Flamingo-80B. For brevity, we refer to the last as *Flamingo* throughout the paper. While increasing the parameter count of the frozen LM and the trainable vision-text GATED XATTN-DENSE modules, we maintain a fixed-size frozen vision encoder and trainable Perceiver Resampler across the different models (small relative to the full model size). See Appendix B.1.1 for further details.

2.3 Multi-visual input support: per-image/video attention masking

The image-causal modelling introduced in Equation (1) is obtained by masking the full text-to-image cross-attention matrix, limiting which visual tokens the model sees at each text token. At a given text token, the model attends to the visual tokens of the image that appeared just before it in the interleaved sequence, rather than to all previous images (formalized and illustrated in Appendix A.1.3). Though the model only *directly* attends to a single image at a time, the dependency on all previous images remains via self-attention in the LM. This single-image cross-attention scheme importantly allows the model to seamlessly generalise to any number of visual inputs, regardless of how many are used during training. In particular, we use only up to 5 images per sequence when training on our interleaved datasets, yet our model is able to benefit from sequences of up to 32 pairs (or "shots") of images/videos and corresponding texts during evaluation. We show in Section 3.3 that this scheme is more effective than allowing the model to cross-attend to all previous images directly.

2.4 Training on a mixture of vision and language datasets

We train the Flamingo models on a mixture of three kinds of datasets, all scraped from the web: an interleaved image and text dataset derived from webpages, image-text pairs, and video-text pairs.

M3W: Interleaved image and text dataset. The few-shot capabilities of Flamingo models rely on training on interleaved text and image data. For this purpose, we collect the *MultiModal MassiveWeb* (*M3W*) dataset. We extract both text and images from the HTML of approximately 43 million webpages, determining the positions of images relative to the text based on the relative positions of the text and image elements in the Document Object Model (DOM). An example is then constructed by inserting `<image>` tags in plain text at the locations of the images on the page, and inserting a special `<EOC>` (*end of chunk*) token (added to the vocabulary and learnt) prior to any image and at the end of the document. From each document, we sample a random subsequence of $L = 256$ tokens and take up to the first $N = 5$ images included in the sampled sequence. Further images are discarded in order to save compute. More details are provided in Appendix A.3.

Pairs of image/video and text. For our image and text pairs we first leverage the ALIGN [50] dataset, composed of 1.8 billion images paired with alt-text. To complement this dataset, we collect our own dataset of image and text pairs targeting better quality and longer descriptions: LTIP (Long Text & Image Pairs) which consists of 312 million image and text pairs. We also collect a similar dataset but with videos instead of still images: VTP (Video & Text Pairs) consists of 27 million short videos (approximately 22 seconds on average) paired with sentence descriptions. We align the syntax of paired datasets with the syntax of M3W by prepending `<image>` and appending `<EOC>` to each training caption (see Appendix A.3.3 for details).

Multi-objective training and optimisation strategy. We train our models by minimizing a weighted sum of per-dataset expected negative log-likelihoods of text, given the visual inputs:

$$\sum_{m=1}^{M} \lambda_m \cdot \mathbb{E}_{(x,y)\sim\mathcal{D}_m}\left[-\sum_{\ell=1}^{L} \log p(y_\ell | y_{<\ell}, x_{\leq\ell})\right], \tag{2}$$

where $\mathcal{D}_m$ and λ_m are the m-th dataset and its weighting, respectively. Tuning the per-dataset weights λ_m is key to performance. We accumulate gradients over all datasets, which we found outperforms a "round-robin" approach [17]. We provide further training details and ablations in Appendix B.1.2.

