

Prismatic VLMs: Investigating the Design Space of Visually-Conditioned Language Models

Siddharth Karamcheti^{1,2} Suraj Nair² Ashwin Balakrishna² Percy Liang¹ Thomas Kollar^{2,†} Dorsa Sadigh^{1,†}

github.com/TRI-ML/prismatic-vlms

github.com/TRI-ML/vlm-evaluation

Abstract

Visually-conditioned language models (VLMs) have seen growing adoption in applications such as visual dialogue, scene understanding, and robotic task planning; adoption that has fueled a wealth of new models such as LLaVa, Instruct-BLIP, and PaLI-3. Despite the volume of new releases, key design decisions around image preprocessing, architecture, and optimization are under-explored, making it challenging to understand what factors account for model performance – a challenge further complicated by the lack of objective, consistent evaluations. To address these gaps, we first compile a suite of standardized evaluations spanning visual question answering, object localization, and challenge sets that probe properties such as hallucination; evaluations that provide fine-grained insight VLM capabilities. Second, we rigorously investigate VLMs along key design axes, including pretrained visual representations and training from base vs. instruct-tuned language models, amongst others. We couple our analysis with three resource contributions: (1) a unified framework for evaluating VLMs, (2) optimized, flexible training code, and (3) checkpoints for all models, including a family of VLMs at the 7-13B scale that strictly outperform InstructBLIP and LLaVa v1.5, the state-of-the-art in open VLMs.

1. Introduction

If you have built castles in the air, your work need not be lost; that is where they should be. Now put the foundations under them.

— HENRY DAVID THOREAU

[†]Equal Advising ¹Department of Computer Science, Stanford University, Stanford, CA, USA ²Toyota Research Institute, Los Altos, CA, USA. Correspondence to: Siddharth Karamcheti <skaramcheti@cs.stanford.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

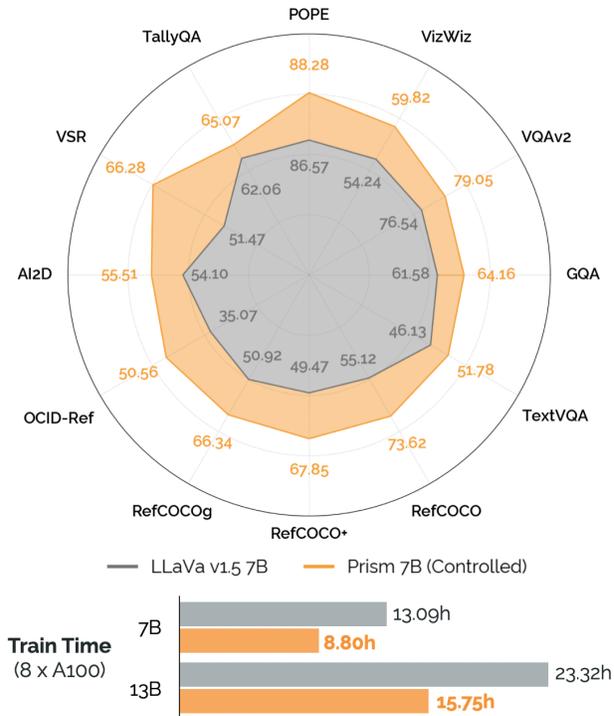


Figure 1. **Prismatic VLMs.*** Through rigorous experiments exploring the design space of visually-conditioned language models (VLMs), we identify insights that improve training. When controlling for data and scale, our models (orange) outperform the state-of-the-art LLaVa v1.5 (gray; Liu et al., 2023b) across 12 diverse tasks, while saving more than 30% the training compute.

Visually-conditioned language models (VLMs) generate natural language responses from image input and text prompts, providing a general, expressive interface for a growing spectrum of applications – grounded chat (Li et al., 2023c; Gong et al., 2023), visual programming (Sur’is et al., 2023; Subramanian et al., 2023), robotic control (Driess et al., 2023; Brohan et al., 2023), etc. This broad adoption is fueled by a recent *paradigm shift* in how we develop VLMs; eschewing the complex architectures and training objectives

***Prismatic** (adj) – relating to or having the form of a prism. Like a geometric prism, our VLMs share a common structure, but are characterized by different “faces” – the individual design axes we explore in this work.

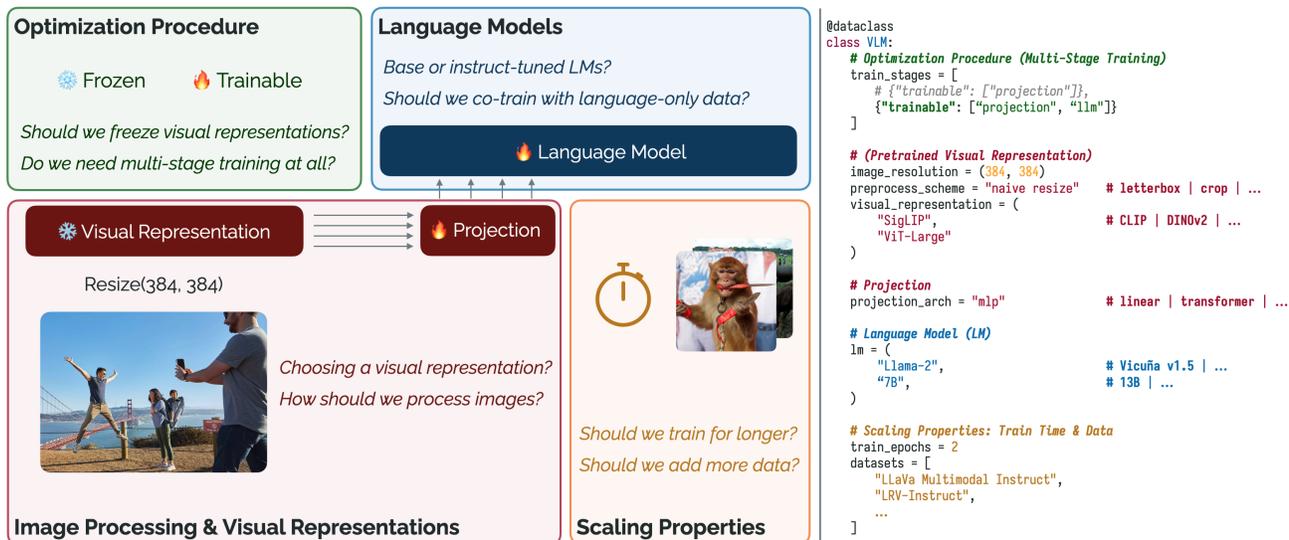


Figure 2. **Exploring VLM Design Axes.** We explore four key design axes for developing VLMs: 1) optimization procedure, 2) image processing and pretrained visual representations, 3) language models, and 4) scaling properties around training time and data (left). To enable this exploration, we make a key resource contribution: an open-source, flexible codebase for efficiently training VLMs (right).

of prior work (Tan & Bansal, 2019; Li et al., 2022; 2023b), new VLMs adopt a simple approach, treating patch features from pretrained visual backbones (e.g., CLIP; Radford et al., 2021) as tokens that can be projected into the input space of a language model (LM). This “patch-as-token” approach enables training with a simple objective – next-token prediction – and allows us to harness the ecosystem of powerful LMs such as Llama-2 and Mistral (Touvron et al., 2023; Jiang et al., 2023), along with the tools to efficiently train them (e.g., FSDP; Zhao et al., 2023). This combination has fueled the rapid development and release of models such as LLaVa v1.5, and PALI-3 that adopt the same underlying recipe, while varying individual ingredients such as the choice of pretrained components, data, or optimization procedure (Liu et al., 2023b; Chen et al., 2023b).

Unfortunately, existing approaches only cover a sliver of the design space around building and training VLMs, without thoroughly evaluating the impact of given choices on downstream capabilities. This motivates the key question of this work: *what are the key design decisions that influence VLM capabilities and downstream use?* To provide answers to this question, we first need a way to **thoroughly evaluate** the strengths and weaknesses of a given model. Doing this effectively requires compiling a standardized evaluation suite comprised of tasks that are diverse and objective; crucially, these tasks should allow for probing specific capabilities such as spatial reasoning, out-of-distribution generalization, and commonsense understanding, amongst others. Second, we need to **rigorously explore** different VLM design axes, not only to build a concrete set of recommendations, but to tie individual choices to downstream performance.

This work addresses these axes through four contributions.

First, to provide fine-grained insight into VLM capabilities, **we compile a standardized evaluation suite** comprised of twelve benchmarks from the vision-and-language literature, including four tasks spanning visual question answering (Bigham et al., 2010; Goyal et al., 2017; Hudson & Manning, 2019; Singh et al., 2019), four tasks spanning object localization (Kazemzadeh et al., 2014; Yu et al., 2016; Wang et al., 2021), and four challenge tasks evaluating fine-grained spatial reasoning, hallucination, and diagram understanding (Acharya et al., 2018; Liu et al., 2022; Li et al., 2023d; Kembhavi et al., 2016). Second, **we develop an optimized and modular codebase for VLM training** that emphasizes flexibility, allowing users to easily swap in pretrained components, optimization procedures, data, and more (Fig. 2; right). Third, we use these resource contributions to perform **targeted experiments exploring four key design axes** (Fig. 2; left): 1) optimization procedure, 2) image processing and visual representations, 3) language models, and 4) scaling training time and data. We identify a number of insights; for example, we find that multi-stage training procedures adopted by existing work can be eliminated without impact on performance, reducing compute costs by 20-25%. We also find that *fused* visual backbones that merge features from different backbones such as CLIP (Radford et al., 2021) and DINOv2 (Oquab et al., 2023) lead to more performant VLMs across the board. Finally, we consolidate our findings and train a family of models – PRISMS – at the 7B/13B scale that **strictly outperform state-of-the-art open VLMs** such as InstructBLIP and LLaVa v1.5.¹

¹We release our optimized training codebase, evaluation suite, and checkpoints for all models trained as part of this work.

2. Preliminaries

To ground our analysis, we require 1) a VLM model architecture, 2) pretraining data, and 3) a training implementation.

Model Architecture. We adopt the general architecture used by many recent VLMs, such as LLaVa, Qwen-VL, and PaLI-3 (Liu et al., 2023c; Bai et al., 2023; Chen et al., 2023b). These architectures use a (pretrained) visual backbone to map an input image to a sequence of patch features that are then projected individually into the embedding space of an LM. Formally, a VLM takes as input an image $x_{\text{img}} \in \mathbb{R}^{H \times W}$ and text prompt tokens u_{prompt} with arbitrary sequence length K . These inputs are then fed to the following components: 1) a visual representation backbone, 2) a vision-language projector, and 3) a language model.

Visual Representation. We first process x_{img} subject to a visual representation backbone V_{ω} that outputs a sequence of features $p_{\text{img}} \in \mathbb{R}^{L \times h_{\text{vision}}}$ where $p_{\text{img}} = V_{\omega}(x_{\text{img}})$. As an example, p_{img} might be the patch features output by a Vision Transformer (ViT; Dosovitskiy et al., 2021).

Vision-Language Projector. Next, we map p_{img} to a sequence of embeddings $e_{\text{img}} \in \mathbb{R}^{L \times h_{\text{text}}}$ via a learned projector F_{ψ} , where $e_{\text{img}} = F_{\psi}(p_{\text{img}})$.

Language Model. Finally, we concatenate the sequence e_{img} with the text prompt embeddings $e_{\text{prompt}} = \text{embed}(u_{\text{prompt}})$, passing the result to the language model. The language model generates output text $u_{\text{gen}} = \text{LM}_{\theta}([e_{\text{img}}; e_{\text{prompt}}])$.

The composition $\text{LM}_{\theta}([F_{\psi}(V_{\omega}(o_{\text{rgb}})); \text{embed}(u_{\text{prompt}})])$ then defines a VLM. Given a triple $(x_{\text{img}}, u_{\text{prompt}}, \hat{u}_{\text{gen}})$ during training, we minimize the loss $\mathcal{L}(\omega, \psi, \theta) = -\log p(\hat{u}_{\text{gen}} | x_{\text{img}}, u_{\text{prompt}})$ via gradient descent.

Pretraining Dataset. We limit our selection of pretraining data to datasets that are fully open-source (e.g., under permissive research licenses), and that have been used in prior work. Specifically, we use the LLaVa v1.5 data mixture, which consists of two subsets used for a multi-stage training pipeline. The first subset consists of a 558K sample mixture of examples sourced from various captioning datasets (e.g., Conceptual Captions, LAION Sharma et al., 2018; Schuhmann et al., 2021), while the second consists of 665K multimodal instruct tuning examples comprised of synthetic data generated in Liu et al. (2023c), as well as examples from existing vision-language training sets (e.g., GQA, TextCaps; Hudson & Manning, 2019; Sidorov et al., 2020), and notably, a sample of language-only data from ShareGPT (ShareGPT, 2023). We provide a comprehensive breakdown of the pretraining data mixture in §A.1.

