

Molmo and PixMo: Open Weights and Open Data for State-of-the-Art Vision-Language Models

Matt Deitke^{*†ψ} Christopher Clark^{*†} Sangho Lee[†] Rohun Tripathi[†] Yue Yang[†]
 Jae Sung Park^ψ Mohammadreza Salehi^ψ Niklas Muennighoff[†] Kyle Lo[†] Luca Soldaini[†]
 Jiasen Lu[†] Taira Anderson[†] Erin Bransom[†] Kiana Ehsani[†] Huong Ngo[†]
 YenSung Chen[†] Ajay Patel[†] Mark Yatskar[†] Chris Callison-Burch[†] Andrew Head[†]
 Rose Hendrix[†] Favyen Bastani[†] Eli VanderBilt[†] Nathan Lambert[†] Yvonne Chou[†]
 Arnavi Chheda[†] Jenna Sparks[†] Sam Skjonsberg[†] Michael Schmitz[†] Aaron Sarnat[†]
 Byron Bischoff[†] Pete Walsh[†] Chris Newell[†] Piper Wolters[†] Tanmay Gupta[†] Kuo-Hao Zeng[†]
 Jon Borchardt[†] Dirk Groeneveld[†] Crystal Nam[†] Sophie Lebrecht[†] Caitlin Wittlif[†]
 Carissa Schoenick[†] Oscar Michel[†] Ranjay Krishna^{†ψ} Luca Weihs[†]
 Noah A. Smith^{†ψ} Hannaneh Hajishirzi^{†ψ} Ross Girshick^{†ψ} Ali Farhadi^{†ψ} Aniruddha Kembhavi^{†ψ}

[†]Allen Institute for AI

^ψUniversity of Washington

Abstract

Today’s most advanced vision-language models (VLMs) remain proprietary. The strongest open-weight models rely heavily on synthetic data from proprietary VLMs to achieve good performance, effectively distilling these closed VLMs into open ones. As a result, the community has been missing foundational knowledge about how to build performant VLMs from scratch. We present **Molmo**, a new family of VLMs that are state-of-the-art in their class of openness. Our key contribution is a collection of new datasets called **PixMo**, including a dataset of highly detailed image captions for pre-training, a free-form image Q&A dataset for fine-tuning, and an innovative 2D pointing dataset, all collected without the use of external VLMs. The success of our approach relies on careful modeling choices, a well-tuned training pipeline, and, most critically, the quality of our newly collected datasets. Our best-in-class 72B model not only outperforms others in the class of open weight and data models, but also outperforms larger proprietary models including Claude 3.5 Sonnet, and Gemini 1.5 Pro and Flash, second only to GPT-4o based on both academic benchmarks and on a large human evaluation. Our model weights, new datasets, and source code are available at <https://molmo.allenai.org/blog>.

1. Introduction

Large multimodal models are used ubiquitously today. Proprietary models—GPT-4o, Gemini-1.5 Pro, Claude 3.5

Sonnet—produce comprehensive image descriptions and accurately answer complex visual questions. Unfortunately, the most performant of these vision-language models (VLMs) remain proprietary with neither model weights, data, nor code being publicly released.

To foster scientific exploration, numerous research efforts have attempted to reproduce similar capabilities in open models. Early works, exemplified by LLaVA [69], produced fully open weights and training data but now lag significantly behind the state-of-the-art. More recent, stronger open-weight models have trended towards less open data: the training data may either be proprietary (e.g., [10]) or, in cases where it is released, there is a heavy reliance on synthetic data generated by proprietary systems, e.g., models are trained on datasets like ShareGPT4V [15] which uses GPT-4V [88] to generate a large set of detailed image captions. The resulting VLMs, therefore, are effectively distillations of proprietary VLMs. As it stands, the scientific community is missing foundational knowledge about how to build performant VLMs from scratch (more discussion about this and related work are in the Appendix).

In this work, we present the **Molmo** (**M**ultimodal **O**pen **L**anguage **M**odel) family of state-of-the-art open VLMs with released model weights and released vision-language training data without any reliance on synthetic data from other VLMs, including proprietary ones. The success of our approach relies on careful model design choices, a well-tuned training pipeline, and most critically, the quality of our new datasets, collectively named **PixMo** (**P**ixels for **M**olmo), which are fully open.

High-quality multimodal data, both for pre-training and

^{*}Equal contribution

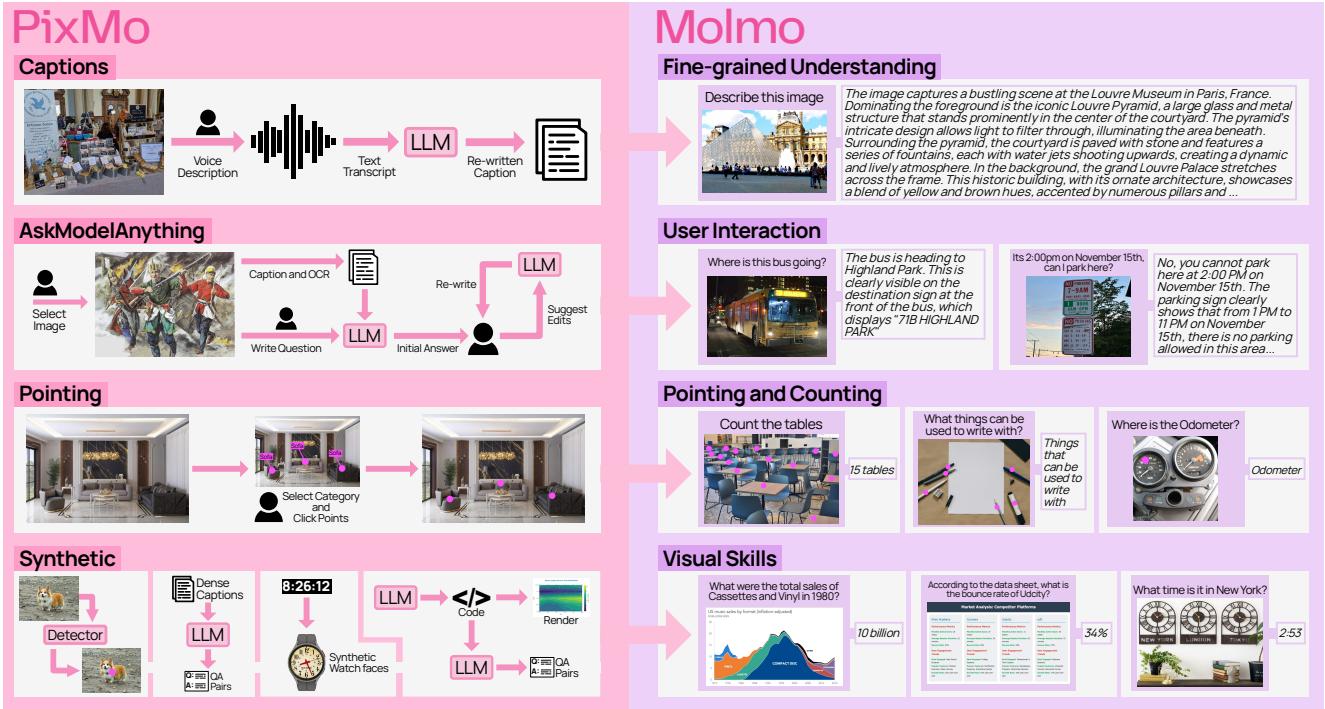


Figure 1. Datasets in **PixMo** (left) and the capabilities they enable in **Molmo** (right). PixMo consists of three annotated datasets and four synthetic datasets, all constructed without the use of VLMs. The annotated datasets include: dense captions (for pre-training), instruction following (for fine-tuning), and pointing (for fine-tuning, to support grounding and counting). The four synthetic datasets augment these datasets by targeting additional skills (*e.g.*, clock reading, document understanding).

fine-tuning, is a key missing piece for training open VLMs that are competitive with closed ones. The academic community has struggled to collect such datasets due to high costs and the difficulty of obtaining high-quality annotations from crowd sourcing platforms. To build PixMo, we introduce several key data collection innovations that allow us to quickly collect high-quality data from untrained annotators, see Figure 1.

PixMo includes a dataset of 712k images with very long (200+ word) detailed captions. Collecting this data was difficult because directly asking annotators to write such captions produces poor results: they tend to focus on a few salient visual elements [17], typing long paragraphs is time-consuming, and annotators can potentially copy-and-paste responses from proprietary VLMs, circumventing our goal of avoiding distillation. Instead, we ask annotators to describe images in *speech* for 60 to 90 seconds. Empirically, we found that with this modality switching “trick” annotators provide far more detailed descriptions in less time, and for each description we collect an audio receipt (*i.e.*, the annotator’s recording) proving that a VLM was not used.

PixMo also includes an array of fine-tuning datasets. To collect instruction-following data, we have users interactively edit responses with a language-only LLM to obtain high-quality and accurate free-form responses. We gather 162k annotations on 73k images in this way. We also collect a unique new data source that grounds language in images

with 2D points. Using points enables us to collect grounding data much faster than would be possible using bounding boxes or segmentation masks since it is much easier to annotate, and we take advantage of this by collecting over 2.3 million grounding annotations for a diverse range of objects, expressions, and scenes. This novel pointing data enables our models to answer some questions more naturally by pointing to the pixels that support the answer, improves counting accuracy (the model *counts by pointing*), and we believe it will open up an important future direction in which VLMs enable agents (*e.g.*, robots, web agents) to *act* by pointing in their environments, *e.g.*, to a navigation waypoint, to an object to pick up, or to a user interface button to press. Finally, we introduce several novel synthetic datasets (meaning with no or minimal human annotations, but still *not* using a VLM) with data targeting particular skills (clock reading, chart understanding, table understanding, *etc.*) that complements existing open-source datasets.

We train models on these datasets following a mostly standard design using a pre-trained LLM and vision encoder, but with some new improvements including a simplified two-stage training pipeline, a novel overlapping multi-crop strategy, an efficient method of training on images with multiple annotations, and some new insights in how to set up the optimizer and vision/language connector. We evaluate the Molmo family of models on 11 academic benchmarks and with a human evaluation that allows us to rank

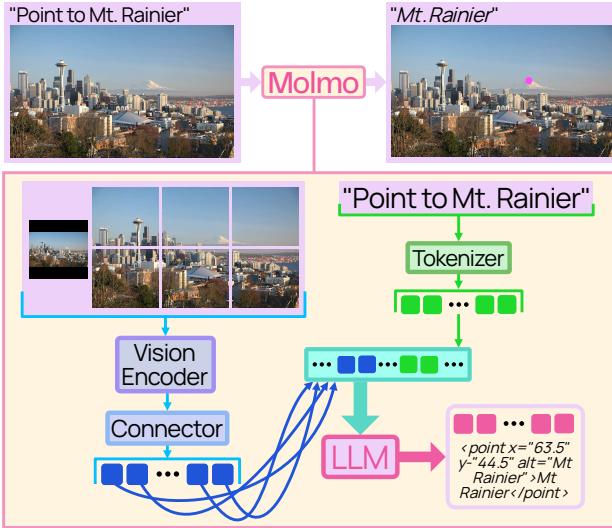


Figure 2. **Molmo** follows the simple and standard design of connecting a vision encoder and a language model.

models by user preference. Our most efficient model, MolmoE-1B, based on the OLMoE-1B-7B [87] mixture-of-experts LLM, nearly matches the performance of GPT-4V on both our academic benchmarks and user preference. Molmo-7B-O and Molmo-7B-D, based on OLMo-7B [37] and Qwen2 7B [120], respectively, perform comfortably between GPT-4V and GPT-4o [90] on both academic benchmarks and user preference. Our best-in-class Molmo-72B model, based on Qwen2 72B, achieves the highest academic benchmark score and ranks second by human preference, just behind GPT-4o. Our best model outperforms many state-of-the-art proprietary systems, including Gemini 1.5 Pro and Flash [103], and Claude 3.5 Sonnet [7]. We will also release a 100% fully open Molmo model, based on a MetaCLIP [118] vision encoder and OLMo LLM, for which every bit of training data is publicly available. In addition, we perform an expansive set of ablations to better inform the scientific community of how various model and data design choices affect VLMs.

2. Architecture

Our model architecture (Figure 2) follows a standard design, combining pre-trained language and vision models (*e.g.*, [69]). It has four components: (1) a pre-processor that converts the input image into multiscale, multi-crop images, (2) a ViT image encoder [31] that computes per-patch features for each image independently, (3) a connector that pools and projects patch features into the LLM’s embedding space, and (4) a decoder-only LLM [95, 109].

From this template, we build a family of models by selecting a vision encoder and LLM, keeping the training data and recipe consistent across choices (except for learning rates). We primarily use OpenAI’s ViT-L/14 336px CLIP



Figure 3. An image cropped without (left) and with (right) overlap. Highlighted regions show areas used by the LLM. Overlapping crops ensure that central patches are encoded with neighboring context; for example, the patches containing the bike’s brand name is always part of a crop where the entire name is visible.

model [96] due to strong performance in initial experiments, but similar results are achievable with SigLIP [130] and the *fully open* MetaCLIP [118] (see Section 6). For the LLM, we experiment across scales and openness levels: *fully open* OLMo-7B-1024-preview, *fully open* OLMoE-1B-7B (our most efficient model), open-weight Qwen2 7B [120], and open-weight Qwen2 72B (our best-performing model).

Cropping. Most ViTs only support square images at a fixed resolution that is generally too low for fine-grained tasks such as OCR or detailed captioning. To address this issue, we follow recent works [19, 30, 70, 85, 124] by dividing the image into multiple square crops that tile the image. Additionally, the full image, resized to the ViT’s resolution, provides a low-resolution overview. Each crop is processed independently by the ViT. See the Appendix for details.

One limitation of cropping is that border patches lack context from adjacent patches (see Figure 3). To mitigate this, we allow crops to overlap so each patch has context from at least some neighboring patches. Patch features from the overlap are *not* passed to the connector or LLM so that the passed patch features exactly tile the high-resolution image. Overlapping slightly reduces the tiled image resolution, but this can be offset by using more crops. Overlapping significantly improves results, as shown in Section 6.

Vision-language connector. Once crops are encoded by the vision encoder, we build patch features by concatenating features from the third-to-last and tenth-from-last ViT layers, which improves performance slightly over using a single layer. Each 2×2 patch window is then pooled into a single vector using a multi-headed attention layer, where the mean of the patches serves as the query. This attention pooling outperforms simple feature concatenation (see Section 6). Finally, pooled features are mapped to the LLM’s embedding space via an MLP.

Arranging vision tokens. Pooled patch features (vision tokens) are sequenced left-to-right, top-to-bottom, starting with patches from the low-resolution full image, followed

by high-resolution crop patches arranged in row-major order. Special tokens are inserted to mark the start and end of both low- and high-resolution patch sequences, with row-end tokens added between rows to indicate row transitions.

Dropout. Residual dropout is applied to the LLM, but not the image encoder and vision-language connector. During pre-training on dense captions, dropout is applied only to text tokens to encourage reliance on the encoded image rather than language priors. This is not used in fine-tuning, as shorter target responses result in too little dropout. Text-only dropout during pre-training enhances captioning and downstream performance, as shown in Section 6.

Multi-annotated images. Our multimodal data often includes multiple annotations per image (*e.g.*, VQA v2.0 has multiple question-answer pairs). To train efficiently, we arrange *all* of text annotation tokens for an image in one long sequence, masking attention so tokens for each annotation attend to the image tokens, each other, but not to tokens from different annotations. This setup is equivalent to training on individual image-text pairs but avoids redundant image encoding, reducing the number of processed images by two-thirds and shortening training time by over half, with only a 25% increase in sequence length for our data mix.

3. Data

PixMo contains seven datasets, three with human annotations and four created with synthetic data generation pipelines (see Figure 1). Below, we describe these datasets and data collection methods; additional details and examples are in the Appendix.

PixMo-Cap. We collected PixMo-Cap as a source of high-quality pre-training data, featuring a diverse set of images paired with highly detailed dense captions. We began by sourcing web images across ~70 diverse topics (*e.g.*, street signs, memes, food, drawings, websites, blurry photos, *etc.*). For each image, three annotators initially provided detailed descriptions by speaking for at least 60 seconds. In later stages, we used one annotator per image with a 90-second minimum, which improved efficiency without sacrificing quality. We prompted the annotators with seven questions to answer, detailed in the Appendix.

The annotators’ audio was transcribed using a standard speech-to-text system, yielding raw transcripts. A final high-quality image caption was then created by prompting a language-only LLM to summarize multiple raw transcripts per image or, for single transcripts, to enhance its quality (*e.g.*, removing spoken artifacts, normalizing style). In total, we collected 712k distinct images with 1.3M transcripts and captions. Our captions average 196 words, compared to 11 words in COCO captions [17] and 37 words in localized narratives [93], highlighting their greater detail.

PixMo-AskModelAnything. We collected this data to en-

able the model to answer diverse questions it might encounter in real-world use. To create image-question-answer triplets, annotators worked with a language-only LLM. An annotator selected an image from a large pool and wrote a question about it. Then, we ran a standard non-VLM OCR model and a PixMo-Cap-trained model on the image. The language-only LLM answered the question from the OCR data and dense caption. The annotator could accept or reject the answer and if rejected, they specified the issue and requested a revision until the answer was satisfactory. We collected 162k question-answer pairs in 73k images.

PixMo-Points. We collected pointing data to achieve three goals: (1) enable the model to point to items described by text, (2) enable the model to count by pointing, and (3) use pointing as a form of visual explanation when answering questions. For the first two goals, annotators were asked to point at something in an image, describe it, and then point to each instance of it in the image, ensuring exhaustive coverage. We also collected “not present” data so models can learn to handle cases where an item is *not* in the image. Pointing data also naturally supports answering counting questions with a chain-of-thought formed by the sequence of points. This resulted in 2.3M question-points pairs from 223k images. To enable points as explanations, we adapted the PixMo-AskModelAnything pipeline to let annotators pass the LLM a list of text-annotated points, prompting the LLM to use them in its answer when relevant. We collected 79k point-explanation annotations on 14k images.

PixMo-CapQA. We generated 214k question-answer pairs, covering diverse topics and styles, from 165k images by prompting a language-only LLM to ask and answer questions given only the ground-truth caption for an image.

PixMo-Docs. We used an extensive and carefully tuned prompting framework to prompt an LLM to generate code for 255k text and figure-heavy images, including charts, documents, tables, and diagrams. We then prompted the LLM to generate 2.3M question-answer pairs based on privileged access to the code (the images were not used).

PixMo-Clocks. We rendered synthetic clocks matched with a time-telling question-answer pair. The images use ~50 different watch bodies and ~160k realistic diverse watch faces set to random times. We collected 826k examples.

