PerceptionGPT: Effectively Fusing Visual Perception into LLM

Renjie Pi[1] Lewei Yao[1] Jiahui Gao[2] Jipeng Zhang[1] Tong Zhang[1]
[1]The Hong Kong University of Science and Technology
[2]The University of Hong Kong

Abstract

*The integration of visual inputs with large language models (LLMs) has led to remarkable advancements in multi-modal capabilities, giving rise to vision large language models (VLLMs). However, effectively harnessing LLMs for intricate visual perception tasks, such as detection and segmentation, remains a challenge. Conventional approaches achieve this by transforming perception signals (e.g., bounding boxes, segmentation masks) into sequences of discrete tokens, which struggle with the precision errors and introduces further complexities for training. In this paper, we present a novel end-to-end framework named **PerceptionGPT**, which represent the perception signals using LLM's dynamic token embedding. Specifically, we leverage lightweight encoders and decoders to handle the perception signals in LLM's embedding space, which takes advantage of the representation power of the high-dimensional token embeddings. Our approach significantly eases the training difficulties associated with the discrete representations in prior methods. Furthermore, owing to our compact representation, the inference speed is also greatly boosted. Consequently, PerceptionGPT enables accurate, flexible and efficient handling of complex perception signals. We validate the effectiveness of our approach through extensive experiments. The results demonstrate significant improvements over previous methods with only 4% trainable parameters and less than 25% training time.*

1. Introduction

The rapid advancements in deep learning and natural language processing have given rise to large language models (LLMs) capable of comprehending and generating human-like text [3, 4, 6, 8, 31, 37, 38, 42]. Recently, the development of visual large language models (VLLMs), which combine visual inputs with LLMs, has demonstrated impressive multi-modal capabilities and opened up new possibilities beyond text-based tasks [2, 7, 24, 30, 39, 52].

However, enabling VLLMs to perform complex visual perception tasks, such as object detection and segmentation, remains a significant challenge. Current state-of-the-art approaches can be divided into two categories: 1) two-stage-based approaches that leverage a vision expert alongside the reasoning ability of the LLM to handle visual perception tasks [19, 33, 41, 48]. While these approaches excel at visual tasks, their reliance on an external vision expert makes them inflexible. In addition, the VLLMs of such methods lack the ability to truly interpret visual perception signals, thereby limiting their applicability; 2) End-to-end approaches that integrate visual perception capabilities into the LLM [2, 5, 32, 44] , which we refer to as perception-enhanced vision-language models (P-VLMs). These approaches enable the model to encode and decode **visual perception signals** (e.g., bounding boxes, segmentation masks, depth map, etc.) by themselves, without using external visual experts, and further endows the model to interpret perception information. However, the design choices of previous P-VLM approaches demonstrate several weaknesses, which not only affect the performance, but also poses challenges during training.

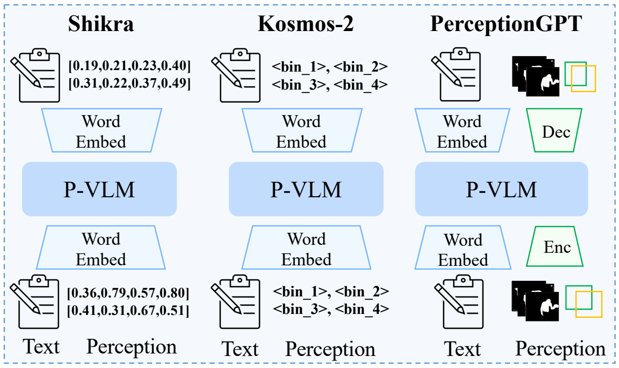


Figure 1. Illustration of different strategies to encode and decode visual perception information. Previous approaches formulate the visual information into discrete tokens in the same way as text. On the other hand, our PerceptionGPT leverages lightweight visual encoder (Enc) and decoders (Dec) to fuse visual perception signals into the embedding space of LLM.

In contrast to natural languages, visual perception signals are inherently continuous and lack causal dependency.