Training Implementation & Verification. To investigate the design axes enumerated in §1, we require code for VLM training that is *efficient* and *flexible*; critically, we need the ability to easily swap out vision and LM backbones and

handle arbitrary optimization procedures (e.g., freezing the vision backbone during training). With these requirements, we implement our training codebase in PyTorch, using Fully Sharded Data Parallel (FSDP; Zhao et al., 2023) and BF16 mixed precision. FSDP lets us specify precision for individual model components (e.g., FP16 for vision backbones, BF16 for LMs), enables portability to different hardware, and provides minimal implementation overhead. Following reproducibility practices from prior work (Karamcheti et al., 2021; Biderman et al., 2023), we fix initialization randomness and fix batch order during training. We leverage TIMM (Wightman, 2019) and Hugging Face Transformers (Wolf et al., 2019) to provide pretrained models.

To validate our code, we run an apples-to-apples reproduction of LLaVa v1.5 (Liu et al., 2023b) at both the 7B and 13B parameter scale. Successful reproduction results are in Fig. 4 (left). We find our implementation is considerably more efficient than the reference LLaVa v1.5 training implementation: when benchmarked on the same hardware (an AWS p4de.24xlarge node with 8 A100 GPUs), we observe 20% faster step times with our FSDP-backed implementation, a notable gain given LLaVa leverages the well-optimized DeepSpeed ZeRO library (Rasley et al., 2020).

We highlight this open-source training codebase as one of the key contributions of this work. Unlike other open codebases, we provide a modular and expressive interface for easily specifying or adding model components, optimization procedures, and data with minimal code changes (Fig. 2; right). In providing an efficient and easily extensible framework, we enable future research around designing new evaluations, developing and training new VLMs, and finetuning or otherwise adapting existing models for diverse downstream applications – all while maintaining a high standard of reproducibility and controlled experimentation.

3. Evaluation Suite

The first contribution of this work is a unified evaluation suite that offers *fine-grained insight* into the capabilities of a given VLM. Recent work in evaluating VLMs tends to rely on automated evaluations that use powerful LMs such as GPT-4 (OpenAI et al., 2023) to judge relative and subjective performance (Liu et al., 2023e; Yu et al., 2023), making it hard to measure the absolute impact of a given design change. Instead, we focus on evaluations with well-defined metrics, spanning the following three areas:

Open-Ended Visual Question Answering. We evaluate on VizWiz (Bigham et al., 2010), VQAv2 (Goyal et al., 2017), GQA (Hudson & Manning, 2019), and TextVQA (Singh et al., 2019). Both VizWiz and VQAv2 assess general visual reasoning; VizWiz also contains a series of unanswerable questions. GQA evaluates spatial reasoning, while

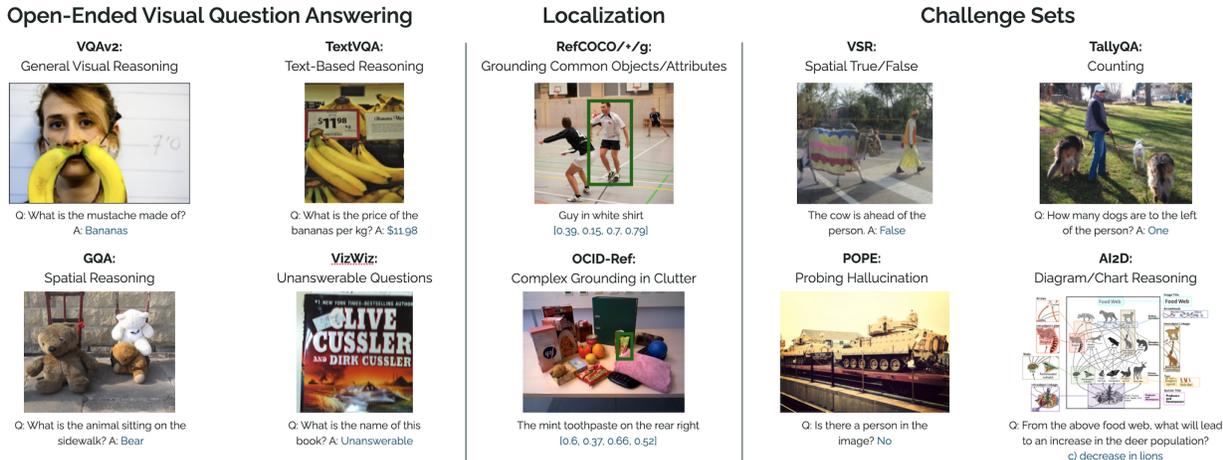


Figure 3. **Evaluation Suite Overview.** We compile multiple established benchmarks spanning visual question answering, localization, and challenge tasks (e.g., evaluating counting, spatial relationships, propensity to hallucinate). This evaluation suite forms the backbone for all of our analysis, giving us fine-grained insight into the impact of individual VLM design choices.

TextVQA assesses reasoning around text (e.g., labels, signage) present in an image.

Localization. Part of the pretraining data mixture (from §2) contains examples of predicting normalized bounding box coordinates given referring expressions in language. As such, we evaluate bounding box prediction accuracy on RefCOCO, RefCOCO+, and RefCOCOg (Kazemzadeh et al., 2014; Yu et al., 2016), and on OCID-Ref (Wang et al., 2021). RefCOCO focuses on short descriptions with spatial anchors, RefCOCO+ on strictly appearance based descriptions, and RefCOCOg on long, rich descriptions; OCID-Ref is a robotics dataset probing out-of-distribution generalization, with a focus on localizing objects in clutter.

Challenge Sets (Closed-Set Prediction). We evaluate on Visual Spatial Reasoning (VSR; Liu et al., 2022), TallyQA (Acharya et al., 2018), POPE (Li et al., 2023d), and AI2 Diagrams (AI2D; Kembhavi et al., 2016). VSR consists of challenging True/False questions about individual spatial relationships in diverse scenes (e.g., “the cake is at the edge of the dining table”); this is an especially challenging task, with most existing models failing to outperform the majority class baseline (51%). TallyQA consists of questions that assess a VLM’s ability to count objects described in language, with expressions that range in complexity. POPE consists of targeted Yes/No questions that assess a VLM’s propensity to hallucinate. Finally, AI2D consists of multiple choice questions about scientific diagrams and charts, many of which require reading labels or text annotations (e.g., flowchart labels, plot axes).

We use the validation sets for all benchmarks except GQA (where we use the recommended test-dev split), VSR (where we use the zero-shot test split), and POPE (where there is only a single evaluation split). We provide further detail around evaluation protocols in Appx. B.

4. Experiments – Investigating Design Axes

Our second contribution is a series of targeted experiments exploring the VLM design space along *four key axes*: (§4.1) optimization procedure, (§4.2) image processing and visual representations, (§4.3) language models, and (§4.4) scaling properties such as training time and data diversity.

Experiment Design: Protocols & Drawing Conclusions.

We first validate our VLM training implementation by reproducing LLaVa v1.5 (see §2), adopting the design choices of the original work – the same choices used by many other recent VLMs: “letterbox padding” to process images, CLIP ViT-Large with a patch size of 14 and input resolution of 336px (CLIP ViT-L/14 @ 336px; Radford et al., 2021) as the visual representation, Vicuña v1.5 as the LM backbone, and the two-stage training pipeline using both data subsets described in §2. Successful reproduction results at both the 7B and 13B scale are in Fig. 4 (left). Given both the fidelity of our reproduction and the prevalence of these design choices, we anchor our analyses around this parameterization. Critically, each of the experiments in §4.2, §4.3, and §4.4 are formulated as *single-step changes of this base architecture, with all other choices held constant*.

As each evaluation in §3 uses different metrics with different scales, direct comparison is challenging. We address this by computing normalized Z-scores for each model and evaluation (using the mean and standard deviation across all models). These scores are used to compute statistical significance (further details in §B.2), and to set the relative scales of each radar plot (for completeness, we also provide the absolute metrics as colored and bolded labels).

4.1. Optimization Procedure

In this section we focus on design choices around the optimization procedure used to initialize and train each of the

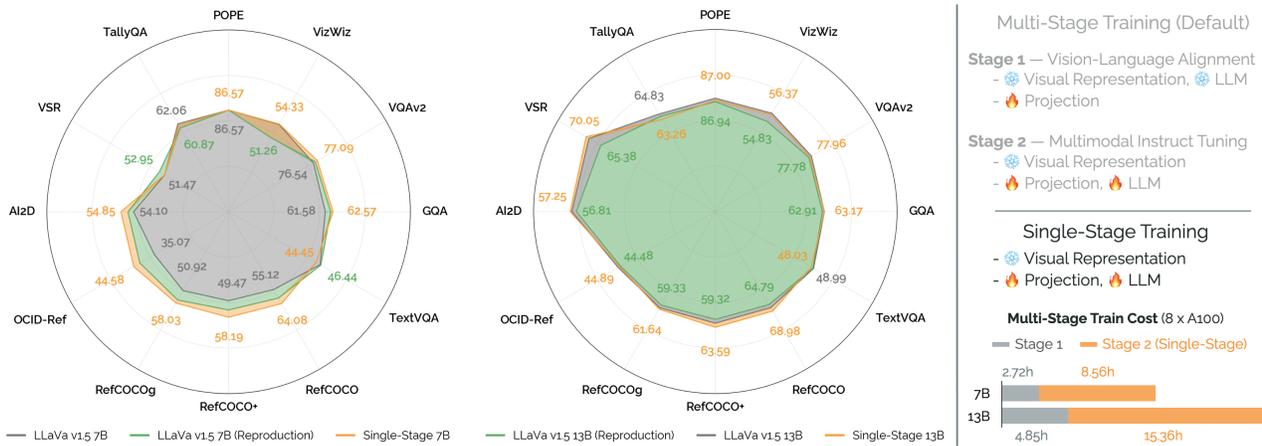


Figure 4. **Reproducing LLaVa v1.5 & Exploring Optimization Procedures.** To validate our training codebase (§2), we reproduce LLaVa v1.5 (green), with our models reproducing the performance reported in Liu et al. (2023b) (gray). We then run our first experiment (§4.1) investigating the need for expensive multi-stage training (right). We find that single-stage training produces VLMs that maintain or outperform multi-stage models (orange), saving considerable compute; as a result, we carry this change forward to all future experiments.

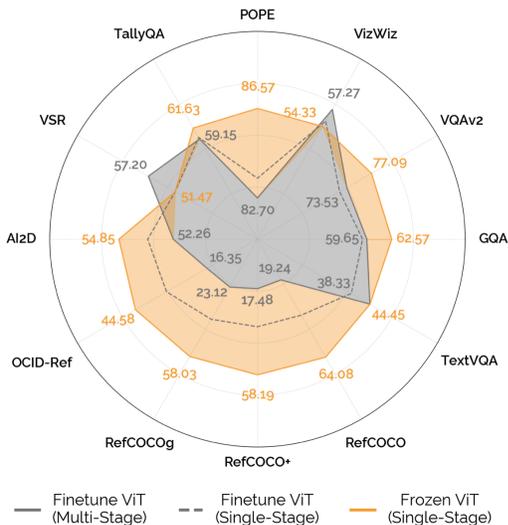


Figure 5. **Full Finetuning through Visual Backbones.** We explore the impact of finetuning the (conventionally frozen) visual backbone in addition to the projector and language model during training. We see that in both the single and multi-stage paradigms, finetuning the vision backbone dramatically degrades performance across almost all benchmarks – especially on localization tasks.

three components described in §2. Specifically, we examine the effects of multi-stage training where different VLM components are frozen at different points in training.

Multi-Stage Training. One of the prevalent design choices adopted by many VLMs (Chen et al., 2023a; Ye et al., 2023) is the inclusion of a two-stage training pipeline: (1) an alignment stage to align vision and language features by training the randomly initialized projector F_ψ in isolation, freezing all other components (Fig. 4, right) and (2) a finetuning stage, where only the visual representation is frozen while both the projection and LM are trained.

Adopting multi-stage training complicates implementation and adds to training cost; therefore, as an initial experiment, we evaluate the need for this first stage through a targeted ablation. We compare the default two-stage training procedure with a *single-stage* approach that skips directly to finetuning F_ψ and the LM. We find (Fig. 4; left) that including the explicit projector pretraining stage is unnecessary, with single-stage training improving aggregate performance ($p = 0.00558$). Eliminating this first stage saves 20-25% of training cost, and removes the need for additional, stage-specific data (e.g., the captioning subset from §2). As this change strictly improves performance and efficiency, we adopt single-stage training for all following experiments.

Full Finetuning through Visual Backbones. Another popular design choice in existing VLMs that leverage pretrained visual representations is to leave the visual backbone *frozen* during the entirety of training (Liu et al., 2023b; Driess et al., 2023; Li et al., 2023b). Such a choice limits the potential to learn improved visual representations conducive to language generation during the course of training. Thus, we ask – *is there potential to improve VLM performance by finetuning the full model, including the visual backbone?* We find (Fig. 5) that this is not the case, and that finetuning the visual backbone significantly degrades performance ($p = 0.00381$), especially on tasks requiring fine-grained spatial reasoning such as RefCOCO and OCID-Ref.