PixMo-Count. We used a standard non-VLM object detector [136] on web images to create image and counting QA pairs. For each image, we selected the class with the most detections after strict confidence thresholding. Following CountBenchQA [10], we manually verified 120 samples per count from 2 to 10, creating validation and test sets of 540 images each. These diverse images form a more challenging counting QA set than CountBenchQA, which has reported limitations [10]. The remaining samples with counts between 0 and 10 form a training set of 36k images, each annotated with points (object centers) and a QA pair.

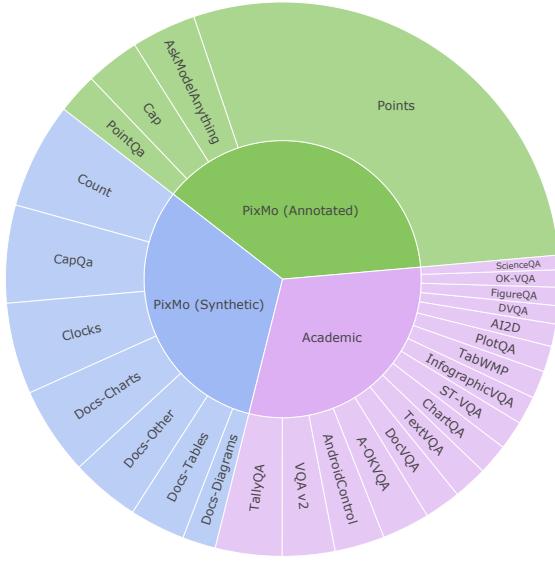


Figure 4. Datasets used for fine-tuning, shown in proportion to their sampling rates. Green denotes human-annotated data we collected, blue denotes synthetic data we generated, and purple represents pre-existing academic datasets. PixMo-Docs has been subdivided into charts, tables, diagrams, and other.

4. Training

Pre-training. We pre-train all model parameters on PixMo-Cap to generate either the caption or one of the audio transcripts for a given image. A prompt specifies which style to generate and, 90% of the time, includes a length hint to guide the model’s output length. This hint improves caption and pre-training quality, as shown in Section 6.

Previous work has often included a separate training stage to tune only the vision-language connector [10, 27, 59, 69, 106]. We find this step unnecessary when pre-training on PixMo-Cap (see Section 6), also explored in [48]. Instead, we apply a higher learning rate with a shorter warmup for the connector parameters, allowing them to adjust more quickly at the start of training. Skipping this stage reduces training time and complexity, and eliminates the need for the noisy web-scale data typically used in this phase.

We train for four epochs using AdamW [50, 73] with a cosine learning rate decaying to 10% of its peak. Learning rates are set to 2e-4 (connector), 6e-6 (ViT), and 2e-5 (LM), with a 200-step warmup for the connector and 2000 steps for the ViT and LM. Gradient clipping is applied separately to the LM, image encoder, and connector parameters. Full hyper-parameters are provided in the Appendix.

Fine-tuning. We fine-tune the model on a mix of PixMo datasets and open-source training datasets, including: VQA v2.0 (COCO 2014 subset) [36], TextVQA [100], OK-VQA [81], ChartQA (re-weighted to balance human and augmented examples) [82], DocVQA [83], InfographicVQA [84], AI2D (transparent and opaque label

boxes) [49], A-OKVQA [99], AndroidControl [62], ScienceQA [76], TabMWP [77], ST-VQA [11], TallyQA [2], DVQA [46], FigureQA [47], and PlotQA [86].

We sample datasets at rates proportional to the square root of their size, with manual down weighting of some very large synthetic datasets (PlotQA, FigureQA, DVQA, and PixMo-Clocks). We observe that pointing tasks learn more slowly than QA tasks, so we significantly up-weight the pointing data. Final mixture rates are shown in Figure 4, with full details in the Appendix.

The academic datasets in this mixture teach specific skills and help the model perform well on corresponding benchmark test sets. However, these datasets often have answer styles that are not ideal for user interactions, as answers are usually very short and may reflect unique stylistic quirks from data collection (*e.g.*, DocQA requires verbatim text from documents, while ChartQA specifies digits without commas). To prevent these styles from affecting user-facing responses, we prompt the model with a task-specific style tag (*e.g.*, prefixing VQA v2.0 questions with “vqa2:”). The model learns to use these styles only when requested.

We use style tags for all datasets except PixMo-AskModelAnything, -CapQA, -Points, -Count and -Cap. For PixMo-Cap, we create ~30 prompts for caption generation. For pointing data, we create ~100 question templates that ask for the location or count of the target expression. The model then returns a list of points and, for counting questions, the total count. These prompts and templates are randomly sampled during training. We still use a style tag for pointing-as-an-explanation data since we find performance in this mode can be less reliable, so it should only be used when users request it.

For pointing, the model outputs points as plain-text coordinates normalized between 0 and 100. When pointing to multiple items, points are ordered top-down, left-to-right, with each point numbered (see Figure 2 and details in the Appendix). Pointing enables a unique chain-of-thought approach to counting where the model counts by sequentially pointing to each occurrence of the target object, improving performance (see Section 6).

5. Evaluation

We evaluate on academic benchmarks, noting that comparisons require care, as prompting, alignment with benchmark-specific answer styles, and use of benchmark training data can significantly affect performance. To complement this, we conduct a human evaluation to rank models based on user preference.

For academic benchmarking, we gather or compute results for all models on 10 common datasets and the PixMo-Count test set, which we include due to its higher difficulty compared to existing counting benchmarks. We prioritize author-published results but fill in missing results

model	AI2D test [49]	ChartQA test [82]	VQA v2.0 test/dev [36]	DocVQA test [83]	InfoQA test [84]	TextVQA val [100]	RealWorldQA [116]	MMMU val [129]	MathVista testmini [78]	CountBenchQA [10]	PixMo-Count test	Average	Elo score	Elo rank	
<i>API call only</i>															
GPT-4V [88]	89.4	78.1	77.2	87.2	75.1	78.0	61.4	63.1	58.1	69.9	45.0	71.1	1041	10	
GPT-4o-0513 [90]	94.2	85.7	78.7	92.8	79.2	77.4	75.4	69.1	63.8	87.9	59.6	78.5	1079	1	
Gemini 1.5 Flash [103]	91.7	85.4	80.1	89.9	75.3	78.7	67.5	56.1	58.4	81.6	61.1	75.1	1054	7	
Gemini 1.5 Pro [103]	94.4	87.2	80.2	93.1	81.0	78.7	70.4	62.2	63.9	85.8	64.3	78.3	1074	3	
Claude-3 Haiku [7]	86.7	81.7	68.4	88.8	56.1	67.3	45.5	50.2	46.4	83.0	43.9	65.3	999	18	
Claude-3 Opus [7]	88.1	80.8	66.3	89.3	55.6	67.5	49.8	59.4	50.5	83.6	43.3	66.7	971	21	
Claude-3.5 Sonnet [7]	94.7	90.8	70.7	95.2	74.3	74.1	60.1	68.3	67.7	89.7	58.3	76.7	1069	4	
<i>Open weights only</i>															
PaliGemma-mix-3B [10]	72.3	33.7	76.3	31.3	21.4	56.0	55.2	34.9	28.7	80.6	60.0	50.0	937	27	
Phi3.5-Vision-4B [1]	78.1	81.8	75.7	69.3	36.6	72.0	53.6	43.0	43.9	64.6	38.3	59.7	982	19	
Qwen2-VL-7B [111]	83.0	83.0	82.9	94.5	76.5	84.3	70.1	54.1	58.2	76.5	48.0	73.7	1025	14	
Qwen2-VL-72B [111]	88.1	88.3	81.9	96.5	84.5	85.5	77.8	64.5	70.5	80.4	55.7	79.4	1037	12	
InternVL2-8B [104]	83.8	83.3	76.7	91.6	74.8	77.4	64.2	51.2	58.3	57.8	43.9	69.4	953	23	
InternVL2-Llama-3-76B [104]	87.6	88.4	85.6	94.1	82.0	84.4	72.7	58.2	65.5	74.7	54.6	77.1	1018	16	
Pixtral-12B [3]	79.0	81.8	80.2	90.7	50.8	75.7	65.4	52.5	58.0	78.8	51.7	69.5	1016	17	
Llama-3.2V-11B-Instruct [5]	91.1	83.4	75.2	88.4	63.6	79.7	64.1	50.7	51.5	73.1	47.4	69.8	1040	11	
Llama-3.2V-90B-Instruct [5]	92.3	85.5	78.1	90.1	67.2	82.3	69.8	60.3	57.3	78.5	58.5	74.5	1063	5	
<i>Open weights + data († distilled)</i>															
LLaVA-1.5-7B [69]	55.5	17.8	78.5	28.1	25.8	58.2	54.8	35.7	25.6	40.1	27.6	40.7	951	26	
LLaVA-1.5-13B [69]	61.1	18.2	80.0	30.3	29.4	61.3	55.3	37.0	27.7	47.1	35.2	43.9	960	22	
xGen-MM-interleave-4B† [119]	74.2	60.0	81.5	61.4	31.5	71.0	61.2	41.1	40.5	81.9	50.2	59.5	979	20	
Cambrian-1-8B† [106]	73.0	73.3	81.2	77.8	41.6	71.7	64.2	42.7	49.0	76.4	46.6	63.4	952	25	
Cambrian-1-34B† [106]	79.7	75.6	83.8	75.5	46.0	76.7	67.8	49.7	53.2	75.6	50.7	66.8	953	24	
LLaVA OneVision-7B† [59]	81.4	80.0	84.0	87.5	68.8	78.3	66.3	48.8	63.2	78.8	54.4	72.0	1024	15	
LLaVA OneVision-72B† [59]	85.6	83.7	85.2	91.3	74.9	80.5	71.9	56.8	67.5	84.3	60.7	76.6	1051	8	
<i>The Molmo family: Open weights, Open data, Open training code, Open evaluations</i>															
MolmoE-1B	86.4	78.0	83.9	77.7	53.9	78.8	60.4	34.9	34.0	87.2	79.6	68.6	1032	13	
Molmo-7B-O	90.7	80.4	85.3	90.8	70.0	80.4	67.5	39.3	44.5	89.0	83.3	74.6	1051	9	
Molmo-7B-D	93.2	84.1	85.6	92.2	72.6	81.7	70.7	45.3	51.6	88.5	84.8	77.3	1056	6	
Molmo-72B	96.3	87.3	86.5	93.5	81.9	83.1	75.2	54.1	58.6	91.2	85.2	81.2	1077	2	

Table 1. We present academic benchmark results for 10 common datasets, plus a new counting benchmark, PixMo-Count, which features more challenging natural images than CountBenchQA. We categorize models into four groups: (top) proprietary models accessible only via API calls, (upper middle) models with released weights but closed data, (lower middle) models with released weights and training data (noting some of these use distillation (†) from proprietary VLMs via synthetic data), and (bottom) the Molmo family of models.

with the best previously reported values from technical reports or sources like the OpenVLM Leaderboard. If data is still missing, we compute it ourselves. Notably, computing results is challenging, as performance can vary significantly (*e.g.*, by 10%) based on evaluation details. Additionally, critical information such as prompts or data processing steps is often unavailable, making it hard to reproduce results, highlighting the need for evaluation openness.

In our human evaluation, we collect 15k diverse image-text prompt pairs and queried the VLMs for responses. We sample and present the resulting image-text-response triplets for all VLM pairings to a group of ~870 human annotators, who provide pairwise preference rankings. Across all model pairs, we gather over 325k ratings (~450 per model pair). From this data, we calculate an Elo ranking

using the Bradley-Terry model, following the methodology of Chatbot Arena [21].

For Molmo, we evaluate all academic datasets with 36 crops (up from 12 used in training), except for counting tasks, as pointing capabilities do not generalize well with different numbers of test crops. A small amount of high-res post-training can resolve this issue, see the Appendix.

When possible, we use relevant style prompts¹ (*e.g.*, “vqa2:”). For evaluation-only datasets, we use the VQA v2.0 (for short answer) or A-OKVQA (for multiple choice) style tags to elicit the often expected short answer style. For human evaluation, we omit style tags and use 12 crops, as some counting questions use pointing. Evaluators are only shown the output text, not the points.

¹We use AI2D with transparent boxes; see Appendix for opaque boxes.

ViT-L/14	cap F_1	11-avg
OpenAI CLIP 336px	54.1	76.9
MetaCLIP 336px	54.1	77.2
SigLIP-So400m 384px	54.4	77.1
DINOv2 336px	53.2	75.6

(a) **Vision encoder.** Encoders that were trained on noisy web-scale image-text pairs perform similarly (rows 1-3). Surprisingly, DINOv2, which is trained on images only (no text, no label supervision), is competitive on these tasks. MetaCLIP and DINOv2 are *fully open*.

cropping	cap F_1	11-avg
single	46.7	62.8
multi, no overlap	53.4	75.7
multi, overlap	54.1	76.9

(d) **Cropping.** Using the entire image only (single crop) performs poorly. Our novel overlapping crop method (see Figure 3), which prevents loss of context, performs the best.

# crops train, test	cap F_1	11-avg
4, 4	52.0	71.0
4, 12	52.0	74.1
4, 36	52.0	74.2
12, 12	54.1	74.9
12, 36	54.1	76.9
36, 36	54.0	77.2

(b) **Image resolution.** Using more crops at training and testing time generally improves performance. However, captioning and counting can perform poorly when # of crops are unequal, so for these tasks we always set the number of test crops equal to the training value.

setting	cap F_1	11-avg
off	53.0	76.2
on	54.1	76.9

(e) **Length conditioning.** Captioning with length hints is a superior pre-training task compared to captioning alone as evident by the improved captioning and downstream results.

pre-train, fine-tune	cap F_1	11-avg
off, off	53.1	74.6
off, on	53.1	76.6
on, on	53.7	77.0
on (text only), on	54.1	76.9

(c) **Dropout.** Dropout in the LLM improves pre-training and fine-tuning results. In pre-training, applying dropout to captioning text tokens only further improves results. This design may encourage the model to rely more on vision tokens rather than past text tokens.

2×2 pooling	cap F_1	11-avg
stacking	53.7	76.1
attention	54.1	76.9

(f) **Pooling.** Pooling 2×2 windows of vision tokens using mean-query attention performs better than simply stacking the four features as input to the vision-language connector MLP.

Table 2. **Model ablations.** Default settings are marked in gray. See the Appendix for additional ablations.

Broadly speaking, the academic benchmark results and human evaluation agree, with the exception of Qwen2-VL [111], which performs strongly on the academic benchmarks and comparatively underperforms in the human evaluation. We highlight a few key results from Table 1:

- MolmoE-1B, our most efficient model based on the OLMoE-1B-7B mixture-of-experts LLM, nearly matches GPT-4V on academic benchmarks and Elo.
- Molmo models based on OLMo-7B-1024-preview and Qwen2 7B LLMs perform between GPT-4V and GPT-4o on academic benchmarks and Elo.
- Our best-in-class Qwen2 72B based model achieves the highest academic benchmark score and ranks second in Elo, just behind GPT-4o.
- Our best model also outperforms many state-of-the-art proprietary systems, including Gemini 1.5 Pro and Flash and Claude 3.5 Sonnet.

Molmo-72B also underwent an independent Elo evaluation via Chatbot Arena, where it outperforms all open models but ranks lower than several proprietary models (*e.g.*, GPT-4o and Claude 3.5 Sonnet).² The full results table is in the Appendix. The difference likely stems from the types of questions evaluated. While we cannot perform a full analysis since the questions are not public, we do note our data includes many counting and image-description questions which are particular strengths of Molmo.

Molmo excels at answering questions about natural images, matching or outperforming all models on the zero-shot RealWorldQA benchmark and achieving state-of-the-art results on the highly competitive VQA v2.0.

²<https://1marena.ai/?leaderboard> vision arena, English category, accessed Nov. 13, 2024.

On OCR-centric benchmarks (ChartQA, DocQA, InfoQA, TextVQA), Molmo surpasses other open models and some proprietary ones but trails slightly behind Qwen2-VL. On counting tasks (CountBenchQA and PixMo-Count), Molmo leads all models due to our new pointing data and chain-of-thought point-and-count abilities. However, on reasoning tasks (MMMU, MathVista) Molmo lags, likely because its training mix lacks data focused on advanced reasoning.

We conduct several additional skill-specific evaluations, summarized here with details in the Appendix. On a clock-reading benchmark [121], Molmo at all scales dramatically outperforms other VLMs including proprietary ones, but trails specialized non-VLM models [121]. To assess Molmo’s potential for *action*, we tested Molmo-72B on AndroidControl [62], achieving 88.7% low-level and 69.0% high-level accuracy, comparable to the reported 83.2% and 70.8% in [62]. On NLP benchmarks, Molmo shows a slight performance drop versus its component LLM, which can be offset by additional text-only data. We also introduce a new pointing benchmark using SAM [51], where Molmo models at all scales demonstrate strong performance.

6. Ablations

We performed extensive ablations on model design (Table 2) and training data (Table 3), reporting two metrics: an F_1 metric (“cap F_1 ”) we developed to measure the precision and recall of captions generated by the model after pre-training (details in the Appendix), and the average accuracy on our 11 benchmark suite (“11-avg”), using validation sets when available. We report F_1 because we believe it reflects broad-range image understanding learned during pre-training, and many of our modeling design choices were

# PixMo-Cap images	cap F_1	11-avg
0 (0.0%)	-	74.9
89k (12.5%)	49.6	75.5
178k (25.0%)	51.6	76.3
356k (50.0%)	52.6	76.2
712k (100.0%)	54.1	76.9

(a) **PixMo-Cap scaling.** Increasing the quantity of PixMo-Cap captioning data in both pre-training and fine-tuning improves captioning and downstream tasks.

data	cap F_1	11-avg
stage 0.5 LAION	53.9	76.9
ShareGPT4V+o (158k images)	36.3	74.9
<i>PixMo-Cap images:</i>		
our raw transcripts only	45.2	76.4
our cleaned transcripts only	53.0	76.5
our raw & cleaned transcripts	54.1	76.9
captioned by GPT-4o	52.9	77.5

data	11-avg
academic only	72.5
plus PixMo-Docs	74.0
PixMo-* plus academic	76.9
remove PixMo-AMA	76.8
remove PixMo-CapQA	77.0
remove PixMo-Docs	75.8
remove PixMo-Clocks	76.9
remove pointing task	76.2

(b) **Pre-training data.** Our human annotated data performs on par with distilling GPT-4o by using it to caption the same set of images, demonstrating the effectiveness of our data.

(c) **Supervised fine-tuning data.** The PixMo-***** datasets not only give the model new capabilities (*e.g.*, pointing), but also generally improve results on the 11 dataset benchmark.

