Pixel Aligned Language Models

Jiarui Xu[1,2*]   Xingyi Zhou[1]   Shen Yan[1]   Xiuye Gu[1]

Anurag Arnab[1]   Chen Sun[1]   Xiaolong Wang[2]   Cordelia Schmid[1]

[1]Google Research   [2]UC San Diego

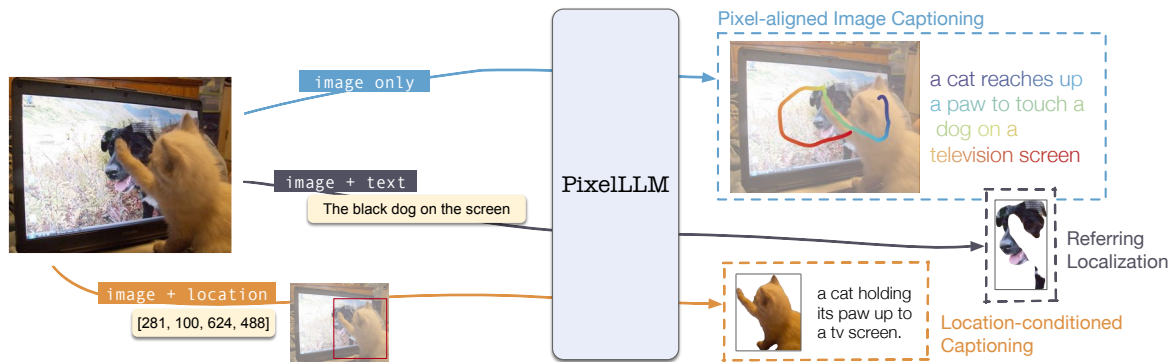https://jerryxu.net/PixelLLM/

Figure 1. We propose Pixel-Aligned Language Model (PixelLLM) to equip large language models with localization capability. The model is pre-trained on localized image captioning data [38], where each word is labeled with a pixel location, to learn the alignment between words and image pixels. PixelLLM can be applied to various localization tasks, for example, *location-conditioned captioning* when taking location as input, and *referring localization* when generating locations as outputs.

Abstract

Large language models have achieved great success in recent years, so as their variants in vision. Existing vision-language models can describe images in natural languages, answer visual-related questions, or perform complex reasoning about the image. However, it is yet unclear how localization tasks, such as word grounding or referring localization, can be performed using large language models. In this work, we aim to develop a vision-language model that can take locations, for example, a set of points or boxes, as either inputs or outputs. When taking locations as inputs, the model performs location-conditioned captioning, which generates captions for the indicated object or region. When generating locations as outputs, our model regresses pixel coordinates for each output word generated by the language model, and thus performs dense word grounding. Our model is pre-trained on the Localized Narrative dataset, which contains pixel-word-aligned captioning from human attention. We show our model can be applied to various location-aware vision-language tasks, including referring localization, location-conditioned captioning, and dense object captioning, archiving state-of-the-art performance on RefCOCO and Visual Genome.

*Work done during a Google internship. ✉{jiaruixu, zhouxy}@google.com

1. Introduction

Imagine a baby waving their hands, pointing to the colorful toys, and yelling the toy names. Pointing and naming is a natural and convenient way to describe the visual world, and provide dense visual-language alignment that is synchronized over time. Can we design an intelligent model [15, 35, 40, 42] that leverages such information to obtain vision-language alignment? Prior research [2, 10, 14, 25, 26, 32, 44, 57] has tried to align visual information to the pre-trained LLMs for various vision-language tasks such as producing long and detailed descriptions [32] or conversation [26]. However, most of these works take the entire image as input, and produce all outputs in text. Detailed understanding of specific regions and objects, and their exact locations, is not well-studied in the context of LLMs. We aim to propose a new architecture and a training procedure that can achieve the goal, which helps us answer the important research question: can large *language* models achieve spatial understanding and reasoning from the *visual* world? If so, how?

To this end, we introduce PixelLLM, a vision-language model with fine-grained localization ability by densely aligning each output word to a pixel location. We realize this by adding a small MLP on top of the word features of a

language model, to regress to a pixel location of each word. The weights of the language model can be kept frozen, or updated by Low-rank finetuning (LoRA) [18]. Furthermore, our model can take the location prompt or text prompt, to generate outputs specific to the prompt. Fig. 1 shows the tasks that our model can do or can adapt to.

We show our concrete architecture in Fig. 2. It includes an image encoder, a prompt encoder, and a prompt feature extractor which maps prompt-conditioned image features to the text embedding space. These prompt-conditioned image features and an optional text prompt, are then directly fed as the prefix of a large-language model, which produces the captioning and per-word localization outputs. We show this architecture is general and can adapt to various vision-language tasks, with any combination of language or location as input or output.

While our model requires dedicated word-pixel aligned data to train, we note that such annotations already exist at large scale [38]. The Localized Narratives dataset [38] contains annotations of human annotators narrating a given image, together with a mouse trajectory of the annotators' attention during the narration. This gives synchronized locations for all words in the narration sentence, which can be used to train our model. While not all word-location pairs are visually meaningful or accurate, we argue that they are valuable as they are from actual human attention.

We evaluate our model on popular vision tasks by adapting our architecture and fine-tuning on downstream datasets, including referring localization on RefCOCO [53], location conditioned captioning on RefCOCO and Visual Genome [23], and dense object captioning on Visual Genome [23]. Our model achieves the state-of-the-art performance on all these tasks, with 89.8 P@0.5 on RefCOCO referring localization, 19.9 CIDEr on Visual Genome conditioned captioning, and 17.0 mAP on dense object captioning. Ablations on RefCOCO [53](Tab. 4) show our dense per-pixel localization formulation is the key to the high performance, with a 3.7 point gain compared to alternative localization formulations.

