Visual Language Model for

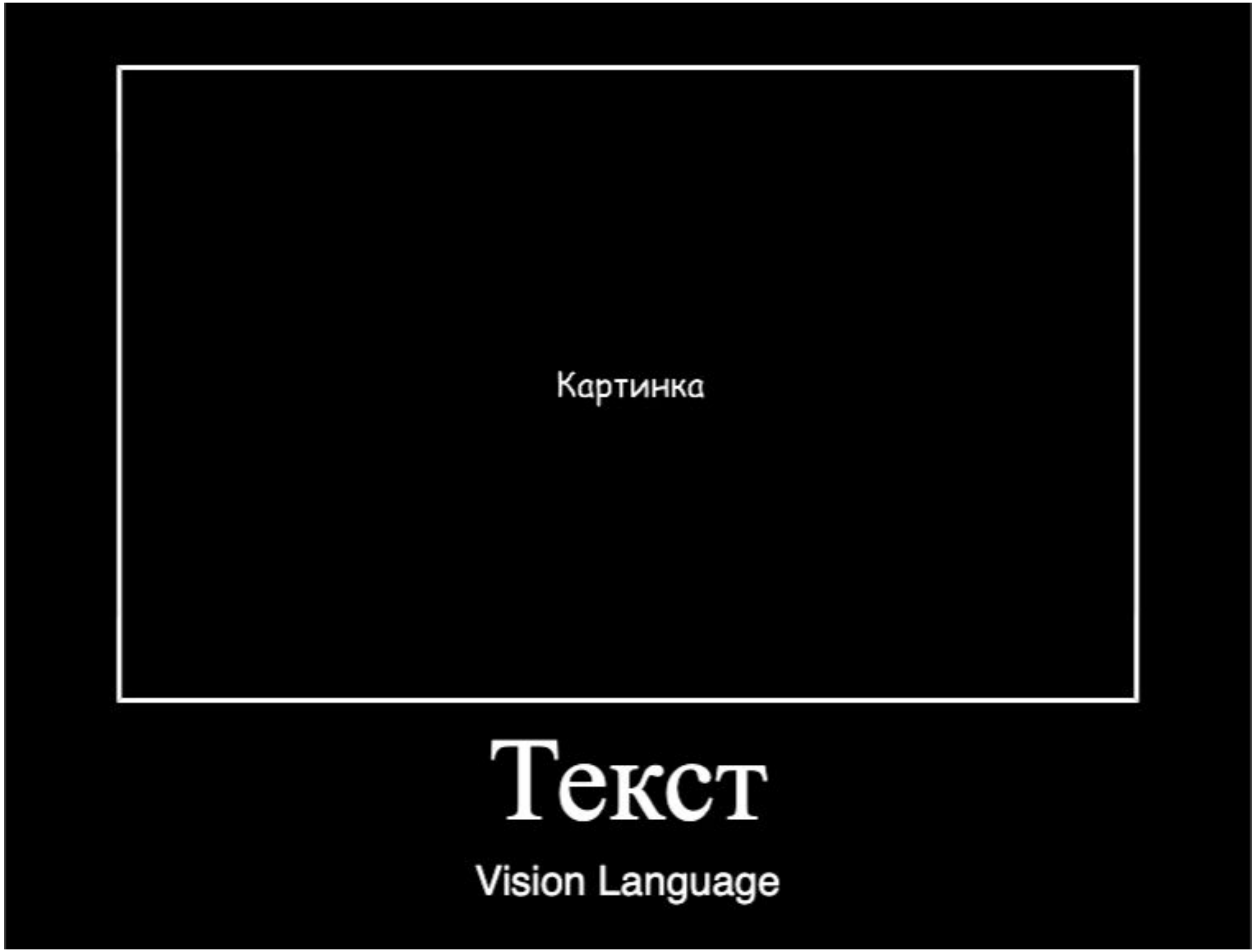
Few-Shot Learning

by DeepMind

🦩 Flamingo

Иван Мошков

Мотивация



Задачи

1. Visual question-answering
2. Captioning
3. Visual dialogue

Еще мотивация

- Хотим генеративную модель, понимающую и текст и картинки
- Хотим как в GPT-3 уметь in-context few-shot

Flamingo

Как бы мы это делали?

Вход:

<image> Что изображено на этой картинке?

