CENG501 – Deep Learning

Week 10

Fall 2024

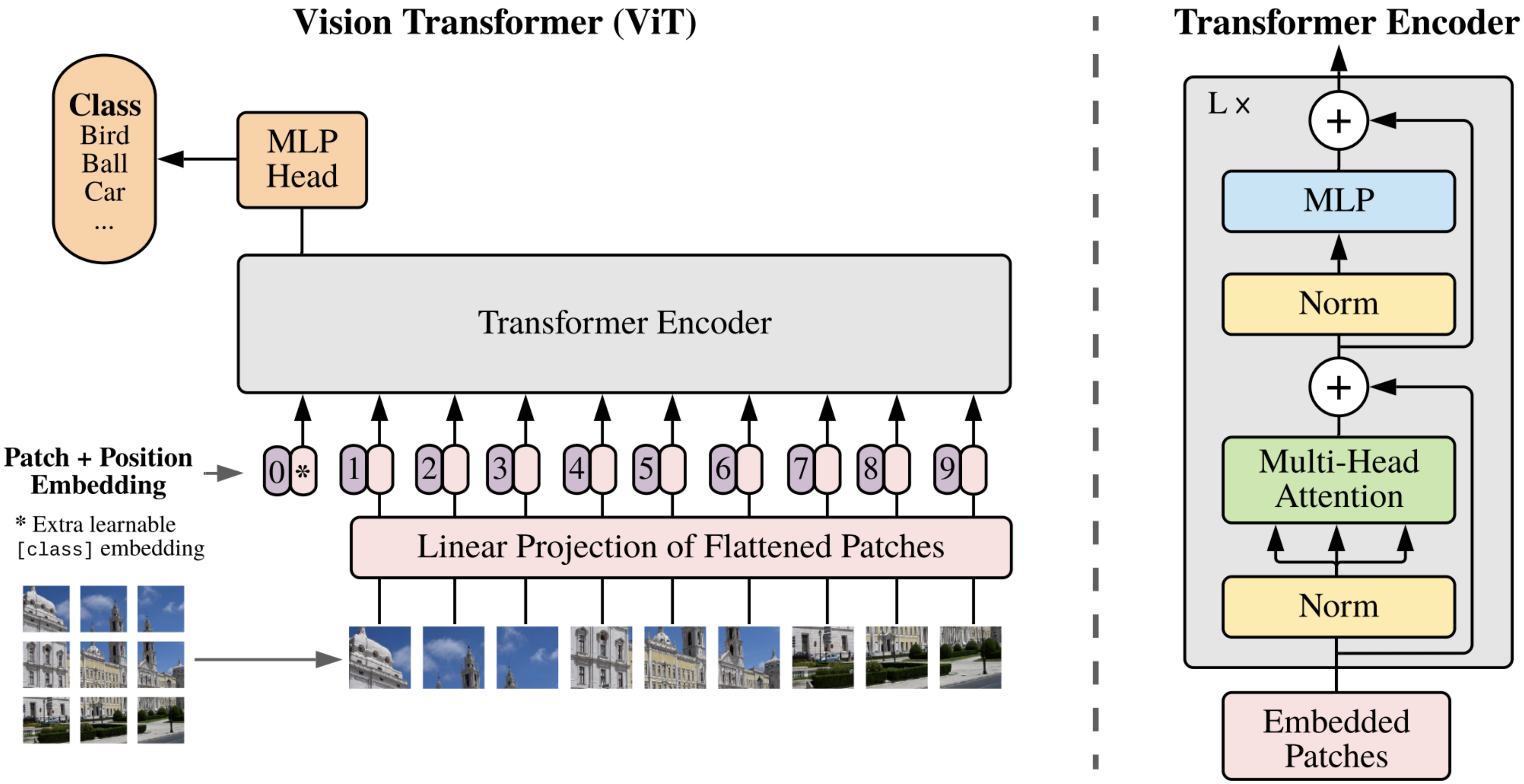
Sinan Kalkan

Dept. of Computer Engineering, METU

Previously on CENG501

ViT: Architecture

Applied to classification tasks only!

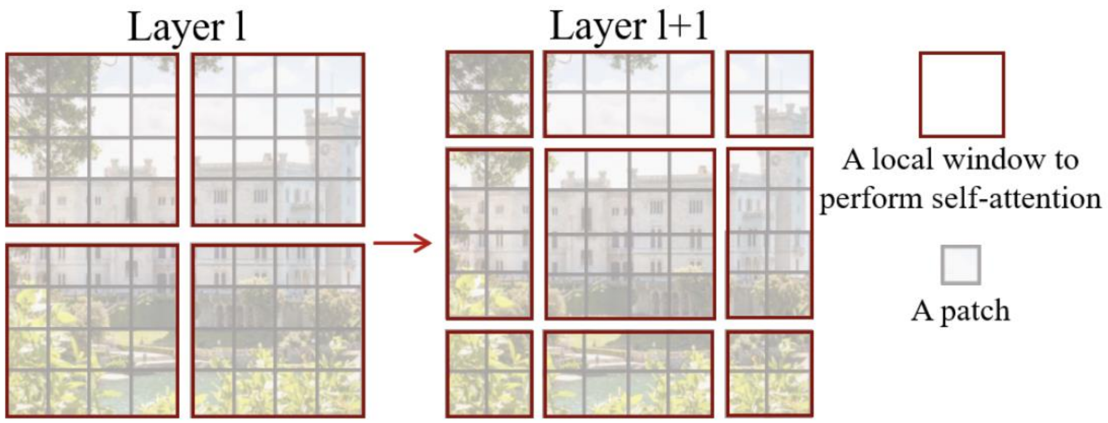


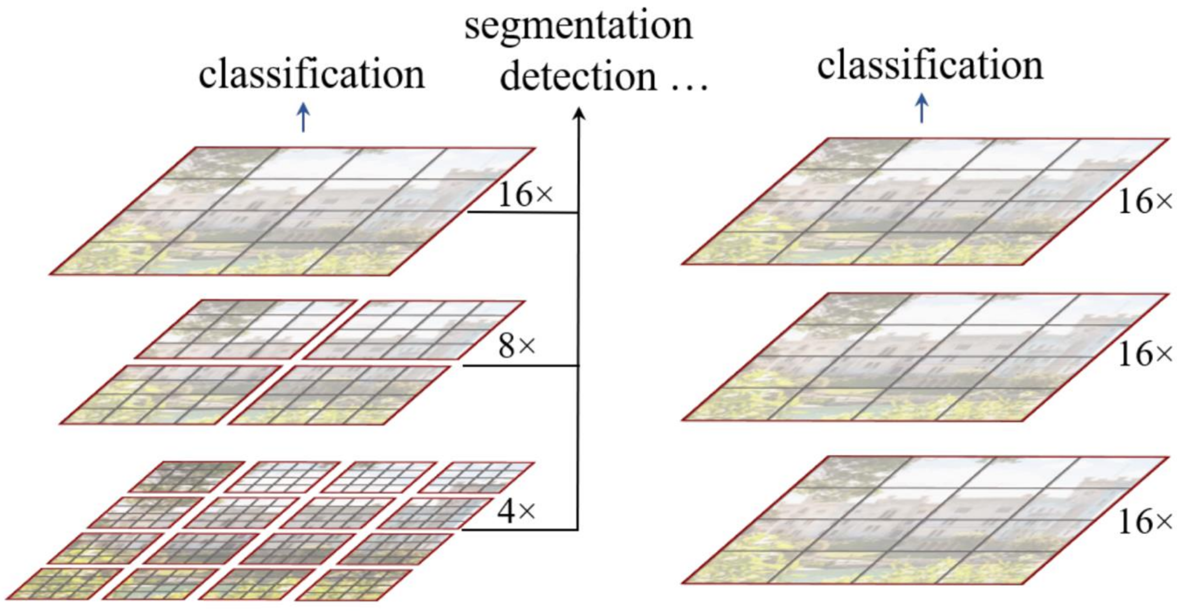
Swin Transformer v1

Previously on CENG501

- Motivation:
  - ViT is promising but limited to classification
  - Challenges in using Tranformers:
    - large variations in scales of visual entities,
    - more pixels compared to words in text
  - Existing Transformers use fixed token size across layers

- Contributions:
  - Limit self-attention to non-overlapping windows while allowing cross-window attention
  - Change token size across layers



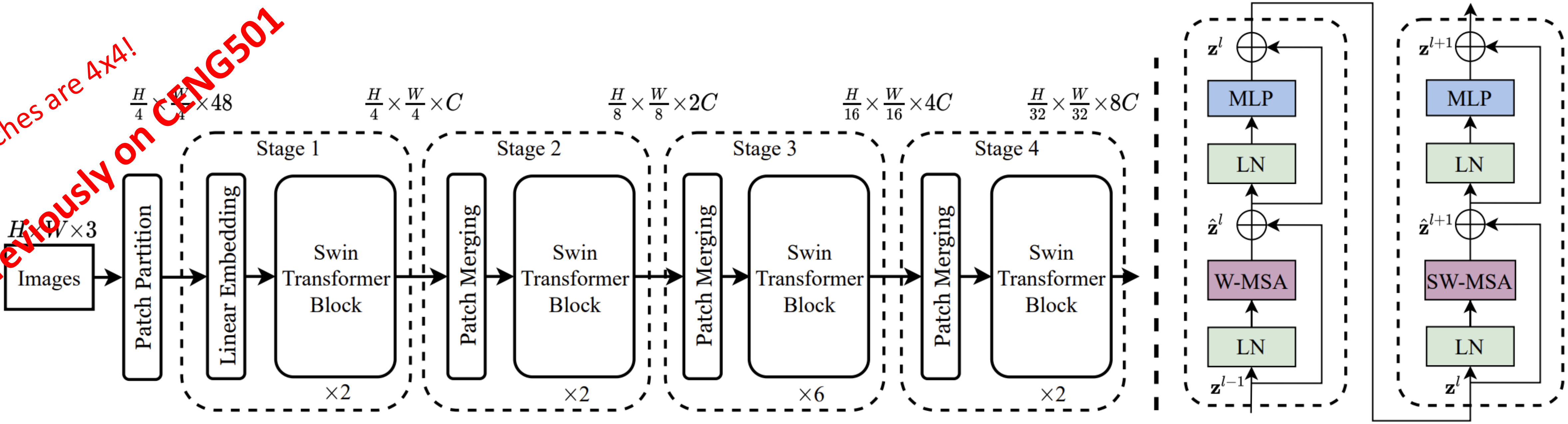


(a) Swin Transformer (ours)

(b) ViT

Patches are 4x4!

Previously on CENG501



(a) Architecture

(b) Two Successive Swin Transformer Blocks

Figure 3. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks (notation presented with Eq. (3)). W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.

Patch Merging

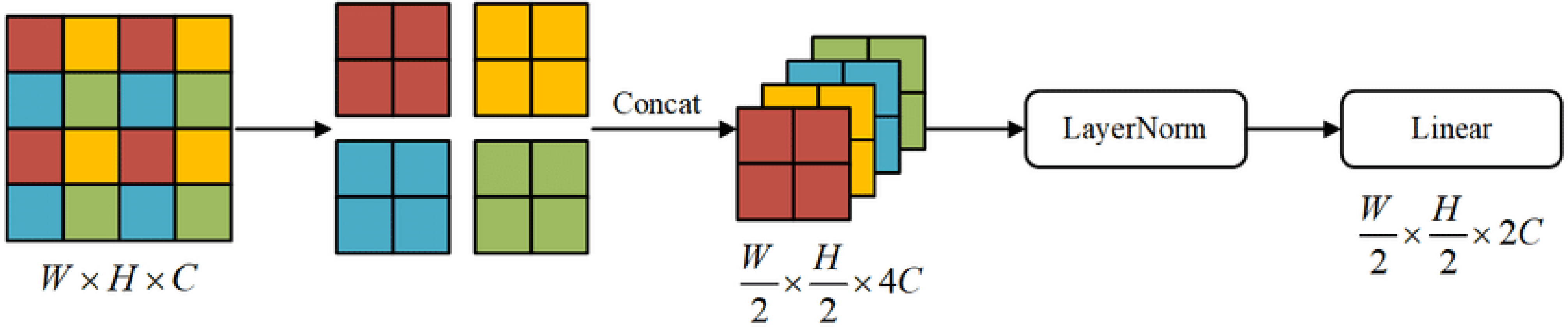
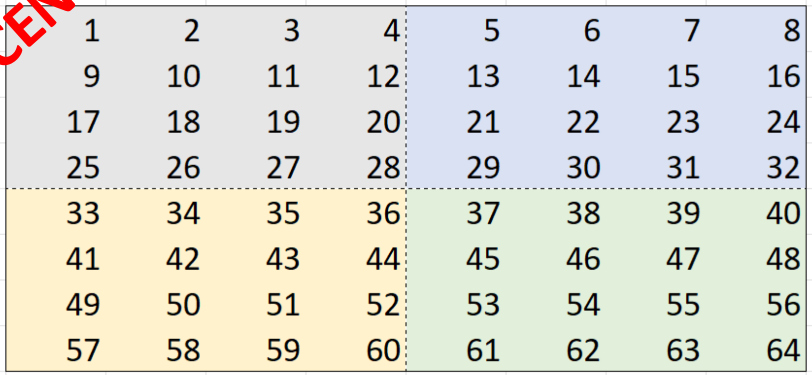