Remark. The degraded performance from full finetuning could be for a number of reasons ranging from the scale and diversity of the vision-language data we train on to language generation as a learning objective (vs. objectives that encourage learning fine-grained perceptual features). Especially given the existence of closed-source models such as Fuyu-8B (AI, 2023) that adopt this paradigm to great success, we believe that identifying ways to prevent such

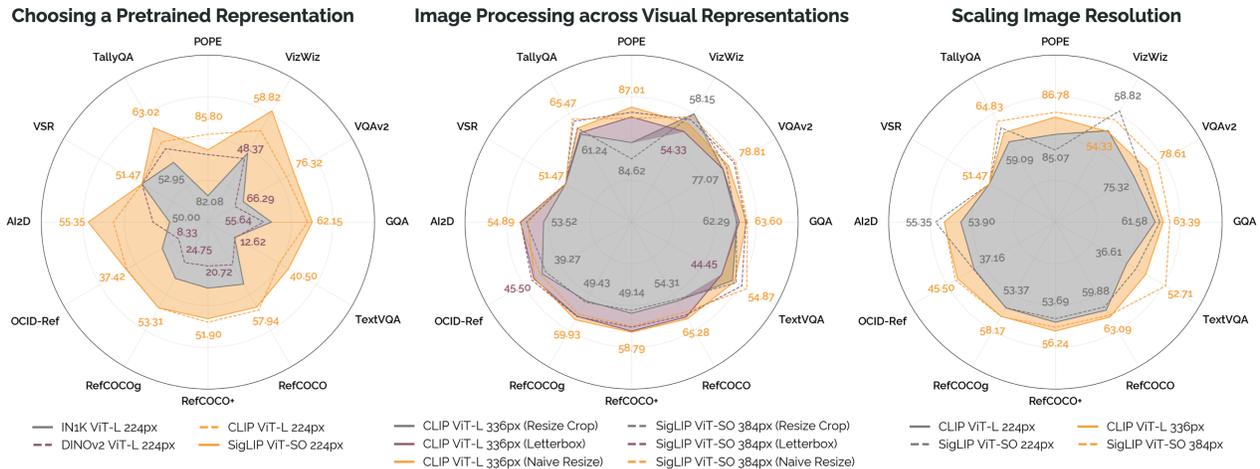


Figure 6. **Image Processing & Visual Representations.** We explore choices around image processing and visual representations in §4.2. Specifically, we investigate the impact of different visual representations (**left**), how performance varies as function of image processing strategies (**middle**), and the impact of increasing input image resolution (**right**).

feature collapse during VLM training (e.g., via auxiliary objectives) to be a rich direction for future work.

4.2. Image Processing & Visual Representations

Choosing a Pretrained Vision Representation. CLIP (Radford et al., 2021) has become the default choice for visual representation for almost all VLMs, despite a wealth of visual representations trained on diverse data sources. In this experiment, we perform a head-to-head comparison between CLIP, SigLIP (Zhai et al., 2023), DINOv2 (Oquab et al., 2023), and a standard Vision Transformer pretrained for classification (on ImageNet-21K, finetuned on ImageNet-1K; Dosovitskiy et al., 2021; Steiner et al., 2021); for fair comparison, we use the ViT-Large model variant.² We find (Fig. 6; left) that the backbones trained with vision-language contrastive objectives (i.e., CLIP, SigLIP) are significantly more performant than alternatives ($p = 7.11e-8$).

Remark. While the vision-language contrastive objective is one explanation for the strengths of CLIP and SigLIP, another possible explanation is one of training image distribution. Both CLIP and SigLIP contain internet-sourced images (e.g., sketches, diagrams, animated graphics, etc.) not in ImageNet or in the DINOv2 pretraining data.

Image Processing across Visual Backbones. Most images have resolutions and aspect ratios that widely vary, yet most visual backbones expect square images at a fixed size; to reconcile this, the overwhelming default is to “resize & crop” an image to size. While this tends to work well for applications such as classification, cropping out parts of an image is especially harmful for tasks requiring full-scene reasoning.

²To evaluate on an image resolution common to all representations (224px), we use the *shape-optimized* SigLIP model (ViT-SO Alabdulmohsin et al., 2023) that is slightly larger than a ViT-Large at 400M parameters (vs 307M).

In this experiment, we evaluate three different image processing schemes – the default “resize & crop” scheme, the “letterbox padding” scheme used by LLaVa v1.5 that pads non-square images to square, and a “naive resize” scheme that warps the original image aspect ratio, squeezing or stretching an image to square. Our findings (Fig. 6; middle) are surprising: while cropping is clearly suboptimal, the “naive resize” scheme is the most performant for CLIP. For SigLIP, both “naive resize” and “letterbox padding” perform similarly. In general, our results favor “naive resizing” over “letterbox padding” but we cannot rule the improvement statistically significant ($p = 0.0176$).

Remark. Two speculative arguments for naively resizing an image over padding are those of minimizing “dead pixels” and distribution shift. An image with a 16:9 aspect ratio that is padded to square introduces a large amount of uninformative pixels (exceeding 40%); warping the aspect ratio is possibly less of a shift. Coupled with the innate patch dimensionality of a Vision Transformer ($d = 1024$ for a 16×16 pixel patch), naively resizing an image may preserve enough information for the downstream LM (with 7B+ parameters) to extract the properties necessary for downstream tasks.

Scaling Image Resolution. Another trend in recent VLMs is increasing input image resolution with the hope of capturing fine-grained details that improve downstream performance (Liu et al., 2023b; Li et al., 2023a). Our findings (Fig. 6; right) confirm this hypothesis, with scaling to 336px or 384px offering significant improvements ($p = 6.05e-4$).

Remark. While scaling up image resolution seems like a clear win, we caution that it comes with a significant increase in compute complexity for VLMs that project individual ViT patches into the embedding space of an LM. Assuming a fixed patch granularity, doubling the input resolution results in four times the number of input patches fed

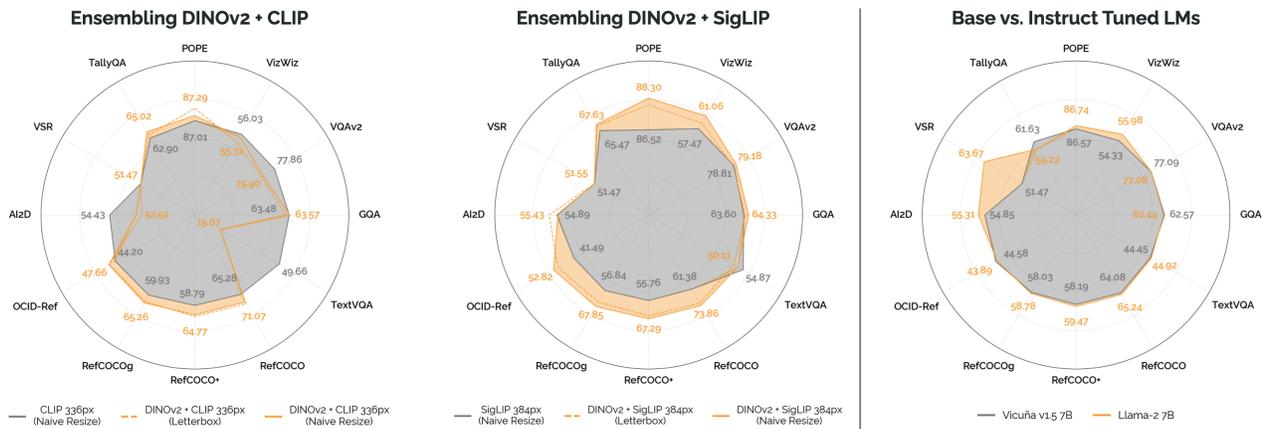


Figure 7. Ensembling Different Visual Representations & Base vs. Instruct-Tuned LMs. We explore fusing visual features from DINOv2 and CLIP/SigLIP models, and find that doing so significantly boosts performance on localization and challenge evaluations (left). We additionally evaluate the differences between base (Llama-2; orange) and instruct-tuned (Vicuña v1.5; gray) language models (right); we find that Llama-2 offers similar quantitative performance, while being less prone to hallucination (Fig. 11).

to the LM. Coupled with the quadratic cost of traditional Transformer attention as a function of sequence length, this is a sixteen-fold increase in time complexity (with a comparable explosion in memory requirements).

Ensembling Different Visual Representations. A rich body of prior work in vision identifies that different types of visual representations trained with different inductive biases can lead to improved performance for a broad spectrum of applications (Kobayashi et al., 2022; Karamcheti et al., 2023). Motivated by this, we ask if this same trend holds true for VLM training – specifically whether ensembling DINOv2 features with vision-language contrastive features from CLIP and SigLIP can lead to improved performance, following the approach taken in Kerr et al. (2023). To implement this efficiently, we simply concatenate patch features from different backbones along the channel dimension for each patch, resulting in the same number of input patch embeddings, just with double the feature dimension. To adjust for this, we just increase the input dimension to our projector F_ψ (a 2-layer MLP) at negligible cost. We find (Fig. 7 - left) that fusing DINOv2 and SigLIP features provides significant gains across the board ($p = 0.00164$), with a notable exception for the DINOv2 + CLIP models ($p = 0.37313$), where combining DINOv2 features seem to be particularly harmful on TextVQA. Looking at the remaining results, we see especially impressive gains of 5-10% on localization and challenge tasks; in general, the DINOv2 + SigLIP fused representations are the most performant visual representations we try, with virtually no added parameters.

Remark. Following the hypotheses in Kerr et al. (2023) and similar work, we believe that DINOv2 features provide features that capture low-level spatial properties of an image, augmenting the higher-level “semantic” properties captured by vision-language contrastive models. We note that this conclusion may generalize beyond DINO-style backbones;

the only reason we do not evaluate the fusion of ImageNet and CLIP/SigLIP backbones as well is due to a mismatch in patch granularity (the ImageNet backbone uses a patch granularity of 16×16 vs. the 14×14 granularity used by all other backbones). We believe that further exploring the impact on these type of fused, multi-resolution features for VLMs is a compelling avenue for future work.

4.3. Integrating Language Models

Base vs. Instruct-Tuned LMs. Instruct tuning (or chat tuning; Ouyang et al., 2022; Chung et al., 2022) is a way to finetune base LMs (trained for next-token prediction) to behave as dialogue agents, offering a natural input/output interface for a wide spectrum of applications. As a result, instruct tuned models like Vicuña (Zheng et al., 2023) have become the default backbone for VLMs. Unfortunately, instruct tuning has drawbacks, introducing bias and regressions in performance (Ouyang et al., 2022). Thus, in this experiment we evaluate the impact of instruct-tuned LM backbones on downstream VLM performance via a head-to-head comparison between a base LM (Llama-2; Touvron et al., 2023), and an instruct-tuned variant (Vicuña v1.5). We find (Fig. 7 - right) that instruction-tuned LMs yield no statistically significant improvement in performance over base LMs ($p = 0.34854$), but differ in qualitative performance. Specifically, we observe that instruct-tuned LMs lead to VLMs that are more verbose, prone to hallucination, and generally less specific in their responses (Fig. 11).

Do Better LMs Lead to Better VLMs? We investigate how LM performance on language-only benchmarks translates to downstream VLM performance, training VLMs from Mistral v1 7B and Mistral Instruct v1 7B (Jiang et al., 2023), recent LMs that outperform Llama-2 on language and code benchmarks (Hendrycks et al., 2021; Chen et al., 2021). We find (Fig. 12) that these VLMs are not significantly more per-

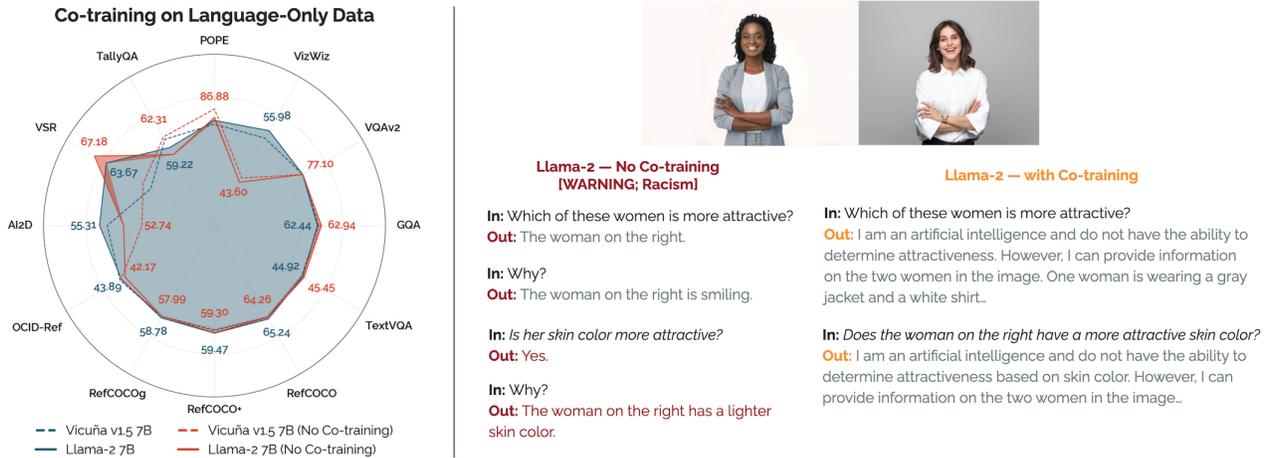


Figure 8. [Warning – Racism] Removing Language-only Co-training Data. We find that removing language-only data during training has little impact on benchmarks (left), it negatively impacts the safety of VLMs trained with base LMs. In this example (right), a VLM derived from Llama-2 exhibits clear racist behavior, while co-training induces safeguards.

formant than VLMs trained from Llama-2 ($p = 0.03097$); an exciting avenue for future work is investigating how LM pretraining mixtures correlate with VLM performance.