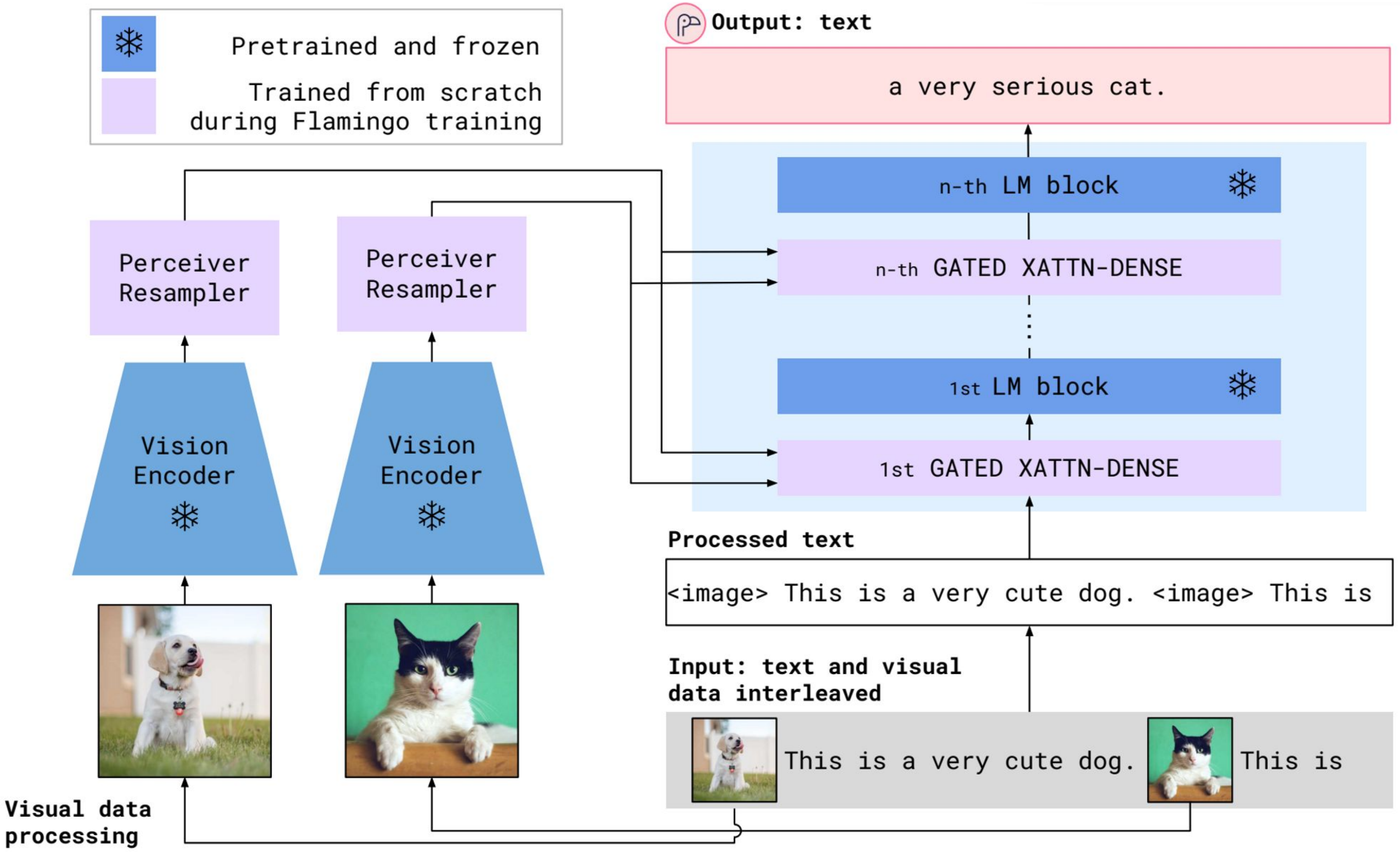
Embeddings:

img_1, word_1, word_2, … word_k

Трансформер:

Большой желтый кот

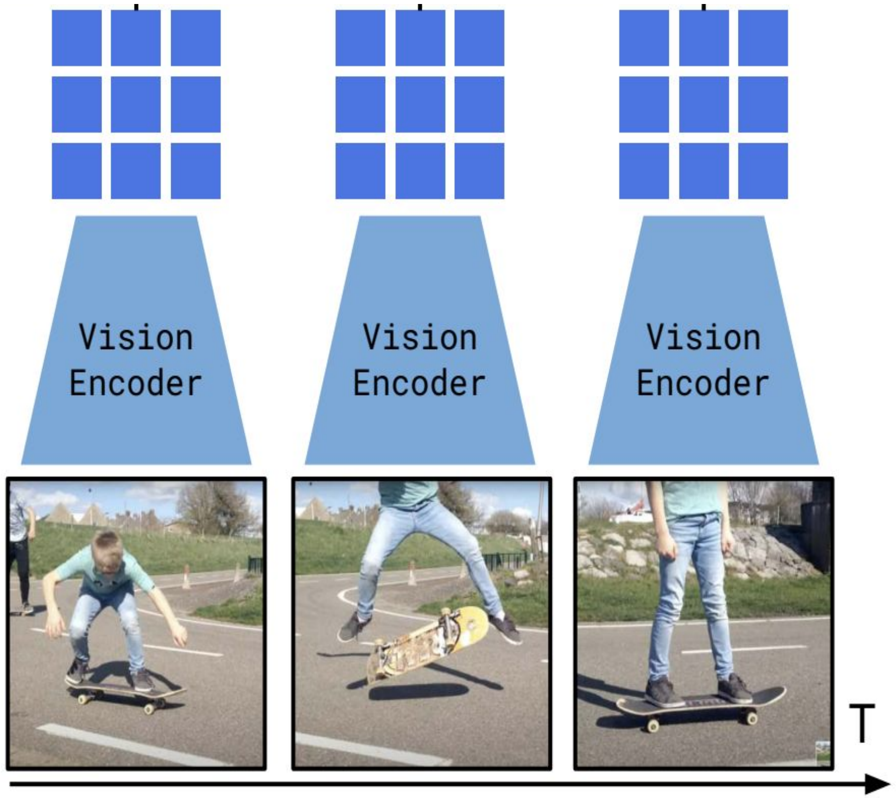
Архитектура



Flamingo

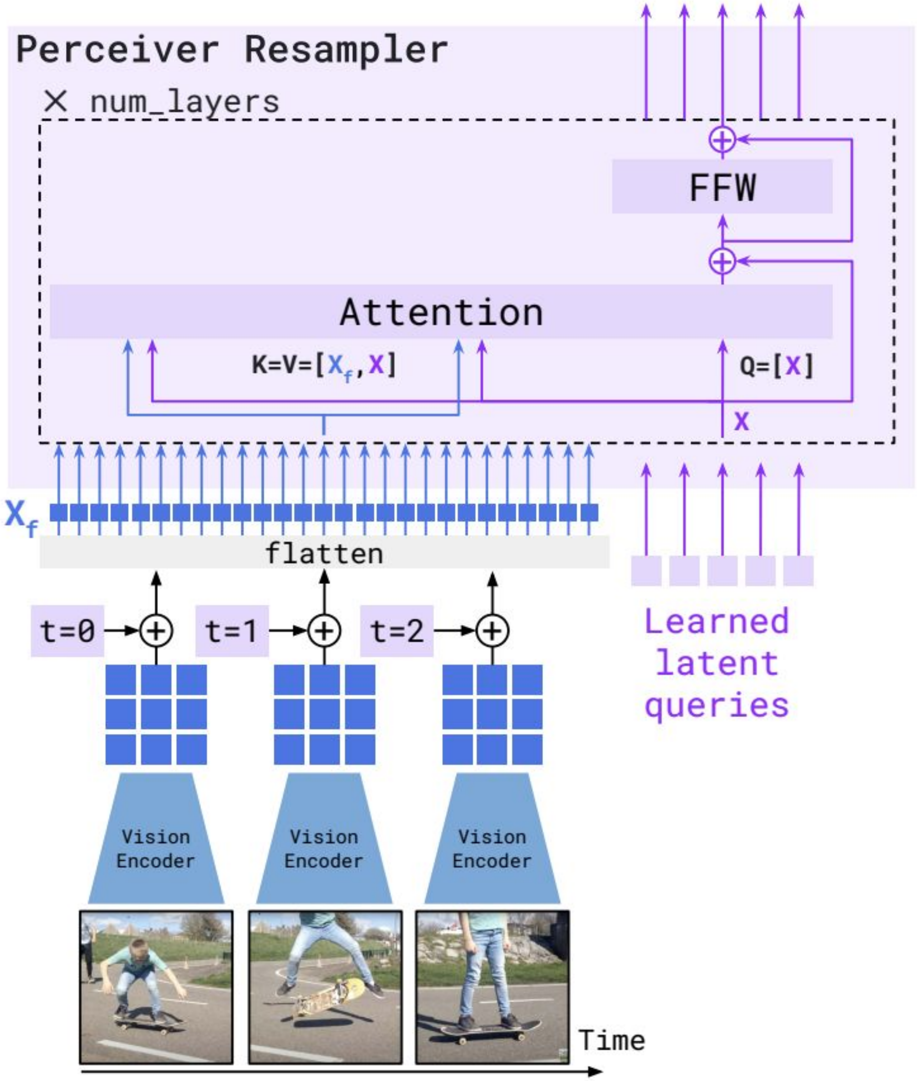
Vision Encoder

- Взяли свой продвинутый ResNet (NFNet)