$W\times H\times C$
Concat
$\frac{W}{2}\times\frac{H}{2}\times 4C$
LayerNorm
Linear
$\frac{W}{2}\times\frac{H}{2}\times 2C$

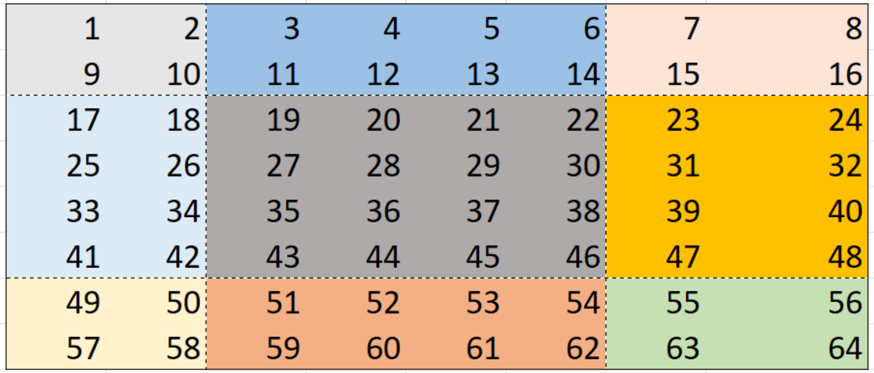
Image: http://dx.doi.org/10.48550/arXiv.2206.03336

Previously on CENG501



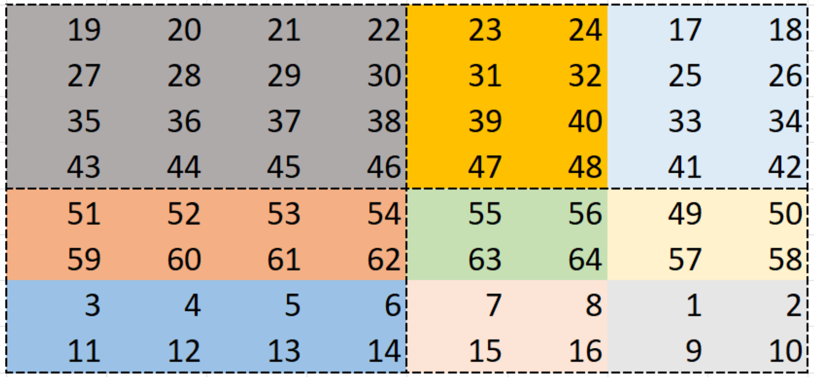
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

A: Attention in local windows



1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

B: Attention in shifted windows (2,2)



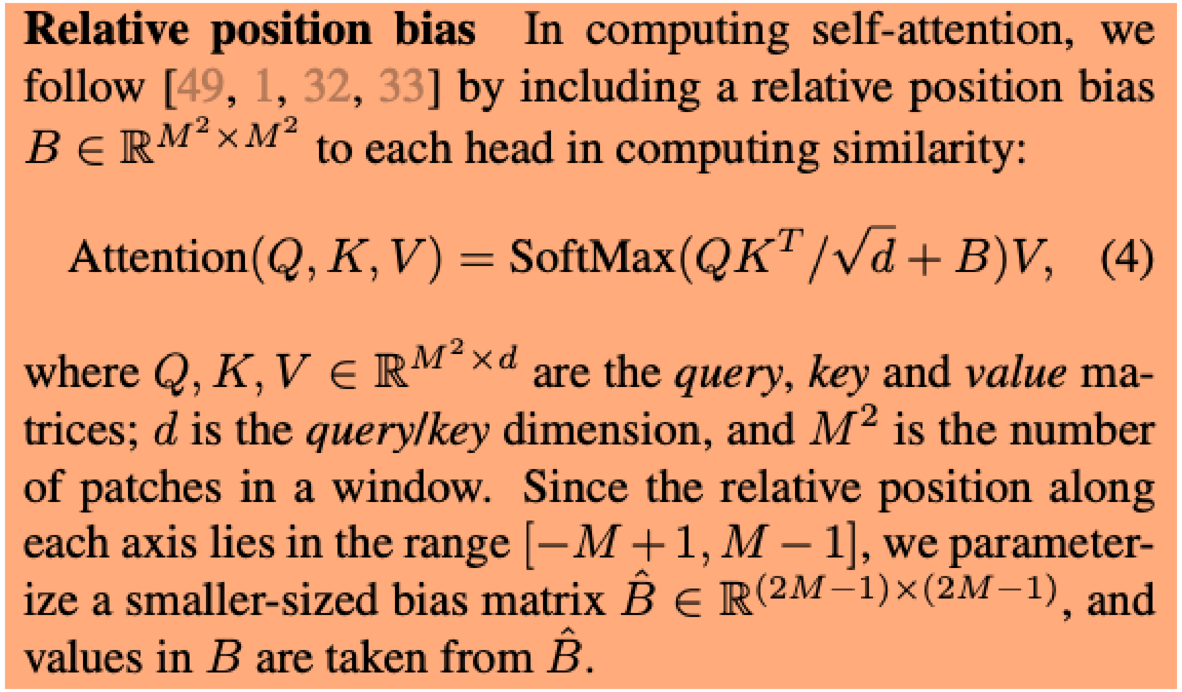
19	20	21	22	23	24	17	18
27	28	29	30	31	32	25	26
35	36	37	38	39	40	33	34
43	44	45	46	47	48	41	42
51	52	53	54	55	56	49	50
59	60	61	62	63	64	57	58
3	4	5	6	7	8	1	2
11	12	13	14	15	16	9	10

C: Batched windows after cyclic shift

Figure: https://amaarora.github.io/posts/2022-07-04-swintransformerv1.html

Previously on CENG501

Swin Transformer v1

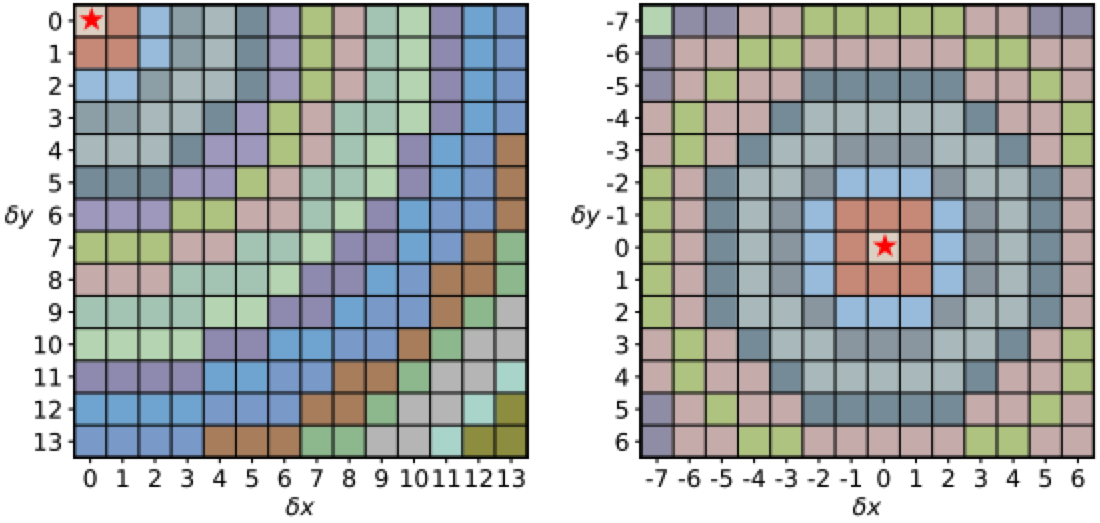


Relative position bias In computing self-attention, we follow [49, 1, 32, 33] by including a relative position bias $B \in \mathbb{R}^{M^2 \times M^2}$ to each head in computing similarity:

$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + B)V, \quad (4)$$

where $Q, K, V \in \mathbb{R}^{M^2 \times d}$ are the *query*, *key* and *value* matrices; d is the *query*/*key* dimension, and M^2 is the number of patches in a window. Since the relative position along each axis lies in the range $[-M+1, M-1]$, we parameterize a smaller-sized bias matrix $\hat{B} \in \mathbb{R}^{(2M-1)\times(2M-1)}$, and values in B are taken from $\hat{B}$.

Plot from https://arxiv.org/pdf/2107.14222:



(a) top-left (b) center

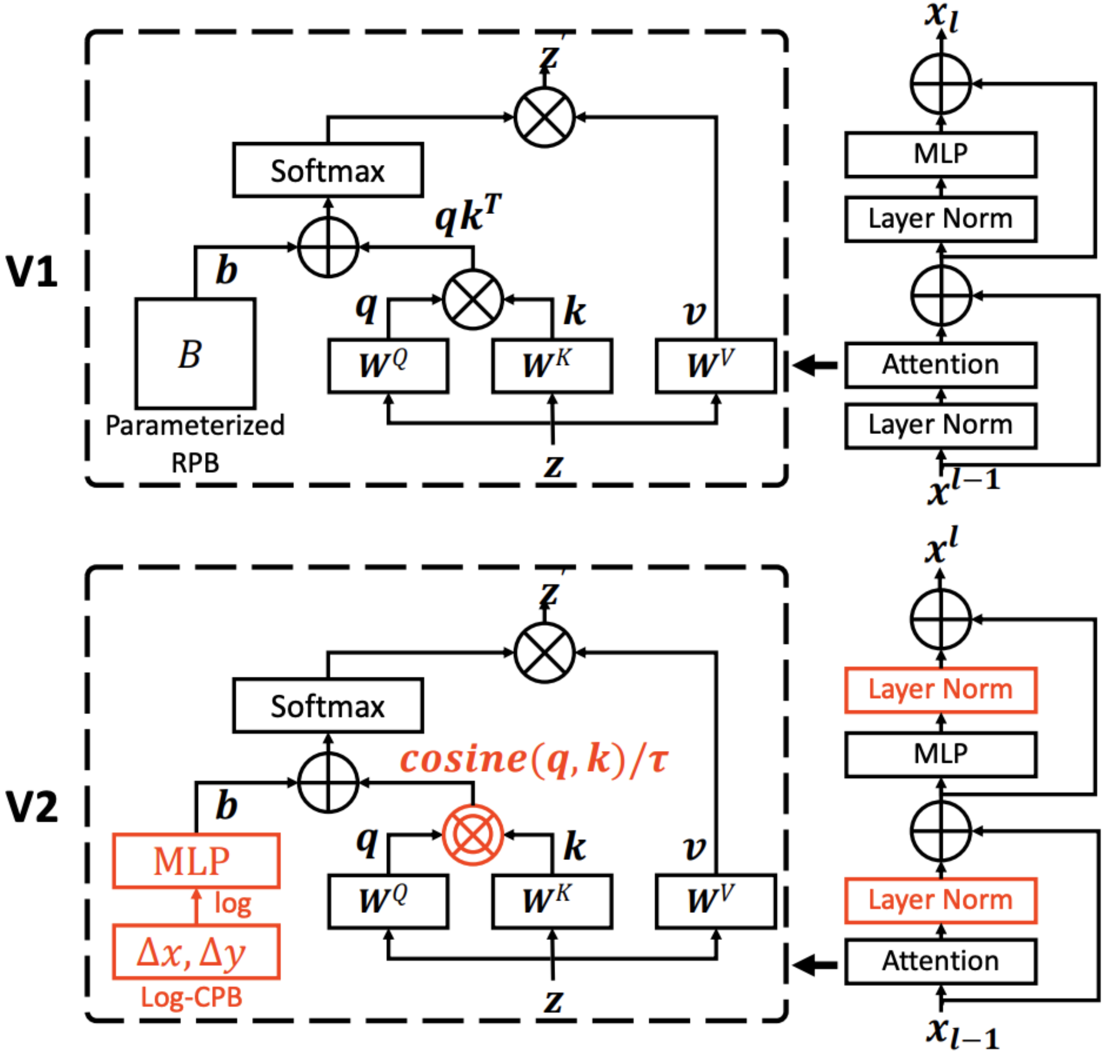
Figure 6: Visualization of Euclidean method. The red star ★ presents the reference position. Different color means different bucket. The relative positions with the same color share the same encoding.