2.5 Task adaptation with few-shot in-context learning

Once Flamingo is trained, we use it to tackle a visual task by conditioning it on a multimodal interleaved prompt. We evaluate the ability of our models to rapidly adapt to new tasks using **in-context learning**, analogously to GPT-3 [11], by interleaving support example pairs in the form of $(image, text)$ or $(video, text)$, followed by the query visual input, to build a prompt (details in Appendix A.2). We perform **open-ended** evaluations using beam search for decoding, and **close-ended**

Method	FT	Shot	OKVQA **(I)**	VQAv2 **(I)**	COCO **(I)**	MSVDQA **(V)**	VATEX **(V)**	VizWiz **(I)**	Flick30K **(I)**	MSRVTTQA **(V)**	iVQA **(V)**	YouCook2 **(V)**	STAR **(V)**	VisDial **(I)**	TextVQA **(I)**	NextQA **(I)**	HatefulMemes **(I)**	RareAct **(V)**
Zero/Few shot SOTA	✗	(X)	[34] 43.3 (16)	[114] 38.2 (4)	[124] 32.2 (0)	[58] 35.2 (0)	-	-	-	[58] 19.2 (0)	[135] 12.2 (0)	-	[143] 39.4 (0)	[79] 11.6 (0)	-	-	[85] 66.1 (0)	[85] 40.7 (0)
Flamingo-3B	✗	0	41.2	49.2	73.0	27.5	40.1	28.9	60.6	11.0	32.7	55.8	39.6	46.1	30.1	21.3	53.7	58.4
	✗	4	43.3	53.2	85.0	33.0	50.0	34.0	72.0	14.9	35.7	64.6	41.3	47.3	32.7	22.4	53.6	-
	✗	32	45.9	57.1	99.0	42.6	59.2	45.5	71.2	25.6	37.7	76.7	41.6	47.3	30.6	26.1	56.3	-
Flamingo-9B	✗	0	44.7	51.8	79.4	30.2	39.5	28.8	61.5	13.7	35.2	55.0	41.8	48.0	31.8	23.0	57.0	57.9
	✗	4	49.3	56.3	93.1	36.2	51.7	34.9	72.6	18.2	37.7	70.8	**42.8**	50.4	33.6	24.7	62.7	-
	✗	32	51.0	60.4	106.3	47.2	57.4	44.0	72.8	29.4	40.7	77.3	41.2	50.4	32.6	28.4	63.5	-
Flamingo	✗	0	50.6	56.3	84.3	35.6	46.7	31.6	67.2	17.4	40.7	60.1	39.7	52.0	35.0	26.7	46.4	**60.8**
	✗	4	57.4	63.1	103.2	41.7	56.0	39.6	75.1	23.9	44.1	74.5	42.4	**55.6**	36.5	30.8	68.6	-
	✗	32	**57.8**	**67.6**	**113.8**	**52.3**	**65.1**	**49.8**	**75.4**	**31.0**	**45.3**	**86.8**	42.2	**55.6**	**37.9**	**33.5**	**70.0**	-
Pretrained FT SOTA	✔	(X)	54.4 [34] (10K)	80.2 [140] (444K)	143.3 [124] (500K)	47.9 [28] (27K)	76.3 [153] (500K)	57.2 [65] (20K)	67.4 [150] (30K)	46.8 [51] (130K)	35.4 [135] (6K)	138.7 [132] (10K)	36.7 [128] (46K)	75.2 [79] (123K)	54.7 [137] (20K)	25.2 [129] (38K)	79.1 [62] (9K)	-

Table 1: **Comparison to the state of the art.** A *single* Flamingo model reaches the state of the art on a wide array of image **(I)** and video **(V)** understanding tasks with few-shot learning, significantly outperforming previous best zero- and few-shot methods with as few as four examples. More importantly, using only 32 examples and without adapting any model weights, Flamingo *outperforms* the current best methods – fine-tuned on thousands of annotated examples – on seven tasks. Best few-shot numbers are in **bold**, best numbers overall are underlined.

evaluations using our model's log-likelihood to score each possible answer. We explore **zero-shot generalization** by prompting the model with two text-only examples from the task, with no corresponding images. Evaluation hyperparameters and additional details are given in Appendix B.1.5.

