Google DeepMind

July 2024

PaliGemma: A versatile 3B VLM for transfer

Lucas Beyer[*,†], Andreas Steiner[*], André Susano Pinto[*], Alexander Kolesnikov[*], Xiao Wang[*], Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bošnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier Henaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen and Xiaohua Zhai[*,†]

[*]Core team, [†]Project lead

PaliGemma is an open Vision-Language Model (VLM) that is based on the SigLIP-So400m vision encoder and the Gemma-2B language model. It is trained to be a versatile and broadly knowledgeable base model that is effective to transfer. It achieves strong performance on a wide variety of open-world tasks. We evaluate PaliGemma on almost 40 diverse tasks including standard VLM benchmarks, but also more specialized tasks such as remote-sensing and segmentation.

1. Introduction

PaliGemma is an open model, continuing the line of PaLI vision-language models in a combination with the Gemma family of language models.

PaLI is a series of state-of-the-art vision-language models, starting with the first PaLI [23] showing promising scaling results up to 17 B, using classification pretrained ViT [131] and mT5 [126] language model. PaLI-X [24] and PaLM-E [36] then pushed this further, combining ViT-22 B [29] and a 32 B UL2 [104] language model or the 540 B PaLM [28] language model, respectively, and getting further increased performance on vision-language tasks, albeit saturating performance on standard image classification and retrieval tasks. Finally, PaLI-3 [25] demonstrates that through better pretraining with SigLIP [133] and more careful multimodal data curation, a 2 B vision and 3 B language model (*i.e.* a 5 B vision-language model) matches the 10x larger PaLI-X and 100x larger PaLM-E across most benchmarks.

PaliGemma continues this trend, combining the 400 M SigLIP and the 2 B Gemma models [82] into a sub-3 B VLM that still maintains performance comparable to PaLI-X, PaLM-E, and PaLI-3.

Gemma [82] is a family of auto-regressive decoder-only open large language models built from the same research and technology used to create the Gemini [7] models. The models come in different sizes (2 B, 7 B), both pretrained and instruction fine-tuned. PaliGemma uses the 2 B pretrained version.

The main goal of our work is to provide a versatile base VLM. Hence, we show that it reaches state-of-the-art results not only on standard COCO captions, VQAv2, InfographicVQA and others, but also on more exotic Remote-Sensing VQA, TallyVQA, several video captioning and QA tasks, as well as referring expression *segmentation* (see full task list in Appendix B).

2. Related work

Over the course of the past few years, vision-language models have gained considerable importance in computer vision. The first generation, spearheaded by CLIP [94] and ALIGN [49] by scaling up ConVIRT [135] and VirTex [32], is an extension of large-scale classification pretraining [55, 131], to leverage all data from the web without the need for onerous human labeling, replacing a fixed and large set of classes by a caption embedding instead. The caption embeddings are mostly obtained using language encoders (similar to BERT [33]) and allow to open up the vocabulary of classification and retrieval tasks. The second generation, akin to T5 [95] in language, is a unification of captioning and question-answering tasks via generative encoder-decoder modeling [27, 111, 120, 138], often backed by the progress in generative language models.

Corresponding author(s): lbeyer,xzhai@google.com

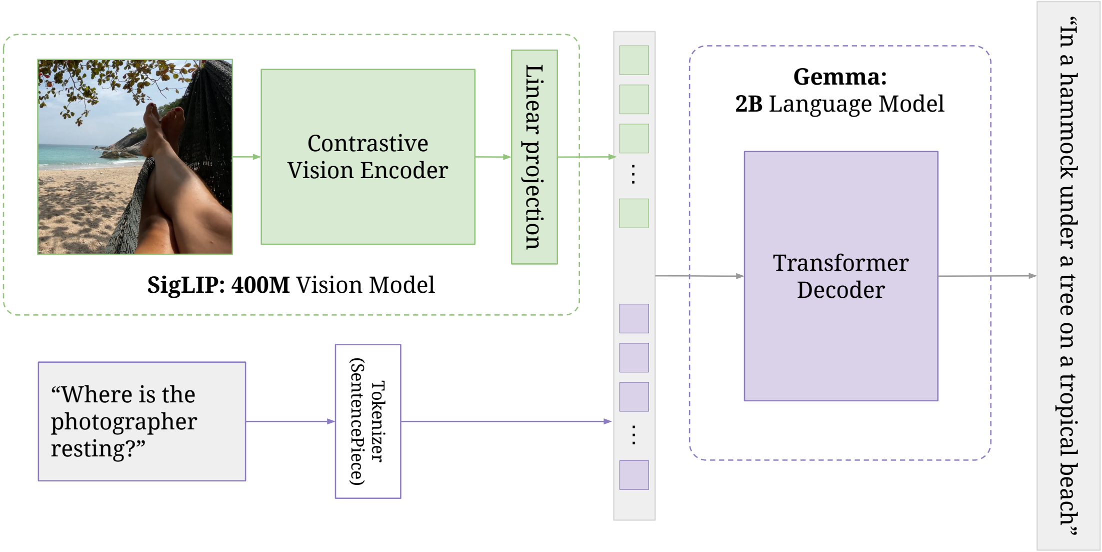

Contrastive Vision Encoder
Linear projection
SigLIP: 400M Vision Model
Gemma:
2B Language Model
Transformer Decoder
"In a hammock under a tree on a tropical beach"
"Where is the photographer resting?"
Tokenizer (SentencePiece)

Figure 1 | PaliGemma's architecture: a SigLIP image encoder feeds into a Gemma decoder LM.

These were then further scaled up by, among others, Flamingo [6], BLIP-2 [62] and, PaLI [23]. Finally, most recent works [7, 70, 87, 113] perform an additional "instruction tuning" step that is intended to make the raw model more user-friendly. In addition to building systems, several recent more systematic studies [59, 81, 107] aim to find out what really matters in VLMs. PaliGemma is an open base VLM without instruction tuning, and this report answers a few more questions regarding what matters. More discussion in Appendix A.

3. Model

In this section we present details about PaliGemma's architecture and training. Several of our decisions are further ablated in Section 5.

At a high level, PaliGemma is a VLM, taking as input one or more images, and a textual description of the task (the prompt or question, which we often refer to as the `prefix`). PaliGemma then autoregressively generates a prediction in the form of a text string (the answer, which we often refer to as the `suffix`).

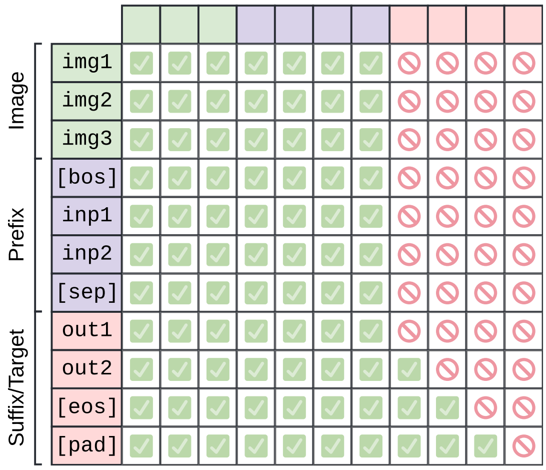
This simple image+text in, text out API is flexible enough to cover many standard tasks, such as image classification, captioning, visual question-answering and dialogue. Additionally, as shown in the literature, by converting more complex structured outputs into "text", this API can also cover more tasks such as: detection [22], instance segmentation [25, 115], panoptic segmentation, depth prediction, colorization, and many more [56, 73, 139]. This conversion can be hand-engineered and task-specific, such as done in pix2seq [22] for detection, or learned as is the case for segmentation [56] and dense output tasks in general.

During PaliGemma's pretraining, we limit ourselves to "text" covering natural language, object detection, and instance segmentation, but this API remains versatile and the pretrained models can be finetuned for other output types.

3.1. Architecture

PaliGemma consists of three components:

- An image encoder, for which we use a publicly available SigLIP [133] checkpoint, specifically the "shape optimized" [5] ViT-So400m image encoder. This model was contrastively pretrained at large scale via the sigmoid loss, and has shown state-of-the-art performance, especially for its small size.
- A decoder-only language model, for which we use the publicly available Gemma-2B v1.0 [82] raw pretrained checkpoint, which strikes a great balance between performance



		🟩	🟩	🟩	🟪	🟪	🟪	🟪	🟥	🟥	🟥	🟥
Image	img1	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘
	img2	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘
	img3	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘
Prefix	[bos]	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘
	inp1	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘
	inp2	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘
	[sep]	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘
Suffix/Target	out1	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘	⊘
	out2	✓	✓	✓	✓	✓	✓	✓	✓	⊘	⊘	⊘
	[eos]	✓	✓	✓	✓	✓	✓	✓	✓	✓	⊘	⊘
	[pad]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	⊘

Figure 2 | PaliGemma's Prefix-LM masking: block attention throughout image and prefix, autoregressive attention on the suffix. Each square indicates whether the row can attend to the column.

and size. As we will show, this language model is good enough to match or surpass the performance of VLMs using much larger language models, including previous PaLIs.

- A linear layer projecting SigLIP's output tokens into the same dimensions as Gemma-2B's vocab tokens, so they can be concatenated. In early experiments, we found that more complicated alternatives (*e.g.* MLPs) do not provide a clear advantage, and hence decided to use the simplest option (Sec 5.5).

The image is passed through the image encoder, which turns it into a sequence of N_{img} tokens. The text is converted into N_{txt} tokens using Gemma's SentencePiece [58] tokenizer, and embedded with Gemma's vocabulary embedding layer. The image tokens are projected with the (zero initialized) linear projection. Then the sequence of input tokens to the decoder is created as follows (and also as visible in Figure 2):

```
tokens = [image tokens...,
          BOS, prefix tokens..., SEP,
          suffix tokens..., EOS, PAD...]
```

We always resize the image to a fixed square size (224, 448, or 896 pixels). This leads to a fixed number of image tokens per model variant (respectively 256, 1024, or 4096 tokens), which we place in the front, making image tokens straightforward to interpret without the need for special location markers. The BOS token then marks the start of text tokens. We use \n as SEP token, it does not appear in any of our prefixes. We also tokenize SEP separately to avoid it being merged (by the tokenizer) with either the end of the prefix or the beginning of the suffix. In order to maximize model capacity for such a small model, we have full (unmasked) attention on the whole input, *i.e.* the image and prefix tokens. In this way, image tokens can also "lookahead" at the task at hand (prefix) in order to update their representation. The suffix is our output and necessarily covered by an auto-regressive mask, including the PAD tokens. When we mention sequence length (N_{txt}), we typically mean prefix and suffix combined, ignoring image tokens.

3.2. Pretraining

The training of PaliGemma follows the same steps as previous PaLI models, with only small modifications. Training consists of several stages, which we detail in this section:

- **Stage0:** Unimodal pretraining - we use existing off-the-shelf components.
- **Stage1:** Multimodal pretraining - long pretraining on a carefully chosen mixture of multimodal tasks. Notably, nothing is frozen.
- **Stage2:** Resolution increase - short continued pretraining at higher resolution.
- **Stage3:** Transfer - turn the base model into a task-specific specialist.

3.2.1. Stage0: Unimodal pretraining

First, the unimodal components of the model are pretrained individually, in order to benefit from their well-studied and scaled training recipes. For PaliGemma specifically, we do not perform any custom unimodal pretraining, instead relying on existing publicly available checkpoints.

Following PaLI-3's strong experimental results, we use a SigLIP image encoder. While PaLI-3 (and others [6, 26]) use a large image model such as ViT-G, we use the much smaller but similarly strong "shape optimized" ViT-So400m model.

PaLI traditionally uses an encoder-decoder language model; however all recently publicly released language models are decoder-only Transformers. We opt for the Gemma-2B model, which strikes a good balance between size and performance. Larger language models, such as the popular 7 B or 70 B sizes, are often significantly better at tasks like mathematical reasoning. However, PaLI-3 has shown that across a wide range of vision-language tasks, a well-trained small 5 B model (2 B vision + 3 B language) can attain the same performance as the much larger 55 B PaLI-X (22 B vision + 32 B language) and 562 B PaLM-E (22 B vision + 540 B language), including tasks such as ScienceQA. With PaliGemma we continue this push for smaller models and show that we can keep the same performance with less than 3 B total parameters.

3.2.2. Stage1: Multimodal pretraining

In this stage, we combine the unimodal models as explained in Section 3.1 and train the whole model on a broad mixture of large-scale vision-language tasks. Contrary to most recent VLMs, our core goal is to train a base model that finetunes well to a wide range of tasks, not merely to align the modalities. Intuitively, we want a mix of tasks which force the model to acquire a broad range of "skills", regardless of the task's user (or benchmark) friendliness out of the box. More on this in Section 3.2.5.

It is common practice, also followed by previous PaLI versions, to keep the image encoder frozen during the first multimodal pretraining stage. This is partially due to findings as in LiT [132] reporting multimodal tuning of pretrained image encoders degrading their representations. However, more recent work such as CapPa [110] and LocCa [115] have shown that captioning and other harder-to-learn tasks can provide valuable signal to image encoders, allowing them to learn spatial and relational understanding capabilities which contrastive models like CLIP or SigLIP typically lack. Hence, again in the spirit of learning more skills during pretraining, we depart from common practice and do not freeze the image encoder. However, the challenges outlined in LiT remain. In order to avoid destructive supervision signal from the initially unaligned language model, we use a slow linear warm-up for the image encoder's learning-rate (Figure 3), which ensures that the image encoder's quality is not deteriorated from the initially misaligned gradients coming through the LLM.

We train Stage1 at resolution 224px (hence, $N_{\text{img}} = 256$ image tokens) and sequence length $N_{\text{txt}} = 128$ for a total of 1 billion examples. While we provide an ablation in Section 5.1 showing that a 10x to 30x shorter Stage1 still provides good results on popular benchmarks, we wish to imbue as much visual knowledge to the base model as possible, and cover a broad set of concepts, cultures, and languages [17, 37, 68, 85, 92, 93, 136].

3.2.3. Stage2: Resolution increase

The model resulting from Stage1 is already a useful base model for many tasks (see example images in Appendix B). However, it only understands images at 224 × 224 pixel resolution, which is too small for several tasks. For instance, detection and segmentation of smaller objects, and tasks related to reading smaller texts such as charts, infographics, or documents, all strongly benefit from higher resolution (see Table 1). Hence, we train two further model checkpoints for increased resolution, first to 448 × 448 and then to 896 × 896 pixel resolution.

Since stage1 took care of providing the model with a broad set of knowledge and skill, stage2 can focus on extending the model's ability to parse higher-resolution images. We thus run Stage2 with fewer total examples, while increasing the cost and information density of each example. For resolution 448, we train for an additional 50 M examples, and for resolution 896, we add another 10 M examples.

For simplicity, Stage2 consists of the exact same mixture of tasks and datasets as Stage1, but with significantly increased sampling of tasks that require high resolution. Additionally, these upweighted tasks all can be modified to provide much longer suffix sequence lengths. For instance, for OCR tasks, we can simply request the model to read *all* text on the image in left-to-right, top-to-bottom order. For detection and segmentation tasks, we can request the model to detect or segment *all* objects for which annotation is provided. Hence, we also increase the text sequence length to $N_{\text{txt}} = 512$ tokens.

While PaLI has always had this resolution increasing stage, and for image classification the importance of resolution is long known [55, 109], several recent works [81, 114, 121] have raised the importance of resolution in VLMs too. We add to this body of knowledge by providing several ablation studies regarding Stage2 in Section 5.7.

3.2.4. Stage3: Transfer

The result of Stages 1 and 2 is a family of three PaliGemma checkpoints, at 224px, 448px, and 896px resolution, which are pre-equipped with broad visual knowledge. However, these checkpoints are not "user (or benchmark) friendly" as their pretraining has focused solely on density of learning signal, as opposed to usable interface.

These base models need to be transferred to serve their intended final purpose. That could take the form of fine-tuning on a specific, specialized task, such as COCO Captions, Remote Sensing VQA, Video Captioning, or InfographicQA. Adapt to new inputs such as multiple images (NLVR2) or bounding boxes draw in the image (WidgetCap). Or it could take the form of instruction [70] or even chat [46] tuning.

To show the effectiveness of the base models, we transfer them to a wide range of individual academic benchmarks, using a simple unified transfer recipe with few hyper-parameters. And to showcase the versatility beyond academic tasks, we also provide a "mix" transfer checkpoint, which transfers to a subset of these tasks at the same time, along with detailed captioning and long question-answering data. While this is not instruction tuning, it is a step in that direction.

We also transfer PaliGemma to tasks which take multiple images as input. NLVR2 is one such task, which asks one question about two images, and requires looking at both to give the correct answer. Other such tasks are standard short-video understanding tasks subsampled to 16 frames. In all these cases, we follow PaLI-3 and encode each image separately, then concatenate the image tokens without any special separator or embedding tokens. Thus, 16 frames at 224px resolution result in $N_{\text{img}} = 4096$ image tokens, the same amount as a single image at 896px resolution.

For all transfers, we perform fine-tuning of all the model parameters. The hyper-parameters we modify per-task are the following, in decreasing order of importance:

- Resolution (*i.e.* checkpoint): **224**, 448, 896.
- Epochs: **1, 3, 10**, 30, 100.
- Learning-rate: 3e-5, **1e-5**, 3e-6.
- Label-smoothing: **0.0**, 0.1, 0.3.
- Dropout in the LLM: **0.0**, 0.1, 0.3.
- Weight decay: **0.0** or 0.1 × learning-rate.
- Freeze ViT: **false**, true.
- Beam-search may benefit captioning.

The above are typical values we suggest exploring, with the recommended initial attempt value in bold. We provide the best setting for each individual task in Appendix J. We study the sensitivity to transfer hyper-parameters in Section 6.2, and the "transferability" in general in Section 6, showing that good results can be achieved with the aforementioned initial attempt values.

3.2.5. Pretraining task mixture

Just like for previous PaLI models, the pretraining (Stage1 and Stage2) is designed to result in a model that transfers well, not necessarily a model that is usable out of the box ("0 shot"). The intuition here is that we want a mix of tasks which force the model to acquire a broad range of "skills". We prefix each task with its unique prefix to avoid conflicting learning signals across skills [14]. At transfer time (Stage3), the model then merely needs to recognize which skill is useful for the task, and rewire itself to use that while following the output syntax and vocabulary of the task. In our experience, these can all be done relatively quickly and based on few examples (Section 6.3). We do not use any of our transfer datasets during pretraining, and furthermore remove all near-duplicates of their images from the pretraining datasets [55].

Largely following previous PaLI works, these are the pretraining tasks:

`caption {lang}`
We include the simple captioning objective on various datasets, including WebLI in over 100 languages, and CC3M-35L. Previous PaLIs use an encoder-decoder language model with the SplitCap objective, however for PaliGemma with the decoder-only language model, plain captioning is a more informative and simpler objective.

`ocr`
Concatenation (in raster order) of all text on the image transcribed by a public OCR system. Potentially skipping random snippets of OCR in order to fit sequence length without biasing recognition towards the beginning of raster order.

`answer en {question}`
Generated VQA on CC3M-35L following [19] with questions in 35 languages but English answers. Additionally, English-only object-centric questions on OpenImages following [91]:
listing: `What objects are in the image?`,
presence: `Is {thing} in the image?`,
multi-object presence: `Which of {thing}, {thing}... are in the image?`,
and newly, counting: `How many {thing}?`.

`question {lang} {English answer}`
Generated VQG on CC3M-35L following [19] generating questions in 35 languages, for a given English answer.

`detect {thing} ; {thing} ; ...`
Multi-object detection similar to Pix2Seq [22] on generated open-world data via pseudo-labeling as described in OWL-ViTv2 [83].

`segment {thing} ; {thing} ; ...`
Multi-object instance segmentation as in PaLI-3 [25] on generated open-world data similar to OWL-ViTv2 [83] and SAM [54].

`caption <ymin><xmin><ymax><xmax>`
Grounded captioning of what is in the box, following LocCa [115]. The box is indicated by the same location tokens as used in detection and segmentation: normalized image coordinates binned to 1024 tokens.

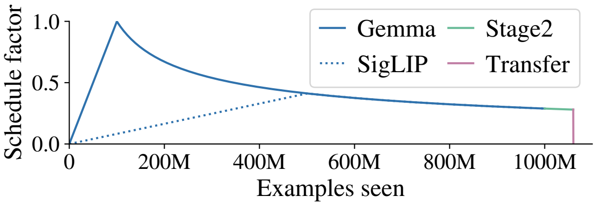


Figure 3 | Learning-rate schedule across stages.

Notably distinct from the widely used LLaVa's GPT-4 generated instruction following data, none of PaliGemma's pretraining tasks is the output of a larger commercial VLM.

Finally, we believe that it is important to detect and remove all images in our pretraining datasets which are near-duplicates of images in the transfer tasks we evaluate in this report [55], as well as a few more popular computer vision benchmarks. Doing so, we more accurately capture PaliGemma's capability to transfer to new tasks.