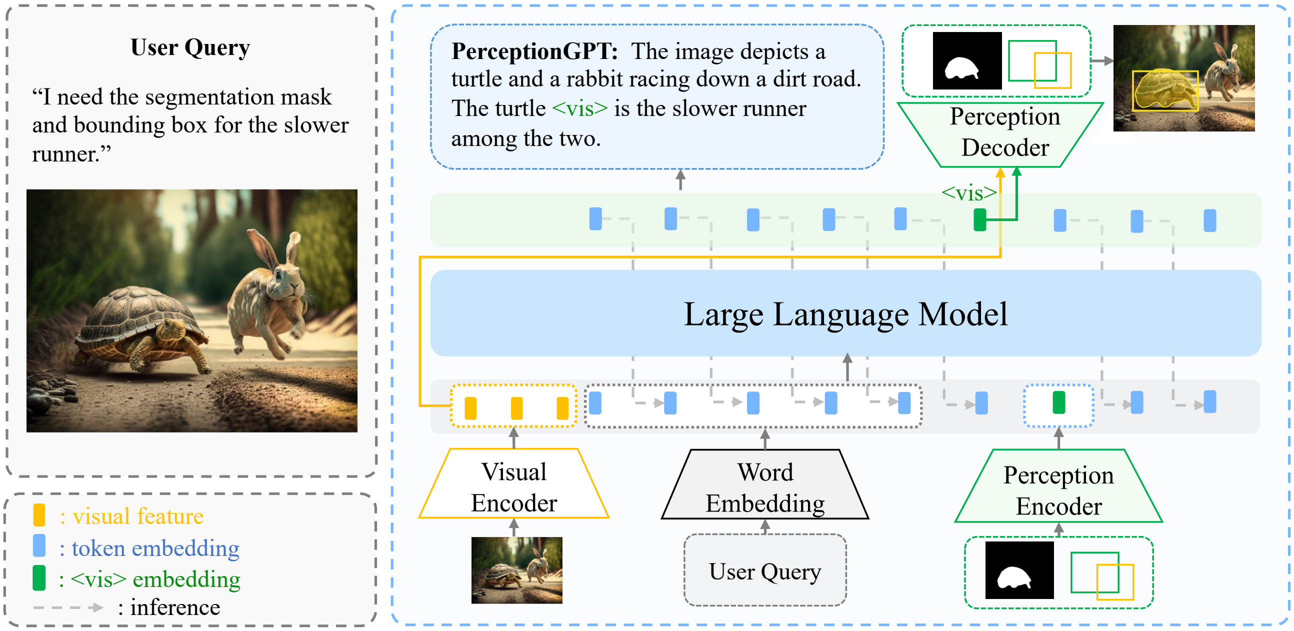

User Query
"I need the segmentation mask and bounding box for the slower runner."
: visual feature
: token embedding
: <vis> embedding
: inference
PerceptionGPT: The image depicts a turtle and a rabbit racing down a dirt road. The turtle <vis> is the slower runner among the two.
Perception Decoder
<vis>
Large Language Model
Visual Encoder
Word Embedding
Perception Encoder
User Query

Figure 2. The illustration of PerceptionGPT framework. Rather than outputting the location and coordinates in the form of discrete tokens, each box and mask can be represented by one single dynamic embedding, and handled by visual perception encoders and decoders.

For instance, segmentation masks can be presented in arbitrary shapes, and the pixels within the masks are not interdependent. However, previous P-VLM approaches overlook these distinctions and indiscriminately represent perception signals as sequences of discrete tokens, which introduces several issues: 1) discretizing continuous perception signals unavoidably introduces precision errors, potentially leading to decreases in accuracy; 2) discrete tokens are limited in their ability to express continuous signals, resulting in redundant tokens. For instance, representing the contour of a segmentation mask requires more than 30 tokens [43, 51], while completely disregarding the mask's interior; 3) this sub-optimal discrete representation adds complexity to the training process. Specifically, all parameters in the model need to be released for training, necessitating a considerable training time and significant computational resources. For example, Shikra [5] takes 960 GPU hours on 80G A100, while Kosmos-2 [32] takes 6144 GPU hours on 32G V100.

In this paper, we propose **PerceptionGPT**, a novel framework that bypasses the discretization of perception signals and represent them in their inherent continuous forms. Our fundamental insight is that the LLM's high-dimensional token embedding is able to capture the essential information to represent the perception signals. This potential was not harnessed by previous approaches due to discrete representation. Specifically, we introduce a unique token called <*vis*> , which acts as a marker within the context to indicate the presence of a perception signal. Unlike static discrete tokens employed by previous methods [5, 19], the embedding of <*vis*> is dynamic, which is capable of encompassing a wide range of perception information. For example, by using a lightweight encoder, we can encode segmentation masks of various shapes into the embedding. Consequently, the <*vis*> 's embedding can be decoded back into these masks, adapting their shapes based on the preceding context, through a lightweight decoder.

Owing to the design of our framework, we are able to use a combination of auto-regressive language modeling loss and objective functions specifically designed for vision tasks (e.g., GIoU loss for bounding box [36], DICE loss for segmentation masks [40]). The language modeling loss enables the Large Language Model (LLM) to generate responses to user inputs and to decide when to produce visual outputs through the generation of <*vis*> tokens. Concurrently, the task-specific losses empower the model with enhanced visual perception capabilities, which further enables efficiently acquiring visual perception ability by taking advantage of the intrinsic properties of the perception signals.

Our proposed method, **PerceptionGPT**, offers several notable advantages. Firstly, by leveraging dynamic token embeddings to represent perception signals, PerceptionGPT significantly mitigate the training difficulty. Consequently, PerceptionGPT achieves superior performance by tuning less than 4% of the parameters compared to previous approaches (see Table 6). Secondly, our approach enables more accurate representations of visual perception by predicting exact values, effectively addressing the issue of precision errors. Thirdly, in contrast to previous methods,

	Model	Image	Region	Bounding	Segmentation	Multi-Instance	End-to-End
		Caption	Caption	Box		Segmentation	
Two Stage	Visual ChatGPT [46]	✓	✗	✓	✓	✓	✗
	DetGPT [33]	✓	✗	✓	✗	✗	✗
	LISA [19]	✓	✗	✗	✓	✗	✗
End-to-End	MiniGPT-4 [52]	✓	✗	✗	✗	✗	✓
	LLaVA [24]	✓	✗	✗	✗	✗	✓
	InstructBLIP [9]	✓	✗	✗	✗	✗	✓
	GPT4RoI [50]	✓	✓	✗	✗	✗	✓
	Shikra [5]	✓	✓	✓	✗	✗	✓
	Kosmos-2 [32]	✓	✓	✓	✗	✗	✓
	PerceptionGPT	✓	✓	✓	✓	✓	✓

Table 1. Comparisons of functionalities supported by different vision-language models. Our PerceptionGPT is an end-to-end model that supports image-level and region-level understanding, object localization and segmentation tasks.

we only require the dynamic embedding of a single token, <*vis*> , to represent each perception signal (see Table 4). This eliminates the need for redundant tokens and significantly accelerates the decoding process. We specifically demonstrate the effectiveness of PerceptionGPT through two representative visual perception tasks: detection and segmentation. However, our approach can easily be extended to integrate other perception tasks such as depth or pose estimation.