3 Experiments

Our goal is to develop models that can rapidly adapt to diverse and challenging tasks. For this, we consider a wide array of 16 popular multimodal image/video and language benchmarks. In order to validate model design decisions during the course of the project, 5 of these benchmarks were used as part of our development (DEV) set: COCO, OKVQA, VQAv2, MSVDQA and VATEX. Performance estimates on the DEV benchmarks may be biased, as a result of model selection. We note that this is also the case for prior work which makes use of similar benchmarks to validate and ablate design decisions. To account for this, we report performance on an additional set of 11 benchmarks, spanning captioning, video question-answering, as well as some less commonly explored capabilities such as visual dialogue and multi-choice question-answering tasks. The evaluation benchmarks are described in Appendix B.1.4. We keep all evaluation hyperparameters fixed across all benchmarks. Depending on the task, we use four few-shot prompt templates we describe in more detail in Appendix B.1.5. We emphasize that *we do not validate any design decisions on these 11 benchmarks* and use them solely to estimate unbiased few-shot learning performance of our models.

Concretely, estimating few-shot learning performance of a model involves prompting it with a set of *support* samples and evaluating it on a set of *query* samples. For the DEV benchmarks that are used both to validate design decisions and hyperparameters, as well as to report final performance, we therefore use four subsets: *validation support*, *validation query*, *test support* and *test query*. For other benchmarks, we need only the latter two. We report in Appendix B.1.4 how we form these subsets.

We report the results of the Flamingo models on few-shot learning in Section 3.1. Section 3.2 gives *Flamingo* fine-tuned results. An ablation study is given in Section 3.3. Appendix B.2 provides more results including Flamingo's performance on the ImageNet and Kinetics700 classification tasks, and on our contrastive model's performance. Appendix C includes additional qualitative results.

3.1 Few-shot learning on vision-language tasks

Few-shot results. Results are given in Table 1. *Flamingo* outperforms by a large margin *all* previous zero-shot or few-shot methods on the 16 benchmarks considered. This is achieved with as few as four examples per task, demonstrating practical and efficient adaptation of vision models to new tasks. More importantly, *Flamingo* is often competitive with state-of-the-art methods additionally fine-tuned on up to hundreds of thousands of annotated examples. On six tasks, *Flamingo* even outperforms the fine-tuned SotA despite using a *single* set of model weights and only 32 task-specific examples.

Method	VQAV2		COCO	VATEX	VizWiz		MSRVTTQA	VisDial		YouCook2	TextVQA		HatefulMemes
	test-dev	test-std	test	test	test-dev	test-std	test	valid	test-std	valid	valid	test-std	test seen
🦩 32 shots	67.6	-	113.8	65.1	49.8	-	31.0	56.8	-	86.8	36.0	-	70.0
🦩 Fine-tuned	**82.0**	**82.1**	138.1	**84.2**	**65.7**	**65.4**	**47.4**	61.8	59.7	118.6	**57.1**	54.1	**86.6**
SotA	81.3† [133]	81.3† [133]	**149.6**† [119]	81.4† [153]	57.2† [65]	60.6† [65]	46.8 [51]	**75.2** [79]	**75.4**† [123]	**138.7** [132]	54.7 [137]	**73.7** [84]	84.6† [152]

Table 2: **Comparison to SotA when fine-tuning *Flamingo*.** We fine-tune *Flamingo* on all nine tasks where *Flamingo* does not achieve SotA with few-shot learning. *Flamingo* sets a new SotA on five of them, outperfoming methods (marked with †) that use tricks such as model ensembling or domain-specific metric optimisation (e.g., CIDEr optimisation).