3.2.6. *Other pretraining details*

Throughout pretraining, we use an "infinite" learning-rate schedule following [131], which provides a straightforward way of chaining several stages without decaying the learning-rate between them. Figure 3 shows the full schedule: pretraining is one continuous rsqrt curve for all stages. The transfer can then act as a cooldown, fully annealing the learning rate. We recommend transferring with a simple setup that tunes the full model using a cosine learning-rate schedule with a short linear warm-up and decaying to zero. This is not well represented by Figure 3 due to its comparatively short duration.

The model was entirely trained in the open-source `big_vision` codebase [12] on Cloud TPUv5e [38]. However, some of the pretraining datasets remain private. During training, we partition data, as well as model parameters and optimizer state (Zero-DP style [96]) across all available devices using JAX [16] with GSPMD [125]. This fully-sharded data-parallel (FSDP [137]) sharding strategy is achieved by constructing global arrays and annotating the sharding accordingly, with the XLA compiler [97] taking care of the concrete implementation of the computation and communication between devices. We measured a model FLOPS utilization (MFU) of 55%, resulting in 5189 tokens/second/device. Model parameters and optimizer state are kept in float32 to guarantee stable training, but we verified that inference works just as well with bfloat16 model parameters.

One training run of the final PaliGemma model using TPUv5e-256 takes slightly less than 3 days for Stage1 and 15h for each Stage2. Stage1 sees slightly less than 350 B tokens, and both Stage2 combined about 90 B tokens. Transfers take between 20min and 10h on TPUv3-32, depending on the task.

In order to avoid model brittleness to different image processing details in different frameworks, we randomize the image preprocessing details such as resize method, JPEG encoding, and apply very slight `inception_crop`.

4. Results

In order to verify the transferability of PaliGemma to a wide variety of tasks, we transfer the pretrained models on more than 30 academic benchmarks via fine-tuning. Importantly, none of these tasks or datasets are part of the pretraining data mixture, and their images are explicitly removed from the web-scale pretraining data. Results are presented in Table 1.

To select the hyper-parameters for transfer, we first sweep the parameters mentioned in Section 3.2.4, starting from the recommended value. We do not necessarily perform the full cross-product, and we sometimes extend or supersample the range, if it seems promising. Importantly, we make any such decisions and hyper-parameter choices based on the transfer task's validation split, and if none is provided, we hold out a small "minival" set from the training data. Once we found good hyper-parameter values for a task, we re-train using the full training and validation data, and report final test numbers. Details on tasks, metrics, data splits are in Appendix B and final hyper-parameters in Appendix J. In Section 6.2 we show that a single recommended value for each hyper-parameter without any exploration

Table 1 | Results (1 random run of 5) obtained with PaliGemma. Tasks marked with ⌞ indicate zero-shot evaluation of the transferred model above. Where numbers depend on server submissions we report standard deviation from validation splits. Highlighted rows indicate resolution sensitive tasks. Per-task details and hyper-parameters are in Appendix B and J.

Task	224px	448px	896px
Image captioning			
COCOcap	141.9 ±0.3	144.6 ±0.5	-
⌞NoCaps	121.7 ±0.3	123.6 ±0.7	-
COCO-35L (en)	139.2 ±0.4	141.2 ±0.6	-
COCO-35L (avg34)	113.7 ±0.3	115.8 ±0.1	-
⌞XM3600 (en)	78.0 ±0.8	80.0 ±0.3	-
⌞XM3600 (avg35)	41.9 ±0.0	42.4 ±0.1	-
Screen2Words	117.6 ±0.7	119.6 ±0.7	-
TextCaps	127.5 ±1.0	153.9 ±0.5	-
SciCap	162.3 ±0.7	181.5 ±0.6	-
WidgetCap	136.1 ±1.4	148.4 ±0.3	-
Visual question answering			
VQAv2	83.2 ±0.4	85.6 ±0.2	-
⌞MMVP	47.3	45.3	-
⌞POPE	86.0	87.0	-
⌞Objaverse Multiview	62.7	62.8	-
OKVQA	63.5 ±0.3	63.2 ±0.2	-
AOKVQA-MC	76.4 ±0.4	76.9 ±0.3	-
AOKVQA-DA	61.9 ±0.6	63.2 ±0.5	-
GQA	65.6 ±0.4	67.0 ±0.3	-
⌞xGQA (avg7)	57.3 ±0.2	57.9 ±0.5	-
NLVR2	90.0 ±0.2	88.9 ±0.3	-
⌞MARVL (avg5)	80.6 ±0.3	76.8 ±0.3	-
AI2D	72.1 ±0.7	73.3 ±0.7	-
ScienceQA	95.4 ±0.3	95.9 ±0.2	-
RSVQA-lr	92.6 ±0.3	93.1 ±0.7	-
RSVQA-hr (test)	92.6 ±0.1	92.8 ±0.1	-
RSVQA-hr (test2)	90.6 ±0.1	90.5 ±0.2	-
ChartQA (human)	40.0 ±0.7	54.2 ±1.1	-
ChartQA (aug)	74.2 ±0.2	88.5 ±0.3	-
VizWizVQA	73.7 ±0.2	75.5 ±0.5	-
TallyQA (simple)	81.7 ±0.2	84.9 ±0.1	-
TallyQA (complex)	69.6 ±0.1	72.3 ±0.2	-
CountBenchQA	81.9 ±1.6	83.1 ±2.1	-
OCR-VQA	72.3 ±0.4	74.6 ±0.4	74.9
TextVQA	55.5 ±0.2	73.2 ±0.2	76.5
DocVQA	43.7 ±0.5	78.0 ±0.3	84.8
InfoVQA	28.5 ±0.4	40.5 ±0.3	47.8
ST-VQA	63.3 ±0.5	81.8 ±0.3	84.4
Image segmentation			
RefCOCO (testA)	75.7 ±0.1	77.9 ±0.1	78.7
RefCOCO (testB)	70.7 ±0.2	72.4 ±0.2	73.9
RefCOCO+ (testA)	71.9 ±0.1	74.2 ±0.2	76.1
RefCOCO+ (testB)	64.5 ±0.6	64.5 ±0.2	66.9
RefCOCOg (test)	68.2 ±0.1	71.0 ±0.1	72.7
Video input			
ActivityNet-QA	50.8 ±0.4	-	-
ActivityNet-CAP	34.6 ±0.6	-	-
MSRVTT-QA	50.1 ±0.1	-	-
MSRVTT-CAP	70.5 ±0.9	-	-
MSVD-QA	60.2 ±0.3	-	-
VATEX	79.7 ±0.4	-	-

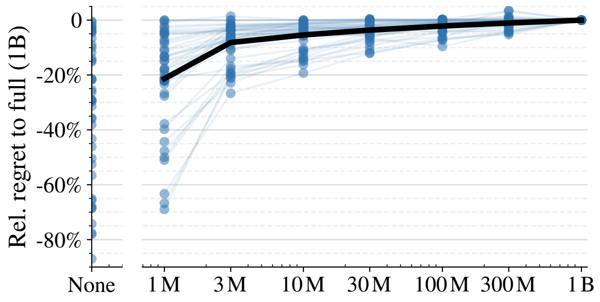


Figure 4 | Relative regret of transfers when varying the amount of pretraining during Stage 1. Per task plot in appendix K.1.

works almost as well on most tasks.

For all but video tasks, we report results on at least two resolutions to provide an impression of which tasks benefit from increased resolution. We provide many resolution-related ablations in Section 5.7.

Notably, we have not found any significant benefit from data augmentation. We simply resize the input images to a square fixed resolution, even for tasks such as RefCOCO segmentation (more on that in Section 5.7 and Appendix C).

5. Ablations

We conduct diverse ablations to gain deeper understanding of what matters for training and transferring VLMs. Unless noted otherwise, all ablations are run with the same setup as the main models, except for making the Stage1 pretraining 10x shorter (*i.e.* 100 M examples seen), and transfer results are reported on validation sets instead of withheld test-sets. For each experiment, we present only the salient result summary in the main text, but we provide a full per-task breakdown of results in the Appendix.

5.1. Multimodal pretraining duration

With its 1 B examples seen, our multimodal pretraining (Stage1) is on the longer side, similar to BLIP-2 [62], InternVL [26], QwenVL [10], Idefics2 [59] (all around 1 B), but unlike ShareGPT4-v [21], Mini-Gemini [65], LLaVa [70] and its derivatives (around 1 M).

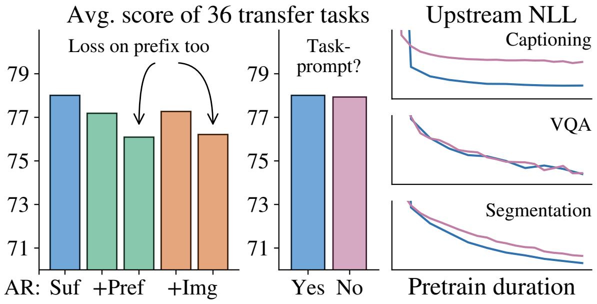


Figure 5 | Learning setup for Stage1. Left: Where to apply the auto-regressive mask and loss. Middle and right: whether to include a task indicator.

To the best of our knowledge, the benefits from longer pretraining have not been studied in isolation. We run pretrainings of various shorter durations, all the way down to completely skipping Stage1, and show the impact in Figure 4, with a complete break-down across tasks in the Appendix K.1. For the case of skipping Stage1, we use the best transfer result when sweeping over three learning-rates for each task.

The result shows that shorter training generally hurts, and skipping Stage1 entirely is the worst setting. Some task are affected significantly, while others only deteriorate a little, highlighting the need for a broad and diverse set of evaluation tasks. The 100 M pretraining duration appears to be a good trade-off for ablations: it is 10x shorter while not significantly hurting any task.

5.2. Causal masking and learning objective

We ablate several of our key choices in the pretraining learning objective in Figure 5, full per-task breakdown in Appendix K.2.

First, we investigate design choices for auto-regressive masking. PaliGemma uses a prefix-LM strategy which allows full (bi-directional) attention on the “input” part of the data, *i.e.* the `image` and `prefix` tokens, see also Figure 2. The motivation is that it allows more tokens to actively participate in the “thinking” process from the start, as the image tokens now can attend to the prefix tokens which represent the query. This is empirically confirmed in Figure 5 (left), where the green bars also include the auto-regressive masking on the `prefix` tokens, and the orange bars further extend the auto-regressive mask to the `image` tokens. Both sets of bars work, but perform clearly worse than PaliGemma's prefix-LM setting represented by the blue bar.

Second, we apply the next-token-prediction loss on the `suffix` (the output) only. In principle, it can also be applied on the `prefix` tokens, once they are auto-regressively masked. This could provide more learning signal to the model by asking it to "guess the question" for example. Again, Figure 5 shows that while it works, doing so clearly reduces average performance.

Finally, we eased the multi-task learning by using task-prefixes [14]. Figure 5 (middle) shows that after transfers, this choice of pretraining has no noticeable effect. However, it would be incorrect to conclude that it has no effect on the model's training. Indeed, pretraining validation perplexities on three representative tasks of pretraining are shown in Figure 5 (right): when the `prefix` makes the task obvious, such as in VQA, a task-prefix has no effect. For tasks where the `prefix` does not perfectly disambiguate the exact task, however, the model (expectedly) does become noticeably more uncertain in its predictions without task-prefix.

Overall, the prefix-LM with task-prefix and supervision only on the `suffix` tokens is an effective VLM pretraining objective.

5.3. New token initialization

We add new tokens to Gemma's vocabulary to support PaliGemma's ability to perform more structured computer vision tasks. We add 1024 location tokens (`<loc0000>` to `<loc1023>`), which correspond to binned normalized image coordinates and are used in detection, referring expression, and grounded captioning tasks. We also add 128 VQVAE [112] tokenized single-object mask tokens [25, 86] (`<seg000>` to `<seg127>`) to support referring expression segmentation.

This poses the question of how to initialize the embeddings of these new tokens, given all other vocabulary tokens have already been trained as part of Gemma's pretraining. One option is to use a standard small Gaussian noise initialization ($\sigma = 0.02$, blue in Fig 6). However, [42] argues for matching the average of the pretrained embeddings plus small noise (AvgEmb, mauve in Fig 6). We compare these strategies in Figure 6 and see that, while the AvgEmb strategy significantly improves initial perplexity of tasks using the new tokens (top-right zoom-in, step 0), this gain vanishes after a thousand steps of training. The standard initialization strategy not only results in significantly better Stage1 perplexities at the end of pretraining (top-right, full plot), but also results in significantly better transfer of the model to tasks using these tokens, here RefCOCO segmentation MIoU (bottom right).

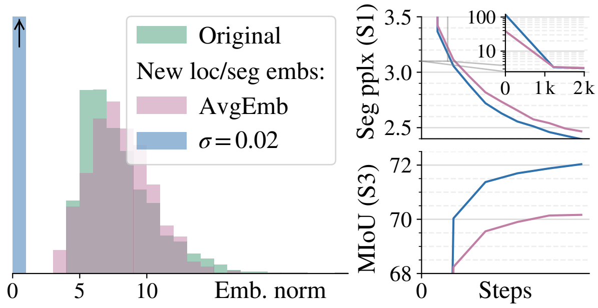


Figure 6 | Initialization of new tokens matters. Left: distribution of embedding norms at init. Right: Top: initial loss is better with AvgEmb, but this changes after a few thousand steps. Bottom: the model pretrained with $\sigma = 0.02$ inits transfers better to a task heavily using the new tokens.

5.4. To freeze or not to freeze?

The current common wisdom in VLMs [23–25, 45, 52, 60, 62, 66, 70] is to keep the image encoder and sometimes the LLM frozen during multimodal pretraining (our Stage1). However, inspired by the positive results from CapPa [110] and LocCa [115] which show that pretraining an image encoder using captioning objectives essentially solves contrastive's blind spot [43] to relation and localization, we pretrained PaliGemma with no frozen parts. We now ablate the effect of freezing or tuning various parts of the model during Stage1 in Figure 7, full per-task breakdown in Appendix K.3. Similar to concurrent works [81, 107], we find not freezing any part

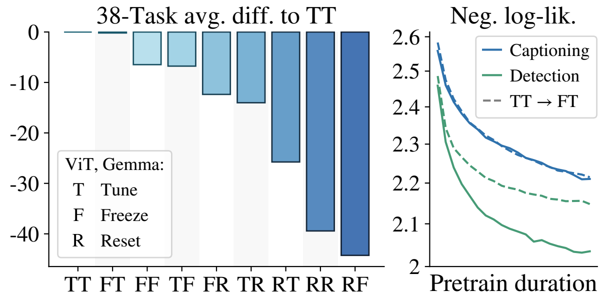


Figure 7 | Training setup for Stage1. Left: The more is frozen or reset, the more performance deteriorates. Right: The effect of freezing ViT is most visible in some pretraining perplexities.

of the model is indeed advantageous. First, after transfers, there is no difference to keeping the image encoder frozen (left, TT and TF). Second, however, the validation perplexity (hence, predictability) of tasks requiring spatial understanding (right, green) is significantly improved.

Further, we show that all other options that include freezing the language model [111] are significantly worse. Finally, resetting (and training, R) any part of the model hurts performance dramatically, confirming that Stage0 (*i.e.* leveraging pre-trained components) is indeed crucial for attaining good results.

5.5. Connector design

Throughout our experiments we use a linear connector to map SigLiP output embeddings to the inputs of Gemma. Given that an MLP connector [69] is a popular choice in the VLM literature, we also ablate this choice.

We consider two connector choices: a linear connector and an MLP (1 hidden layer, with GeLU non-linearity). We also consider two Stage1 pretraining settings: tune all weights (TT), or freeze everything but the connector (FF).

When tuning all weights, average transfer score is nearly identical for linear vs MLP, achieving 77.2 and 77.1 points respectively. In the "all-frozen" scenario, linear vs MLP achieve 70.7 vs 69.7. Surprisingly, we observe a small performance deterioration with the MLP connector.

Overall, we conclude that in our case, the linear connector seems preferable to the MLP connector.

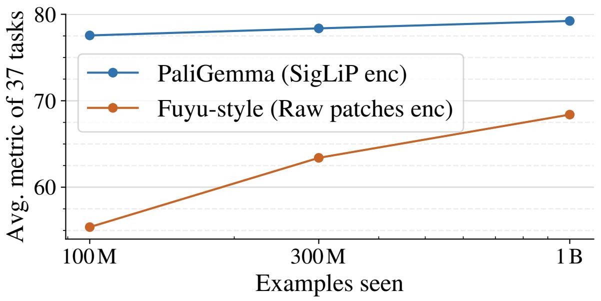


Figure 8 | Not using a SigLiP image encoder at all and instead passing a linear projection of raw RGB patches to Gemma works, but is significantly less sample-efficient.

5.6. Image encoder: with or without?

Most VLMs follow the setup of having an image encoder, such as CLIP/SigLIP (most works) or VQGAN (the Chameleon line of work [2, 3, 105, 129]), to turn the image into soft tokens before passing them to the LLM. We are aware of only two works that attempt to simplify this overall setup by removing the image encoder entirely and passing raw image patches into a decoder-only LLM, namely Fuyu [11] and EVE [34]. Unfortunately, the former provides no training details or ablations. The latter, which is a concurrent work, provides some details about training and various ablations, but with mixed results.

Removing the SigLIP encoder in PaliGemma results in a model of the same unified decoder-only architecture. We run our Stage1 and transfers in this setting. Since the architecture significantly changed, we also re-tune the learning-rate of Stage1. Figure 8 (per-task breakdown in Appendix K.4) shows that while this architecture still significantly lags behind, the scaling with pretraining duration seems potentially promising. This is especially noteworthy considering that PaliGemma's SigLIP encoder has seen 40 B image-text pairs during its Stage0 pre-training, while the Fuyu-style model sees images for the first time in the Stage1 pre-training shown here, and only sees up to 1 B of them in our experiment. This ablation confirms that such decoder-only VLMs might be a promising future direction towards simpler multimodal models, although

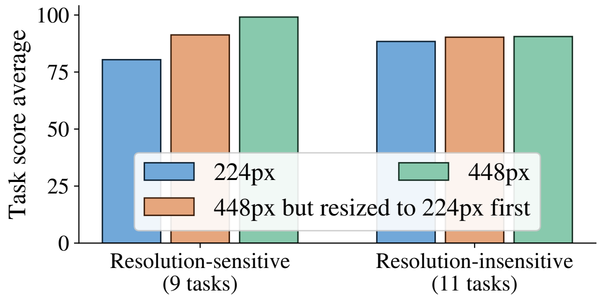


Figure 9 | Increasing resolution has two effects: increased information content of the input image, and increased model capacity via sequence length. For tasks that benefit from increased resolution, both of these effects contribute roughly equally to the overall gain.

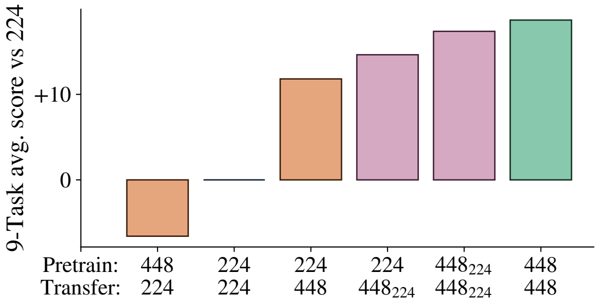


Figure 10 | Different ways of increasing resolution. While resolution during transfers works (orange), it is not the best setup. Using "windows" works better (mauve, s_w means image resolution s with window-size w), but simply running Stage2 at increased resolution with no tricks works best, justifying the need to provide multiple checkpoints. Per-task breakdown in Appendix K.6.

they currently still suffer in training efficiency due to not being able to reuse vision components.

5.7. Image resolution

Image resolution in VLMs is an important topic that has recently received increased attention [35, 59, 81, 107, 121]. PaliGemma uses a very simple approach to deal with resolution: Stage1 is pretrained at relatively low and economical 224px resolution, and short Stage2 then "upcycles" this checkpoint to higher resolutions (448px and 896px). Hence, the final PaliGemma model comes with three different checkpoints for three different resolutions.

In this section we justify our approach and the necessity for providing three separate checkpoints, and compare it to a recently popular "windowing" approach. For the ablation studies, we consider resolutions 224px and 448px, and restrict evaluation to single-image tasks where resolution has significant impact on performance according to Table 1, which we call "Resolution-sensitive" tasks.

5.7.1. Resolution or sequence length?

We generally see either neutral or improved performance across tasks when increasing the resolution of the input image (see Table 1). However, it is unclear whether this improvement comes from the fact that the image has higher resolution and therefore more information, or whether it is thanks to the resulting longer sequence length and thus increased model FLOPs and capacity. We disentangle these by running Stage2 and transfers at 448 px resolution, *but downscaling each image to 224 px before resizing it to 448 px*. Thus, the model gets to see the information content of a low-res (224 px) image but with the model capacity of the high-res (448 px) setting.

The result in Fig 9 shows that, for those tasks for which resolution has an effect at all, the reason for the improved performance is split roughly equally between these two causes. This is true for every individual task and not just an effect of averaging, see Appendix K.5.

5.7.2. Need resolution-specific checkpoints?

PaliGemma provides one checkpoint per resolution. But is this really needed? Could we not provide just a single, maybe high-resolution checkpoint, and adapt it as needed during transfers?