Our contributions are summarized below:

- We introduce PixelLLM, a vision-language model that outputs a caption for an input image together with the localization of each word. PixelLLM takes as input an image, and optionally a location prompt or a text prompt.
- Our model can utilize the dedicated localized narrative dataset [38], which comes with image captions and trajectories localizing each word, for per-word localization.
- Our model is flexible and can be adapted to various vision-language tasks, including referring localization and segmentation, location-conditioned captioning, and dense captioning, with state-of-the-art performance.

Code and models will be released.

2. Related work

Large-scale vision and language models. Building large vision and language models that are generally capable of visual captioning or question answering is a trending research topic [7, 10, 19, 39, 44, 46, 54, 57]. Flamingo [1] inserts visual features into language models through a gating function. BLIP2 [26] connects vision features [13] to frozen large-language models [40] using learnable queries [3]. LLaVa [32] distills large language models to visual inputs by creating instructional image-text pairs. The majority of these works focus on language-only outputs. While language is general enough for image-level tasks, region-level localization tasks are more challenging. Our work builds on existing vision-language models [26, 44], and adds localization ability using a per-token regression head.

Modeling locations in language models. We are not the first to study localization in the context of large language models [4, 24, 47, 52, 55]. Pix2Seq [5] first proposes to model localization as spatial bin tokens in the vocabulary, and tackle object detection as an auto-regressive problem. Following this idea, UniTab [50] embeds bounding boxes bin tokens in captions, which enables predicting object locations when seeing nouns in the output. Kosmos [36] uses a similar representation, and creates interleaved location-caption datasets by distilling grounding models [27]. A key distinction for our work is the localization representation. While existing works try to model locations as a word in the language model vocabulary, we use a regression representation. Also, most existing works [24, 32, 36] rely on distilled data from existing models [32, 48], and our representation enables us to utilize the existing human annotation from Localized Narratives dataset [38].

Referring expression localization is a popular task in language and vision, aiming to localize a sentence query [33, 53]. MDETR [21] formulates this with the cross-attention mechanism in DETR [3] style, where the language features are the queries and the visual features are keys. UniTab [50] integrates box output as the follow-up spatial tokens in a language decoder. Shikra [4] directly extracts formatted bounding boxes from the raw string generated by the large language model. We tackle this task by regression given the language and vision features using a large-language model.

Attention trajectory modeling is a new task enabled by the localized narratives dataset [38]. It aims to align a long and detailed image caption to a trajectory of human attention, collected from the mouse trace while annotating the caption. The only existing work that evaluates on this task is MITR [34], which trains linear layers on frozen region features. While this is effective, the knowledge learned in MITR couldn't transfer to other tasks given the task-specific design. To the best of our knowledge, we are the first to propose an end-to-end training framework for this task and train it at scale.
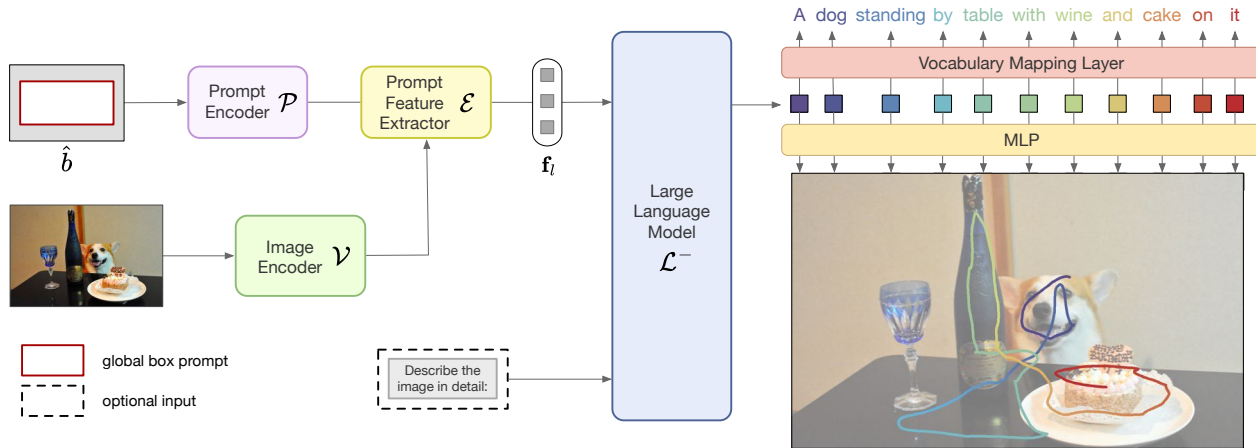
Figure 2. **Overview of PixelLLM architecture for pixel-aligned captioning.** We first encode the input location prompt (global box prompt in this case) and the input image with the prompt encoder $\mathcal{P}$ and the image encoder $\mathcal{V}$ respectively. Then we input the prompt feature l and the image feature f into the prompt feature extractor to extract location-specific visual feature f_l. The large language model $\mathcal{L}$ then auto-regressively predicts the next text tokens conditioned on previous text tokens and the visual feature. We apply a simple MLP layer on the token features before the vocabulary mapping layer of LLM, which predicts the coordinates of each text token. The alignment between the caption and the trace is represented by color gradient.

3. Preliminary

Given an input image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, our first goal is to generate an image caption $\mathbf{s}$, a sequence of word tokens[1] : $\mathbf{s} = [w_1, w_2, \cdots, w_n]$. Each token $w_i \in [0, |V|]$ is an integer indexing a vocabulary V. This task is known as image captioning [6], and is widely studied in vision and language research [7, 26, 44].