Previously on CENG501

Swin Transformer v2

Scaling up model capacity

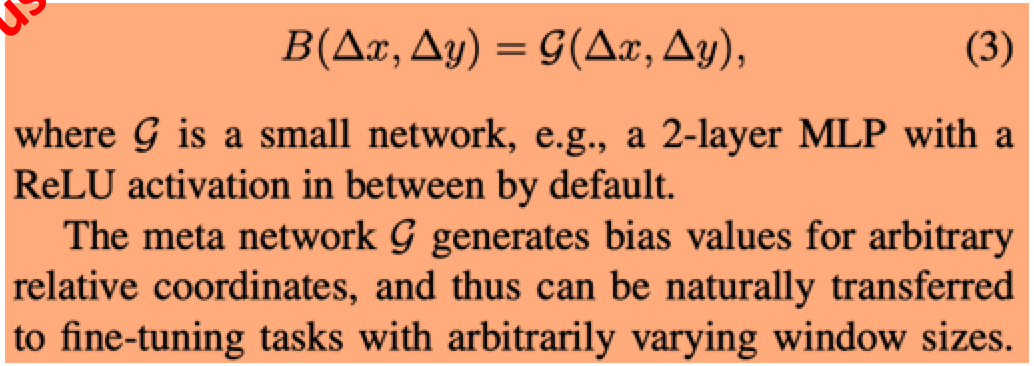
- Residual post normalization: Prevents output to diverge
- Scaled cosine attention:
  - learnt attention maps of some blocks and heads are frequently dominated by a few pixel pairs, especially in the res-post-norm configuration => cosine yields values in a smaller range



Scaling Up Window Resolution

Previously on CENG501

- Continuous relative position bias

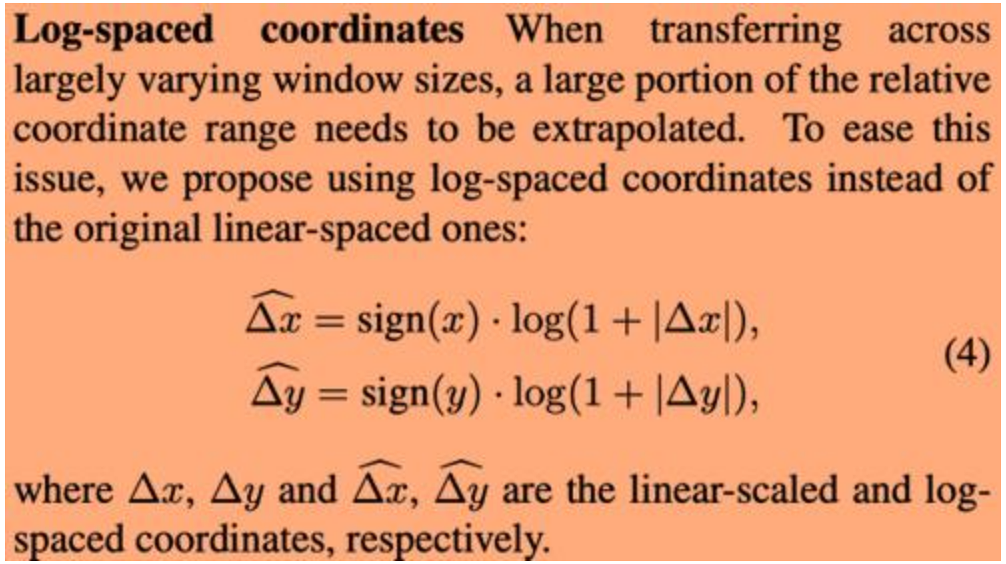


$$B(\Delta x, \Delta y) = \mathcal{G}(\Delta x, \Delta y), \quad (3)$$

where $\mathcal{G}$ is a small network, e.g., a 2-layer MLP with a ReLU activation in between by default.

The meta network $\mathcal{G}$ generates bias values for arbitrary relative coordinates, and thus can be naturally transferred to fine-tuning tasks with arbitrarily varying window sizes.

Log-spaced coordinates

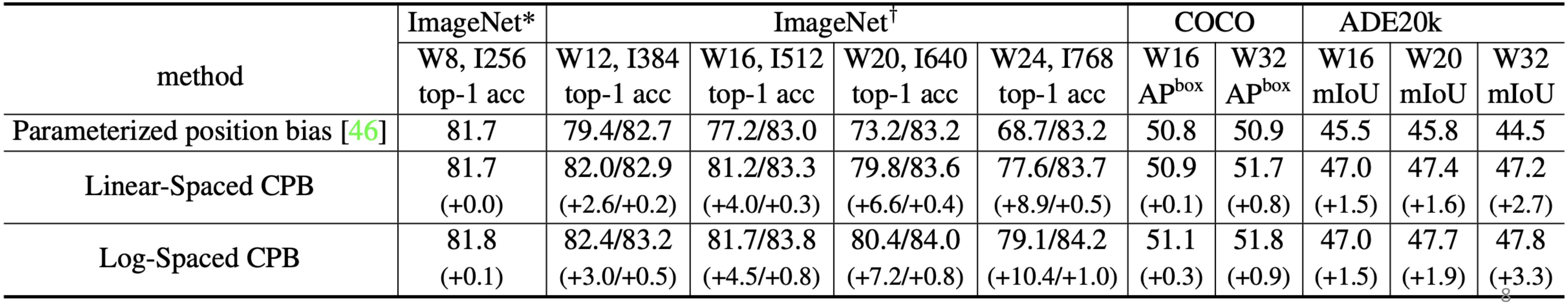


Log-spaced coordinates When transferring across largely varying window sizes, a large portion of the relative coordinate range needs to be extrapolated. To ease this issue, we propose using log-spaced coordinates instead of the original linear-spaced ones:

$$\begin{aligned} \widehat{\Delta x} &= \text{sign}(x) \cdot \log(1 + |\Delta x|), \\ \widehat{\Delta y} &= \text{sign}(y) \cdot \log(1 + |\Delta y|), \end{aligned} \quad (4)$$

where Δx, Δy and $\widehat{\Delta x}$, $\widehat{\Delta y}$ are the linear-scaled and log-spaced coordinates, respectively.

Trained on — Tested/Finetuned on — Finetuned on



	ImageNet*	ImageNet†				COCO		ADE20k		
method	W8, I256 top-1 acc	W12, I384 top-1 acc	W16, I512 top-1 acc	W20, I640 top-1 acc	W24, I768 top-1 acc	W16 AP^{box}	W32 AP^{box}	W16 mIoU	W20 mIoU	W32 mIoU
Parameterized position bias [46]	81.7	79.4/82.7	77.2/83.0	73.2/83.2	68.7/83.2	50.8	50.9	45.5	45.8	44.5
Linear-Spaced CPB	81.7 (+0.0)	82.0/82.9 (+2.6/+0.2)	81.2/83.3 (+4.0/+0.3)	79.8/83.6 (+6.6/+0.4)	77.6/83.7 (+8.9/+0.5)	50.9 (+0.1)	51.7 (+0.8)	47.0 (+1.5)	47.4 (+1.6)	47.2 (+2.7)
Log-Spaced CPB	81.8 (+0.1)	82.4/83.2 (+3.0/+0.5)	81.7/83.8 (+4.5/+0.8)	80.4/84.0 (+7.2/+0.8)	79.1/84.2 (+10.4/+1.0)	51.1 (+0.3)	51.8 (+0.9)	47.0 (+1.5)	47.7 (+1.9)	47.8 (+3.3)

Previously on CENG501

Swin Transformer v2

Self-Supervised Pre-training

- Prior work used JFT-3B
- Use SimMIM to train 3B-size Swin Transformer

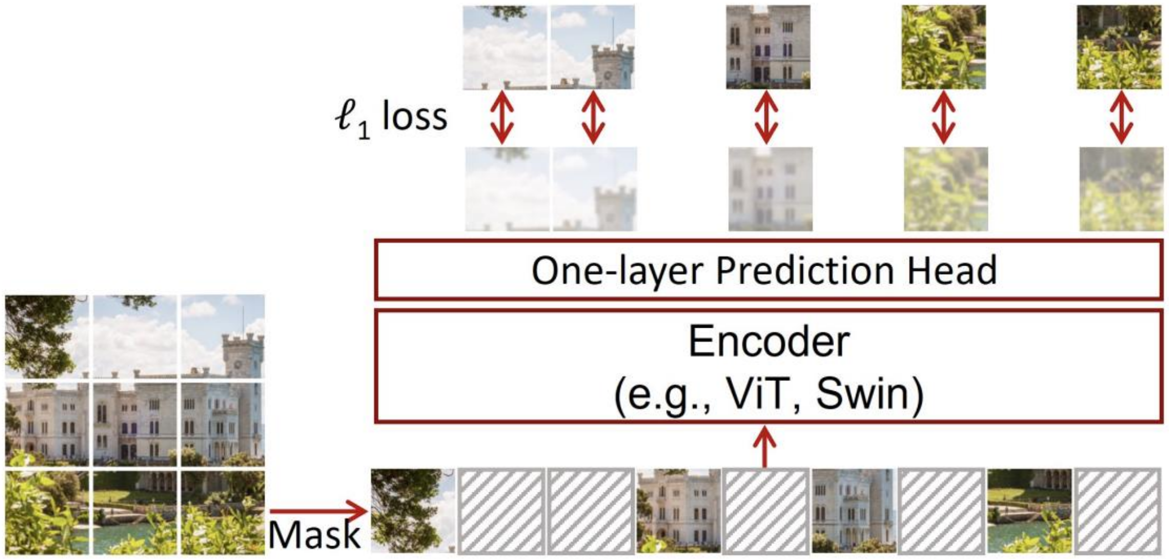


Figure 1. An illustration of our simple framework for masked language modeling, named *SimMIM*. It predicts raw pixel values of the randomly masked patches by a lightweight one-layer head, and performs learning using a simple ℓ_1 loss.

From SimMIM paper.

Previously on CENG501

ConvNeXt v1

- “Modernized” ResNet

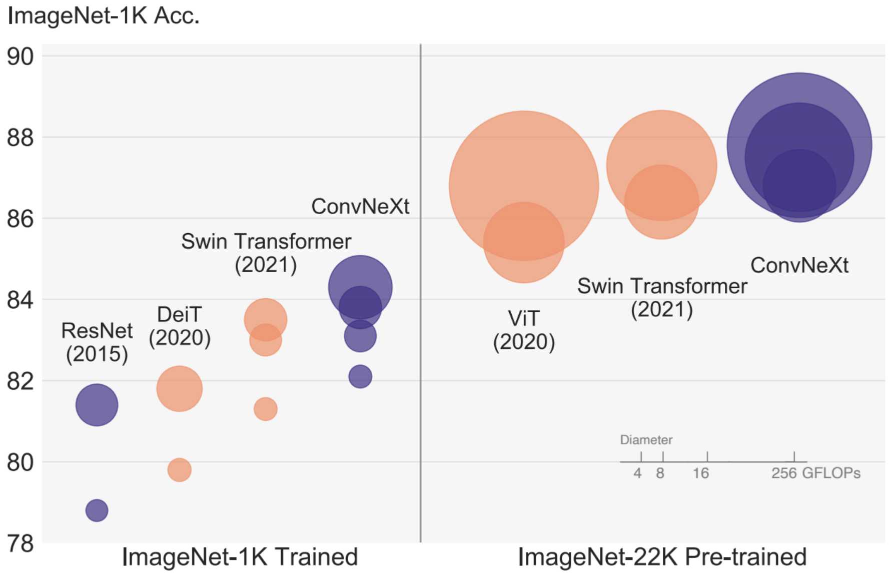
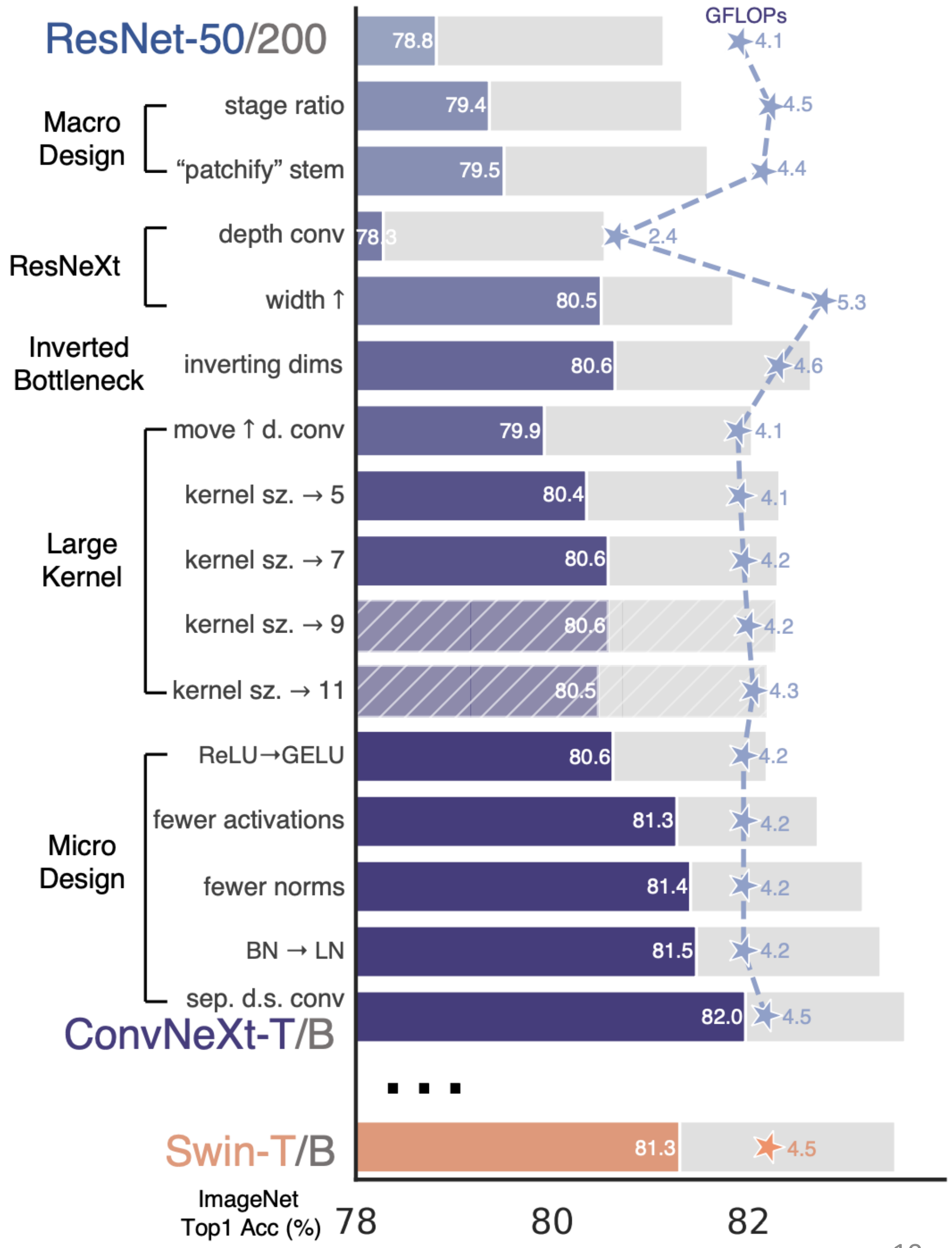


Figure 1. **ImageNet-1K classification** results for ● ConvNets and ○ vision Transformers. Each bubble’s area is proportional to FLOPs of a variant in a model family. ImageNet-1K/22K models here take $224^2/384^2$ images respectively. ResNet and ViT results were obtained with improved training procedures over the original papers. We demonstrate that a standard ConvNet model can achieve the same level of scalability as hierarchical vision Transformers while being much simpler in design.



