arXiv:2308.01390v2 [cs.CV] 7 Aug 2023

OpenFlamingo: An Open-Source Framework for Training Large Autoregressive Vision-Language Models

Anas Awadalla[*1] Irena Gao[*2] Josh Gardner[1] Jack Hessel[3] Yusuf Hanafy[1]
Wanrong Zhu[5] Kalyani Marathe[1] Yonatan Bitton[6] Samir Gadre[7]
Shiori Sagawa[2] Jenia Jitsev[4,9] Simon Kornblith[8] Pang Wei Koh[1,8]
Gabriel Ilharco[1] Mitchell Wortsman[1] Ludwig Schmidt[1,3,4]

Abstract

We introduce OpenFlamingo, a family of autoregressive vision-language models ranging from 3B to 9B parameters. OpenFlamingo is an ongoing effort to produce an open-source replication of DeepMind's Flamingo models [3]. On seven vision-language datasets, OpenFlamingo models average between 80 - 89% of corresponding Flamingo performance. This technical report describes our models, training data, hyperparameters, and evaluation suite. We share our models and code at https://github.com/mlfoundations/open_flamingo.

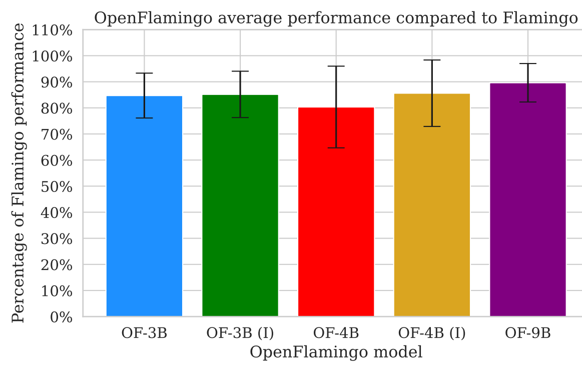

Figure 1: OpenFlamingo performance as a fraction of corresponding Flamingo performance, averaged across evaluation settings (7 datasets × 5 options for number of in-context examples). Demonstrations are chosen using RICES (Retrieval-based In-Context Example Selection). More details regarding selecting demonstrations can be found in Section 3.4. We compare OpenFlamingo-3B and -4B models to Flamingo-3B, and OpenFlamingo-9B to Flamingo-9B. Error bars are standard deviations over settings. "OF-3B (I)" refers to OpenFlamingo-3B (Instruct), the 3B model trained with a language-instruction-tuned backbone.

1 Introduction

A popular format for vision and language models is (image, text) → text, i.e., models take as input an image and some text, and produce text as output, e.g., BLIP-2 [22].The flexible format directly supports tasks like image classification and visual question answering (VQA).

However, assuming a single image as input is limiting: autoregressive vision-language models enable new capabilities by instead mapping an arbitrarily interleaved *sequence* of images and text to textual outputs. This interface provides important flexibility: the input sequence can include demonstrations for a new task, enabling few-shot, in-context learning [3] or multi-round multimodal chatbot interactions. Evaluations suggest that autoregressive vision-language models can be performant foundation models [5]: models like Flamingo [3], CM3 [1], Kosmos-1 [12], PALM-E [8], and multimodal GPT-4 [28] generalize well across diverse vision-language tasks.

[*]Equal contribution. [1]University of Washington [2]Stanford University [3]Allen Institute for AI [4]LAION [5]University of California Santa Barbara [6]Hebrew University [7]Columbia University [8]Google DeepMind [9]Juelich Supercomputing Center, Research Center Juelich. Correspondence to <anasa2@cs.washington.edu, irena@cs.stanford.edu, schmidt@cs.washington.edu>.

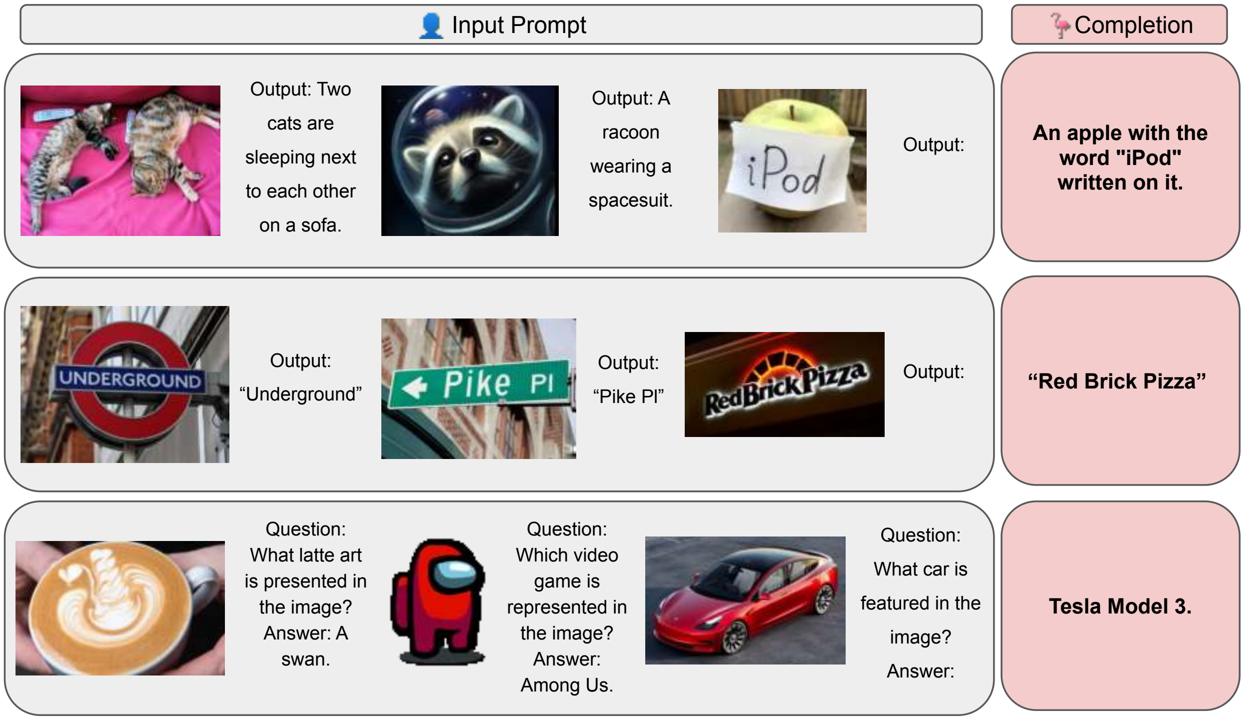

Figure 2: OpenFlamingo-9B (pictured) can process interleaved image-and-text sequences. This interface allows OpenFlamingo to learn many vision-language tasks through in-context demonstrations.

Unfortunately, these autoregressive vision-language models are closed-source, and their weights, training data, code, and hyperparameters are proprietary. This limits the academic community's ability to conduct research on autoregressive vision-language models, e.g., to understand how web-scraped image-text data affects models' performance and safety. Open-source alternatives, such as LLaVA [25], LLaMA-Adapter [41], BLIP-2 [23], and mPLUG-Owl [39], only take in single images, and they often directly train on curated datasets like COCO [24] rather than web data.

In this technical report, we document our experiences building an open-source reproduction of the Flamingo models [3]. Following Flamingo, we augment the layers of pretrained, frozen language models so that they cross attend to the outputs of a frozen vision encoder while predicting the next token. The cross-modal module is trained on web-scraped image-text sequences, in our case, two open source datasets: LAION-2B [32] and Multimodal C4 [45]. Our stack is built using publicly available components, including CLIP as a vision encoder [30] and open-source language models as decoders [27, 35].

We call the resulting family of five models OpenFlamingo. These models range from 3B to 9B parameters, with both standard and instruction-tuned [37] language model backbones. When averaging performance across 7 evaluation datasets, OpenFlamingo-3B and -9B models attain 85% and 89% of their corresponding Flamingo models respectively (Figure 1). Models and code are open-sourced at https://github.com/mlfoundations/open_flamingo.

2 Related work

Generative vision-language models output text conditioned on an image-text sequence. While many such architectures, such as BLIP-

Table 1: Architecture details of the OpenFlamingo models. All five models use a CLIP ViT-L/14 vision encoder [30]. A cross-attention interval of 4 means that a cross-attention module is inserted every 4th language model layer. Note that OpenFlamingo models labeled (Instruct) use language models that were finetuned on language-only tasks; we have not instruction-tuned OpenFlamingo models on vision-language tasks.