Optionally, we hope to take additional prompt inputs, to specify the region or concepts that the model should focus on. The prompts can be locations, indicated by a sequence of points $\mathbf{x} \in \mathbb{R}^{m \times 2}$ or a box $b \in \mathbb{R}^4$, or texts (in the form of a sequence of word tokens $\mathbf{t} = [w_1, \cdots]$). Before introducing our proposed approach, we review existing techniques for standard image captioning and input prompt encoding.

Image captioning produces a sentence $\mathbf{s}$ given an input image $\mathbf{I}$. Popular architectures [7, 26, 44] first encode the image as a feature $\mathbf{f} \in \mathbb{R}^{N \times C}$ using an image encoder $\mathcal{V}$ (*e.g.*, a ViT [12]), where N is the number of tokens and C is the dimension size. They then feed $\mathbf{f}$ to an auto-regressive [16, 43] language model $\mathcal{L}$. The language model produces the word sequence one-by-one, conditioning on the vision feature and previously predicted words: $w_i = \mathcal{L}(\mathbf{f}, \mathbf{w}_{1:i-1})$. The architecture of the language model can be a decoder-only architecture with a stack of self-attention layers [44], or a pair of encoder-decoder with cross-attention [40].

Prompt encoder. In addition to an image input, prompt encoders adapt other input modalities to the same feature space as the image feature $\mathbf{f}$. We follow SAM [22] to encode points or box coordinates using a location prompt encoder $\mathcal{P}$ consisting of a sine-cosine position embedding followed by linear layers. For text prompt, we follow GIT [44] and BLIP2 [26] to concatenate query word embedding with visual features as the prefix features for the language model.

With these components, one can already compose a prompt-conditioned image captioning model. However, it is not yet feasible to generate location outputs, *e.g.*, localizing the queried concept or grounding the answer to a region. Next, we introduce our Pixel-Aligned Language Model that seamlessly integrates per-token localization ability into vision-language models.

4. Pixel-Aligned Language Model

Localization has been studied in multiple forms in computer vision. For instance, localizing all instances given a set of vocabularies (object detection [30]), localizing a natural language query (referring localization [53]), or associating each entity in a sentence to a bounding box (entity grounding [37]).

Under the framework of vision-language models, we propose to formulate localization as *aligning each word in the output sentence to a pixel location*. Specifically, besides the sentence output $\mathbf{s}$, we also output a sequence of points $\mathbf{p}$ with the same length as the sentence, $\mathbf{p} = [p_1, p_2, \cdots, p_n]$, $p_i \in \mathbb{R}^2$, each corresponds to the respective word token in the sentence. In contrast to existing works [27, 50] that only nouns may be grounded to a region, we don't enforce the model to ignore non-visual tokens, so that the model can also learn relation terms, *e.g.* "holding".

Next, we first introduce novel components of our architecture (Sec. 4.1). We then describe how we train it on large-scale densely annotated datasets (Sec. 4.2). Finally,

---

[1]Technically, a word can be multiple tokens. For presentation clarity, we assume a word is a token. We use "word" and "token" interchangeably.

we show our architecture can be applied to various localization tasks with no or minor modifications (Sec. 4.3).

4.1. Architecture

Fig. 2 gives an overview of our architecture. The input is an image $\mathbf{I}$ and an optional location prompt b. If no location prompt is provided, we use a global box prompt of the entire image, *i.e.* $b = (0, 0, H, W)$, where H and W are the image size. Otherwise, the model is expected to focus on the locations indicated by the prompt. The output is the sentence $\mathbf{s}$ and its aligned point trajectory $\mathbf{p}$.

Using the image encoder $\mathcal{V}$ and location prompt encoder $\mathcal{P}$ introduced above, we obtain the image feature $\mathbf{f} = \mathcal{V}(\mathbf{I})$ and location prompt feature $\mathcal{P}(b)$. Here $\mathbf{f}$ is the feature of the entire image, we use an prompt feature extractor $\mathcal{E}$ to extract the feature specified by the location prompt:

$$\mathbf{f_l} = \mathcal{E}(\mathbf{f}, \mathcal{P}(b)) \tag{1}$$

where $\mathbf{f}_l \in \mathbb{R}^{N \times C}$ is the location-specific visual feature.

We instantiate the prompt feature extractor $\mathcal{E}$ using a two-way transformer with a set of learnable tokens $\mathbf{q}$, inspired by QFormer [26]. Specifically, the two-way transformer takes $[\mathcal{P}(b), \mathbf{q}]$ and $\mathbf{f}$ as the query or key/value alternatively at each layer, and finally takes the learnable token features at the last layer. The output feature $\mathbf{f}_l$ conveys the feature specific to the location prompt b. Conceptually, prompt feature extractor has a similar function as ROIAlign [17], but it is learnable and doesn't require feature interpolation and sampling. We compare and discuss the differences in Sec. 5.3.2.

Dense location outputs from language models. Given the location-specific feature $\mathbf{f}_l$, we can already feed it to a language model to do captioning, using auto-regressive decoding: $w_i = \mathcal{L}(\mathbf{f}_l, \mathbf{w}_{1:i-1})$. We note the last linear layer of a language model is a vocabulary mapping layer, mapping from the language feature space to the one-hot vocabulary index. Let $\mathcal{L}^-$ denote the language model without the last vocabulary mapping layer, the decoding process (for simplicity, we show greedy decoding here) can be rewritten as

$$w_i = \text{argmax}(\mathbf{v} \cdot \mathcal{L}^-(\mathbf{f}_l, \mathbf{w}_{1:i-1})) \tag{2}$$

where $\mathbf{v} \in \mathbb{R}^{|V| \times C}$ is the weight of the linear vocabulary mapping layer.

To use the same language features for localization, we simply add a small MLP in parallel with the vocabulary mapping layer, which maps the language feature to a 2-dimension location output[2]:

$$p_i = \text{MLP}(\mathcal{L}^-(\mathbf{f}_l, \mathbf{w}_{1:i-1})) \tag{3}$$

---

[2]There are multiple trace points per word token, we show the single point prediction for simplicity. In practice, we output the two points that tightly bound the trace of each token.