We summarize the contributions of our paper as follows:

- We propose a novel framework for efficiently training perception-enhanced vision language model using dynamic token embedding to represent perception signals.
- Our approach eliminates the precision error and reduces the redundant tokens suffered by previous methods, which not only boosts the performance, but also increases the inference efficiency.
- We conduct extensive experiments on various benchmarks. Notably, we achieve competitive performances on referring expression comprehension and referring expression segmentation tasks with less than 4% parameters and 25% training time of Shikra [5].

2. Related works

2.1. Vision Large Language Models

In recent years, significant progress has been made in large language models [3, 4, 8, 14, 31, 37, 38, 42], pushing the boundaries of language understanding and generation, which have demonstrated human-level abilities in various tasks. The success of language models has also driven research on vision-language interaction, resulting in the development of various multi-modal models [2, 9, 9, 11, 20, 24, 30, 39, 52]. These models have demonstrated promising performances in generating detailed descriptions and conducting conversations based on images.

2.2. Two-stage Vision Language Assistant

Recent research trends merge LLMs with vision expert models for tasks needing reasoning. API-based approaches [13, 41, 46, 48] use LLMs as planners for visual expert APIs. DetGPT [33] applies VLLMs for instruction interpretation and uses external detectors for object localization. LISA [19] combines VLLMs with the Segmentation Anything Model (SAM) [17] for predicting segmentation mask. Although those methods excel on visual perception tasks, they require an external vision expert, which limits their flexibility and applicability for tasks that lack such expert models. In addition, the VLLM is still unable to understand the perception signal as inputs.

2.3. Perception-Enhanced Vision Language Model

More recently, a few works [2, 5, 32, 44, 50] have made the initial attempt to integrate visual perception capability into LLMs, which have demonstrated promising results and open up a series of new possibilities. These model mainly represent the perception information as a series of discrete tokens. Specifically, [32, 44] introduce new tokens into the LLM to represent the 2D coordinates, while [5] directly use numbers to represent the bounding boxes, which improves accuracy at the cost of longer sequence lengths and slower inference. Despite the success of such methods, their discrete formulation of perception signals faces disadvantages such as precision error, redundant tokens and difficulty in training. On the other hand, we propose a framework to represent the perception signals in their natural continuous forms, which addresses the above issues and provides a solution for training a strong P-VLM efficiently.

3. Method

In this section, we present our PerceptionGPT, designed to equip the Perception-enhanced Vision-Language Model (P-VLM) with advanced visual perception capabilities.

Model Type	Method	RefCOCO			RefCOCO+			RefCOCOg	
		val	testA	testB	val	testA	testB	val	test
Specialist SOTAs	SeqTR [51]	83.72	86.51	81.24	71.45	76.26	64.88	74.86	74.21
	MDETR [15]	87.51	90.40	82.67	81.13	85.52	72.96	83.35	83.31
	G-DINO-L [25]	90.56	93.19	88.24	82.75	88.95	75.92	86.13	87.02
Generalist VL SOTAs	GPV-2 [16]	51.59	-	-	-	-	-	-	-
	OFA-L [43]	79.96	83.67	76.39	68.29	76.00	61.75	67.57	67.58
	Unified-IO [26]	78.60	-	-	-	-	-	-	-
	OFASys [1]	-	80.10	-	-	-	-	-	-
	VisionLLM-H [44]	-	86.70	-	-	-	-	-	-
	Shikra-7B [5]	87.01	90.61	80.24	81.60	87.25	73.20	82.27	82.19
	Shikra-13B [5]	87.83	91.11	81.81	82.89	87.79	74.41	82.64	83.16
	PerceptionGPT-7B	**88.59**	**92.51**	**84.60**	**82.05**	**88.60**	**74.21**	**83.75**	**84.69**
	PerceptionGPT-13B	**89.17**	**93.20**	**85.96**	**83.72**	**89.19**	**75.31**	**84.13**	**85.20**

Table 2. Results on referring expression comprehension (REC). We compare our PerceptionGPT with both the generalist models and specialist models. Our method achieves competitive performance on REC tasks amongst generalist models and comparable with SOTA performance of specialist model, with only 4% trainable parameters as in Shikra [5].

3.1. Framework of PerceptionGPT

We illustrate the framework of PerceptionGPT in Figure 2. The model mainly consists of a large language model (LLM) such as Vicuna [6], a pretrained vision transformer [35] (ViT) as image encoder, and a set of lightweight visual perception encoders and decoders. In addition, a projection layer is required to map the visual features from the image encoder to the same dimension as the LLM.

Dynamic Token Embedding Rather than representing visual perception signals using discrete tokens as in previous approaches [2, 5, 26, 32, 43, 44], we resort to the LLM's dynamic token embedding. Specifically, we introduce a special token <*vis*> , which indicates the presence of a perception signal. Unlike previous P-VLM that employ static tokens, the embedding of <*vis*> is dynamic and can represent various perception signals, such as bounding boxes of any sizes, or masks with arbitrary shapes.