Table 3. **Data ablations.** Default settings are marked in gray .

strategy	CBQA	PCQA
count	87.9	80.2
point then count	89.4	86.3
count then point	81.5	77.6
pointing + regex	88.4	85.4

(a) **Counting strategy.** Pointing is the key ingredient in Molmo’s counting abilities.

order	CBQA	PCQA
on	89.4	86.3
off	85.4	74.1

(b) **Point order.** Training on ordered points (top-down, left-right) is better than unordered points.

points, length	CBQA	PCQA
actual, correct	89.4	86.3
random, correct	85.9	76.3
random, random	76.3	75.7

(c) **Inference compute.** Simply increasing inference compute with extra tokens does not help.

tokens	CBQA	PCQA
plain-text	89.4	86.3
special	85.8	80.9

(d) **Special point tokens.** Encoding point coordinates as plain-text works best.

Table 4. **Counting ablations.** Defaults are in gray . CBQA is the CountBenchQA test set and PCQA is the PixMo-Count validation set.

based on this evaluation since it does not require running the more costly fine-tuning stage. While performing these ablation we observed captioning improvements generally, but not always, correspond to benchmark suite improvements. Our ablations test modifications to the Molmo-7B-D model configuration with key findings summarized below and further details in Table 2 and 3 captions and the Appendix.

Model ablations. We vary several design choices, finding:

- Vision encoders trained on noisy web-scale data (CLIP, SigLIP, MetaCLIP) all work roughly the same, including the *fully open* MetaCLIP (Table 2a). *Every model component and every bit of data in a Molmo model equipped with MetaCLIP and OLMO is open.*
- Excluding pointing and captioning, increasing the image resolution by increasing the number of crops generally improves performance, and tuning at a high resolution yields a slight gain (Table 2b).
- Our novel text-only dropout used in pre-training improves captioning performance (Table 2c).
- Using multiple crops, instead of a single low-resolution input, is extremely important. Our novel overlapping crop design yields significant improvements (Table 2d).
- Our novel length-conditioned captioning is a strong pre-training task, improving downstream results (Table 2e).
- Attention pooling yields improvements to both metrics over the baseline feature stacking approach (Table 2f).

Data ablations. We train with various data choices, finding:

- Scaling PixMo-Cap from 0 to 712k images significantly improves captioning and benchmark metrics (Table 3a).
- Adding noisy web-scale data does not improve our pre-training, our PixMo-Cap data yields better results than

a similar amount of data from ShareGPT4v/o, and our human annotated captioning data works roughly as well as captioning data generated by GPT-4o on the PixMo-Cap images (Table 3b).

- The PixMo supervised fine-tuning datasets improve benchmark task performance beyond the academic datasets, mainly by improving document-heavy and counting tasks (Table 3c).

Counting. We ablate several details of counting using models fine-tuned on just PixMo-Points and PixMo-Count data:

- Chain-of-thought point-then-count performs significantly better than generating a count only or generating a count followed by points. Pointing on its own, counted by regular expression, is only slightly worse (Table 4a).
- Training with points in a predicable spatial order (top-down, left-to-right) works significantly better than training with unordered points (Table 4b).
- Pointing uses extra inference computation that may improve results regardless of the generated text. However, we find that using correct point locations and counts works significantly better than using random locations and/or numbers of points (Table 4c).
- Representing point coordinates in plain-text works better than introducing special location tokens (Table 4d).

Human evaluation. A human evaluation of select ablation models in Table 5 shows that PixMo data, especially PixMo-Cap and PixMo-AskModelAnything, is important for generating responses that users like. Academic datasets improve human scores, but are extremely poor if used on their own. GPT-4o captions on our images also perform well, which we believe is due to recent advances in GPT and

model	Elo score	win % vs. default
<i>Claude-3.5 Sonnet</i>	1047	65%
PixMo-Cap w/ GPT-4o captions (Table 3b)	1018	55%
PixMo-*, remove PixMo-CapQA (Table 3c)	1015	50%
Molmo-7B-D default	1014	n/a
PixMo-*, no academic datasets (Table 3c)	1013	42%
<i>GPT-4V</i>	1010	47%
DINOv2 vision encoder (Table 2a)	999	45%
PixMo-*, remove PixMo-AMA (Table 3c)	995	40%
no PixMo-Cap data (Table 3a)	990	35%
academic only (Table 3c)	897	17%

Table 5. Elo scores and win rates (excluding ties) for select ablations of Molmo-7B-D and two API-only models for context.

the diversity of our image collection (*e.g.*, the ShareGPT datasets significantly underperform our data even at the same scale, see Tables 3b and 3a). While distilling from proprietary models might be effective, we emphasize that it is critical for the scientific community to understand how to train competitive VLMs without doing so. **Molmo** and **PixMo** take an important step towards this understanding.

Appendix

The appendix includes the following sections:

- §A - Model Details
- §B - Training Details
- §C - Evaluation Details
- §D - Result Details
- §E - Ablation Details
- §F - Data Details
- §G - Dataset Examples
- §H - Related Work

A. Model Details

We present additional details about image encoding, hyper-parameters, and implementation choices.

A.1. Image Encoding

Our method of encoding images is shown in Figure 5. Cropping is done by first choosing a rectangular grid (*e.g.*, a 2×2 , 3×1 , etc.) where each square of the grid matches the ViT’s input size. When using overlapping crops, these squares are moved closer together so that they overlap by a fixed margin (we use a margin of 4 patches or 56 pixels), which reduces the overall size of the grid.

Then the image is up-scaled to fit within that grid as well as possible while preserving its aspect ratio by making either its height or width the same size as the grid. The grid is chosen to require the least amount of up-scaling, and in the event of ties, to minimize its size. We also set a maximum number of crops, and if the image cannot be covered by that many crops, the image will instead be down-scaled to fit the grid, and the grid is chosen to minimize

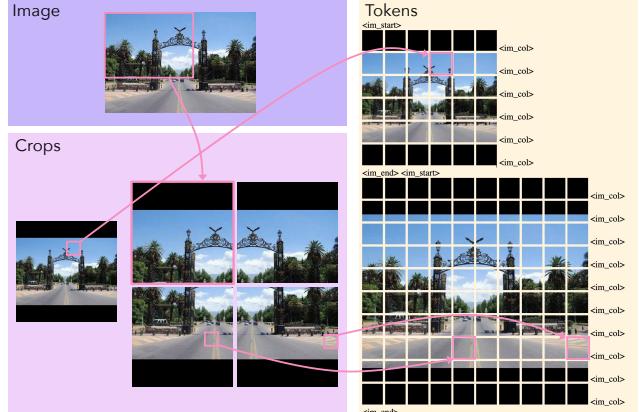


Figure 5. **Converting an image into tokens.** The image (top left) is turned into a single low-res and several overlapping high-res crops (bottom left). Padding (the black borders) is used so each crop is square and the aspect ratio of the image is preserved. The final token sequence for the image (right, arranged top-down left-to-right with line breaks for clarity) is built by extracting patch-level features from the crops, shown here using images of the patches, and special tokens. An image start and image end token are placed before/after the high-res and low-res patches, and column tokens are inserted after each row of patches. This example uses 4 high-res crops and extracts features from 36 (6×6) patches per crop, in practice Molmo typically uses 12 high-res crops and extracts features from 144 (12×12) patches per crop.

the amount of down-scaling required while not exceeding the maximum number of crops. In either case, the re-scaled image is padded with black borders so that it exactly fits the grid, and then crops are extracted from this padded image. The low-resolution crop is built by resizing and padding the image so it matches image ViT’s supported resolution.

Each crop is processed independently by the ViT and connector to get visual embeddings of each patch. A learned embedding is added to the patch features from each crop (before the connector is applied) depending on whether that patch includes no padding, some padding, or is all padding, so the model can distinguish padding from images that naturally have black borders. These embeddings are arranged with special tokens as described in Section 2, also shown in Figure 5 right. For image/text inputs we encode the input image first, followed by any text.

A.2. Hyper-Parameters

Hyper-parameters for the Molmo models and the AdamW [50, 73] optimizers are shown in Table 6. The connector MLP uses the same intermediate dimension as the LLM, so its size depends on the LLM. The connector pooling layer and ViT architecture are the same between all models. All runs used a cosine learning rate schedule ending at 10% of the peak learning rate [72].

Learning rates are similar between the models, except we

	1B-E	7B-D	7B-O	72B-D
Image Encoder	Params	290m		
	Dim	1024		
	MLP Dim	4096		
	Act.	GELU		
	Heads	16		
	KV Heads	16		
	Layers	23		
	Image Size	336×336		
V/L Connector	Patch Size	14		
	Dropout	0.0		
	Params	12m	110m	74m
	Pool Size		2×2	
LLM	Pool Dim		1024	
	Pool Heads		16	
	MLP Dim	1024	37888	22016
	Act.		SwiGLU	
	Dropout		0.0	
Pre-Train	Params	1.2b (6.9b)	7.6b	7.3b
	Embed	50304	152064	100352
	Dim	2048	3584	4096
	MLP Dim	2048×64	37888	22016
	Act.		SwiGLU	
	Heads	16	28	32
	KV Heads	16	4	32
	Layers	16	28	32
	Theta	10k	1m	0.5m
	Dropout		0.1	1m
Fine-Tune	Warmup ViT		2000	
	Warmup Con.		200	
	Warmup LLM		2000	
	LR ViT		6e-6	
	LR Con.		2e-4	
	LR LLM	2e-5	2e-5	2e-5
	Cosine Decay		10%	
	Eps.		1e-6	
	Betas		0.9, 0.95	
	Batch Size		128	
	Steps		22.3k	
	Warmup ViT		200	
	Warmup Con.		200	
	Warmup LLM		200	
	LR ViT	5e-6	5e-6	5e-6
	LR Con.	5e-6	5e-6	5e-6
	LR LLM	2e-5	1e-5	1e-5
	Cosine Decay		10%	
	Eps.		1e-6	
	Betas		0.9, 0.95	
	Batch Size		256	
	Steps	30k	30k	32k
				20k

Table 6. **Model and training hyper-parameters.** Molmo-1B-E has 1.2b active parameters, but 6.9b total. Its LLM MLP layers have 64 experts with 8 active at once.

find it helpful to reduce the learning rate for Molmo-72B. We also find Molmo-72B learns faster than the other models and can therefore be trained for fewer steps. Molmo-7B-O was trained for slightly longer due to a minor configuration difference, but we do not think it affected performance.

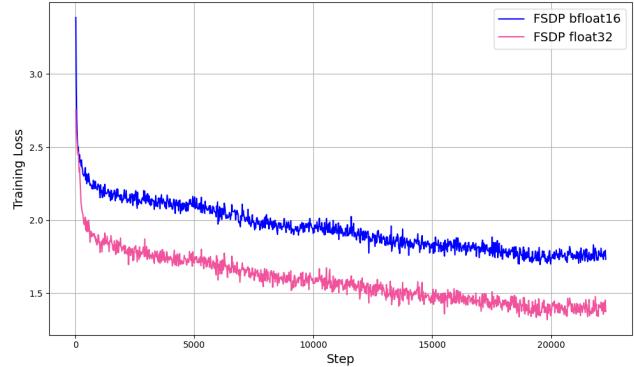


Figure 6. **Training loss curves** for Molmo-7B-D with model weights and gradient reduction in bfloat16 (blue) and float32 (pink). Float32 is our default configuration.

A.3. Implementation

Our implementation uses PyTorch with Fully Sharded Data Parallel (FSDP) [135] based on the OLMo codebase [37]. We do not use FlashAttention [28, 29] since it does not support the more complex masks that are required for multi-annotated images, but we find using PyTorch’s Scaled Dot Product Attention (SDPA) achieves close to the same speed.

To improve throughput, we utilize PyTorch’s Automatic Mixed Precision (AMP) module³, which enables most operations to run in half-precision with bfloat16 numbers. However, as shown in Figure 6, keeping model weights and performing gradient reduction in half-precision degrades training loss, so these are retained in full precision. Additionally, computations for layer normalization [8] and Rotary Position Embedding (RoPE) [101] are explicitly carried out in full precision.

When computing gradients with FSDP, each GPU computes a gradient on a small mini-batch of examples, after which gradients are averaged across all devices. We always compute the per-device gradient by dividing the total loss on that device by the *average* number of loss-tokens across all devices, not the number of loss tokens on that particular device. This avoids a subtle bias that effectively up-weights examples with a small number of loss tokens (*e.g.*, with short responses) since those examples tend to be paired with a smaller divisor if using the device-local number of loss tokens. Using the average number of loss tokens across all devices largely resolves this issue since our global batches are much larger than the device-local batches. This issue has been discussed in other places⁴ [40] and is known to have affected many codebases.⁵ We observe that captioning performance can drop by 0.5-1 points without this fix.

During fine-tuning, mixing is done within each batch so

³<https://pytorch.org/docs/stable/amp.html>

⁴<https://unsloth.ai/blog/gradient>

⁵<https://github.com/huggingface/trl/issues/2175>

batches contain examples from a variety of tasks. We set a maximum sequence length of 2304 for both pre-training and fine-tuning, and truncate examples longer than that (in practice, truncation only happens for certain synthetic datasets like DVQA [46] which contains many annotations per image, or for the occasional outlier example in other datasets).

We find training to be stable, without loss spikes or NaNs, likely in part because we use pre-trained models.

B. Training Details

Here we discuss the training mixture and how tasks are formatted during pre-training and fine-tuning.

B.1. Pre-Training Task Details

During pre-training, we train on each image paired with its caption and one of its audio transcripts. For images with multiple transcripts, we select one randomly each epoch. We use multi-annotation training (see Section C) to train on both the caption and the transcript jointly.

We prompt the model with either “long_caption:” or “transcript:” for captions and transcript respectively (a natural language prompt is used instead during instruction fine-tuning). We also add a length hint: an integer providing a noisy hint as to the correct output length. This hint is computed as the length of the transcript/caption in characters, plus a noise factor drawn from a random normal with a standard deviation of 25. The hint is then divided by 15 and rounded down to keep the hint in roughly the range of 0 to 100. This noise is added so that the length functions more like a guideline than a hard constraint, leaving the model some flexibility to adjust the caption as appropriate for the image. For example, even with a long length hint, it’s preferable that a caption for a very plain image be short instead of becoming repetitive or inane due to lack of content to describe.

We add the hint to the prompt 90% of the time, for example: “long_caption_83:” for a length hint of 83, and 10% of the time no length hint is used to maintain the ability to output a default caption.

Adjusting the length hint allows a trade off between precision and recall in captioning, see Figure 7 (see Section C for captioning metric details). In all of these settings, the average caption length when using a length hint is within 10 characters of the expected length, showing the models follow the length hints well. For our ablations, we report scores with a length hint of 65, which performs similarly or slightly better than using no length hint.

Preliminary experiments with mixing in other sources of captions (COCO Captions [17], Localized Narratives [93], or captions derived from Visual Genome annotations [52]) did not improve scores on our captioning metric, so we use PixMo-Cap alone for pre-training.

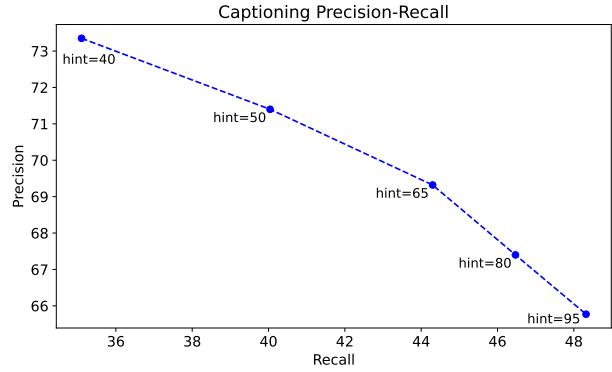


Figure 7. **Captioning precision and recall with different length hints** for Molmo-7B-D after pre-training. A short hint reduces recall since the model describes fewer things, but can boost accuracy since the description tends to focus on the more salient, easier-to-understand parts of the image.

name	rate	images	anno.	tokens	avg. crops
PixMo (Annotated)	38.1	1m	3.3m	350m	10.6
Points	28.8	220k	2.3m	160m	10.1
AskModelAnything	3.8	71k	160k	17m	10.0
Cap	3.2	712k	712k	160m	10.9
PointQA	2.4	14k	76k	11m	11.0
PixMo (Synthetic)	31.6	1.3m	3.3m	120m	11.0
Count	6.2	36k	37k	3m	11.9
CapQA	5.7	160k	210k	38m	10.8
Clocks	5.3	800k	800k	20m	10.5
Docs-Charts	5.2	120k	1.1m	34m	12.8
Docs-Other	4.0	71k	610k	15m	12.8
Docs-Tables	3.3	47k	420k	12m	12.1
Docs-Diagrams	1.9	16k	140k	3.6m	12.5
Academic	30.3	880k	25m	1b	8.0
TallyQA	3.9	130k	250k	4.6m	6.1
VQA v2.0	3.1	83k	440k	7.9m	6.7
AndroidControl	2.9	74k	300k	13m	11.0
A-OKVQA	2.8	17k	17k	380k	6.8
DocVQA	2.1	10k	39k	1m	12.9
TextVQA	2.0	22k	35k	700k	12.7
ChartQA	1.8	18k	28k	850k	9.3
ST-VQA	1.7	18k	25k	530k	4.6
InfographicVQA	1.6	4.4k	24k	670k	12.0
TabWMP	1.6	23k	23k	930k	2.3
PlotQA	1.5	160k	20m	930m	12.4
AI2D	1.3	6.2k	15k	630k	6.4
DVQA	1.1	200k	2.3m	51m	5.0
FigureQA	1.1	100k	1.3m	25m	6.1
OK-VQA	1.0	9k	9k	180k	6.8
ScienceQA	0.8	5k	6.2k	460k	4.3

Table 7. **Full list of instruction fine-tuning tasks.** Columns show the sampling rate, the total number of images and annotations (*i.e.*, the number of question/answer pairs), the number of text tokens using the Qwen2 tokenizer, and the average number of crops. The number of crops per an image can be at most 13 (one low-res, and 12 high-res), but can be lower for datasets with smaller images. Shaded rows show the total counts for all datasets in the category.

B.2. Fine-Tuning Task Details

Table 7 shows a list of our fine-tuning tasks. We only train on the train sets. Formatting and task-specific details are described below.

Multiple choice questions. For multiple choice questions in academic datasets (AI2D, A-OKVQA, ScienceQA), we append “Choices :”, a newline, and then new-line separated options with capital letter answer labels. The model predicts the answer label only. Note some multiple-choice questions appear in other, more diverse formats in PixMo-CapQA and PixMo-AskModelAnything.

Multiple answers. For datasets with multiple answers per question (*e.g.*, VQA v2.0), we only use the most common answer for training. If there are multiple answers that are equally common, we randomly select from among them each epoch.