Note we do not feed back the localization output for auto-regressive decoding, to avoid affecting the original text decoding process. The location prediction runs on the fly together with language decoding, only adding minor computation overhead. This design is language model agnostic, and can be applied to any language model without interfering with the original language generation ability. To take text prompt as input, we directly concatenate text prompt word embeddings with the visual feature $\mathbf{f_l}$.

4.2. Training

We train our model using a human-annotated caption-location aligned dataset Localized Narrative [38] (LN). Localized Narrative asks annotators to narrate a given image while simultaneously moving the mouse over the region they are describing. The narration and the mouse trace are synchronized, which gives the location of each single word of the narration. Although the mouse trace could be noisy, it is still a cheap and effective way to obtain dense location supervision. Thus, this dataset contains all the needed triple annotations $(\mathbf{I}, \mathbf{s}, \mathbf{p})$: the image $\mathbf{I}$, the captioning sentence $\mathbf{s}$, and the location trajectory $\mathbf{p}$.

We use the standard label-smoothed cross-entropy loss to train the captioning output, and use an L_1 regression loss to train the localization output:

$$L = \frac{1}{n} \sum_{i=1}^{n} (\text{CE}(\mathcal{L}(\mathbf{f}_l, \mathbf{w}_{1:i-1}), w_i) + \lambda|\hat{p}_i - p_i|) \tag{4}$$

where $\hat{p}_i$ is the predicted location for the i-th word, λ is the localization loss weight, and n is the caption length.

4.3. Adapting to downstream vision tasks.

Our architecture can take any combination of text/ location as inputs or outputs, and thus can be applied to various location-related vision tasks. In this section, we show examples of adapting our architecture for three popular tasks: referring localization and segmentation, location-conditioned captioning, and dense object captioning.

Referring localization and segmentation takes an image $\mathbf{I}$ and a sentence query $\mathbf{t}$ as input, and aims to produce a bounding box $\hat{b} \in \mathbb{R}^4$ that corresponds to the query. To apply our framework to this task, we set the location prompt as a global box (i.e., $\bar{b} = (0, 0, W, H)$), and use the query sentence as the conditioned sentence in Eq. 3. By default, our model outputs a trajectory, rather than a single bounding box. While one can form a bounding box by taking the boundaries of the trajectory, we observe this is suboptimal, as the trajectory boundaries are not tight object boundaries required in evaluation. We thus train the model to output an accurate object bounding box at the <EOS> token, using the same regression MLP layers. We take the single box output
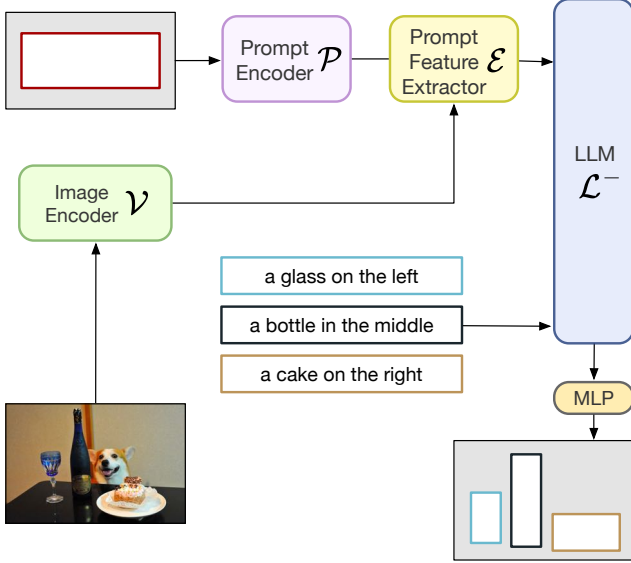
Figure 3. **Our model for referring expression localization pipeline.** To apply PixelLLM, we don't need to generate the text tokens. Instead, we directly input the query $\mathbf{t}$ into the LLM $\mathcal{L}^-$ to extract the token features before the vocabulary mapping layer. We then apply MLP to the last token predict the bounding boxes.

at the `<EOS>` token as the output:

$$\hat{b} = \text{MLP}\Big(\mathcal{L}^-(\mathcal{E}(\mathcal{V}(\mathbf{I}), \mathcal{P}(\bar{b})), [\mathbf{t}, \texttt{<EOS>}])\Big) \quad (5)$$

As our model already include the image backbone and prompt encoder from SAM [22], we can further obtain the segmentation mask by simply plugging in the mask decoder of SAM. Our model can thus also used for referring segmentation, by producing a mask on top of the predicted box $\hat{b}$.

Location-conditioned captioning takes an image $\mathbf{I}$ and a bounding box b as a location prompt, and produces a caption sentence $\mathbf{s}^b$ that corresponds to the indicated object from the box query. Our model can be directly applied to this task, using our prompt encoder and auto-regressive language model, while ignoring the per-word location output.

$$\mathbf{s}_i^b = \mathcal{L}(\mathcal{E}(\mathcal{V}(\mathbf{I}), \mathcal{P}(b)), \mathbf{s}_{1:i-1}^b) \quad (6)$$