	Ablated setting	*Flamingo*-3B original value	Changed value	Param. count ↓	Step time ↓	COCO CIDEr↑	OKVQA top1↑	VQAv2 top1↑	MSVDQA top1↑	VATEX CIDEr↑	Overall score↑
		***Flamingo*-3B model**		3.2B	1.74s	86.5	42.1	55.8	36.3	53.4	**70.7**
(i)	Training data	All data	w/o Video-Text pairs	3.2B	1.42s	84.2	43.0	53.9	34.5	46.0	67.3
			w/o Image-Text pairs	3.2B	0.95s	66.3	39.2	51.6	32.0	41.6	60.9
			Image-Text pairs→ LAION	3.2B	1.74s	79.5	41.4	53.5	33.9	47.6	66.4
			w/o M3W	3.2B	1.02s	54.1	36.5	52.7	31.4	23.5	53.4
(ii)	Optimisation	Accumulation	Round Robin	3.2B	1.68s	76.1	39.8	52.1	33.2	40.8	62.9
(iii)	Tanh gating	✓	✗	3.2B	1.74s	78.4	40.5	52.9	35.9	47.5	66.5
(iv)	Cross-attention architecture	GATED XATTN-DENSE	VANILLA XATTN	2.4B	1.16s	80.6	41.5	53.4	32.9	50.7	66.9
			GRAFTING	3.3B	1.74s	79.2	36.1	50.8	32.2	47.8	63.1
(v)	Cross-attention frequency	Every	Single in middle	2.0B	0.87s	71.5	38.1	50.2	29.1	42.3	59.8
			Every 4th	2.3B	1.02s	82.3	42.7	55.1	34.6	50.8	68.8
			Every 2nd	2.6B	1.24s	83.7	41.0	55.8	34.5	49.7	68.2
(vi)	Resampler	Perceiver	MLP	3.2B	1.85s	78.6	42.2	54.7	35.2	44.7	66.6
			Transformer	3.2B	1.81s	83.2	41.7	55.6	31.5	48.3	66.7
(vii)	Vision encoder	NFNet-F6	CLIP ViT-L/14	3.1B	1.58s	76.5	41.6	53.4	33.2	44.5	64.9
			NFNet-F0	2.9B	1.45s	73.8	40.5	52.8	31.1	42.9	62.7
(viii)	Freezing LM	✓	✗ (random init)	3.2B	2.42s	74.8	31.5	45.6	26.9	50.1	57.8
			✗ (pretrained)	3.2B	2.42s	81.2	33.7	47.4	31.0	53.9	62.7

Table 3: **Ablation studies.** Each row should be compared to the baseline Flamingo run (top row). Step time measures the time spent to perform gradient updates on all training datasets.

Finally, despite having only used the DEV benchmarks for design decisions, our results generalize well to the other benchmarks, confirming the generality of our approach.

Scaling with respect to parameters and shots. As shown in Figure 2, the larger the model, the better the few-shot performance, similar to GPT-3 [11]. The performance also improves with the number of shots. We further find that the largest model better exploits larger numbers of shots. Interestingly, even though our Flamingo models were trained with sequences limited to only 5 images on *M3W*, they are still able to benefit from up to 32 images or videos during inference. This demonstrates the flexibility of the Flamingo architecture for processing a variable number of videos or images.

3.2 Fine-tuning *Flamingo* as a pretrained vision-language model

While not the main focus of our work, we verify that when given more data, Flamingo models can be adapted to a task by fine-tuning their weights. In Table 2, we explore fine-tuning our largest model, *Flamingo*, for a given task with no limit on the annotation budget. In short, we do so by fine-tuning the model on a short schedule with a small learning rate by additionally unfreezing the vision backbone to accommodate a higher input resolution (details in Appendix B.2.2). We find that we can improve results over our previously presented in-context few-shot learning results, setting a new state of the art on five additional tasks: VQAv2, VATEX, VizWiz, MSRVTTQA, and HatefulMemes.

3.3 Ablation studies

In Table 3, we report our ablation results using *Flamingo*-3B on the *validation* subsets of the five DEV benchmarks with 4 shots. Note that we use smaller batch sizes and a shorter training schedule compared to the final models. The **Overall score** is obtained by dividing each benchmark score by its state-of-the-art (SotA) performance from Table 1 and averaging the results. More details and results are given in Appendix B.3 and Table 10.

Importance of the training data mixture. As shown in row **(i)**, getting the right training data plays a crucial role. In fact, removing the interleaved image-text dataset *M3W* leads to a *decrease of more than* 17% in performance while removing the conventional paired image-text pairs also decreases