Pointing. Pointing uses an HTML-like format. (x,y) coordinates are scaled to 0-100. For a single point, the format is:

```
<point x="10.0" y="10.0" alt="alt
text">Inline text</point>
```

For multiple points the format is:

```
<points x1="10.0" y1="10.0" x2="20.0"
y1="20.0" ... alt="alt text">Inline
text</points>
```

Numbering the points makes counting easier because the total count is always the number of the last point.

When interacting with users, we generally replace the point(s) text with the inline text, and show the image with the points using the alt text as hover text. For pointing and counting, the inline and alt text are both the name of what is being pointed at. For pointing-as-an-explanation these fields can be different.

PixMo-Points. Counting or pointing with a very large number of objects can lead to very long sequence lengths. To avoid memory errors we do not train on data with more than 40 counts; we expect to remove this limitation in future iterations of Molmo.

PixMo-AskModelAnything. “How many” questions are common in PixMo-AskModelAnything, but are not accompanied with pointing data. We observe that this can lead to the model failing to point when asked counting questions. To resolve this, we heuristically detect such questions and prefix them with an instruction to not point (*e.g.*, “Answer without points.”), randomly selected from a pool of 20 such instructions.

AI2D. AI2 Diagrams requires labeling regions of the images with letters, and then training the model to answer questions by predicting the correct region by returning its

	GPUs	pre-train time	GPU hr.	GPUs	fine-tune time	GPU hr.
1B-E	8	33.3	264	64	13.3	850
7B-D	64	8.6	550	128	11.2	1.4k
7B-O	64	8.9	570	128	13.5	1.7k
72B	128	33.3	4.2k	256	32.4	8.3k

Table 8. Training times for the Molmo models using H100 GPUs.

letter. Evaluations in the literature have been mixed between labeling the regions with opaque boxes (*e.g.*, [5, 89]) and transparent boxes (*e.g.*, [10, 71, 106]). We train in both settings and present our main results with transparent boxes. Results with opaque boxes are in Section D.

For AI2D questions where the answers are just letters, we list the multiple-choice options without a letter prefix.

AI2D does not have a validation set, so we built our own custom validation set by separating out 384 images (with roughly 2000 questions-answer pairs). None of our models are trained on this set.

ChartQA. The ChartQA train set contains many synthetic questions (21k synthetic *vs.* 7k non-synthetic), which we observe can be noisy and lower quality. To reduce the weight of these examples we re-weight ChartQA so the total weight of the synthetic and non-synthetic examples are equal. This also means the training data better matches the validation and test data which are evenly split between synthetic and non-synthetic questions.

A-OKVQA. We train on the multiple choice questions and, for questions not marked as difficult direct answer, also use them as direct answer questions by not using the answer options. We use different style tags for direct answer and multiple choice versions of the questions.

TabWMP. For TabWMP we treat the task as short answer and do not show the multiple-choice options.

AndroidControl. We train on four input-output configurations: low-level instruction to action, high-level goal to action, low-level and high-level inputs to action, and high-level goal to action with chain-of-thought reasoning. Only the instruction and screenshot are provided to the model as input; accessibility trees, action history, and a prompt with details like available actions are omitted. Target actions are represented as text output strings and (x,y) coordinates are scaled to 0-100 just as with regular pointing.

B.3. Training Time

Training time and number of GPUs used are shown in Table 8. All models were trained with H100 GPUs with infiniband connectivity.

C. Evaluation Details

Captioning metric (cap F_1). We measure captioning quality, relative to an evaluation set of 1500⁶ images, using the harmonic mean of captioning *precision* and *recall*, *i.e.* the F_1 score. The evaluation set was gathered through a similar protocol as PixMo-Cap (selecting a small number of images matching a diverse set of categories), but the images were selected manually and are disjoint from images in PixMo-Cap. Each evaluation image has up to six audio transcripts associated with it. To define the precision and recall of a caption for an image, let g be the generated caption and T be the set of ground-truth transcripts for the image. We prompt GPT-4o to enumerate a list of all distinct atomic statements contained in g and, separately, the transcripts in T . We then prompt GPT-4o to match each item in the list of atomic statements from g to items in the list of atomic statements from T . To compute recall, we consider matches as true positives and unmatched items from T 's list as false negatives. To compute precision, we prompt GPT-4o with the raw transcripts and the list of statements from g and ask it to say if each statement is *consistent* (a true positive) or *inconsistent* (a false positive) with the transcripts. (We avoid using the atomic statements from T when computing precision because it's a potentially noisy processing step that is not necessary.) We average precision and recall over all images in the evaluation set and compute the F_1 score of the averaged precision and recall values to produce our final summary metric: cap F_1 .

While this metric is imperfect (*e.g.*, GPT-4o makes mistakes, the transcripts do not contain all true statements about the image, *etc.*) we found that improvements to cap F_1 corresponded to improvements in our subjective impressions of caption quality, and thus it was a useful internal metric for guiding model and data design. Most of our model design and pre-training data decisions were based on improving captioning quality, see Section D and Figure 9 for more discussion.

Human evaluation. We defined 10 question categories and crowd sourced image-question pairs, using the same workers as for other annotation tasks. This resulted in the following categories and image-question pair counts:

- Output formatting: 1525
- Fine-grained detailed QA: 1510
- General: 1504
- Documents: 1499
- Captioning: 1493
- Counting: 1490
- Homework: 1489
- Charts: 1473

⁶A few evaluations were done with a super-set of 2730 images. We do not expect this to have affected results significantly since the 1500 were a random subset of the 2730.

- Named entities: 1448
- Creativity: 1420

We performed two human evaluations, the first for a large set of models presented in our main results Table 1 and a second for a smaller set of ablation models in Table 5. To collect feedback, we presented an annotator with an image, a question, and the output of “model A” and “model B”, without revealing the model identities. The annotator had five options: *tie (both bad)*, *tie (both good)*, *model A is better*, *model B is better*, or *I don't know*. The last option was for cases where the annotator did not know the correct answer (*e.g.*, a math problem they do not know how to solve). For the first study, we sampled image-question pairs randomly from the 10 categories, until we had collected ~450 feedback responses per model pair. For the second study, we used a refined methodology in which we first manually verified the quality of the question-image pairs, resulting in a fixed set of 500 questions (exactly 50 per category), all of which were used for each pair of models. After collecting the feedback, we removed the *I don't know* responses and computed an Elo ranking using the Bradley-Terry model, following the methodology of Chatbot Arena [21].

AndroidControl. To evaluate Molmo on AndroidControl, we provided only the task instruction (high-level or low-level) and the current screenshot as input. The results are reported on the in-domain data test set (IDD) and the metric is step-wise accuracy.

D. Result Details

Chatbot Arena. In Table 9 we report a summarized version of the vision leaderboard for queries in English from the Chatbot Arena [21], an independent third-party VLM evaluation. Molmo-72B outperforms all the fully open and open weight models but lags behind some of the propriety VLMs. As noted in the paper, we did our own Elo evaluation (see Section 5), in which Molmo-72B ranks higher (2nd place). The difference in rating likely stems from the types of questions evaluated. We do note our data includes many counting and image-description questions which are particular strengths of Molmo.

Clock reading. PixMo-Clocks is a novel source of clock reading data, a data type that is missing from most VLM training data (for which data information is published; we cannot know for models without published data details, such as API-only models and many open-weight models such as Pixtral and Llama 3). PixMo-Clocks are entirely synthetic and show a variety of watch bodies and faces against plain backgrounds (see Figure 17 for examples).

We tested how well Molmo trained on this data performs on the in-the-wild clock reading benchmark introduced in [121]. The benchmark sources clock images from three different datasets, COCO [65], OpenImages [53] and

model	score	95% CI	openness
Gemini-Exp-114 [103]	1278	+28/-27	API only
ChatGPT-4o-latest (20240903) [90]	1256	+13/-13	API only
Gemini-1.5-Pro-002 [103]	1220	+15/-14	API only
Gemini-1.5-Flash-002 [103]	1219	+15/-17	API only
GPT-4o-2024-05-13 [90]	1213	+9/-9	API only
Claude 3.5 Sonnet (20240620) [7]	1187	+9/-7	API only
Claude 3.5 Sonnet (20241022) [7]	1184	+15/-15	API only
Gemini-1.5-Pro-001 [103]	1158	+9/-8	API only
GPT-4-Turbo-2024-04-09 [88]	1157	+7/-10	API only
Gemini-1.5-Flash-8B-Exp-0827 [103]	1137	+15/-13	API only
GPT-4o-2024-08-06 [90]	1131	+18/-20	API only
Gemini-1.5-Flash-8B-001 [103]	1133	+10/-15	API only
GPT-4o-mini-2024-07-18 [89]	1124	+7/-9	API only
Molmo-72B	1115	+18/-17	Fully Open
Qwen2-VL-72B [111]	1113	+15/-17	Open Weight
InternVL2-26B [104]	1096	+11/-10	Open Weight
Pixtral-12B-2409 [3]	1085	+13/-14	Open Weight
Llama-3.2V-90B-Instruct [5]	1085	+12/-14	Open Weight
Gemini-1.5-Flash-001 [103]	1087	+8/-8	API only
Molmo-7B-D	1076	+15/-18	Fully Open
Yi-Vision [4]	1070	+21/-26	Distilled
Claude 3 Opus [7]	1073	+6/-8	API only
Qwen2-VL-7B [111]	1068	+15/-14	Open Weight
Llama-3.2V-11B-Instruct [5]	1061	+14/-14	Open Weight

Table 9. **Chatbot Arena’s vision leaderboard** for English queries. The table is up to date as of Nov. 13, 2024. We show up to 20 rows for clarity.

Clock Movies, a newly collected dataset based on the film *The Clock* (2010).⁷ They are highly out-of-distribution relative to the PixMo-Clocks training data. We also benchmarked several API-only models and open-weight (+ open-data) models for comparison. We compare all of these VLMs against the model presented from [121] that is specialized at the single task of clock reading.

We used the same query for all the VLMs: “What time is being shown? Please respond only with the time as hours and minutes in HH:MM format.”, and followed the official evaluation protocol.⁸ Table 10 highlights that all the VLMs including proprietary models struggle to read clocks, with the exception of Molmo; see the notable performance gaps between Molmo and other VLMs. Molmo-72B surprisingly underperforms Molmo-7B-D and MolmoE-1B. This might be partially due to that PixMo-Clocks accounts only for 5.3% of the fine-tuning data mixture and we trained Molmo-72B for fewer steps than the others. Augmenting PixMo-Clocks with real-world clock images could potentially increase performance, closing the gap between Molmo and the specialized clock reading model.

Despite training on synthetic data, we qualitatively observe the clocking-reading capabilities generalize effectively to more complex questions and to captioning. An example is in Figure 1 lower right of the main paper.

⁷<https://www.imdb.com/title/tt2008009>

⁸<https://github.com/charigyang/itsabouttime>

model	acc.	hour acc.	min. acc.
GPT-4o-0513 [90]	2.7	14.2	8.6
Gemini 1.5 Pro [103]	0.9	11.6	5.1
Claude-3.5 Sonnet [7]	6.6	22.3	17.5
PaliGemma-mix-3B [10]	6.1	21.0	15.8
Phi3.5-Vision-4B [1]	1.9	12.0	7.6
Qwen2-VL-72B [111]	<u>9.1</u>	<u>24.9</u>	<u>18.4</u>
InternVL2-Llama-3-76B [104]	3.3	16.3	9.9
Pixtral-12B [3]	1.7	11.9	6.7
Llama-3.2V-90B-Instruct [5]	3.4	17.9	10.1
LLaVA-1.5-13B [69]	0.8	11.6	5.7
xGen-MM-interleave-4B [119]	2.0	11.9	8.0
Cambrian-1-34B [106]	1.8	11.1	7.2
LLaVA OneVision-72B [59]	5.7	17.9	15.4
MolmoE-1B	65.8	77.9	74.1
Molmo-7B-O	64.2	76.3	73.8
Molmo-7B-D	68.2	78.6	76.0
Molmo-7B	65.6	77.1	73.7
Specialized single-task model [121]	78.9	84.2	82.9

Table 10. **Clock reading benchmark results**. We report the averages of overall, hour and minute accuracies, each evaluated on three different test sets based on COCO, OpenImages and Clock Movies, respectively. **Bold** numbers represent the highest VLM scores while the best numbers, excluding Molmo, are underlined. We categorize models into five groups: (first) API-only, (second) open-weight, (third) open-weight and open-data, (four) the Molmo family and (five) the specialized clock reading model.

Pointing. To evaluate the model’s pointing performance, we constructed an evaluation set of 493 image and pointing question pairs. Each example was manually verified to ensure that either there is no target object or each target object instance is annotated with a single point and an accurate segmentation mask. The segmentation masks are generated by SAM [51] using each ground-truth point as a prompt.

For cases with no target object, precision and recall are calculated as 1 if the model responds correctly (*e.g.* outputs “This isn’t in the image.”) and 0 otherwise. When a target object is present, we first compute the pairwise distances between the predicted points and the ground-truth points to serve as the cost in the Jonker-Volgenant algorithm [25, 45], which we then use to assign each predicted point to one of the ground-truth points. We then use the verified segmentation masks to determine if each predicted point with an assignment is a true positive or false positive. Specifically, we calculate precision as the fraction of predicted points located within the segmentation mask of their assigned ground-truth point, and recall as the fraction of segmentation masks covered by predicted points.

Table 11 demonstrates Molmo’s superior pointing capability. Similar to captioning and counting, pointing performance declines when the number of crops are unequal at training and test time.

High-resolution fine-tuning. In Table 2b of the main paper, we showed that training the model with higher resolu-

model	precision	recall	F_1
MolmoE-1B	73.0	72.9	72.2
Molmo-7B-O	75.7	75.5	75.1
Molmo-7B-D	75.0	74.6	74.3
Molmo-7B-D (36 crops)	58.4	58.7	58.1
Molmo-72B	75.8	75.4	75.2

Table 11. **Pointing evaluation results.** Pointing can perform poorly when # of crops are unequal at training and test time.

# crops train, test	CountBenchQA	Pixmo-Count val	11-avg
12, 36*	87.7	73.9	75.8
12, 36	88.5	85.2	76.9
36, 36	88.9	87.4	77.2
12 → 36, 36	88.9	87.4	77.2

Table 12. **High-resolution fine-tuning results.** Result on counting datasets (CountBenchQA and PixMo-Count val set) and the overall average (11-avg) using different numbers of crops at train and test. Note: 12, 36* uses a higher number of crops (36 crops) at test time for counting datasets, which leads to a much worse accuracy. Our default setting (highlighted in gray) uses the same number of crops (12 crops) during training and inference for counting datasets. We experiment with fine-tuning the 12-crop model at higher resolution and evaluating with 36 crops (12 → 36, 36).

tion (*i.e.*, more image crops) yields slight improvements on the 11-avg metric (from 76.9 to 77.2 when increasing the number of crops used in training from 12 to 36). Rather than directly training at a higher resolution, we explore fine-tuning the model initially trained with 12 crops using a higher resolution. Specifically, we continue training the 12-crop model for 3000 additional steps (10% of the fine-tuning steps) with 36 crops, roughly halving the learning rates of the vision encoder ($lr=2e-6$), connector ($lr=2e-6$), and language model ($lr=5e-6$). We keep the global batch size at 256 and use a warmup of 200 steps for all modules.

Table 12 presents the results. Note that simply increasing the number of crops at inference time (first row) leads to degraded performance on counting tasks ($88.5 \rightarrow 87.7$ for CountBenchQA and $85.2 \rightarrow 73.9$ for PixMo-Count). This suggests that a mismatch between training and testing resolutions adversely affects counting performance. As a result, our default model (second row) uses the same number of crops (12 crops) for counting datasets.

After fine-tuning the model with higher resolution (fourth row), we observe that its counting performance can recover when evaluated with 36 crops, matching that of the model trained directly with 36 crops (third row), without sacrificing the overall 11-avg performance. This demonstrates that a brief period of high-resolution fine-tuning can effectively restore counting capabilities without affecting the average performance.

Text-only benchmarks. PixMo consists exclusively of multimodal image-text data, without any text-only data. To

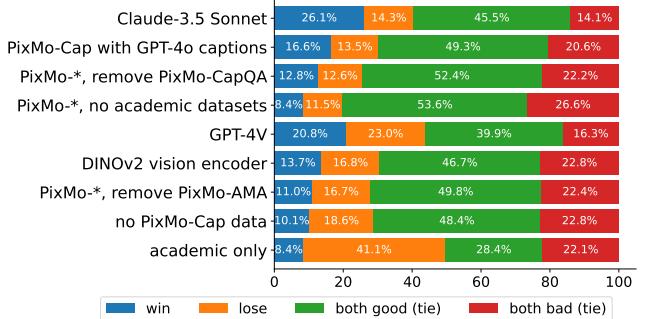


Figure 8. **Human evaluation outcomes for matches between various models vs. Molmo-7B-D.** We expand upon the win rate (excluding ties) shown in Table 5 to report the full breakdown of wins, losses, ties (both good), and ties (both bad). We removed the *I don't know* responses, which accounted for 2.9% of all human feedback, before calculating the outcome rates.

investigate the potential impact of training solely on multi-modal data on performance in text-only tasks, we report the results on common text benchmarks which assess a wide range of capabilities. We carefully follow the setup used by Llama 3 [5] for each task, ensuring that we can reproduce their numbers within the reported confidence intervals. As shown in Table 13, the Qwen2 language model employed in Molmo-7B-D appears to lose some knowledge across various tasks as a result of multimodal fine-tuning.

We run a small experiment at the 7B scale to test whether adding text-only data from Tulu 3⁹ [113] to our fine-tuning data mixture can address this issue. We use two different ratios: the entire dataset and a version with 10% down-sampling. Incorporating this text data enhances model performance on text-only tasks, particularly those involving mathematical reasoning and programming. Interestingly, down-sampling to 10% of the data leads to better results on most text-only tasks and improves the average performance across the 11 multimodal academic benchmarks.

Human evaluation. Table 5 of the main paper reports the win rates of several ablation and API-only models *vs.* Molmo-7B-D *when ties are excluded* (a standard metric reported in the LMSYS Chatbot Arena). Ties make up a significant portion of the matches, so we report the full breakdown of match outcomes in Figure 8 to better characterize the human evaluation. For example, when paired against Claude-3.5 Sonnet 45.5% of matches resulted in a tie where both responses were good, 14.1% in a tie where both responses were bad, Claude won 26.1% of the time, and Molmo-7B-D won 14.3% of the time. As a second example, compared to Molmo fine-tuned only on academic data the breakdown is 28.4% (ties, both good), 22.1% (ties, both bad), 8.4% (it wins), 41.1% (Molmo-7B-D wins).