Model	**Language model**	**Cross-attention interval**	`<image>` **and** `<|endofchunk|>`
OpenFlamingo-3B	MPT-1B [27]	1	Trainable
OpenFlamingo-3B (Instruct)	MPT-1B (Instruct) [27]	1	Trainable
OpenFlamingo-4B	RedPajama-3B [35]	2	Frozen
OpenFlamingo-4B (Instruct)	RedPajama-3B (Instruct) [35]	2	Frozen
OpenFlamingo-9B	MPT-7B [27]	4	Trainable

2 and LLaVa, can incorporate only one image in their context [6, 16, 22, 25, 39, 41], autoregressive vision-language models accept interleaved image-text sequences, enabling in-context learning.

We chose to replicate Flamingo because of its strong in-context learning abilities. Aggregated across evaluation sets, Flamingo models see steady performance improvements up to 32 in-context examples [3]. This is in contrast with other autoregressive vision-language models, for example Kosmos-1 [12]; on captioning tasks COCO [24] and Flickr-30K [29], Kosmos-1 shows performance improvements up to 4 in-context examples, but performance degrades when using 8 in-context examples.

Open-source image-text datasets. Proprietary autoregressive vision-language models are typically trained on closed-source datasets [1, 3, 8, 12]. For example, Flamingo relies on image-text pairs from the ALIGN dataset [14] and interleaved image-text sequences from the M3W dataset [3]; both are unavailable to the public. Recent efforts to replicate these web-scraped datasets include LAION-2B, a dataset of image-text pairs, and Multimodal C4 [45] and OBELISC [18], datasets of image-text sequences. We use LAION-2B and Multimodal C4 for training OpenFlamingo models. Laurençon et al. [18] also train 9B and 80B Flamingo-style models; their models differ in the choice of pretraining dataset (OBELISC instead of Multimodal C4) and language model (LLaMA-9B [41] instead of the MPT and RedPajama-3B models [27, 35]).

3 Approach

3.1 Architecture

We match the Flamingo architecture [3]. Given an interleaved sequence of images with text tokens, OpenFlamingo models predict the next text token conditioned on all previous text tokens and the last preceding image. Text tokens attend to their corresponding images via *dense cross-attention modules*, which we attach to the layers of a frozen, autoregressive language model. To embed images, we extract patch features from a frozen vision encoder and pass these through a trainable Perceiver resampler [13].

As a preprocessing step, we first mark the locations of images in the text sequence with `<image>` tokens. We also insert `<|endofchunk|>` tokens after the text tokens following an image; *e.g.* the sequence x Hello world, where x is an image, would be preprocessed into `<image> Hello world <|endofchunk|>` .

Unlike Flamingo, we do not support video inputs at this time. We leave this for future work.

Table 1 describes the five OpenFlamingo models based on their language model and density

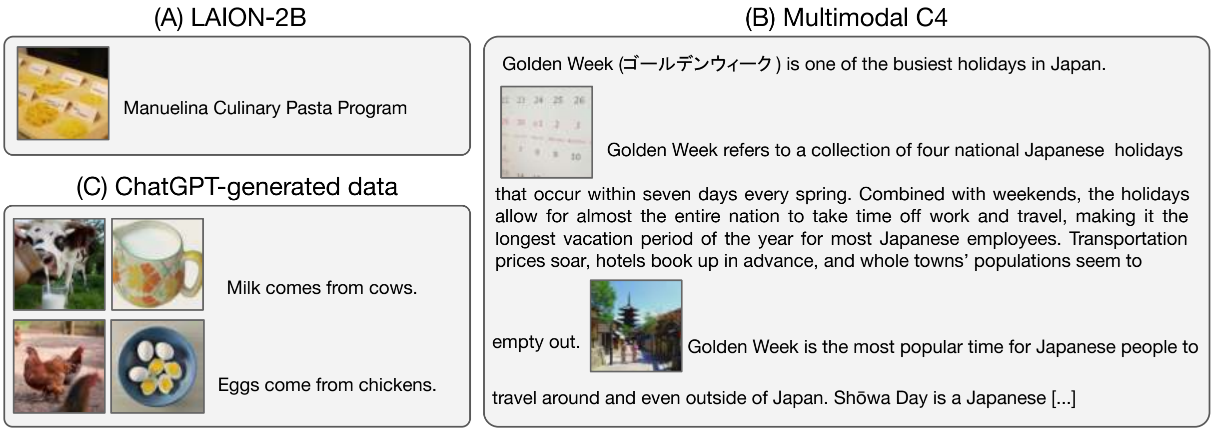

Figure 3: Samples from (A) LAION-2B [32], (B) Multimodal C4 [45], and (C) ChatGPT-generated data.

Table 2: Statistics for training datasets. "ChatGPT" stands for the ChatGPT-generated sequences. The median numbers of images and tokens per sequence were calculated using a random sample of 1,000 sequences.

Dataset	Median images per sequence	Median tokens per sequence
LAION-2B	1	17
MMC4	2	256
ChatGPT	3	56

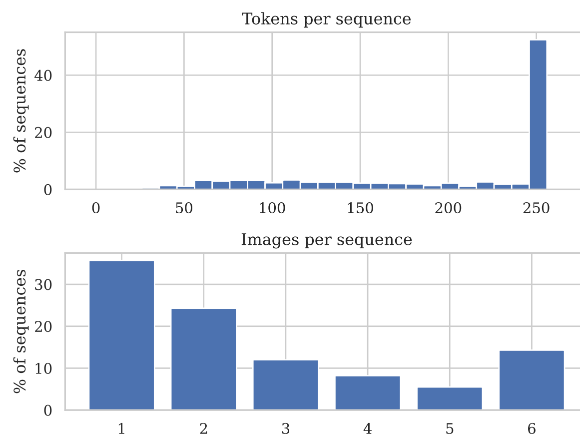

Figure 4: Histograms of the number of text tokens and images per MMC4 sequence, based on a sample of 1,000 sequences. Sequences are long with few images.

of cross-attention layers; all models use CLIP ViT-L/14 [30] as a vision encoder. In most cases, the `<image>` and `<|endofchunk|>` embeddings are trainable, while other text embeddings are frozen. For the OpenFlamingo-4B models, all embeddings are frozen, including the randomly initialized `<image>` and `<|endofchunk|>` embeddings. This was due to complications with gradient masking when using Fully Sharded Data Parallel (§3.3).

3.2 Training data

We train our models on a mixture of image-text pairs and interleaved image-text sequences. During training, we sample dataset shards with replacement using the WebDataset format [34].

LAION-2B [32]. When training Flamingo, Alayrac et al. [3] use ALIGN [14], a closed-source dataset of over 1B single images paired with short alt-text captions. To train OpenFlamingo, we replace ALIGN with LAION-2B, an open-source web-scraped dataset consisting of 2B image-text pairs (Figure 3A). We use part of the English subset and truncate captions to 32 tokens. All image-text pairs in LAION-2B have a cosine similarity of at least 0.28 according to CLIP ViT-B/32.

Multimodal C4 [45]. In addition to image-text pairs, Alayrac et al. [3] train Flamingo using M3W, an internal web-scraped dataset of 43M interleaved image-text sequences. We replace M3W with Multimodal C4 (MMC4), an open-source dataset of 101M interleaved samples (Figure 3B). Unlike M3W or OBELISC [18], which directly parse HTML documents to extract multimodal sequences, MMC4 uses CLIP to soft align images with sentences in a document. To ensure data quality, we exclude images if their cosine similarity with the subsequent text falls below 0.24, according to CLIP ViT-L/14. Sequences contain between 1 and 6 images (median 2). To encourage learning from sequences with multiple images, we reject single-image sequences with probability 0.5. The resulting distribution is shown in Figure 4. Additional notes on MMC4 filtering are in Appendix B.

Synthetic data. For the OpenFlamingo-4B models, we also experimented with training on ChatGPT-generated synthetic data (Figure 3C) These 417K image-text sequences were generated by prompting ChatGPT to generate a sequence of interleaved text and image alt-texts (in place of images). The alt-texts are used to retrieve a corresponding images from LAION-5B. Additional details of the prompting and data construction process are described in Appendix C. The median number of images per sequence is higher than in MMC4, while the median number of text tokens is lower (Table 2). We release these sequences through the OpenFlamingo repository.

3.3 Training details

OpenFlamingo models were trained for 60M interleaved (MMC4) examples[1] and 120M LAION-2B examples. All models are trained using the next-token prediction objective and optimized with AdamW. The learning rate is linearly increased at the beginning of training, and then held constant at 1e-4 throughout training. We apply weight decay of 0.1 on the dense cross attention layers. The batch size for LAION-2B is twice the batch size of the interleaved dataset (MMC4, optionally with ChatGPT-generated sequences), and the loss weights are set to Flamingo defaults of 1 and 0.2 for MMC4 and LAION-2B respectively. We accumulate gradients over both datasets between optimizer steps.