- Обучили его как CLIP
- Заморозили



Perceiver Resampler

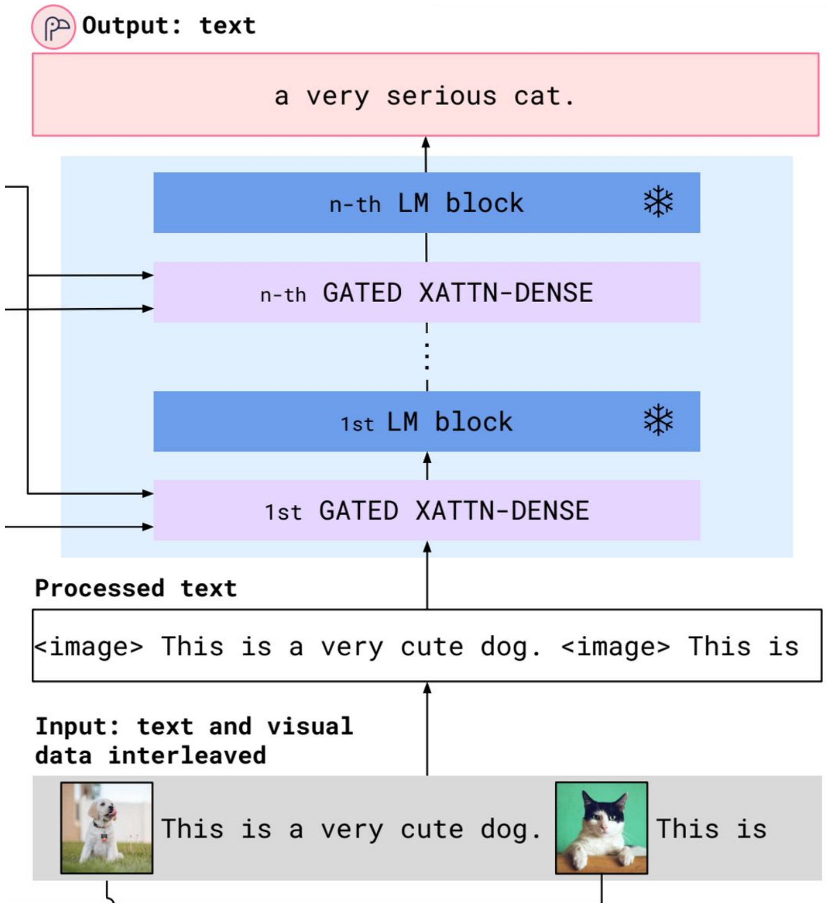
- Принимает на вход **feature-map** произвольной длины
- Выдает **feature-map** фиксированной длины