AI2D with opaque boxes. Table 14 shows result with and

⁹<https://allenai.org/papers/tulu-3-report.pdf>

model	MMLU [38]	MMLU-Pro [114]	GSM-8k [24]	MATH [39]	ARC-C [23]	HumanEval [16]	11-avg
Qwen2-7B (language model)	70.2	42.1	71.8	40.3	87.5	47.6	-
Molmo-7B-D	64.6	32.2	58.8	11.5	81.5	36.6	76.9
Molmo-7B-D + Tulu 3	64.9	38.6	67.7	8.3	84.5	51.2	76.9
Molmo-7B-D + Tulu 3 \times 0.1 down-sample	65.4	37.3	71.2	27.5	84.9	55.5	77.1

Table 13. **Text-only benchmark results.** 11-avg denotes the average performance on 11 academic benchmarks.

model	opaque	transparent
MolmoE-1B	75.7	86.4
Molmo-7B-O	79.8	90.7
Molmo-7B-D	82.4	93.2
Molmo-72B	86.4	96.3

Table 14. AI2D test scores with transparent and opaque boxes.

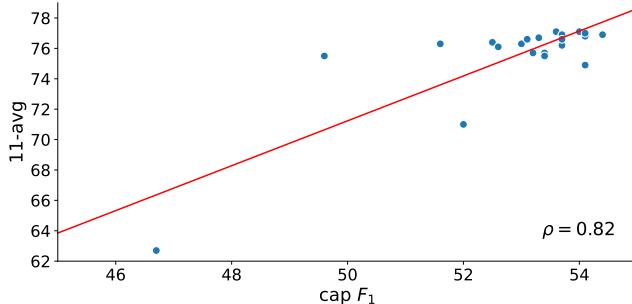


Figure 9. **Relationship between cap F_1 and 11-avg.** Our model development was driven by increasing cap F_1 . Here, we show a scatter plot of cap F_1 vs. the 11 benchmark average (11-avg) from 22 ablation experiments, including all ablations that: (1) affect pre-training and (2) use PixMo-Cap. The Pearson correlation (ρ) is 0.82 and a least-squares regression line is shown in red.

without opaque boxes on AI2D. The two options are described and discussed in Section B.2.

Cap F_1 and 11-avg correlation. For the majority of the project we did not look at downstream tasks¹⁰ and instead made most modeling decisions to maximize our captioning metric. At the conclusion of the project, we used our ablation experiments to analyze the relationship between cap F_1 and the 11 benchmark average (11-avg), shown in Figure 9. The scatter plot includes results from the 22 experiments that meet two conditions: (1) the experiment affects pre-training and (2) the experiment uses PixMo-Cap. We exclude a small number of experiments that use different pre-training data, *e.g.* ShareGPT4o/v, because cap F_1 becomes an out-of-domain evaluation that is not directly comparable to in-domain results. We observe a strong correlation (Pearson $\rho = 0.82$), suggesting that optimizing for dense captioning may be a reasonable proxy for a broad range of downstream tasks—though we have not established a causal relationship and this suggestion should be taken with that caveat in mind.

Leaderboards. We submitted Molmo-72B-D to sev-

¹⁰We did a small number of sanity checks on VQA v2.0.

eral leaderboards. Molmo-72B-D achieves first on the VQA v2.0 leaderboard on the A-OKVQA leaderboard, achieves third on DocQA and InfoQA, behind QwenVL-72B, InternVL2-Pro.¹¹

E. Ablations Details

E.1. Discussion of Main Paper Ablations

Vision encoder. We adopted OpenAI’s CLIP early on and used it for our main results and as the default in our ablations. Later, we evaluated the three alternative choices in Table 2a. All encoders are ViT-L/14 with 336×336 pixel inputs, except for SigLIP which uses 384×384 pixels. For MetaCLIP, we started with the weights of the 224×224 model and resized the positional embeddings to 336×336 before using it in Molmo. To equalize computation, we slightly reduced the maximum number of crops for SigLIP so that the average vision token count is similar for all models. Overall, the three encoders that were trained on web-scale noisy image-text data perform very similar to each other on both metrics. Of significant note, this includes MetaCLIP which is a *fully open* model (data and weights) meaning that *every model component and every bit of data in a Molmo model equipped with MetaCLIP and OLMO is open*. In retrospect, we should have used MetaCLIP as our default vision encoder, but we evaluated it too late in the process to retrain all Molmo models and ablations that were already based on OpenAI’s CLIP vision encoder.

Also, surprisingly, when using the DINOv2 backbone—which is trained on images only (no text, no label supervision)—Molmo performs only slightly worse than the vision-language supervised vision encoders. DINOv2 also performs well in our user study (Table 5), with a win-rate (excluding ties) of 45% compared to our standard Molmo-7B-D configuration (*i.e.*, Molmo-7B-D wins 55% of all non-tie matches against its DINOv2-based variant).

Image resolution. Table 2b shows that using more crops (and thus higher image resolution) for training and testing generally improves results. We found that some tasks, like document-heavy ones, even benefit from using more crops in inference than the number used for training. However, for captioning and pointing (and thus counting), results degrade when the number of test crops does not equal the number of training crops. Therefore, for captioning and counting tasks we always force these values to be the same. As shown in

¹¹Results as of Nov. 21, 2024.

Table 12 this awkward detail can be remediated by a small amount of high-resolution fine-tuning and then always using that same number of crops during inference for all tasks.

Dropout. Dropout in the LLM generally improves both pre-training and fine-tuning (Table 2c). We also find that our novel text-token-only dropout, in which dropout is only applied to the text tokens of the caption, not to the vision or prompt tokens, improves the captioning metric. We hypothesize that this restricted dropout encourages the model to rely more on the vision tokens, rather than guess based on the previous text tokens, when generating tokens which may reduce hallucinations.

Length conditioning. Our captioning pre-training task includes a length hint. In Table 2e we ablate this design choice and find that it significantly impacts the captioning metric, but also improves the downstream tasks. Note that length conditioning only changes the pre-training task; the fact that it improves the downstream metrics indicates that captioning with length conditioning is a better pre-training task than just captioning.

PixMo-Cap scaling. In Table 3a we show the scaling effects of PixMo-Cap data by training with smaller fractions of the data in both pre-training and as part of the fine-tuning data mixture. Both metrics clearly improve as the amount of captioning data varies from none at all to the full set of 712k images. We also tested the model with no PixMo-Cap data in our user study (Table 5), where it had a win-rate (excluding ties) of only 35% compared to our standard Molmo-7B-D configuration. Removing PixMo-Cap data has a severe negative impact on its user preference score.

Pre-training data. We consider different choices of pre-training data in Table 3b. Pre-training VLMs, not just the vision encoder, with web-scale noisy image-text data is a popular data choice in contemporary methods (*e.g.*, [5, 10]). We test if this has any advantage using data from LAION [98]. To do this we add a preliminary training stage that tunes the model for 50k steps and a batch size of 1024 on image/text pairs from LAION 2B. In this stage only the V/L connector is tuned, the LLM and image encoder are frozen. This pre-trained model is then trained on the dense captions and then our instruction tuning mixture as normal. We find no improvement in metrics using this strategy, allowing us to keep the training pipeline simple.

Another popular choice is to use ShareGPT4V/o [15], which involves distilling from GPT-4 through captions. Using this data instead of PixMo-Cap performs worse on both metrics even when approximately controlling for the data scale (compare to 178k PixMo-Cap images in Table 3a). In contrast, if we caption all PixMo-Cap images with GPT-4o and train on those captions, both metrics perform strongly. We think this is likely because PixMo-Cap has a more diverse image distribution, and due to captioning improvements in GPT-4o. Finally, we compare our default setting

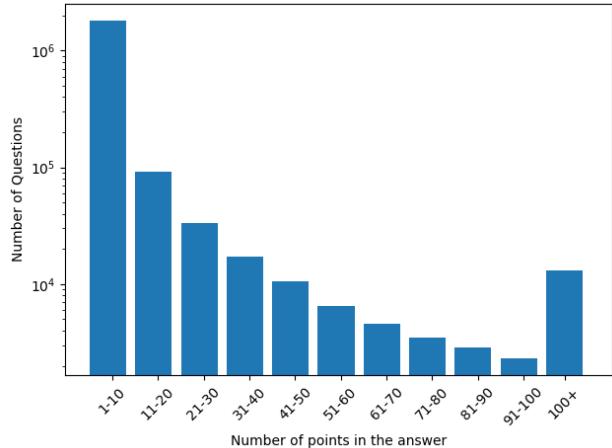


Figure 10. **PixMo-Points distribution of counts.** We show the number of pointing questions (on a log scale) with answers in different ranges (*e.g.*, 1 to 10, 11 to 20, *etc.*).

to either using only the raw audio transcripts or only using the LLM cleaned transcripts, both of which perform slightly worse than our default strategy of using both.

Supervised fine-tuning data. We explore choices of fine-tuning data in Table 3c. Using only academic data sets (specifically the ones in Fig. 4, but excluding Android-Control) performs significantly worse than our full mixture (72.2% *vs.* 76.8%). The gap is primarily explained by PixMo-Docs, which improves results on document-heavy tasks, and the counting data from PixMo-Points and PixMo-Count. The other fine-tuning PixMo datasets have a small, and sometimes slightly negative, impact on the 11 benchmarks; they primarily add new skills to the model and improve user experience when chatting with it, as shown by user preference scores in Table 5.

Counting. In Table 4d we compare encoding points in plain-text as numbers between 0.0 and 100.0 with one significant digit of precision (our default) *vs.* adding 1000 special point tokens to the model’s tokenizer, maintaining the same spatial precision. We find that using special point tokens performs substantially worse than the simple plain-text representation.

E.2. Additional Ablations

Additional model ablations are presented in Table 15 for vision encoder layers, learning rate warmup, and gradient normalization. See table captions for more details.

F. Data Details

PixMo-Points. The PixMo-Points dataset has a total of 229k unique images and a total of 1.98M referring expressions. It has an average of 8.7 distinct expressions per image with an average 5.5 points per expression, and an average of 47.7 total points per image. Additionally, there are

layers	cap	F_1	11-avg	steps (ViT / con. / LLM)	cap	F_1	11-avg	grad norm	cap	F_1	11-avg
3rd-to-last & 10th-to-last	54.1	76.9		2000 / 200 / 2000	54.1	76.9		component-wise	54.1	76.9	
only 3rd-to-last	53.7	76.6		200 / 200 / 200	53.7	76.9		global	53.6	76.9	
only 10th-to-last	52.5	76.3						global, fine-tune only	54.1	76.9	

(a) **Vision encoder layers.** Concatenating features from multiple ViT layers (near the end and towards the middle) *vs.* using features only from each of those layers.

(b) **Learning rate warmup.** The number of learning rate warmup steps used in caption pre-training for the three principal model components: ViT, connector, and LLM.

(c) **Grad norm.** Gradient normalization applied separately to the ViT, connector, and LLM parameters, globally to all parameters (row 2), or component-wise for pre-training and globally for fine-tuning (row 3).

Table 15. **Additional model ablations.** Defaults are in gray.

359k instances with no target object (no points). Figure 10 shows the distribution of number of points for expressions with non-zero points. PixMo-Points is a much larger and more diverse dataset than previous works such as gRefCOCO [66] (which contains a total of 20k images, 60k distinct instances, 278k expressions, of which 80k are multi-target and 32k are no-target expressions) and also much larger than RefCOCO, RefCOCOg and RefCOCO+ [126], each with about 86k, 142k and 141k unique referring expressions respectively and no multi-target references. Additionally, PixMo-Points focuses on referring to points and not segmentation masks, making it significantly more efficient to collect.

PixMo-Cap. We prompted our annotators with the following questions to answer in their spoken image descriptions.

1. *What is the image at first glance?*
2. *What are the objects and their counts?*
3. *What does the text say?*
4. *What are the positions of the objects?*
5. *What subtle details are noticeable?*
6. *What is in the background?*
7. *What is the style and color?*

PixMo-Docs. We developed a generation framework for synthesizing text- and figure-heavy images. The core idea is to harness the coding capabilities of a text-only LLM to generate programs that render image data. These programs are then used as context for another LLM to construct instruction-tuning datasets.

Our framework supports seven programming languages/rendering libraries, including Matplotlib, Plotly, LaTeX, HTML, Vega-Lite, Mermaid, and Graphviz. Using these tools, we designed specialized pipelines to generate charts, tables, diagrams, and various types of documents.

The framework accepts text input to control the generation process. For instance, given the input “restaurant menu”, the system selects the appropriate tools to generate relevant data. To diversify the final datasets, we use a comprehensive set of input queries. Additionally, we enhance data diversity by incorporating personas [35], which control the content and style of the synthetic data. For example, when generating “restaurant menu” data with the persona “A barbecue enthusiast known for their amazing grilled food at every Tennessee Vols game”, the framework

produces a data point featuring a “Southern fusion menu combining traditional BBQ with international flavors, presented on a wooden board background”. This approach allows us to enrich the variety within each category of synthetic data.

We use Claude-3.5 Sonnet [7] for code generation and GPT-4o-mini [89] during the instruction-tuning data generation stage, prioritizing cost efficiency.

G. Dataset Examples

We include **randomly selected** examples from the **PixMo-*** datasets. Prompts are shown in bold, and points are shown with pink dots.

- PixMo-Cap - Figure 12
- PixMo-AskModelAnything- Figure 13
- PixMo-Points - Figure 14
- PixMo-Points with explanations - Figure 15
- PixMo-CapQA - Figure 16
- PixMo-Clocks - Figure 17
- PixMo-Count - Figure 18
- PixMo-Docs (charts) - Figure 19
- PixMo-Docs (tables) - Figure 20
- PixMo-Docs (diagrams) - Figure 21
- PixMo-Docs (other) - Figure 22

H. Related Work

Vision-language contrastive models. Vision-language models have become popular in the last few years. Models such as CLIP [96] and ALIGN [43] that are trained on noisy web data provide strong language-aligned image encoders and perform well on downstream classification and image-text retrieval tasks, without any task specific tuning. Previous works [33, 115] proposed similar ideas before transformers [109] became popular. Since CLIP was released, other works have focused on making the CLIP pipeline fully open [20, 118]. However, vision encoders trained with noisy web data have limitations in discerning details, as discussed in [107].

Multimodal LLMs. Multimodal LLMs often use CLIP-style image encoders and align image embeddings with the LLM input space via a connector module [26, 41, 59, 69, 71, 85, 92, 105, 112, 131]. Some works have also explored

Category	Model	VLM		LLM Backbone		Vision Encoder	
		Open Weights	Open Data + Code	Open Weights	Open Data + Code	Open Weights	Open Data + Code
Molmo	Molmo-72B	Open	Open	Open	Closed	Open	Closed
	Molmo-7B-D	Open	Open	Open	Closed	Open	Closed
	Molmo-7B-O	Open	Open	Open	Open	Open	Closed
	MolmoE-1B	Open	Open	Open	Open	Open	Closed
API Models	GPT-4o	Closed	Closed	Closed	Closed	Closed	Closed
	GPT-4V	Closed	Closed	Closed	Closed	Closed	Closed
	Gemini 1.5 Pro	Closed	Closed	Closed	Closed	Closed	Closed
	Gemini 1.5 Flash	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3.5 Sonnet	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3 Opus	Closed	Closed	Closed	Closed	Closed	Closed
	Claude 3 Haiku	Closed	Closed	Closed	Closed	Closed	Closed
	Qwen VL2 72B	Open	Closed	Open	Closed	Open	Closed
Open Weights	Qwen VL2 7B	Open	Closed	Open	Closed	Open	Closed
	Intern VL2 LLAMA 76B	Open	Closed	Open	Closed	Open	Closed
	Intern VL2 8B	Open	Closed	Open	Closed	Open	Closed
	Pixtral 12B	Open	Closed	Open	Closed	Open	Closed
	Phi3.5-Vision 4B	Open	Closed	Open	Closed	Open	Closed
	PaliGemma 3B	Open	Closed	Open	Closed	Open	Closed
	LLAVA OneVision 72B	Open	Distilled	Open	Closed	Open	Closed
	LLAVA OneVision 7B	Open	Distilled	Open	Closed	Open	Closed
Open Weights & Data	Cambrian-134B	Open	Distilled	Open	Closed	Open	Closed
	Cambrian-18B	Open	Distilled	Open	Closed	Open	Closed
	xGen - MM - Interleave 4B	Open	Distilled	Open	Closed	Open	Closed
	LLAVA-1.5 13B	Open	Open	Open	Closed	Open	Closed
	LLAVA-1.5 7B	Open	Open	Open	Closed	Open	Closed

Figure 11. VLM Openness Comparison. We characterize the openness of VLMs based on two attributes (open weights, open data and code) across three model components (the VLM and its two pre-trained components, the LLM backbone and the vision encoder). In addition to open vs. closed, we use the "distilled" label to indicate that the data used to train the VLM includes images and text generated by a different, proprietary VLM, meaning that the model cannot be reproduced without a dependency on the proprietary VLM.

using multiple image encoders in tandem with CLIP-style encoders [74, 106], such as using self-supervised learning (SSL) encoders [91]. Many works use a pre-training stage for just the connector weights [12, 34, 69, 106] while others do not have an explicit connector training stage [10, 18, 48, 67]. In contrast, two other common architecture strategies are (1) directly connecting the image embedding to different LLM layer embeddings via cross-attention [5, 6, 58, 134] and (2) removing the image encoder and directly inputting the pixels [9, 56]. The cross-attention

design naturally allows for the introduction of a large number of new parameters, which enables freezing the LLM while still training an effective VLM. This approach has the advantage of maintaining text-only task performance (*cf.* Table 13). Due to the compute constraints for training and inference of these models, there has also been a rise in efficient multimodal LLMs [14, 22, 64, 79, 123, 128, 137].

The best performing multimodal LLMs [7, 90, 103] are proprietary closed source models. While they are very capable, not much is known about how these models are

trained and what data they use. In contrast, many works release their model weights [1, 3, 5, 10, 111] but don’t release their training recipes or don’t disclose all the data used. Other works provide all the training details and data [54, 59, 106, 119, 133], but use data generated by proprietary VLMs such as [15]. Hence, there is a need for a fully open SoTA training pipeline that does not use previously trained multimodal LLMs to generate data.

Vision-language instruction tuning datasets. The rise in popularity of VLMs has also led to a rise of methods to build visual instruction-tuning data. A common approach is to annotate an image with vision models (or use ground-truth annotations), and then use a LLM to generate QA pairs [57, 69, 125, 137] from those annotations. However, these approaches are limited since the automatically generated annotations can be noisy, and even ground-truth annotations often do not comprehensively describe all the details in the image. PixMo-CapQA takes a similar approach but uses the detailed captions from PixMo-Cap which provide more comprehensive image descriptions. Many recent methods have used proprietary VLMs to annotate images directly [13, 15, 68, 71, 110], which is effective but makes the training pipeline dependent on a closed source VLM.

It is also very common to pair templated instructions with existing annotated datasets to build instruction tuning data (e.g., [5, 44, 57, 75]). While Molmo also uses academic datasets, we prefer style tags over natural language instructions since we believe our data, and in particular PixMo-AskModelAnything, provides better training for conversational user interactions.

