-wrapper on the HC-RefLoCo (Wei et al., 2024) benchmark for zero-shot dataset transfer experiments. All of these datasets contain multiple distractor objects in each image for the given text query. Dataset's statistics are reported in Table 2.

Metric. We measure performance with the standard precision@1 (P@1) metric, described in (Qiao et al., 2021). A true positive is defined when the predicted box has an Intersection-Over-Union (IoU) greater than 0.5 with the ground truth box. The metric is averaged over the evaluation set.

Table 2: REC datasets statistics.

	Split	Size			# words
Dataset	used	train	val	test	/ query
RefCOCO	unc	120,624	10,834	10,752	3.6
RefCOCO+	unc	120,191	10,758	10,615	3.5
RefCOCOg	umd	80,512	4,896	9,602	8.4
HC-RefLoCo	—	—	13,360	31,378	84.4

4.1.2 The VLMs

We evaluate the impact of LLM-wrapper on two different VLMs. In all cases, we use the official model checkpoints and unless specified otherwise, we use the model versions that are not fine-tuned for the REC task, meaning that the models have not been exposed to any of the RefCOCO/+/g datasets. Using the model setup described below, we feed from 2 to 20 boxes to LLM-wrapper per prompt.

Grounding-DINO (GD) (Liu et al., 2024c) aligns visual queries with text through stages of modality fusion and contrastive learning. Initially designed for open-vocabulary grounding, the model produces 900 bounding boxes, achieving a high recall but an under-performing precision on the REC task. For REC, we use GD by selecting the bounding box with the highest score relative to any token in the query. However, this method sometimes selects boxes based on query parts unrelated to the main object, such as other nouns in the sentence. To address this, we introduce Grounding-DINO-REC (GDrec), a more targeted approach for REC. GDrec identifies the query's subject, defined as the first noun group detected by SpaCy's dependency parser (Honnibal et al., 2020), and selects the bounding box that scores best against this subject. GDrec significantly outperforms GD on all datasets, and particularly on RefCOCOg, where the noun identification is more challenging, with e.g., +7 P@1 for zero-shot models as shown in Table 3. To ensure a rather short prompt, we limit the number of box proposals by setting the box confidence score threshold to 0.15 for GDrec and to 0.2 for GD. GD / GDrec, not fine-tuned, use a SwinT (T) backbone while the fine-tuned versions are based on SwinB (B), as they are the only publicly available models. We also run experiments on a subset of RefCOCOg using Grounding-DINO 1.5 (Ren et al., 2024) (GD-1.5), a recent detector behind API (online at: Grounding-DINO-1.5-API), which provides 300 free API calls. GD-1.5 extends GD with a larger backbone, ViT-L (Fang et al., 2024), and training dataset.

Florence-2 (Flo2) (Xiao et al., 2023) is a sequence-to-sequence multi-task model. We use the Florence-2 Large version from the Hugging Face Hub. It is composed of a DaViT vision encoder (Ding et al., 2022) and a multi-modal encoder-decoder. Flo2 can be prompted for several tasks, and we keep and concatenate the boxes from the 'open vocabulary detection' and the 'phrase grounding' task modes.

4.1.3 The LLMs and their fine-tuning

Our main experiments are conducted using two different LLMs: Mixtral 8x7B Instruct (Jiang et al., 2024) (v0.1) and Llama 3 8B Instruct (AI@Meta, 2024) with Hugging Face's implementation. We use Hugging Face's supervised fine-tuning pipeline (SFT) (HuggingFace, 2024), that allows to implement the training choices discussed in Section 3. Specifically, for Llama 3 8B, we train 352M parameters, which is 4.20% of the original model size. For Mixtral 8x7B, we train 114M parameters, which is 0.24% of the original size. To further study the impact of the LLM scale, we also experiment with two additional families of

Table 3: **Main results of `LLM-wrapper` on the REC task**, in P@1↑, on RefCOCO/+/g datasets. `(T)` and `(B)` stand for the `SwinT` and `SwinB` backbones respectively.

Adaptation	Model access	VLM	LLM	RefCOCOg val-umd	RefCOCOg test-umd	RefCOCO val-unc	RefCOCO test-unc	RefCOCO+ val-unc	RefCOCO+ test-unc
∅ (zero-shot)		GD (T)		60.09	59.32	50.69	50.94	51.65	51.79
Fine-tuning	*White-box*	*GD (B)*		*78.51*	*77.99*	*83.86*	*84.12*	*73.46*	*73.46*
LLM-wrapper	Black-box	GD (T)	Mixtral	77.57 ↑**17.5**	77.05 ↑**17.7**	74.61↑**23.9**	73.46↑**22.5**	60.32↑**8.7**	60.08↑**8.3**
LLM-wrapper	Black-box	GD (T)	Llama3	78.12 ↑**18.0**	77.36 ↑**18.0**	74.78↑**24.1**	73.98↑**23.0**	64.18↑**12.5**	63.82↑**12.0**
∅ (zero-shot)		GDrec (T)		67.61	68.37	51.82	52.12	53.28	53.16
Fine-tuning	*White-box*	*GDrec (B)*		*80.19*	*79.85*	*83.84*	*84.21*	*73.68*	*73.58*
LLM-wrapper	Black-box	GDrec (T)	Mixtral	78.47 ↑**10.9**	77.92 ↑**9.6**	72.61↑**20.8**	71.48↑**19.4**	63.79↑**10.5**	63.69↑**10.5**
LLM-wrapper	Black-box	GDrec (T)	Llama3	78.25 ↑**10.6**	78.01 ↑**9.6**	73.97↑**22.2**	73.07↑**21.0**	64.13↑**10.9**	64.08↑**10.9**
∅ (zero-shot)		Flo2		68.28	66.90	56.32	57.01	53.71	54.43
Fine-tuning	*White-box*	*Flo2*		*90.32*	*91.06*	*93.08*	*93.42*	*88.20*	*88.49*
LLM-wrapper	Black-box	Flo2	Mixtral	77.74 ↑**9.5**	76.98 ↑**10.1**	67.29↑**11.0**	66.62↑**9.6**	57.38↑**3.7**	57.48↑**3.1**
LLM-wrapper	Black-box	Flo2	Llama3	77.94 ↑**9.7**	77.15 ↑**10.3**	70.19↑**13.9**	70.00↑**13.0**	61.95↑**8.2**	61.87↑**7.4**

models, Gemma 2 (Gemma, 2024) and GPT-Neo (Gao et al., 2021). The latter is a class of LLMs, based on a replication of the GPT-3 architecture, trained on the large curated Pile dataset (Gao et al., 2021), and which was not 'instructed' (Ouyang et al., 2022). We train `LLM-wrapper` with Adam (Kingma, 2014), with a batch-size of four, until convergence. We discuss in Section 4.4 the sample efficiency of `LLM-wrapper`. Unless stated otherwise, we use a learning rate of 10^{-5} and a rank of $r = 128$ for LoRA. These hyper-parameters work consistently across three datasets, two LLMs and two VLMs, demonstrating the robustness of `LLM-wrapper`. We study the performance robustness of `LLM-wrapper` for different hyper-parameters in Section 4.4. We provide further ablations in Appendix B and some statistics on invalid generated outputs in Appendix C.