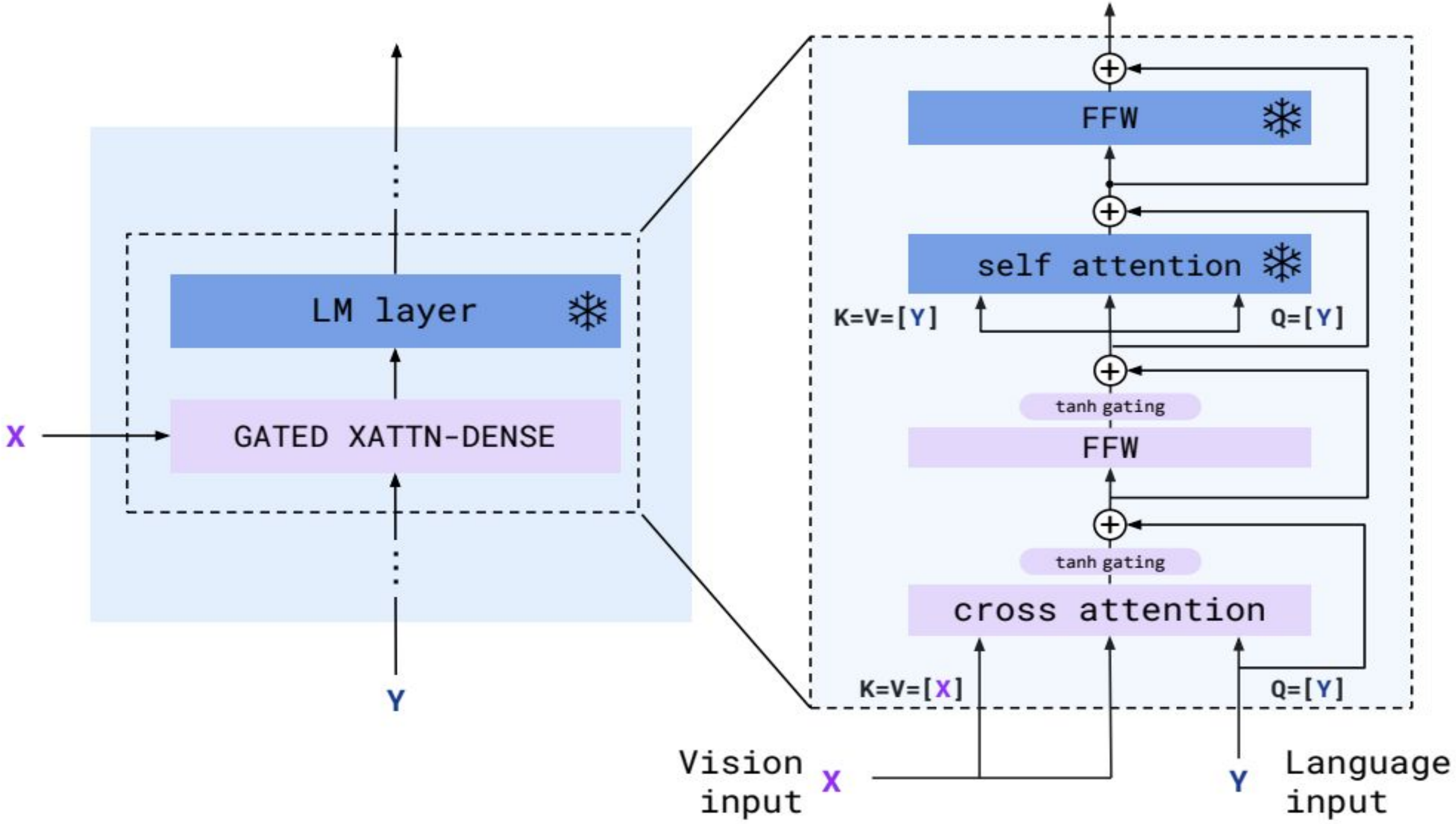
Flamingo

Language Encoder

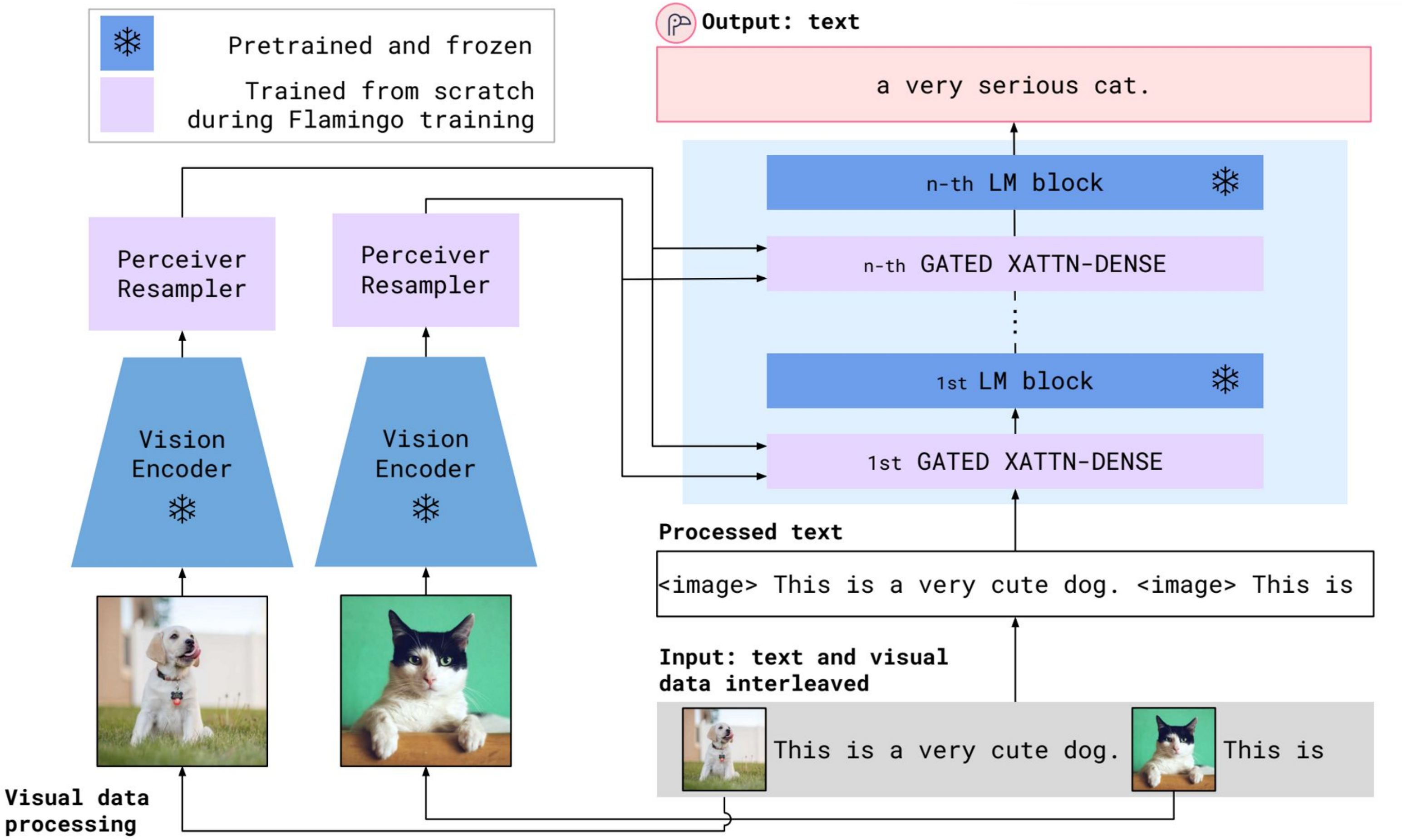
- GPT-like моделька Chinchilla (70B)
- Никак не дообучали
- Заморозили
- Добавили обучаемые слои Cross-Attention



Cross Attention



Опять архитектура



🦩 Flamingo

Данные

1. Веб-страницы с картинками
2. Картинки с описаниями
3. Короткие видосики с описаниями

Взвешиваем лосс

Эксперименты!