Our approach to having annotators work with a LLM when generating QA pairs is similar to the approach in [80], but we extend this idea to image/language data.

Synthetic vision-language datasets. Prior approaches to synthetic chart generation typically only support one or two types of charts [46, 47, 86], often with a heavy focus on bar charts or line plots. PixMo-Docs uses code as the text-only representation for the LLM which lets us support much more diverse formats, including heat-maps, violin plots, chord diagrams, geographic plots, tree maps among many others. Our use of HTML for document generation is also similar to [55], however we consider many additional approaches to representing documents besides HTML.

Synthetic clock data has been considered [121]. Our approach uses real watch faces instead of rendering clocks purely from a simulator, which gives our synthetic data more diversity (e.g., watches with no second hand, stylized decorations or coloring, background images, a separate inner piece to show seconds). Combining these two datasets might yield additional improvements.

VLM grounding. Multimodal LLMs that support grounding language in an image are becoming more common [61, 92, 94, 97, 117, 127, 132]. These works commonly use au-

tomated object detectors and/or existing referring expression datasets [52, 66, 126] for training data. Of these datasets, GRES [66] is most similar to PixMo-Points in that it is human-annotated and includes arbitrary expressions (not just object categories), none-present annotations, and allows expressions to refer to multiple object in the image. However, it only grounds a limited category of objects (e.g., only COCO categories), and rarely grounds expressions in a large numbers of objects in a single image. For PixMo-Points we source a diverse set of images and collect points from human annotators. We annotate points instead of segmentation masks, which enables us to collect 1.98M unique referring expression instances with an average of 5.5 points per expression.

Bootstrapping from LLMs. Closed text-only LLMs are commonly used for data generation and curation [69]. We also made use of closed text-only LLMs when building several of the PixMo datasets. Given our stance against using VLMs for building datasets, it’s worth justifying the use of closed LLMs. It is true using closed LLMs means the current data pipeline is not entirely open. However, once *open* LLMs (e.g., [37]) become sufficiently good they can be used in place of closed ones to build a dataset functionally equivalent to PixMo. Our philosophy is that we should not wait for open LLM research to achieve this goal and instead we should pursue research on building open VLMs *in parallel*. We note that using one VLM to build another VLM is entirely different than using an LLM, because the dependency is circular and therefore cannot result in a fully open system at a later point in time.

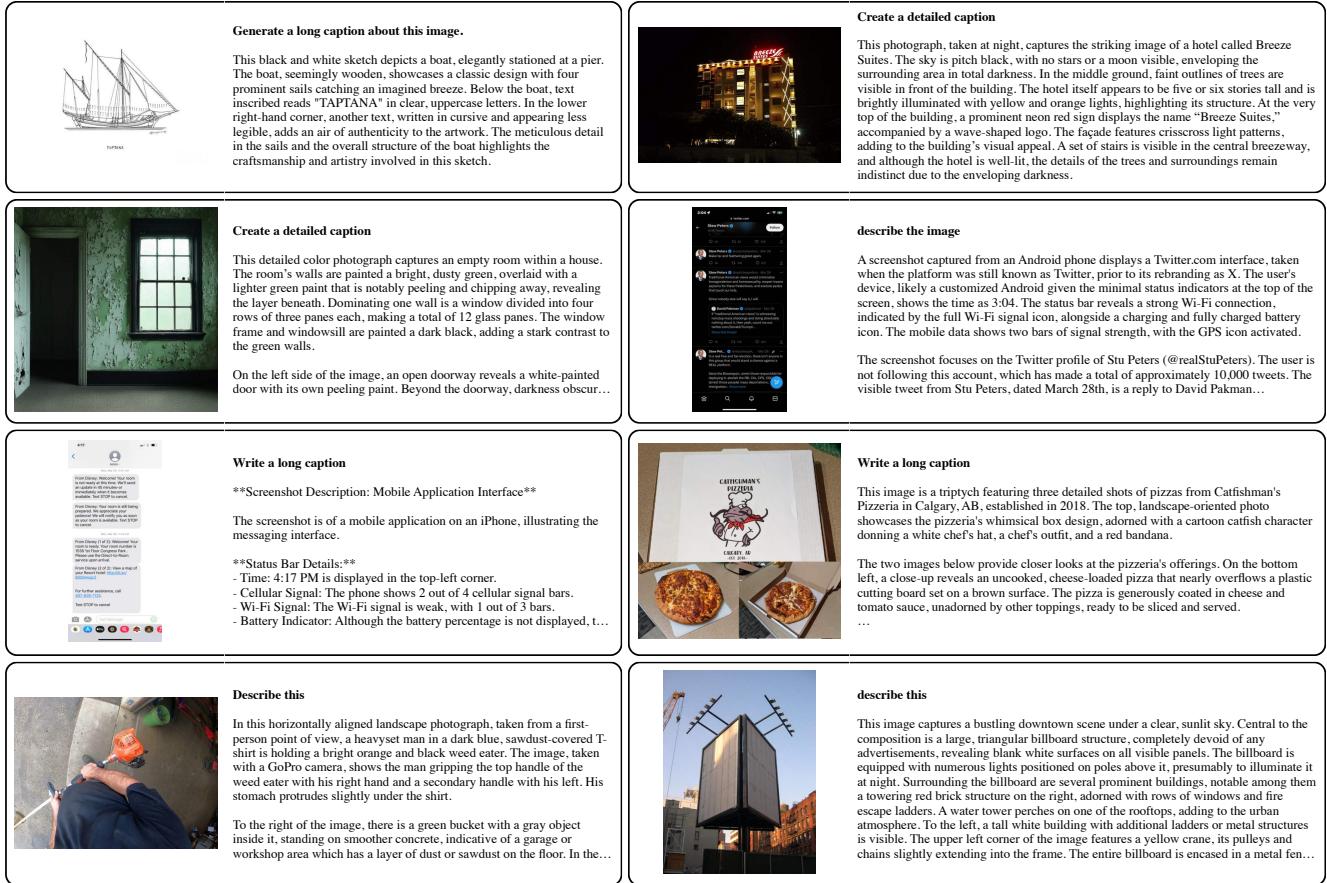


Figure 12. Randomly selected examples from PixMo-Cap with our prompt templates.

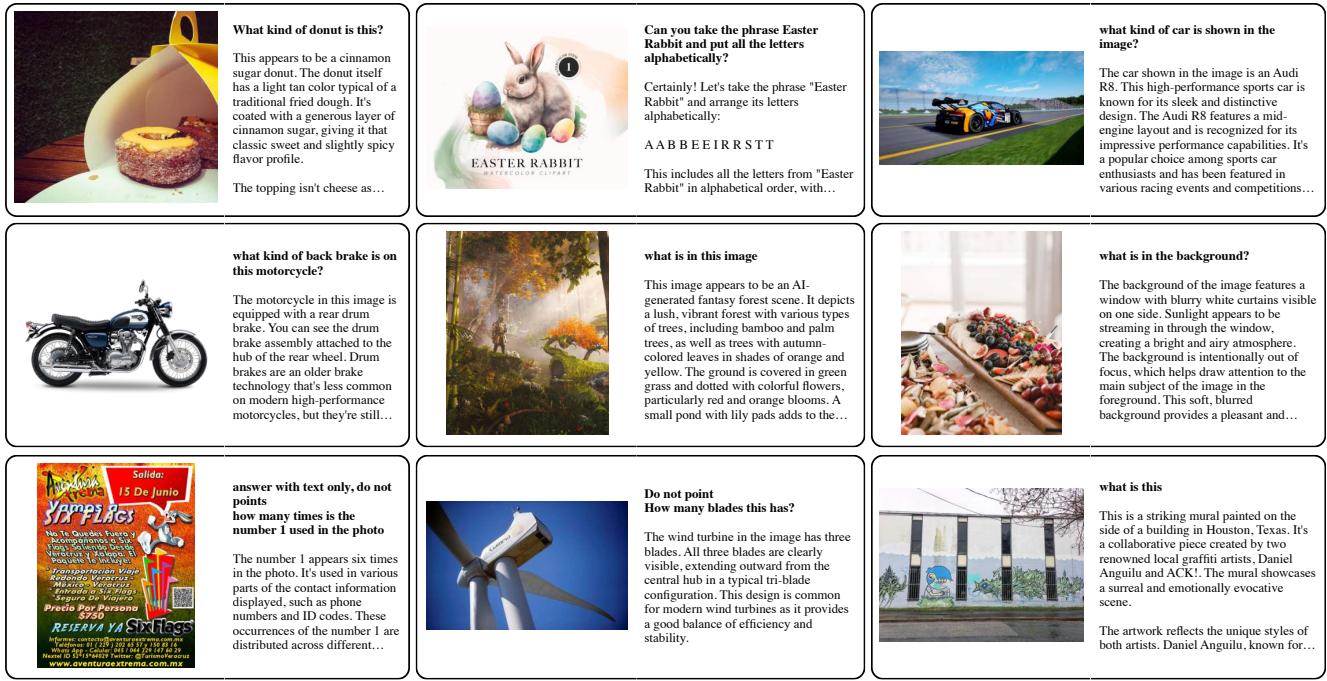


Figure 13. Randomly selected examples from PixMo-AskModelAnything.

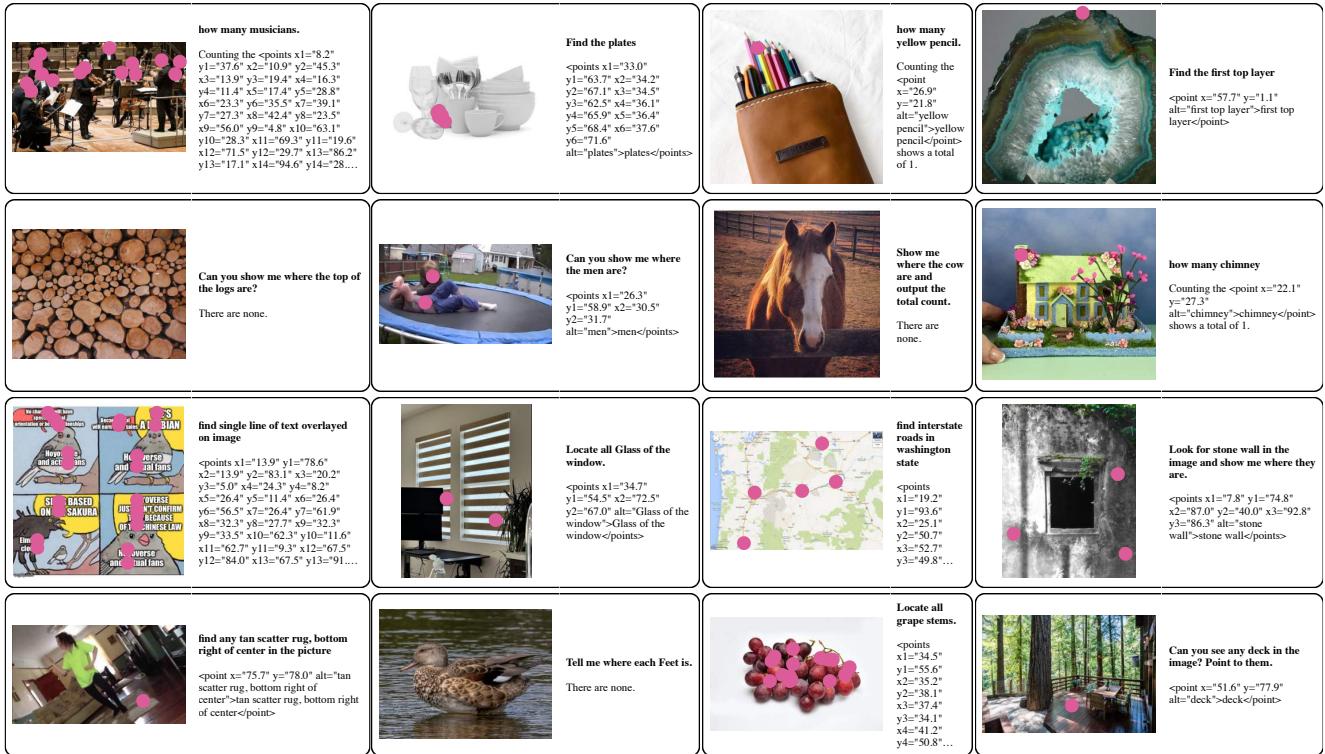


Figure 14. Randomly selected examples from **PixMo-Points**. Even when text has been cut off, all points still appear in the image. Our templated prompts can be ungrammatical for some of these options, but we find they are still sufficient to let the model respond correctly to natural language instructions.

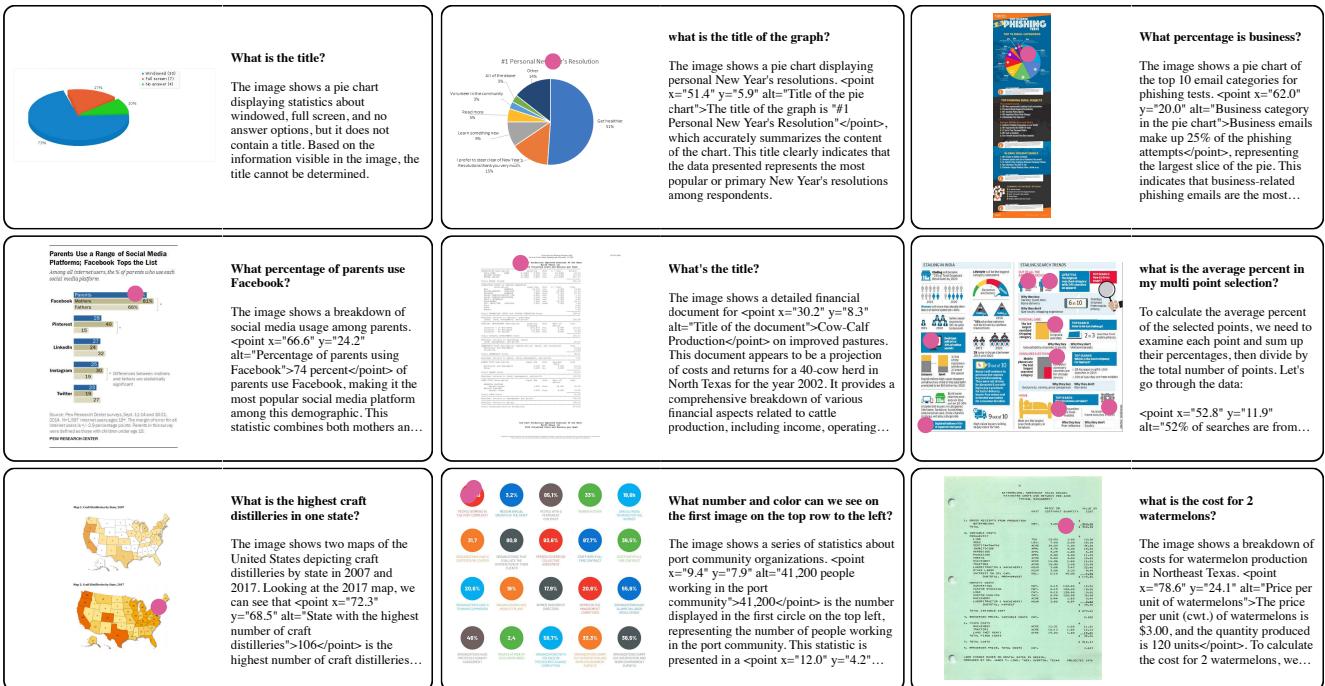


Figure 15. Randomly selected examples from the experimental **PixMo-Points** data that includes points with explanations.

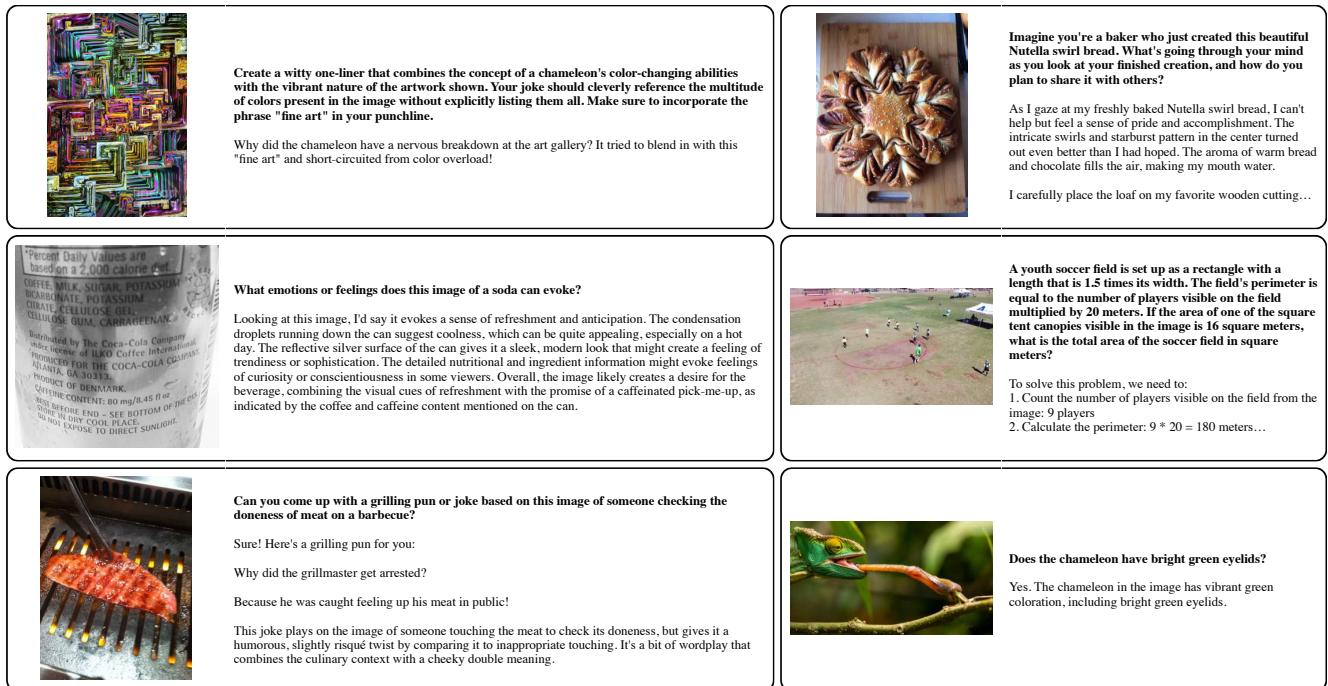


Figure 16. Randomly selected examples from the synthetic **PixMo-CapQA** data generated from PixMo-Cap.