Figure 10 shows clearly that providing either only a 224 px or only a 448 px checkpoint would not be sufficient: Transferring the 448 px checkpoint at 224 px resolution (first bar, orange) leads to significantly worse results than using the 224 px checkpoint (second bar, zero), even though the latter had slightly less pretraining. Similarly, transferring the 224 px checkpoint at resolution 448 px (third bar, orange), while improving the results significantly, still lags far behind transferring the checkpoint whose native resolution is 448 px (last bar, green).

Thus, in the absence of flexible-resolution modeling tricks such as FlexiViT [13] or NaViT [30], we recommend running extended pretraining for increasing resolution (Stage2) and providing separate checkpoints for all supported resolutions.

5.7.3. *To resize or to window?*

Another recently common way of increasing input resolution is by "windowing" the models [114, 121, 134], *i.e.* applying the same model on windows of the model's native resolution from the higher-resolution image. We evaluate the simplest variant of this alternative: the 448 px resolution image is cut into four pieces, each one passed through the SigLIP image encoder separately. All four sets of 256 image embedding tokens are then concatenated and passed to Gemma. This is also related to using windowed attention in a ViT taking the full image [64]. We experimented with adding extra "window ID" position embeddings to indicate which window tokens come from, but this did not significantly change any of the results.

The mauve bars in Figure 10 correspond to windowing settings. While overall the performance is worse than that of a native 448px model, the windowing approach can be a promising way of transferring a model to a higher resolution *when no higher-resolution checkpoint is made available*, and when running a Stage2-like continued pretraining is not feasible.

Windowing might still seem preferable for speed reasons. However, we only observed at most a 5% speedup in training across various setups from windowing. This is explained by Gemma being significantly larger than ViT-So400m, and the Gemma part of the model is unaffected by windowing.

5.7.4. *Stage2 mixture re-weighting*

Finally, the pretraining mixture changes between Stage1 and Stage2. While previous PaLI versions change the mixture makeup, for PaliGemma we always use the full set of tasks, only changing their weighting, sampling "resolution-related" tasks (OCR, detection, segmentation) more frequently at higher resolutions. As an ablation, we run a Stage2 training with the same mixture ratios as Stage1. After transfers, this checkpoint is significantly worse on only three tasks (DocVQA, ChartQA, XM3600), but otherwise within per-task variance (Full results in Appendix K.7).

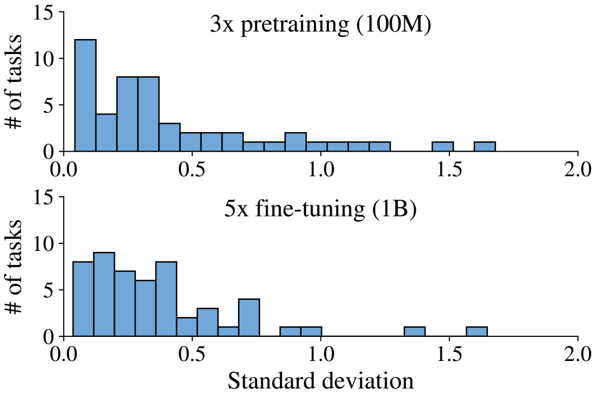


Figure 11 | For most tasks, the standard deviation of final metric values between re-runs with different seeds is below 0.5. This is true both for Stage1 (top, single transfers of three 100M runs), as well as for transfers (bottom, values from Table 1).

Thus, while changing the mixing ratio helped a little, it seems that when the intent is to train a base model for fine-tuning, the precise mixture ratios might not be as important as when training a model intended for sampling zero-shot, where it would significantly affect sampling frequencies.

6. Transferability

To show that PaliGemma is suitable for transfer under different scenarios, we run experiments that quantify its repeatability, sensitivity to hyperparameters, and number of transfer examples.

6.1. Repeatability (variance)

In the main results (Table 1), we show the standard deviation across 5 transfer reruns using the best hyper-parameter (listed in J). It is generally very small across most tasks, meaning transfer from the pretrained checkpoint is highly repeat-

Table 2 | Relative regret of using the suggested hyper-parameter setting for all tasks instead of performing hyper-parameter search. Per task plot in Appendix K.8.

Rel. Regret	#	Tasks
[None, 2.5%)	37	All other tasks
[2.5%, 5.0%)	2	ChartQA (human): 3.2% RefCOCO (val): 4.8%
[5.0%, 10.0%)	2	RefCOCOg (val): 5.8% ScienceQA: 6.7%
[10.0%, 100%]	2	RefCOCO+ (val): 10.7% SciCap: 60.5%

able. In Figure 11 we further show that the standard deviation of transferring from three reruns of Stage1 falls within the same range, meaning pretraining itself is also highly repeatable.

6.2. Transfer hyper-parameter sensitivity

However one question a reader may have is whether the choice of transfer hyper-parameter is important. To ablate that we run all tasks at 224px and under a single and simple hyper-parameter setup, which was also highlighted in bold in Section 3.2.4: lr=1e-5, bs=256, no dropout, no label smoothing, no weight decay, and not freezing anything. The only task dependent parameter is the number of epochs for which we use each task's best one but cap it at 10.

The results (Table 2) show that this simplified and single setup works very well for the majority of the tasks. The main exceptions we found were for tasks like RefCOCO and SciCap tasks which seem to benefit significantly from increasing the number of epochs while enabling label smoothing and dropout.

6.3. Transfer with limited examples

To analyze how many examples are needed to make PaliGemma solve a new task, we finetune PaliGemma with limited number of examples (64, 256, 1024, 4096). We sweep transfer with varying learning rates, epochs and batch size and report the best number without separate minival, to indicate the potential.

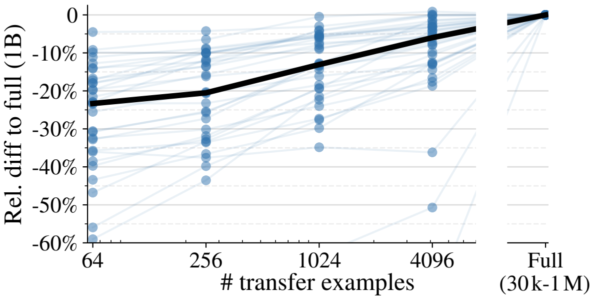


Figure 12 | Relative regret of using a limited number of transfer examples. Thick line represents the median. Per task plot in appendix K.9.

We run every setting with 5 different seeds, which also affect which examples are used. We found this important, as finetuning with limited examples exhibits high variance for some tasks (*e.g.* RefCOCO mIOU varied within 10%-30%). As a note, this variance also occurs when repeating with the same examples, but different batch order. Importantly, seed selection is not overfitting to the metric as the selected model performs equally well in the validation and test splits.But it does allows us to draw conclusions without needing to solve the open problem of making few-example fine-tuning stable.

Overall, when comparing the best runs of each hyper-parameter and seed with the results obtained with the full dataset, Figure 12 shows that it is not necessary to have a transfer dataset in the order of 10 k examples. The majority of the tasks can reach within 10% of the full-data score when using 4 k examples and 20% when using only 256 examples. In many cases the score with 64 transfer examples are good enough to prototype using PaliGemma for a new application.

7. Noteworthy tidbits

We also briefly discuss several small but interesting discoveries or we made, providing more details on each in the Appendix.

Plain resize to square for segmentation. We found simple resize to square (224×224) to work just as well as popular aspect-ratio preserving zoom and crop augmentations. (Appendix C)

Counting: introducing CountBenchQA. Due to skewed number distribution and varying image quality [51], we found TallyQA lacking in its ability to assess current VLM's ability to count. This is why we introduce CountBenchQA, a VLM-ready version of the CountBench [88] dataset. (Appendix D).

Issues in published WidgetCaps numbers. We found issues in at least three previous work's evaluation of WidgetCaps, rendering numerical comparisons invalid. (Appendix E).

Image annotations work as well as prompts. Marking the widget to be captioned with a red box in the image gives the same results as indicating it with `<loc>` tokens in the prompt. (Appendix F)

RoPE interpolation unnecessary for upscaling. For Stage2, we tried interpolating the RoPE position indices for the image tokens to preserve their semantics from Stage1. However, we saw no benefit from doing so.

Zero-shot generalization to 3D renders. Although never explicitly trained for it, PaliGemma generalizes surprisingly well to 3D renders form Objaverse without fine-tuning. (Appendix G)

Our MMVP result is SOTA by a large margin. PaliGemma at 224px achieves 47.3% paired accuracy, while GPT4-V and Gemini achieve 38.7% and 40.7%, respectively, and all other models including LLaVa perform below chance.

8. Conclusion

PaliGemma is a new, small, open base VLM that shines when transferred to a broad range of tasks. Our results show that VLMs on the "smaller" side can provide state-of-the-art performance across a wide variety of benchmarks. We also hope that providing the base model without instruction tuning serves as a useful starting point for further research in instruction tuning, specific applications, and encourages clearer separation of base models and fine-tunes in VLM research.

References

[1] M. Acharya, K. Kafle, and C. Kanan. Tallyqa: Answering complex counting questions. In *AAAI*, 2019.

[2] A. Aghajanyan, B. Huang, C. Ross, V. Karpukhin, H. Xu, N. Goyal, D. Okhonko, M. Joshi, G. Ghosh, M. Lewis, and L. Zettlemoyer. CM3: A causal masked multimodal model of the internet. *CoRR*, abs/2201.07520, 2022. URL https://arxiv.org/abs/2201.07520.

[3] A. Aghajanyan, L. Yu, A. Conneau, W. Hsu, K. Hambardzumyan, S. Zhang, S. Roller, N. Goyal, O. Levy, and L. Zettlemoyer. Scaling laws for generative mixed-modal language models. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 265–279. PMLR, 2023. URL https://proceedings.mlr.press/v202/aghajanyan23a.html.

[4] H. Agrawal, K. Desai, Y. Wang, X. Chen, R. Jain, M. Johnson, D. Batra, D. Parikh, S. Lee, and P. Anderson. nocaps: novel object captioning at scale. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8948–8957, 2019.

[5] I. Alabdulmohsin, X. Zhai, A. Kolesnikov, and L. Beyer. Getting vit in shape: Scaling laws for compute-optimal model design. In *NeurIPS*, 2023.

[6] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan. Flamingo: a visual language model for few-shot learning, 2022. URL https://arxiv.org/abs/2204.14198.

[7] R. Anil, S. Borgeaud, Y. Wu, J. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, D. Silver, S. Petrov, M. Johnson, I. Antonoglou, J. Schrittwieser, A. Glaese, J. Chen, E. Pitler, T. P. Lillicrap, A. Lazaridou, O. Firat, J. Molloy, M. Isard, P. R. Barham, T. Hennigan, B. Lee, F. Viola, M. Reynolds, Y. Xu, R. Doherty, E. Collins, C. Meyer, E. Rutherford, E. Moreira, K. Ayoub, M. Goel, G. Tucker, E. Piqueras, M. Krikun, I. Barr, N. Savinov, I. Danihelka, B. Roelofs, A. White, A. Andreassen, T. von Glehn, L. Yagati, M. Kazemi, L. Gonzalez, M. Khalman, J. Sygnowski, and et al. Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805, 2023. doi: 10.48550/ARXIV.2312.11805. URL https://doi.org/10.48550/arXiv.2312.11805.

[8] E. K. M. S. H. H. A. F. Aniruddha Kembhavi, Mike Salvato. A diagram is worth a dozen images. In *European Conference on Computer Vision (ECCV)*, 2016. URL https://api.semanticscholar.org/CorpusID:2682274.

[9] G. Baechler, S. Sunkara, M. Wang, F. Zubach, H. Mansoor, V. Etter, V. Cărbune, J. Lin, J. Chen, and A. Sharma. Screenai: A vision-language model for ui and infographics understanding, 2024. URL https://arxiv.org/abs/2402.04615.

[10] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *CoRR*, abs/2308.12966, 2023. doi: 10.48550/ARXIV.2308.12966. URL https://doi.org/10.48550/arXiv.2308.12966.

[11] R. Bavishi, E. Elsen, C. Hawthorne, M. Nye, A. Odena, A. Somani, and S. Taşırlar. Introducing our multimodal models, 2023. URL https://www.adept.ai/blog/fuyu-8b.

[12] L. Beyer, X. Zhai, and A. Kolesnikov. Big vision. https://github.com/google-research/big_vision, 2022.

[13] L. Beyer, P. Izmailov, A. Kolesnikov, M. Caron, S. Kornblith, X. Zhai, M. Minderer, M. Tschannen, I. Alabdulmohsin, and F. Pavetic. Flexivit: One model for all patch sizes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14496–14506, 2023.

[14] L. Beyer, B. Wan, G. Madan, F. Pavetic, A. Steiner, A. Kolesnikov, A. S. Pinto, E. Bugliarello, X. Wang, Q. Yu, et al. A study of autoregressive decoders for multi-tasking in computer vision. *arXiv preprint arXiv:2303.17376*, 2023.

[15] A. F. Biten, R. Tito, A. Mafla, L. Gomez, M. Rusinol, C. Jawahar, E. Valveny, and D. Karatzas. Scene text visual question answering. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2019. doi: 10.1109/iccv.2019.00439. URL http://dx.doi.org/10.1109/ICCV.2019.00439.

[16] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[17] E. Bugliarello, F. Liu, J. Pfeiffer, S. Reddy, D. Elliott, E. M. Ponti, and I. Vuli'c. IGLUE: A benchmark for transfer learning across modalities, tasks, and languages. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, page 2370–2392, Balitmore, MA, July 2022. PMLR. URL https://proceedings.mlr.press/v162/bugliarello22a.html.

[18] J. Cha, W. Kang, J. Mun, and B. Roh. Honeybee: Locality-enhanced projector for multimodal LLM. *CoRR*, abs/2312.06742,

2023. doi: 10.48550/ARXIV.2312.06742. URL https://doi.org/10.48550/arXiv.2312.06742.

[19] S. Changpinyo, D. Kukliansy, I. Szpektor, X. Chen, N. Ding, and R. Soricut. All you may need for VQA are image captions. In M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1947–1963, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.142. URL https://aclanthology.org/2022.naacl-main.142.

[20] D. L. Chen and W. B. Dolan. Collecting highly parallel data for paraphrase evaluation. In D. Lin, Y. Matsumoto, and R. Mihalcea, editors, *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 190–200. The Association for Computer Linguistics, 2011. URL https://aclanthology.org/P11-1020/.

[21] L. Chen, J. Li, X. Dong, P. Zhang, C. He, J. Wang, F. Zhao, and D. Lin. Sharegpt4v: Improving large multi-modal models with better captions. *arXiv preprint arXiv:2311.12793*, 2023.

[22] T. Chen, S. Saxena, L. Li, D. J. Fleet, and G. E. Hinton. Pix2seq: A language modeling framework for object detection. In *ICLR*, 2022.

[23] X. Chen, X. Wang, S. Changpinyo, A. J. Piergiovanni, P. Padlewski, D. Salz, S. Goodman, A. Grycner, B. Mustafa, L. Beyer, A. Kolesnikov, J. Puigcerver, N. Ding, K. Rong, H. Akbari, G. Mishra, L. Xue, A. Thapliyal, J. Bradbury, W. Kuo, M. Seyedhosseini, C. Jia, B. K. Ayan, C. Riquelme, A. Steiner, A. Angelova, X. Zhai, N. Houlsby, and R. Soricut. PaLI: A jointly-scaled multilingual language-image model. *CoRR*, arXiv:2209.06794, 2022.

[24] X. Chen, J. Djolonga, P. Padlewski, B. Mustafa, S. Changpinyo, J. Wu, C. R. Ruiz, S. Goodman, X. Wang, Y. Tay, S. Shakeri, M. Dehghani, D. Salz, M. Lucic, M. Tschannen, A. Nagrani, H. Hu, M. Joshi, B. Pang, C. Montgomery, P. Pietrzyk, M. Ritter, A. J. Piergiovanni, M. Minderer, F. Pavetic, A. Waters, G. Li, I. Alabdulmohsin, L. Beyer, J. Amelot, K. Lee, A. P. Steiner, Y. Li, D. Keysers, A. Arnab, Y. Xu, K. Rong, A. Kolesnikov, M. Seyedhosseini, A. Angelova, X. Zhai, N. Houlsby, and R. Soricut. PaLI-X: On scaling up a multilingual vision and language model. *CoRR*, arXiv:2305.18565, 2023.

[25] X. Chen, X. Wang, L. Beyer, A. Kolesnikov, J. Wu, P. Voigtlaender, B. Mustafa, S. Goodman, I. Alabdulmohsin, P. Padlewski, D. Salz, X. Xiong, D. Vlasic, F. Pavetic, K. Rong, T. Yu, D. Keysers, X. Zhai, and R. Soricut. PaLI-3 vision language models: Smaller, faster, stronger. *CoRR*, arXiv:2310.09199, 2023.

[26] Z. Chen, J. Wu, W. Wang, W. Su, G. Chen, S. Xing, M. Zhong, Q. Zhang, X. Zhu, L. Lu, B. Li, P. Luo, T. Lu, Y. Qiao, and J. Dai. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. *arXiv preprint arXiv:2312.14238*, 2023.

[27] J. Cho, J. Lei, H. Tan, and M. Bansal. Unifying vision-and-language tasks via text generation. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1931–1942. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/cho21a.html.

[28] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi,

S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel. Palm: Scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24:240:1–240:113, 2023. URL http://jmlr.org/papers/v24/22-1144.html.

[29] M. Dehghani, J. Djolonga, B. Mustafa, P. Padlewski, J. Heek, J. Gilmer, A. P. Steiner, M. Caron, R. Geirhos, I. Alabdulmohsin, R. Jenatton, L. Beyer, M. Tschannen, A. Arnab, X. Wang, C. R. Ruiz, M. Minderer, J. Puigcerver, U. Evci, M. Kumar, S. van Steenkiste, G. F. Elsayed, A. Mahendran, F. Yu, A. Oliver, F. Huot, J. Bastings, M. Collier, A. A. Gritsenko, V. Birodkar, C. N. Vasconcelos, Y. Tay, T. Mensink, A. Kolesnikov, F. Pavetic, D. Tran, T. Kipf, M. Lucic, X. Zhai, D. Keysers, J. J. Harmsen, and N. Houlsby. Scaling vision transformers to 22 billion parameters. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 7480–7512. PMLR, 2023. URL https://proceedings.mlr.press/v202/dehghani23a.html.

[30] M. Dehghani, B. Mustafa, J. Djolonga, J. Heek, M. Minderer, M. Caron, A. Steiner, J. Puigcerver, R. Geirhos, I. M. Alabdulmohsin, et al. Patch n'pack: Navit, a vision transformer for any aspect ratio and resolution. *Advances in Neural Information Processing Systems*, 36, 2024.

[31] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. VanderBilt, L. Schmidt, K. Ehsani, A. Kembhavi, and A. Farhadi. Objaverse: A universe of annotated 3d objects. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 13142–13153. IEEE, 2023. doi: 10.1109/CVPR52729.2023.01263. URL https://doi.org/10.1109/CVPR52729.2023.01263.

[32] K. Desai and J. Johnson. Virtex: Learning visual representations from textual annotations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11162–11173, 2021.

[33] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1423. URL https://doi.org/10.18653/v1/n19-1423.

[34] H. Diao, Y. Cui, X. Li, Y. Wang, H. Lu, and X. Wang. Unveiling encoder-free vision-language models, 2024. URL https://arxiv.org/abs/2406.11832.

[35] X. Dong, P. Zhang, Y. Zang, Y. Cao, B. Wang, L. Ouyang, S. Zhang, H. Duan, W. Zhang, Y. Li, H. Yan, Y. Gao, Z. Chen, X. Zhang, W. Li, J. Li, W. Wang, K. Chen, C. He, X. Zhang, J. Dai, Y. Qiao, D. Lin, and J. Wang. Internlm-xcomposer2-4khd: A pioneering large vision-language model handling resolutions from 336 pixels to 4k hd, 2024. URL https://arxiv.org/abs/2404.06512.

[36] D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. Florence. PaLM-E: An embodied multimodal language model. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 8469–8488. PMLR, 2023. URL https://proceedings.mlr.press/v202/driess23a.html.

[37] G. Geigle, R. Timofte, and G. Glavaš. African or european swallow? benchmarking large vision-language models for fine-grained object classification, 2024. URL https://arxiv.org/abs/2406.14496.

[38] Google Cloud. Introduction to Cloud TPU. https://cloud.google.com/tpu/docs/intro-to-tpu, 20xx. Accessed: 2024-07-04.

[39] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[40] D. Gurari, Q. Li, A. J. Stangl, A. Guo, C. Lin, K. Grauman, J. Luo, and J. P. Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3608–3617, 2018.

[41] M. He, Y. Liu, B. Wu, J. Yuan, Y. Wang, T. Huang, and B. Zhao. Efficient multimodal learning from data-centric perspective. *CoRR*, abs/2402.11530, 2024. doi: 10.48550/ARXIV.2402.11530. URL https://doi.org/10.48550/arXiv.2402.11530.