Lightweight Visual Perception Encoder-Decoder Our PerceptionGPT leverages specially designed modules for encoding different perception signals into the dynamic token embedding of <*vis*> , and for restoring the those signals back to their original representation from the dynamic embedding. The architecture choices of such modules to process perception signals can be flexible, which could be determined based on the property of the signal. In our implementation, we includes two perception encoder-decoder pairs, each for detection and segmentation, respectively. For bounding box detection, bot the encoder and decoder are simple three-layer MLPs. For segmentation masks, the encoder comprises a ResNet followed by a linear layer, and the decoder consistes of a two-layer, bidirectional transformer block architecture. We leave the detailed design choices in the Appendix.

Multi-Layer Visual Feature Fusion Previous VLLM approaches predominantly depend on the visual feature from last layer of pretrained ViT. However, this is suboptimal for perception tasks, since the representations from top layers usually contain richer semantic features, while lacking fine-grained visual information. Inspired by layer-wise feature aggregation in computer vision [22], we propose to make use of visual features across all layers of ViT. Specifically, we learn an adaptive weighting term for each layer, and leverage the weighted-sum of those layer representations:

$$V = \sum_{i=1}^{n} w_i \cdot V_i \quad \text{s.t.} \quad \sum_{i=1}^{n} w_i = 1 \tag{1}$$

where V is the input image feature to the LLM, w_i is the learnt weighting for image feature V_i from i^{th} layer. We demonstrate the impact of layer fusion in Section 4.4.

3.2. Training and Inference

Training Objective The design of our PerceptionGPT allows us to harness the benefits of purposefully crafted training objectives for visual perception tasks, which take advantage of the special characteristics inherent to the perception signals. Throughout the training process, we employ a combination of language modeling loss and task-specific losses. The overall objective of PerceptionGPT during training is:

$$\mathcal{L}_{\text{all}}(S_{tar}, S_{in}, P_{tar}, I) = \mathcal{L}_{\text{lang}}(S_{tar}, S_{in}, I) \tag{2}$$

$$+\mathcal{L}_{\text{vis}}(S_{in}, P_{tar}, I) \tag{3}$$

where I is the input image, S_{tar}, S_{in}, P_{tar} are the target text, input instruction and target for visual perception signal, respectively. In our case, since we incorporate bounding box and segmentation mask as perception signals, our objective function can be formulated as:

$$\begin{aligned} \mathcal{L}_{\text{all}}(S_{tar}, S_{in}, b_{\text{gt}}, m_{\text{gt}}, I) &= \mathcal{L}_{\text{lang}}(S_{tar}, S_{in}, I) \quad (4) \\ &+ \mathcal{L}_{\text{box}}(b_{\text{gt}}, S_{in}, I) + \mathcal{L}_{\text{mask}}(m_{\text{gt}}, S_{in}, I) \quad (5) \end{aligned}$$

where b_{gt} and m_{gt} are the ground truth bounding boxes and segmentation masks, respectively.

We adopt conventional auto-regressive loss for $\mathcal{L}_{\text{lang}}(\cdot)$:

$$\mathcal{L}_{\text{lang}}(S_{tar}, S_{in}, I) = -\sum_{t=1}^{L} \log p\left[s_{tar}^{t} | \mathcal{F}(s_{tar}^{(<t)}, S_{in}, I)\right] \quad (6)$$

where $\mathcal{F}$ represents the P-VLM. I represents the image; y_t denotes the t^{th} token of the target output, and L stands for its length. $\mathcal{L}_{\text{lang}}$ supervises the model to generate corresponding output sentences based on the image and the input texts. In addition, it also teaches the model when to generate the <*vis*> for predicting the perception signal.

For bounding box loss $\mathcal{L}_{\text{box}}(\cdot)$, we adopt the combination of L1-norm and GIoU [36] losses; for mask loss $\mathcal{L}_{\text{mask}}(\cdot)$, we combine binary cross-entropy loss (BCE) with DICE loss [40]. We leave the details of those losses in the Appendix. The use of visual task-specific training objectives helps take advantage of the inherent property of such perception signals, which not only boosts performance, but also alleviates the difficulty for training.

Inference Procedure During inference, given an image and a textual instruction, the image encoder first extracts the visual tokens from the image, which are then mapped to the dimension of LLM's embedding space via the projection layer. Then, the mapped image features are concatenated with text embeddings to serve as the input to the LLM. Subsequently, the LLM begins to perform next-token-generation similar to previous VLLMs [24, 52].

Perception Signal as Input. When the input contains a perception signal, our lightweight perception encoder maps it into the embedding space of the large language model (LLM), which is treated as the embedding for <*vis*> token. This embedding is then concatenated with other embeddings before being processed by the LLM.

Perception Signal as Output. During inference, when a token is decoded as <*vis*> , its associated embedding is extracted and processed by the perception decoder to reconstruct the original signal.

4. Experiments

4.1. Training and Evaluation

Datasets Similar as in Shikra [5], to equip PerceptionGPT with visual perception ability, we adopt RefCOCO [49], RefCOCO+ [49], RefCOCOg [28], Visual Gemone [18] and Flicker30k [34]. Since Visual Genome and Flicker30k do not have segmentation mask annotations, we leverage the powerful SAM [17] as an auto-labelling system to generate masks from bounding box annotations. For captioning, we leverage the COCO [21] dataset and the image caption data in curated by LLAVA [24].

Hyperparameters If not otherwise specified, we use the following hyper-parameters throughtout all experiments: We initialize the LLM component with Vicuna [6] weights, we adopt LoRA with rank set to 32, the learning rate is set to 3e-4, the batch size is 32 on each GPU during training. We run experiments on 8 A40 GPUs with 80G memory for 70 hours in total. For ablation study, we use Vicuna-7B as the LLM backbone to conduct experiments.