[1]OpenFlamingo-4B models use both MMC4 and ChatGPT-generated data as interleaved sequences; 60M interleaved examples translates to approximately 240K ChatGPT-generated sequences and 59.8M MMC4 sequences. Other models train on 60M MMC4 examples.

Table 3: Training used either DistributedDataParallel (DDP) or FullyShardedDataParallel (FSDP) [43].

Model	**GPU type**	**Sharding strategy**	**Precision**
OF-3B	A100-80GB	DDP	fp32
OF-3B (I)	A100-40GB	DDP	fp32
OF-4B	A100-40GB	FSDP	fp32
OF-4B (I)	A100-40GB	FSDP	fp32
OF-9B	A100-80GB	DDP	amp_bf16

Distributed training. We train all models using 64 GPUs distributed across 8 nodes on Stabilty AI's cluster (Table 3). OpenFlamingo-4B models were trained using model sharding with Fully Sharded Data Parallel [43]; other models were trained using only data parallel.

Loss curves. Figure 5 tracks LAION-2B and MMC4 loss over the course of training. After an initial improvement, MMC4 loss decreases very slowly. We speculate that, since MMC4 sequences tend to include long paragraphs between images (Figure 2), most text tokens can be generated without referencing the image. Thus, the loss may be dominated by whether the frozen language model can fit unrelated paragraphs of text.

3.4 Evaluation method

We evaluate OpenFlamingo on seven vision-language datasets including captioning (COCO [7], Flickr-30K [40]), visual question answer-

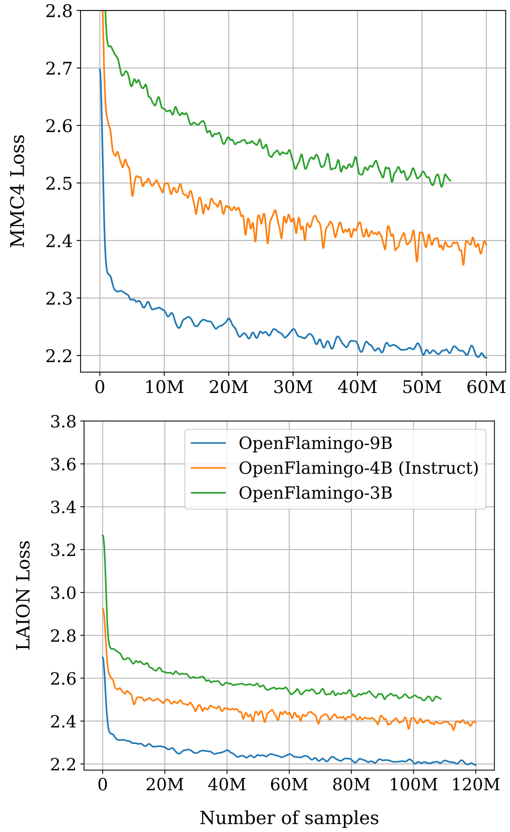

Figure 5: MMC4 and LAION-2B language modeling loss throughout training. Curves shown with Gaussian smoothing with window size 100.

ing (VQAv2 [2], OK-VQA [26], TextVQA [33], VizWiz [11]), and rank classification (HatefulMemes [15]). For each dataset, we measure performance at 0, 4, 8, 16, and 32 in-context examples. Evaluation was done in automatic mixed precision, with linear layers computed in bfloat16.

Selecting in-context examples. For each evaluation example, we sample in-context examples from the training split uniformly at random. Additionally, in Appendix A.2, we include evaluations of OpenFlamingo using Retrieval-based In-Context Example Selection (RICES) [38].

Evaluation subsets. We evaluate on the dataset splits used by Alayrac et al. [3]. We run each evaluation across three seeds, where the randomness is over selected in-context demonstrations, and average the results to obtain our final scores.

Prompts. For captioning tasks, we format demonstrations as <image> Output: [caption], replacing [caption] with the ground-truth caption. For VQA, we format examples as <image> Question: [question] Short answer: [answer]. For HatefulMemes, we prompt the model with <image> is an image with: '[text]' written on it. Is it hateful? Answer: [answer].

Following Alayrac et al. [3], we prompt the model with two in-context examples during zero-shot evaluations, removing their images, and for classification tasks, we implement prompt ensembling by averaging logits across 6 permutations of the in-context examples.

Decoding parameters. We evaluate captioning and VQA using beam search with 3 beams, stopping generation at 20 tokens for captioning, 5 tokens for VQA, or whenever the model produces an <|endofchunk|> token. For HatefulMemes, we compute the log-likelihood of completions "yes" and "no" and answer with the most likely completion.

Metrics. For captioning, we use CIDEr score [36]. For VQA, we report VQA accuracy, *i.e.*, exact match accuracy over a set of ground truth answers [2]. For HatefulMemes, we compute AUC ROC.

4 Results

In Table 4, we compare OpenFlamingo and Flamingo models across 0, 4, and 32 in-context examples. On average, OpenFlamingo-3B, -3B (Instruct), -4B (Instruct), and -9B attain more than 86% of the performance of their corresponding Flamingo models (Figure 1).

In the 0- and 4-shot regimes, OpenFlamingo models approach or match Flamingo performances on

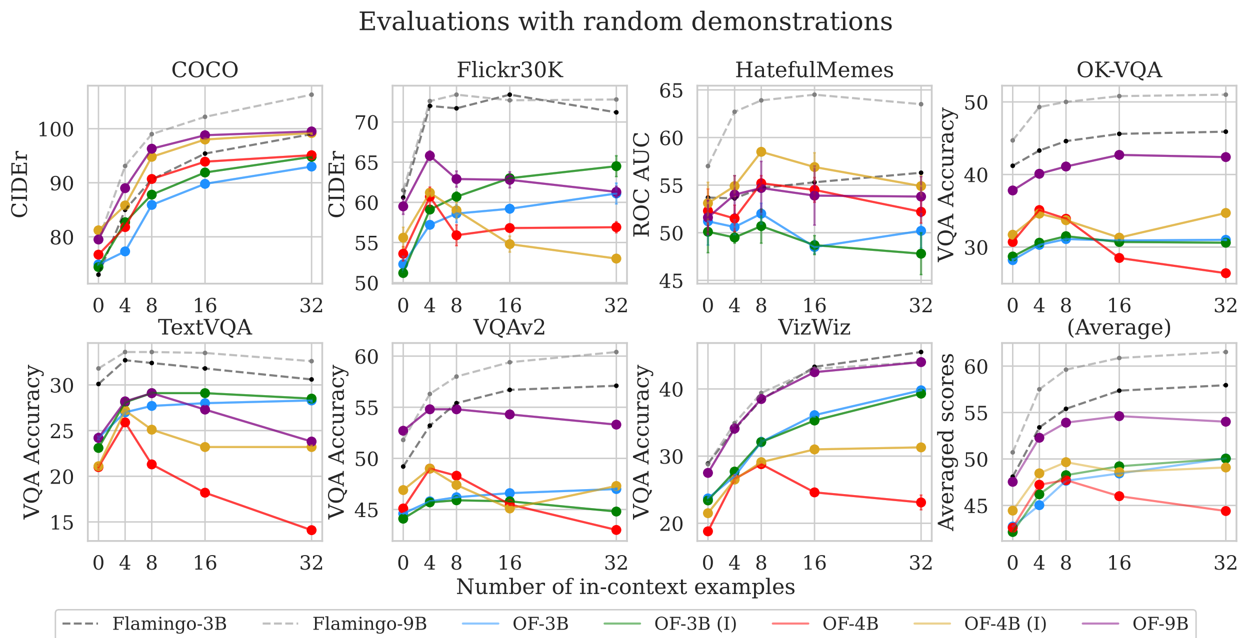

Figure 6: Evaluation results per dataset across 0, 4, 8, 16, and 32 in-context examples. Each point is the average across three evaluation runs, where the randomness is over choice of in-context demonstrations. Error bars are standard deviations over random seeds. Results are reported in tabular form in Table 11.

several datasets. For example, OpenFlamingo-9B improves upon Flamingo-9B's 0-shot performance on VQAv2 (51.8% → 52.7% VQA accuracy) and COCO (79.4 → 79.5 CIDEr), and OpenFlamingo-9B approaches Flamingo-9B's 0-shot performance on Flickr-30K and VizWiz. Moreover, OpenFlamingo-9B approaches the 4-shot performance of Flamingo-9B on COCO, VQAv2, and VizWiz.

However, on OK-VQA and TextVQA, OpenFlamingo models are notably weaker than their Flamingo counterparts: OpenFlamingo-9B underperforms Flamingo-9B in 0-shot evaluations by 6.9 percentage points on OK-VQA and 7.8 percentage points on TextVQA. OpenFlamingo-3B also underperforms Flamingo-3B by 4.6 percentage points in 0-shot VQAv2 accuracy. The reason for generally low VQA performance is unclear, although discussions in §5.2 may be related.