Previously on CENG501

ConvNeXt v2

- Training ConvNeXt v1 with Masked AE performs poorly
- Introduce
  - fully convolutional masked autoencoder framework and
  - a new Global Response Normalization to enhance inter-channel competition

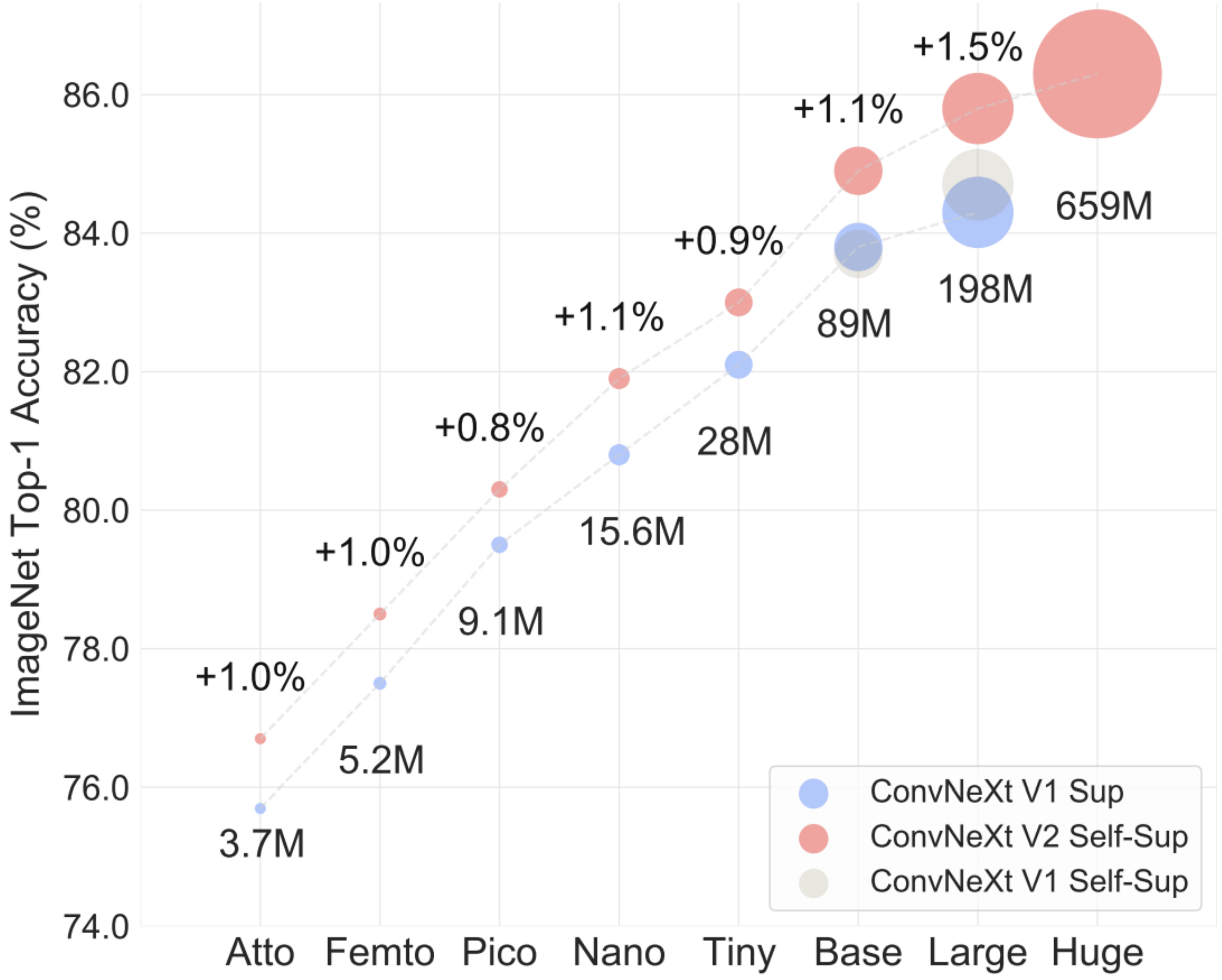


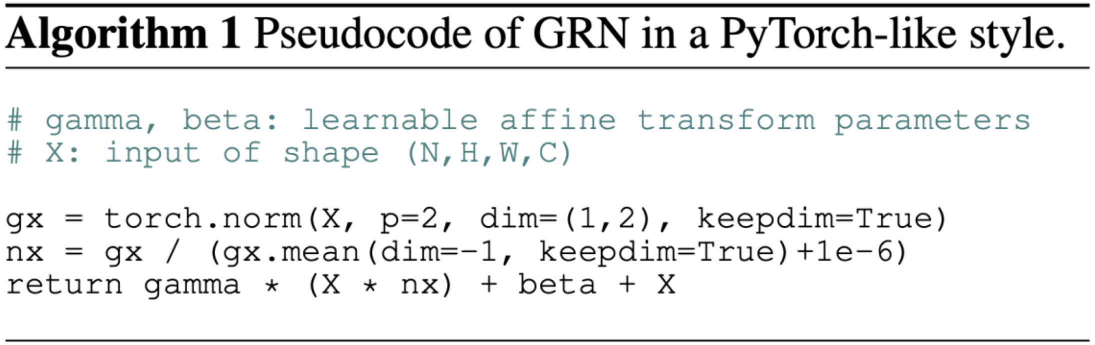
Figure 1. **ConvNeXt V2 model scaling.** The ConvNeXt V2 model, which has been pre-trained using our fully convolutional masked autoencoder framework, performs significantly better than the previous version across a wide range of model sizes.

Previously on CENG501

ConvNeXt v2

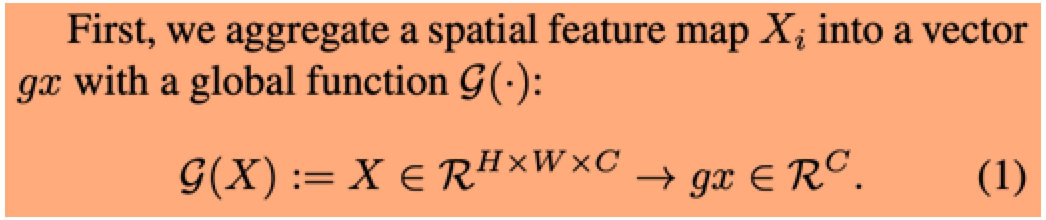
Global Response Normalization

- Inspiration from lateral inhibition in brain
- Three steps:
  1. global feature aggregation,
  2. feature normalization, and
  3. feature calibration.



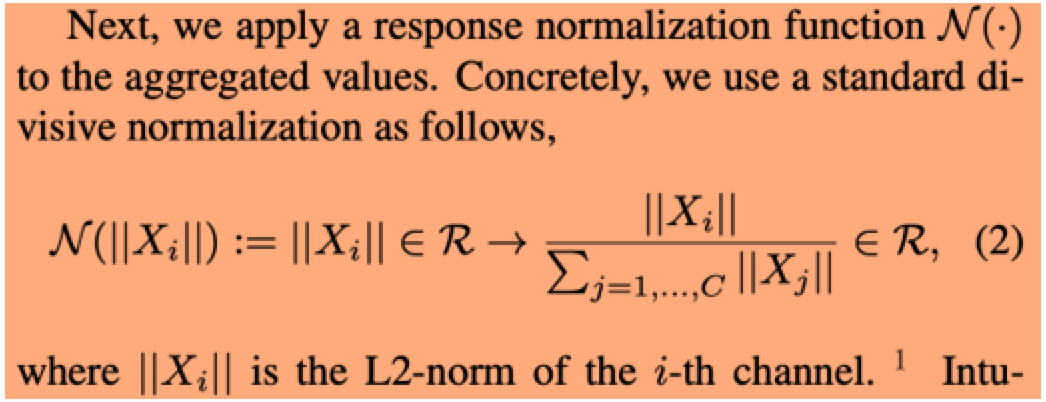
Algorithm 1 Pseudocode of GRN in a PyTorch-like style.

```
# gamma, beta: learnable affine transform parameters
# X: input of shape (N,H,W,C)

gx = torch.norm(X, p=2, dim=(1,2), keepdim=True)
nx = gx / (gx.mean(dim=-1, keepdim=True)+1e-6)
return gamma * (X * nx) + beta + X
```



First, we aggregate a spatial feature map X_i into a vector gx with a global function $\mathcal{G}(\cdot)$:

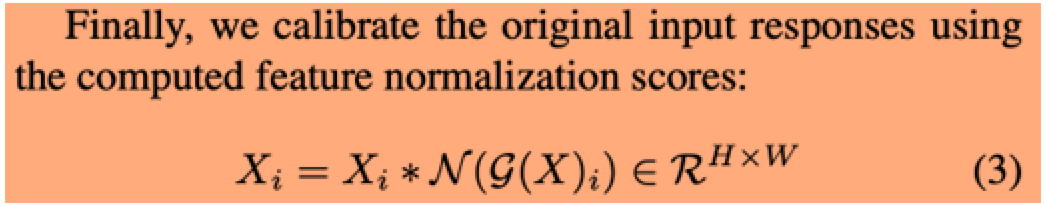
$$\mathcal{G}(X) := X \in \mathcal{R}^{H\times W\times C} \rightarrow gx \in \mathcal{R}^{C}. \quad (1)$$



Next, we apply a response normalization function $\mathcal{N}(\cdot)$ to the aggregated values. Concretely, we use a standard divisive normalization as follows,

$$\mathcal{N}(||X_i||) := ||X_i|| \in \mathcal{R} \rightarrow \frac{||X_i||}{\sum_{j=1,\ldots,C} ||X_j||} \in \mathcal{R}, \quad (2)$$

where $||X_i||$ is the L2-norm of the i-th channel. [1] Intu-



Finally, we calibrate the original input responses using the computed feature normalization scores:

$$X_i = X_i * \mathcal{N}(\mathcal{G}(X)_i) \in \mathcal{R}^{H\times W} \quad (3)$$