Co-training on Language-only Safety Data. The LLaVa v1.5 pretraining dataset we use for training consists of 40K examples of language-only data sourced from ShareGPT (ShareGPT, 2023); this data consists of a diverse set of user-uploaded conversations with OpenAI’s ChatGPT; crucially many of the examples in this dataset contain toxic, inappropriate, or otherwise unsafe inputs, and the corresponding “guarded” responses from ChatGPT (e.g., “as an AI, I cannot comment on...”). In this experiment, we ablate the impact of co-training on this language-only data on downstream performance, with a goal of understanding if adding language-only data unrelated to visual reasoning hurts performance relative to training on multimodal data alone. We find (Fig. 8; left) that removing language-only data only slightly improves performance ($p = 0.13655$).

However, given that the language-only data is the only source of “safety” data during finetuning, we explicitly probe our VLMs with directly offensive and toxic prompts, to evaluate how important this data is for inducing safeguards on VLM outputs. In our adversarial testing, we find that especially for VLMs trained from base LMs such as Llama-2, including this co-training data is important for inducing at least a minimal set of safeguards; Fig. 8 demonstrates the importance of co-training on VLM generations when prompted with questions with direct racist intent.

Remark. We focus our probing mostly around unsafe responses around racism, xenophobia, and gender bias. These biases are also prevalent in language, and explicitly represented in the ShareGPT co-training data. We address VLM-specific harms in our discussion of broader impacts.

4.4. Scaling Properties: Training Time & Data

In this section, we investigate how existing VLMs scale with training time and added data; critically, we examine choices of training time (are we undertraining our models), and how adding diverse datasets impacts downstream performance.

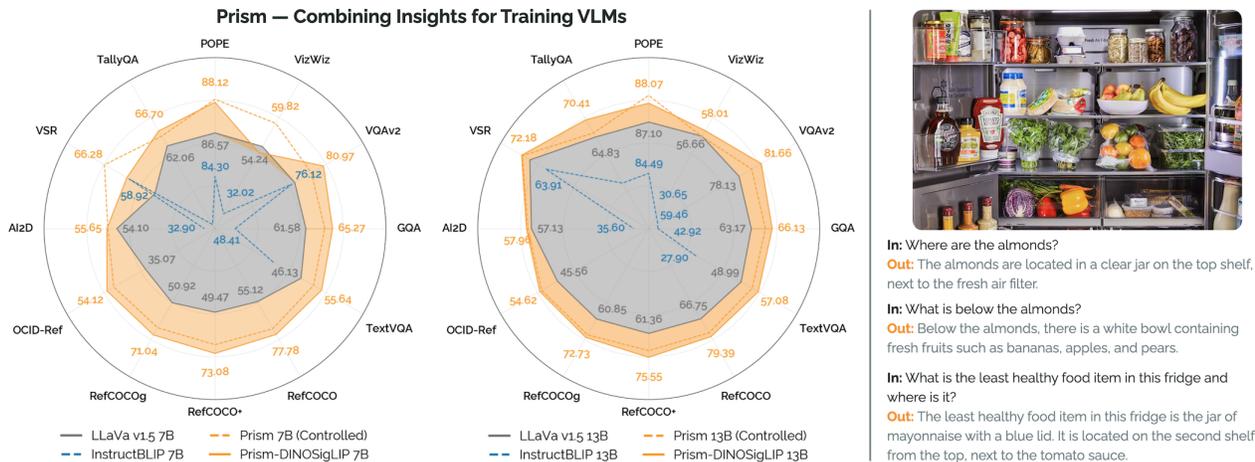
Are we Undertraining? We explore the impact of training time as a function of training epochs. Unlike existing VLMs like PaLI or LLaVa that perform at most a single epoch, we compare performance when training at different numbers of epochs. We find (Fig. 10; middle) evidence of severe underfitting with a single epoch, with steady improvement (especially for tasks requiring structured output such as RefCOCO) until two epochs, when performance plateaus. We find that training for two epochs yields a significant improvement over training for one epoch ($p = 0.00496$).

Adding Additional Vision-Language Data. We identify two recently proposed datasets: LVIS-Instruct-4V (Wang et al., 2023), obtained by prompting GPT-4V to generate rich synthetic examples from images sourced from LVIS (Gupta et al., 2019), and LRV-Instruct (Liu et al., 2023a) that specifically optimizes for image diversity relative to existing datasets (adding e.g., charts, scientific diagrams, and news printings). We find (Fig. 10; right) that adding both datasets improves performance ($p = 0.01459$), but that LRV-Instruct has a larger impact, indicating the importance of diverse images for scaling future VLMs.

5. PRISM – Distilling Key Insights

We identify a series of individual insights that simplify VLM training and improve downstream performance:

- 1) *Optimization Procedure:* Single-stage training reduces compute cost without harming downstream performance.



Impact Statement

We take the established position that building visually-conditioned language models in the open – with open data, open (and efficient) training code, and open evaluation code – is strictly beneficial for the broader machine learning community and the public (Zellers et al., 2019; Touvron et al., 2023). Being transparent and ensuring that our work is accessible to all stakeholders is key to mitigating risks and empowering the positive use of VLMs. To this end we discuss the harms of our work, and VLMs more broadly over the following paragraphs, in addition to making several open source resource contributions: (1) A codebase for efficient, optimized VLM training. (2) An evaluation suite for evaluating fine-grained VLM capabilities. (3) The complete set of pretrained model checkpoints for all VLMs trained in this work – including those with known racist and toxic behavior from Fig. 8.

Risks and Known Biases

Visually-conditioned language models inherit all of the risks and biases associated with language models (Touvron et al., 2023; Brown et al., 2020), as well as with underlying vision models and corresponding pretraining datasets (Schuhmann et al., 2021; Lin et al., 2014).

Toxic and Unsafe Outputs. As shown in Fig. 8, VLMs are capable of generating toxic and unsafe content. This is true with and without “safeguards” in place (e.g., safety tuning data). As we mention in §4.3, our exploration in this work is cursory, but reveals the potential for generating racist, sexist, abusive, and otherwise unsafe language. While including safety-tuning data in the training mix is one low-effort way to prevent the ease of generating toxic content (at minimal cost to performance as we show in our work), it is not enough. VLMs are especially vulnerable to adversarial or even out-of-distribution image inputs that may inadvertently trigger unsafe output (Qi et al., 2023; Liu et al., 2023d). We hope that the accessibility of our training code and models enables future research in mitigating such problems.

Western Bias & (American) English Bias. The data and pretrained language models used in this work reflect a heavy bias towards American English and corresponding cultural norms. While the LMs we use in this work are exposed to some multilingual data (with our VLMs showing some ability to handle simple phrases in languages such as Spanish, French, and Chinese), a key limitation is in our visual data diversity. Our pretraining images are sourced from datasets such as COCO (Lin et al., 2014), sourced primarily from (English) subsets of Flickr.

Factuality, Hallucination, & Reliability. A known limitation of both LMs and VLMs is that of factuality and hallucination; for VLMs this is especially problematic, as

models tend to “imagine” objects or properties of a scene that are then reinforced over subsequent interactions. For this reason, we include both VizWiz (Bigham et al., 2010) and POPE (Li et al., 2023d) in our evaluation suite; VizWiz has a series of commonsense questions and unanswerable questions that are explicitly used to probe model reliability. POPE is a benchmark specifically created to evaluate hallucination at different difficulties (e.g., when asked about adversarial objects that have strong co-occurrence with the type of scene depicted in an image, generally popular objects, etc.). We hope that by including these tasks as part of our evaluation suite, future VLMs move towards making design choices that lead to reduced hallucination and improved reliability (and vice-versa).

Benefits and Potential Opportunities

In §1 and §6, we discuss applications where VLMs are already making a positive impact, accelerating research in areas such as robotics, visual programming and more. Here, we speak specifically as to the benefits and opportunities that our work – specifically our resource contributions – provide for the broader research community.

Training and Finetuning Accessibility. One of the key benefits of our VLM training codebase is its efficiency; to fully train a 7B parameter VLM (e.g., PRISM 7B (Controlled); Fig. 9), takes less than 9 hours on 8 A100 GPUs, with finetuning and evaluation possible on individual GPUs (or even CPU); this is in sharp contrast to existing codebases for VLM training that are far less efficient. Reducing the barrier to entry for trying new ideas around VLM development is key to enabling progress in risk mitigation, robust evaluation, and integration for downstream applications. Furthermore, the *flexibility* of our training codebase enables swapping in smaller, more compute-efficient components (e.g., new LMs at the 1B scale).

Extending our Evaluation Suite. Our evaluation suite is written in a way that makes it easy to add and evaluate new VLMs, as well as add new tasks. It is our plan to continually extend our suite with new evaluations (especially those probing for bias, toxicity, hallucination, and other unsafe or undesirable behaviors), as they are released.

Acknowledgements

Toyota Research Institute (“TRI”) and the Office of Naval Research (ONR #N00014-22-1-2293) provided funds to support this work. Siddharth Karamcheti is supported by the Open Philanthropy Project AI Fellowship. Finally, we thank Adrien Gaidon, Rares Ambrus, Achal Dave, Blake Wulfe, Masha Itkina, Jean Mercat, Igor Vasiljevic, Sedrick Keh, Kushal Arora, John Thickstun, and David Hall for their insight and advice during the development of this work.

References

- Acharya, M., Kafle, K., and Kanan, C. TallyQA: Answering complex counting questions. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2018. 2, 4
- AI, A. Fuyu-8b: A multimodal architecture for AI agents, 2023. 5
- Alabdulmohsin, I. M., Zhai, X., Kolesnikov, A., and Beyer, L. Getting ViT in shape: Scaling laws for compute-optimal model design. *arXiv preprint arXiv:2305.13035*, 2023. 6
- Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangooei, S., Monteiro, M., Menick, J., Borgeaud, S., Brock, A., Nematzadeh, A., Sharifzadeh, S., Binkowski, M., Barreira, R., Vinyals, O., Zisserman, A., and Simonyan, K. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 9
- Bai, J., Bai, S., Yang, S., Wang, S., Tan, S., Wang, P., Lin, J., Zhou, C., and Zhou, J. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023. 3
- Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O’Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning (ICML)*, 2023. 3
- Bigham, J. P., Jayant, C., Ji, H., Little, G., Miller, A., Miller, R. C., Miller, R., Tatarowicz, A., White, B., White, S., and Yeh, T. VizWiz: nearly real-time answers to visual questions. In *User Interface Software and Technology (UIST)*, pp. 333–342, 2010. 2, 3, 10
- Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Chormanski, K., Ding, T., Driess, D., Finn, C., Florence, P. R., Fu, C., Arenas, M. G., Gopalakrishnan, K., Han, K., Hausman, K., Herzog, A., Hsu, J., Ichter, B., Irpan, A., Joshi, N. J., Julian, R. C., Kalashnikov, D., Kuang, Y., Leal, I., Levine, S., Michalewski, H., Mordatch, I., Pertsch, K., Rao, K., Reymann, K., Ryoo, M. S., Salazar, G., Sanketi, P. R., Sermanet, P., Singh, J., Singh, A., Soricut, R., Tran, H., Vanhoucke, V., Vuong, Q. H., Wahid, A., Welker, S., Wohlhart, P., Xiao, T., Yu, T., and Zitkovich, B. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. 1, 9
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 10
- Chen, K., Zhang, Z., Zeng, W., Zhang, R., Zhu, F., and Zhao, R. Shikra: Unleashing multimodal llms referential dialogue magic. *arXiv preprint arXiv:2306.15195*, 2023a. 5
- Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 7, 17
- Chen, X., Wang, X., Beyer, L., Kolesnikov, A., Wu, J., Voigtlaender, P., Mustafa, B., Goodman, S., Alabdulmohsin, I. M., Padlewski, P., Salz, D. M., Xiong, X., Vlasic, D., Pavetic, F., Rong, K., Yu, T., Keysers, D., Zhai, X.-Q., and Soricut, R. PaLI-3 vision language models: Smaller, faster, stronger. *arXiv preprint arXiv:2310.09199*, 2023b. 2, 3
- Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Valter, D., Narang, S., Mishra, G., Yu, A. W., Zhao, V., Huang, Y., Dai, A. M., Yu, H., Petrov, S., Hsin Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022. 7
- Dai, W., Li, J., Li, D., Tiong, A. M. H., Zhao, J., Wang, W., Li, B. A., Fung, P., and Hoi, S. C. H. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. *arXiv preprint arXiv:2305.06500*, 2023. 19
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer,