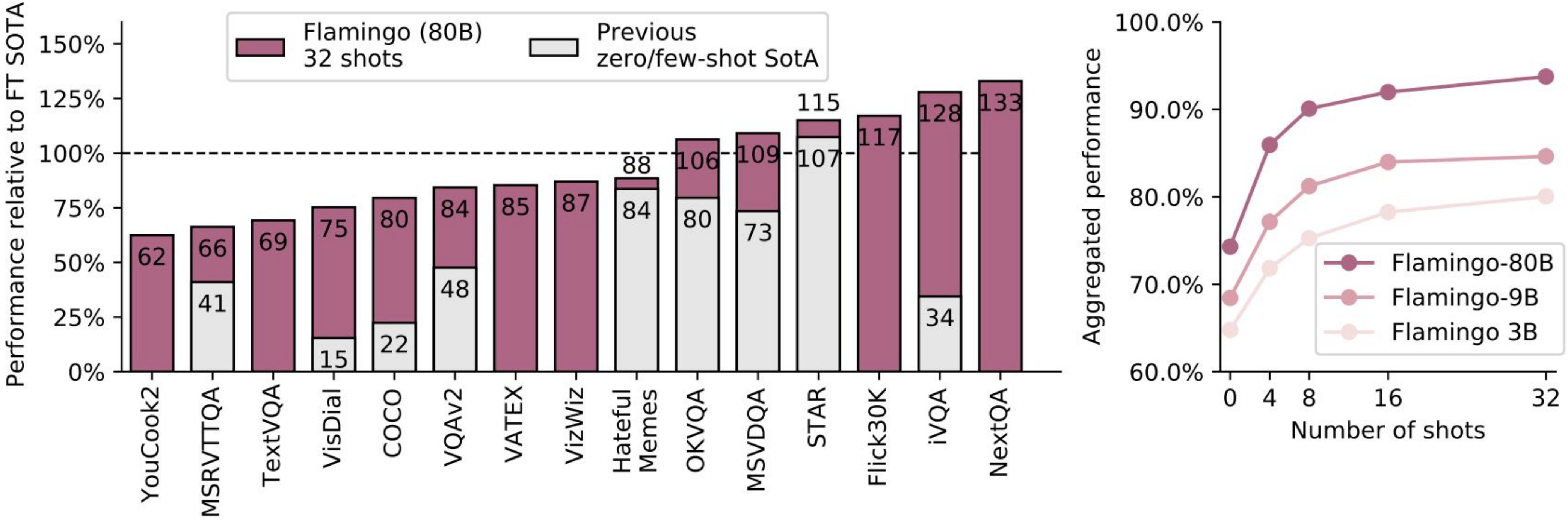


Figure 2: **Flamingo results overview.** *Left*: Our largest model, dubbed *Flamingo*, outperforms state-of-the-art fine-tuned models on 6 of the 16 tasks we consider with no fine-tuning. For the 9 tasks with published few-shot results, *Flamingo* sets the new few-shot state of the art. *Note:* We omit RareAct, our 16th benchmark, as it is a zero-shot benchmark with no available fine-tuned results to compare to. *Right*: Flamingo performance improves with model size and number of shots.

Больше экспериментов!

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).

Конец

Conclusion. We proposed Flamingo, a general-purpose family of models that can be applied to image and video tasks with minimal task-specific training data. We also qualitatively explored interactive abilities of *Flamingo* such as "chatting" with the model, demonstrating flexibility beyond traditional vision benchmarks. Our results suggest that connecting pre-trained large language models with powerful visual models is an important step towards general-purpose visual understanding.