Previously on CENG501

FastViT

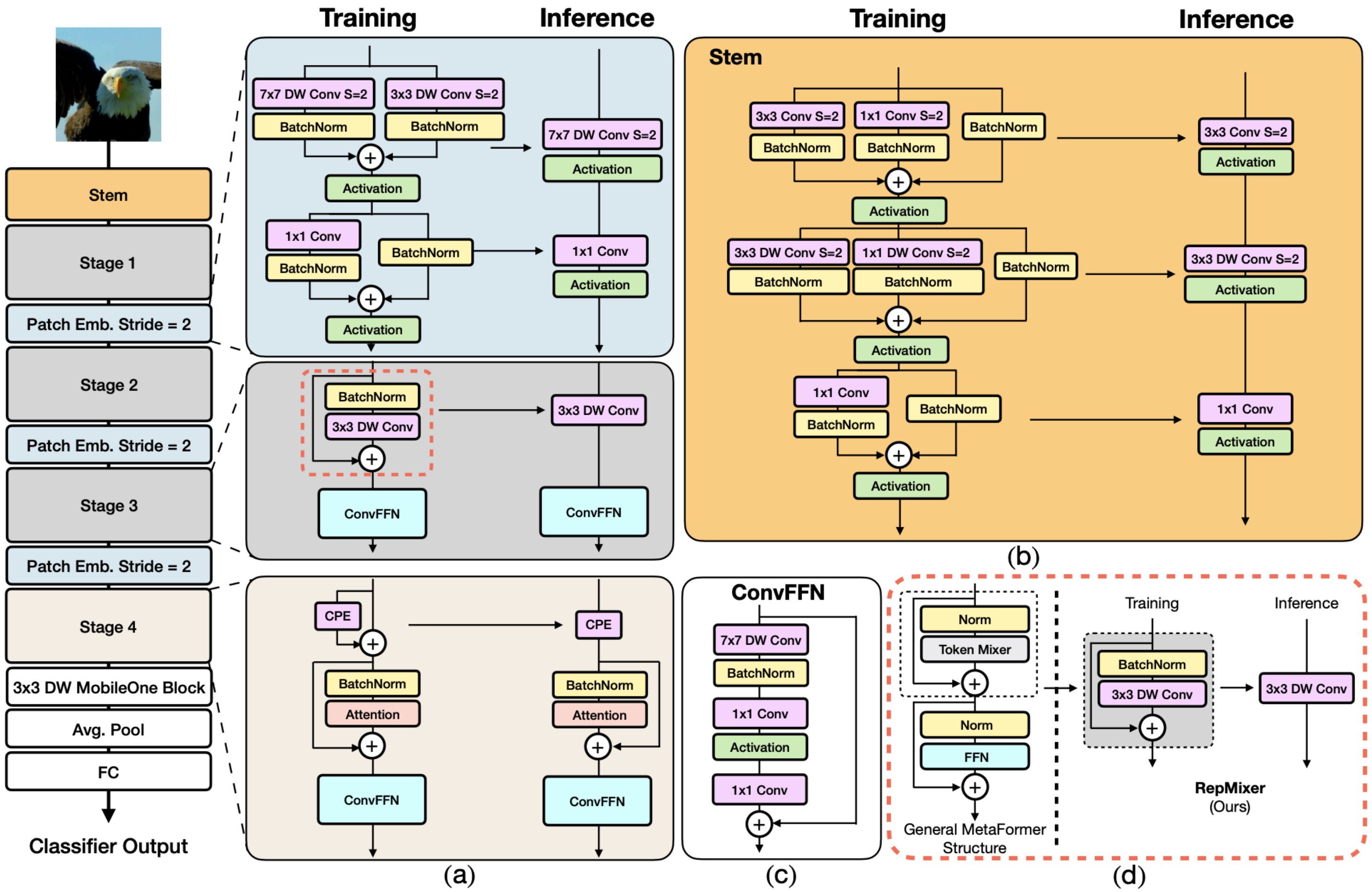
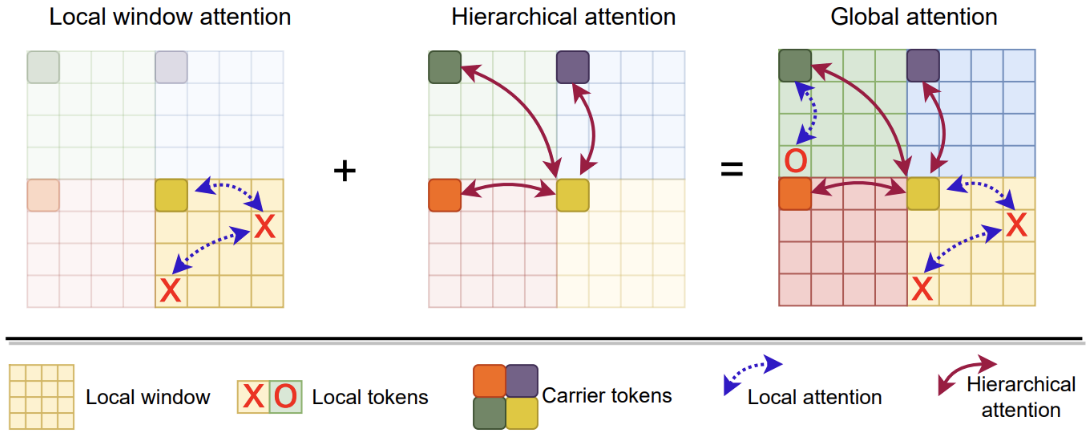


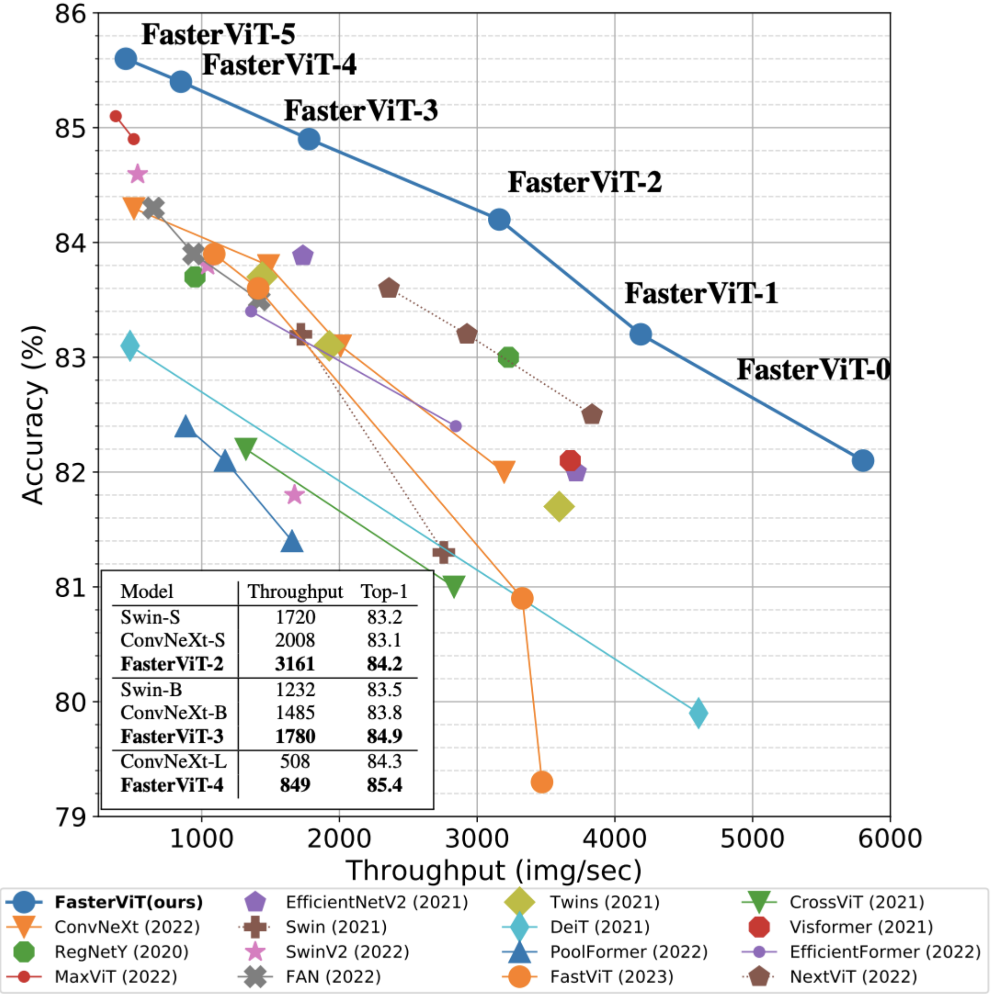
Figure 2: (a) Overview of FastViT architecture which decouples train-time and inference-time architecture. Stages 1, 2, and 3 have the same architecture and uses RepMixer for token mixing. In stage 4, self attention layers are used for token mixing. (b) Architecture of the convolutional stem. (c) Architecture of convolutional-FFN (d) Overview of *RepMixer* block, which reparameterizes a skip connection at inference.

Faster ViT

Previously on CENG501

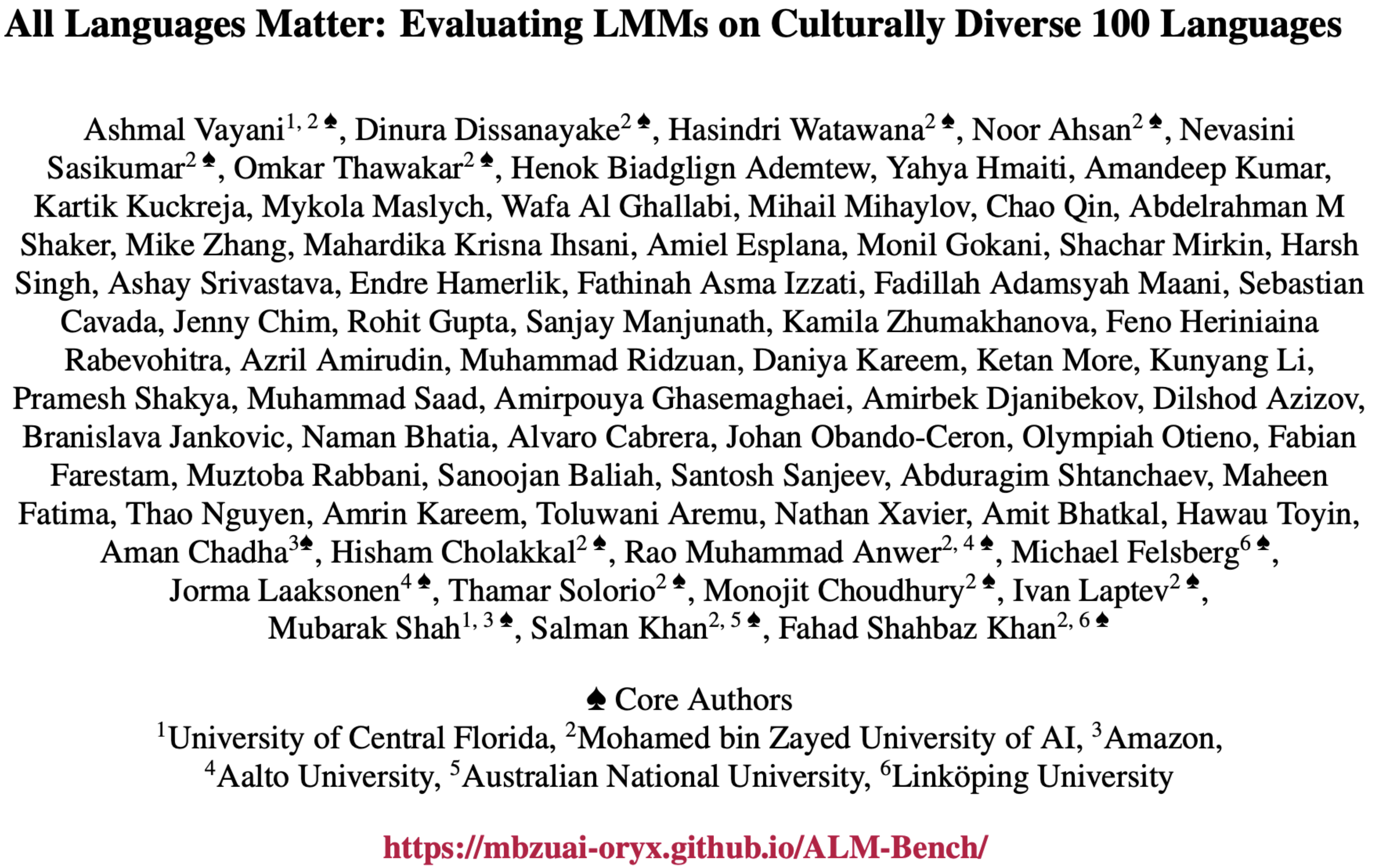
Figure 2: Visualization of the proposed Hierarchical Attention in the feature space. By performing local window attention and hierarchical attention we can achieve global information propagation at reduced costs.





Model	Throughput	Top-1
Swin-S	1720	83.2
ConvNeXt-S	2008	83.1
FasterViT-2	**3161**	**84.2**
Swin-B	1232	83.5
ConvNeXt-B	1485	83.8
FasterViT-3	**1780**	**84.9**
ConvNeXt-L	508	84.3
FasterViT-4	**849**	**85.4**

Figure 1: Comparison of image throughput and ImageNet-1K Top-1 accuracy. Throughput is measured on A100 GPU with batch size of 128.



All Languages Matter: Evaluating LMMs on Culturally Diverse 100 Languages

Ashmal Vayani[1, 2]♠, Dinura Dissanayake[2]♠, Hasindri Watawana[2]♠, Noor Ahsan[2]♠, Nevasini Sasikumar[2]♠, Omkar Thawakar[2]♠, Henok Biadglign Ademtew, Yahya Hmaiti, Amandeep Kumar, Kartik Kuckreja, Mykola Maslych, Wafa Al Ghallabi, Mihail Mihaylov, Chao Qin, Abdelrahman M Shaker, Mike Zhang, Mahardika Krisna Ihsani, Amiel Esplana, Monil Gokani, Shachar Mirkin, Harsh Singh, Ashay Srivastava, Endre Hamerlik, Fathinah Asma Izzati, Fadillah Adamsyah Maani, Sebastian Cavada, Jenny Chim, Rohit Gupta, Sanjay Manjunath, Kamila Zhumakhanova, Feno Heriniaina Rabevohitra, Azril Amirudin, Muhammad Ridzuan, Daniya Kareem, Ketan More, Kunyang Li, Pramesh Shakya, Muhammad Saad, Amirpouya Ghasemaghaei, Amirbek Djanibekov, Dilshod Azizov, Branislava Jankovic, Naman Bhatia, Alvaro Cabrera, Johan Obando-Ceron, Olympiah Otieno, Fabian Farestam, Muztoba Rabbani, Sanoojan Baliah, Santosh Sanjeev, Abduragim Shtanchaev, Maheen Fatima, Thao Nguyen, Amrin Kareem, Toluwani Aremu, Nathan Xavier, Amit Bhatkal, Hawau Toyin, Aman Chadha[3]♠, Hisham Cholakkal[2]♠, Rao Muhammad Anwer[2, 4]♠, Michael Felsberg[6]♠, Jorma Laaksonen[4]♠, Thamar Solorio[2]♠, Monojit Choudhury[2]♠, Ivan Laptev[2]♠, Mubarak Shah[1, 3]♠, Salman Khan[2, 5]♠, Fahad Shahbaz Khan[2, 6]♠