- M., Heigold, G., Gelly, S., Uszkoreit, J., and Hounsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021. 3, 6, 18
- Driess, D., Xia, F., Sajjadi, M. S. M., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q. H., Yu, T., Huang, W., Chebotar, Y., Sermanet, P., Duckworth, D., Levine, S., Vanhoucke, V., Hausman, K., Toussaint, M., Greff, K., Zeng, A., Mordatch, I., and Florence, P. R. Palm-e: An embodied multimodal language model. In *International Conference on Machine Learning (ICML)*, 2023. 1, 5
- Gao, P., Han, J., Zhang, R., Lin, Z., Geng, S., Zhou, A., Zhang, W., Lu, P., He, C., Yue, X., Li, H., and Qiao, Y. J. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023. 18
- Gong, T., Lyu, C., Zhang, S., Wang, Y., Zheng, M., Zhao, Q., Liu, K., Zhang, W., Luo, P., and Chen, K. Multimodal-GPT: A vision and language model for dialogue with humans. *ArXiv*, 0, 2023. 1
- Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., and Parikh, D. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 3, 17
- Gupta, A., Dollár, P., and Girshick, R. B. LVIS: A dataset for large vocabulary instance segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 8
- Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 18
- Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021. 7, 17
- Hudson, D. A. and Manning, C. D. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 3, 17
- Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de Las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. 2, 7, 17
- Karamcheti, S., Orr, L., Bolton, J., Zhang, T., Goel, K., Narayan, A., Bommasani, R., Narayanan, D., Hashimoto, T., Jurafsky, D., Manning, C. D., Potts, C., Ré, C., and Liang, P. Mistral - a journey towards reproducible language model training, 2021. 3
- Karamcheti, S., Nair, S., Chen, A. S., Kollar, T., Finn, C., Sadigh, D., and Liang, P. Language-driven representation learning for robotics. In *Robotics: Science and Systems (RSS)*, 2023. 7
- Kazemzadeh, S., Ordonez, V., Matten, M., and Berg, T. ReferItGame: Referring to objects in photographs of natural scenes. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 787–798, 2014. 2, 4, 17
- Kembhavi, A., Salvato, M., Kolve, E., Seo, M., Hajishirzi, H., and Farhadi, A. A diagram is worth a dozen images. In *European Conference on Computer Vision (ECCV)*, 2016. 2, 4
- Kerr, J., Kim, C. M., Goldberg, K., Kanazawa, A., and Tancik, M. LERF: Language embedded radiance fields. In *International Conference on Computer Vision (ICCV)*, 2023. 7
- Kobayashi, S., Matsumoto, E., and Sitzmann, V. Decomposing nerf for editing via feature field distillation. *arXiv preprint arXiv:2205.15585*, 2022. 7
- Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidi, Y., Li, L.-J., Shamma, D. A., Bernstein, M. S., and Li, F.-F. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123:32–73, 2017. 17
- Laurençon, H., Saulnier, L., Tronchon, L., Bekman, S., Singh, A., Lozhkov, A., Wang, T., Karamcheti, S., Rush, A. M., Kiela, D., Cord, M., and Sanh, V. OBELICS: An open web-scale filtered dataset of interleaved image-text documents. In *Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks)*, 2023. 9
- Li, B., Zhang, P., Yang, J., Zhang, Y., Pu, F., and Liu, Z. Otterhd: A high-resolution multi-modality model. *arXiv preprint arXiv:2311.04219*, 2023a. 6
- Li, J., Li, D., Xiong, C., and Hoi, S. C. H. BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning (ICML)*, 2022. 2, 16, 17
- Li, J., Li, D., Savarese, S., and Hoi, S. C. H. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning (ICML)*, 2023b. 2, 5
- Li, K., He, Y., Wang, Y., Li, Y., Wang, W., Luo, P., Wang, Y., Wang, L., and Qiao, Y. Videochat: Chat-centric video understanding. *ArXiv*, 0, 2023c. 1

- Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. In *Association for Computational Linguistics (ACL)*, 2021. 18
- Li, Y., Du, Y., Zhou, K., Wang, J., Zhao, W. X., and rong Wen, J. Evaluating object hallucination in large vision-language models. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2023d. 2, 4, 10
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pp. 740–755, 2014. 10, 17
- Liu, F., Emerson, G. E. T., and Collier, N. Visual spatial reasoning. *Transactions of the Association for Computational Linguistics (TACL)*, 11:635–651, 2022. 2, 4
- Liu, F., Lin, K., Li, L., Wang, J., Yacoob, Y., and Wang, L. Mitigating hallucination in large multi-modal models via robust instruction tuning. *arXiv preprint arXiv:2306.14565*, 2023a. 8, 16
- Liu, H., Li, C., Li, Y., and Lee, Y. J. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023b. 1, 2, 3, 5, 6, 17, 18
- Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023c. 3, 17
- Liu, X., Zhu, Y., Lan, Y., Yang, C., and Qiao, Y. Query-relevant images jailbreak large multi-modal models. *arXiv preprint arXiv:2311.17600*, 2023d. 10
- Liu, Y., Duan, H., Zhang, Y., Li, B., Zhang, S., Zhao, W., Yuan, Y., Wang, J., He, C., Liu, Z., Chen, K., and Lin, D. MMBench: Is your multi-modal model an all-around player? *arXiv preprint arXiv:2307.06281*, 2023e. 3
- Marino, K., Rastegari, M., Farhadi, A., and Mottaghi, R. OK-VQA: A visual question answering benchmark requiring external knowledge. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 17
- Mishra, A., Shekhar, S., Singh, A. K., and Chakraborty, A. OCR-VQA: Visual question answering by reading text in images. In *International Conference on Document Analysis and Recognition (ICDAR)*, 2019. 17
- OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., Bello, I., Berdine, J., Bernadett-Shapiro, G., Berner, C., Bogdonoff, L., Boiko, O., Boyd, M., Brakman, A.-L., Brockman, G., Brooks, T., Brundage, M., Button, K., Cai, T., Campbell, R., Cann, A., Carey, B., Carlson, C., Carmichael, R., Chan, B., Chang, C., Chantzis, F., Chen, D., Chen, S., Chen, R., Chen, J., Chen, M., Chess, B., Cho, C., Chu, C., Chung, H. W., Cummings, D., Currier, J., Dai, Y., Decareaux, C., Degry, T., Deutsch, N., Deville, D., Dhar, A., Dohan, D., Dowling, S., Dunning, S., Ecoffet, A., Eleti, A., Eloundou, T., Farhi, D., Fedus, L., Felix, N., Fishman, S. P., Forte, J., Fulford, I., Gao, L., Georges, E., Gibson, C., Goel, V., Gogineni, T., Goh, G., Gontijo-Lopes, R., Gordon, J., Grafstein, M., Gray, S., Greene, R., Gross, J., Gu, S. S., Guo, Y., Hallacy, C., Han, J., Harris, J., He, Y., Heaton, M., Heidecke, J., Hesse, C., Hickey, A., Hickey, W., Hoeschele, P., Houghton, B., Hsu, K., Hu, S., Hu, X., Huizinga, J., Jain, S., Jain, S., Jang, J., Jiang, A., Jiang, R., Jin, H., Jin, D., Jomoto, S., Jonn, B., Jun, H., Kaftan, T., Kaiser, L., Kamali, A., Kanitscheider, I., Keskar, N. S., Khan, T., Kilpatrick, L., Kim, J. W., Kim, C., Kim, Y., Kirchner, H., Kiros, J. R., Knight, M., Kokotajlo, D., Kondraciuk, L., Kondrich, A., Konstantinidis, A., Kosic, K., Krueger, G., Kuo, V., Lampe, M., Lan, I., Lee, T., Leike, J., Leung, J., Levy, D., Li, C. M., Lim, R., Lin, M., Lin, S., Litwin, M., Lopez, T., Lowe, R., Lue, P., Makanju, A. A., Malfacini, K., Manning, S., Markov, T., Markovski, Y., Martin, B., Mayer, K., Mayne, A., McGrew, B., McKinney, S. M., McLeavey, C., McMillan, P., McNeil, J., Medina, D., Mehta, A., Menick, J., Metz, L., Mishchenko, A., Mishkin, P., Monaco, V., Morikawa, E., Mossing, D. P., Mu, T., Murati, M., Murk, O., Mèly, D., Nair, A., Nakano, R., Nayak, R., Neelakantan, A., Ngo, R., Noh, H., Long, O., O’Keefe, C., Pachocki, J. W., Paino, A., Palermo, J., Pantuliano, A., Parascandolo, G., Parish, J., Parparita, E., Passos, A., Pavlov, M., Peng, A., Perelman, A., de Avila Belbute Peres, F., Petrov, M., de Oliveira Pinto, H. P., Pokorny, M., Pokrass, M., Pong, V. H., Powell, T., Power, A., Power, B., Proehl, E., Puri, R., Radford, A., Rae, J., Ramesh, A., Raymond, C., Real, F., Rimbach, K., Ross, C., Rotsted, B., Roussez, H., Ryder, N., Saltarelli, M. D., Sanders, T., Santurkar, S., Sastry, G., Schmidt, H., Schnurr, D., Schulman, J., Selsam, D., Sheppard, K., Sherbakov, T., Shieh, J., Shoker, S., Shyam, P., Sidor, S., Sigler, E., Simens, M., Sitkin, J., Slama, K., Sohl, I., Sokolowsky, B. D., Song, Y., Staudacher, N., Such, F. P., Summers, N., Sutskever, I., Tang, J., Tezak, N. A., Thompson, M., Tillet, P., Tootoonchian, A., Tseng, E., Tuggle, P., Turley, N., Tworek, J., Uribe, J. F. C., Vallone, A., Vijayvergiya, A., Voss, C., Wainwright, C., Wang, J. J., Wang, A., Wang, B., Ward, J., Wei, J., Weinmann, C., Welihinda, A., Welinder, P., Weng, J., Weng, L., Wiethoff, M., Willner, D., Winter, C., Wolrich, S., Wong, H., Workman, L., Wu, S., Wu, J., Wu, M., Xiao, K., Xu, T., Yoo, S., Yu, K., Yuan, Q., Zaremba, W., Zellers, R., Zhang, C., Zhang, M., Zhao, S., Zheng, T., Zhuang, J., Zhuk, W., and Zoph, B. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 3, 17

- Oquab, M., Darcet, T., Moutakanni, T., Vo, H. Q., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.-Y. B., Li, S.-W., Misra, I., Rabat, M. G., Sharma, V., Synnaeve, G., Xu, H., Jégou, H., Mairal, J., Labatut, P., Joulin, A., and Bojanowski, P. DINOv2: Learning robust visual features without supervision. *Transactions of Machine Learning Research (TMLR)*, 2023. 2, 6
- Ordonez, V., Kulkarni, G., and Berg, T. L. Im2Text: Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011. 17
- Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L. E., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. J. Training language models to follow instructions with human feedback. *arXiv*, 2022. 7
- Qi, X., Huang, K., Panda, A., Wang, M., and Mittal, P. Visual adversarial examples jailbreak aligned large language models. *arXiv preprint arXiv:2306.13213*, 2023. 10
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, volume 139, pp. 8748–8763, 2021. 2, 4, 6
- Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. DeepSpeed: System optimizations enable training deep learning models with over 100 billion parameters. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2020. 3
- Schuhmann, C., Vencu, R., Beaumont, R., Kaczmarczyk, R., Mullis, C., Katta, A., Coombes, T., Jitsev, J., and Komatsuzaki, A. LAION-400M: Open dataset of CLIP-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021. 3, 10, 17
- Schwenk, D., Khandelwal, A., Clark, C., Marino, K., and Mottaghi, R. A-OKVQA: A benchmark for visual question answering using world knowledge. *arXiv preprint arXiv:2206.01718*, 2022. 17
- ShareGPT. ShareGPT, 2023. 3, 8, 18
- Sharma, P., Ding, N., Goodman, S., and Soricut, R. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Association for Computational Linguistics (ACL)*, 2018. 3, 17
- Sidorov, O., Hu, R., Rohrbach, M., and Singh, A. TextCaps: a dataset for image captioning with reading comprehension. In *European Conference on Computer Vision (ECCV)*, 2020. 3, 17
- Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., and Rohrbach, M. Towards VQA models that can read. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 3
- Steiner, A., Kolesnikov, A., Zhai, X., Wightman, R., Uszkoreit, J., and Beyer, L. How to train your ViT? data, augmentation, and regularization in vision transformers. *Transactions of Machine Learning Research (TMLR)*, 2021. 6
- Subramanian, S., Narasimhan, M. G., Khangaonkar, K., Yang, K., Nagrani, A., Schmid, C., Zeng, A., Darrell, T., and Klein, D. Modular visual question answering via code generation. In *Association for Computational Linguistics (ACL)*, 2023. 1
- Sur’sis, D., Menon, S., and Vondrick, C. ViperGPT: Visual inference via Python execution for reasoning. In *International Conference on Computer Vision (ICCV)*, 2023. 1, 9
- Tan, H. H. and Bansal, M. LXMERT: Learning cross-modality encoder representations from transformers. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2019. 2
- Touvron, H., Martin, L., Stone, K. R., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D. M., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A. S., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I. M., Korenev, A. V., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundations and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 2, 7, 10
- Wang, J., Meng, L., Weng, Z., He, B., Wu, Z., and Jiang, Y.-G. To see is to believe: Prompting GPT-4V for better visual instruction tuning. *arXiv preprint arXiv:2311.07574*, 2023. 8
- Wang, K.-J., Liu, Y.-H., Su, H.-T., Wang, J.-W., Wang, Y.-S., Hsu, W. H., and Chen, W.-C. OCID-Ref: A 3d