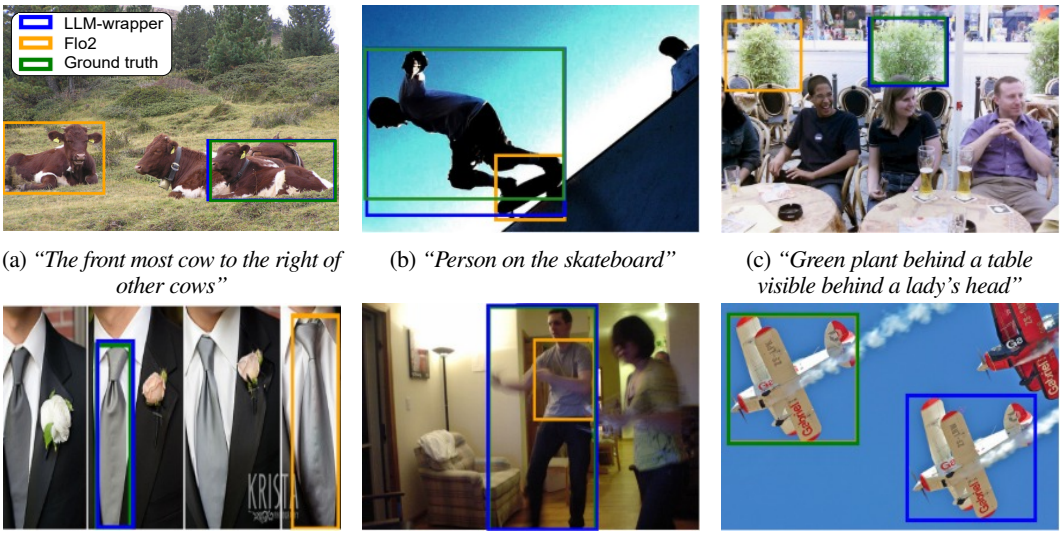
4.2 `LLM-wrapper` CAN ADAPT VLMs TO THE REC TASK

In Table 3, we report the performances of VLMs on the REC task in three settings: zero-shot off-the-shelf VLM, after classic white-box fine-tuning, and after black-box adaptation with `LLM-wrapper`. The results are obtained by adapting the VLM using exclusively the training data specific to each benchmark.

Our first observation confirms that while VLMs demonstrate remarkable zero-shot performance on new tasks and datasets, their performance is still significantly lower than models specifically adapted to the given task (white-box fine-tuning). As shown in Table 3, for both `GD` (and its variant `GDrec`) and `Flo2`, there is a notable gap in REC performance, ranging from -11.5 to -37 P@1, between the zero-shot and fine-tuned versions.

Interestingly, we observe that for all combinations of VLMs and LLMs, `LLM-wrapper` can adapt VLMs to the new REC task, despite the black-box setting. For instance, on RefCOCOg, `LLM-wrapper` brings improvements over the zero-shot models ranging from +9.5 to +18 P@1. Notably, while our proposed variant `GDrec` outperforms `GD` by large margins in a zero-shot setting (e.g., +9.05 P@1-test) due to better subject identification, both models perform similarly when adapted with `LLM-wrapper`. This shows that `LLM-wrapper` particularly improves VLMs that lack abilities useful for the task, such as subject identification in the case of REC for `GD`. On RefCOCO, `LLM-wrapper` improves again by significant margins zero-shot VLMs, e.g., +23.0 P@1 (test) for `GD` with Llama 3 8B. Given that textual queries in RefCOCO are very short (3.6 words on average), subject identification is relatively easy for these queries. Therefore, these gains indicate that `LLM-wrapper` greatly helps to disambiguate the object of interest from the distractors. On RefCOCO+, the improvement brought by `LLM-wrapper` is lower but still significant (between +3.1 and +12.5 P@1). This is expected as RefCOCO+ is designed to exclude location words, and most referring expressions are thus purely appearance-based descriptions (Qiao et al., 2021) where `LLM-wrapper` brings less value for adaptation.

We present some qualitative results in Figure 2, where we show predictions from `Flo2` before and after being adapted with `LLM-wrapper`, with Llama 3 8B. We observe that the adaptation with `LLM-wrapper` enables better spatial understanding (Fig. 2a, Fig. 2d), subject identification (Fig. 2b, Fig. 2e) and relational reasoning (Fig. 2b, Fig. 2c, Fig. 2e). We display additional qualitative results in Appendix D.



(a) *"The front most cow to the right of other cows"* (b) *"Person on the skateboard"* (c) *"Green plant behind a table visible behind a lady's head"*

(d) *"The tie at the second from the left"* (e) *"A boy wearing a grey shirt"* (f) *"White plane in the front"*

Figure 2: **Qualitative results of `Flo2` outputs on RefCOCOg, before and after adaptation by `LLM-wrapper`**, provided with the query in caption. Adapting `Flo2` (in **orange**) with `LLM-wrapper` (using Llama 3 8B, and represented in **blue**) leads to improved reasoning and better bounding box selection.

Finally, `LLM-wrapper` is not designed to outperform classic white-box fine-tuning but demonstrates competitive performance in some settings. For example, with `GD` and `GDrec` on RefCOCOg, `LLM-wrapper` achieves comparable results despite using a smaller vision backbone. Additionally, `LLM-wrapper` can complement white-box fine-tuning by adapting VLMs already optimized for the REC task. As shown in Appendix A, applying `LLM-wrapper` on top of fine-tuned models does not degrade performances and, in some cases, provides a slight boost. This highlights `LLM-wrapper`'s compatibility with state-of-the-art methods and its ability to enhance existing adaptations.