Dense object captioning aims to first detect all objects in a given image, and then caption them. Our framework does not by default detect objects. To get bounding box candidates, we add a proposal head after our image encoder. We then feed the resulting bounding boxes separately to the location prompt, to perform location-conditioned captioning for each. Specifically, we use a Simple Feature Pyramid [29] to upsample the visual features $\mathbf{f}$ to a pyramid of features, and use a CenterNet [56] head for detection. We fine-tune the model with the detection losses and the caption losses together end-to-end.
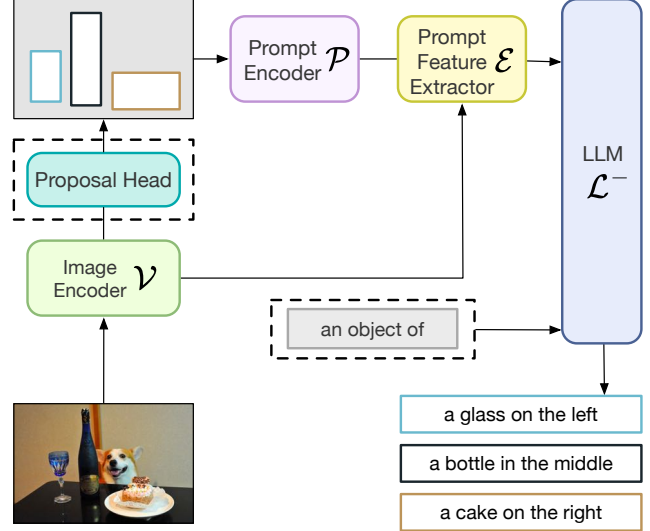

Figure 4. **Our model for location-conditioned captioning and dense object captioning.** For location-conditioned captioning, the input bounding boxes are provided. For dense object captioning, we first apply a proposal head on the image feature to generate the bounding boxes. We input bounding boxes and image features into the prompt encoder and prompt feature extractor to extract the location-specific feature for each bounding box. The language model auto-regressively predicts the caption of each object.

5. Experiments

We first introduce our implementation details. Then we compare our results against the state of the art on various downstream vision tasks. Lastly, we analyze the effectiveness of our proposed components.

5.1. Implementation details

Architecture. Our visual encoder $\mathcal{V}$ consists of two parallel backbones: ViT-H initialized with SAM [22] and ViT-L initialized with EVA02 [13]. The reason for using two backbones is twofold. First, using the SAM backbone and keeping it frozen enables us to inherit the strong segmentation ability from SAM, and provide strong localization features for our model. Second, while keeping the SAM backbone frozen, we use another tunable backbone to learn semantic features. We concatenate the two features on channel dimension when feeding to the prompt feature extractor.

For the language model $\mathcal{L}$, we use the instruction fine-tuned T5-XL [9, 40] following BLIP2 [26]. We apply low-rank adaption (LoRA) [18] of rank 32 on query and value projection layers of self-attention blocks to adapt the T5-XL to the vision tasks. The other parameters of T5-XL are frozen during training. The prompt feature extractor is a 2-layer transformer with 32 learnable tokens. Following PrefixLM paradigm [40], we concatenate the prompt feature extractor outputs with the text embedding of the prefix text, and feed it to the language model $\mathcal{L}$. To generate the
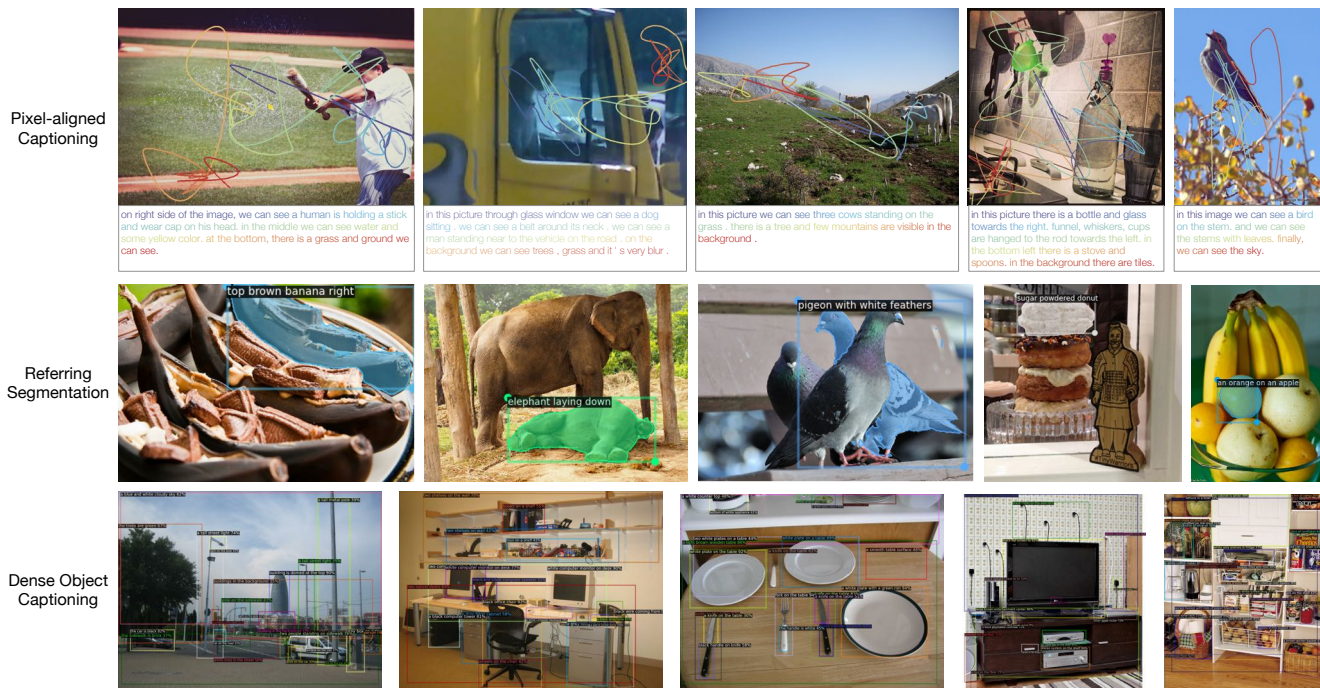
Figure 5. Qualitative results on pixel-aligned captioning (row 1), referring segmentation (row 2), and dense object captioning (row 3). The generated trace semantically corresponds to the caption, represented by color gradient ▮▮▮. In referring segmentation, our model correctly understands the descriptive referring expressions, *e.g.* "sugar powdered", "with white feathers". For dense object captioning, our model could generate the region-level caption that captures the spatial relationship, *e.g.* "shelves on the wall". Zoom in for the best view.

segmentation mask, we directly employ the mask decoder of SAM [22] on the prompt embedding of the predicted bounding box. We also include a learnable embedding to enable finetuning the mask on downstream datasets.