♠ Core Authors
[1]University of Central Florida, [2]Mohamed bin Zayed University of AI, [3]Amazon,
[4]Aalto University, [5]Australian National University, [6]Linköping University

https://mbzuai-oryx.github.io/ALM-Bench/

https://arxiv.org/pdf/2411.16508

Today

- Vision Models
  - Pretraining
- Vision-Language Models

Administrative Notes

- No quiz this week

- Time plan for the projects
  1. Milestone (November 24, midnight):
     - Github repo will be ready
     - Read & understand the paper
     - Download the datasets
     - Prepare the Readme file excluding the results & conclusion
  2. Milestone (December 8, midnight)
     - The results of the first experiment
  3. Milestone (January 5, midnight)
     - Final report (Readme file)
     - Repo with all code & trained models

Pretraining Vision Transformers

Masked Autoencoders

Masked Autoencoders Are Scalable Vision Learners

Kaiming He*,† Xinlei Chen* Saining Xie Yanghao Li Piotr Dollár Ross Girshick
*equal technical contribution †project lead
Facebook AI Research (FAIR)

CVPR 2022

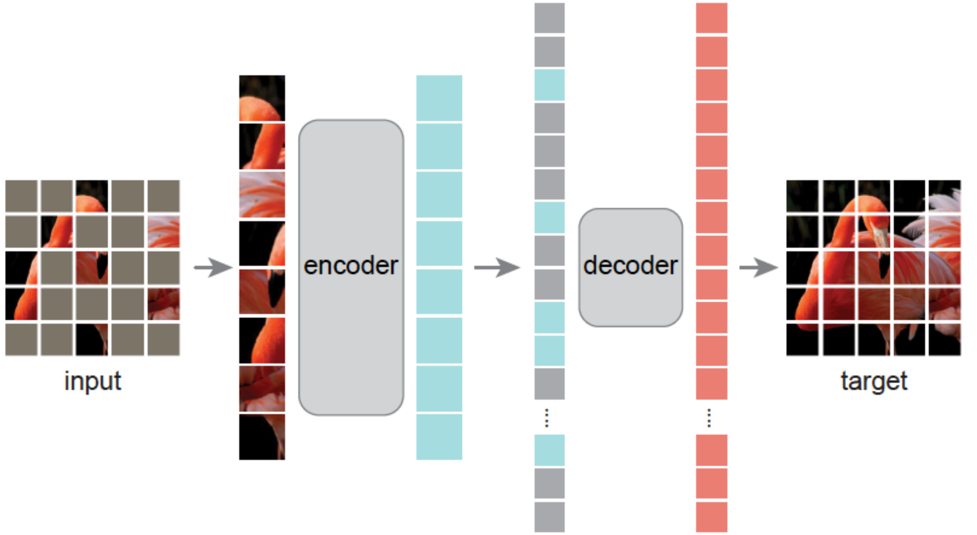


Figure 1. **Our MAE architecture.** During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

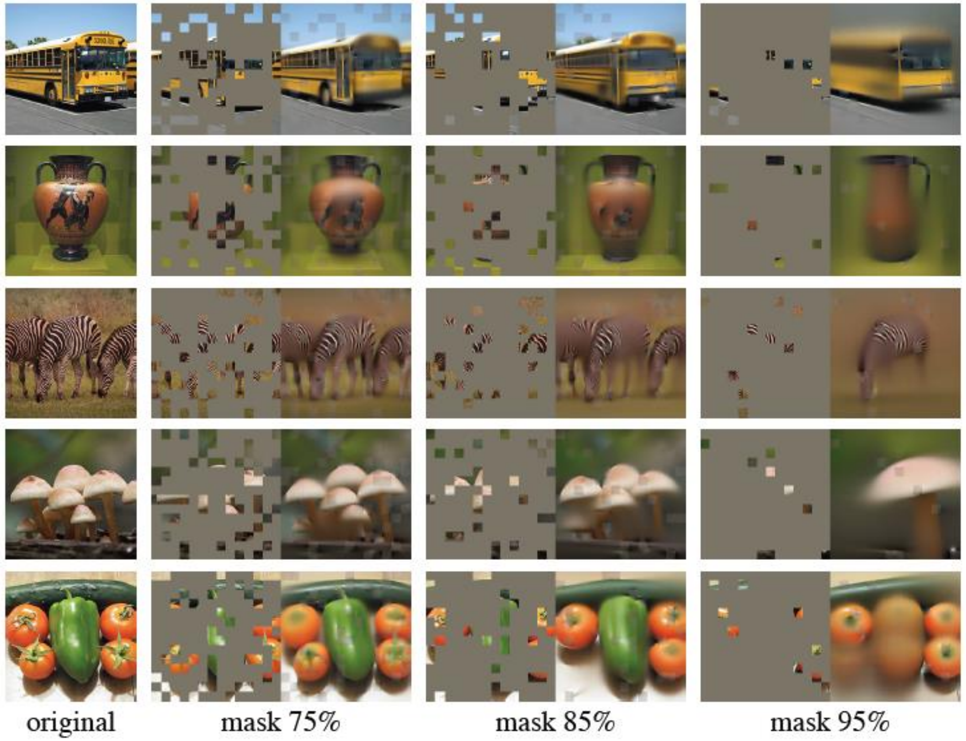
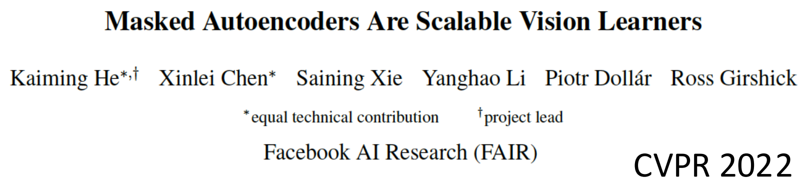


Figure 4. Reconstructions of ImageNet *validation* images using an MAE pre-trained with a masking ratio of 75% but applied on inputs with higher masking ratios. The predictions differ plausibly from the original images, showing that the method can generalize.

MSE loss between pixels for masked tokens only!

Masked Autoencoders



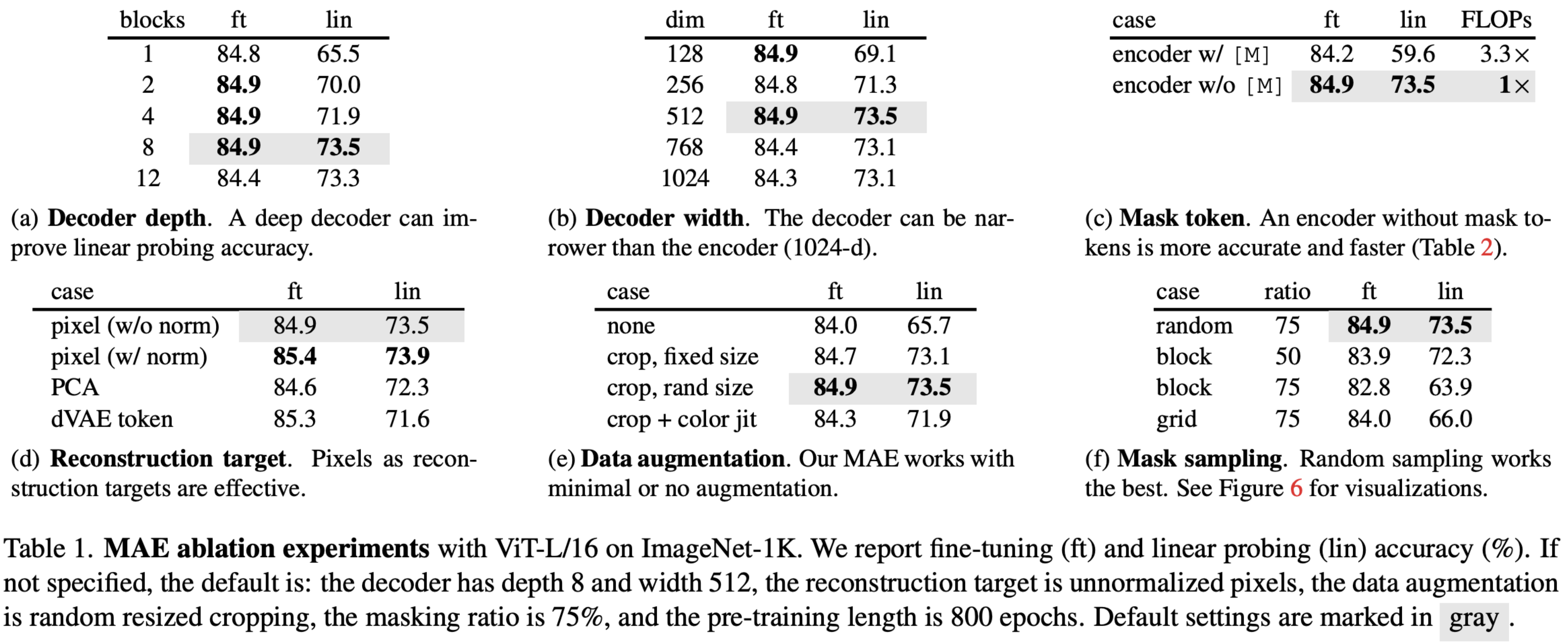
Masked Autoencoders Are Scalable Vision Learners

Kaiming He*,† Xinlei Chen* Saining Xie Yanghao Li Piotr Dollár Ross Girshick

*equal technical contribution †project lead

Facebook AI Research (FAIR)

CVPR 2022



blocks	ft	lin
1	84.8	65.5
2	**84.9**	70.0
4	**84.9**	71.9
8	**84.9**	**73.5**
12	84.4	73.3

(a) **Decoder depth**. A deep decoder can improve linear probing accuracy.

dim	ft	lin
128	**84.9**	69.1
256	84.8	71.3
512	**84.9**	**73.5**
768	84.4	73.1
1024	84.3	73.1

(b) **Decoder width**. The decoder can be narrower than the encoder (1024-d).

case	ft	lin	FLOPs
encoder w/ `[M]`	84.2	59.6	3.3×
encoder w/o `[M]`	**84.9**	**73.5**	**1×**

(c) **Mask token**. An encoder without mask tokens is more accurate and faster (Table 2).

case	ft	lin
pixel (w/o norm)	84.9	73.5
pixel (w/ norm)	**85.4**	**73.9**
PCA	84.6	72.3
dVAE token	85.3	71.6

(d) **Reconstruction target**. Pixels as reconstruction targets are effective.

case	ft	lin
none	84.0	65.7
crop, fixed size	84.7	73.1
crop, rand size	**84.9**	**73.5**
crop + color jit	84.3	71.9

(e) **Data augmentation**. Our MAE works with minimal or no augmentation.

case	ratio	ft	lin
random	75	**84.9**	**73.5**
block	50	83.9	72.3
block	75	82.8	63.9
grid	75	84.0	66.0

(f) **Mask sampling**. Random sampling works the best. See Figure 6 for visualizations.

Table 1. **MAE ablation experiments** with ViT-L/16 on ImageNet-1K. We report fine-tuning (ft) and linear probing (lin) accuracy (%). If not specified, the default is: the decoder has depth 8 and width 512, the reconstruction target is unnormalized pixels, the data augmentation is random resized cropping, the masking ratio is 75%, and the pre-training length is 800 epochs. Default settings are marked in gray .

Masked Autoencoders

Masked Autoencoders Are Scalable Vision Learners

Kaiming He[*,†] Xinlei Chen[*] Saining Xie Yanghao Li Piotr Dollár Ross Girshick
[*] equal technical contribution [†] project lead
Facebook AI Research (FAIR)