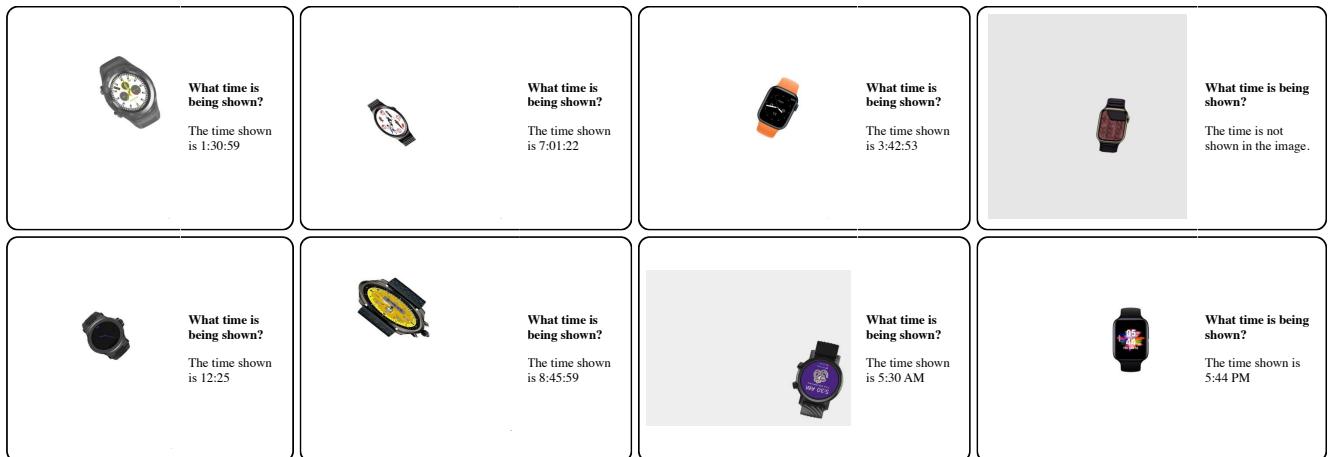


Figure 17. Randomly selected examples from the synthetic **PixMo-Clocks** data after our data augmentation.



Figure 18. Randomly selected examples from the synthetic **PixMo-Count** data.

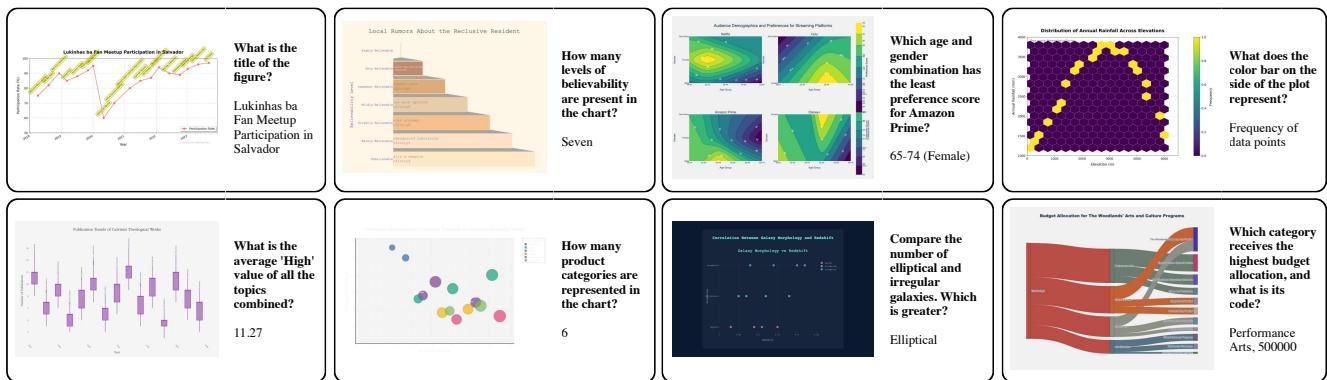


Figure 19. Randomly selected chart examples from the synthetic PixMo-Docs data.

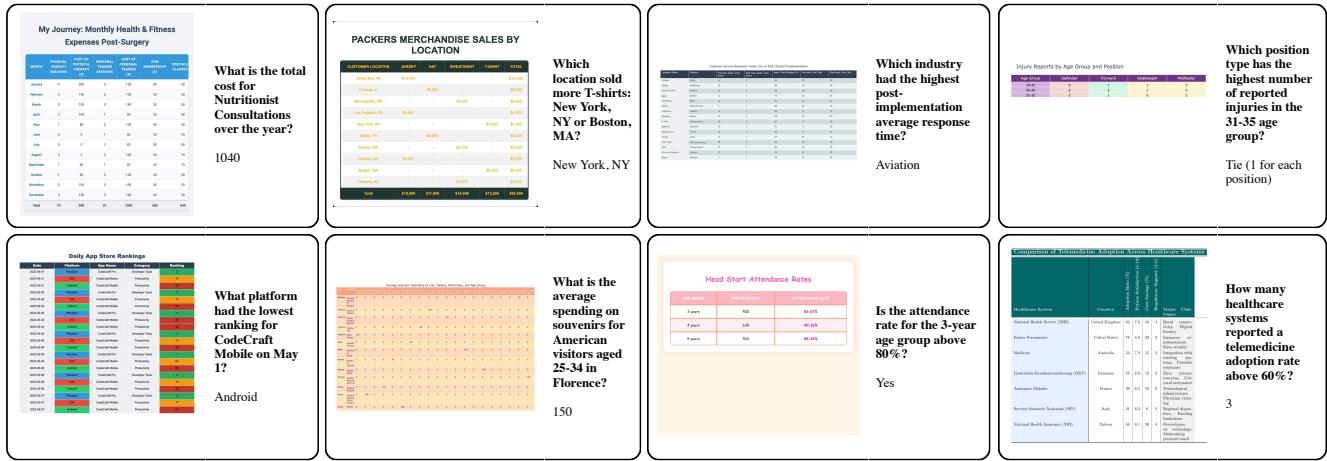


Figure 20. Randomly selected table examples from the synthetic PixMo-Docs data.

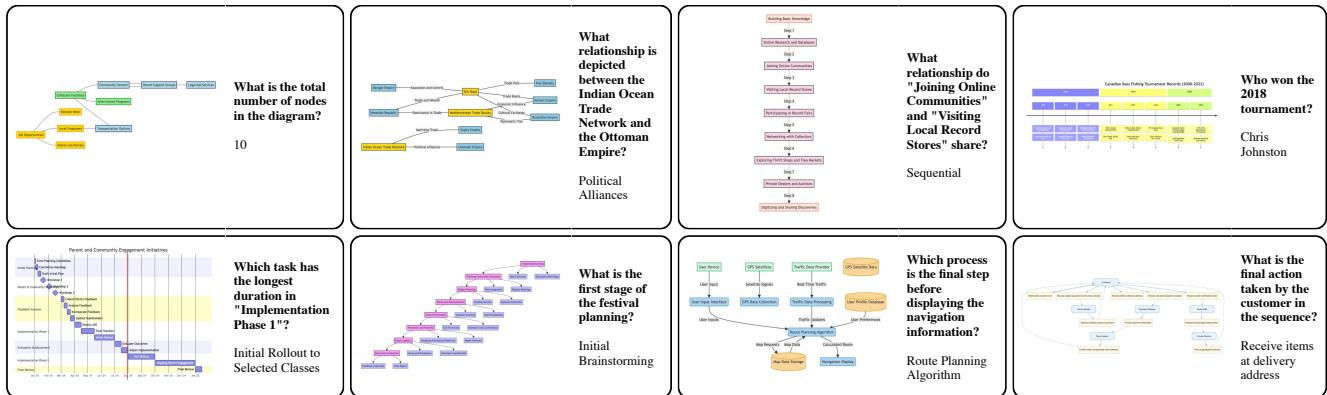


Figure 21. Randomly selected diagram examples from the synthetic PixMo-Docs data.

<p>Employee Performance Review Summary for 2021</p> <p>Key Performance Indicators (KPIs):</p> <ul style="list-style-type: none"> High Priority: Lead development of new software product, achieved 10% market share in Q3. Customer Satisfaction: 12% YoY increase in customer retention rate. Individual Achievements: <ul style="list-style-type: none"> John Doe: Led marketing department's 15% sales increase. Jane Smith: Developed internal communication strategy. Mike Johnson: Streamlined internal processes. <p>Feedback Analysis:</p> <ul style="list-style-type: none"> Appreciation for timely service and work ethic. Recognition for exceptional customer satisfaction. Areas for Improvement: <ul style="list-style-type: none"> Communication Delivery Efficiency <p>Conclusion:</p> <p>Overall, the team exceeded expectations across all dimensions. Our retention rate is 12% higher than our industry average. All key metrics are at or above budget. Individual achievements are commendable, particularly Jane Smith's leadership role.</p>	<p>How many new key accounts did John Smith secure?</p> <p>five</p>	<p>Meal Plan for Sustained Mental Energy</p> <p>Benefits:</p> <ul style="list-style-type: none"> High protein, low-carbohydrate meals to support mental energy. Protein-rich meals help maintain blood sugar levels throughout the day. Complex carbohydrates provide sustained energy. Fiber-rich foods aid in digestion and prevent energy crashes. <p>Nutritional Benefits:</p> <ul style="list-style-type: none"> High protein intake supports cognitive function and memory. Carbohydrates provide sustained energy for mental tasks. Fats, especially healthy fats like omega-3s, support brain health. <p>Lunch:</p> <ul style="list-style-type: none"> Salad with Chicken and Quinoa Grilled Turkey Sandwich with Avocado and Lettuce Quinoa and Black Bean Bowl Salad with Grilled Salmon and Asparagus <p>Breakfast:</p> <ul style="list-style-type: none"> Oatmeal with Fresh Fruits Smoothie with Protein Powder and Berries Avocado Toast with Egg Whites Greek Yogurt with Honey and Nuts <p>Dinner:</p> <ul style="list-style-type: none"> Grilled Chicken and Veggies Stir-Fry with Shrimp and Broccoli Beef and Spinach Stir-Fry Salmon with Asparagus and Rice <p>Snacks:</p> <ul style="list-style-type: none"> Almonds and Hummus Apple Slices with Peanut Butter Carrot Sticks with Hummus Yogurt and Granola <p>Drinks:</p> <ul style="list-style-type: none"> Water Green Tea Smoothies Herbal Teas <p>What are the key components in the lunch section?</p> <p>Complete protein and healthy fats</p>	<p>Scholarship Application Verification Form</p> <p>Applicant Information:</p> <p>Name: John Doe Address: 123 Main Street, Anytown, USA Phone: (555) 123-4567 Email: john.doe@example.com</p> <p>Qualifications:</p> <p>Academic Record: GPA 3.8, SAT 1450, ACT 32 Community Involvement: President of Math Club, Volunteered at local food bank Leadership Experience: Classmate, Team Captain of School Football Team Work Experience: Part-time job at local grocery store</p> <p>Financial Need Assessment:</p> <p>Annual Household Income: \$60,000 Number of Dependents: 2 Total Monthly Expenses: \$2,500 Remaining Funds after Expenses: \$1,000</p> <p>What is the total monthly cost of rent and utilities?</p> <p>€900/month</p>	<p>WILDCARD RESTAURANT RECEIPT</p> <p>Subscriptions:</p> <ul style="list-style-type: none"> Food delivery service: \$100 Local grocery store: \$50 Subscription box: \$20 <p>Food Costs:</p> <ul style="list-style-type: none"> Meat: \$150 Vegetables: \$50 Dairy: \$50 Beverages: \$50 <p>Events and Meetings:</p> <ul style="list-style-type: none"> Business networking event: \$50 Local craft fair: \$20 <p>How much does an ACE Membership cost?</p> <p>\$49</p>																		
<p>Charitable Book Drive Donation Receipt</p> <p>Book Details:</p> <ul style="list-style-type: none"> Title: "The Great Gatsby" Author: F. Scott Fitzgerald Publisher: Penguin Classics Year Published: 1925 Genre: Literary Fiction <p>Charity Receipts:</p> <ul style="list-style-type: none"> Books for Books Foundation: \$100 Local Library: \$50 Community Center: \$20 <p>Acoustic Books:</p> <table border="1"> <thead> <tr> <th>Item</th> <th>Quantity</th> <th>Estimated Value</th> </tr> </thead> <tbody> <tr> <td>Books</td> <td>100</td> <td>\$100</td> </tr> <tr> <td>CDs</td> <td>50</td> <td>\$50</td> </tr> <tr> <td>DVDs</td> <td>30</td> <td>\$30</td> </tr> <tr> <td>Books</td> <td>20</td> <td>\$20</td> </tr> <tr> <td>CDs</td> <td>10</td> <td>\$10</td> </tr> </tbody> </table> <p>Total Estimated Value: \$200</p>	Item	Quantity	Estimated Value	Books	100	\$100	CDs	50	\$50	DVDs	30	\$30	Books	20	\$20	CDs	10	\$10	<p>Are donations tax-deductible?</p> <p>Yes</p>	<p>What is the name of the climate movement?</p> <p>Global Green Alliance</p>	<p>Chalita Suansane</p> <p>Profile: Chalita Suansane is a dynamic performer known for her fluid movements and traditional Thai dance. She has performed in numerous international competitions and festivals.</p> <p>Education: Bachelor of Arts in Performance Arts from Mahidol University, Thailand.</p> <p>Film Roles:</p> <ul style="list-style-type: none"> Chalita Suansane in "The Art of Dance" (2018) Chalita Suansane in "The Hermit Master" (2019) Chalita Suansane in "The Last Dance" (2020) <p>Television Series:</p> <ul style="list-style-type: none"> Chalita Suansane in "Dance Masters" (2017) Chalita Suansane in "Asia's Got Talent" (2018) <p>Awards:</p> <ul style="list-style-type: none"> Best Actress - International Dance Festival Awards, 2018 Best Actress - National Dance Competition, 2019 <p>Moves:</p> <ul style="list-style-type: none"> Chalita Suansane's signature move is the "Kraom Krua." <p>Languages:</p> <p>Thai, English, French</p> <p>What skills does she have?</p> <p>Acting, Dancing, Singing, Traditional Thai performance</p>	<p>National Dance Education Standards and Curriculum Guidelines for Kosovo</p> <p>Introduction:</p> <p>The National Dance Education Standards and Curriculum Guidelines for Kosovo are designed to provide a comprehensive framework for the teaching and learning of dance. They aim to develop students' technical skills, artistic expression, and cultural understanding.</p> <p>Assessment Methods:</p> <ul style="list-style-type: none"> Performance Written Assessments Self-Assessment <p>Core Standards:</p> <ul style="list-style-type: none"> Performance Knowledge and Understanding Skills and Techniques Connections <p>Teacher Qualification:</p> <p>Minimum: Bachelor's degree in a related field; Pedagogical training in the field of dance education; Certification from the relevant authority.</p> <p>Excellence Guidelines:</p> <ul style="list-style-type: none"> Artistic Excellence Technical Excellence Cultural Awareness Community Engagement <p>Implementation Strategy:</p> <ul style="list-style-type: none"> Curriculum Development Teacher Training Student Support Community Outreach <p>Resources:</p> <ul style="list-style-type: none"> National Dance Education Standards and Curriculum Guidelines for Kosovo Curriculum Framework for Dance Education Teacher Guide Student Workbook Performance Portfolio Assessment Rubrics Sample Lessons Professional Development Opportunities Research Papers Guest Lectures Workshops Conferences Competitions Performances <p>What is the preferred qualification for teachers according to the document?</p> <p>Master's degree in Dance Education or Choreography</p>
Item	Quantity	Estimated Value																				
Books	100	\$100																				
CDs	50	\$50																				
DVDs	30	\$30																				
Books	20	\$20																				
CDs	10	\$10																				

Figure 22. Other randomly selected documents from the synthetic PixMo-Docs data.

References

- [1] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024. 6, 14, 20
- [2] Manoj Acharya, Kushal Kafle, and Christopher Kanan. TallyQA: Answering complex counting questions. In *AAAI*, 2019. 5
- [3] Pravesh Agrawal, Szymon Antoniak, Emma Bou Hanna, Devendra Chapolat, Jessica Chudnovsky, Saurabh Garg, Theophile Gervet, Soham Ghosh, Amélie Héliou, Paul Jacob, et al. Pixtral 12b. *arXiv preprint arXiv:2410.07073*, 2024. 6, 14, 20
- [4] 01. AI, :, Alex Young, Bei Chen, Chao Li, Chengan Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, and more. Yi: Open foundation models by 01.ai. *arXiv preprint arXiv:2403.04652*, 2024. 14
- [5] Meta AI. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 6, 12, 14, 15, 17, 19, 20
- [6] 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. In *NeurIPS*, 2022. 19
- [7] Anthropic. The claude 3 model family: Opus, sonnet, haiku, 2024. 3, 6, 14, 18, 19
- [8] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. In *NeurIPS Deep Learning Symposium*, 2016. 10
- [9] Rohan Bavishi, Erich Elsen, Curtis Hawthorne, Maxwell Nye, Augustus Odena, Arushi Somani, and Sağnak Taşırlar. Fuyu-8b: A multimodal architecture for ai agents, 2023. 19
- [10] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bošnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier Henaff, Xi Xiong, Radu Soricu, Jeremiah Harmsen, and Xiaohua Zhai. PaliGemma: A versatile 3B VLM for transfer. *arXiv preprint arXiv:2407.07726*, 2024. 1, 4, 5, 6, 12, 14, 17, 19, 20
- [11] Ali Furkan Biten, Ruben Tito, Andres Mafra, Lluis Gomez, Marçal Rusinol, Ernest Valveny, CV Jawahar, and Dimosthenis Karatzas. Scene text visual question answering. In *ICCV*, 2019. 5
- [12] Junbum Cha, Wooyoung Kang, Jonghwan Mun, and Byungseok Roh. Honeybee: Locality-enhanced projector for multimodal llm. In *CVPR*, 2024. 19
- [13] Guiming Hardy Chen, Shunian Chen, Ruifei Zhang, Junying Chen, Xiangbo Wu, Zhiyi Zhang, Zhihong Chen, Jianquan Li, Xiang Wan, and Benyou Wang. Allava: Harnessing gpt4v-synthesized data for a lite vision-language model. *arXiv preprint arXiv:2402.11684*, 2024. 20
- [14] Kaibing Chen, Dong Shen, Hanwen Zhong, Huasong Zhong, Kui Xia, Di Xu, Wei Yuan, Yifei Hu, Bin Wen, Tianke Zhang, Changyi Liu, Dewen Fan, Huihui Xiao, Jiahong Wu, Fan Yang, Size Li, and Di Zhang. Evlm: An efficient vision-language model for visual understanding. *arXiv preprint arXiv:2407.14177*, 2024. 19
- [15] Lin Chen, Jisong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahu Lin. ShareGPT4V: Improving large multi-modal models with better captions. *arXiv preprint arXiv:2311.12793*, 2023. 1, 17, 20
- [16] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 16
- [17] 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 preprint arXiv:1504.00325*, 2015. 2, 4, 11
- [18] Xi Chen, Xiao Wang, Lucas Beyer, Alexander Kolesnikov, Jialin Wu, Paul Voigtlaender, Basil Mustafa, Sebastian Goodman, Ibrahim Alabdulmohsin, Piotr Padlewski, et al. Pali-3 vision language models: Smaller, faster, stronger. *arXiv preprint arXiv:2310.09199*, 2023. 19
- [19] Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, Ji Ma, Jiaqi Wang, Xiaoyi Dong, Hang Yan, Hewei Guo, Conghui He, Botian Shi, Zhenjiang Jin, Chao Xu, Bin Wang, Xingjian Wei, Wei Li, Wenjian Zhang, Bo Zhang, Pinlong Cai, Licheng Wen, Xiangchao Yan, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahu Lin, Yu Qiao, Jifeng Dai, and Wenhui Wang. How far are we to GPT-4V? closing the gap to commercial multimodal models with open-source suites. *arXiv preprint arXiv:2404.16821*, 2024. 3
- [20] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *CVPR*, 2023. 18
- [21] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, and Ion Stoica. Chatbot arena: An open platform for evaluating LLMs by human preference. In *ICML*, 2024. 6, 13
- [22] Xiangxiang Chu, Limeng Qiao, Xinyang Lin, Shuang Xu, Yang Yang, Yiming Hu, Fei Wei, Xinyu Zhang, Bo Zhang, Xiaolin Wei, and Chunhua Shen. Mobilevlm : A fast, strong and open vision language assistant for mobile devices. *arXiv preprint arXiv:2312.16886*, 2023. 19
- [23] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. 16
- [24] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reいchiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 16
- [25] David F Crouse. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 2016. 14
- [26] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Albert Li, Pascale Fung, and Steven C. H. Hoi. Instructclip: Towards general-purpose vision-language models with instruction tuning. In *NeurIPS*, 2023. 18
- [27] Wenliang Dai, Nayeon Lee, Boxin Wang, Zhuoling Yang, Zihan Liu, Jon Barker, Tuomas Rintamaki, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NVLM: Open frontier-class multimodal LLMs. *arXiv preprint arXiv:2409.11402*, 2024. 5
- [28] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *ICLR*, 2024. 10