Training details. We first pretrain our model on We-bLI dataset[7] with the image captioning objective only, to initialize the prompt feature extractor weights following BLIP2 [26]. During this pretraining, we follow BLIP2 (T5-XL) [26] to randomly split the image caption into the prefix and the suffix We concatenate the prefix text with visual features and input them into the encoder of T5-XL. The suffix text is used as the generation target for the T5-XL decoder. We run the pretraining for 10 epochs, using a standard [26] captioning input size 224×224.

We then train on the Localized Narrative [38] dataset with the joint captioning and localization objectives in Eq. 4. The default caption in LN contains multiple sentences covering multiple objects in the image. We consider each sentence (split by period) as one training target, rather than concatenating all sentences together. We train for 5 epochs using an input size of 384×384. The parameters of the prompt encoder, SAM, mask decoder, and T5-XL are frozen during all training stages. In all settings, we set the localization weight λ to 0.1. We use a label-smooth factor of 0.1 for our captioning loss. We provide further details on datasets in the supplement.

5.2. Joint Captioning and trace generation

We first evaluate our model trained on the Localized Narrative [38] dataset, on its validation split. We first show qualitative results on joint captioning and trace generation on the top row of Fig. 5. We can see that the generated trace semantically corresponds well to the generated caption. We next quantitatively evaluate our model under the controlled trace generation setting on the COCO subset of Localized Narrative and compare to MITR [34], using their Local Bipartite Matching score (LBM) as the evaluation metric (lower better). We refer to the MITR paper for details about the metric. PixelLLM achieves 0.153 LBM (k=0), outperforms 0.163 of MITR [34]. This confirms our model's ability of trace generation task compared to existing works. We also highlight that our trace outputs are produced from language model features, rather than visual features as in MITR [34].

5.3. SOTA Comparison on Downstream Tasks

We next fine-tune our model pretrained on Localized Narratives [38] above on downstream tasks and compare with the state-of-the-art models.

5.3.1 Referring Localization and Segmentation

As introduced in Sec. 4.3, referring localization aims to produce a bounding box or a segmentation mask given

Models	RefCOCO			RefCOCO+			RefCOCOg	
	val	testA	testB	val	testA	testB	val	test
bounding box P@0.5								
TransVG[11]	81.0	82.7	78.4	64.8	70.7	56.9	68.7	67.7
UNITER[8]	81.4	87.0	74.2	75.9	81.5	66.7	74.0	68.7
OFA[45]	80.0	83.7	76.4	68.3	76.0	61.8	67.6	67.6
UniTAB[50]	86.3	88.8	80.6	78.7	83.2	69.5	80.0	80.0
MDETR[21]	86.8	89.6	81.4	79.5	84.1	70.6	81.6	80.9
VisionLLM[47]	86.7		-	-	-	-	-	-
Shikra-7B[4]	87.0	90.6	80.2	81.6	**87.4**	71.1	82.3	82.2
Ferret-7B[52]	87.5	91.4	82.5	80.8	**87.4**	73.1	83.9	84.8
PixelLLM (Ours)	**89.8**	**92.2**	**86.4**	**83.2**	87.0	**78.9**	**84.6**	**86.0**
segmentation mask cIoU								
ReLA[31]	73.8	76.5	70.2	66.0	71.0	57.7	65.0	66.0
X-Decoder[58]	-	-	-	-	-	-	64.6	-
SEEM[59]	-	-	-	-	-	-	65.7	-
LISA[24]	74.9	**79.1**	72.3	65.1	70.8	58.1	67.9	70.6
PixelLLM (Ours)	**76.9**	78.5	**74.4**	**69.2**	**72.1**	**64.5**	**70.7**	**72.4**

Table 1. **State-of-the-art comparison on referring localization and segmentation on RefCOCO.** We report the official evaluation metrics: precision at IoU threshold 0.5 for referring (box) localization, and mask cIoU for referring segmentation. Numbers of other methods are taken from the original publications. Our model achieves state-of-the-art on all settings for both box and mask outputs.

a query text. Following common practice in RefCOCO datasets [21], we use the combination of RefCOCOs [30, 33, 53] and GoldG [21] as the training set for referring expression localization and segmentation. Note each bounding box (segmentation mask) is annotated with several referring expressions, and we train on all of them using a regression loss on our bounding box prediction (Eq. 2).

We report the official evaluation metrics, bounding box P@0.5 (precision at 0.5 IoU threshold), and segmentation cIoU (cumulative IoU). Tab. 1 shows the results on validation and test split of RefCOCO[30], RefCOCO+[53] and RefCOCOg[33]. Our PixelLLM outperforms previous and concurrent works [4, 21, 24, 47, 50, 52, 58, 59] on almost all splits of three datasets: 2.3 P@0.5 and 1.4 cIoU on RefCOCO val, 1.7 P@0.5 and 1.5 cIoU on RefCOCO+ val, versus the second best. We compare to 7B models of [4, 52], which are in comparable size with ours (4B). While this is a system-level comparison, as models use different backbones and pretraining, it justifies our model's capability generalizes to this tasks, as our model is not designed specifically for referring.