4.2. Qualitative Results

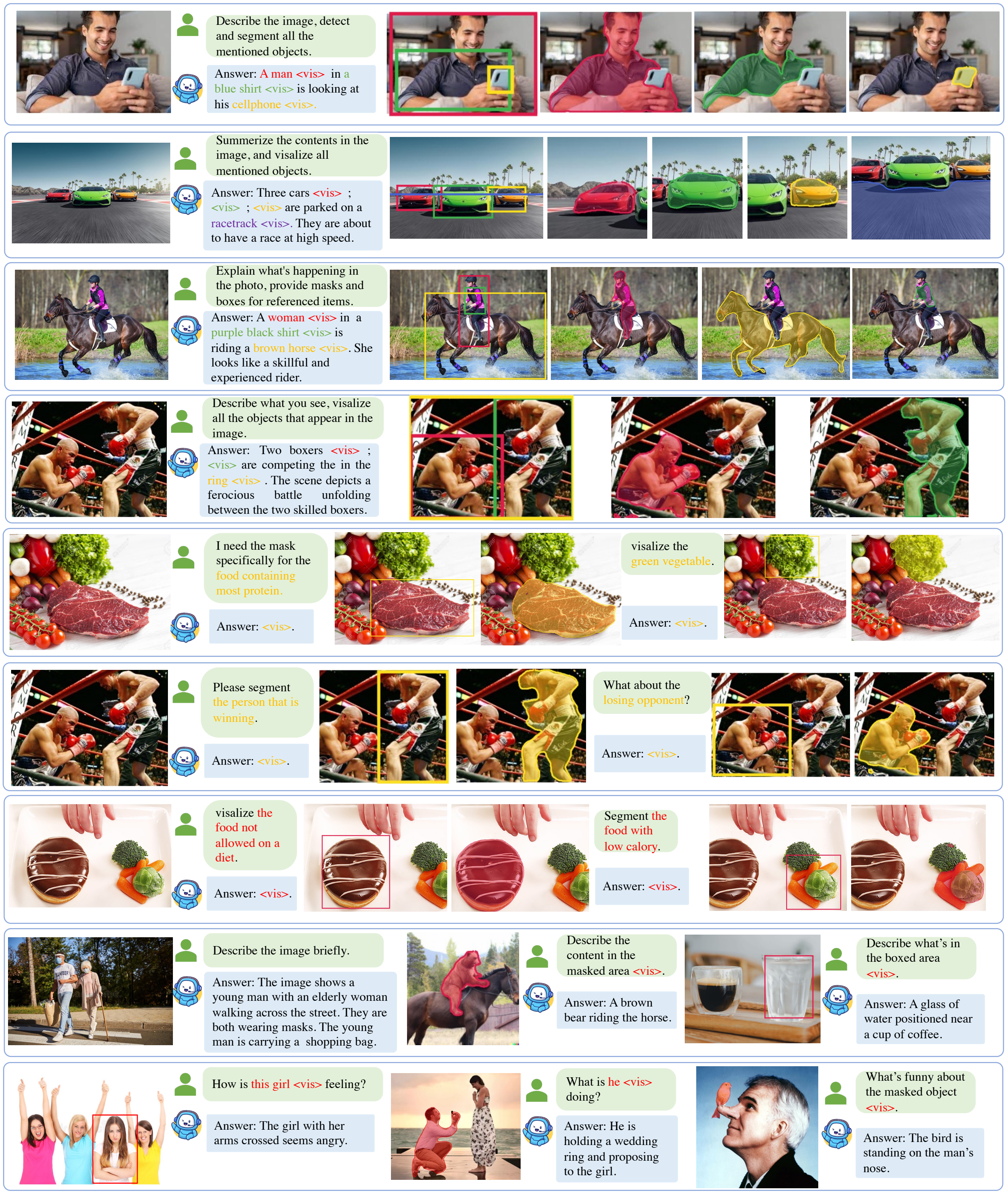
We demonstrate some generated results of our PerceptionGPT in Figure 3, which showcases the following capabilities: (1) Spotting Captioning (first 4 rows), which is capable of generating the captions while spoting the objects with boxes and segmentation masks. (2) Reasoning-based detection and Segmentation (row 5-7). (3) Image-level and region-level captioning and question-answering (row 8-9). We surprisingly find that PerceptionGPT is able to restore perception signal via reasoning, even though such ability is not specifically considered in the training data.

4.3. Quantitive Results

We show the performance of PerceptionGPT by conducting evaluation on a variety of benchmarks.

Refering Expression Comprehension (REC) The REC task mainly aims to understand the image and a textual phrase, and then localize the referred object by drawing a bounding box around it. This task requires understanding of both image contents and the textual phrase. We compare our PerceptionGPT with both generalist and specialist approaches in Table 2, which demonstrate the superior performance our our model. Notably, our PerceptionGPT outperforms Shikra [5] with only 4% trainable parameters.

Refering Expression Segmentation (RES) The RES task requires the prediction of the segmentation mask that separates the referred object from other contents in the image. Compared with the REC task, the RES task requires image-text understanding at the finer pixel level. We compare our PerceptionGPT with other approaches in Table 3, which demonstrates that PerceptionGPT is able to achieve SOTA for end-to-end approach, and performs on par, if not better than, the two-stage approach LISA [19] that leverages a powerful SAM [17] as visual expert.



Describe the image, detect and segment all the mentioned objects.

Answer: A man <vis> in a blue shirt <vis> is looking at his cellphone <vis>.