4.3 Benefits of wrapping VLMs' outputs with `LLM-wrapper`

Ensembling VLMs. One advantage of `LLM-wrapper` is its free-text input format, which allows adapting to various inputs. Indeed, we show that `LLM-wrapper` can learn to *ensemble* outputs from different VLMs, as shown in Table 4. Specifically, we concatenate the predictions of the two VLMs in the prompt described in Section 3. The results show that when ensembling `GDrec` and `Flo2`, scores are boosted by +3.0 P@1 (val-umd) and +2.1 P@1 (test-umd) when compared to those obtained with the best-performing VLM adapted with `LLM-wrapper`, namely `GDrec`. This demonstrates that `LLM-wrapper` is capable of reasoning on multiple sources and leveraging the strengths of different models.

Indeed, we observe that `Flo2` has a high precision, while `GD` and `GDrec` have lower precision but high recall. We show qualitatively in Figure 3 an example where `LLM-wrapper` leverages the complementarity of the two models. The figure displays predictions from `Flo2` (e), `GDrec` (f), and `LLM-wrapper` ensembling predictions from both `Flo2` and `GDrec` (d). We observe that while `Flo2` fails to detect the

Table 4: **Results of VLMs ensembling using `LLM-wrapper`** with Llama 3 8B, in P@1↑ on RefCOCOg using 'umd' splits. (Comparable findings for Mixtral).

Adaptation	VLM	P@1 - val ↑	P@1 - test ↑
∅ (zero-shot VLM)	`GDrec`	67.61	68.37
∅ (zero-shot VLM)	`Flo2`	68.28	66.90
`LLM-wrapper`	`Flo2`	77.94	77.15
`LLM-wrapper`	`GDrec`	78.25	78.01
`LLM-wrapper`	`Flo2` + `GDrec`	81.25	80.13

Query: *"A bottle of wine between the vegetables"*

(a) Ground truth (b) All candidates of `Flo2` (c) All candidates of `GDrec`

(d) Final pred. of `LLM-wrapper` (e) Final pred. of `Flo2` (f) Final pred. of `GDrec`

Figure 3: **Visualizations of the candidates and predictions** for the query "A bottle of wine between the vegetables". We visualize the ground truth (a) and the set of box candidates generated by `Flo2` (b) and `GDrec` (c). In the second row, we visualize the final predictions of `Flo2` (e) and `GDrec` (f) and in (d) the prediction of `LLM-wrapper` applied on the ensemble of both VLMs' outputs, using Llama 3 8B. We observe that `LLM-wrapper` discards the distractor bottle and selects the correct object.

target object as a possible candidate (b), the additional proposals from `GDrec` (c) enable `LLM-wrapper` to find the correct object (d) that each independent models missed.

Transferring a trained `LLM-wrapper` to a new VLM. Another advantage of using text as the input format is that `LLM-wrapper` does not rely on model-specific activation values, making it transferable from one VLM to another. For instance, `LLM-wrapper` can be fine-tuned on `Flo2` and transferred to `GD` or `GDrec`. This is illustrated in Table 5, where, for instance, when fine-tuned on `GDrec`'s or `Flo2`'s outputs, transferring it at inference time to the other model's outputs gives an increase from +5.6 to +6.3 P@1 over zero-shot VLMs. This shows that during fine-tuning, `LLM-wrapper` learns spatial and semantic notions that generalize to other models. This capacity of `LLM-wrapper` is particularly useful for private models, such as `GD-1.5` (Ren et al., 2024), where creating the training set can be expensive – for instance, getting predictions for RefCOCOg train would cost ≈ $1,600 ($20 per 1,000 API calls). To illustrate this use case, we use a RefCOCOg val subset corresponding to 300 free API calls to `GD-1.5`. When `LLM-wrapper` is fine-tuned on `GDrec`'s outputs and applied on `GD-1.5`'s outputs for inference, results are boosted by a significant +29.0 P@1. To ensure that the val-subset reflects the full val set's difficulty, we evaluate `GDrec` and `Flo2` on both sets. The results are consistent, confirming that the val-subset is a good proxy for the full set and that the performance gains from `LLM-wrapper`

Table 5: **Results of `LLM-wrapper` when using different VLMs' outputs during fine-tuning and inference,** in P@1↑ on RefCOCOg using 'umd' splits. Results obtained with Llama 3 8B. Comparable findings for Mixtral. †Scores obtained on a subset of 300 samples from RefCOCOg val-umd.

Adaptation	VLM (fine-tuning)	VLM (inference)	P@1 - val ↑ (subset 300) ↑	P@1 - val ↑ (full)	P@1 - test ↑ (full)
∅ (zero-shot VLM)	∅	`GDrec`	66.00†	67.61	68.37
`LLM-wrapper`	`Flo2`	`GDrec`	74.00† ↑**8.0**	73.90 ↑**6.3**	73.45 ↑**5.1**
∅ (zero-shot VLM)	∅	`Flo2`	71.00†	68.28	66.90
`LLM-wrapper`	`GDrec`	`Flo2`	75.33† ↑**4.3**	73.86 ↑**5.6**	73.03 ↑**6.1**
∅ (zero-shot VLM)	∅	`GD-1.5`	47.67†	—	—
`LLM-wrapper`	`GDrec`	`GD-1.5`	76.67† ↑**29.0**	—	—

Table 6: **LLM-wrapper performance on zero-shot dataset transfer**. Results obtained with Llama 3 8B. ('FT' stands for 'fine-tuning'. † Flo2 is fine-tuned using 'a collection of public supervised data on a wide range of downstream tasks', including but not limited to RefCOCO/+/g (Xiao et al., 2023).)