5.3.2 Dense Object Captioning

Dense object captioning aims to detect and caption objects in an image. As introduced in Sec. 4.3, our model adds a proposal head on top of our visual features. The proposal head is randomly initialized and finetuned for this data. We train and evaluate dense object captioning on Visual Genome [23] dataset. We report the official evaluation metric, mAP in Tab. 2. In this setting, we use the same proposal network [56] and language model [44] with

Model	mAP
FCLN [20]	5.39
JIVC [49]	9.31
ImgG [28]	9.25
COCD [28]	9.36
COCG [28]	9.82
CAG-Net [51]	10.51
TDC+ROCSU [41]	11.49
GRiT [48]	15.48
PixelLLM (Ours)	**17.02**

Table 2. **State-of-the-art comparison on dense object captioning on Visual Genome** [23]. We report the official evaluation metric mAP. Our model performs the best among all methods.

GRiT [48]. A key difference between our architecture and GRiT [48] is the region feature extractor, where GRiT uses ROIAlign [17], to explicitly extract image features within the bounding boxes. PixelLLM employs a prompt encoder and prompt feature extractor which don't require the image feature sampling and interpolation like ROIAlign. Our PixelLLM outperforms GRiT by 1.5 mAP, this again shows the generalization ability of our architecture and pertaining as well as the effectiveness of our prompt feature extractor.

5.3.3 Location Conditioned Captioning

Finally, we evaluate location-conditioned captioning on RefCOCOg[33] and Visual Genome[23]. The input is a ground truth bounding box, and the output is the caption corresponding to the indicated region. The evaluation metrics are per-box METEOR and CIDEr, similar to standard image captioning [6]. We reuse our dense object captioning

| | RefCOCOg | | Visual Genome | |
Model	METEOR	CIDEr	METEOR	CIDEr
GRiT [48]	13.8	75.8	17.1	142.0
Kosmos-2 [36]	14.1	62.3	-	-
GPT4RoI [55]	-	-	17.4	145.2
PixelLLM	**14.3**	**82.3**	**19.9**	**148.9**

Table 3. **State-of-the-art comparison on Location-conditioned Captioning on RefCOCOg [33] and Visual Genome [23].** We report averaged per-object METEOR and CIDEr on each dataset. Our model outperforms both our GRiT [48] baseline and recent state-of-the-art models.

Localization Method	Pretrain data	RefCOCO P@0.5
Raw string ([4, 52])	GoldG	84.4
Extra token ([36, 50])	GoldG	86.1
Regress. on token features	GoldG	87.6
Regress. on token features (ours)	LN+GoldG	89.8

Table 4. **Ablation on Localization Supervision** We compare to alternative localization methods including using raw string [4, 52] and using extra tokens [36, 50]. Both alternatives could only use the sparsely labeled data GoldG [21], and not the densely labeled data Localized Narratives (LN) [38]. Our regression formulation outperforms the alternatives, and can benefit more by using the dense supervision [38].

model in Sec. 5.3.2 and drop the proposal head.

We compare to Kosmos-2[36] and GPT4RoI [55], who reported results on this task. We also report results of GRiT [48] evaluated by ourselves. Kosmos-2[36] encodes the bounding box coordinates in the text prefix and generates the caption conditioned on it. while our PixelLLM explicitly encodes the bounding box coordinates using our prompt encoder. Tab. 3 shows the results. Our PixelLLM outperforms previous works, where the improvements can come from our location formulation as well as the localized narrative pretraining.

5.4. Ablation study

Importance of per-token localization. Next, we ablate the importance of our per-token regression formulation for localization in LLMs. We compare two existing alternatives: (1) encode the bounding box coordinates as raw strings, and directly use the original LLM output head [4, 52]. (2) discretize the bounding box coordinates into bins following Pix2Seq [5], and encode the bin number as extra tokens in LLM's vocabulary [36, 50]. We note both these two ideas are used to sparsely encode locations, i.e., encode locations for nouns. Otherwise, if we apply these ideas densely for all words, the output sequence length would be $5\times$ longer, which is not easily feasible to train well. Our regression formulation allows us to train both sparse localization data [21] (supervise selected words) or dense data [38] (supervise location outputs for all words).

Tab. 4 shows the results. First, with all sparse supervi-

Language Model	Params	w/ LoRA	RefCOCO P@0.5
T5 Small	80M	✗	67.0
T5 Base	250M	✗	70.3
T5 Large	780M	✗	73.6
T5 XL	3B	✗	81.9
T5 Small	80M	✓	75.6
T5 Base	250M	✓	80.8
T5 Large	780M	✓	84.8
T5 XL	3B	✓	89.8

Table 5. **Ablation on language model size and LoRA.** We report RefCOCO official metrics under different language model sizes. We observe the localization performance increases consistently with language model size, and using LoRA finetuning is better than freezing the language models.

sion, our formulation (3rd row) slightly improves the two alternatives (+1.5 points). The improvement could be from a better localization decoding process: the alternatives mix decoding regular words and localizations, which the two tasks can interfere with each other. In our model localization is decoded in parallel. Second, our formulation enables us to utilize the densely annotated Localized Narratives [38] as additional training data (4th row). This further improves our performance by 2.3 points. This convincingly shows that our dense word-pixel alignment training objective is beneficial for referring localization tasks.

Model size and LoRA fine-tuning. We ablate the language model size and LoRA fine-tuning. We report the P@0.5 on RefCOCO validation set in Tab. 5. As we increase the size of the language model, the accuracy on RefCOCO improves consistently both with and without LoRA fine-tuning. This is expected, as larger models encapsulate more knowledge, which yields stronger understanding and adapting ability. We note that the T5 models with LoRA [18] outperform the without LoRA counterparts by 8-10 points. This indicates that LoRA could help the frozen language model adapt to localization tasks better. It is evidence that the frozen large language model like T5 encompasses strong localization ability, which could be revealed by our PixelLLM.