Summerize the contents in the image, and visalize all mentioned objects.

Answer: Three cars <vis> ; <vis> ; <vis> are parked on a racetrack <vis>. They are about to have a race at high speed.

Explain what's happening in the photo, provide masks and boxes for referenced items.

Answer: A woman <vis> in a purple black shirt <vis> is riding a brown horse <vis>. She looks like a skillful and experienced rider.

Describe what you see, visalize all the objects that appear in the image.

Answer: Two boxers <vis> ; <vis> are competing the in the ring <vis> . The scene depicts a ferocious battle unfolding between the two skilled boxers.

I need the mask specifically for the food containing most protein.

Answer: <vis>.

visalize the green vegetable.

Answer: <vis>.

Please segment the person that is winning.

Answer: <vis>.

What about the losing opponent?

Answer: <vis>.

visalize the food not allowed on a diet.

Answer: <vis>.

Segment the food with low calory.

Answer: <vis>.

Describe the image briefly.

Answer: The image shows a young man with an elderly woman walking across the street. They are both wearing masks. The young man is carrying a shopping bag.

Describe the content in the masked area <vis>.

Answer: A brown bear riding the horse.

Describe what's in the boxed area <vis>.

Answer: A glass of water positioned near a cup of coffee.

How is this girl <vis> feeling?

Answer: The girl with her arms crossed seems angry.

What is he <vis> doing?

Answer: He is holding a wedding ring and proposing to the girl.

What's funny about the masked object <vis>.

Answer: The bird is standing on the man's nose.

Figure 3. Visualization of results from PerceptionGPT. Our proposed framework enables effectively fusing visual perception capability into P-VLM while maintaining its generation and reasoning ability. Row [1-4], row [5-7] demonstrate spot captioning and reasoning segmentation/detection, respectively. Row [8-9] demonstrates image-level captioning and region-level captioning and question-answering.

Model Type	Method	RefCOCO			RefCOCO+			RefCOCOg	
		val	testA	testB	val	testA	testB	val	test
Two-stage	LISA [19]	74.9	79.1	72.3	65.1	70.8	58.1	67.9	70.6
End-to-end	MCN [27]	62.4	64.2	59.7	50.6	55.0	44.7	49.2	49.4
	VLT [10]	67.5	70.5	65.2	56.3	61.0	50.1	55.0	57.7
	CRIS [45]	70.5	73.2	66.1	62.3	68.1	53.7	59.9	60.4
	LAVT [47]	72.7	75.8	68.8	62.1	68.4	55.1	61.2	62.1
	ReLA [23]	73.8	76.5	70.2	66.0	71.0	57.7	65.0	66.0
	X-Decoder [53]	-	-	-	-	-	-	64.6	-
	SEEM [54]	-	-	-	-	-	-	65.7	-
	PerceptionGPT-7B	**75.1**	**78.6**	**71.7**	**68.5**	**73.9**	**61.3**	**70.3**	**71.7**
	PerceptionGPT-13B	**75.3**	**79.1**	**72.1**	**68.9**	**74.0**	**61.9**	**70.7**	**71.9**

Table 3. Results on refer segmentation (RES) task. Our PerceptionGPT significantly outperforms other end-to-end methods on all dataset splits, and also surpasses the two-stage approach LISA [19] by a large margin on majority of the splits.

Method	Quant Error Free	Box	Mask
Shikra	✗	20	NA
Unified-IO	✗	2	256
Kosmos-2	✗	2	NA
VisionLLM	✗	2	16
PerceptionGPT	✓	1	1

Table 4. Number of tokens needed to represent boxes or segmentation mask. Our PerceptionGPT is able to represent both box and mask with only one token, while prevents quantization error.

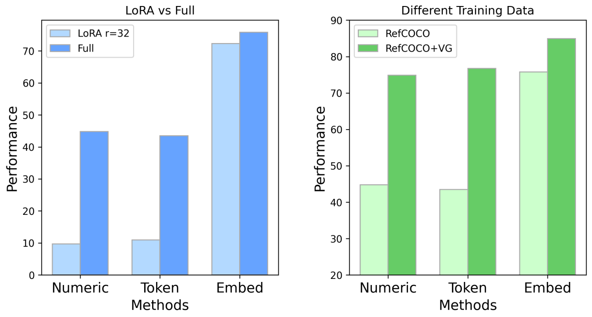


Figure 4. Performances of various perception signal representations. Left: full vs LoRA training; Right: different amount of training data. Dynamic token embedding alleviates training difficulty, making PerceptionGPT efficient in terms of both parameter and data.

Conventional Vision-Language Tasks We evaluate our PerceptionGPT's ability on the conventional vision-language tasks, namely image captioning (IC) and visual question answering (VQA). We finetune our PerceptionGPT on the training split of the datasets before evaluation and compare the results in Table 5, which demonstrate that our method is comparable with specialized and generalist models on conventional image-language tasks. The superiority of our method compared with Shikra can be attributed to the parameter-efficient training strategy that we adopt. Training only a small number of parameters alleviates the loss of original knowledge possessed by the LLM, which is made possible by the use of dynamic token embeddings to represent perception signals.

4.4. Ablation Study

PerceptionGPT Alleviates Learning Difficulty Representing visual perception signals using our dynamic token embedding greatly alleviates training difficulty. As shown in Table 6 and Figure 4, we compare with two representations: 1) numerical, as adopted by [5], directly use the number tokens in LLM; 2) Vocabulary, which introduces new coordinate tokens into the LLM's vocabulary, as used in [32] and [44]. We train the models with only the concatenated RefCOCO training splits, and observe that neither numerical nor vocabulary tokenizations performs well when trained with LoRA, while our PerceptionGPT is able to perform well even with rank set to 8, as shown in Table 6 and left of Figure 4. In right side of Figure 4, we also show that our approach enables achieving superior performance with less training data.