Adaptation	VLM	Finetuning Data	Inference Data	P@1 - val ↑	P@1 - test ↑
∅ (zero-shot VLM)	Flo2	—	HC-RefLoCo	48.64	48.87
White-box FT	*Flo2*	*RefCOCO/+/g*†	*HC-RefLoCo*	*56.74* ↑**8.10**	*55.62* ↑**6.75**
LLM-wrapper	Flo2	RefCOCOg	HC-RefLoCo	67.40 ↑**18.76**	67.26 ↑**18.39**
∅ (zero-shot VLM)	Flo2	—	RefCOCO	56.32	57.01
LLM-wrapper	Flo2	RefCOCO	RefCOCO	70.19 ↑**13.87**	70.00 ↑**12.99**
LLM-wrapper	Flo2	RefCOCOg	RefCOCO	69.00 ↑**12.68**	68.88 ↑**11.87**
∅ (zero-shot VLM)	Flo2	—	RefCOCO+	53.71	54.43
LLM-wrapper	Flo2	RefCOCO+	RefCOCO+	61.95 ↑**8.24**	61.87 ↑**7.44**
LLM-wrapper	Flo2	RefCOCOg	RefCOCO+	61.00 ↑**7.29**	61.07 ↑**6.64**

on GD-1.5 are significant. This showcases how LLM-wrapper can be used on continuously updated versions of models without the need to re-train it with each model update.

Transferring a trained LLM-wrapper to new datasets. LLM-wrapper demonstrates strong generalization across datasets, reducing the need for training data and resources on new target domains. To demonstrate this property, we fine-tune Flo2 with LLM-wrapper on RefCOCOg and evaluate it on RefCOCO and RefCOCO+. Results in Table 6 show substantial gains over zero-shot Flo2, with improvements up to +12.7 P@1 on RefCOCO and +7.3 P@1 on RefCOCO+. Although slightly below results from direct fine-tuning on target datasets, these improvements demonstrate LLM-wrapper's ability to transfer knowledge effectively. To test generalization to more complex scenarios, we evaluate Flo2 adapted with LLM-wrapper on RefCOCOg to HC-RefLoCo, a benchmark with longer and more complex referring expressions and no training split. LLM-wrapper achieves +18.8 and +18.4 P@1 improvements on the validation and test sets, respectively, compared to zero-shot Flo2. Interestingly, when white-box fine-tuned Flo2 is transferred to HC-RefLoCo, its performance boost over zero-shot Flo2 is less than half of that achieved by LLM-wrapper. These results highlight LLM-wrapper's ability to handle in a zero-shot setting complex referring expressions, with an average length of 84 words, despite being fine-tuned on ten times shorter expressions. More information on the impact of text queries' complexity is given in Appendix B.2. Qualitative examples on HC-RefLoCo are shown in Appendix D.1.

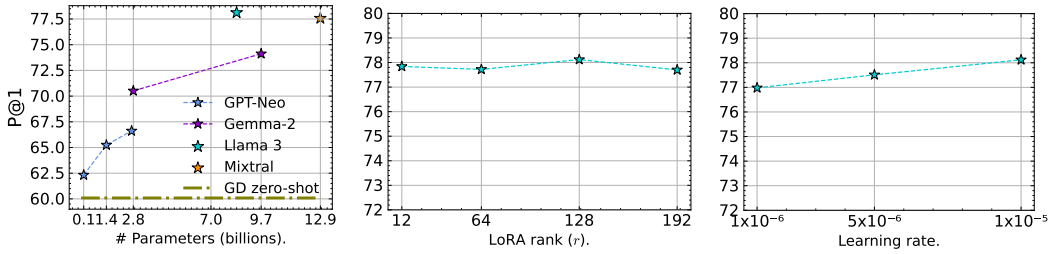
4.4 Ablation Studies

Unless specified otherwise, we run ablations on RefCOCOg val-umd, using GD's outputs and Llama 3 8B.

Impact of LLM scale. In Figure 4a, we report the P@1 REC performance (y-axis) for various LLM sizes (x-axis). We observe that LLM-wrapper is effective across all model families, and that the gain in performance positively correlates with the LLM size. Indeed, though the sub-billion model, GPT-Neo 139M, slightly boosts the P@1 performance of GD zero-shot (+2.2 P@1), this increase is only incremental when compared to Llama 3 8B's gains (+18 P@1). Further analysis shows a Pearson correlation of 0.88 between the LLM's original performance on reasoning benchmarks, measured by the HellaSwag score (Zellers et al., 2019), and REC performance (details in Appendix B.1). Overall, these findings demonstrate that larger, more capable LLMs are substantially more effective, while smaller models offer only limited improvements.

Robustness to hyper-parameters. In Figure 4b, we show that LLM-wrapper is not overly sensitive to the value of the LoRA rank, which controls the number of fine-tuned parameters. In the explored range, $r \in \{12,64,128,192\}$, corresponding to fine-tuning$\{0.41\%, 2.15\%, 4.20\%, 6.18\%\}$ of total parameters, the performance varies by only ± 0.4 P@1. We provide a more in-depth analysis of the impact of the number of fine-tuned parameters on performance in Appendix B.1. Similarly, we observe in Figure 4c that LLM-wrapper is loosely sensitive to the learning rate choice within the range $\{10^{-6},10^{-5}\}$. With learning rate values higher that 10^{-5}, we observe unstable training behaviors.

Ablation on the number of training samples. Figure 5 displays the P@1 REC performance on RefCOCOg val with respect to the number of training samples, when fine-tuning Llama 3 8B on RefCOCOg train. It shows a sharp increase in P@1 for all VLMs (most impressive for GD) during the first 30k samples, which takes ∼2h of training in our setting. Thus, even with a restricted amount of samples, LLM-wrapper



(a) LLM size. (b) LoRA parametrization. (updated figure adding r=12 value) (c) Learning rate.

Figure 4: **Study of `LLM-wrapper`'s sensitivity to hyper-parameters.** Impact on `LLM-wrapper`'s performances (with `GD` on RefCOCOg) of the (a) LLM scale; (b) LoRA rank r; and (c) learning rate.

can boost performances. In the extreme case of no training samples for the LLM, i.e., a zero-shot LLM, we observe on Table 7 that in most settings the VLM and the VLM + zero-shot LLM have very similar results. This follows the observation that, while LLMs have an extensive general knowledge, they may lack off-the-shelf reasoning (Kazemi et al., 2023; Fu et al., 2024).

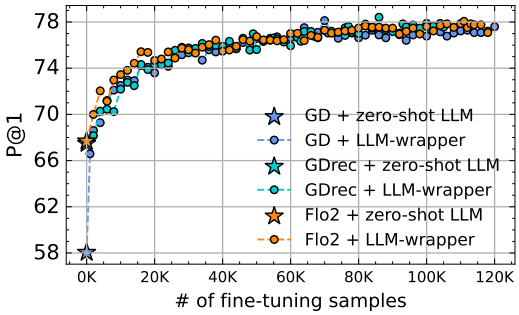


Figure 5: **Performance (P@1) of `LLM-wrapper` on RefCOCOg (val) with respect to the number of training samples.** We fine-tune Llama 3 8B on RefCOCOg (train).