6. Conclusion

We proposed a vision-language model, PixelLLM, which can take an image and any combination of location or text as input or output. PixelLLM generates captions, and aligns each output word to a pixel location. Our model achieved state-of-the-art performance on referring localization, dense captioning, and conditioned captioning. We hope our work can inspire follow-up works on connecting vision and language, and building general intelligent vision systems.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *NeurIPS*, 35, 2022. 2

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023. 1

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2

[4] Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv:2306.15195*, 2023. 2, 7, 8

[5] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. *ICLR*, 2022. 2, 8

[6] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv:1504.00325*, 2015. 3, 7

[7] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. In *ICLR*, 2023. 2, 3, 6

[8] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020. 7

[9] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv:2210.11416*, 2022. 5

[10] 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. 1, 2

[11] Jiajun Deng, Zhengyuan Yang, Tianlang Chen, Wengang Zhou, and Houqiang Li. Transvg: End-to-end visual grounding with transformers. In *ICCV*, 2021. 7

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929*, 2020. 3

[13] Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *arXiv:2303.11331*, 2023. 2, 5

[14] Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023. 1

[15] Google. Bard. https://bard.google.com/, 2023. 1

[16] Alex Graves. Generating sequences with recurrent neural networks. *arXiv:1308.0850*, 2013. 3

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 4, 7

[18] 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. *arXiv:2106.09685*, 2021. 2, 5, 8

[19] Shaohan Huang, Li Dong, Wenhui Wang, Yaru Hao, Saksham Singhal, Shuming Ma, Tengchao Lv, Lei Cui, Owais Khan Mohammed, Qiang Liu, et al. Language is not all you need: Aligning perception with language models. *arXiv:2302.14045*, 2023. 2

[20] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *CVPR*, 2016. 7

[21] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In *ICCV*, 2021. 2, 7, 8

[22] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *ICCV*, 2023. 3, 5, 6

[23] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017. 2, 7, 8

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

[25] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Jingkang Yang, and Ziwei Liu. Otter: A multi-modal model with in-context instruction tuning. *arXiv preprint arXiv:2305.03726*, 2023. 1

[26] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, 2023. 1, 2, 3, 4, 5, 6

[27] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *CVPR*, 2022. 2, 3

[28] Xiangyang Li, Shuqiang Jiang, and Jungong Han. Learning object context for dense captioning. In *AAAI*, 2019. 7

[29] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *ECCV*, 2022. 5

[30] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 3, 7

[31] Chang Liu, Henghui Ding, and Xudong Jiang. Gres: Generalized referring expression segmentation. In *CVPR*, 2023. 7

[32] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *NeurIPS*, 2023. 1, 2

[33] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *CVPR*, 2016. 2, 7, 8

[34] Zihang Meng, Licheng Yu, Ning Zhang, Tamara L Berg, Babak Damavandi, Vikas Singh, and Amy Bearman. Connecting what to say with where to look by modeling human attention traces. In *CVPR*, 2021. 2, 6

[35] OpenAI. Chatgpt. https://openai.com/blog/chatgpt, 2022. 1

[36] 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:2306.14824*, 2023. 2, 8

[37] 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. In *ICCV*, 2015. 3

[38] Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. Connecting vision and language with localized narratives. In *ECCV*, 2020. 1, 2, 4, 6, 8

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 2

[40] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 2020. 1, 2, 3, 5

[41] Zhuang Shao, Jungong Han, Demetris Marnerides, and Kurt Debattista. Region-object relation-aware dense captioning via transformer. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. 7

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

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017. 3

[44] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *arXiv:2205.14100*, 2022. 1, 2, 3, 7

[45] 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. In *ICML*, 2022. 7

[46] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *CVPR*, 2023. 2

[47] Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, et al. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *arXiv:2305.11175*, 2023. 2, 7

[48] Jialian Wu, Jianfeng Wang, Zhengyuan Yang, Zhe Gan, Zicheng Liu, Junsong Yuan, and Lijuan Wang. Grit: A generative region-to-text transformer for object understanding. *arXiv:2212.00280*, 2022. 2, 7, 8

[49] Linjie Yang, Kevin Tang, Jianchao Yang, and Li-Jia Li. Dense captioning with joint inference and visual context. In *CVPR*, 2017. 7

[50] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. Unitab: Unifying text and box outputs for grounded vision-language modeling. In *ECCV*, 2022. 2, 3, 7, 8

[51] Guojun Yin, Lu Sheng, Bin Liu, Nenghai Yu, Xiaogang Wang, and Jing Shao. Context and attribute grounded dense captioning. In *CVPR*, 2019. 7

[52] Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. Ferret: Refer and ground anything anywhere at any granularity. *arXiv:2310.07704*, 2023. 2, 7, 8

[53] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *ECCV*, 2016. 2, 3, 7

[54] Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv:2303.16199*, 2023. 2

[55] Shilong Zhang, Peize Sun, Shoufa Chen, Min Xiao, Wenqi Shao, Wenwei Zhang, Kai Chen, and Ping Luo. Gpt4roi: Instruction tuning large language model on region-of-interest. *arXiv:2307.03601*, 2023. 2, 8

[56] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv:1904.07850*, 2019. 5, 7

[57] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv:2304.10592*, 2023. 1, 2

[58] Xueyan Zou, Zi-Yi Dou, Jianwei Yang, Zhe Gan, Linjie Li, Chunyuan Li, Xiyang Dai, Harkirat Behl, Jianfeng Wang, Lu Yuan, et al. Generalized decoding for pixel, image, and language. In *CVPR*, 2023. 7

[59] Xueyan Zou, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Gao, and Yong Jae Lee. Segment everything everywhere all at once. *arXiv:2304.06718*, 2023. 7