CVPR 2022

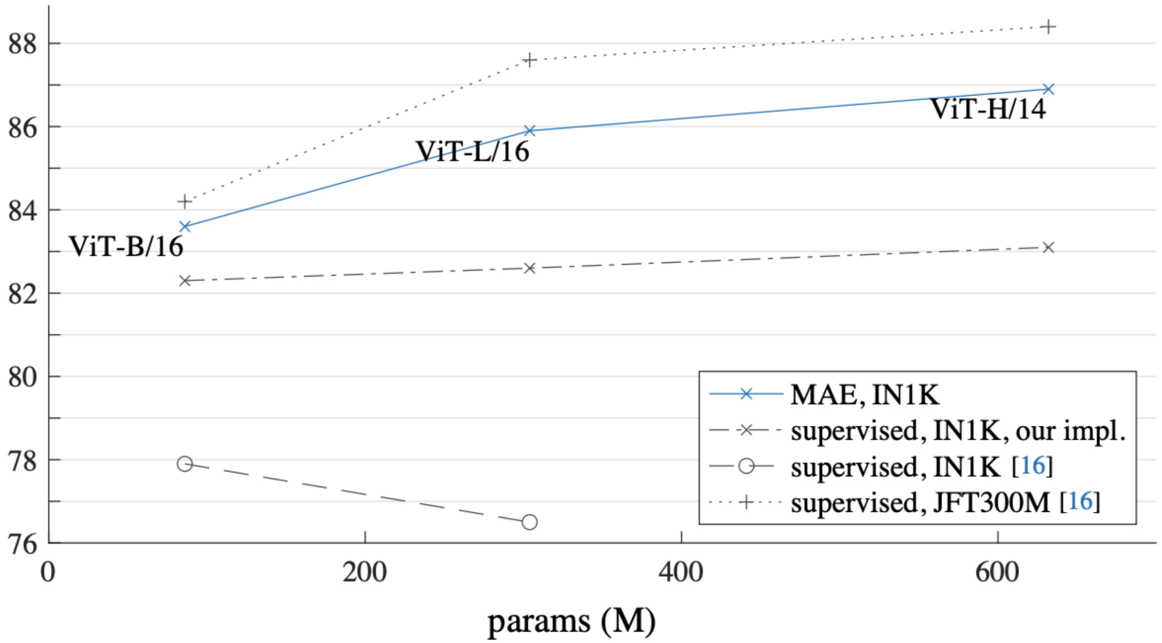
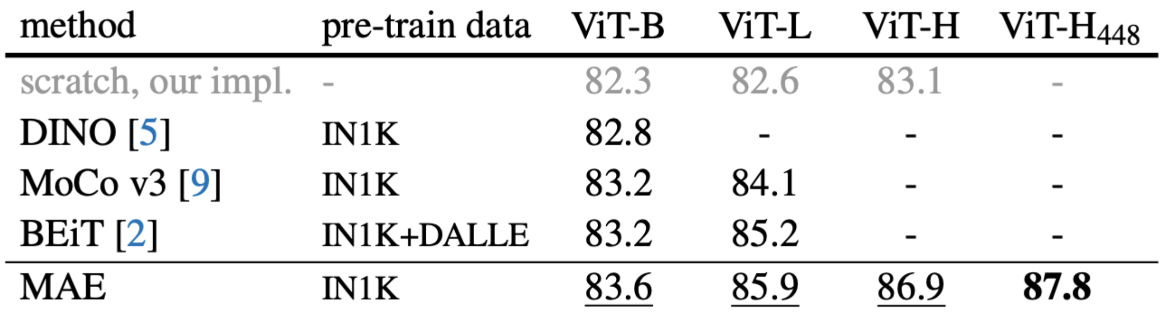


Figure 8. **MAE pre-training *vs.* supervised pre-training**, evaluated by fine-tuning in ImageNet-1K (224 size). We compare with the original ViT results [16] trained in IN1K or JFT300M.



method	pre-train data	ViT-B	ViT-L	ViT-H	ViT-H$_{448}$
scratch, our impl.	-	82.3	82.6	83.1	-
DINO [5]	IN1K	82.8	-	-	-
MoCo v3 [9]	IN1K	83.2	84.1	-	-
BEiT [2]	IN1K+DALLE	83.2	85.2	-	-
MAE	IN1K	83.6	85.9	86.9	**87.8**

Table 3. **Comparisons with previous results on ImageNet-1K**. The pre-training data is the ImageNet-1K training set (except the tokenizer in BEiT was pre-trained on 250M DALLE data [50]). All self-supervised methods are evaluated by end-to-end fine-tuning. The ViT models are B/16, L/16, H/14 [16]. The best for each column is underlined. All results are on an image size of 224, except for ViT-H with an extra result on 448. Here our MAE reconstructs normalized pixels and is pre-trained for 1600 epochs.

SimMIM

SimMIM: a Simple Framework for Masked Image Modeling

CVPR 2022

Zhenda Xie[1*] Zheng Zhang[2*] Yue Cao[2*]
Yutong Lin[3] Jianmin Bao[2] Zhuliang Yao[1] Qi Dai[2] Han Hu[2*]
[1]Tsinghua University [2]Microsoft Research Asia [3]Xi'an Jiaotong University
{t-zhxie,zhez,yuecao,t-yutonglin,jianmin.bao,t-zhuyao,qid,hanhu}@microsoft.com

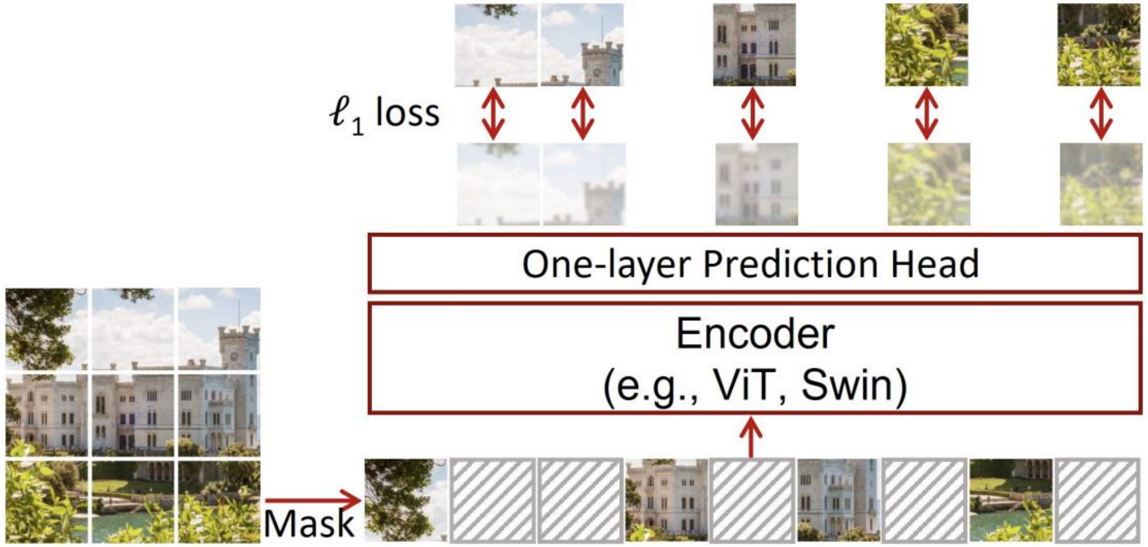
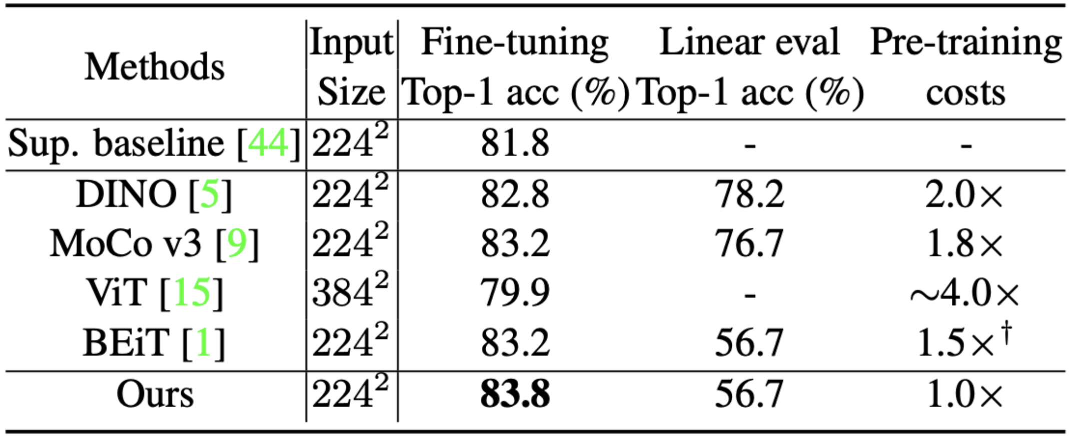


Figure 1. An illustration of our simple framework for masked language modeling, named *SimMIM*. It predicts raw pixel values of the randomly masked patches by a lightweight one-layer head, and performs learning using a simple ℓ_1 loss.



Methods	Input Size	Fine-tuning Top-1 acc (%)	Linear eval Top-1 acc (%)	Pre-training costs
Sup. baseline [44]	224^2	81.8	-	-
DINO [5]	224^2	82.8	78.2	2.0×
MoCo v3 [9]	224^2	83.2	76.7	1.8×
ViT [15]	384^2	79.9	-	∼4.0×
BEiT [1]	224^2	83.2	56.7	1.5×†
Ours	224^2	**83.8**	56.7	1.0×

Table 6. System-level comparison using ViT-B as the encoder. Training costs are counted in relative to our approach. † BEiT requires an additional stage to pre-train dVAE, which is not counted.

DINO v1 & v2 (ICCV'21 & TMLR'24)

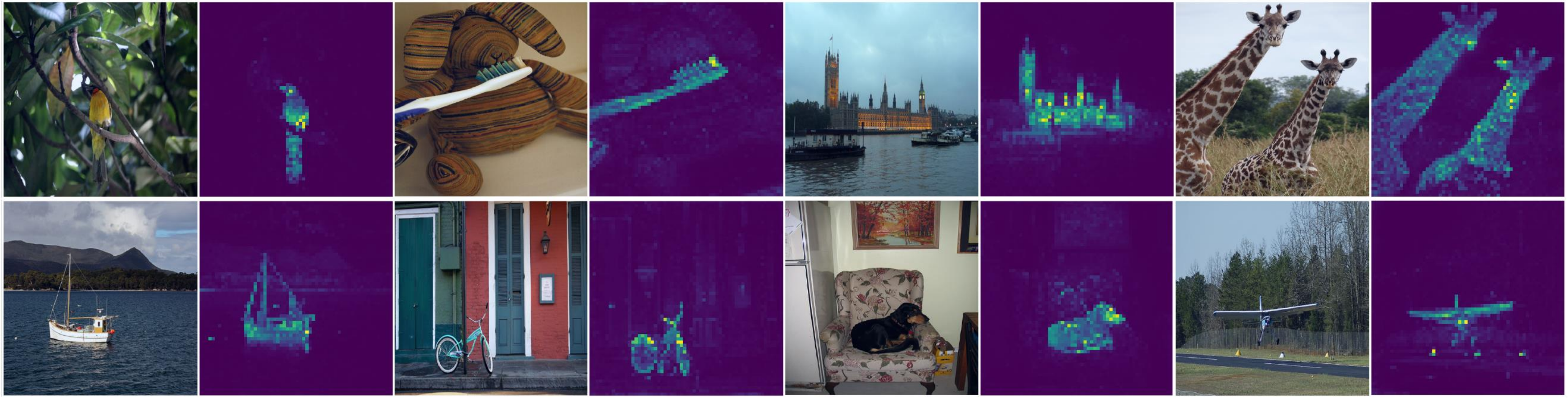


Figure 1: **Self-attention from a Vision Transformer with 8×8 patches trained with no supervision.** We look at the self-attention of the `[CLS]` token on the heads of the last layer. This token is not attached to any label nor supervision. These maps show that the model automatically learns class-specific features leading to unsupervised object segmentations.

DINO v1 & v2 (ICCV’21 & TMLR’24)

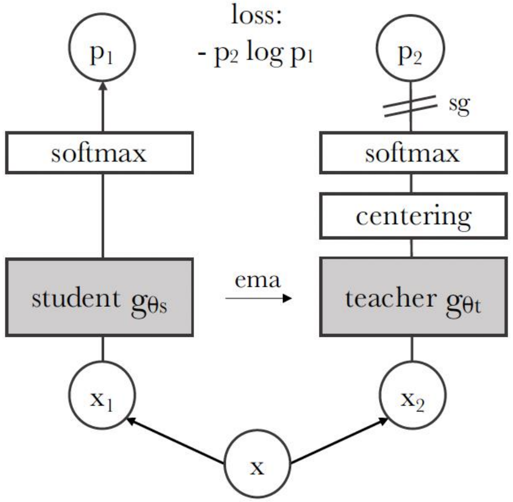
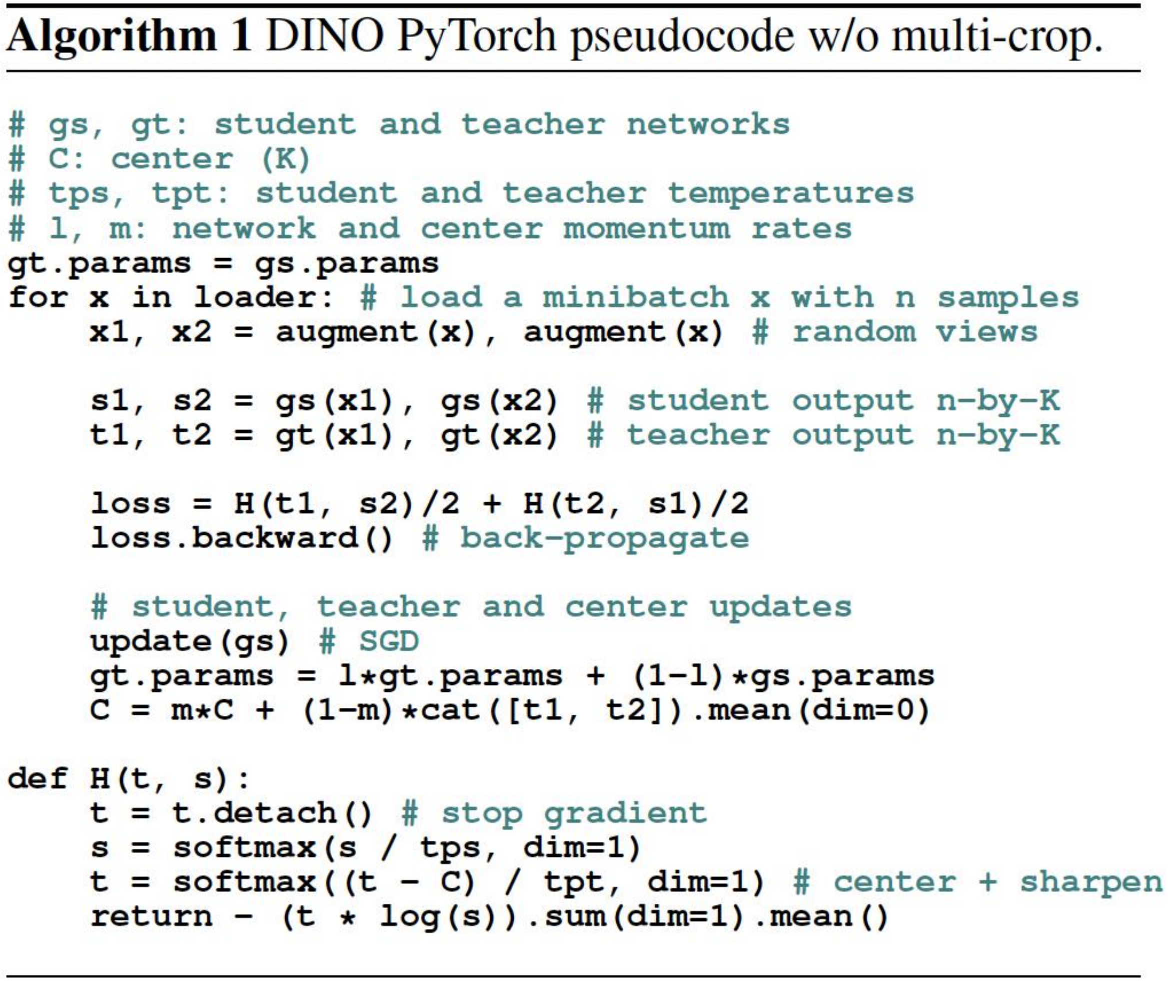