VLM	P@1-val↑		P@1-test↑	
	VLM only	+ LLM zero-shot	VLM only	+ LLM zero-shot
GD	60.09	58.05	59.32	58.47
GDrec	67.61	67.48	68.37	68.17
GD-1.5	47.67†	59.00†	—	—
Flo2	68.28	67.69	66.90	66.44

Table 7: **Results of wrapping the VLM's outputs with a zero-shot LLM.** 'VLM only' are the scores of the VLM without any adaptation, and '+ LLM zero-shot' corresponds to `LLM-wrapper` without any fine-tuning of the LLM (Llama 3 8B here). † see Table 5.

5 Conclusion

This work introduces `LLM-wrapper`, a simple approach for the black-box adaptation of VLMs to the REC task, that leverages an LLM to reason on VLMs' outputs, translated into natural language. We demonstrate that `LLM-wrapper` significantly boosts the performance of VLMs, for several combinations of LLMs and VLMs, and, in some settings, even bridges the gap to classic fine-tuning. We also show how `LLM-wrapper` can ensemble predictions from different VLMs to leverage their respective strengths and how it can transfer across VLMs and to out-of-domain datasets. Thanks to efficient and well-studied methods for LLM fine-tuning, `LLM-wrapper` is simple to use in practice, requiring limited hyper-parameter tuning, and computationally efficient. Future works include relying on fewer examples to fine-tune `LLM-wrapper`, and applying `LLM-wrapper` to different tasks, such as text-video retrieval (Fang et al., 2021).

Limitations. `LLM-wrapper` comes with some limitations. First, an additional inference cost is introduced by integrating LLM reasoning on VLM outputs. Second, the effectiveness of `LLM-wrapper` relies on the quality of the underlying VLM. Diverse and accurate bounding boxes are crucial for success, and bounding box information alone may not be sufficient in some cases. For instance, as shown in Fig. 2f, distinguishing the 'White plane in the front' requires understanding the planes' direction, which is not provided via bounding boxes' coordinates and labels alone. Similar examples are shown in Appendix D.2. To address this issue, a promising direction is to enhance `LLM-wrapper` by using the LLM to identify missing visual information and suggest augmented queries for the VLM. This would help disambiguate cases where additional visual cues are needed.

REFERENCES

AI@Meta. Llama 3 model card. 2024. 5

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, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karén Simonyan. Flamingo: a visual language model for few-shot learning. In *Conference on Neural Information Processing Systems, NeurIPS*, 2022. 1

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint*, 2021. 1

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv preprint*, 2023. 3

Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, 2024. 1

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Conference on Neural Information Processing Systems, NeurIPS*, 2022. 4

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems, NeurIPS*, 2024. 4

Mingyu Ding, Bin Xiao, Noel Codella, Ping Luo, Jingdong Wang, and Lu Yuan. Davit: Dual attention vision transformers. In *European Conference on Computer Vision, ECCV*, 2022. 5

Zheng Ding, Jieke Wang, and Zhuowen Tu. Open-vocabulary universal image segmentation with maskclip. In *International Conference on Machine Learning, ICML*, 2023. 1

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, and Angela Fan et al. The llama 3 herd of models. *arXiv preprint*, 2024. 2

Han Fang, Pengfei Xiong, Luhui Xu, and Yu Chen. Clip2video: Mastering video-text retrieval via image clip. *arXiv preprint*, 2021. 10

Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *Image and Vision Computing*, 2024. 5

Deqing Fu, Ghazal Khalighinejad, Ollie Liu, Bhuwan Dhingra, Dani Yogatama, Robin Jia, and Willie Neiswanger. Isobench: Benchmarking multimodal foundation models on isomorphic representations. *arXiv preprint*, 2024. 10

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint*, 2021. 6

Team Gemma. Gemma 2: Improving open language models at a practical size. *arXiv preprint*, 2024. 6

Ross B. Girshick. Fast R-CNN. In *International Conference on Computer Vision, ICCV*, 2015. 2

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021. 15

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. spaCy: Industrial-strength Natural Language Processing in Python. 2020. 5, 16

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations, ICLR*, 2022. 3, 4

HuggingFace. Supervised fine-tuning trainer, 2024. 5

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts. *arXiv preprint*, 2024. 2, 5

Mehran Kazemi, Hamidreza Alvari, Ankit Anand, Jialin Wu, Xi Chen, and Radu Soricut. Geomverse: A systematic evaluation of large models for geometric reasoning. *arXiv preprint*, 2023. 10

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the conference on empirical methods in natural language processing, EMNLP*, 2014. 2

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint*, 2014. 6

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. Vera: Vector-based random matrix adaptation. In *International Conference on Learning Representations, ICLR*, 2024. 3

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning, ICML*, 2023. 1

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing, ACL/IJCNLP*, 2021. 3

Long Lian, Boyi Li, Adam Yala, and Trevor Darrell. Llm-grounded diffusion: Enhancing prompt understanding of text-to-image diffusion models with large language models. *Trans. Mach. Learn. Res.*, 2024a. 2

Long Lian, Baifeng Shi, Adam Yala, Trevor Darrell, and Boyi Li. Llm-grounded video diffusion models. In *The Twelfth International Conference on Learning Representations, ICLR*, 2024b. 2

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Conference on Neural Information Processing Systems, NeurIPS*, 2023. 1, 3

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *International Conference on Machine Learning, ICML*, 2024a. 3

Shihong Liu, Zhiqiu Lin, Samuel Yu, Ryan Lee, Tiffany Ling, Deepak Pathak, and Deva Ramanan. Language models as black-box optimizers for vision-language models. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2024b. 3

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision, ECCV*, 2024c. 1, 2, 3, 5

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2016. 1, 2, 5

Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya,

Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Cristian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, and et al. Gemma: Open models based on gemini research and technology. *arXiv preprint*, 2024. 2

Changdae Oh, Hyeji Hwang, Hee-young Lee, YongTaek Lim, Geunyoung Jung, Jiyoung Jung, Hosik Choi, and Kyungwoo Song. Blackvip: Black-box visual prompting for robust transfer learning. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2023. 3