The above may be attributed to the following: 1) the discrete representations are suboptimal, since the spatial coordinates do not have causal relationship; 2) the discrete representations can not leverage the specially designed loss functions for vision tasks as in PerceptionGPT; 3) the dynamic token embeddings encapsulate rich perception information in a dense form, allowing for a more effective representation than a sequence of discrete tokens.

Impact of Layer Fusion We evaluate different stratgies for using Vision Transformers (ViT) visual features, as shown in Figure 5. We discover that upper layer features

Datasets	PerceptGPT	Shikra	FM-80B	FM-9B	Kosmos-1	BLIP-2	Unified-IO	VPGTrans
VQAv2	**85.1**	83.3	56.3	51.8	51.0	65.2	77.9	65.2
OK-VQA	**56.2**	53.8	50.6	44.7	-	45.9	54.0	45.0
Flickr30k	**77.1**	73.9	-	-	67.1	-	-	-
COCO	**123.2**	117.5	84.3	79.4	84.7	-	122.1	114.2

Table 5. Comparison on VQA and Image Captioning tasks. For VQA, we conduct evaluation on VQAv2 [12] and OK-VQA [29] using Accuracy (%) as the metric. For Image Captioning, we evaluate them on COCO [21] and Flickr30k [34] with CIDEr score.

	Trainable	Lora	RefCOCO			RefCOCO+			RefCOCOg	
Method	Params	Rank	val	testA	testB	val	testA	testB	val	test
Numerical	27M	8	0.79	0.63	0.51	0.28	0.36	0.25	0.43	0.41
	29M	32	10.5	10.1	9.64	8.72	10.4	9.27	8.63	10.2
	6.8B	full	45.7	53.6	49.2	43.5	44.1	39.7	42.1	41.3
Vocab	27M	8	1.31	0.97	1.24	0.59	0.63	0.42	0.94	0.80
	29M	32	13.5	12.1	10.4	9.69	11.4	10.2	9.90	9.84
	6.7B	full	44.9	49.1	47.2	42.3	42.1	40.6	42.2	39.5
Embed	31M	8	**69.7**	**73.2**	**72.5**	**68.4**	**70.3**	**68.1**	**71.6**	**70.5**
	33M	32	**71.2**	**75.4**	**74.1**	**70.5**	**71.9**	**69.6**	**72.5**	**72.9**
	6.8B	full	**75.5**	**79.6**	**78.1**	**74.9**	**74.4**	**72.1**	**76.4**	**75.2**

Table 6. Experiment on the influence of different representations for bounding boxes. We train with only the concatenated training sets of RefCOCO, RefCOCO+ and RefCOCOg. Neither numerical nor vocabulary representations perform well with LoRA training. On the other hand, leveraging our dynamic embedding representation, PerceptionGPT achieves good performances even using LoRA with low ranks.

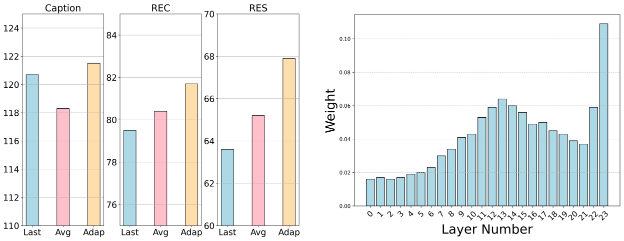


Figure 5. Left: The performance of different strategy for fusing visual features on various tasks. Right: The magnitude of learnt adaptive weights for visual features across different ViT layers.

	7b		13b	
METHOD	box	mask	box	mask
Numerical	3.62	42.1	4.30	57.1
Vocab	0.26	4.01	0.45	6.16
Token Embed	0.15	0.18	0.21	0.23

Table 7. The inference time taken to decode a box or a mask with different representations. PerceptionGPT represents perception signal in a single dynamic token embedding, greatly boosting the inference efficiency.

are key for tasks like image captioning due to their high-level semantic content, while lower layer features are better for detailed visual tasks. Our adaptive fusion strategy dynamically adjusts weights for each layer's features. The figure's right side displays the learned weight distribution, highlighting each layer's feature contributions

Inference Speed Comparison We compare the inference speed between different formulations of visual perception signals in Table 7. The number of discrete coordinates to represent a mask contour is set to 32, which ensures acceptable mask quality. Since PerceptionGPT requires only one dynamic token embedding to carry the perception information, the inference speed can be greatly boosted, especially for complex signals such as segmentation masks. Specifically, for a 7B P-VLM to decode a mask, PerceptionGPT takes only 0.3% and 3.7% inference time of Numerical and Vocabulary formulations, respectively.

5. Conclusion

In this paper, we propose **PerceptionGPT**, a novel framework for perception-enhanced vision language models (P-VLMs). Our approach addresses the limitations of existing methods by taking advantage of the representation power of the LLM's dynamic token embeddings. Our PerceptionGPT achieves promising results by tuning only a small fraction of parameters, resulting in compact perception representations and significantly accelerated inference. We hope this work provides new insights into future research of P-VLMs.