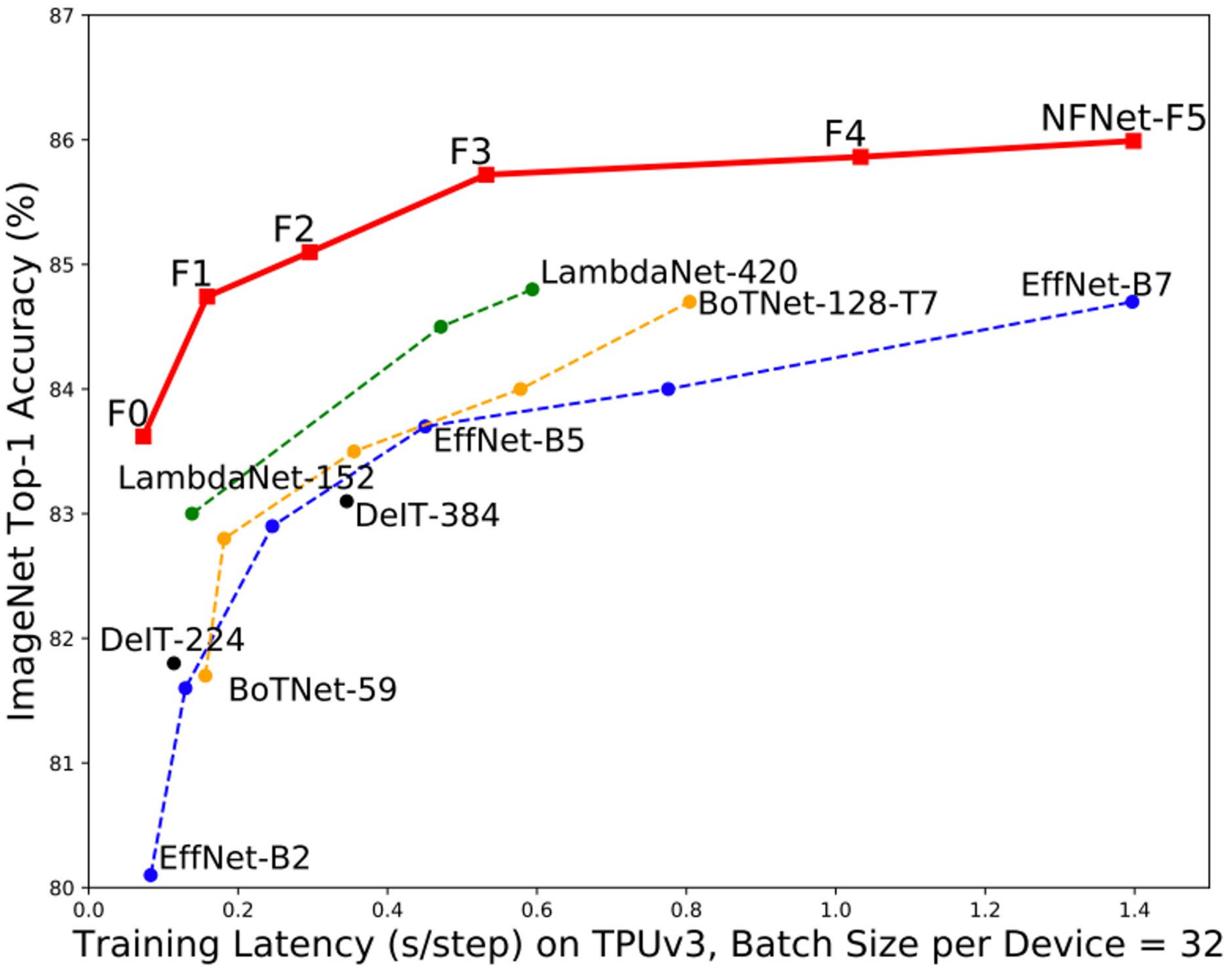
Источники

1. DeepMind Paper
2. DeepMind Blogpost
3. Small Overview
4.

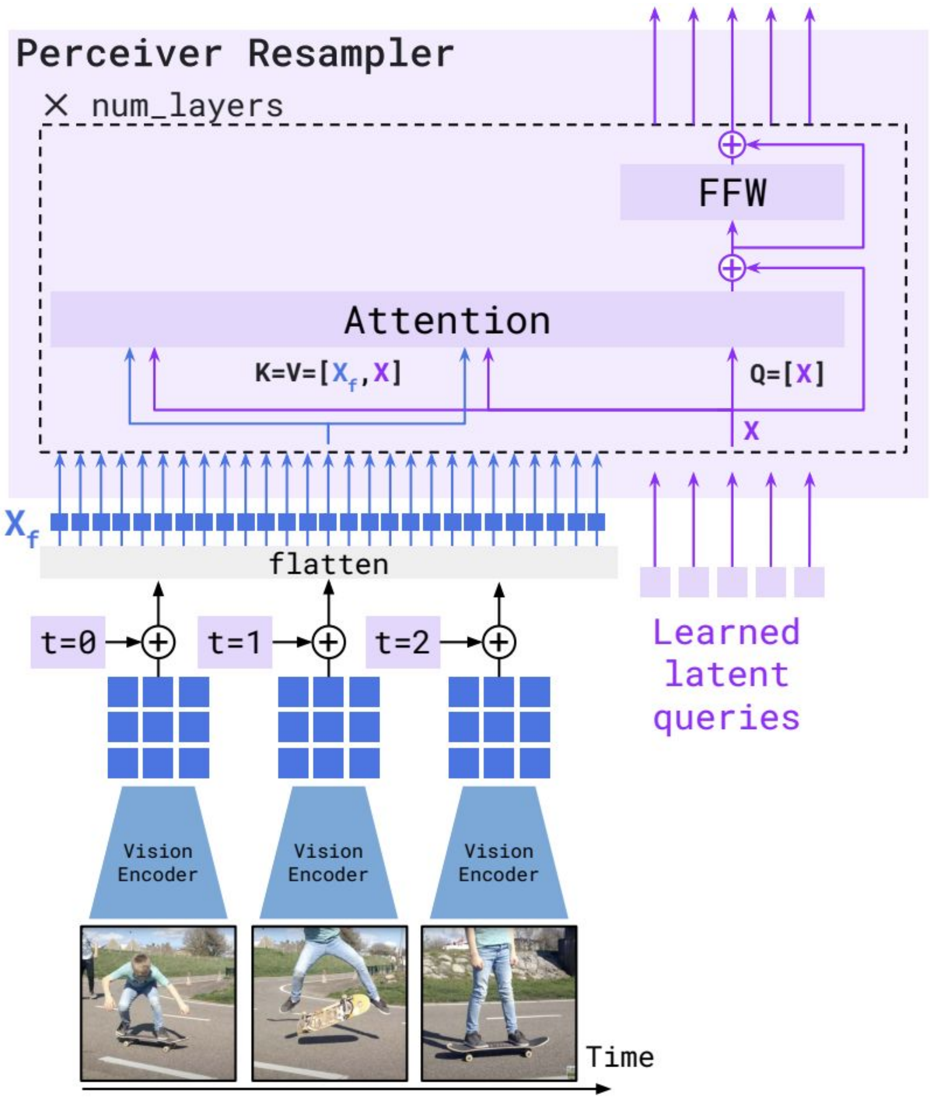
Для справки

🦩 Flamingo

NFNet



Resampler




```
def perceiver_resampler(
    x_f,  # The [T, S, d] visual features (T=time, S=space)
    time_embeddings,  # The [T, 1, d] time pos embeddings.
    x,  # R learned latents of shape [R, d]
    num_layers,  # Number of layers
):
  """The Perceiver Resampler model."""

  # Add the time position embeddings and flatten.
  x_f = x_f + time_embeddings
  x_f = flatten(x_f)  # [T, S, d] -> [T * S, d]
  # Apply the Perceiver Resampler layers.
  for i in range(num_layers):
    # Attention.
    x = x + attention_i(q=x, kv=concat([x_f, x]))
    # Feed forward.
    x = x + ffw_i(x)
  return x
```

Flamingo

Masking details