Extrapolating to more in-context examples. In Figure 6, we plot performance as a function of the number of in-context examples. We observe that the OpenFlamingo-3B and -9B models generally improve with the number of in-context examples. However, the rate of improvement is lower than the Flamingo models: in the bottom right corner of Figure 6, we observe that gaps between OpenFlamingo-9B and Flamingo-9B widen with the number of in-context examples. We speculate that this behavior may stem from the quality of our pre-training data, which mostly consists of sequences with few images (Table 2). In contrast with the -3B and -9B models, which generally improve with more in-context examples, the OpenFlamingo-4B models unexpectedly degrade in performance after 4 or 8 shots. The 4B models use RedPajama language models [35] instead of MPT backbones [27]; they also use frozen `<image>` and `<|endofchunk|>` embeddings. We investigate the effect of the latter in §5.1.

Trends by model size. OpenFlamingo-9B generally outperforms smaller models, except on

Benchmark	Shots	Fl-3B	Fl-9B	OF-3B	OF-3B (I)	OF-4B	OF-4B (I)	OF-9B
	0	73.0	79.4	74.9 (0.2)	74.4 (0.6)	76.7 (0.2)	**81.2 (0.3)**	79.5 (0.2)
	4	85.0	**93.1**	77.3 (0.3)	82.7 (0.7)	81.8 (0.4)	85.8 (0.5)	89.0 (0.3)
COCO [7]	32	99.0	**106.3**	93.0 (0.6)	94.8 (0.3)	95.1 (0.3)	99.2 (0.3)	99.5 (0.1)
	0	60.6	**61.5**	52.3 (1.0)	51.2 (0.2)	53.6 (0.9)	55.6 (1.3)	59.5 (1.0)
	4	72.0	**72.6**	57.2 (0.4)	59.1 (0.3)	60.7 (1.2)	61.2 (0.5)	65.8 (0.6)
Flickr-30K [40]	32	71.2	**72.8**	61.1 (1.3)	64.5 (1.3)	56.9 (0.7)	53.0 (0.5)	61.3 (0.7)
	0	49.2	51.8	44.6 (0.0)	44.1 (0.1)	45.1 (0.1)	46.9 (0.0)	**52.7 (0.2)**
	4	53.2	**56.3**	45.8 (0.0)	45.7 (0.1)	49.0 (0.0)	49.0 (0.0)	54.8 (0.0)
VQAv2 [2]	32	57.1	**60.4**	47.0 (0.1)	44.8 (0.1)	43.0 (0.2)	47.3 (0.0)	53.3 (0.1)
	0	41.2	**44.7**	28.2 (0.2)	28.7 (0.1)	30.7 (0.1)	31.7 (0.1)	37.8 (0.2)
	4	43.3	**49.3**	30.3 (0.5)	30.6 (0.2)	35.1 (0.0)	34.6 (0.0)	40.1 (0.1)
OK-VQA [26]	32	45.9	**51.0**	31.0 (0.1)	30.6 (0.1)	26.4 (0.2)	34.7 (0.3)	42.4 (0.0)
	0	30.1	**31.8**	24.2 (0.2)	23.1 (0.2)	21.0 (0.3)	21.1 (0.4)	24.2 (0.5)
	4	32.7	**33.6**	27.0 (0.3)	28.1 (0.4)	25.9 (0.0)	27.2 (0.3)	28.2 (0.4)
TextVQA [33]	32	30.6	**32.6**	28.3 (0.2)	28.5 (0.1)	14.1 (0.2)	23.2 (0.2)	23.8 (0.2)
	0	**28.9**	28.8	23.7 (0.5)	23.4 (0.3)	18.8 (0.1)	21.5 (0.2)	27.5 (0.2)
	4	34.0	**34.9**	27.0 (0.3)	27.7 (0.1)	26.6 (0.5)	26.5 (0.4)	34.1 (0.7)
VizWiz [11]	32	**45.5**	44.0	39.8 (0.1)	39.3 (0.4)	23.1 (1.1)	31.3 (0.2)	44.0 (0.5)
	0	53.7	**57.0**	51.2 (2.5)	50.1 (2.2)	52.3 (2.3)	53.1 (2.2)	51.6 (1.8)
	4	53.6	**62.7**	50.6 (0.8)	49.5 (0.6)	51.5 (1.4)	54.9 (1.1)	54.0 (2.0)
HatefulMemes [15]	32	56.3	**63.5**	50.2 (1.8)	47.8 (2.2)	52.2 (1.2)	54.9 (1.1)	53.8 (2.1)

Table 4: Evaluation results across seven vision-language datasets using 0, 4, and 32 in-context examples. "OF-3B (I)" refers to OpenFlamingo-3B (Instruct), the 3B model trained with a language-instruction-tuned backbone, while "Fl-3B" refers to Flamingo-3B. Flamingo results taken from Alayrac et al. [3]. The highest number in each row is bolded. Full results (including 8- and 16-shot performance) are in Table 11.

HatefulMemes and for large numbers of in-context examples on Flickr-30K and TextVQA. However, OpenFlamingo-4B models often underperform the smaller 3B models, including on Flickr-30K, HatefulMemes, TextVQA, and VizWiz.

Effect of language instruction-tuning. We train two OpenFlamingo models at each of the 3B and 4B scales: one model using a base language model, and one with an instruction-tuned variant of the same language model. In the lower right corner of Figure 6, we observe that the instruction-tuned variants of MPT-1B and RedPajama-3B on average outperform the base models. The difference is starkest for RedPajama-3B. Transfer of language instruction tuning to vision-language tasks was previously reported in Huang et al. [12], Li et al. [23].

Comparison to fine-tuned state-of-the-art. Figure 7 plots each model's performance relative to fine-tuned state-of-the-art performance, as listed on Papers With Code on June 19, 2023. OpenFlamingo-9B averages more than 62% of fine-tuned state-of-the-art performance with 32 RICES-selected in-context examples, compared to 72% achieved by Flamingo-9B. For more details on the fine-tuned SoTAs, see Appendix A.1.

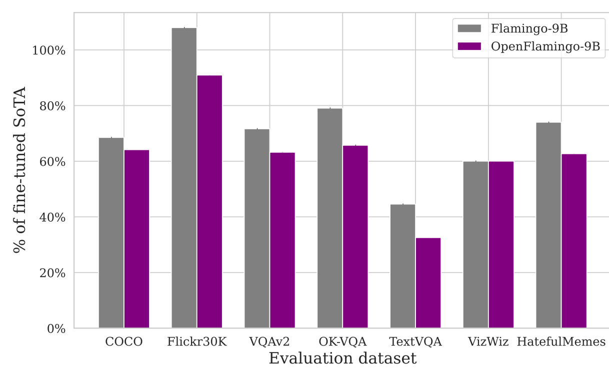

Figure 7: OpenFlamingo-9B and Flamingo-9B performance relative to fine-tuned SoTA performance.

		0-shot	4-shot	8-shot
COCO	trainable	46.5	58.6	61.2
	frozen	41.9 (−4.6)	54.5 (−4.1)	57.4 (−3.8)
VQAv2	trainable	17.6	23.2	28.7
	frozen	5.5 (−12.1)	8.4 (−14.8)	18.8 (−9.9)

Table 5: COCO and VQAv2 validation performance when using trainable `<image>` and `<|endofchunk|>` embeddings compared to frozen, randomly initialized embeddings. The model used in this experiment is based on CLIP ViT-L/14 and OPT 125M, with cross-attention every layer, and trained on 20M interleaved samples, including ChatGPT-sequences.

5 Discussion

5.1 Frozen embeddings

In §4, we observed that OpenFlamingo-4B models underperform their 3B counterparts on most datasets. One notable way the OpenFlamingo-4B models differ from the 3B and 9B models is that their `<image>` and `<|endofchunk|>` embeddings are randomly initialized and frozen, rather than trained.

In Table 5, we investigate the effect of this difference. We train small models using OPT-125M as a language model [42] to 20M interleaved samples (one-third of full training). Freezing the `<image>` and `<|endofchunk|>` embeddings results in a drop of 4.6 CIDEr for 0-shot COCO, and 12.1% accuracy for 0-shot VQAv2. This suggests that frozen `<image>` and `<|endofchunk|>` embeddings may impact downstream trends.

5.2 VQAv2 validation trends

During development, we used the VQAv2 validation set as a temperature check for visual question answering capabilities. In this section, we discuss trends observed during development.

Training dynamics. To understand how evaluation performance evolves over the course of training, Figure 8 plots validation performance of OpenFlamingo-9B on COCO and VQAv2 throughout training. While COCO performance steadily improves, VQAv2 progress is flatter.