[42] J. Hewitt. Initializing new word embeddings for pretrained language models. https://nlp.stanford.edu/~johnhew//vocab-expansion.html, 2021.

[43] C.-Y. Hsieh, J. Zhang, Z. Ma, A. Kembhavi, and R. Krishna. Sugarcrepe: fixing hackable benchmarks for vision-language compositionality. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2024. Curran Associates Inc.

[44] T.-Y. Hsu, C. L. Giles, and T.-H. Huang. Scicap: Generating captions for scientific figures. *arXiv preprint arXiv:2110.11624*, 2021.

[45] S. Huang, L. Dong, W. Wang, Y. Hao, S. Singhal, S. Ma, T. Lv, L. Cui, O. K. Mohammed, B. Patra, Q. Liu, K. Aggarwal, Z. Chi, N. J. B. Bjorck, V. Chaudhary, S. Som, X. Song, and F. Wei. Language is not all you need: Aligning perception with language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[46] Y. Huang, Z. Meng, F. Liu, Y. Su, N. Collier, and Y. Lu. Sparkles: Unlocking chats across multiple images for multimodal instruction-following models, 2024. URL https://openreview.net/forum?id=oq5EF8parZ.

[47] D. Hudson and C. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *Computer Vision and Pattern Recognition (CVPR)*, abs/1902.09506, 2019. doi: 10.48550/arXiv.1902.09506. URL https://doi.org/10.48550/arXiv.1902.09506.

[48] A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira. Perceiver:

General perception with iterative attention. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 4651–4664. PMLR, 2021. URL http://proceedings.mlr.press/v139/jaegle21a.html.

[49] C. Jia, Y. Yang, Y. Xia, Y. Chen, Z. Parekh, H. Pham, Q. V. Le, Y. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 4904–4916. PMLR, 2021. URL http://proceedings.mlr.press/v139/jia21b.html.

[50] R. Kabra, L. Matthey, A. Lerchner, and N. J. Mitra. Leveraging vlm-based pipelines to annotate 3d objects. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*. PMLR, 2024.

[51] I. Kajić, O. Wiles, I. Albuquerque, M. Bauer, S. Wang, J. Pont-Tuset, and A. Nematzadeh. Evaluating numerical reasoning in text-to-image models, 2024.

[52] S. Karamcheti, S. Nair, A. Balakrishna, P. Liang, T. Kollar, and D. Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models, 2024. URL https://arxiv.org/abs/2402.07865.

[53] S. Kazemzadeh, V. Ordonez, M. Matten, and T. Berg. ReferItGame: Referring to objects in photographs of natural scenes. In A. Moschitti, B. Pang, and W. Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 787–798, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1086. URL https://aclanthology.org/D14-1086.

[54] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollar, and R. Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, October 2023.

[55] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby. Big transfer (BiT): General visual representation learning. In *European Conference on Computer Vision (ECCV)*, 2020.

[56] A. Kolesnikov, A. S. Pinto, L. Beyer, X. Zhai, J. Harmsen, and N. Houlsby. Uvim: A unified modeling approach for vision with learned guiding codes. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[57] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715, 2017.

[58] T. Kudo and J. Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In E. Blanco and W. Lu, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012.

[59] H. Laurençon, L. Tronchon, M. Cord, and V. Sanh. What matters when building vision-language models? *CoRR*, abs/2405.02246, 2024. doi: 10.48550/

ARXIV.2405.02246. URL https://doi.org/10.48550/arXiv.2405.02246.

[60] H. Laurençon, L. Saulnier, L. Tronchon, S. Bekman, A. Singh, A. Lozhkov, T. Wang, S. Karamcheti, A. M. Rush, D. Kiela, M. Cord, and V. Sanh. Obelics: An open web-scale filtered dataset of interleaved image-text documents, 2023. URL https://arxiv.org/abs/2306.16527.

[61] J. Li, D. Li, C. Xiong, and S. C. H. Hoi. BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 12888–12900. PMLR, 2022. URL https://proceedings.mlr.press/v162/li22n.html.

[62] J. Li, D. Li, S. Savarese, and S. C. H. Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 19730–19742. PMLR, 2023. URL https://proceedings.mlr.press/v202/li23q.html.

[63] Y. Li, G. Li, L. He, J. Zheng, H. Li, and Z. Guan. Widget captioning: Generating natural language description for mobileuser interface elements. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

[64] Y. Li, H. Mao, R. Girshick, and K. He. Exploring plain vision transformer backbones for object detection, 2022. URL https://arxiv.org/abs/2203.16527.

[65] Y. Li, Y. Zhang, C. Wang, Z. Zhong, Y. Chen, R. Chu, S. Liu, and J. Jia. Mini-gemini: Mining the potential of multi-modality vision language models. *arXiv preprint arXiv:2403.18814*, 2024.

[66] B. Lin, Z. Tang, Y. Ye, J. Cui, B. Zhu, P. Jin, J. Huang, J. Zhang, Y. Pang, M. Ning, and L. Yuan. Moe-llava: Mixture of experts for large vision-language models, 2024. URL https://arxiv.org/abs/2401.15947.

[67] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Doll'a r, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL http://arxiv.org/abs/1405.0312.

[68] F. Liu, E. Bugliarello, E. M. Ponti, S. Reddy, N. Collier, and D. Elliott. Visually grounded reasoning across languages and cultures. In M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10467–10485, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.818. URL https://aclanthology.org/2021.emnlp-main.818.

[69] H. Liu, C. Li, Y. Li, and Y. J. Lee. Improved baselines with visual instruction tuning, 2023.

[70] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[71] S. Lobry, D. Marcos, J. Murray, and D. Tuia. Rsvqa: Visual question answering for remote sensing data. *IEEE Transactions on Geoscience and Remote Sensing*, 58(12):8555–8566, Dec. 2020. ISSN 1558-0644. doi: 10.1109/tgrs.2020.2988782. URL http://dx.doi.org/10.1109/TGRS.2020.2988782.

[72] J. Lu, C. Clark, S. Lee, Z. Zhang, S. Khosla, R. Marten, D. Hoiem, and A. Kemb-

havi. Unified-io 2: Scaling autoregressive multimodal models with vision, language, audio, and action. *CoRR*, abs/2312.17172, 2023. doi: 10.48550/ARXIV.2312.17172. URL https://doi.org/10.48550/arXiv.2312.17172.

[73] J. Lu, C. Clark, R. Zellers, R. Mottaghi, and A. Kembhavi. UNIFIED-IO: A unified model for vision, language, and multimodal tasks. In *ICLR*, 2023.

[74] P. Lu, S. Mishra, T. Xia, L. Qiu, K.-W. Chang, S.-C. Zhu, O. Tafjord, P. Clark, and A. Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[75] T. Luo, C. Rockwell, H. Lee, and J. Johnson. Scalable 3d captioning with pretrained models, 2023. URL https://arxiv.org/abs/2306.07279.

[76] J. Mao, J. Huang, A. Toshev, O. Camburu, A. L. Yuille, and K. Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[77] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[78] A. Masry, X. L. Do, J. Q. Tan, S. Joty, and E. Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2263–2279, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.177. URL https://aclanthology.org/2022.findings-acl.177.

[79] M. Mathew, D. Karatzas, R. Manmatha, and C. V. Jawahar. Docvqa: A dataset for VQA on document images. *CoRR*, abs/2007.00398, 2020. URL https://arxiv.org/abs/2007.00398.

[80] M. Mathew, V. Bagal, R. Tito, D. Karatzas, E. Valveny, and C. V. Jawahar. Infographicvqa. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Jan. 2022. doi: 10.1109/wacv51458.2022.00264. URL http://dx.doi.org/10.1109/WACV51458.2022.00264.

[81] B. McKinzie, Z. Gan, J. Fauconnier, S. Dodge, B. Zhang, P. Dufter, D. Shah, X. Du, F. Peng, F. Weers, A. Belyi, H. Zhang, K. Singh, D. Kang, A. Jain, H. Hè, M. Schwarzer, T. Gunter, X. Kong, A. Zhang, J. Wang, C. Wang, N. Du, T. Lei, S. Wiseman, G. Yin, M. Lee, Z. Wang, R. Pang, P. Grasch, A. Toshev, and Y. Yang. MM1: methods, analysis & insights from multimodal LLM pre-training. *CoRR*, abs/2403.09611, 2024. doi: 10.48550/ARXIV.2403.09611. URL https://doi.org/10.48550/arXiv.2403.09611.

[82] T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, P. Tafti, L. Hussenot, A. Chowdhery, A. Roberts, A. Barua, A. Botev, A. Castro-Ros, A. Slone, A. Héliou, A. Tacchetti, A. Bulanova, A. Paterson, B. Tsai, B. Shahriari, C. L. Lan, C. A. Choquette-Choo, C. Crepy, D. Cer, D. Ippolito, D. Reid, E. Buchatskaya, E. Ni, E. Noland, G. Yan, G. Tucker, G. Muraru, G. Rozhdestvenskiy, H. Michalewski, I. Tenney, I. Grishchenko, J. Austin, J. Keeling, J. Labanowski, J. Lespiau, J. Stanway, J. Brennan, J. Chen, J. Ferret, J. Chiu, and et al. Gemma: Open models based on gemini research and technology. *CoRR*, abs/2403.08295, 2024. doi: 10.48550/ARXIV.2403.08295. URL https://doi.org/10.48550/arXiv.2403.08295.

[83] M. Minderer, A. A. Gritsenko, and N. Houlsby. Scaling open-vocabulary ob-

ject detection. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=mQPNcBWjGc.

[84] A. Mishra, S. Shekhar, A. K. Singh, and A. Chakraborty. Ocr-vqa: Visual question answering by reading text in images. In *ICDAR*, 2019.

[85] T. Nguyen, M. Wallingford, S. Santy, W. Ma, S. Oh, L. Schmidt, P. W. Koh, and R. Krishna. Multilingual diversity improves vision-language representations. *CoRR*, abs/2405.16915, 2024. doi: 10.48550/ARXIV.2405.16915. URL https://doi.org/10.48550/arXiv.2405.16915.

[86] J. Ning, C. Li, Z. Zhang, Z. Geng, Q. Dai, K. He, and H. Hu. All in tokens: Unifying output space of visual tasks via soft token. *arXiv preprint arXiv:2301.02229*, 2023.

[87] OpenAI. Gpt-4 technical report, 2023. URL https://arxiv.org/abs/2303.08774.

[88] R. Paiss, A. Ephrat, O. Tov, S. Zada, I. Mosseri, M. Irani, and T. Dekel. Teaching clip to count to ten. *arXiv preprint arXiv:2302.12066*, 2023.

[89] Z. Peng, L. Dong, H. Bao, Q. Ye, and F. Wei. Beit v2: Masked image modeling with vector-quantized visual tokenizers. *CoRR*, abs/2208.06366, 2022. doi: 10.48550/ARXIV.2208.06366. URL https://doi.org/10.48550/arXiv.2208.06366.

[90] J. Pfeiffer, G. Geigle, A. Kamath, J.-M. Steitz, S. Roth, I. Vulić, and I. Gurevych. xGQA: Cross-lingual visual question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2497–2511, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.196. URL https://aclanthology.org/2022.findings-acl.196.

[91] A. Piergiovanni, W. Kuo, and A. Angelova. Pre-training image-language transformers for open-vocabulary tasks, 2022. URL https://arxiv.org/abs/2209.04372.

[92] A. Pouget, L. Beyer, E. Bugliarello, X. Wang, A. P. Steiner, X. Zhai, and I. Alabdulmohsin. No filter: Cultural and socioeconomic diversity in contrastive vision-language models. *CoRR*, abs/2405.13777, 2024. doi: 10.48550/ARXIV.2405.13777. URL https://doi.org/10.48550/arXiv.2405.13777.

[93] C. Qiu, D. Oneață, E. Bugliarello, S. Frank, and D. Elliott. Multilingual multimodal learning with machine translated text. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4178–4193, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.308. URL https://aclanthology.org/2022.findings-emnlp.308.

[94] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning, ICML*, 2021.

[95] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

[96] S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He. ZeRO: memory optimizations toward training trillion parameter models. In C. Cuicchi, I. Qualters, and W. T. Kramer, editors, *Proceedings of the International Conference for High Performance*

Computing, Networking, Storage and Analysis, SC 2020, Virtual Event / Atlanta, Georgia, USA, November 9-19, 2020, page 20. IEEE/ACM, 2020. doi: 10.1109/SC41405.2020.00024. URL https://doi.org/10.1109/SC41405.2020.00024.

[97] A. Sabne. XLA : Compiling machine learning for peak performance, 2020.

[98] D. Schwenk, A. Khandelwal, C. Clark, K. Marino, and R. Mottaghi. A-okvqa: A benchmark for visual question answering using world knowledge. *arXiv*, 2022.

[99] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro. Megatron-LM: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053, 2019. URL http://arxiv.org/abs/1909.08053.

[100] O. Sidorov, R. Hu, M. Rohrbach, and A. Singh. Textcaps: A dataset for image captioning with reading comprehension. In A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part II*, volume 12347 of *Lecture Notes in Computer Science*, pages 742–758. Springer, 2020. doi: 10.1007/978-3-030-58536-5_44. URL https://doi.org/10.1007/978-3-030-58536-5_44.

[101] A. Singh, V. Natarjan, M. Shah, Y. Jiang, X. Chen, D. Parikh, and M. Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8317–8326, 2019.

[102] A. Suhr, S. Zhou, A. Zhang, I. Zhang, H. Bai, and Y. Artzi. A corpus for reasoning about natural language grounded in photographs. In A. Korhonen, D. Traum, and L. Marquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6418–6428, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1644. URL https://aclanthology.org/P19-1644.

[103] Q. Sun, Y. Cui, X. Zhang, F. Zhang, Q. Yu, Z. Luo, Y. Wang, Y. Rao, J. Liu, T. Huang, and X. Wang. Generative multimodal models are in-context learners. *CoRR*, abs/2312.13286, 2023. doi: 10.48550/ARXIV.2312.13286. URL https://doi.org/10.48550/arXiv.2312.13286.

[104] Y. Tay, M. Dehghani, V. Q. Tran, X. Garcia, J. Wei, X. Wang, H. W. Chung, D. Bahri, T. Schuster, H. S. Zheng, D. Zhou, N. Houlsby, and D. Metzler. UL2: unifying language learning paradigms. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/pdf?id=6ruVLB727MC.

[105] C. Team. Chameleon: Mixed-modal early-fusion foundation models. *CoRR*, abs/2405.09818, 2024. doi: 10.48550/ARXIV.2405.09818. URL https://doi.org/10.48550/arXiv.2405.09818.

[106] A. V. Thapliyal, J. Pont Tuset, X. Chen, and R. Soricut. Crossmodal-3600: A massively multilingual multimodal evaluation dataset. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 715–729, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.45. URL https://aclanthology.org/2022.emnlp-main.45.

[107] S. Tong, E. Brown, P. Wu, S. Woo, M. Middepogu, S. C. Akula, J. Yang, S. Yang, A. Iyer, X. Pan, A. Wang, R. Fergus, Y. LeCun, and S. Xie. Cambrian-1: A Fully Open, Vision-Centric Exploration of Multimodal LLMs. *arXiv preprint arXiv:2406.16860*, 2024.

[108] S. Tong, Z. Liu, Y. Zhai, Y. Ma, Y. LeCun, and S. Xie. Eyes wide shut? exploring the

visual shortcomings of multimodal llms, 2024.

[109] H. Touvron, A. Vedaldi, M. Douze, and H. Jégou. Fixing the train-test resolution discrepancy, 2022. URL https://arxiv.org/abs/1906.06423.

[110] M. Tschannen, M. Kumar, A. Steiner, X. Zhai, N. Houlsby, and L. Beyer. Image captioners are scalable vision learners too. *NeurIPS*, 2023.

[111] M. Tsimpoukelli, J. Menick, S. Cabi, S. M. A. Eslami, O. Vinyals, and F. Hill. Multimodal few-shot learning with frozen language models. *CoRR*, abs/2106.13884, 2021. URL https://arxiv.org/abs/2106.13884.

[112] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6306–6315, 2017.

[113] Vikhyat. Moondream. https://github.com/vikhyat/moondream, 2024. Accessed: 2024-07-04.

[114] A. Visheratin. Breaking resolution curse of vision-language models. https://huggingface.co/blog/visheratin/vlm-resolution-curse, 2024.

[115] B. Wan, M. Tschannen, Y. Xian, F. Pavetic, I. Alabdulmohsin, X. Wang, A. S. Pinto, A. Steiner, L. Beyer, and X. Zhai. Locca: Visual pretraining with location-aware captioners. *CoRR*, abs/2403.19596, 2024. doi: 10.48550/ARXIV.2403.19596. URL https://doi.org/10.48550/arXiv.2403.19596.

[116] B. Wang, G. Li, X. Zhou, Z. Chen, T. Grossman, and Y. Li. Screen2words: Automatic mobile ui summarization with multimodal learning. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pages 498–510, 2021.

[117] W. Wang, H. Bao, L. Dong, J. Bjorck, Z. Peng, Q. Liu, K. Aggarwal, O. K. Mohammed, S. Singhal, S. Som, and F. Wei. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *CoRR*, abs/2208.10442, 2022. doi: 10.48550/ARXIV.2208.10442. URL https://doi.org/10.48550/arXiv.2208.10442.

[118] W. Wang, Q. Lv, W. Yu, W. Hong, J. Qi, Y. Wang, J. Ji, Z. Yang, L. Zhao, X. Song, J. Xu, B. Xu, J. Li, Y. Dong, M. Ding, and J. Tang. Cogvlm: Visual expert for pretrained language models. *CoRR*, abs/2311.03079, 2023. doi: 10.48550/ARXIV.2311.03079. URL https://doi.org/10.48550/arXiv.2311.03079.

[119] X. Wang, J. Wu, J. Chen, L. Li, Y.-F. Wang, and W. Y. Wang. Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[120] Z. Wang, J. Yu, A. W. Yu, Z. Dai, Y. Tsvetkov, and Y. Cao. Simvlm: Simple visual language model pretraining with weak supervision, 2022. URL https://arxiv.org/abs/2108.10904.

[121] P. Wu and S. Xie. V*: Guided visual search as a core mechanism in multimodal llms, 2023. URL https://arxiv.org/abs/2312.14135.

[122] B. Xiao, H. Wu, W. Xu, X. Dai, H. Hu, Y. Lu, M. Zeng, C. Liu, and L. Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. *CoRR*, abs/2311.06242, 2023. doi: 10.48550/ARXIV.2311.06242. URL https://doi.org/10.48550/arXiv.2311.06242.

[123] D. Xu, Z. Zhao, J. Xiao, F. Wu, H. Zhang, X. He, and Y. Zhuang. Video question answering via gradually refined attention

over appearance and motion. In *ACM Multimedia*, 2017.

[124] J. Xu, T. Mei, T. Yao, and Y. Rui. Msr-vtt: A large video description dataset for bridging video and language. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5288–5296, 2016. doi: 10.1109/CVPR.2016.571.

[125] Y. Xu, H. Lee, D. Chen, B. A. Hechtman, Y. Huang, R. Joshi, M. Krikun, D. Lepikhin, A. Ly, M. Maggioni, R. Pang, N. Shazeer, S. Wang, T. Wang, Y. Wu, and Z. Chen. GSPMD: general and scalable parallelization for ML computation graphs. *CoRR*, abs/2105.04663, 2021. URL https://arxiv.org/abs/2105.04663.

[126] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*, 2020.

[127] K. Z. J. W. W. X. Z. Yifan Li, Yifan Du and J.-R. Wen. Evaluating object hallucination in large vision-language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=xozJw0kZXF.

[128] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg. Modeling context in referring expressions. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, volume 9906 of *Lecture Notes in Computer Science*, pages 69–85. Springer, 2016. doi: 10.1007/978-3-319-46475-6_5. URL https://doi.org/10.1007/978-3-319-46475-6_5.

[129] L. Yu, B. Shi, R. Pasunuru, B. Muller, O. Golovneva, T. Wang, A. Babu, B. Tang, B. Karrer, S. Sheynin, C. Ross, A. Polyak, R. Howes, V. Sharma, P. Xu, H. Tamoyan, O. Ashual, U. Singer, S. Li, S. Zhang, R. James, G. Ghosh, Y. Taigman, M. Fazel-Zarandi, A. Celikyilmaz, L. Zettlemoyer, and A. Aghajanyan. Scaling autoregressive multi-modal models: Pretraining and instruction tuning. *CoRR*, abs/2309.02591, 2023. doi: 10.48550/ARXIV.2309.02591. URL https://doi.org/10.48550/arXiv.2309.02591.

[130] Z. Yu, D. Xu, J. Yu, T. Yu, Z. Zhao, Y. Zhuang, and D. Tao. Activitynet-qa: A dataset for understanding complex web videos via question answering. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 9127–9134. AAAI Press, 2019. doi: 10.1609/AAAI.V33I01.33019127. URL https://doi.org/10.1609/aaai.v33i01.33019127.

[131] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer. Scaling vision transformers. *Computer Vision and Pattern Recognition (CVPR)*, 2022.