OpenAI. Fine-tuning api, 2024. 1

Yassine Ouali, Adrian Bulat, Brais Matinez, and Georgios Tzimiropoulos. Black box few-shot adaptation for vision-language models. In *International Conference on Computer Vision, ICCV*, 2023. 3

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems, NeurIPS*, 2022. 6

Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint*, 2023. 15

Yanyuan Qiao, Chaorui Deng, and Qi Wu. Referring expression comprehension: A survey of methods and datasets. *IEEE Trans. Multim.*, 2021. 2, 5, 6

Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning, ICML*, 2021. 1, 2

Tianhe Ren, Qing Jiang, Shilong Liu, Zhaoyang Zeng, Wenlong Liu, Han Gao, Hongjie Huang, Zhengyu Ma, Xiaoke Jiang, Yihao Chen, Yuda Xiong, Hao Zhang, Feng Li, Peijun Tang, Kent Yu, and Lei Zhang. Grounding DINO 1.5: Advance the "edge" of open-set object detection. *arXiv preprint*, 2024. 1, 5, 8

Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. In *European Conference on Computer Vision, ECCV*, 2016. 2

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *arXiv preprint*, 2023. 2

Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, Jiazheng Xu, Bin Xu, Juanzi Li, Yuxiao Dong, Ming Ding, and Jie Tang. Cogvlm: Visual expert for pretrained language models. *arXiv preprint*, 2023. 3

Fangyun Wei, Jinjing Zhao, Kun Yan, Hongyang Zhang, and Chang Xu. A large-scale human-centric benchmark for referring expression comprehension in the lmm era. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. 2, 5

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022. 1

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. *arXiv preprint*, 2023. 1, 2, 3, 5, 9

Chi Xie, Zhao Zhang, Yixuan Wu, Feng Zhu, Rui Zhao, and Shuang Liang. Described object detection: Liberating object detection with flexible expressions. In *Conference on Neural Information Processing Systems, NeurIPS*, 2023. 1

Yiyang Yao, Peng Liu, Tiancheng Zhao, Qianqian Zhang, Jiajia Liao, Chunxin Fang, Kyusong Lee, and Qing Wang. How to evaluate the generalization of detection? A benchmark for comprehensive open-vocabulary detection. In *Conference on Artificial Intelligence, AAAI*, 2024. 1

Lang Yu, Qin Chen, Jiaju Lin, and Liang He. Black-box prompt tuning for vision-language model as a service. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI*, 2023. 3

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions. In *European Conference on Computer Vision, ECCV*, 2016. 2, 5

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *ACL*, 2019. 9, 15

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *International Conference on Computer Vision, ICCV*, 2023. 1

A ADDITIONAL RESULTS: LLM-WRAPPER ON ALREADY REC-TUNED VLMS

We explore the use of LLM-wrapper on VLMs that are already optimized to deal with the REC task. This supplementary experiment aims to confirm whether LLM-wrapper is compatible with pre-existing REC-specific adaptations. Our analysis includes two VLMs from our main experiments after white-box fine-tuning on REC data i.e., GDrec SwinB and Flo2 fine-tuned. We also include an additional VLM, Kosmos-2 (Peng et al., 2023), that is directly designed to ground referring expressions. For each candidate VLM, we compare P@1 scores, with and without additional LLM-wrapper tuning using RefCOCOg-train (umd) data. The evaluation is made on RefCOCOg val-umd and test-umd and results are shown in Table 8.

Table 8: **Results of LLM-wrapper on the REC task**, in P@1↑, when applied to already REC-adapted VLMs. ('FT' stands for 'fine-tuning' and '(B)' for the 'SwinB' backbone.)

			RefCOCOg	
Adaptation	**VLM**	**LLM**	val-umd	test-umd
White-box FT	GDrec(B)		80.19	79.85
White-box FT + LLM-wrapper	GDrec(B)	Mixtral	82.58 ↑**2.39**	81.95 ↑**2.10**
White-box FT + LLM-wrapper	GDrec(B)	Llama3	81.66 ↑**1.47**	81.47 ↑**1.62**
White-box FT	Flo2 FT		90.32	91.06
White-box FT + LLM-wrapper	Flo2 FT	Mixtral	90.38 ↑**0.06**	90.92 ↓**0.14**
White-box FT + LLM-wrapper	Flo2 FT	Llama3	90.50 ↑**0.18**	91.03 ↓**0.03**
Designed for REC	Kosmos-2		60.60	61.41
Designed for REC + LLM-wrapper	Kosmos-2	Mixtral	62.03 ↑**1.43**	62.59 ↑**1.18**
Designed for REC + LLM-wrapper	Kosmos-2	Llama3	62.09 ↑**1.49**	62.39 ↑**0.98**

While the performance gains are modest — an expected outcome as the VLMs are already adapted and optimized for the REC task —, it is important to note that LLM-wrapper avoids degrading performances by any significant value and, in some cases, achieves a slight improvement of up to +2.4 P@1. These results highlight the compatibility of LLM-wrapper with any state-of-the-art VLM, whether previously fine-tuned on REC data or not.

B ADDITIONAL ABLATIONS ON POSSIBLE VARIABLES IMPACTING PERFORMANCES

B.1 IMPACT OF THE NUMBER OF FINE-TUNED PARAMETERS ON PERFORMANCE

Our ablation study on LoRA's rank r, presented in Figure 4b, shows the P@1 variations for different r values. As the number of trainable parameters scales linearly with LoRA's rank r, this analysis highlights how trainable parameters count impacts performance. To ablate more precisely this aspect, we analyze the impact of the number of fine-tuned parameters on the P@1 performance in Table 9. In this table, we also report the original performance of the respective LLMs on the standard MMLU benchmark (Hendrycks et al., 2021) and HellaSwag (Zellers et al., 2019), a benchmark for LLM commonsense reasoning evaluation. As part of our analysis, we also compute the Pearson correlation between the P@1 scores and other variables from Table 9 and summarize results in Table 10.

Table 10 shows that the absolute number of fine-tuned parameters has a loose correlation with performance (pearson=-0.07). This supports findings from Table 9 showing the limited impact of the number of parameters, such as:

- Variations in LoRA's rank r do not have a large effect on the P@1 score of Llama3 8B on RefCOCOg (cf Table 9, rows 1 to 4).
- Llama3 8B ($r = 64$, 176M trainable parameters) and Mixtral 8×7B ($r = 128$, 114M trainable parameters) yield very similar P@1 scores (77.72 vs. 77.57), despite Llama3 8B fine-tuning a much higher percentage (2.15% vs. 0.24%) of parameters. The similarity of results, despite different fine-tuning settings, could then be due to architectural differences, with Mixtral being a Mixture of Experts model.