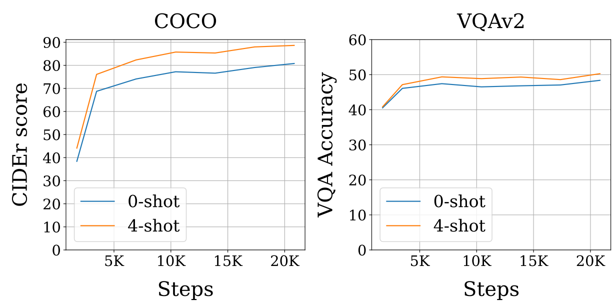

Figure 8: Validation split performance for OpenFlamingo-9B across training: while COCO CIDEr improves throughout training, VQAv2 performance is more stagnant.

This matches trends reported by Li et al. [23].

Effect of language model. Although additional training did not dramatically affect VQAv2 performance, changing language model backbones did. Table 7 illustrates this effect on the VQAv2 validation split; notably, switching from OPT-1.3B to MPT-1B (Instruct) added nearly 10 percentage points in 0-shot performance. We hypothesize that the language model has similarly large effects for other VQA tasks.

Common VQA failure modes (Table 6). OpenFlamingo models struggle with counting; on the VQAv2 validation split, OpenFlamingo-9B scores 30.5% on questions with numerical answers, compared to 70.6% on yes / no questions. Additionally, because VQA accuracy uses an exact match criterion for generations, models must answer concisely to score well; OpenFlamingo models are often too verbose. Finally, VQA questions can ask about objects other than the central object in the image; models sometimes answer about the central item instead.

5.3 Applications of OpenFlamingo

Multiple models have already developed on top of OpenFlamingo. Li et al. [20] fine-tuned OpenFlamingo on MIMIC-IT [19], a multi-image/video instruction following dataset, creating Otter, a

Counting	Verbosity	Non-central object
		
Q: How many people are on the sidewalk?	Q: What is this sheep trying to do?	Q: What color are the curtains?
OF-9B: “one”	OF-9B: “it is trying to get”	OF-9B: “green”
Ground truth: {*“4”, “5”*}	Ground truth: {*“get out”, “escape”*}	Ground truth: {*“yellow”, “gold”*}

Table 6: OpenFlamingo-9B errors from the VQAv2 validation split. Common failure modes for OpenFlamingo including counting, giving answers that are too verbose (and thus truncated), and answering about the central object in the image rather than the non-central object in the question.

	VQAv2 validation	
	Shots	
Language model	**0**	**4**
OPT-125M	17.6	23.2
OPT-1.3B	32.8	27.2
MPT-1B (Instruct)	41.9	43.7
MPT-7B	47.4	49.4

Table 7: VQAv2 validation performance at 20M interleaved samples across different language models. Performance largely differs between language models.

multimodal assistant. Gong et al. [10] released Multimodal-GPT, an OpenFlamingo model instruction fine-tuned on both vision and language instruction datasets. We hope the community continues to use OpenFlamingo models.

5.4 Limitations

OpenFlamingo models carry the same risks as their foundational language models. In particular, these models train on web-scraped data, and they have not undergone safety-focused fine-tuning. Models thus may produce unexpected, inappropriate, or inaccurate outputs. We hope to further investigate the safety properties of autoregressive vision-language models like OpenFlamingo.

6 Conclusion

In this technical report, we described OpenFlamingo, a family of five autoregressive vision-language models across the 3B, 4B, and 9B scales. OpenFlamingo remains an active research project, and we continue to work on training and releasing high-quality autoregressive vision-language models. We hope our contribution enables more researchers to train and study such models.

Acknowledgements

We would like to thank Jean-Baptiste Alayrac and Antoine Miech for their advice on reproducing Flamingo. We also thank Rohan Taori, Nicholas Schiefer, Deep Ganguli, Thomas Liao, Tatsunori Hashimoto, and Nicholas Carlini for their help with assessing the safety risks of our first release of OpenFlamingo. Thanks to Stability AI for compute resources.

References

[1] Armen Aghajanyan, Po-Yao (Bernie) Huang, Candace Ross, Vladimir Karpukhin, Hu Xu, Naman Goyal, Dmytro Okhonko, Mandar Joshi, Gargi Ghosh, Mike Lewis, and Luke Zettlemoyer. Cm3: A causal masked multimodal model of the internet. *arXiv preprint arXiv:2201.07520*, 2022.

[2] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Devi Parikh, and Dhruv Batra. Vqa: Visual question

answering. *International Journal of Computer Vision*, 123:4–31, 2015.

[3] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35: 23716–23736, 2022.

[4] Romain Beaumont. Clip retrieval: Easily compute clip embeddings and build a clip retrieval system with them. https://github.com/rom1504/clip-retrieval, 2022.

[5] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[6] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022.

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

[8] Danny Driess, F. Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Ho Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Peter R. Florence. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[9] Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. Datacomp: In search of the next generation of multimodal datasets. *arXiv preprint arXiv:2304.14108*, 2023.

[10] Tao Gong, Chengqi Lyu, Shilong Zhang, Yudong Wang, Miao Zheng, Qianmengke Zhao, Kuikun Liu, Wenwei Zhang, Ping Luo, and Kai Chen. Multimodal-gpt: A vision and language model for dialogue with humans. *arXiv preprint arXiv:2305.04790*, 2023.

[11] Danna Gurari, Qing Li, Abigale Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P. Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3608–3617, 2018.

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

[13] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021.

[14] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

[15] Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. The hateful memes challenge: Detecting hate speech in multimodal memes. *arXiv preprint arXiv:2005.04790*, 2020.

[16] Jing Yu Koh, Ruslan Salakhutdinov, and Daniel Fried. Grounding language models to images for multimodal generation. *arXiv preprint arXiv:2301.13823*, 2023.

[17] Gokul Karthik Kumar and Karthik Nandakumar. Hate-clipper: Multimodal hateful meme classification based on cross-modal interaction of clip features. *arXiv preprint arXiv:2210.05916*, 2022.

[18] Hugo Laurençon, Lucile Saulnier, Léo Tronchon, Stas Bekman, Amanpreet Singh, Anton Lozhkov,

Thomas Wang, Siddharth Karamcheti, Alexander M. Rush, Douwe Kiela, Matthieu Cord, and Victor Sanh. Obelisc: An open web-scale filtered dataset of interleaved image-text documents. *arXiv preprint arXiv:2306.16527*, 2023.

[19] Bo Li, Yuanhan Zhang, Liangyu Chen, Jinghao Wang, Fanyi Pu, Jingkang Yang, C. Li, and Ziwei Liu. Mimic-it: Multi-modal in-context instruction tuning. *arXiv preprint arXiv:2306.05425*, 2023.

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

[21] Chenliang Li, Haiyang Xu, Junfeng Tian, Wei Wang, Ming Yan, Bin Bi, Jiabo Ye, Hehong Chen, Guohai Xu, Zheng da Cao, Ji Zhang, Songfang Huang, Feiran Huang, Jingren Zhou, and Luo Si. mplug: Effective and efficient vision-language learning by cross-modal skip-connections. *arXiv preprint arXiv:2205.12005*, 2022.

[22] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[23] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[25] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

[26] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3190–3199, 2019.

[27] MosaicML. Introducing mpt-7b: A new standard for open-source, commercially usable llms, 2023.

[28] R OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[29] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *IEEE international conference on computer vision*, pages 2641–2649, 2015.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[31] Shibani Santurkar, Yann Dubois, Rohan Taori, Percy Liang, and Tatsunori Hashimoto. Is a caption worth a thousand images? a controlled study for representation learning. *arXiv preprint arXiv:2207.07635*, 2022.

[32] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

[33] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8309–8318, 2019.

[34] Thomas Breuel. WebDataset: A high-performance Python-based I/O system for large (and small) deep learning problems, with strong support for PyTorch. Available at: https://github.com/webdataset/webdataset, 2020.

[35] Together.xyz. Releasing 3b and 7b redpajama-incite family of models including base, instruction-tuned & chat models. https://www.together.xyz/blog/redpajama-models-v1, 2023.

[36] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575, 2014.

[37] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[38] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3081–3089, 2022.

[39] Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*, 2023.

[40] Peter Young, Alice Lai, Micah Hodosh, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

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

[42] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[43] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

[44] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J. Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and vqa. *arXiv preprint arXiv:1909.11059*, 2019.

[45] Wanrong Zhu, Jack Hessel, Anas Awadalla, Samir Yitzhak Gadre, Jesse Dodge, Alex Fang, Youngjae Yu, Ludwig Schmidt, William Yang Wang, and Yejin Choi. Multimodal c4: An open, billion-scale corpus of images interleaved with text. *arXiv preprint arXiv:2304.06939*, 2023.

Table 8: Fine-tuned state-of-the-art numbers used in this report.