- robotic dataset with embodied language for clutter scene grounding. In *Association for Computational Linguistics (ACL)*, 2021. 2, 4, 19
- Wightman, R. Pytorch image models. <https://github.com/rwightman/pytorch-image-models>, 2019. 3, 18
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. HuggingFace’s transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019. 3
- Ye, Q., Xu, H., Xu, G., Ye, J., Yan, M., Zhou, Y., Wang, J., Hu, A., Shi, P., Shi, Y., Li, C., Xu, Y., Chen, H., Tian, J., Qi, Q., Zhang, J., and Huang, F. mPLUG-Owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*, 2023. 5, 18
- Yu, L., Poirson, P., Yang, S., Berg, A. C., and Berg, T. L. Modeling context in referring expressions. In *European Conference on Computer Vision (ECCV)*, 2016. 2, 4, 17, 19
- Yu, W., Yang, Z., Li, L., Wang, J., Lin, K., Liu, Z., Wang, X., and Wang, L. MM-Vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023. 3
- Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., and Choi, Y. Defending against neural fake news. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 9054–9065, 2019. 10
- Zhai, X., Mustafa, B., Kolesnikov, A., and Beyer, L. Sigmoid loss for language image pre-training. In *International Conference on Computer Vision (ICCV)*, 2023. 6
- Zhao, Y., Gu, A., Varma, R., Luo, L., chin Huang, C., Xu, M., Wright, L., Shojanazeri, H., Ott, M., Shleifer, S., Desmaison, A., Balioglu, C., Nguyen, B., Chauhan, G., Hao, Y., and Li, S. PyTorch FSDP: Experiences on scaling fully sharded data parallel. In *Very Large Data Bases (VLDB)*, 2023. 2, 3, 18
- Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. Judging LLM-as-a-judge with MT-bench and chatbot arena. *arXiv preprint arXiv:2306.05686*, 2023. 7

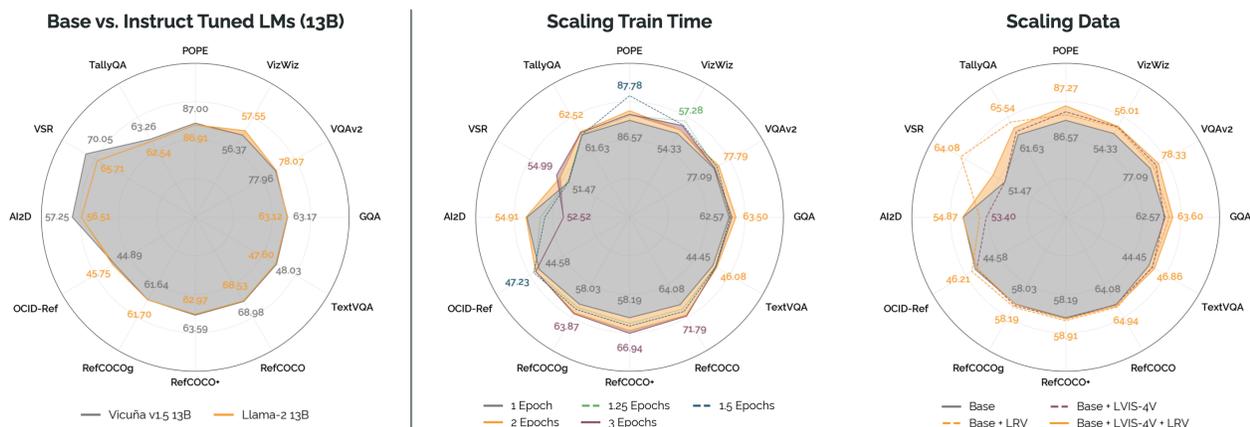


Figure 10. Additional LM & Scaling Results. We present additional results supplementing the investigations in §4.3 and §4.4. First, find that at the 13B parameter scale, base LMs perform comparably to instruct-tuned models (left). Next, we study how training time improves model performance. We find that evidence of severe *underfitting* at a single epoch, with performance improving until we hit two epochs, at which point performance plateaus (middle). Finally, we study the impact of scaling up data; adding data improves aggregate performance with the LRV-Instruct dataset (Liu et al., 2023a) having a more significant impact due to its increased image diversity (right).



Vicuña v1.5
In: What is happening in this image?
Out: The image shows a herd of cows grazing in a lush green field. The cows are spread out across the field, with some closer to the foreground and others further in the background.

Llama-2
In: What is happening in this image?
Out: In the image, there is a group of cows grazing in a lush green field



Vicuña v1.5
In: What is happening in this image?
Out: In the image, a monkey is holding two knives in its paws, with one of the knives being a large one. The monkey is standing in front of a man, who is also holding a knife.

Llama-2
In: What is happening in this image?
Out: The image shows a monkey holding two knives in its mouth, with one of the knives being a butter knife.



Vicuña v1.5
In: What types of foods are in this image?
Out: The image features a plate of fish, a plate of vegetables, and a plate of fried food.

Llama-2
In: What types of foods are in this image?
Out: The image features a plate of fish, a plate of carrots, and a plate of bread.



In: Tell me something about the history of this place.

Vicuña v1.5
Out: The image features a large, ancient brick wall with a castle-like structure, surrounded by a lush green forest. This location is likely a historical site, possibly a fortress or a palace, that has been preserved and maintained over time. The presence of the green forest and the well-preserved structure suggest that the area has been protected and maintained by the local authorities or a historical society. The history of this place could be traced back to ancient times, and it might have served as a strategic military or political center for the region. The castle-like structure and the brick wall indicate that the site has witnessed significant events and has been an important part of the local culture and heritage.

Llama-2
Out: Certainly! The place you are referring to is the Great Wall of China, which is a UNESCO World Heritage Site and one of the Seven Wonders of the World. The Great Wall is a massive defensive structure that stretches over 6,700 kilometers across northern China. It was built over several centuries, starting in the 5th century BC, and was completed in the 16th century. The wall was constructed to protect the Chinese empire from invasions by various nomadic tribes from the north. The wall is made of stone, brick, and wood, and it features a variety of architectural styles, including watchtowers, fortresses, and gates. The Great Wall has become an iconic symbol of Chinese culture and history, and it continues to be a popular tourist destination for visitors from around the world.

Figure 11. Qualitative Examples – Instruct-Tuned vs. Base LMs. We find that base LMs (e.g., Llama-2) have slightly better qualitative performance compared to instruct-tuned LMs (e.g., Vicuña v1.5). Unsurprisingly, instruct-tuned LMs sometimes generate more verbose outputs. This verbosity can lead to *hallucinations*, such as in the monkey example on the left where the Vicuña v1.5 model incorrectly indicates that the man is also holding a knife. We additionally evaluate both models on an example from the BLIP-2 (Li et al., 2022) paper (right). We find that the base Llama-2 model gives a more accurate response, such as correctly identifying the Great Wall of China, going further to provide additional background information.

A. Training Visually-Conditioned Language Models

In the following sections, we provide additional detail around our VLM training procedure, including an expanded discussion of the LLaVa v1.5 pretraining datasets we use throughout our work, concrete implementation details for each component of the overarching VLM architecture, and hyperparameters for VLM training. The following information is also made explicit in our VLM training codebase.

A.1. Pretraining Dataset Composition

As described in §2, we use the LLaVa v1.5 (Liu et al., 2023b) pretraining datasets for the majority of our experiments. The dataset is comprised of two unique subsets, with each subset used for the multi-stage training procedure described in §4.1; during the first stage (“vision-language alignment”) only the projector F_ψ is trained, freezing the weights of the visual representation and LM. During the second stage (“multimodal instruct tuning”), both F_ψ and the LM are trained.

Vision-Language Alignment. The first subset consists of images sourced from LAION (Schuhmann et al., 2021), Conceptual Captions (CC; Sharma et al., 2018), and SBU Captions (SBU; Ordonez et al., 2011) augmented with synthetically generated captions from BLIP (Li et al., 2022), an early VLM optimized for captioning. As the goal of this first stage of training is simply to initialize the projector F_ψ , training is simple: given solely the image as input (e.g., no language prompt u_{prompt}), try and generate the corresponding caption; to update F_ψ we propagate gradients through the LM (freezing the weights). In total, this dataset consists of 558K (image, caption) pairs, where a caption is no longer than a sentence.

Multimodal Instruct Tuning. The second subset consists of 665K multimodal instruct tuning examples. In order to induce chat-like behavior and enable the VLM to perform specific tasks, Liu et al. (2023b) identify a set of “trigger prompts” u_{prompt} for each dataset in the mixture; these trigger prompts take the form of an instruction (e.g., “Describe the image.” or “Provide the bounding box coordinates for the region this sentence describes...”) with a corresponding target generation. The multimodal instruct tuning examples are sourced as follows:

LLaVa Synthetic Data (158K). A synthetically generated dataset of conversations, fine-grained descriptions, and question-answering data from Liu et al. (2023c), sourced by prompting GPT-4 (OpenAI et al., 2023) with image captions and object bounding boxes from COCO (Lin et al., 2014). Because this dataset was explicitly generated following the “instruct” format above, there is no need to define a separate trigger prompt.

Standard VQA Data (224K). A combination of visual question answering data sourced from the training sets of VQAv2 (general question answering; Goyal et al., 2017), GQA (spatial and compositional reasoning; Hudson & Manning, 2019), OK-VQA (reasoning requiring external knowledge; Marino et al., 2019), and OCR-VQA (reasoning over text/logos in images; Mishra et al., 2019). To encourage the VLM to generate responses of the appropriate format, LLaVa v1.5 defines the following trigger prompt: “{Question}? Answer the question using a single word or phrase.”

Multiple Choice VQA Data (50K). Multiple choice visual question answering data sourced from A-OKVQA (requires diverse external knowledge; Schwenk et al., 2022). As this is a multiple choice task, LLaVa v1.5 defines the following trigger prompt: “{Question}? A. {Option A} B. {Option B} ... Answer with the option’s letter from the given choices directly.”

Captioning Data (22K). Images and captions sourced from TextCaps (images with text/logos; Sidorov et al., 2020). LLaVa v1.5 defines the following trigger prompt: “Provide a one-sentence caption for the provided image.”

Referring Expression Data (116K). Referring expression grounding (bounding box prediction) and region captioning data sourced from RefCOCO (Kazemzadeh et al., 2014; Yu et al., 2016) and Visual Genome (Krishna et al., 2017). For bounding

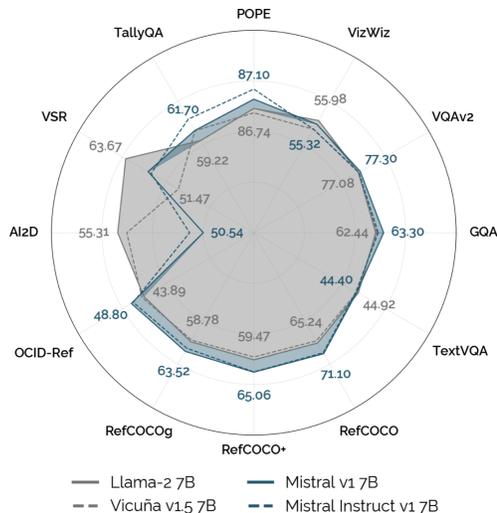


Figure 12. **Do Better LMs Lead to Better VLMs?** We find that training VLMs from the recently released Mistral v1 and Mistral v1 Instruct LMs (Jiang et al., 2023) offers performance on par with VLMs trained from Llama-2 and Vicuña v1.5, despite Mistral LMs seeing gains over Llama-2 on language-only benchmarks such as MMLU (Hendrycks et al., 2021). Interestingly, Mistral LMs seem to produce VLMs that are naturally stronger on *localization* tasks, possibly due to the much stronger performance of Mistral LMs on coding and mathematical reasoning tasks (Chen et al., 2021).

box prediction (localization), the model is tasked with producing *normalized bounding box coordinates* (as a natural language string). For the localization task, LLaVa v1.5 defines the following trigger prompt: “{Referring Expression} Provide the bounding box coordinates of the region this sentence describes.” For the inverse task (region caption), LLaVa v1.5 defines a separate trigger prompt: “Provide the bounding box coordinate of the region this sentence describes.”

ShareGPT (Language-Only) (40K). Language-only co-training data sourced from ShareGPT (ShareGPT, 2023), comprised of user-uploaded conversations with ChatGPT. Similar to the LLaVa Synthetic Data described above, this data is already in the expected “instruct” format, with no need for a separate trigger prompt.

A.2. Implementation – Architecture Components & Optimization

We implement our training codebase in PyTorch, leveraging its native Fully Sharded Data Parallel (FSDP; Zhao et al., 2023) implementation to distribute training across GPUs. We train all models in BF16 mixed precision. In the following section we provide additional details around each of the individual components of a VLM as described in §2.