Table 9: **Impact of the number of trainable parameters** on `LLM-wrapper`'s performances on RefCOCOg.

LLM	Rank (r)	Trainable params	Total params	Trainable params %	P@1 (val splits)	LLM MMLU	LLM HellaSwag
Llama3 8B	12	33M	8.1B	0.41 %	77.84	66.6	82
Llama3 8B	64	176M	8.2B	2.15%	77.72	66.6	82
Llama3 8B	128	352M	8.4B	4.20 %	78.12	66.6	82
Llama3 8B	192	529M	8.6B	6.18%	77.70	66.6	82
Mixtral 8x7B	128	114M	46.8B	0.24 %	77.57	70.6	84.4
Gemma-2 9B	128	465M	9.7B	4.79 %	74.12	71.3	81.9
Gemma-2 2B	128	199M	2.8B	7.08 %	70.51	52.2	72.9

Table 10: **Pearson Correlation with `LLM-wrapper`'s P@1** for various variables.

Variable	Pearson Correlation with P@1
LLM Total Params (count)	0.33
LLM Trainable Params (count)	-0.07
LLM Trainable Params (%)	-0.64
LLM MMLU score	0.72
LLM HellaSwag score	0.88

- Gemma-2 9B ($r = 128$, 465M trainable params) underperforms compared to Llama3 8B ($r = 64$, 176M trainable params), even though its setting includes fine-tuning more than double the number of parameters.

These findings suggest that the architecture of the LLM and training specifics may have more influence on performances. Furthermore, our analysis finds a strong correlation between `LLM-wrapper`'s P@1 scores and the LLM's performances on reasoning tasks, with a Pearson correlation of 0.88 with HellaSwag. This indicates that `LLM-wrapper`'s performance is mostly dependent on the original performance of the LLM on reasoning tasks. In conclusion, the number of fine-tuned parameters has a small impact, but the LLM's architecture and other training factors play a more significant role regarding performance.

B.2 Impact of the complexity of referring expressions on performance

To ablate the robustness of `LLM-wrapper` to the degree of complexity of referring expressions, we count the average number of words and noun groups[1] for all queries in the datasets' validation splits (for instance, the sentence '*Green plant behind a table visible behind a lady's head*' yields three noun groups: '*Green plant*', '*a table*', and '*a lady's head*'). Results are summarized in Table 11.

Table 11: **`LLM-wrapper`'s performance with respect to referring expression complexity**, on all our datasets. (All statistics and P@1 scores are computed on datasets' val splits. LLM-wrapper is implemented on `Flo2`, with Llama 3).

Dataset	Avg # of words	Avg # of noun groups	Avg. # of boxes	P@1 ↑ (val splits)	P@1 boost ↑ (wrt 0-shot VLM)
RefCOCO	3.6	1.4	2.8	70.19	+13.9
RefCOCO+	3.6	1.6	2.8	61.95	+8.2
RefCOCOg	8.3	2.7	3.7	77.94	+9.7
HC-RefLoCo	84.4	22.9	17.6	67.40	+18.8

`LLM-wrapper` demonstrates robust performances across datasets, showing no clear correlation between the number of entities in referring expressions and either the P@1 score or the performance boost.

[1] Noun groups in text queries are identified using Spacy's *noun_chunks* method (Honnibal et al., 2020).

Interestingly, on the challenging HC-RefLoCo dataset, which features long queries with many entities (more than 20 noun groups per query on average), `LLM-wrapper` maintains strong reasoning capabilities and performs well on the numerous boxes proposed by the VLM (more than 17 per prompt on average).

C Failure case analysis

A failure case arises when the LLM does not succeed in producing only the index of a candidate bounding box as output. In such cases, the LLM may output non-integer values or indices that are out of range of the candidate boxes' list. However, these issues are very rare. Indeed, in Table 12, we report the percentage of invalid outputs, when `Flo2` is adapted with `LLM-wrapper`, with either Mixtral or Llama3. We observe that issues due to invalid generation outputs occur on less than 0.5% of the evaluation sets. In this rare cases, we use a simple fallback strategy: the best-ranked box from the zero-shot VLM is used as prediction.

Table 12: **Percentages of invalid LLM generation** when using `LLM-wrapper` on `Flo2`

		RefCOCOg		**RefCOCO**		**RefCOCO+**	
Method	**LLM**	**val**	**test**	**val**	**test**	**val**	**test**
`LLM-wrapper`	Mixtral	0.02%	0.05%	0.03%	0.10%	0.13%	0.09%
`LLM-wrapper`	Llama3	0.02%	0%	0.41%	0.35%	0.16%	0.18%

For qualitative examples of failure cases due to reasoning issues, rather than generative issues, visualizations are given in Appendix D.2.

D Additional qualitative examples

In this entire section, we use `LLM-wrapper` to adapt a zero-shot `Flo2` VLM, using Llama3 8B to perform black-box fine-tuning on RefCOCOg data.

D.1 Qualitative success cases on longer queries

We show qualitative examples of `LLM-wrapper`'s successes against white-box fine-tuned `Flo2` on HC-RefLoCo in Figure 6 and Figure 7. In these two examples, `LLM-wrapper` is able to properly process more than 10 candidate boxes, aligned with long and complex queries, to identify the correct box, while fine-tuned `Flo2` is predicting distractor objects.

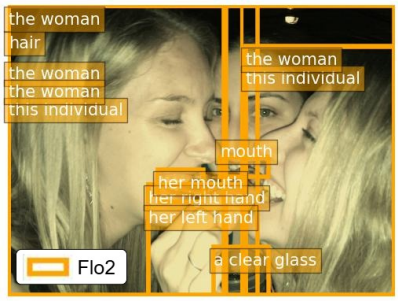
D.2 Qualitative failure cases

We display qualitative examples of `LLM-wrapper`'s failures against zero-shot `Flo2` on RefCOCOg. In particular, as mentioned in Section 5, the main failure case of `LLM-wrapper` occurs when bounding box coordinates and labels alone are insufficient to ground certain referring expressions that require additional visual cues. For example, in Fig. 2f, correctly localizing the query 'White plane in the front' requires knowing the orientation of the planes. We give more examples showing this type of failures in Figure 8. For instance, `LLM-wrapper` is missing visual information on the zebra's head direction in Fig. 8a, on the relative position of the curtains and chairs in Fig. 8b and on the position of each person with respect to the camera in Fig. 8c.