Method	Dataset	Score
mPLUG [21]	COCO	155.1
Unified VLP [44]	Flickr-30K	67.4
Pali-17B [6]	VQAv2	84.3
Pali-17B [6]	OK-VQA	64.5
Pali-17B [6]	TextVQA	73.1
Pali-17B [6]	VizWiz	73.3
Hate-CLIPper [17]	HatefulMemes	85.8

A Extended results

Table 11 provides full evaluation results for 0, 4, 8, 16, and 32 in-context examples. For ease of comparison to Flamingo, we calculate each OpenFlamingo model's performance as a fraction of corresponding Flamingo performance in Figure 11.

A.1 Comparison to fine-tuned SoTAs

In Figure 9, we compare OpenFlamingo models to fine-tuned SoTA performances for different numbers of in-context examples. The fine-tuned methods used were pulled from PapersWithCode on 06/19/23 (Table 8).

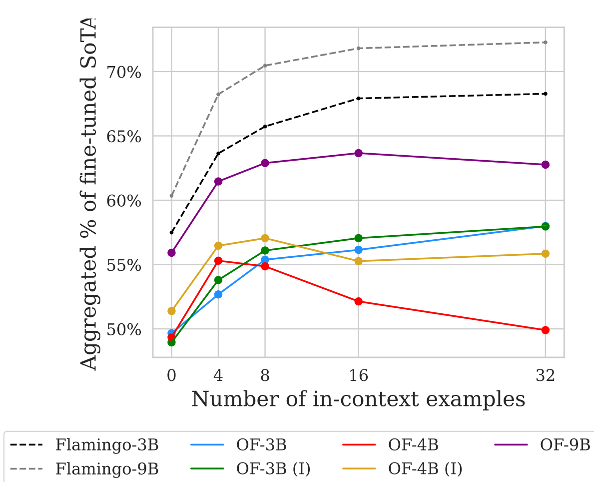

Figure 9: We plot each model's performance relative to fine-tuned state-of-the-art performance, averaged across datasets.

Benchmark	Shots	Random	RICES
COCO	4	89.0	93.1 (+4.1)
	32	99.5	109.0 (+9.5)
Flickr-30K	0	59.5	39.2 (−20.3)
	4	65.8	52.2 (−13.6)
	8	62.9	58.7 (−4.2)
	32	61.3	63.0 (+1.7)
VQAv2	4	54.8	55.1 (+0.3)
	32	53.3	56.8 (+3.5)
OK-VQA	4	40.1	38.3 (−1.8)
	32	42.4	46.3 (+3.9)
TextVQA	4	28.2	34.2 (+6)
	32	23.8	31.1 (+7.3)
VizWiz	4	27.5	41.0 (+13.5)
	32	44.0	46.4 (+2.4)
HatefulMemes	4	54.0	70.1 (+16.1)
	32	53.8	73.6 (+19.8)

Table 9: Using RICES [38] to select in-context examples often outperforms using random demonstrations. Scores in table are for OpenFlamingo-9B.

A.2 Evaluations using RICES

In the main text, we evaluate OpenFlamingo by selecting in-context examples uniformly at random. In this appendix, we include additional evaluation results using Retrieval-based In-Context Example Selection (RICES) [38]. For a given test example, RICES selects the top-k most similar training examples as demonstrations, where similarity is measured by cosine similarity of the images according to the frozen vision encoder (CLIP ViT-L/14). Full results with RICES are listed in Table 12 and illustrated in Figure 10.

In Table 9, we compare OpenFlamingo-9B performance using RICES to performance using randomly selected in-context examples. We observe that RICES significantly boosts performance in most evaluation settings, including by 19.2 ROC AUC using 32 shots on HatefulMemes. However, on Flickr-30K, we observe significant degradations from using RICES: CIDEr degrades by 20.4 in 0-shot evaluations[2] and 13.1 in 4-shot evaluations. We hypothesize that the demonstrations RICES selects in Flickr-30K are more similar to the test example than in other datasets. This leads OpenFlamingo-9B to parrot captions from the in-context examples, including incorrect details. For an example, see Table 10 in Appendix A.

[2] In 0-shot evaluations, RICES is still used to select the two text-only examples used for the prompt (§3.4).

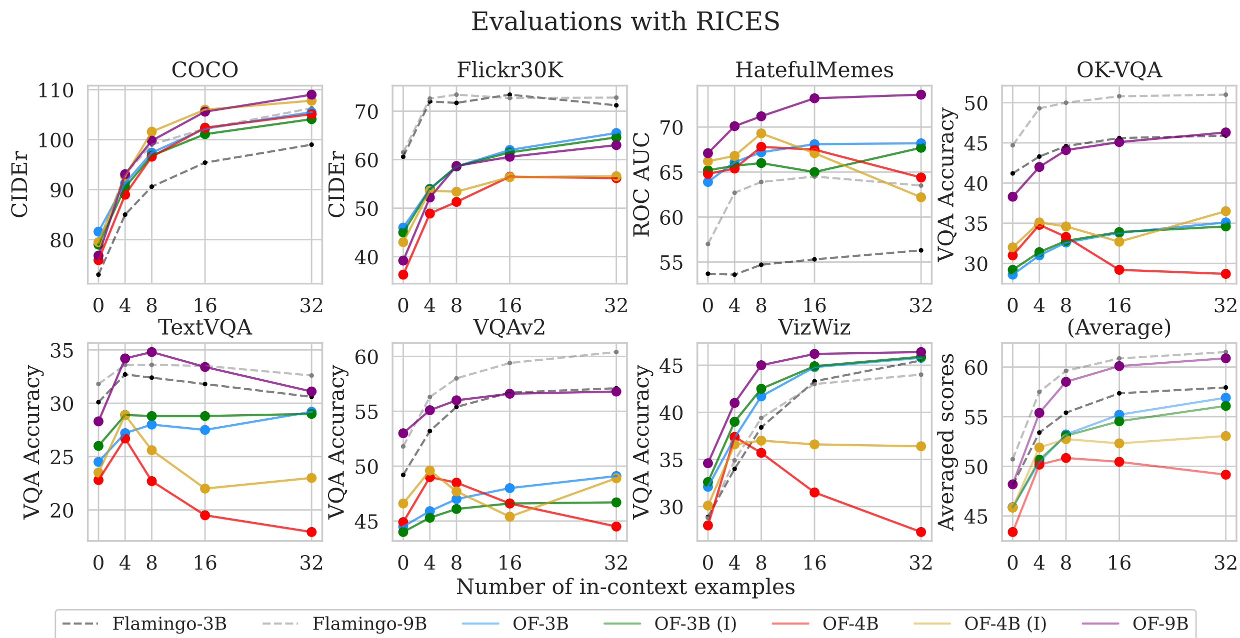

Figure 10: Evaluation results per dataset across 0, 4, 8, 16, and 32 in-context examples. Results are reported in tabular form in Table 12.

B Additional notes on filtering MMC4

When training contrastive vision-language models, filtering image-text pairs by CLIP cosine similarity has proven particularly helpful for improving data quality [31, 9]. We use a similar notion for filtering interleaved sequences in MMC4: if an image and its matched sentence had cosine similarities that fell below a fixed threshold (0.24), according to CLIP ViT-L/14 embeddings, we omitted the image from the sequence, keeping the text. If all images in a sequence are omitted, we discard the sequence entirely. This aims to ensure that images are relevant to the text following it.

However, increasing the image-text similarity threshold has a side effect: it reduces the typical number of images per interleaved sequence. When using similarity 0.32, nearly 58% of a sample of 1,000 MMC4 sequences contain only 1 image per sequence, compared to 38% in Figure 4, which uses a threshold of 0.24. Training with long sequences may be important for producing models that can handle a large amount of in-context examples. Further, we estimate that 88.7% of MMC4 sequences are discarded completely when filtering with threshold 0.32, compared to 42.7% with threshold 0.24.

As future work, we are interested in understanding how to balance length, quality, and dataset size objectives to improve OpenFlamingo models.

C Synthetic data prompt

We provide the prompt used to generate the ChatGPT-generated data (see §3.2) in Table 12. After generating candidate sequences, we query LAION-5B using [4] to infill images. For each unique caption we generate, we attempt to retrieve 10 candidate images from the index using index=laion5B-L-14, aesthetic_score=9, and aesthetic_weight=0.5. After this search, we re-rank the retrieved images using CLIP ViT-L/16@336px and select the image with the highest similarity to interleave.

D Image credits

We include the links to the images we used in Figure 2 in Table 13.