[132] X. Zhai, X. Wang, B. Mustafa, A. Steiner, D. Keysers, A. Kolesnikov, and L. Beyer. Lit: Zero-shot transfer with locked-image text tuning. In *Computer Vision and Pattern Recognition (CVPR)*, 2022.

[133] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. Sigmoid loss for language image pre-training. In *ICCV*, pages 11941–11952, 2023.

[134] H. Zhang, H. You, P. Dufter, B. Zhang, C. Chen, H.-Y. Chen, T.-J. Fu, W. Y. Wang, S.-F. Chang, Z. Gan, and Y. Yang. Ferret-v2: An improved baseline for referring and grounding with large language models, 2024. URL https://arxiv.org/abs/2404.07973.

[135] Y. Zhang, H. Jiang, Y. Miura, C. D. Manning, and C. P. Langlotz. Contrastive learning of medical visual representations from

paired images and text. *arXiv preprint arXiv:2010.00747*, 2020.

[136] Y. Zhang, A. Unell, X. Wang, D. Ghosh, Y. Su, L. Schmidt, and S. Yeung-Levy. Why are visually-grounded language models bad at image classification?, 2024. URL https://arxiv.org/abs/2405.18415.

[137] Y. Zhao, A. Gu, R. Varma, L. Luo, C. Huang, M. Xu, L. Wright, H. Shojanazeri, M. Ott, S. Shleifer, A. Desmaison, C. Balioglu, P. Damania, B. Nguyen, G. Chauhan, Y. Hao, A. Mathews, and S. Li. Pytorch FSDP: experiences on scaling fully sharded data parallel. *Proc. VLDB Endow.*, 16(12):3848–3860, 2023. doi: 10.14778/3611540.3611569. URL https://www.vldb.org/pvldb/vol16/p3848-huang.pdf.

[138] L. Zhou, H. Palangi, L. Zhang, H. Hu, J. J. Corso, and J. Gao. Unified vision-language pre-training for image captioning and VQA. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 13041–13049. AAAI Press, 2020. doi: 10.1609/AAAI.V34I07.7005. URL https://doi.org/10.1609/aaai.v34i07.7005.

[139] X. Zou, Z. Dou, J. Yang, Z. Gan, L. Li, C. Li, X. Dai, H. Behl, J. Wang, L. Yuan, N. Peng, L. Wang, Y. J. Lee, and J. Gao. Generalized decoding for pixel, image, and language. In *Computer Vision and Pattern Recognition (CVPR)*, pages 15116–15127, 2023.

Author Contributions

Model development contributors

Core Contributors

Lucas Beyer
Andreas Steiner
André Susano Pinto
Alexander Kolesnikov
Xiao Wang
Xiaohua Zhai

Contributors

Daniel Salz
Maxim Neumann
Ibrahim Alabdulmohsin
Michael Tschannen
Emanuele Bugliarello
Thomas Unterthiner
Daniel Keysers
Skanda Koppula
Fangyu Liu
Adam Grycner
Alexey Gritsenko
Neil Houlsby
Manoj Kumar
Keran Rong
Julian Eisenschlos
Rishabh Kabra
Matthias Bauer
Matko Bošnjak
Xi Chen
Matthias Minderer
Paul Voigtlaender
Ioana Bica
Ivana Balazevic
Joan Puigcerver
Pinelopi Papalampidi
Olivier Henaff
Xi Xiong
Radu Soricut
Jeremiah Harmsen

Leads

Xiaohua Zhai
Lucas Beyer

Model release contributors and general support

PM

Tris Warkentin

Go-to-Market

Kat Black
Luiz Gustavo Martins
Glenn Cameron
Raj Gundluru
Manvinder Singh

Kaggle

Meg Risdal
Nilay Chauhan
Nate Keating
Nesh Devanathan

Documentation

Elisa Bandy
Joe Fernandez

Ethics and Safety

Antonia Paterson
Jenny Brennan
Tom Eccles
Pankil Botadra
Ben Bariach

Vertex AI

Lav Rai
Minwoo Park
Dustin Luong
Daniel Vlasic
Bo Wu
Wenming Ye

Keras

Divyashree Sreepathihalli
Kiranbir Sodhia
Gabriel Rasskin
Matthew Watson
Varun Singh

Gemma Model

Alek Andreev
Armand Joulin
Surya Bhupatiraju
Minh Giang

Hugging Face Partners

Arthur Zucker
Lucain Pouget
Merve Noyan
Omar Sanseviero
Pablo Montalvo
Pedro Cuenca

Nvidia Partners

Maryam Moosaei
Fangzhou Mu
Santosh Bhavani
Anjali Shah
Vladislav Kozlov
Dong Meng
Niaz Syed
Chintan Patel
Ankit Patel
Marta Stepniewska-Dziubinska
Anna Warno
Xinqi Wang
David Tamok
Steven Baughman
Sandip Bhaskar
Jason Dudash

Executive Sponsors

Joelle Barral
Zoubin Ghahramani

A. More related work

There are generally two ways to build vision-language models (VLMs): the first option is to connect a vision encoder to a large language model while the second option is to use a transformer decoder-only architecture to handle both vision and language modalities.

It is popular to connect frozen image component and language component with lightweight adapters, e.g. linear or MLP projector, resampler [6, 48], Q-Former [62]. Flamingo [6] uses a perceiver-based [48] resampler to connect frozen vision and language models. Idefics2 [59] shows that the perceiver resampler works significantly better than a linear projector. BLIP2 [62] explores using trainable Q-Former [62] to align frozen vision and language models. They first train the Q-Former with frozen image model. Then attach the frozen image model and the Q-Former into frozen language model to continue the Q-Former training. LLaVA [70] opted to train a projection layer between frozen vision backbone and frozen language backbone, with GPT-4 generated small but high quality instruction-following data. Afterwards, they unfreeze the language backbone and finetune the projection layer and language model together. LLaVA-1.5 [69] extends this to MLP connector. Bunny [41] opted to use MLP connector for their models. Honeybee [18] introduced locality-enhanced projectors by using convolution and deformable attention, for better spatial understanding. MM1 [81] claimed that convolution adaptor [18] performs close to average pooling and attention pooling baselines. Cambrian-1 [107] performed a thorough study of vision encoders for VLMs, and proposed spatial vision aggregator to better integrate visual tokens. CogVLM [118] introduced additional trainable visual expert module in the attention and FFN layers of the frozen language model. This way, the language model is able to process visual tokens and language tokens with different experts, while keeping the same level of performance for text-only tasks. We performed an ablation study of linear and MLP vision-language connectors for PaliGemma in Section 5.5.

There are also methods exploring training both the vision and language components, PaliGemma falls into this category. Many VLM systems [10, 23–26] follow a multi-stage training procedure, including a stage to train both vision and language components. PaLI line of work [23–25] gradually scales up the training resolution with different data mixtures in three stages. Florence2 [122] and LocCa [115] train vision-centric models by modeling very diverse tasks with a universal language interface. Unified-IO 2 [72] trains a single encoder-decoder multimodal model on an ensemble of 120 datasets. Kosmos [45] trains the language model and the last layer of a CLIP vision model. BLIP [61] proposed to dump COCO like pseudo captions on million scale web data, and then use filters to choose from original noisy captions and pseudo captions to improve the quality of vision-language training data. BEIT3 [117] treats image data and language data the same way as discrete tokens. The image data is tokenized by the tokenizer of BEIT2 [89]. Image, text, and image-text pairs are randomly masked and the model is trained to recover the randomly masked tokens. EMU2 [103] operates in the continuous visual embedding space and jointly modeling the visual embeddings and text embeddings with a language decoder. The visual embeddings could later be decoded back to image pixels or video clips.

In the category of decoder-only VLMs, Fuyu [11] proposed to use an vision encoder-free architecture, that employs a transformer decoder-only model to process both image and text inputs. The input image is first patchified and then linearly projected to a shared continuous embedding space as text tokens, so that the decoder-only model could process image and text tokens seamlessly. CM3 [2, 3, 129] and Chameleon [105] proposed to convert images into discrete tokens and then model them together with language in the token space with a shared transformer decoder model. Our work also shared promising early results of a Fuyu-style decoder-only setup following this line of work, by ablating the SigLIP vision encoder component from PaliGemma in Section 5.6.

B. Tasks

We provide one training example for each transfer task, exactly as it reaches the model. The image shown has been resized to 224 × 224 pixels in order to convey how much can be recognized at that resolution.

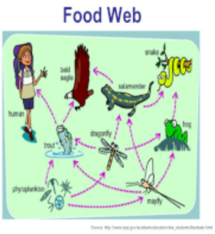
B.1. ActivityNet-CAP: Captioning of short video snippets of activities

Prefix:	`"caption en\n"`
Suffix:	`"The man washes pots and a cutting board."`
Train:	44775 examples (train+val).
Metric:	CIDEr (test split).
Reference:	Krishna et al. [57].

B.2. ActivityNet-QA: Answering questions about short video snippets of activities

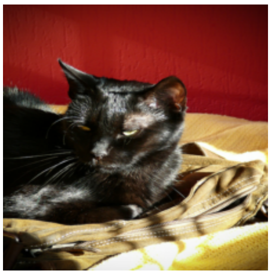
Prefix:	`"answer en what is the person in the video doing\n"`
Suffix:	`"bathe dog"`
Train:	43130 examples (train+val).
Metric:	Exact match accuracy (test split).
Reference:	Yu et al. [130].

B.3. AI2D: VQA on science diagrams



Prefix:	`"answer en Mayfly and dragonfly are eaten by ? choose from: Bald eagle \t Frog \t Phytoplankton \t None of the above\n"`
Suffix:	`"Frog"`
Train:	12413 examples (train split).
Metric:	Exact match accuracy (test split).
Reference:	Aniruddha Kembhavi [8]

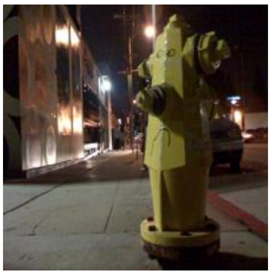
B.4. AOKVQA-DA: VQA using world knowledge (direct answer)



Prefix: "answer en What is a sound this animal makes?\n"
Suffix: "meow"

Train: 18201 examples (train+val splits).
Metric: Exact match accuracy (test server).
Reference: Schwenk et al. [98]

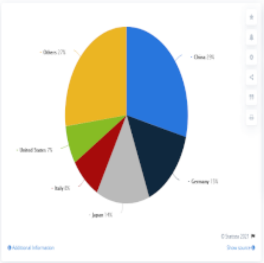
B.5. AOKVQA-MC: VQA using world knowledge (multiple choice)



Prefix: "answer en What has the yellow object been drawn on to resemble? choose from: eagle \t face \t dog \t star\n"
Suffix: "face"

Train: 18201 examples (train+val splits).
Metric: Accuracy (test server).
Reference: Schwenk et al. [98]

B.6. ChartQA: VQA about charts with logical reasoning



Prefix: "What country accounted for 15 percent of the world's machine tool production in 2020?\n"
Suffix: "Germany"

Train 30219 examples (human+augmented train and validation splits).
Metric: Relaxed match accuracy. ChartQA-human (human test split), ChartQA-aug (augmented test split).
Reference: Masry et al. [78]

B.7. COCOcap: COCO image captioning task

Prefix: "caption en\n"
Suffix: "A full view of a living room with pillows and books. "

Train: 113287 images each with 5 captions (train+restval splits).
Metric: CIDEr score (val split).
Reference: Lin et al. [67]

B.8. NoCaps: Novel object captioning

Prefix: `"caption en\n"`
Suffix: `"up close shot of street light with banners and tree's in the distance"`

Train: Zero-shot evaluation of the model trained for COCOcap.
Metric: CIDEr score (val split).
Reference: Agrawal et al. [4]

B.9. COCO-35L: COCO captions translated in 35 languages

Prefix: `"caption no\n"`
Suffix: `"pizza serveres p\u00e5 bordet med gafler og kniver .  ."`

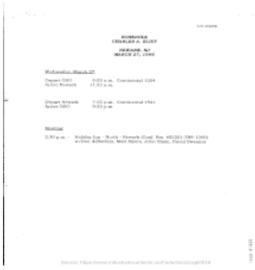
Train: 113287 images each with 5 captions in each of the 35 languages (train split).
Metric: Mean of CIDEr on each of the 35 languages (dev split).
Reference: Thapliyal et al. [106]
Note: The original raw data comes pre-tokenized with sometimes odd punctuation.

B.10. XM3600: caption geographically-diverse images in 36 languages

Prefix: `"caption pt\n"`
Suffix: `"Silhoueta de um corvo"`

Train: Zero-shot evaluation of the model trained for COCO-35L.
Metric: Mean of CIDEr on each of the 36 languages (dev split).
Reference: Thapliyal et al. [106]

B.11. DocVQA: VQA on document images



Prefix: `"What is the arrival time in GSO?\n"`
Suffix: `"9:33 p.m."`

Train: 44812 examples (train+val splits)
Metric: ANLS (test server).
Reference: Mathew et al. [79]

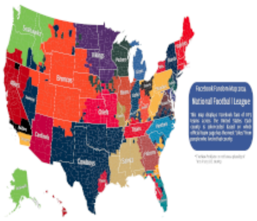
B.12. GQA: VQA on image scene graphs

Prefix:	`"answer en Where in the picture is the plate, in the bottom or in the top?\n"`
Suffix:	`"top"`
Train:	1075062 examples (train+val balanced splits).
Metric:	Exact-match accuracy (testdev balanced split).
Reference:	Hudson and Manning [47]

B.13. xGQA: Cross-lingual VQA

Prefix:	`"answer en Gibt es irgendwelche Schubladen oder Tische?\n"`
Suffix:	`"no"`
Train:	Zero-shot evaluation of the model trained for GQA.
Metric:	Exact-match accuracy (test zero-shot splits).
Reference:	Pfeiffer et al. [90]

B.14. InfoVQA: VQA on infographics



Prefix:	`"answer en WHich team has most number of FB likes in Tennessee\n"`
Suffix:	`"titans"`
Train:	26747 examples (train + val).
Metric:	ANLS (test server).
Reference:	Mathew et al. [80]

B.15. MSRVTT-CAP: Open-domain short video captioning

Prefix:	`"caption en\n"`
Suffix:	`"a woman is presenting a baby stroller"`
Train:	4965 examples (train+valid).
Metric:	CIDEr (test split).
Reference:	Xu et al. [124]

B.16. MSRVTT-QA: Open-domain sort video question answering

Prefix: `"answer en what does a person play?\n"`
Suffix: `"keyboard"`

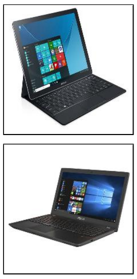
Train: 121646 examples (train+valid splits).
Metric: Exact match accuracy (test split).
Reference: Xu et al. [123]

B.17. MSVD-QA: Answering questions about short video segments of events

Prefix: `"answer en what is a woman wrapping?\n"`
Suffix: `"food"`

Train: 29749 examples (train+valid splits).
Metric: Exact match accuracy (test split).
Reference: Xu et al. [123] based on Chen and Dolan [20]

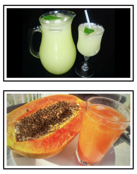
B.18. NLVR2: Reasoning about natural language grounded in photographs



Prefix: `"answer en Each image shows one opened laptop angled so the screen faces rightward.\n"`
Suffix: `"False"`

Train: 93355 examples (train+dev splits).
Metric: Accuracy (test split).
Reference: Suhr et al. [102]

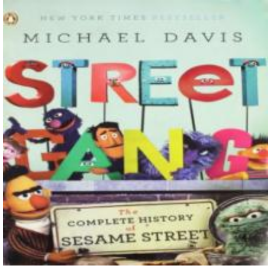
B.19. MaRVL: Reasoning about multilingual language grounded in multicultural photographs



Prefix: `"answer en Picha moja inaonyesha vyombo vyenye maji ya matunda pamoja na matunda pembeni na picha nyingine inaonyesha vyombo vyenye maji ya matunda peke yake.\n"`
Suffix: `"True"`

Train: Zero-shot evaluation of the model trained for NLVR2.
Metric: Accuracy (test splits).
Reference: Liu et al. [68]

B.20. OCR-VQA: VQA by reading text in images


NEW YORK TIMES BESTSELLER
MICHAEL DAVIS
STREET GANG
The COMPLETE HISTORY of SESAME STREET

Prefix: "answer en Is this a comedy book?\n"
Suffix: "Yes"

Train: 901717 examples (train+val splits).
Metric: Exact match accuracy (test split).
Reference: Mishra et al. [84]

B.21. OKVQA: Outside knowledge VQA

Prefix: "answer en What type of race is this?\n"
Suffix: "truck"

Train: 9009 examples (train split).
Metric: Exact match accuracy (val split).
Reference: Marino et al. [77]

B.22. RefCOCO_seg: Referring expression segmentation

Prefix: "the giraffe on the right standing tall\n"
Suffix: "<loc0347><loc0553><loc0788><loc0749>
<seg093><seg106><seg114><seg078><seg064>
<seg012><seg031><seg055><seg050><seg012>
<seg083><seg118><seg084><seg078><seg127>
<seg121>"

Train: 24407 examples: The combined training sets of RefCOCO, RefCOCO+, and RefCOCOg, *but with all val and test images removed*, see LocCa [115] for details.
Metric: mean intersection over union (mIoU) on all test splits.
Reference: RefCOCO and RefCOCO+: Kazemzadeh et al. [53], Yu et al. [128] and RefCOCOg: Mao et al. [76].

B.23. RSVQA-hr: VQA for remote sensing (high res)

Prefix: "answer en Are there more commercial buildings than roads?\n"
Suffix: "no"

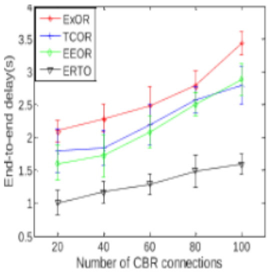
Train: 432239 examples (non-numeric train+val splits).
Metric: Mean accuracy across questions types (reported on test and test2 splits).
Reference: Lobry et al. [71]

B.24. RSVQA-lr: VQA for remote sensing (low res)

Prefix: "answer en Is the number of buildings equal to the number of water areas in the image?\n"
Suffix: "no"

Train: 47173 (non-numeric train+val splits).
Metric: Average accuracy across questions types (non-numeric test split).
Reference: Lobry et al. [71]

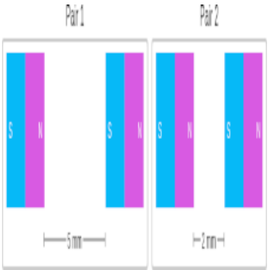
B.25. SciCap: Captions for scientific figures



Prefix: "caption en\n"
Suffix: "end-to-end delay under different traffic load ."

Train: 120188 examples (first sentence, no subfigure train+val splits).
Metric: CIDEr (first sentence, no subfigure test split).
Reference: Hsu et al. [44]

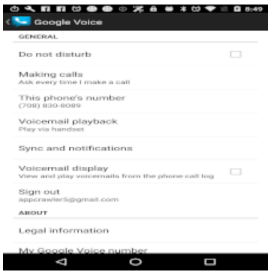
B.26. ScienceQA: Science question answering



Prefix: "Question: Think about the magnetic force between the magnets in each pair. Which of the following statements is true?\nContext: The images below show two pairs of magnets. The magnets in different pairs do not affect each other. All the magnets shown are made of the same material.\nOptions: (A) The magnetic force is weaker in Pair 1., (B) The magnetic force is weaker in Pair 2., (C) The strength of the magnetic force is the same in both pairs.\nAnswer:\n"
Suffix: "The answer is A."