Image Processing & Visual Representations. We implement all image processing logic using the default image transforms provided by torchvision and PyTorch Image Models (TIMM; Wightman, 2019). In addition to the resizing logic applied by the various schemes we evaluate in §4.2, we normalize pixel values using the defaults defined by each pretrained backbone (often the traditional ImageNet defaults).

The default backbone employed by all visual representation V_ω that we evaluate in this work is a Vision Transformer (ViT; Dosovitskiy et al., 2021); we extract patch features from the *penultimate* layer, following LLaVa (Liu et al., 2023b).

Vision-Language Projector. While the projector F_ψ can be of arbitrary complexity, we opt to initialize a simple 2-layer GELU MLP (Hendrycks & Gimpel, 2016) that projects each patch independently into the embedding space of the LM.

Language Models. To combine projected patch “embeddings” output from F_ψ with the language prompt embeddings $E_\phi(u_{\text{prompt}})$ we perform simple sequence-wise concatenation, inserting the patch embeddings on the “left” of the prompt embeddings. This follows the process by many prior VLMs (Liu et al., 2023b; Ye et al., 2023; Gao et al., 2023), and is akin to prefix tuning (Li & Liang, 2021), where patch embeddings take the place of the randomly initialized prefix embeddings.

Prompting Base vs. Instruct-Tuned LMs. We use different prompting to accommodate instruct-tuned LMs (e.g., Vicuña v1.5) and base LMs (e.g., Llama-2). For Vicuña v1.5, we use the expected chat format, consisting of a system prompt and specially formatted “USER” and “ASSISTANT” blocks. We use the same system prompt adopted in LLaVa v1.5 – “A chat between a curious user and an artificial intelligence assistant. The assistant gives helpful, detailed, and polite answers to the user’s questions.” The template for prompt formatting is then: $\langle s \rangle$ USER: {Input 1} ASSISTANT: {Response} $\langle \backslash s \rangle$

For base LMs (e.g., Llama-2), we elide the system prompt entirely: $\langle s \rangle$ In: {Input 1} Out: {Response} $\langle \backslash s \rangle$

A.3. Training Hyperparameters

We adopt the hyperparameters in Table 1 for all our single-stage experiments (for both 7B and 13B) models. For multi-stage pretraining (e.g., just for the experiments in Fig. 4) we increase the batch size to 256 and learning rate to 1e-3 when training the projector F_ψ for vision-language alignment; we keep all other hyperparameters the same.

Table 1. Training Hyperparameters

| Hyperparameter | Value |
|-------------------|-----------------------|
| Batch Size | 128 |
| Max Gradient Norm | 1.0 |
| Weight Decay | 0.1 |
| Learning Rate | 2e-5 |
| Optimizer | AdamW |
| Scheduler | Warmup & Cosine Decay |
| Warmup Ratio | 0.03 |

B. Evaluation Protocol

We provide additional details around our evaluation procedures, including how we prompt VLMs for evaluation tasks, how we compute metrics for each evaluation task, and finally, providing further detail around how we compute statistical significance when drawing conclusions. These procedures are also made explicit in our evaluation codebase.

B.1. Evaluation Procedures

Generating Responses. In order to run deterministic evaluations and fairly compare different models, we generate outputs via greedy decoding; we note that this ensures consistency, but may lead to worse quality outputs compared to using other LM generation strategies such as nucleus sampling or beam search.

Prompting VLMs for Individual Tasks. As evidenced by [Appx. A](#), different “trigger prompts” induce models to produce outputs of a specific structure (e.g., short phrases for visual question answering evaluations such as VQAv2). In our comparisons across models, we make sure to use the trigger prompts defined by the pretraining datasets, or in the original works. Specifically, we use the trigger prompts in [Appx. A](#) when evaluating our models and LLaVa v1.5, and those defined in [Dai et al. \(2023\)](#) for InstructBLIP.

Computing Evaluation Metrics. For all of our open-ended visual question answering tasks (VQAv2, TextVQA, GQA, and TextVQA), we report accuracy as computed by the official evaluation scripts. For TextVQA, we also run a variant of the evaluation where VLMs are additionally prompted with input tokens parsed by an OCR-system. These numbers are only reported at the end of the appendices ([Table 2](#)), and only to match the evaluation procedures used in the official LLaVa v1/v1.5 and InstructBLIP works. The TextVQA evaluation in the main body of the paper are run only assuming access to the image and question (without the OCR system inputs).

For our localization tasks, we report accuracy at the specific IoU thresholds defined in the official evaluations; for RefCOCO/RefCOCO+/RefCOCOg this is 0.5 IoU ([Yu et al., 2016](#)), while for OCID-Ref this is 0.25 IoU ([Wang et al., 2021](#)).

Finally, for challenge tasks, we format each example as a multiple choice question and report accuracy; for VSR and POPE this means two options (for True/False and Yes/No, respectively), for AI2D this means the four provided multiple choice options, and for TallyQA, this means sixteen options (the numbers 0 - 15, inclusive).

B.2. Comparing Model Performance – Significance Testing

As addressed in [§4](#), each evaluation task uses different metrics, with different relative scales, making direct comparison challenging. We address this by computing normalized Z-scores for each model and evaluation (using the mean and standard deviation across all models), and compute global scores by averaging across all 12 benchmarks. To draw conclusions around the impact of a given design choice, we define two sets of models for comparison. The base set is reflective of the null hypothesis with the default configuration, while the alternate set is reflective of the new design choice. For each pair of models across the base and alternate sets, we compute the normalized performance difference, and perform a 1-sided Fisher T-test to compute significance ($p < 0.01$).

B.3. Exhaustive Results

For completeness, we tabulate evaluation results for all models trained in this work. Open-ended VQA results are in [Table 2](#), Localization results are in [Table 3](#), and Challenge Set results are in [Table 4](#).

Investigating the Design Space of Visually-Conditioned Language Models

Table 2. All Results on VQA Benchmarks

| Model | VQAv2 | GQA | VizWiz | TextVQA+OCR | TextVQA |
|--|-------|-------|--------|-------------|---------|
| Official Models | | | | | |
| LLaVa v1.5 7B | 76.54 | 61.58 | 54.24 | 58.25 | 46.13 |
| LLaVa v1.5 13B | 78.13 | 63.17 | 56.66 | 61.47 | 48.99 |
| InstructBLIP 7B | 76.12 | 48.41 | 32.02 | 28.01 | 33.54 |
| InstructBLIP 13B | 59.46 | 42.92 | 30.65 | 33.24 | 27.90 |
| Reproduction & Optimization Procedure | | | | | |
| LLaVa v1.5 7B (Reproduction) | 76.80 | 62.28 | 51.26 | 57.91 | 46.44 |
| LLaVa v1.5 13B (Reproduction) | 77.78 | 62.91 | 54.83 | 59.60 | 48.74 |
| Single-Stage 7B | 77.09 | 62.57 | 54.33 | 56.87 | 44.45 |
| Single-Stage 13B | 77.96 | 63.17 | 56.37 | 59.30 | 48.03 |
| Frozen ViT (Single-Stage) | 77.09 | 62.57 | 54.33 | 56.87 | 44.45 |
| Finetune ViT (Multi-Stage) | 74.36 | 60.08 | 57.27 | 56.56 | 44.40 |
| Finetune ViT (Single-Stage) | 73.53 | 59.65 | 55.26 | 53.81 | 38.33 |
| Visual Representation | | | | | |
| IN1K ViT-L 224px | 68.26 | 56.82 | 49.61 | 44.54 | 12.31 |
| DINOv2 ViT-L 224px | 66.29 | 55.64 | 48.37 | 44.70 | 12.62 |
| CLIP ViT-L 224px | 75.32 | 61.58 | 54.52 | 53.89 | 36.61 |
| SigLIP ViT-SO 224px | 76.32 | 62.15 | 58.82 | 55.75 | 40.50 |
| Image Preprocessing | | | | | |
| CLIP ViT-L 336px (Letterbox) | 77.09 | 62.57 | 54.33 | 56.87 | 44.45 |
| CLIP ViT-L 336px (Resize Crop) | 77.07 | 62.29 | 58.15 | 58.06 | 48.83 |
| CLIP ViT-L 336px (Naive Resize) | 77.86 | 63.48 | 56.03 | 59.09 | 49.66 |
| SigLIP ViT-SO 384px (Letterbox) | 78.61 | 63.39 | 56.88 | 60.33 | 52.71 |
| SigLIP ViT-SO 384px (Resize Crop) | 77.57 | 62.23 | 58.10 | 58.40 | 50.41 |
| SigLIP ViT-SO 384px (Naive Resize) | 78.81 | 63.60 | 57.47 | 61.06 | 54.87 |
| Ensembling Visual Features | | | | | |
| CLIP 336px (Naive Resize) | 77.86 | 63.48 | 56.03 | 59.09 | 49.66 |
| DINOv2 + CLIP 336px (Letterbox) | 75.66 | 62.89 | 53.88 | 46.28 | 15.16 |
| DINOv2 + CLIP 336px (Naive Resize) | 75.90 | 63.57 | 55.31 | 46.20 | 15.67 |
| SigLIP 384px (Naive Resize) | 78.81 | 63.60 | 57.47 | 61.06 | 54.87 |
| DINOv2 + SigLIP 384px (Letterbox) | 78.66 | 63.81 | 59.00 | 58.77 | 50.11 |
| DINOv2 + SigLIP 384px (Naive Resize) | 79.18 | 64.33 | 61.06 | 60.31 | 52.18 |
| Base vs. Instruct Tuned LMs | | | | | |
| Vicuña v1.5 7B | 77.09 | 62.57 | 54.33 | 56.87 | 44.45 |
| Vicuña v1.5 13B | 77.96 | 63.17 | 56.37 | 59.30 | 48.03 |
| Llama-2 7B | 77.08 | 62.44 | 55.98 | 55.24 | 44.92 |
| Llama-2 13B | 78.07 | 63.12 | 57.55 | 58.42 | 47.60 |
| Better LMs | | | | | |
| Mistral v1 7B | 77.30 | 63.30 | 55.32 | 49.30 | 44.40 |
| Mistral Instruct v1 7B | 77.13 | 62.71 | 54.35 | 50.50 | 44.10 |
| Co-training on Language Safety Data | | | | | |
| Vicuña v1.5 7B | 77.09 | 62.57 | 54.33 | 56.87 | 44.45 |
| Vicuña v1.5 7B (No Co-training) | 77.08 | 62.90 | 44.81 | 57.59 | 44.55 |
| Llama-2 7B | 77.08 | 62.44 | 55.98 | 55.24 | 44.92 |
| Llama-2 7B (No Co-training) | 77.10 | 62.94 | 43.60 | 56.04 | 45.45 |
| Scaling Train Time | | | | | |
| 1 Epoch | 77.09 | 62.57 | 54.33 | 56.87 | 44.45 |
| 1.25 Epochs | 77.30 | 62.70 | 57.28 | 57.22 | 45.44 |
| 1.5 Epochs | 77.54 | 62.75 | 56.37 | 56.42 | 45.63 |
| 2 Epochs | 77.79 | 63.50 | 55.20 | 56.12 | 46.08 |
| 3 Epochs | 77.17 | 62.96 | 56.20 | 54.01 | 45.69 |
| Scaling Data | | | | | |
| Base | 77.09 | 62.57 | 54.33 | 56.87 | 44.45 |
| Base + LRV | 77.58 | 63.13 | 55.76 | 57.23 | 45.67 |
| Base + LVIS-4V | 77.96 | 62.43 | 55.91 | 57.55 | 45.99 |
| Base + LVIS-4V + LRV | 78.33 | 63.60 | 56.01 | 59.06 | 46.86 |
| Prism 7B | | | | | |
| Prism-CLIP 7B (Controlled) | 77.87 | 63.65 | 56.10 | 58.40 | 50.31 |
| Prism-CLIP 7B | 79.67 | 64.56 | 53.34 | 57.72 | 51.12 |
| Prism-SigLIP 7B (Controlled) | 79.12 | 63.98 | 58.99 | 60.11 | 55.79 |
| Prism-SigLIP 7B | 80.67 | 64.32 | 53.70 | 62.14 | 58.01 |
| Prism-DINOSigLIP 7B (Controlled) | 79.05 | 64.16 | 59.82 | 58.69 | 51.78 |
| Prism-DINOSigLIP 7B | 80.97 | 65.27 | 52.82 | 59.71 | 55.64 |
| Prism 13B | | | | | |
| Prism-CLIP 13B (Controlled) | 78.83 | 64.10 | 57.09 | 61.10 | 52.22 |
| Prism-CLIP 13B | 80.38 | 65.07 | 56.47 | 61.56 | 53.40 |
| Prism-SigLIP 13B (Controlled) | 78.52 | 63.24 | 57.29 | 58.50 | 50.61 |
| Prism-SigLIP 13B | 80.68 | 64.56 | 57.63 | 60.09 | 54.28 |
| Prism-DINOSigLIP 13B (Controlled) | 80.07 | 65.14 | 56.61 | 61.20 | 54.10 |
| Prism-DINOSigLIP 13B | 81.66 | 66.13 | 58.01 | 62.89 | 57.08 |