	Random demonstrations	**RICES**
Demos	 A person hanging from a telephone pole near the mountains.	 The brown dog is running through the grass with a yellow toy in its mouth.
	 A trio of male musicians are performing with one playing a guitar and singing into a microphone, another holding a harmonica, and the third playing a bass guitar.	 A white dog rushes down a dirt path surrounded by grass and trees.
	 Two men, both in strange hats, working over rocks in a busy urban street.	 The tan dog is carrying a green squeak toy in its mouth.
	 Several people are in a group where a man in a blue shirt is smiling.	 A yellow dog running through a yard covered in leaves while holding a yellow toy in his mouth.
Test example		
OF-9B generations:	A yellow labrador retriever running with a ball.	A yellow dog running through a yard covered in leaves while holding a green squeak toy in his mouth
	GROUND TRUTH: *A white dog fetching a yellow toy.*	

Table 10: Comparison of OpenFlamingo-9B outputs for a Flickr-30K 4-shot evaluation using RICES vs. random demonstrations. With RICES, OpenFlamingo-9B patches together these demonstration captions to answer for the test image, including incorrect details.

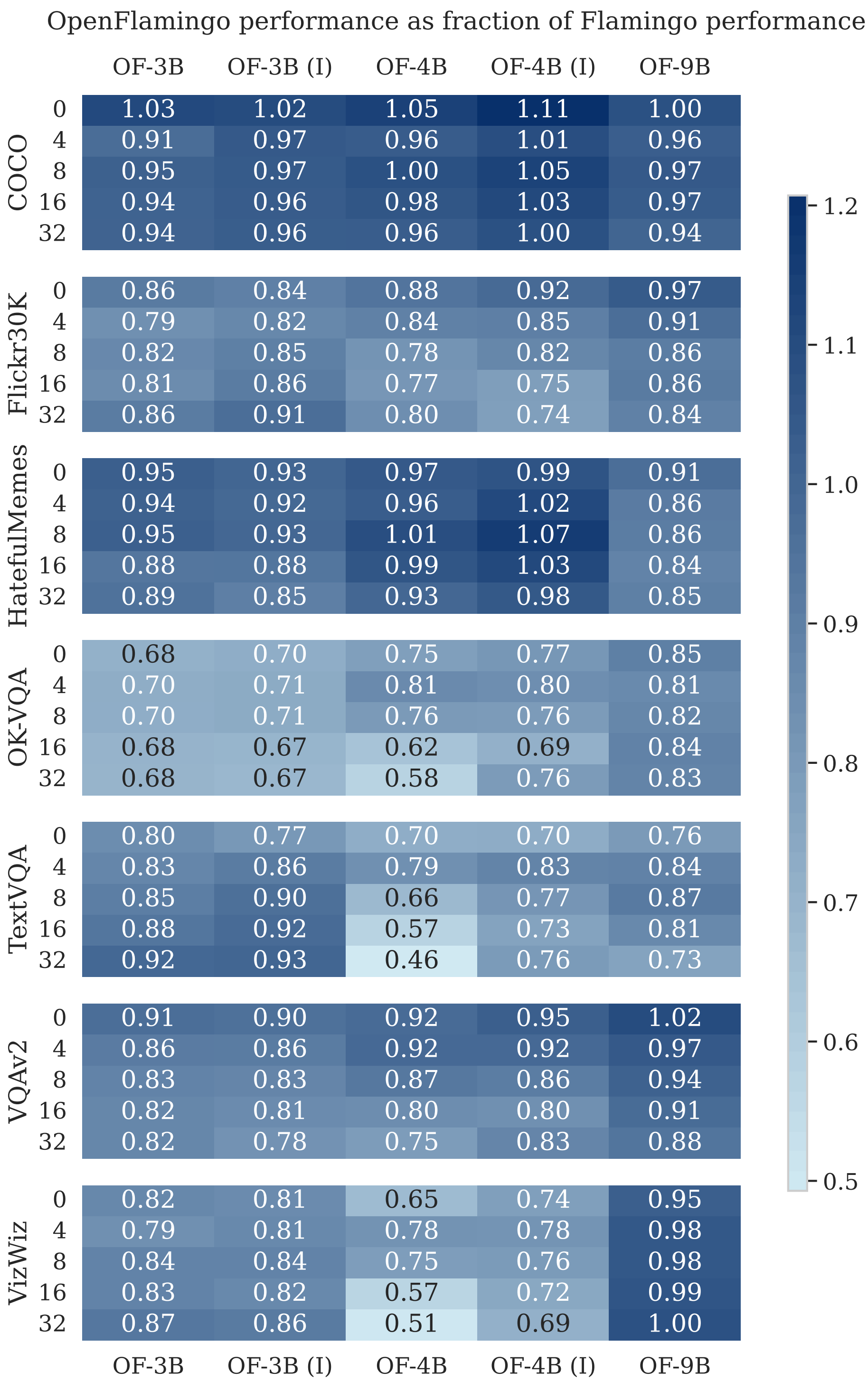

Figure 11: OpenFlamingo performance as a fraction of corresponding Flamingo performance for each evaluation setting. We compare OpenFlamingo-3B and -4B models to Flamingo-3B, and OpenFlamingo-9B to Flamingo-9B. 0, 4, 8, 16, and 32 refer to the number of in-context examples used.

Benchmark	Shots	Fl-3B	Fl-9B	OF-3B	OF-3B (I)	OF-4B	OF-4B (I)	OF-9B
COCO [7]	0	73.0	79.4	74.9 (0.2)	74.4 (0.6)	76.7 (0.2)	**81.2 (0.3)**	79.5 (0.2)
	4	85.0	**93.1**	77.3 (0.3)	82.7 (0.7)	81.8 (0.4)	85.8 (0.5)	89.0 (0.3)
	8	90.6	**99.0**	85.9 (0.6)	87.8 (0.5)	90.7 (0.3)	94.8 (0.2)	96.3 (0.1)
	16	95.4	**102.2**	89.8 (0.2)	91.9 (0.3)	93.9 (0.4)	98.0 (0.3)	98.8 (0.7)
	32	99.0	**106.3**	93.0 (0.6)	94.8 (0.3)	95.1 (0.3)	99.2 (0.3)	99.5 (0.1)
Flickr-30K [40]	0	60.6	**61.5**	52.3 (1.0)	51.2 (0.2)	53.6 (0.9)	55.6 (1.3)	59.5 (1.0)
	4	72.0	**72.6**	57.2 (0.4)	59.1 (0.3)	60.7 (1.2)	61.2 (0.5)	65.8 (0.6)
	8	71.7	**73.4**	58.6 (1.1)	60.7 (0.6)	55.9 (1.3)	59.0 (1.0)	62.9 (1.0)
	16	**73.4**	72.7	59.2 (0.5)	63.0 (0.4)	56.8 (0.5)	54.8 (1.0)	62.8 (1.0)
	32	71.2	**72.8**	61.1 (1.3)	64.5 (1.3)	56.9 (0.7)	53.0 (0.5)	61.3 (0.7)
VQAv2 [2]	0	49.2	51.8	44.6 (0.0)	44.1 (0.1)	45.1 (0.1)	46.9 (0.0)	**52.7 (0.2)**
	4	53.2	**56.3**	45.8 (0.0)	45.7 (0.1)	49.0 (0.0)	49.0 (0.0)	54.8 (0.0)
	8	55.4	**58.0**	46.2 (0.0)	45.9 (0.1)	48.3 (0.0)	47.4 (0.0)	54.8 (0.0)
	16	56.7	**59.4**	46.6 (0.0)	45.8 (0.0)	45.5 (0.1)	45.1 (0.1)	54.3 (0.0)
	32	57.1	**60.4**	47.0 (0.1)	44.8 (0.1)	43.0 (0.2)	47.3 (0.0)	53.3 (0.1)
OK-VQA [26]	0	41.2	**44.7**	28.2 (0.2)	28.7 (0.1)	30.7 (0.1)	31.7 (0.1)	37.8 (0.2)
	4	43.3	**49.3**	30.3 (0.5)	30.6 (0.2)	35.1 (0.0)	34.6 (0.0)	40.1 (0.1)
	8	44.6	**50.0**	31.1 (0.3)	31.5 (0.3)	33.9 (0.1)	33.7 (0.2)	41.1 (0.2)
	16	45.6	**50.8**	30.9 (0.3)	30.7 (0.3)	28.5 (0.2)	31.3 (0.1)	42.7 (0.2)
	32	45.9	**51.0**	31.0 (0.1)	30.6 (0.1)	26.4 (0.2)	34.7 (0.3)	42.4 (0.0)
TextVQA [33]	0	30.1	**31.8**	24.2 (0.2)	23.1 (0.2)	21.0 (0.3)	21.1 (0.4)	24.2 (0.5)
	4	32.7	**33.6**	27.0 (0.3)	28.1 (0.4)	25.9 (0.0)	27.2 (0.3)	28.2 (0.4)
	8	32.4	**33.6**	27.7 (0.1)	29.1 (0.1)	21.3 (0.2)	25.1 (0.2)	29.1 (0.1)
	16	31.8	**33.5**	28.0 (0.2)	29.1 (0.1)	18.2 (0.4)	23.2 (0.1)	27.3 (0.1)
	32	30.6	**32.6**	28.3 (0.2)	28.5 (0.1)	14.1 (0.2)	23.2 (0.2)	23.8 (0.2)
VizWiz [11]	0	**28.9**	28.8	23.7 (0.5)	23.4 (0.3)	18.8 (0.1)	21.5 (0.2)	27.5 (0.2)
	4	34.0	**34.9**	27.0 (0.3)	27.7 (0.1)	26.6 (0.5)	26.5 (0.4)	34.1 (0.7)
	8	38.4	**39.4**	32.1 (0.7)	32.1 (0.6)	28.8 (0.4)	29.1 (0.2)	38.5 (0.1)
	16	**43.3**	43.0	36.1 (0.3)	35.3 (0.1)	24.6 (0.2)	31.0 (0.6)	42.5 (0.4)
	32	**45.5**	44.0	39.8 (0.1)	39.3 (0.4)	23.1 (1.1)	31.3 (0.2)	44.0 (0.5)
HatefulMemes [15]	0	53.7	**57.0**	51.2 (2.5)	50.1 (2.2)	52.3 (2.3)	53.1 (2.2)	51.6 (1.8)
	4	53.6	**62.7**	50.6 (0.8)	49.5 (0.6)	51.5 (1.4)	54.9 (1.1)	54.0 (2.0)
	8	54.7	**63.9**	52.0 (1.1)	50.7 (1.8)	55.2 (0.8)	58.5 (0.3)	54.7 (2.8)
	16	55.3	**64.5**	48.5 (0.7)	48.7 (1.0)	54.5 (1.3)	56.9 (1.5)	53.9 (3.1)
	32	56.3	**63.5**	50.2 (1.8)	47.8 (2.2)	52.2 (1.2)	54.9 (1.1)	53.8 (2.1)