Train: 8315 examples (train+val splits).
Metric: Exact-match accuracy (test split).
Reference: Lu et al. [74]

B.27. Screen2Words: Mobile UI summarization


8:49

Google Voice

GENERAL

Do not disturb □

Making calls
Ask every time I make a call

This phone's number
(708) 830-8089

Voicemail playback
Play via handset

Sync and notifications

Voicemail display □
View and play voicemails from the phone call log

Sign out
appcrawler5@gmail.com

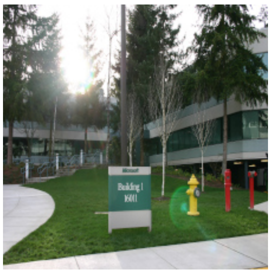
ABOUT

Legal information

My Google Voice number

Prefix:	`"caption en\n"`
Suffix:	`"screen showing general settings page"`
Train:	18107 (train+dev splits).
Metric:	CIDEr (test split).
Reference:	Wang et al. [116]

B.28. ST-VQA: Scene text VQA



Prefix:	`"answer en What company is this?\n"`
Suffix:	`"microsoft"`
Train:	26074 examples (train+val).
Metric:	ANLS (test server).
Reference:	Biten et al. [15]

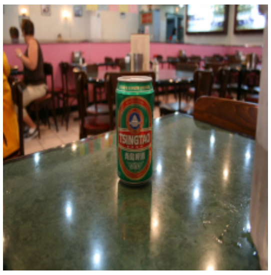
B.29. TallyQA: Complex counting questions

Prefix:	`"answer en How many bowls are in the picture?\n"`
Suffix:	`"1"`
Train:	249318 examples (train split).
Metric:	Accuracy (test split) reported on simple and complex splits.
Reference:	Acharya et al. [1]

B.30. CountBenchQA: Evaluate counting in a structured, controlled way

Prefix:	`"answer en How many sculptures are there in the image?\n"`
Suffix:	`"7"`
Train:	Zero-shot evaluation of the model trained for TallyQA.
Metric:	Exact match (full test split).
Reference:	Introduced in this report, based on Paiss et al. [88]

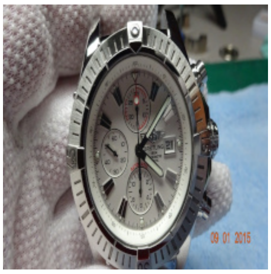
B.31. TextCaps: Image captioning with reading comprehension



Prefix: `"caption en\n"`
Suffix: `"green and red beer can for tsingtao inside a restaurant."`

Train: 21953 examples (train split).
Metric: CIDEr (test split).
Reference: Sidorov et al. [100]

B.32. TextVQA: Visual reasoning based on text in images



Prefix: `"answer en what brand of watch is this?\n"`
Suffix: `"breitling"`

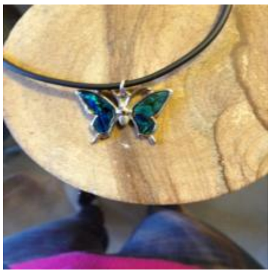
Train: 39602 (train+val splits).
Metric: test server (test-std).
Reference: Singh et al. [101]

B.33. VATEX-CAP: Broad video captioning

Prefix: `"caption en\n"`
Suffix: `"A person swims in a pool and touches the wall while others cheer."`

Train: 24850 examples (train + valid splits).
Metric: CIDEr (test split).
Reference: Wang et al. [119].

B.34. VizWizVQA: VQA from people who are blind



Prefix: `"answer en What colors are in the charm of this necklace?\n"`
Suffix: `"silver green blue"`

Train: 24842 examples (train+val).
Metric: test server (test-std).
Reference: Gurari et al. [40]

B.35. VQAV2: Visual Question Answering

Prefix: `"answer en Does this look like a place you'd love to swim?\n"`
Suffix: `"yes"`

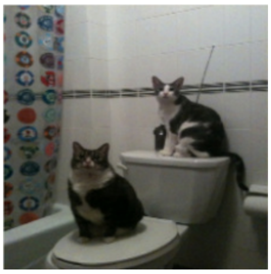
Train: 658111 examples (train+validation)
Metric: `VQAV2`: test server (test-std).
`VQAV2 (minival)`: computed on local split of 10k examples of the validation set.
Reference: Goyal et al. [39]

B.36. MMVP: Hard questions about CLIP-blind image pairs

Prefix: `"answer en From which angle is this image taken?\n"`
Suffix: `"Front"`

Train: Zero-shot evaluation of the model trained for VQAv2.
Metric: Paired accuracy.
Reference: Tong et al. [108]

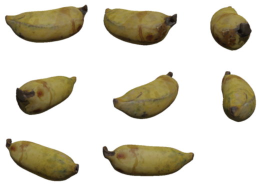
B.37. POPE: Object presence as yes/no VQA (object hallucination)



Prefix: `"answer en Is there a car in the image?\n"`
Suffix: `"no"`

Train: Zero-shot evaluation of the model trained for VQAv2.
Metric: Exact match with “yes” or “no”. The overall score is the average of accuracies on the `random`, `popular`, and `adversarial` splits, which all have the same size.
Reference: Yifan Li and Wen [127].
Note: Car, not cat.

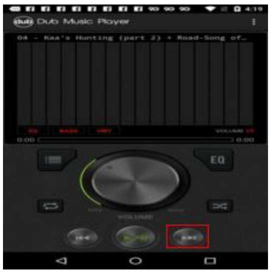
B.38. Objaverse Multiview: view-consistent 3D object annotation



Prefix: "answer en Describe the object in the image?\n"
Suffix: "banana"

Train: Zero-shot evaluation of the model trained for VQAv2.
Metric: Cosine similarity according to Universal Sentence Encoder v4.
Reference: Kabra et al. [50]
Note: Each image is seen individually, the final prediction is the score-weighted average. 44k objects, 1100 categories, 8 views rendered per object, 4 prompts per object view.

B.39. WidgetCap: Descriptions for Mobile User Interface Elements



Prefix: "caption en\n"
Suffix: "fast forward"

Train: 44704 examples (train+dev splits). We annotate the widget to be captioned by overlaying a red bounding box in the image input.
Metric: CIDEr (test split).
Reference: Li et al. [63]

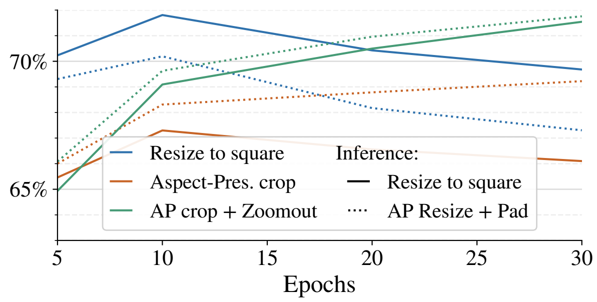


Figure 13 | Three different ways to get 448 × 448 images during training: plain resizing, random aspect-ratio preserving crop, or the latter with an additional random "zoom-out" augmentation. Two options at inference-time: plain resizing or performing an aspect-ratio preserving resize with padding. Lines are not training curves, but multiple training runs of 5, 10, 20, and 30 epochs.

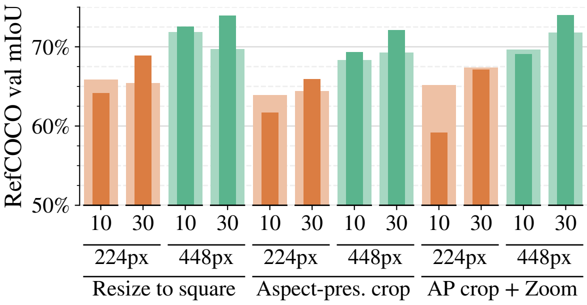


Figure 14 | Wide, outer bars show performance without label-smoothing and dropout (LSDO), thin inner bars with. 10 and 30 denote epochs. First, LSDO always hurts the 224 px 10 ep setting but always helps the 448 px 30 ep setting. Second, resize to square overfits in longer training, but adding LSDO remediates this and allows such simple method to outperform more complicated image-augmentation based ones.

C. Image augmentations for RefCOCO

The current wisdom regarding structured output tasks such as detection and segmentation is that image augmentations are crucial, and especially keeping the original aspect ratio while performing zoom-like augmentations is key.

We found this *not* to be the case, at least in the setting of pretrained VLMs. After extensive experiments, we found that simply resizing the image to a square 224 × 224 pixels achieves best (and state-of-the-art) performance.

Figure 13 shows that one needs to explore both training-time resizing technique and inference-time strategy simultaneously. It seems natural that, when performing aspect-ratio preserving resize with padding at inference time, the training augmentations need to reflect that to some degree. However, one can also simply resize test images to a square at inference time, and then resize the output logits back to the original resolution for evaluation. Doing so works very well in tandem with simple resizing to square at training time and, as one might expect, does not work well with aspect-preserving resizes during training.

Besides this training-inference mismatch, Figure 13 also shows that the plain resize suffers from overfitting when training for longer than 10 epochs, while cropping and zooming do not, as they also act as image augmentations. However, overfitting can also be combatted via other, simpler techniques than image processing.

Figure 14 shows that adding label-smoothing and dropout to the training not only completely eliminates the overfitting, but also allows the simple resize strategy to achieve the same or better performance than the more complicated image augmentation ones. Hence, for our final setting that reaches state-of-the-art results, we stick to simple image resizing with increased training schedule, label-smoothing, and dropout.

D. Introducing CountBenchQA

TallyQA is the only dataset commonly used to evaluate counting, an important skill for VLMs. However, we found that it has two issues [51]: First, the ground-truth label distribution is highly skewed towards small counts (0, 1, 2). Second, while the dataset is large, the ground-truth labels themselves are rather noisy, and the validation and test splits have not undergone any further verification or clean-up.

The recently introduced CountBench [88] dataset of 540 images fixes both these shortcomings. Images and captions are taken from the LAION-400M image-text dataset and each caption mentions the main object class present in the image and specifies its count. The correspondence of images, counts and captions has been manually verified.

There are 540 images in total with 60 images per count in [2, 3, ... 10]. There is no 0 or 1 count, the smallest count is 2. The data is not originally available in a VQA format but just as (image, caption, count) triplets. To construct CountbenchQA, we manually annotated the dataset with questions of the form `How many {object class} are there in the image?`, where we extracted the `{object class}` from the caption manually. where the object class was overly specific or difficult to understand we replaced them with simpler versions (*e.g.* we changed “glass star christmas tree decorations” to "stars", and “founders of Kappa Sigma” to “people”.) There are only 3 exceptions deviating from the above template (these were needed to ensure correctness of the answer in ambiguous cases) and these are:

- How many people are in the foreground of this image?
- How many stories does this cottage have?
- How many petals does each flower have in this image?

For some counts, several images look extremely similar. For 9 items, several images are button/icon symbols with slightly different colours/symbols on them, arranged in a regular grid.

Out of the 540 images, several are not available anymore; the parquet file provided at `https://huggingface.co/datasets/nielsr/countbench` contains entries for all 540 examples but only image data for 490 of those examples. This is why we have two splits, although we computed our metrics using the full split and have not recomputed them since. Our dataset is available at `https://github.com/google-research/big_vision/blob/main/big_vision/datasets/countbenchqa/`.

E. Issues with published WidgetCaps numbers

We have found that at least two prior works reporting results on WidgetCaps do so wrongly. First, previous PaLI numbers (PaLI-X: 153.0, PaLI-3: 159.8) are reported on the validation set while implying they are test-set results. This makes comparison with test-set numbers such as our 148.4 invalid. However, a re-run of our transfers in a comparable setting (training on `train`, evaluating on `dev`) achieves a validation set CIDEr of 140.2 at 224 px and 155.2 at 448 px resolution, placing it firmly between PaLI-X and PaLI-3 at much smaller model size and resolution.

Furthermore, ScreenAI [9] had a mistake in the CIDEr computation for WidgetCap, using only a single out of the five provided ground-truth captions. Re-computing their CIDEr score using all five ground-truth captions changes the score from the 167 originally reported in the paper to 156.4, which is close to PaliGemma's 148.4.

F. Image annotations work as well as prompts

The WidgetCaps requires captioning a specific widget in the image. The widget is given by bounding-box coordinates, which are typically provided in the prompt, including in previous PaLI versions.

Here, we found that drawing a red box in the image during transfers, and not indicating it in the prompt, performs equally well. For example, the best validation result when sweeping hyper-parameters in both settings equally, was 135.99 CIDEr for the drawn red box, and 135.28 for the box coordinates as `<loc>` tokens in the prompt. This difference is well within re-run variation.

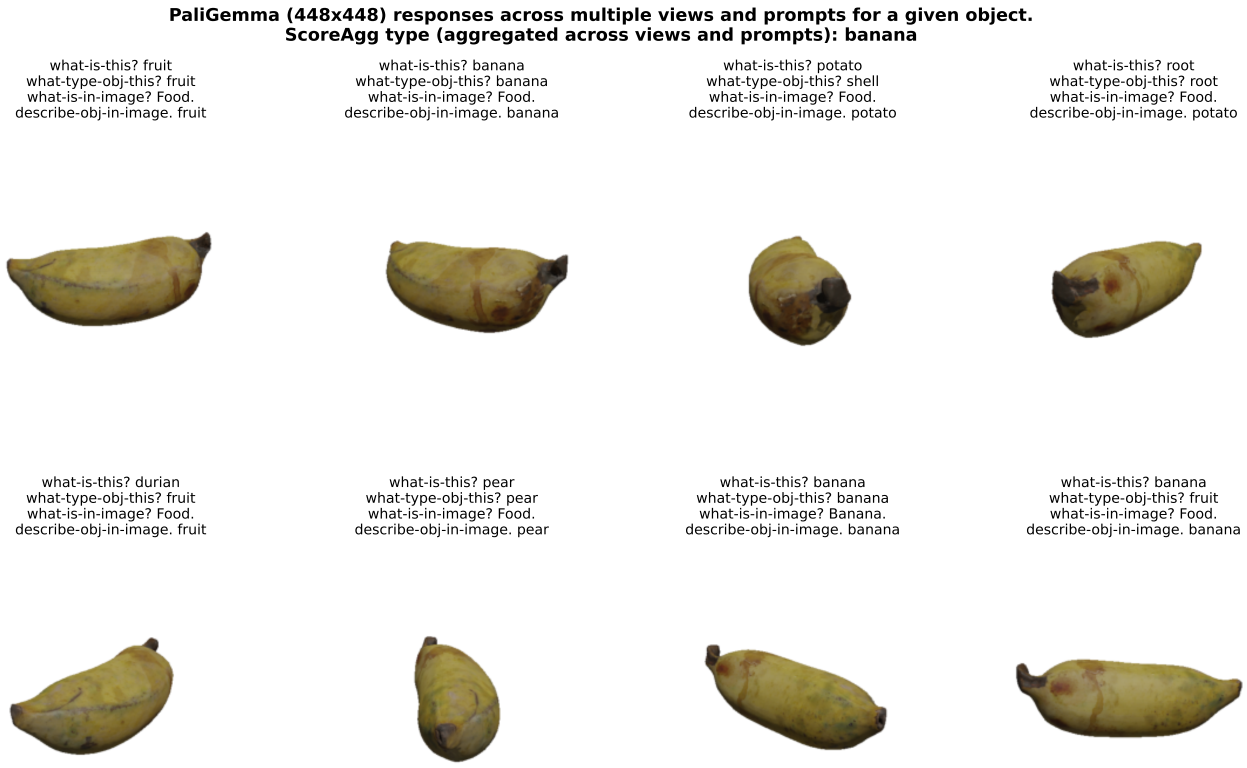

PaliGemma (448x448) responses across multiple views and prompts for a given object.
ScoreAgg type (aggregated across views and prompts): banana
what-is-this? fruit
what-type-obj-this? fruit
what-is-in-image? Food.
describe-obj-in-image. fruit
what-is-this? banana
what-type-obj-this? banana
what-is-in-image? Food.
describe-obj-in-image. banana
what-is-this? potato
what-type-obj-this? shell
what-is-in-image? Food.
describe-obj-in-image. potato
what-is-this? root
what-type-obj-this? root
what-is-in-image? Food.
describe-obj-in-image. potato
what-is-this? durian
what-type-obj-this? fruit
what-is-in-image? Food.
describe-obj-in-image. fruit
what-is-this? pear
what-type-obj-this? pear
what-is-in-image? Food.
describe-obj-in-image. pear
what-is-this? banana
what-type-obj-this? banana
what-is-in-image? Banana.
describe-obj-in-image. banana
what-is-this? banana
what-type-obj-this? fruit
what-is-in-image? Food.
describe-obj-in-image. banana

Figure 15 | One representative example of PaliGemma's predictions across the different views.

G. More details and results with Objaverse

Objaverse [31] is an internet-scale collection of 800 k diverse but noisily or unannotated 3D models. They were uploaded by 100 k artists to the Sketchfab platform. A subset of 44 k objects called Objaverse-LVIS is accompanied by human-verified categories. These can be used to validate object class predictions. This amounts to only 5% of examples with human-verified labels in a restricted subset of categories.

A compelling use of VLMs is to infer the type of each Objaverse object in an open-vocabulary setting, providing multiple views of each object to the model. Furthermore, this is an interesting evaluation benchmark for VLMs: the images come from a more specialized distribution of 3D renders (often untextured), and the query of object type is something we should expect a VLM to handle well.

The previously most popular method, CAP3D [75], is an ad-hoc combination of BLIP2, CLIP, and GPT4, where GPT4 summarizes annotations created by BLIP2 and CLIP. Given access to log-probabilities, [50] show that the aggregating the predictions of each view informed by their log-probabilities (`ScoreAgg`), largely outperforms CAP3D while being significantly simpler. See [50] for more details on the exact setup.

We additionally evaluate PaLI-3, and the open-weight PaliGemma base and VQAv2 fine-tune. We use four VQA prompts to probe for the type of each object: (i) "What is this?" (ii) "What type of object is this?" (iii) "What is in the image?" (iv) "Describe the object in the image". This produces four sets of five sampled responses per view, which are aggregated using ScoreAgg to produce a final prediction. The results are summarized in Table 3 and show that PaliGemma has clear 3D object understanding out of the box, and the VQAv2 fine-tune performs even better. Finally, we also show a representative example of raw predictions in Figure 15; a banana for s(c)ale, if you will.

Table 3 | Results of various Objaverse captioning methods. The mean, standard deviation, and standard error of the mean are computed across the 44 k examples, after the per-example aggregation.

Model	**Score** (mean)	(stddev)	(stderr)
CAP3D (Luo et al., 2023)	36.6	-	-
PaLI-X (55B, 756x756, mixed VQA transfer)	59.0	29.1	0.1
PaLI-3 (5B, 448x448, mixed VQA transfer)	**64.3**	29.2	0.1
PaliGemma (3B, 224x224)	58.4	29.5	0.1
PaliGemma (3B, 224x224, VQAv2 transfer)	62.7	28.2	0.1
PaliGemma (3B, 448x448)	60.4	28.9	0.1
PaliGemma (3B, 448x448, VQAv2 transfer)	**62.8**	28.3	0.1

Table 4 | Results when multitasking. Table 1 is the best achievable single-task result with per-task tuned hyper-paramters. "Single Simple" is then the per-task performance when using the same simplified hyper-parameter setting for all tasks. Finally, "Multi" is the multitasking setup where a single model performs all tasks, either with or without prefix indicating the task. The number in parenthesis is the relative regret versus the preceding column.

Metric	Table 1	Single Simple	Multi Prefix	Multi No Prefix
AI2D	72.1	73.1	72.3 (-1.0%)	72.3 (-1.1%)
COCOcap	141.9	141.7	139.5 (-1.6%)	138.9 (-2.0%)
NoCaps	121.7	121.6	118.4 (-2.6%)	98.9 (-18.7%)
DocVQA (val)	37.8	38.6	32.2 (-16.7%)	31.3 (-19.0%)
GQA	65.6	65.4	64.9 (-0.7%)	64.2 (-1.8%)
xGQA (avg7)	57.3	56.9	55.8 (-2.0%)	52.6 (-7.7%)
InfoVQA (val)	25.5	25.0	20.8 (-17.0%)	21.4 (-14.4%)
OCR-VQA	72.3	73.3	71.4 (-2.6%)	71.4 (-2.6%)
OKVQA	63.5	63.1	67.0 (+6.2%)	58.9 (-6.6%)
RefCOCO (val)	73.4	69.9	68.4 (-2.1%)	68.3 (-2.3%)
RefCOCO+ (val)	68.3	61.0	61.2 (+0.3%)	60.9 (-0.2%)
RefCOCOg (val)	67.7	63.7	61.8 (-3.1%)	61.8 (-3.0%)
RSVQA-hr (test)	92.6	92.7	92.1 (-0.6%)	92.2 (-0.5%)
RSVQA-hr (test2)	90.6	90.8	90.0 (-0.9%)	90.2 (-0.7%)
RSVQA-lr	92.6	93.7	93.4 (-0.3%)	93.0 (-0.8%)
SciCap	162.3	64.0	74.7 (+16.7%)	74.7 (+16.6%)
Screen2Words	117.6	114.7	111.6 (-2.7%)	110.6 (-3.5%)
ST-VQA (val)	61.6	62.0	58.3 (-6.0%)	57.8 (-6.8%)
TallyQA (simple)	81.7	81.3	80.4 (-1.1%)	80.7 (-0.7%)
TallyQA (complex)	69.6	69.6	69.6 (-0.1%)	69.9 (+0.3%)
CountBenchQA	81.9	83.5	80.6 (-3.5%)	78.3 (-6.2%)
TextCaps	127.5	128.3	122.5 (-4.5%)	122.3 (-4.6%)
TextVQA (val)	59.0	57.9	56.0 (-3.2%)	56.5 (-2.5%)
VQAv2 (minival)	82.1	81.9	83.4 (+1.8%)	82.9 (+1.2%)
VizWizVQA (val)	76.4	75.9	75.7 (-0.2%)	76.0 (+0.2%)
WidgetCap	136.1	136.4	129.1 (-5.3%)	131.5 (-3.6%)
Average	84.6	80.2	78.9 (-2.0%)	77.6 (-3.5%)

H. Multitask transfer

Transferring to individual tasks by fine-tuning is good when all one cares about is having a model that solves a specific task. However, it is often desirable to have a single generalist model with a conversational interface. This is typically achieved by instruction tuning, *i.e.* fine-tuning on a mixture of a diverse dataset. We verify that PaliGemma is well-suited for this type of transfer: we transfer it to a mix of 27 of our datasets (excluding some tasks, such as multi-image, for simplicity). We follow the "simplified" hyper-parameter from Section 6.2 for the mixture, and mix uniformly such that each individual task goes through 10 epochs, no matter its size.