- [29] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, 2022. 10
- [30] Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Bin Wang, Linke Ouyang, Songyang Zhang, Haodong Duan, Wenwei Zhang, Yining Li, Hang Yan, Yang Gao, Zhe Chen, Xinyue Zhang, Wei Li, Jingwen Li, Wenhui Wang, Kai Chen, Conghui He, Xingcheng Zhang, Jifeng Dai, Yu Qiao, Dahua Lin, and Jiaqi Wang. InternLM-XComposer2-4KHD: A Pioneering Large Vision-Language Model Handling Resolutions from 336 Pixels to 4K HD. *arXiv preprint arXiv:2404.06512*, 2024. 3
- [31] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3
- [32] Yunhao Fang, Ligeng Zhu, Yao Lu, Yan Wang, Pavlo Molchanov, Jan Kautz, Jang Hyun Cho, Marco Pavone, Song Han, and Hongxu Yin. Vila²: Vila augmented vila. *arXiv preprint arXiv:2407.17453*, 2024.
- [33] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. *NeurIPS*, 2013. 18
- [34] Chaoyou Fu, Haojia Lin, Zuwei Long, Yunhang Shen, Meng Zhao, Yifan Zhang, Xiong Wang, Di Yin, Long Ma, Xiawu Zheng, et al. Vita: Towards open-source interactive omni multimodal llm. *arXiv preprint arXiv:2408.05211*, 2024. 19
- [35] Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024. 18
- [36] 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. In *CVPR*, 2017. 5, 6
- [37] Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, A. Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Author, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Daniel Morrison, Niklas Muenninghoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hanna Hajishirzi. OLMo: Accelerating the science of language models. In *ACL*, 2024. 3, 10, 20
- [38] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021. 16
- [39] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *NeurIPS Track on Datasets and Benchmarks*, 2021. 16
- [40] Joeri R Hermans, Gerasimos Spanakis, and Rico Möckel. Accumulated gradient normalization. In *Asian Conference on Machine Learning*, pages 439–454. PMLR, 2017. 10
- [41] Wenyi Hong, Weihan Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yan Wang, Yean Cheng, Shiyu Huang, Junhui Ji, Zhao Xue, et al. Cogvlm2: Visual language models for image and video understanding. *arXiv preprint arXiv:2408.16500*, 2024. 18
- [42] Anwen Hu, Haiyang Xu, Jiabo Ye, Mingshi Yan, Liang Zhang, Bo Zhang, Chen Li, Ji Zhang, Qin Jin, Fei Huang, and Jingren Zhou. mplug-docowl 1.5: Unified structure learning for ocr-free document understanding. In *Findings of EMNLP*, 2024.
- [43] Chao Jia, Yinfen Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021. 18
- [44] Dongfu Jiang, Xuan He, Huaye Zeng, Cong Wei, Max Ku, Qian Liu, and Wenhui Chen. Mantis: Interleaved multi-image instruction tuning. *arXiv preprint arXiv:2405.01483*, 2024. 20
- [45] Roy Jonker and Ton Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 1987. 14
- [46] Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. DVQA: Understanding data visualizations via question answering. In *CVPR*, 2018. 5, 11, 20
- [47] Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Ákos Kádár, Adam Trischler, and Yoshua Bengio. FigureQA: An annotated figure dataset for visual reasoning. *arXiv preprint arXiv:1710.07300*, 2017. 5, 20
- [48] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. *arXiv preprint arXiv:2402.07865*, 2024. 5, 19
- [49] Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *ECCV*, 2016. 5, 6
- [50] Diederik P Kingma. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5, 9
- [51] 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. In *ICCV*, 2023. 7, 14
- [52] 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 Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123:32 – 73, 2016. 11, 20
- [53] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 2020. 13
- [54] Hugo Laurençon, Andrés Marafioti, Victor Sanh, and Léo Tronchon. Building and better understanding vision-language models: insights and future directions. *arXiv preprint arXiv:2408.12637*, 2024. 20
- [55] Hugo Laurençon, Léo Tronchon, and Victor Sanh. Unlocking the conversion of web screenshots into HTML code with the websight dataset. *arXiv preprint arXiv:2403.09029*, 2024. 20
- [56] Bo Li, Peiyuan Zhang, Jingkang Yang, Yuanhan Zhang, Fanyi Pu, and Ziwei Liu. Otterhd: A high-resolution multi-modality model. *arXiv preprint arXiv:2311.04219*, 2023. 19
- [57] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Fanyi Pu, Jingkang Yang, C. Li, and Ziwei Liu. Mimic-it: Multi-modal in-context instruction tuning. *arXiv preprint arXiv:2306.05425*, 2023. 20
- [58] 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. 19
- [59] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. LLaVA-OneVision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024. 5, 6, 14, 18, 20
- [60] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

- [61] Junyan Li, Delin Chen, Yining Hong, Zhenfang Chen, Peihao Chen, Yikang Shen, and Chuang Gan. Covlm: Composing visual entities and relationships in large language models via communicative decoding. In *ICLR*, 2024. 20
- [62] Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on computer control agents. *arXiv preprint arXiv:2406.03679*, 2024. 5, 7
- [63] Zhang Li, Biao Yang, Qiang Liu, Zhiyin Ma, Shuo Zhang, Jingxu Yang, Yabo Sun, Yuliang Liu, and Xiang Bai. Monkey: Image resolution and text label are important things for large multi-modal models. *CVPR*, 2024.
- [64] Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Junwu Zhang, Munan Ning, and Li Yuan. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*, 2024. 19
- [65] 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. 13
- [66] Chang Liu, Henghui Ding, and Xudong Jiang. GRES: Generalized referring expression segmentation. In *CVPR*, 2023. 18, 20
- [67] Dongyang Liu, Renrui Zhang, Longtian Qiu, Siyuan Huang, Weifeng Lin, Shitian Zhao, Shijie Geng, Ziyi Lin, Peng Jin, Kaipeng Zhang, Wenqi Shao, Chao Xu, Conghui He, Junjun He, Hao Shao, Pan Lu, Yu Qiao, Hongsheng Li, and Peng Gao. SPHINX-x: Scaling data and parameters for a family of multimodal large language models. In *ICML*, 2024. 19
- [68] Fuxiao Liu, Xiaoyang Wang, Wenlin Yao, Jianshu Chen, Kaiqiang Song, Sangwoo Cho, Yaser Yacoob, and Dong Yu. Mmc: Advancing multimodal chart understanding with large-scale instruction tuning. *arXiv preprint arXiv:2311.10774*, 2023. 20
- [69] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 1, 3, 5, 6, 14, 18, 19, 20
- [70] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *CVPR*, 2024. 3
- [71] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024. 12, 18, 20
- [72] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 9
- [73] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 5, 9
- [74] Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, et al. Deepseek-vl: towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*, 2024. 19
- [75] Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. Unified-io 2: Scaling autoregressive multimodal models with vision language audio and action. In *CVPR*, 2024. 20
- [76] Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *NeurIPS*, 2022. 5
- [77] Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *ICLR*, 2023. 5
- [78] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. MathVista: Evaluating mathematical reasoning of foundation models in visual contexts. In *ICLR*, 2024. 6
- [79] Gen Luo, Yiyi Zhou, Tianhe Ren, Shengxin Chen, Xiaoshuai Sun, and Rongrong Ji. Cheap and quick: Efficient vision-language instruction tuning for large language models. In *NeurIPS*, 2023. 19
- [80] Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. ExpertQA: Expert-curated questions and attributed answers. *arXiv preprint arXiv:2309.07852*, 2023. 20
- [81] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. OK-VQA: A visual question answering benchmark requiring external knowledge. In *CVPR*, 2019. 5
- [82] Ahmed Masry, Do Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *ACL*, 2022. 5, 6
- [83] Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. DocVQA: A dataset for VQA on document images. In *WACV*, 2021. 5, 6
- [84] Minesh Mathew, Viraj Bagal, Rubén Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. InfographicVQA. In *WACV*, 2022. 5, 6
- [85] Brandon McKinzie, Zhe Gan, Jean-Philippe Fauconnier, Sam Dodge, Bowen Zhang, Philipp Dufter, Dhruti Shah, Xianzhi Du, Futang Peng, Floris Weers, Anton Belyi, Haotian Zhang, Karanjeet Singh, Doug Kang, Ankur Jain, Hongyu He, Max Schwarzer, Tom Gunter, Xiang Kong, Aonan Zhang, Jianyu Wang, Chong Wang, Nan Du, Tao Lei, Sam Wiseman, Guoli Yin, Mark Lee, Zirui Wang, Ruoming Pang, Peter Grasch, Alexander Toshev, and Yinfel Yang. Mm1: Methods, analysis & insights from multimodal llm pre-training. *arXiv preprint arXiv:2403.09611*, 2024. 3, 18
- [86] Nitesh Methani, Pritha Ganguly, Mitesh M Khapra, and Pratyush Kumar. PlotQA: Reasoning over scientific plots. In *WACV*, 2020. 5, 20
- [87] Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. OL-MoE: Open mixture-of-experts language models. *arXiv preprint arXiv:2409.02060*, 2024. 3
- [88] OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1, 6, 14
- [89] OpenAI. GPT-4o mini system card, 2024. 12, 14, 18
- [90] OpenAI. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 3, 6, 14, 19
- [91] Maxime Oquab, Timothée Darcret, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 19
- [92] 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 arXiv:2306.14824*, 2023. 18, 20
- [93] Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. Connecting vision and language with localized narratives. In *ECCV*, 2020. 4, 11
- [94] Shraman Pramanick, Guangxing Han, Rui Hou, Sayan Nag, Ser-Nam Lim, Nicolas Ballas, Qifan Wang, Rama Chellappa, and Amjad Almahairi. Jack of all tasks, master of many: Designing general-purpose coarse-to-fine vision-language model. In *CVPR*, 2023. 20
- [95] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI Blog*, 2018. 3
- [96] 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. In *ICML*, 2021. 3, 18
- [97] Hanoona Rasheed, Muhammad Maaz, Sahal Shaji, Abdelrahman Shaker, Salman Khan, Hisham Cholakkal, Rao M Anwer, Eric Xing, Ming-Hsuan Yang, and Fahad S Khan. GLaMM: Pixel grounding large multimodal model. In *CVPR*, 2024. 20
- [98] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022. 17
- [99] Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-OKVQA: A benchmark for visual question answering using world knowledge. In *ECCV*, 2022. 5
- [100] Amanpreet Singh, Vivek Natarjan, Meet Shah, Yu Jiang, Xinlei Chen, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *CVPR*, 2019. 5, 6
- [101] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024. 10
- [102] Quan Sun, Yufeng Cui, Xiaosong Zhang, Fan Zhang, Qiying Yu, Zhengxiong Luo, Yueze Wang, Yongming Rao, Jingjing Liu, Tiejun Huang, and Xinlong Wang. Generative multimodal models are in-context learners. In *CVPR*, 2024.
- [103] Gemini Team. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. 3, 6, 14, 19
- [104] OpenGVLab Team. Internvl2: Better than the best—expanding performance boundaries of open-source multimodal models with the progressive scaling strategy, 2024. 6, 14
- [105] Rahul Thapa, Kezhen Chen, Ian Covert, Rahul Chalamala, Ben Athiwaratkun, Shuaiwen Leon Song, and James Zou. Dragonfly: Multi-resolution zoom-in encoding enhances vision-language models. *arXiv preprint arXiv:2406.00977*, 2024. 18
- [106] Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. In *NeurIPS*, 2024. 5, 6, 12, 14, 19, 20
- [107] Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? exploring the visual shortcomings of multimodal LLMs. In *CVPR*, 2024. 18
- [108] Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, S. M. Ali Eslami, Oriol Vinyals, Felix Hill, and Zacharias Janssen. Multimodal few-shot learning with frozen language models. In *NeurIPS*, 2021.
- [109] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3, 18
- [110] Junke Wang, Lingchen Meng, Zejia Weng, Bo He, Zuxuan Wu, and Yu-Gang Jiang. To see is to believe: Prompting gpt-4v for better visual instruction tuning. *arXiv preprint arXiv:2311.07574*, 2023. 20
- [111] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. 6, 7, 14, 20
- [112] Weihua Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, Jiazhen Xu, Bin Xu, Juanzi Li, Yuxiao Dong, Ming Ding, and Jie Tang. Cogvlm: Visual expert for pretrained language models. In *NeurIPS*, 2024. 18
- [113] Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. How far can camels go? exploring the state of instruction tuning on open resources. In *NeurIPS Track on Datasets and Benchmarks*, 2023. 15
- [114] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhui Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *NeurIPS Track on Datasets and Benchmarks*, 2024. 16
- [115] Jason Weston, Samy Bengio, and Nicolas Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, 2011. 18
- [116] xAI. RealWorldQA. <https://huggingface.co/datasets/xai-org/RealworldQA>, 2024. Accessed: 2024-09-24. 6
- [117] Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yu-mao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. In *CVPR*, 2024. 20
- [118] Hu Xu, Saining Xie, Xiaoqing Tan, Po-Yao Huang, Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. Demystifying CLIP data. In *ICLR*, 2024. 3, 18
- [119] Le Xue, Manli Shu, Anas Awadalla, Jun Wang, An Yan, Senthil Purushwalkam, Honglu Zhou, Viraj Prabhu, Yutong Dai, Michael S Ryoo, et al. xGen-MM (BLIP-3): A family of open large multimodal models. *arXiv preprint arXiv:2408.08872*, 2024. 6, 14, 20
- [120] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024. 3
- [121] Charig Yang, Weidi Xie, and Andrew Zisserman. It's about time: Analog clock reading in the wild. In *CVPR*, 2022. 7, 13, 14, 20
- [122] Huanjin Yao, Wenhao Wu, Taojinnan Yang, Yuxin Song, Mengxi Zhang, Haocheng Feng, Yifan Sun, Zhiheng Li, Wanli Ouyang, and Jingdong Wang. Dense connector for mllms. In *NeurIPS*, 2024.
- [123] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024. 19
- [124] Jiaabo Ye, Anwen Hu, Haiyang Xu, Qinghao Ye, Ming Yan, Guohai Xu, Chenliang Li, Junfeng Tian, Qi Qian, Ji Zhang, et al. UReader: Universal OCR-free visually-situated language understanding with multimodal large language model. In *Findings of EMNLP*, 2023. 3
- [125] 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 preprint arXiv:2310.07704*, 2023. 20
- [126] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *ECCV*, 2016. 18, 20
- [127] Yuqian Yuan, Wentong Li, Jian Liu, Dongqi Tang, Xinjie Luo, Chi Qin, Lei Zhang, and Jianke Zhu. Osprey: Pixel understanding with visual instruction tuning. In *CVPR*, 2024. 20
- [128] Zhengqing Yuan, Zhaoxu Li, Weiran Huang, Yanfang Ye, and Lichao Sun. Tinygpt-v: Efficient multimodal large language model

- via small backbones. In *ICML Workshop on Advancing Neural Network Training*, 2024. 19
- [129] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhui Chen. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *CVPR*, 2024. 6
- [130] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *ICCV*, 2023. 3
- [131] Haotian Zhang, Mingfei Gao, Zhe Gan, Philipp Dufter, Nina Wenzel, Forrest Huang, Dhruti Shah, Xianzhi Du, Bowen Zhang, Yang-hao Li, Sam Dodge, Keen You, Zhen Yang, Aleksei Timofeev, Mingze Xu, Hong-You Chen, Jean-Philippe Fauconnier, Zhengfeng Lai, Haoxuan You, Zirui Wang, Afshin Dehghan, Peter Grasch, and Yinfei Yang. Mm1.5: Methods, analysis & insights from multimodal llm fine-tuning. *arXiv preprint arXiv:2409.20566*, 2024. 18
- [132] Haotian Zhang, Haoxuan You, Philipp Dufter, Bowen Zhang, Chen Chen, Hong-You Chen, Tsu-Jui Fu, William Yang Wang, Shih-Fu Chang, Zhe Gan, and Yinfei Yang. Ferret-v2: An improved baseline for referring and grounding with large language models. In *COLM*, 2024. 20
- [133] Pan Zhang, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Rui Qian, Lin Chen, Qipeng Guo, Haodong Duan, Bin Wang, Linke Ouyang, et al. Internlm-xcomposer-2.5: A versatile large vision language model supporting long-contextual input and output. *arXiv preprint arXiv:2407.03320*, 2024. 20
- [134] Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Jiao Qiao. Llama-adapter: Efficient fine-tuning of large language models with zero-initialized attention. In *ICLR*, 2024. 19
- [135] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: Experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023. 10
- [136] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, 2022. 4
- [137] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. In *ICLR*, 2024. 19, 20