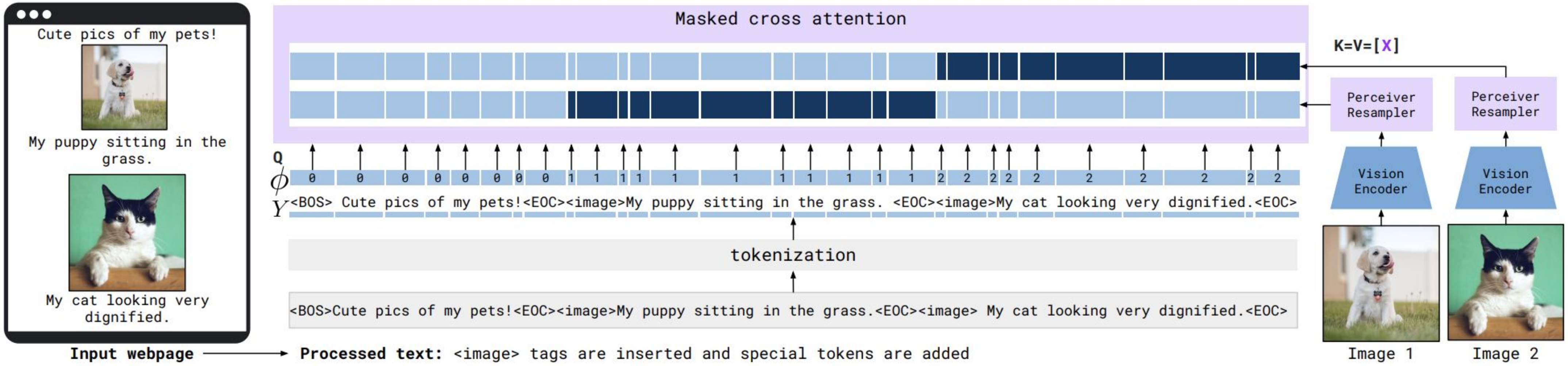


Figure 7: **Interleaved visual data and text support.** Given text interleaved with images/videos, e.g. coming from a webpage, we first process the text by inserting `<image>` tags at the locations of the visual data in the text as well as special tokens (`<BOS>` for "beginning of sequence" or `<EOC>` for "end of chunk"). Images are processed independently by the Vision Encoder and Perceiver Resampler to extract visual tokens. At a given text token, the model only cross-attends to the visual tokens corresponding to the last preceding image/video. ϕ indicates which image/video a text token can attend or 0 when no image/video is preceding. In practice, this selective cross-attention is achieved through masking – illustrated here with the dark blue entries (unmasked/visible) and light blue entries (masked).