Going beyond `LLM-wrapper`'s main failure scenario, we add visualizations of corner failure cases in Figure 9. They illustrate possible issues hindering the LLM's reasoning, i.e., when no proper candidate box is identified by the zero-shot VLM in the first place (Fig. 9a), when the VLM fails to detect important contextual objects, necessary to 'perceive' the scene and localize the object of interest (Fig. 9b), when rich candidate boxes exist but without adequate labels (Fig. 9c).

Query: *"This individual appears to be a woman with long, straight blonde hair that drapes over her right shoulder. She is wearing a light-colored, possibly cream or pale yellow blazer. The woman is engaged in an activity where she is bringing a clear glass, which she holds in her right hand, towards her mouth as if to take a sip of a beverage. Her left hand is not visible in the image. She is positioned on the far left in the group of people captured in the photograph."*

(a) Ground truth



(b) All 13 candidates from `Flo2`

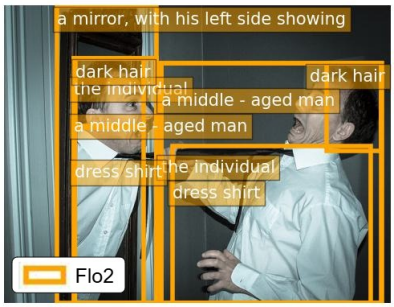
(c) Final pred. of `LLM-wrapper`

(d) Final pred. of `Flo2` FT

Figure 6: **First qualitative result of `LLM-wrapper` tuned on RefCOCOg, evaluated on HC-RefLoCo**. `LLM-wrapper` takes multiple candidates from zero-shot `Flo2` as inputs (in **orange** in Fig. b) to identify the best box (in **blue**, in Fig. c), while white-box fine-tuned `Flo2` fails (in **red** in Fig. d).

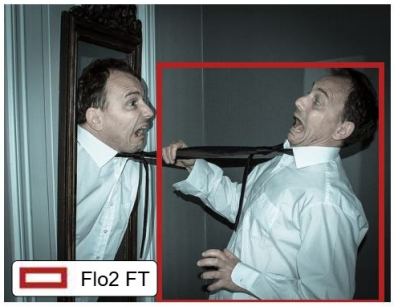
Query: *"The individual is a middle-aged man with short, dark hair, appearing startled or comically alarmed. He is wearing a pale dress shirt and is positioned as if emerging from a mirror, with his left side showing."*

(a) Ground truth



(b) All 10 candidates from `Flo2`

(c) Final pred. of `LLM-wrapper`

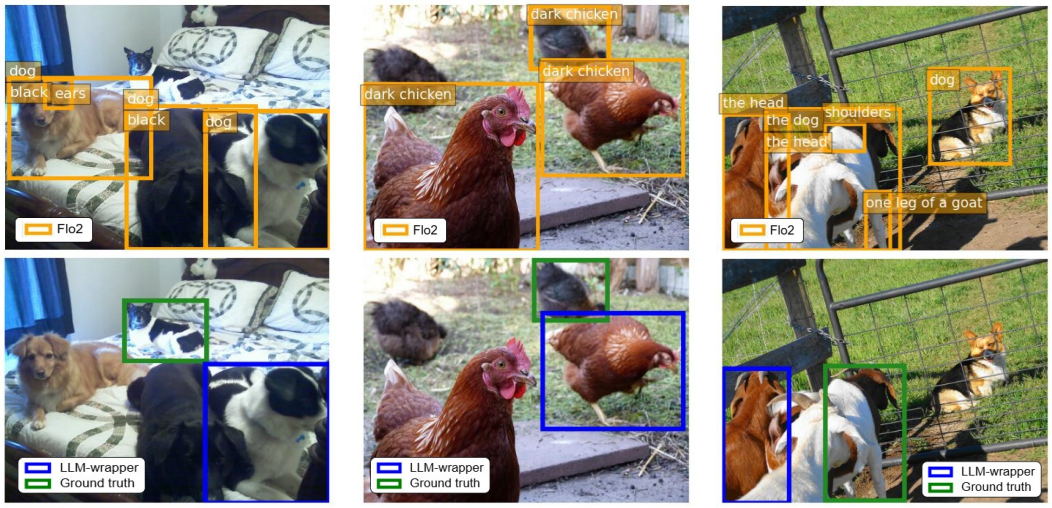


(d) Final pred. of `Flo2` FT

Figure 7: **Second result of `LLM-wrapper` evaluated on HC-RefLoCo**. Same legend as Figure 6.

(a) Query: *"Zebra whose head is not facing down."* (b) Query: *"Chair without the curtain on it"* (c) Query: *"A man with his back turned toward the camera, enjoying a conversation with a friend."*

Figure 8: **Visualizations on RefCOCOg of `LLM-wrapper`'s main failure case**. `LLM-wrapper`'s predictions are in **blue** vs. zero-shot `Flo2`'s predictions in **orange**. In these three examples, `LLM-wrapper` fails to identify the best candidate box as coordinates and labels are not providing enough visual cues for the LLM to choose correctly. For instance, `LLM-wrapper` is missing visual information on the zebra's head direction (Fig. a), on the relative position of the curtains and chairs (Fig. b) and on the position of each person with respect to the camera (Fig. c).



(a) Query: *'Black and white dog with pointy ears.'* (b) Query: *'Dark chicken closest to the fence.'* (c) Query: *'The head and shoulders and one leg of a goat closest to the dog.'*

Figure 9: **Visualizations of three corner failure cases of `LLM-wrapper` on RefCOCOg.** As we use `LLM-wrapper` to adapt a zero-shot `Flo2`, we show on the first row in **orange**, for each sample, the respective `Flo2`'s candidate boxes that `LLM-wrapper` takes as inputs. In the second row, we visualize the prediction of `LLM-wrapper` in **blue**, chosen among `Flo2`'s candidates, as well as the ground truth box. We observe that the LLM reasoning can be hindered if candidate boxes are missing the object of interest (the correct dog in Fig. a), necessary contextual objects (the fence in Fig. b) or proper labels (goats are detected but not properly labeled in Fig. c, bringing confusion with respect to the scene description).