The results in Table 4 show that there is a dramatic change in a few tasks, while most win or lose just a little. Surprisingly, the overall average shows the largest loss comes from changing to a unified hyper-parameter, followed by multitasking, and removing per-task prefix indicator comes with the smallest regret. However, it should be noted that this multitasking setup was not tuned much.

Table 5 | Inference measurements on a TPUv3 [38] with 8 devices (4 chips), prefilling 512 tokens, cache sized 640 tokens. Even though FSDP sharding [137] is very efficient at training time (Section 3.2.6), at inference time it has the downside that each device needs process at least one single example and needs to read all model parameters. It reduces the memory requirements but it still processes one single example as slow as using a single device with a larger memory. Megatron-style sharding [99] on the other hand shards both the parameters *and* the activations, and can make use of multiple accelerators in parallel to process a single example and significantly reduce the total number of memory reads per device.

			Walltime [ms]		**Tokens/sec**		**Utilization [%]**		**Memory [GiB]**	
Sharding	**Params**	**Batch**	Prefill	Extend	Prefill	Extend	Prefill	Extend	Prefill	Extend
FSDP	`float32`	1	107	14.6	4785	68	34.9	0.6	2.04	1.69
		8	107	14.7	38282	545	34.9	0.6	2.04	1.69
		64	772	14.7	42423	4367	38.6	4.5	2.67	1.87
		512	6262	35.0	41866	14617	38.1	14.8	7.48	3.21
	`bfloat16`	1	101	8.2	5073	122	37.0	1.0	0.84	0.64
		8	101	8.3	40555	969	36.9	1.0	0.84	0.64
		64	764	10.2	42914	6258	39.1	6.4	1.48	0.81
		512	6235	30.8	42043	16601	38.3	16.8	6.28	2.14
Megatron	`float32`	1	22	8.2	23018	122	20.6	0.1	2.23	1.61
		8	105	10.3	39149	777	34.9	0.7	2.32	2.01
		64	744	13.7	44055	4660	39.2	4.1	3.16	2.33
		512	6360	41.2	41219	12427	36.7	11.5	7.82	3.75
	`bfloat16`	1	17	5.3	30006	189	26.9	0.3	0.92	0.82
		8	99	6.5	41284	1239	36.7	1.2	1.00	0.80
		64	739	9.4	44348	6835	39.4	6.4	1.85	0.98
		512	6348	34.9	41297	14677	36.7	13.1	6.54	2.41

I. Inference

See table 5 for inference measurements using the code from `big_vision`.

J. Hyper-parameters

For each transfer task, we provide the exact hyper-parameters with which the final score and the publicly released checkpoint was obtained. Tuning was done by looking at a validation-set if available, otherwise on a small "minival" held out from the training set. Most tuning was done at 224px resolution, and either carried over as-is to higher resolutions, or further tuned only slightly at higher resolution. Zero-shot evaluations were done on the checkpoint obtained for the fine-tune task and using hyper-parameters selected based on the fine-tune task validation metric. This can lead to worse performance as the hyper-parameters were not selected for generalization to different test distributions. The full configuration of every transfer task can be found in the provided source-code.

Task	Res	Epochs	Batch size	Learning rate	Weight decay	LLM dropout	Label smoothing	Freeze ViT	Decode
COCOcap	224	5	256	1e-05	1e-06	0.0	0.0	False	beam n=2
	448	5	256	1e-05	1e-06	0.0	0.0	False	beam n=2
COCO-35L	224	25	256	1e-05	1e-06	0.0	0.0	False	greedy
	448	25	256	1e-05	1e-06	0.0	0.0	False	greedy
Screen2Words	224	10	256	1e-05	0.0	0.3	0.2	False	greedy
	448	10	256	1e-05	0.0	0.3	0.2	False	greedy
TextCaps	224	5	256	1e-05	1e-06	0.0	0.0	False	beam n=3
	448	5	256	1e-05	1e-06	0.0	0.0	False	beam n=3
SciCap	224	80	256	3e-05	0.0	0.1	0.1	False	greedy
	448	80	256	3e-05	0.0	0.1	0.1	False	greedy
WidgetCap	224	4	64	3e-06	3e-07	0.1	0.1	False	greedy
	448	4	64	3e-06	3e-07	0.1	0.1	False	greedy
VQAv2	224	10	256	1e-05	1e-06	0.0	0.0	False	greedy
	448	10	256	1e-05	0.0	0.0	0.0	False	greedy
OKVQA	224	10	128	5e-06	0.0	0.0	0.0	False	greedy
	448	10	128	5e-06	0.0	0.0	0.0	False	greedy
AOKVQA-MC	224	15	128	5e-06	0.0	0.0	0.0	False	greedy
	448	15	128	5e-06	0.0	0.0	0.0	False	greedy
AOKVQA-DA	224	10	128	5e-06	0.0	0.0	0.0	False	greedy
	448	10	128	5e-06	0.0	0.0	0.0	False	greedy
GQA	224	1	256	1e-05	0.0	0.0	0.0	False	greedy
	448	1	256	1e-05	0.0	0.0	0.0	True	greedy
NLVR2	224	3	256	1e-05	1e-06	0.0	0.0	False	greedy
	448	10	256	3e-06	3e-07	0.0	0.0	False	greedy
AI2D	224	10	256	1e-05	1e-06	0.0	0.0	False	greedy
	448	10	256	1e-05	1e-06	0.0	0.0	False	greedy
ScienceQA	224	20	128	1e-05	0.0	0.0	0.0	True	greedy
	448	20	128	1e-05	0.0	0.0	0.0	True	greedy
RSVQA-lr	224	3	256	3e-06	0.0	0.0	0.2	False	greedy
	448	3	256	3e-06	0.0	0.0	0.2	False	greedy
RSVQA-hr	224	1	256	1e-05	0.0	0.0	0.0	False	greedy
	448	1	256	1e-05	0.0	0.0	0.0	False	greedy
ChartQA	224	30	256	1e-05	1e-06	0.1	0.2	False	greedy
	448	30	256	1e-05	1e-06	0.1	0.2	False	greedy
VizWizVQA	224	10	256	1e-05	0.0	0.0	0.0	False	greedy
	448	10	256	1e-05	0.0	0.0	0.0	False	greedy
TallyQA	224	2	256	1e-05	0.0	0.0	0.0	False	greedy
	448	2	256	1e-05	1e-06	0.0	0.0	False	greedy
OCR-VQA	224	3	128	3e-06	0.0	0.0	0.0	False	greedy
	448	3	128	3e-06	0.0	0.0	0.0	False	greedy
	896	3	128	1e-05	0.0	0.0	0.0	False	greedy
TextVQA	224	5	256	3e-06	0.0	0.0	0.0	False	greedy
	448	10	256	3e-06	3e-07	0.0	0.0	False	greedy
	896	10	256	3e-06	0.0	0.0	0.0	False	greedy
DocVQA	224	10	256	1e-05	1e-06	0.0	0.0	False	greedy
	448	10	256	1e-05	1e-06	0.0	0.0	False	greedy
	896	10	256	1e-05	1e-06	0.0	0.0	False	greedy
InfoVQA	224	3	256	1e-05	1e-06	0.0	0.4	False	greedy
	448	3	128	1e-05	1e-06	0.0	0.4	False	greedy
	896	3	32	3e-06	3e-07	0.0	0.4	False	greedy
ST-VQA	224	3	256	1e-05	1e-06	0.0	0.1	False	greedy
	448	3	128	1e-05	1e-06	0.0	0.1	False	greedy

Task	Res	Epochs	Batch size	Learning rate	Weight decay	LLM dropout	Label smoothing	Freeze ViT	Decode
	896	3	32	3e-06	3e-07	0.0	0.1	False	greedy
RefCOCO	224	100	256	3e-05	0.0	0.1	0.3	False	greedy
	448	100	256	1e-05	0.0	0.0	0.3	False	greedy
	896	100	64	1e-05	0.0	0.0	0.3	False	greedy
ActivityNet-QA	224	1	128	1e-05	1e-06	0.0	0.0	False	greedy
ActivityNet-CAP	224	1	128	1e-05	1e-06	0.0	0.0	True	greedy
MSRVTT-QA	224	1	128	1e-05	0.0	0.0	0.0	True	greedy
MSRVTT-CAP	224	20	128	1e-05	0.0	0.0	0.0	True	greedy
MSVD-QA	224	1	128	3e-06	3e-07	0.0	0.0	False	greedy
VATEX	224	10	128	3e-06	3e-07	0.0	0.0	False	greedy

K. Full per-task results of ablations

K.1. Pretraining duration

See Figure 16.

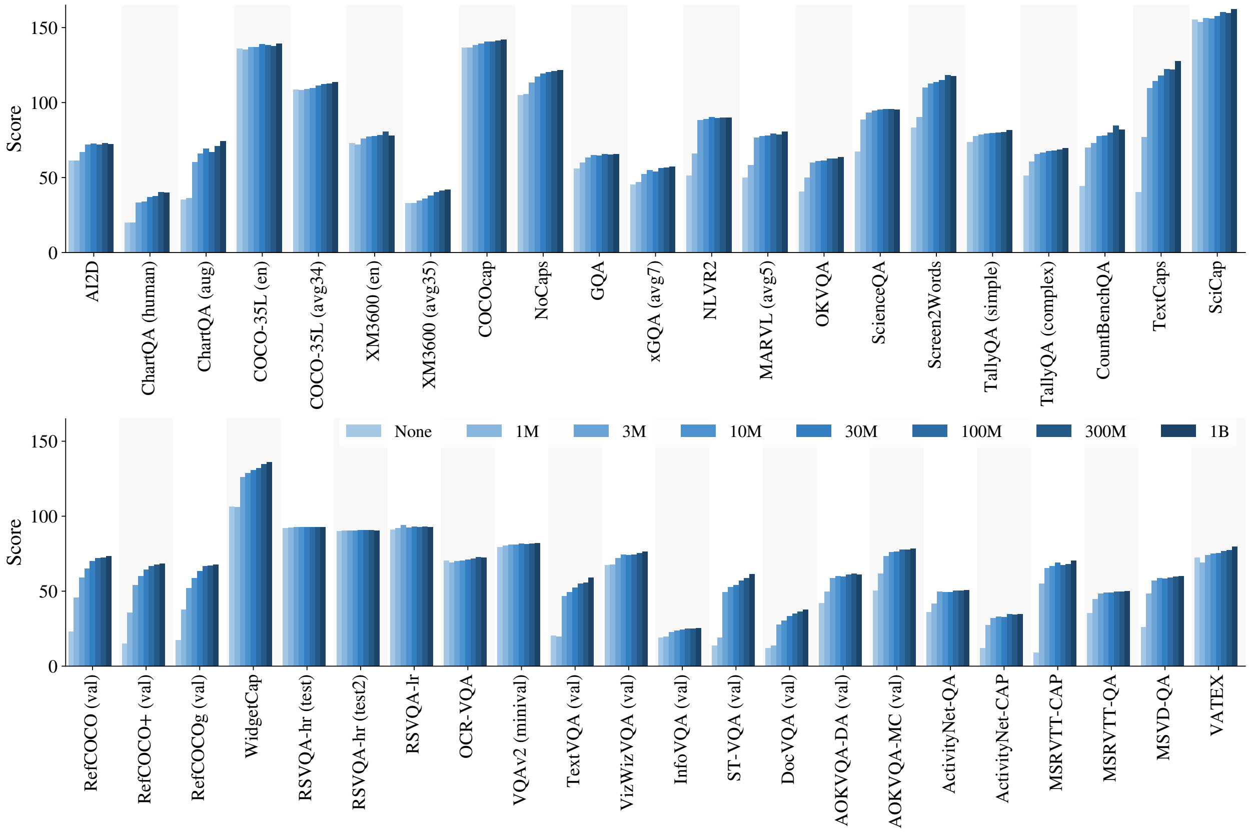


Figure 16 | Most tasks benefit significantly from longer pretraining. The notable exception are the remote sensing tasks that have a significantly different image distribution (see examples in Appendix B). Discussion in Section 5.1.

K.2. Masking and learning objective

See Figure 17 and Figure 18.

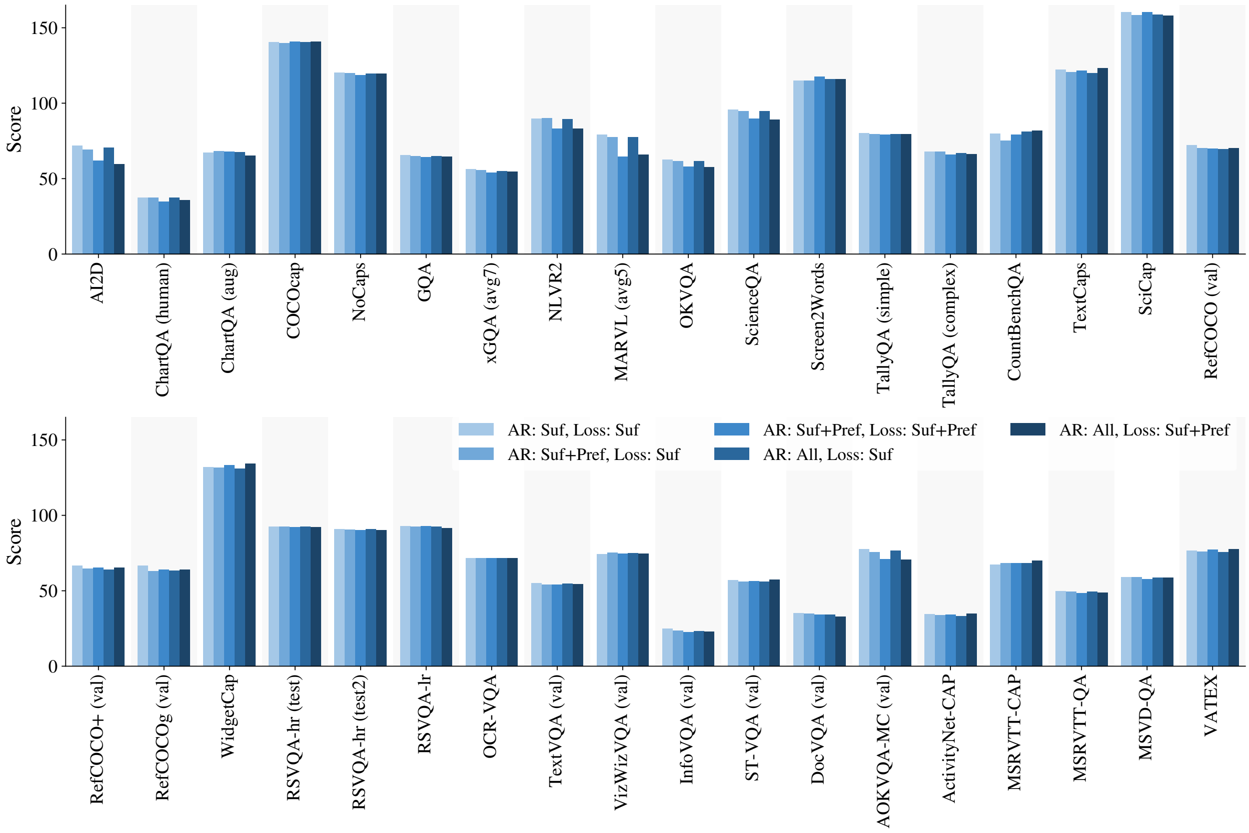


Figure 17 | Per task results when changing the learning objective. Discussion in Section 5.2.

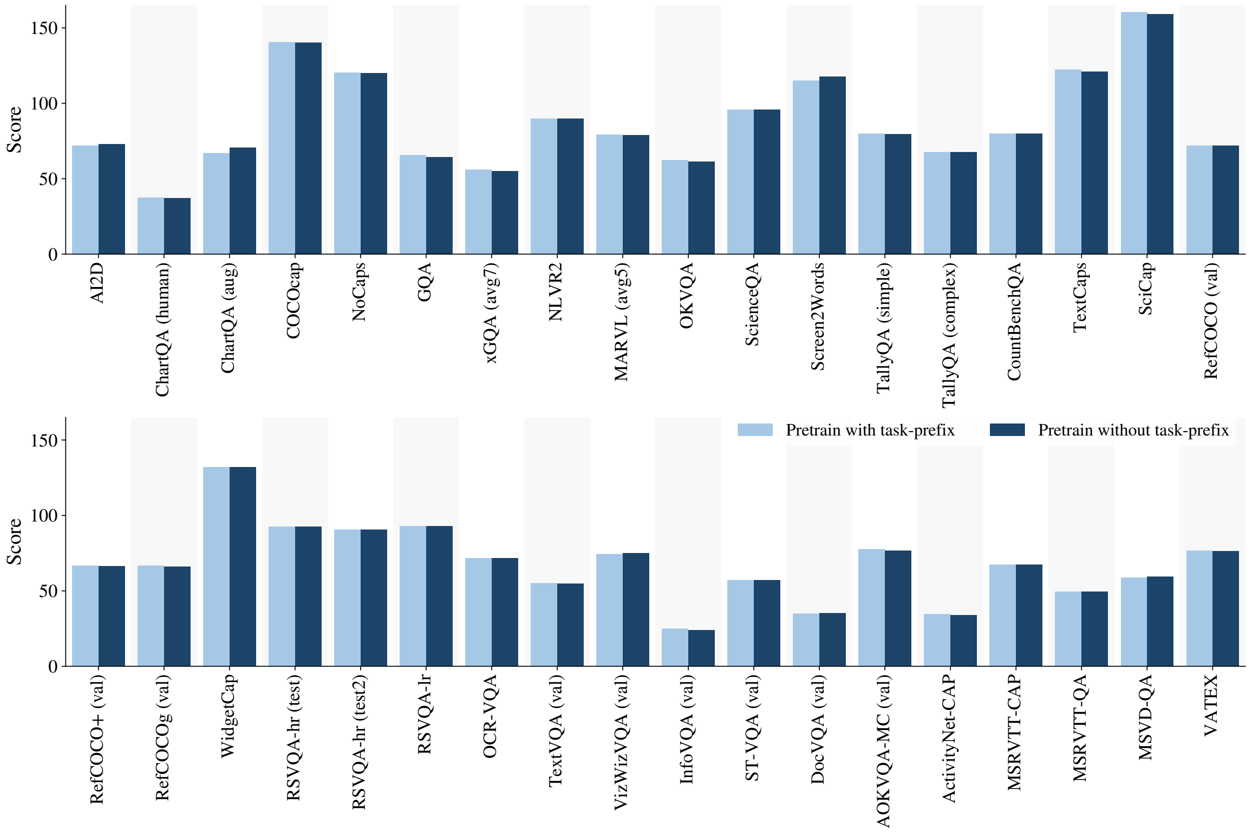


Figure 18 | Per task results pretraining with or without task-prefix. Discussion in Section 5.2.

K.3. To freeze or not to freeze?

See Figure 19.

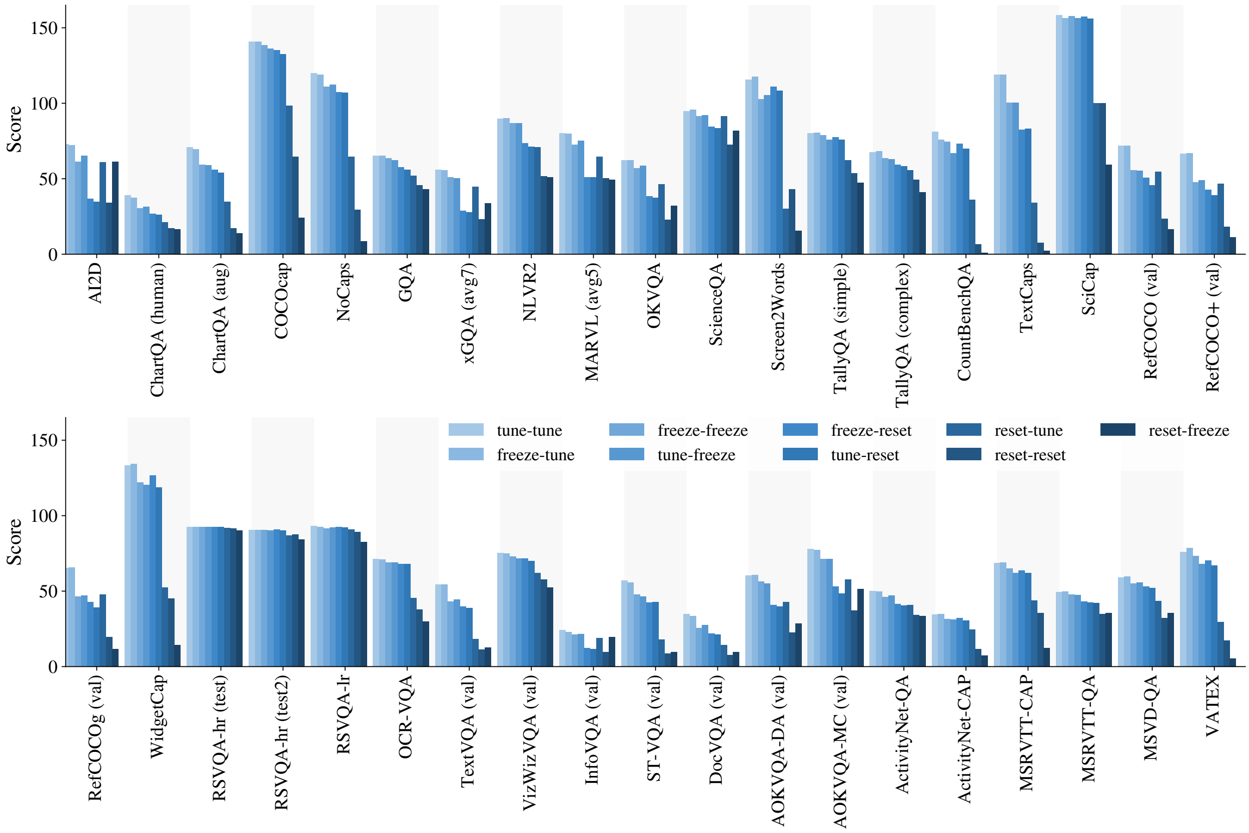


Figure 19 | Per task results when using various freezing and resetting patterns. Discussion in Section 5.4.