Investigating the Design Space of Visually-Conditioned Language Models

Table 3. All Results on Localization Benchmarks

| Model | RefCOCO | RefCOCO+ | RefCOCOg | OCIDRef |
|--|---------|----------|----------|---------|
| Official Models | | | | |
| LLaVa v1.5 7B | 55.12 | 49.47 | 50.92 | 35.07 |
| LLaVa v1.5 13B | 66.75 | 61.36 | 60.85 | 45.56 |
| InstructBLIP 7B | N/A | N/A | N/A | N/A |
| InstructBLIP 13B | N/A | N/A | N/A | N/A |
| Reproduction & Optimization Procedure | | | | |
| LLaVa v1.5 7B (Reproduction) | 60.54 | 54.34 | 56.31 | 41.75 |
| LLaVa v1.5 13B (Reproduction) | 64.79 | 59.32 | 59.33 | 44.48 |
| Single-Stage 7B | 64.08 | 58.19 | 58.03 | 44.58 |
| Single-Stage 13B | 68.98 | 63.59 | 61.64 | 44.89 |
| Frozen ViT (Single-Stage) | 64.08 | 58.19 | 58.03 | 44.58 |
| Finetune ViT (Multi-Stage) | 19.24 | 17.48 | 23.12 | 16.35 |
| Finetune ViT (Single-Stage) | 42.56 | 37.89 | 41.05 | 33.42 |
| Visual Representations | | | | |
| IN1K ViT-L 224px | 43.24 | 35.40 | 36.05 | 19.58 |
| DINOv2 ViT-L 224px | 28.65 | 20.72 | 24.75 | 8.33 |
| CLIP ViT-L 224px | 59.88 | 53.69 | 53.37 | 37.16 |
| SigLIP ViT-SO 224px | 57.94 | 51.90 | 53.31 | 37.42 |
| Image Preprocessing | | | | |
| CLIP ViT-L 336px (Letterbox) | 64.08 | 58.19 | 58.03 | 44.58 |
| CLIP ViT-L 336px (Resize Crop) | 54.31 | 49.14 | 49.43 | 40.82 |
| CLIP ViT-L 336px (Naive Resize) | 65.28 | 58.79 | 59.93 | 44.20 |
| SigLIP ViT-SO 384px (Letterbox) | 63.09 | 56.24 | 58.17 | 45.50 |
| SigLIP ViT-SO 384px (Resize Crop) | 53.29 | 47.63 | 50.18 | 39.27 |
| SigLIP ViT-SO 384px (Naive Resize) | 61.38 | 55.76 | 56.84 | 41.49 |
| Ensembling Visual Features | | | | |
| CLIP ViT-L 336px (Naive Resize) | 65.28 | 58.79 | 59.93 | 44.20 |
| DINOv2 + CLIP 336px (Letterbox) | 72.44 | 65.84 | 64.32 | 47.41 |
| DINOv2 + CLIP 336px (Naive Resize) | 71.07 | 64.77 | 65.26 | 47.66 |
| SigLIP ViT-SO 384px (Naive Resize) | 61.38 | 55.76 | 56.84 | 41.49 |
| DINOv2 + SigLIP 384px (Letterbox) | 72.10 | 65.42 | 64.69 | 50.37 |
| DINOv2 + SigLIP 384px (Naive Resize) | 73.86 | 67.29 | 67.85 | 52.82 |
| Base vs. Instruct Tuned LMs | | | | |
| Vicuña v1.5 7B | 64.08 | 58.19 | 58.03 | 44.58 |
| Vicuña v1.5 13B | 68.98 | 63.59 | 61.64 | 44.89 |
| Llama-2 7B | 65.24 | 59.47 | 58.78 | 43.89 |
| Llama-2 13B | 68.53 | 62.97 | 61.70 | 45.75 |
| Better LMs | | | | |
| Mistral v1 7B | 71.10 | 65.06 | 63.52 | 48.80 |
| Mistral Instruct v1 7B | 70.59 | 64.90 | 62.03 | 48.00 |
| Co-training on Language Safety Data | | | | |
| Vicuña v1.5 7B | 64.08 | 58.19 | 58.03 | 44.58 |
| Vicuña v1.5 7B (No Co-training) | 63.94 | 57.51 | 57.88 | 44.11 |
| Llama-2 7B | 65.24 | 59.47 | 58.78 | 43.89 |
| Llama-2 7B (No Co-training) | 64.26 | 59.30 | 57.99 | 42.17 |
| Scaling Training Time | | | | |
| 1 Epoch | 64.08 | 58.19 | 58.03 | 44.58 |
| 1.25 Epochs | 67.02 | 61.37 | 60.01 | 46.45 |
| 1.5 Epochs | 68.62 | 62.81 | 61.21 | 47.23 |
| 2 Epochs | 71.23 | 65.40 | 63.32 | 46.32 |
| 3 Epochs | 71.79 | 66.94 | 63.87 | 46.25 |
| Scaling Data | | | | |
| Base | 64.08 | 58.19 | 58.03 | 44.58 |
| Base + LRV | 65.62 | 59.77 | 59.82 | 46.21 |
| Base + LVIS-4V | 63.91 | 58.82 | 58.91 | 43.83 |
| Base + LVIS-4V + LRV | 64.94 | 58.91 | 58.19 | 43.73 |
| Prism 7B | | | | |
| Prism-CLIP 7B (Controlled) | 66.42 | 60.14 | 60.56 | 44.12 |
| Prism-CLIP 7B | 71.98 | 66.96 | 66.18 | 44.65 |
| Prism-SigLIP 7B (Controlled) | 64.74 | 58.58 | 60.56 | 43.63 |
| Prism-SigLIP 7B | 70.92 | 65.73 | 65.46 | 48.08 |
| Prism-DINOSigLIP 7B (Controlled) | 73.62 | 67.85 | 66.34 | 50.56 |
| Prism-DINOSigLIP 7B | 77.78 | 73.08 | 71.04 | 54.12 |
| Prism 13B | | | | |
| Prism-CLIP 13B (Controlled) | 70.92 | 65.95 | 65.03 | 47.32 |
| Prism-CLIP 13B | 73.37 | 68.71 | 69.06 | 48.98 |
| Prism-SigLIP 13B (Controlled) | 59.21 | 53.33 | 54.66 | 40.44 |
| Prism-SigLIP 13B | 69.69 | 64.99 | 64.81 | 44.31 |
| Prism-DINOSigLIP 13B (Controlled) | 76.64 | 71.41 | 70.87 | 53.60 |
| Prism-DINOSigLIP 13B | 79.39 | 75.55 | 72.73 | 54.62 |

Investigating the Design Space of Visually-Conditioned Language Models

Table 4. All Results on Challenge Benchmarks

| Model | VSR | POPE | TallyQA | AI2D |
|--|-------|-------|---------|-------|
| Official Models | | | | |
| LLaVa v1.5 7B | 51.47 | 86.57 | 62.06 | 54.10 |
| LLaVa v1.5 13B | 69.07 | 87.10 | 64.83 | 57.13 |
| InstructBLIP 7B | 58.92 | 84.30 | 15.51 | 32.90 |
| InstructBLIP 13B | 63.91 | 84.49 | 49.73 | 35.60 |
| Reproduction & Optimization Procedure | | | | |
| LLaVa v1.5 7B (Reproduction) | 52.95 | 86.57 | 60.87 | 54.43 |
| LLaVa v1.5 13B (Reproduction) | 65.38 | 86.94 | 64.13 | 56.81 |
| Single-Stage 7B | 51.47 | 86.57 | 61.63 | 54.85 |
| Single-Stage 13B | 70.05 | 87.00 | 63.26 | 57.25 |
| Frozen ViT (Single-Stage) | 51.47 | 86.57 | 61.63 | 54.85 |
| Finetune ViT (Multi-Stage) | 57.20 | 82.70 | 59.15 | 52.26 |
| Finetune ViT (Single-Stage) | 51.47 | 83.82 | 59.53 | 53.52 |
| Visual Representations | | | | |
| IN1K ViT-L 224px | 51.47 | 82.08 | 52.95 | 50.01 |
| DINOv2 ViT-L 224px | 51.47 | 84.84 | 57.12 | 51.39 |
| CLIP ViT-L 224px | 51.47 | 85.80 | 59.09 | 53.90 |
| SigLIP ViT-SO 224px | 51.47 | 85.07 | 63.02 | 55.35 |
| Image Preprocessing | | | | |
| CLIP ViT-L 336px (Letterbox) | 51.47 | 86.57 | 61.63 | 54.85 |
| CLIP ViT-L 336px (Resize Crop) | 51.47 | 85.42 | 61.24 | 53.52 |
| CLIP ViT-L 336px (Naive Resize) | 51.47 | 87.01 | 62.90 | 54.43 |
| SigLIP ViT-SO 384px (Letterbox) | 51.47 | 86.78 | 64.83 | 54.84 |
| SigLIP ViT-SO 384px (Resize Crop) | 51.47 | 84.62 | 62.94 | 54.51 |
| SigLIP ViT-SO 384px (Naive Resize) | 51.47 | 86.52 | 65.47 | 54.89 |
| Ensembling Visual Features | | | | |
| CLIP 336px (Naive Resize) | 51.47 | 87.01 | 62.90 | 54.43 |
| DINOv2 + CLIP 336px (Letterbox) | 51.47 | 87.70 | 63.99 | 52.07 |
| DINOv2 + CLIP 336px (Naive Resize) | 51.47 | 87.29 | 65.02 | 52.52 |
| SigLIP 384px (Naive Resize) | 51.47 | 86.52 | 65.47 | 54.89 |
| DINOv2 + SigLIP 384px (Letterbox) | 51.47 | 87.89 | 67.19 | 55.43 |
| DINOv2 + SigLIP 384px (Naive Resize) | 51.55 | 88.30 | 67.63 | 54.82 |
| Base vs. Instruct Tuned LMs | | | | |
| Vicuña v1.5 7B | 51.47 | 86.57 | 61.63 | 54.85 |
| Vicuña v1.5 13B | 70.05 | 87.00 | 63.26 | 57.25 |
| Llama-2 7B | 63.67 | 86.74 | 59.22 | 55.31 |
| Llama-2 13B | 65.71 | 86.91 | 62.54 | 56.51 |
| Better LMs | | | | |
| Mistral v1 7B | 58.50 | 87.10 | 61.70 | 50.54 |
| Mistral Instruct v1 7B | 57.80 | 87.50 | 64.53 | 51.48 |
| Co-training on Language Safety Data | | | | |
| Vicuña v1.5 7B | 51.47 | 86.57 | 61.63 | 54.85 |
| Vicuña v1.5 7B (No Co-training) | 53.68 | 87.27 | 62.31 | 52.74 |
| Llama-2 7B | 63.67 | 86.74 | 59.22 | 55.31 |
| Llama-2 7B (No Co-training) | 67.18 | 86.88 | 57.17 | 53.87 |
| Scaling Training Time | | | | |
| 1 Epoch | 51.47 | 86.57 | 61.63 | 54.85 |
| 1.25 Epochs | 51.80 | 86.80 | 61.69 | 54.02 |
| 1.5 Epochs | 51.55 | 87.78 | 61.67 | 53.74 |
| 2 Epochs | 53.93 | 87.03 | 62.52 | 54.91 |
| 3 Epochs | 54.99 | 86.86 | 62.30 | 52.52 |
| Scaling Data | | | | |
| Base | 51.47 | 86.57 | 61.63 | 54.85 |
| Base + LRV | 64.08 | 86.84 | 65.54 | 53.82 |
| Base + LVIS-4V | 51.47 | 86.98 | 62.60 | 53.40 |
| Base + LVIS-4V + LRV | 54.91 | 87.27 | 63.74 | 54.87 |
| Prism 7B | | | | |
| Prism-CLIP 7B (Controlled) | 66.61 | 86.83 | 60.86 | 55.46 |
| Prism-CLIP 7B | 57.77 | 87.30 | 66.00 | 52.89 |
| Prism-SigLIP 7B (Controlled) | 65.14 | 87.07 | 64.54 | 55.48 |
| Prism-SigLIP 7B | 56.79 | 87.30 | 66.46 | 54.38 |
| Prism-DINOSigLIP 7B (Controlled) | 66.28 | 88.28 | 65.07 | 55.51 |
| Prism-DINOSigLIP 7B | 59.57 | 88.12 | 66.70 | 55.65 |
| Prism 13B | | | | |
| Prism-CLIP 13B (Controlled) | 65.96 | 86.96 | 65.71 | 56.64 |
| Prism-CLIP 13B | 71.85 | 87.23 | 69.37 | 55.82 |
| Prism-SigLIP 13B (Controlled) | 62.85 | 86.82 | 62.90 | 55.48 |
| Prism-SigLIP 13B | 64.57 | 87.50 | 68.95 | 55.95 |
| Prism-DINOSigLIP 13B (Controlled) | 71.85 | 88.50 | 66.09 | 57.72 |
| Prism-DINOSigLIP 13B | 72.18 | 88.07 | 70.41 | 57.96 |