Table 11: Full evaluation results using **demonstrations sampled uniformly at random** across seven vision-language datasets using 0, 4, 8, 16, and 32 in-context examples. Results are averaged across 3 evaluation seeds and reported with standard deviations.

Benchmark	Shots	Fl-3B	Fl-9B	OF-3B	OF-3B (I)	OF-4B	OF-4B (I)	OF-9B
COCO [7]	0	73.0	79.4	**81.6**	79.0	75.9	79.5	76.8
	4	85.0	**93.1**	91.3	90.5	89.0	92.7	**93.1**
	8	90.6	99.0	97.4	96.8	96.6	**101.6**	99.8
	16	95.4	102.2	102.2	101.1	102.4	**106.0**	105.6
	32	99.0	106.3	105.5	104.1	105.1	107.8	**109.0**
Flickr-30K [40]	0	60.6	**61.5**	46.0	45.0	36.3	43.0	39.2
	4	72.0	**72.6**	54.0	53.9	48.9	53.6	52.2
	8	71.7	**73.4**	58.6	58.6	51.3	53.4	58.7
	16	**73.4**	72.7	62.0	61.5	56.5	56.4	60.6
	32	71.2	**72.8**	65.5	64.6	56.2	56.6	63.0
VQAv2 [2]	0	49.2	51.8	44.5	44.0	44.9	46.6	**53.0**
	4	53.2	**56.3**	45.9	45.3	49.0	49.6	55.1
	8	55.4	**58.0**	47.0	46.1	48.5	47.7	56.0
	16	56.7	**59.4**	48.0	46.6	46.6	45.4	56.6
	32	57.1	**60.4**	49.1	46.7	44.5	48.9	56.8
OK-VQA [26]	0	41.2	**44.7**	28.6	29.2	31.0	32.0	38.3
	4	43.3	**49.3**	31.0	31.4	34.8	35.1	42.0
	8	44.6	**50.0**	32.6	32.8	33.3	34.6	44.1
	16	45.6	**50.8**	33.8	33.9	29.2	32.7	45.1
	32	45.9	**51.0**	35.1	34.6	28.7	36.5	46.3
TextVQA [33]	0	30.1	**31.8**	24.5	26.0	22.8	23.5	28.3
	4	32.7	33.6	27.2	28.9	26.7	28.9	**34.2**
	8	32.4	33.6	28.0	28.8	22.7	25.6	**34.8**
	16	31.8	**33.5**	27.5	28.8	19.5	22.0	33.4
	32	30.6	**32.6**	29.2	29.0	17.93	23.0	31.1
VizWiz [11]	0	28.9	28.8	32.1	32.6	28.0	30.1	**34.6**
	4	34.0	34.9	37.3	39.0	37.4	36.6	**41.0**
	8	38.4	39.4	41.7	42.5	35.7	37.0	**45.0**
	16	43.3	43.0	44.8	44.9	31.5	36.6	**46.2**
	32	45.5	44.0	45.8	45.9	27.3	36.4	**46.4**
HatefulMemes [15]	0	53.7	57.0	63.9	65.2	64.8	66.2	**67.1**
	4	53.6	62.7	66.1	65.7	65.4	66.8	**70.1**
	8	54.7	63.9	67.2	66.0	67.8	69.3	**71.2**
	16	55.3	64.5	68.1	65.0	67.5	67.1	**73.2**
	32	56.3	63.5	68.2	67.7	64.4	62.2	**73.6**

Table 12: Full evaluation results using RICES across seven vision-language datasets using 0, 4, 8, 16, and 32 in-context examples.

Row 1		
Link	Link	Link
Row 2		
Link	Link	Link
Row 3		
Link	Link	Link

Table 13: Source of images for Figure 2 for each row the links are from left to right in the order of how the images are presented.

`system prompt` (human authored)

You are a helpful assistant that knows a lot about image descriptions and how they are used with text in broader contexts.

Your job is to generate an interleaved sequence of images (referenced by their image descriptions) and sentences.

Image descriptions should only describe the single image. They should be specific, 1-sentence, self-contained descriptions that capture the salient elements of the image.

The sentences can refer to concepts in any of the images and to each-other. The interleaved sequences are interesting, clever, and represent compelling patterns. To understand them requires knowledge of real-world entities/events, comparison between the images, and commonsense knowledge. Real-world references/facts must be correct.

Remember to:
1) be creative. tell stories, reference real-world entities/events, etc. The images/sentence can play off each-other in fun ways.
2) be interesting. generate sequences that are cool, fun, compelling and require interesting commonsense reasoning across and between images/sentences.
3) make sure the image descriptions are self-contained, and the output format follows the requested format.

`user` (human authored)

Generate a creative, interesting sequence of sentences/images with the following format: (image A, sentence 1, image B, sentence 2, image C, sentence 3)

`assistant` (human authored)

Sure!

Sequence format: (image A, sentence 1, image B, sentence 2, image C, sentence 3). This interleaved sequence of images/sentences contains images of animals from various parts of the world. The sentences first name the depicted animal, and then describe where the animal is native to.

Image A: A photo of a person holding a chinchilla in their hand
Sentence 1: This is a chinchilla. They are mainly found in Chile.
Image B: A photo of a shiba-enu in the fall sitting in the forest
Sentence 2: This is a shiba. They are very popular in japan.
Image C: A photo of a flamingo with a lake in the background
Sentence 3: This is a flamingo. They are found in the Caribbean and South America.

`(... 4x more hand designed/interleaved examples...)`

`user` (human authored)

Generate a creative, interesting sequence of sentences/images with the following format: (image A, image B, sentence 1, image C, image D, sentence 2, image E, image F, sentence 3).

`assistant` (machine generated)

Sure!

This interleaved sequence of images/sentences contains images of nature. The sentences that follow each image are statements that lead to conclusions. The final sentence explains the broader significance of the pattern.
Image A: A photo of a seed being planted in the soil.
Image B: A photo of a small plant sprouting from the soil.
Sentence 1: The plant is growing.
Image C: A photo of trees in a forest.
Image D: A photo of an orchard with rows of apple trees.
Sentence 2: Trees are used to produce fruits.
Image E: A photo of a river running through a valley.
Image F: A photo of a waterfall cascading down a mountain.
Sentence 3: Water is a critical component of many natural processes, from plant growth to the formation of landscapes.

Figure 12: We prompt ChatGPT using a system prompt, 5 interleaved human-authored sequences, and a query pattern, in this case "(Image A, Image B, Sentence 1, Image C, Image D, Sentence 2, Image E, Image F, Sentence 3)". Based on these examples, ChatGPT first explains what sequence it is attempting to generate (here: "... images of nature. The sentences that follow each image are statements that lead to conclusions"). Later, images are retrieved based on the generated "Photo X" descriptions to form interleaved sequences.