K.4. Image encoder: with or without?

See Figure 20.

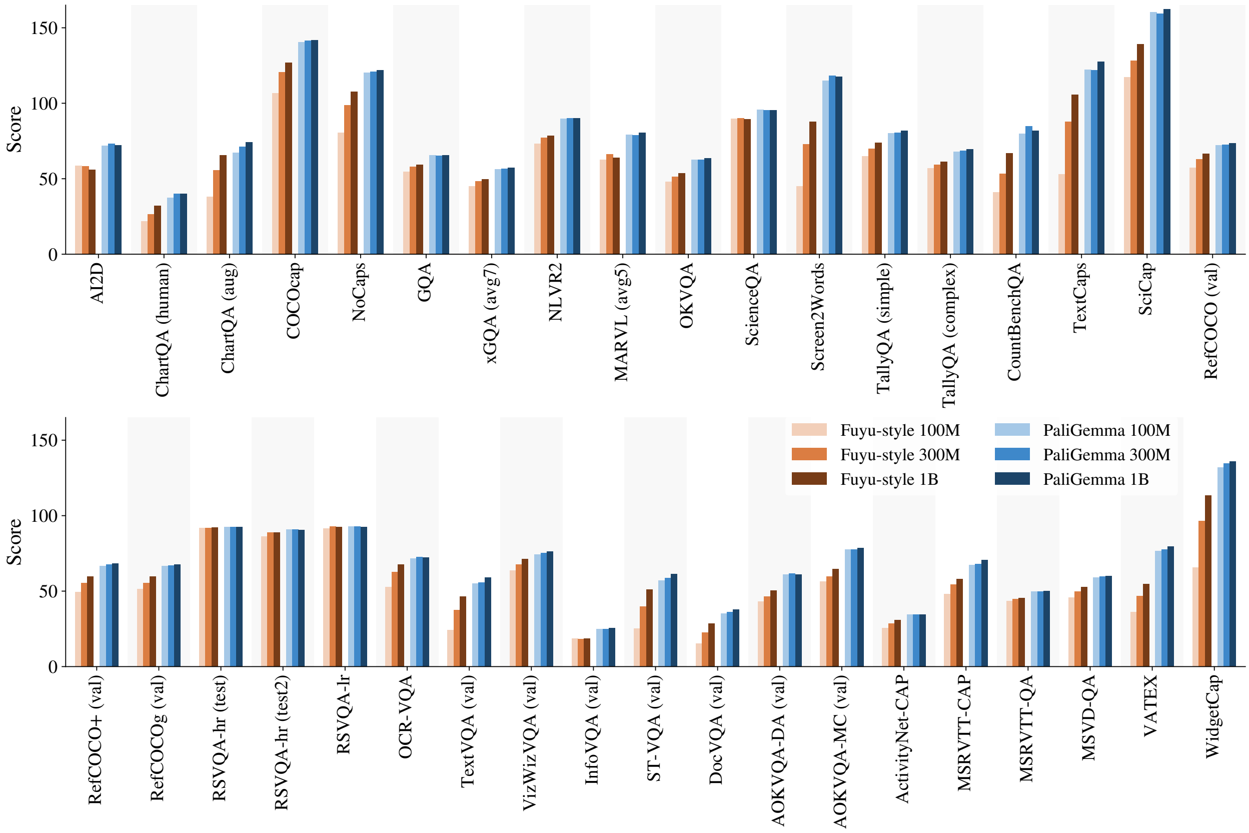


Figure 20 | Per task results when using an image encoder vs not using one (fuyu-style). Discussion in Section 5.6.

K.5. Resolution or sequence length?

See Figure 21.

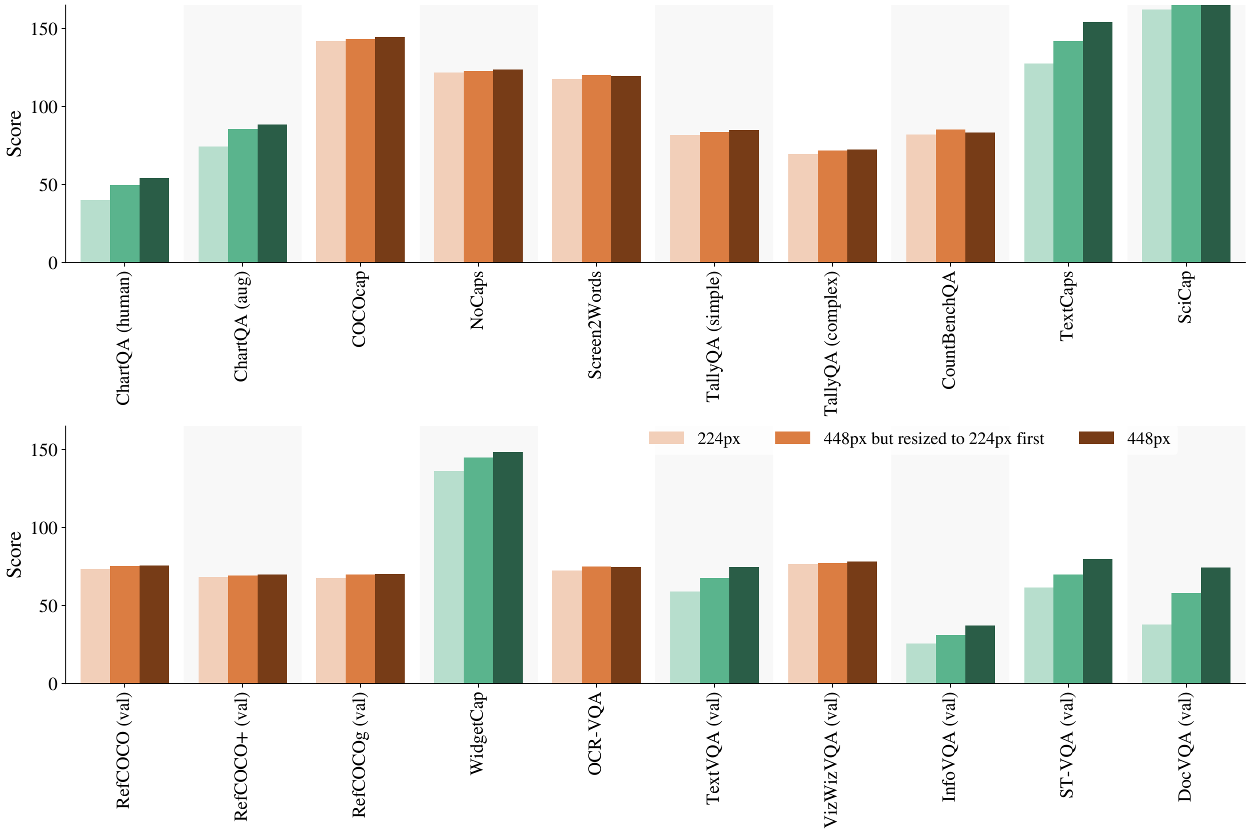


Figure 21 | Per task results when the 448px model's input was first resized to 224. Orange are resolution-insensitive tasks, while green are resolution-sensitive tasks. Discussion in Section 5.7.1.

K.6. Resolution-specific checkpoints and windowing

See Figure 22.

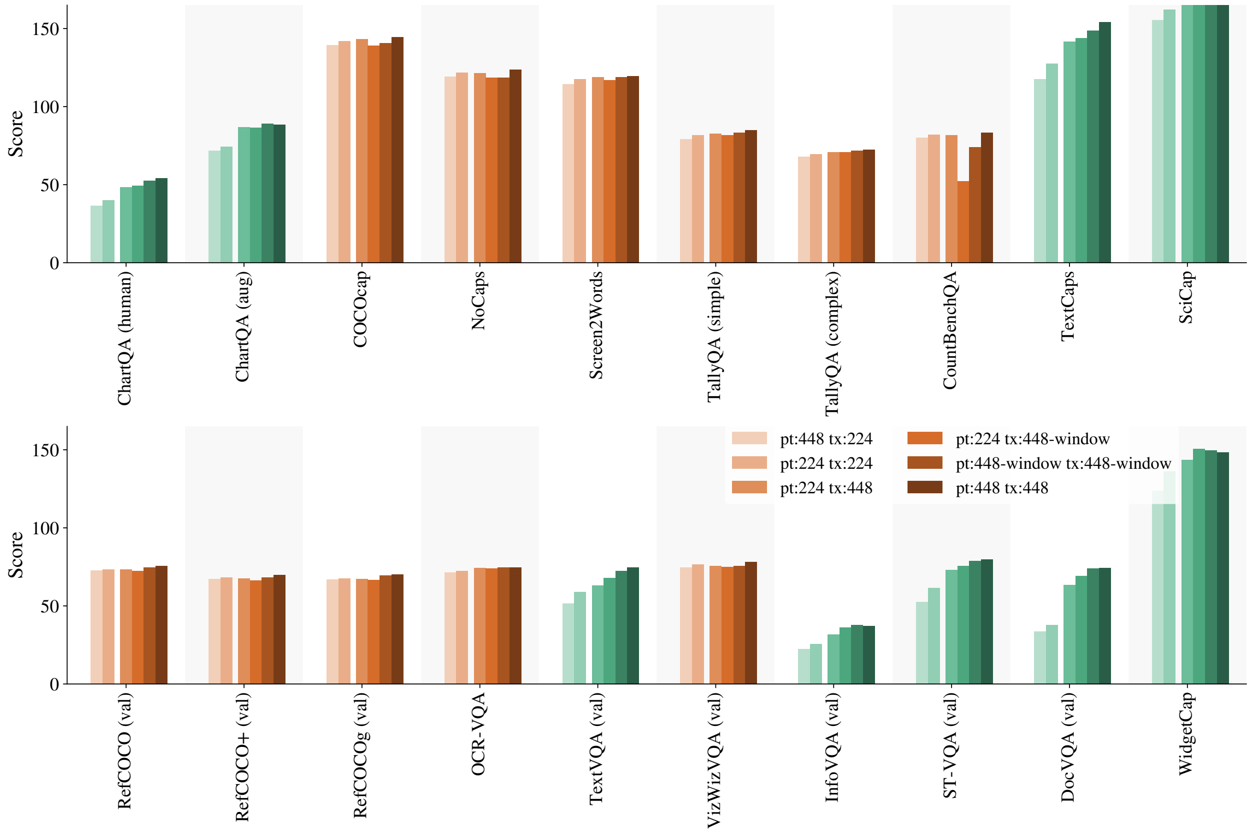


Figure 22 | Per task results when the 448px model's input was first resized to 224. Orange are resolution-insensitive tasks, while green are resolution-sensitive tasks. Discussion in Sections 5.7.2 and 5.7.3.

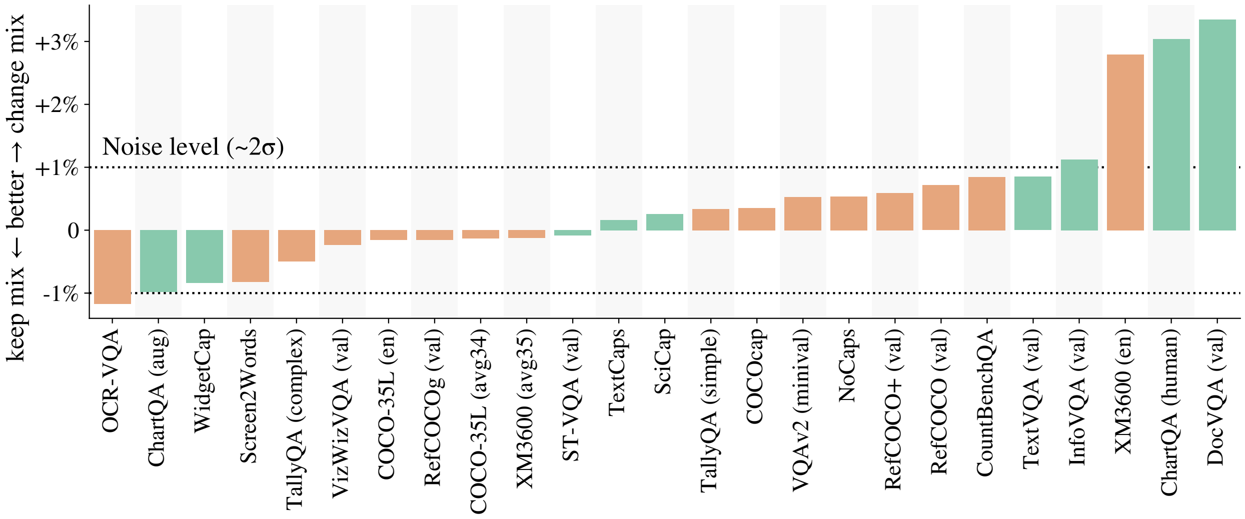


Figure 23 | Per task change in result when performing Stage2 with the exact same pretraining task-mixture as Stage1. Most changes in result are below 2 standard-deviations of re-runs. Discussion in Section 5.7.4.

K.7. Stage2 mixture re-weighting

See Figure 23.

K.8. Transfer with simple hyper parameters

See Figure 24.

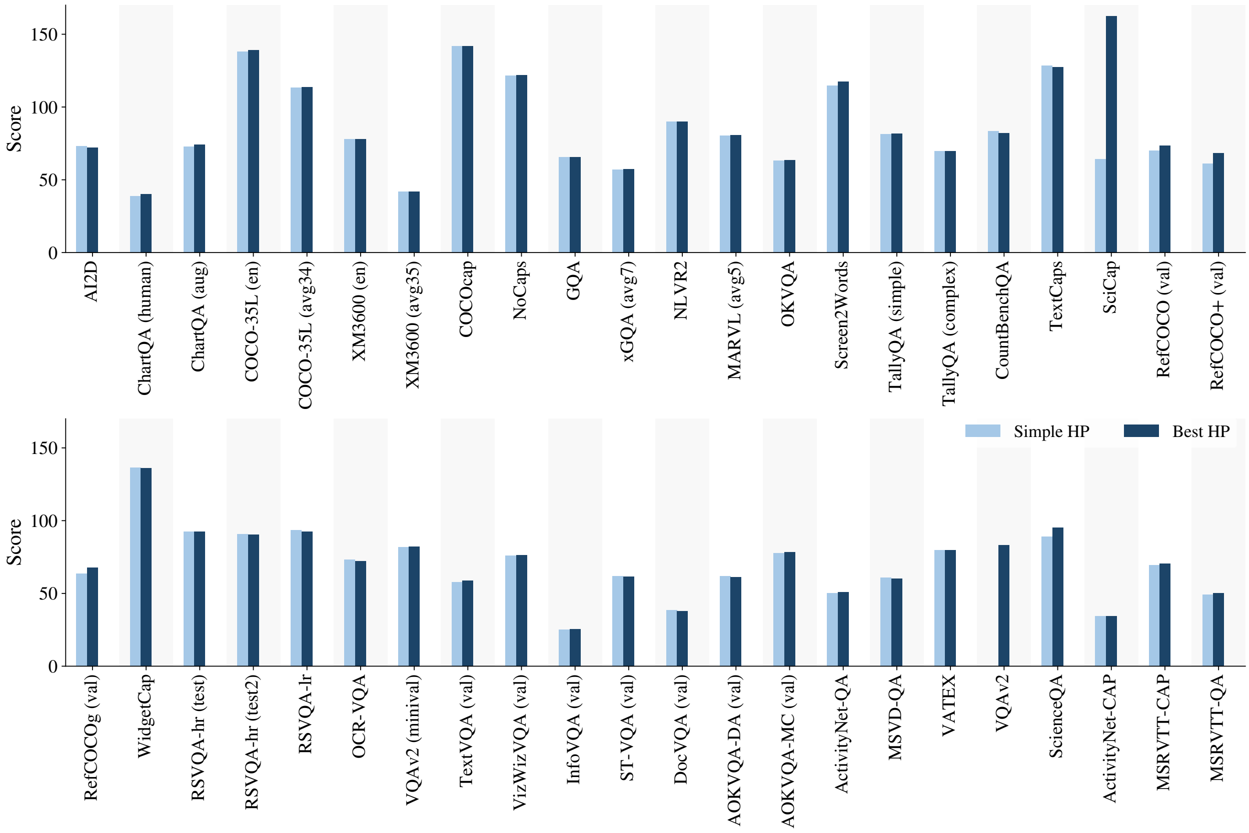


Figure 24 | Per task results when transferring with unique and simple hyper parameter setting for all tasks. For the majority of tasks it works good out of the box.

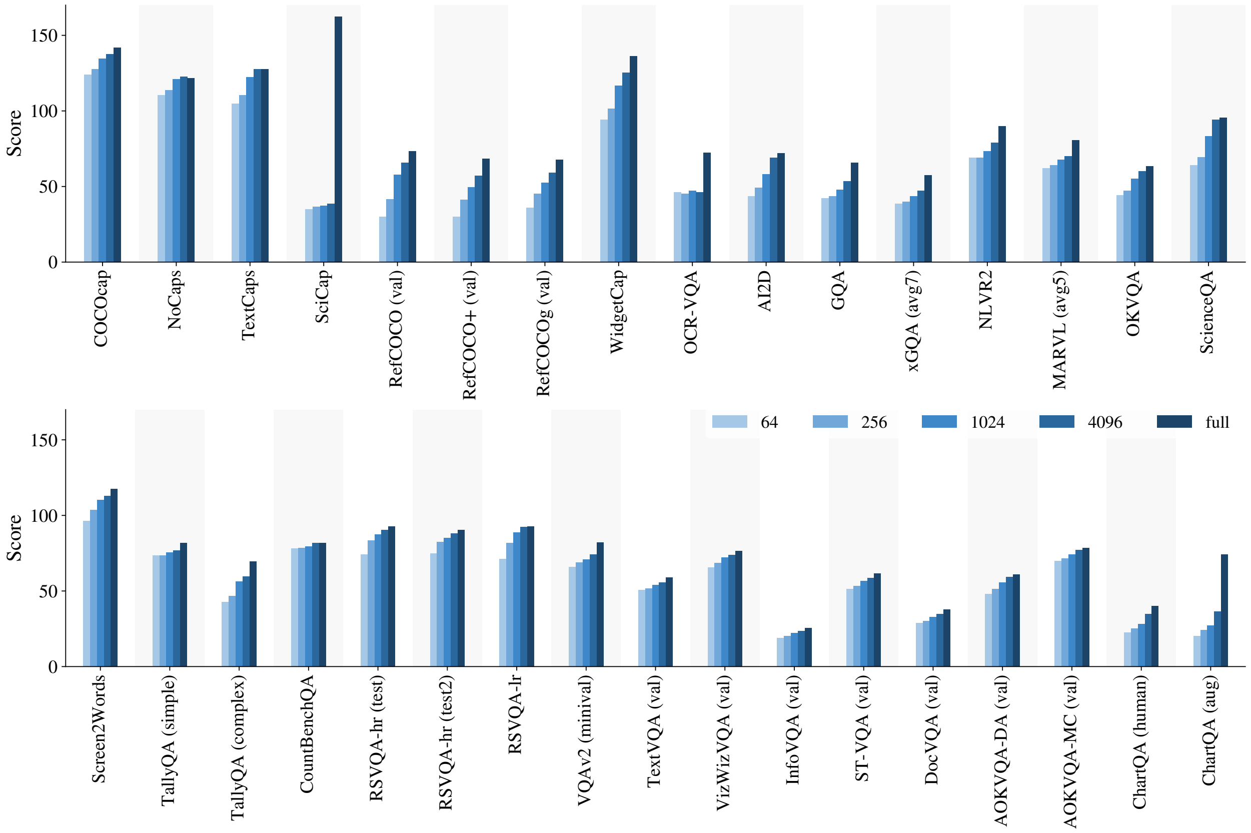


Figure 25 | Per task results when transferring with limited number of examples. We report the max of the values obtained by sweeping learning rates, epochs and batch size.

K.9. Transfer with limited examples

See Figure 25.