References

[1] Jinze Bai, Rui Men, Hao Yang, Xuancheng Ren, Kai Dang, Yichang Zhang, Xiaohuan Zhou, Peng Wang, Sinan Tan, An Yang, Zeyu Cui, Yu Han, Shuai Bai, Wenbin Ge, Jianxin Ma, Junyang Lin, Jingren Zhou, and Chang Zhou. Ofasys: A multi-modal multi-task learning system for building generalist models, 2022. 4

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond, 2023. 1, 3, 4

[3] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022. 1, 3

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1, 3

[5] Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic, 2023. 1, 2, 3, 4, 5, 7

[6] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, 2023. 1, 4, 5

[7] Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. Unifying vision-and-language tasks via text generation. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, pages 1931–1942. PMLR, 2021. 1

[8] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022. 1, 3

[9] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023. 3

[10] Henghui Ding, Chang Liu, Suchen Wang, and Xudong Jiang. Vision-language transformer and query generation for referring segmentation, 2021. 7

[11] Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, Hongsheng Li, and Yu Qiao. Llama-adapter v2: Parameter-efficient visual instruction model, 2023. 3

[12] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering, 2017. 8

[13] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training, 2022. 3

[14] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022. 3

[15] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr – modulated detection for end-to-end multi-modal understanding, 2021. 4

[16] Amita Kamath, Christopher Clark, Tanmay Gupta, Eric Kolve, Derek Hoiem, and Aniruddha Kembhavi. Webly supervised concept expansion for general purpose vision models, 2022. 4

[17] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023. 3, 5

[18] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. Visual genome: Connecting language and vision using crowdsourced dense image annotations, 2016. 5

[19] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. *arXiv preprint arXiv:2308.00692*, 2023. 1, 2, 3, 5, 7

[20] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023. 3

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015. 5, 8

[22] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017. 4

[23] Chang Liu, Henghui Ding, and Xudong Jiang. Gres: Generalized referring expression segmentation, 2023. 7

[24] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. 1, 3, 5

[25] 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, 2023. 4

[26] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks, 2022. 4

[27] Gen Luo, Yiyi Zhou, Xiaoshuai Sun, Liujuan Cao, Chenglin Wu, Cheng Deng, and Rongrong Ji. Multi-task collaborative network for joint referring expression comprehension and segmentation, 2020. 7

[28] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions, 2016. 5

[29] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge, 2019. 8

[30] OpenAI. Gpt-4 technical report, 2023. 1, 3

[31] 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*, 35: 27730–27744, 2022. 1, 3

[32] 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, 2023. 1, 2, 3, 4, 7

[33] Renjie Pi, Jiahui Gao, Shizhe Diao, Rui Pan, Hanze Dong, Jipeng Zhang, Lewei Yao, Jianhua Han, Hang Xu, Lingpeng Kong, and Tong Zhang. Detgpt: Detect what you need via reasoning, 2023. 1, 3

[34] Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models, 2016. 5, 8

[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya 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, 2021. 4

[36] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression, 2019. 2, 5

[37] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilic, Daniel Hesslow, Roman Castagne, Alexandra Sasha Luccioni, François Yvon, Matthias Galle, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022. 1, 3

[38] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022. 1, 3

[39] Yixuan Su, Tian Lan, Huayang Li, Jialu Xu, Yan Wang, and Deng Cai. Pandagpt: One model to instruction-follow them all, 2023. 1, 3

[40] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. 2, 5

[41] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning, 2023. 1, 3

[42] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothee Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1, 3

[43] Peng Wang, An Yang, Rui Men, Junyang Lin, Shuai Bai, Zhikang Li, Jianxin Ma, Chang Zhou, Jingren Zhou, and Hongxia Yang. Ofa: Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework, 2022. 2, 4

[44] Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, and Jifeng Dai. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks, 2023. 1, 3, 4, 7

[45] Zhaoqing Wang, Yu Lu, Qiang Li, Xunqiang Tao, Yandong Guo, Mingming Gong, and Tongliang Liu. Cris: Clip-driven referring image segmentation, 2022. 7

[46] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models, 2023. 3

[47] Zhao Yang, Jiaqi Wang, Yansong Tang, Kai Chen, Hengshuang Zhao, and Philip H. S. Torr. Lavt: Language-aware vision transformer for referring image segmentation, 2022. 7

[48] Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action, 2023. 1, 3

[49] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions, 2016. 5

[50] Shilong Zhang, Peize Sun, Shoufa Chen, Min Xiao, Wenqi Shao, Wenwei Zhang, Yu Liu, Kai Chen, and Ping Luo. Gpt4roi: Instruction tuning large language model on region-of-interest, 2023. 3

[51] Chaoyang Zhu, Yiyi Zhou, Yunhang Shen, Gen Luo, Xingjia Pan, Mingbao Lin, Chao Chen, Liujuan Cao, Xiaoshuai Sun, and Rongrong Ji. Seqtr: A simple yet universal network for visual grounding. In *European Conference on Computer Vision*, pages 598–615. Springer, 2022. 2, 4

[52] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models, 2023. 1, 3, 5

[53] Xueyan Zou, Zi-Yi Dou, Jianwei Yang, Zhe Gan, Linjie Li, Chunyuan Li, Xiyang Dai, Harkirat Behl, Jianfeng Wang, Lu Yuan, Nanyun Peng, Lijuan Wang, Yong Jae Lee, and Jianfeng Gao. Generalized decoding for pixel, image, and language, 2022. 7

[54] Xueyan Zou, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Wang, Lijuan Wang, Jianfeng Gao, and Yong Jae Lee. Segment everything everywhere all at once, 2023. 7