Figure 2: **Self-distillation with no labels.** We illustrate DINO in the case of one single pair of views (x_1, x_2) for simplicity. The model passes two different random transformations of an input image to the student and teacher networks. Both networks have the same architecture but different parameters. The output of the teacher network is centered with a mean computed over the batch. Each networks outputs a K dimensional feature that is normalized with a temperature softmax over the feature dimension. Their similarity is then measured with a cross-entropy loss. We apply a stop-gradient (sg) operator on the teacher to propagate gradients only through the student. The teacher parameters are updated with an exponential moving average (ema) of the student parameters.



Algorithm 1 DINO PyTorch pseudocode w/o multi-crop.

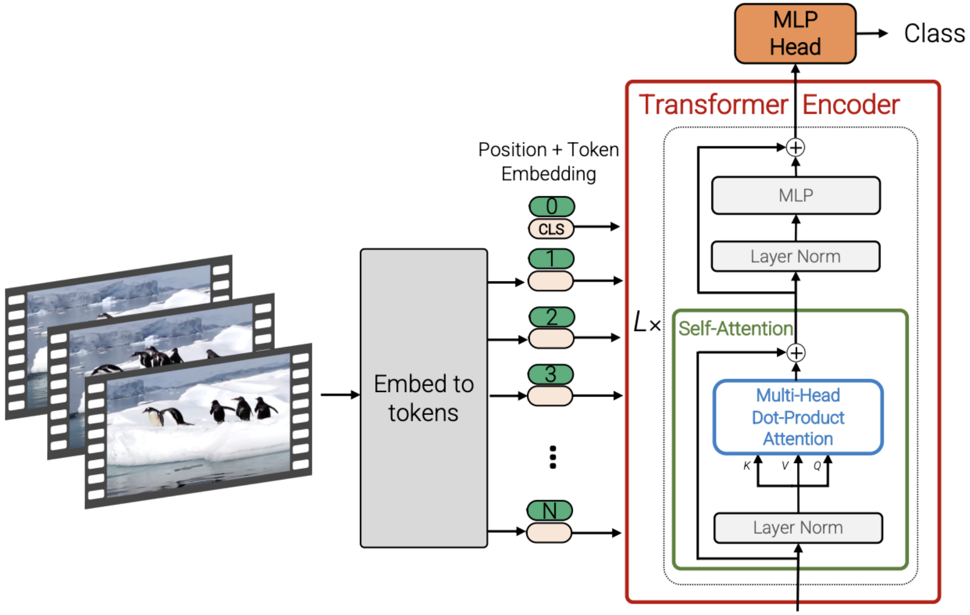
```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

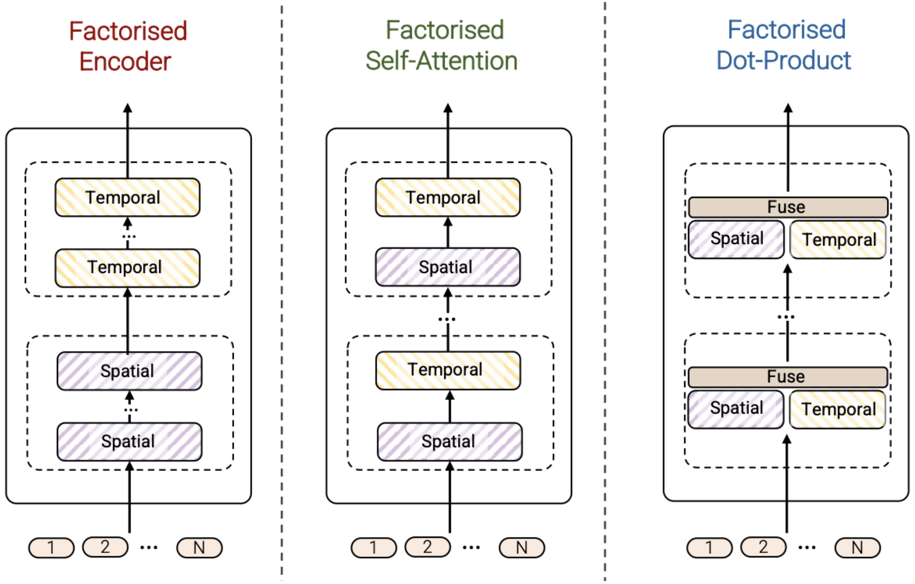
ViViT

ViViT: A Video Vision Transformer

Anurag Arnab[*] Mostafa Dehghani[*] Georg Heigold Chen Sun Mario Lučić[†] Cordelia Schmid[†]
Google Research
{aarnab, dehghani, heigold, chensun, lucic, cordelias}@google.com

ICCV 2021





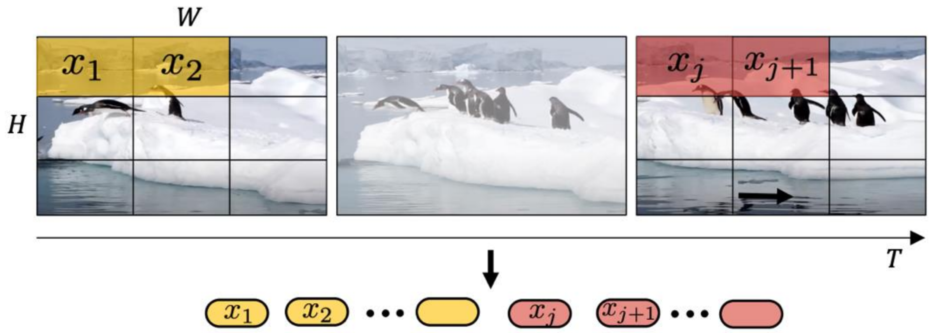


Figure 2: Uniform frame sampling: We simply sample n_t frames, and embed each 2D frame independently following ViT [18].

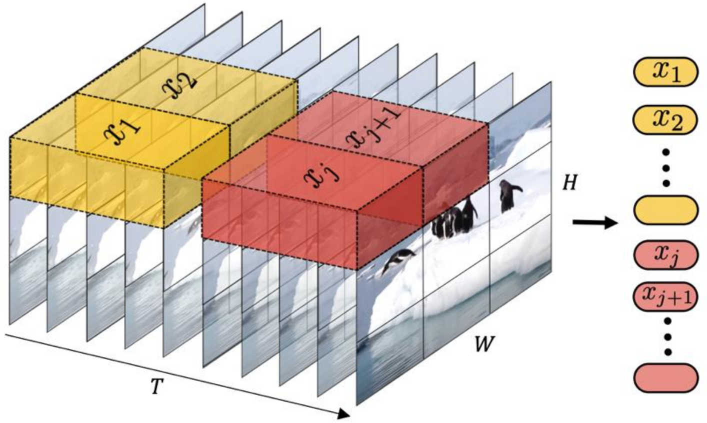


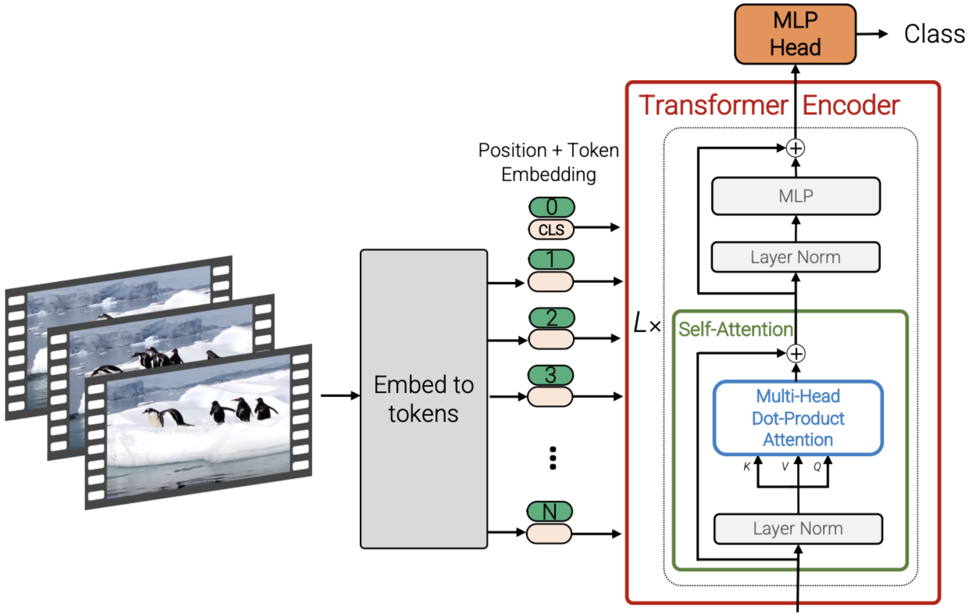
Figure 3: Tubelet embedding. We extract and linearly embed non-overlapping tubelets that span the spatio-temporal input volume.

ViViT

ViViT: A Video Vision Transformer

Anurag Arnab* Mostafa Dehghani* Georg Heigold Chen Sun Mario Lučić† Cordelia Schmid†
Google Research
{aarnab, dehghani, heigold, chensun, lucic, cordelias}@google.com

ICCV 2021



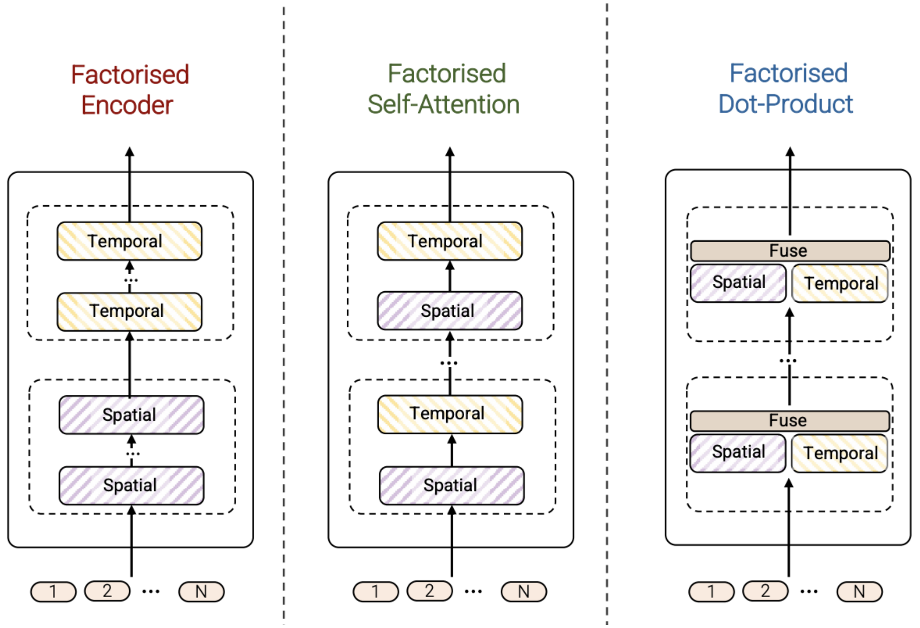


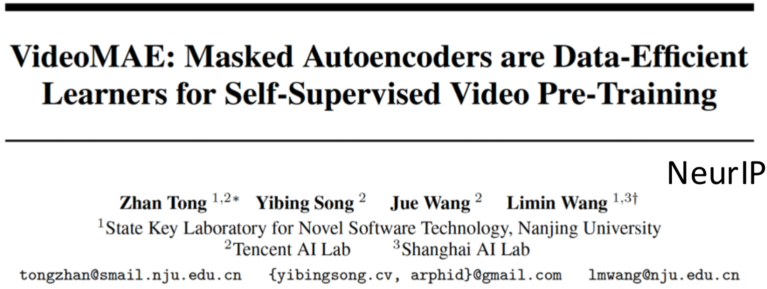
Table 1: Comparison of input encoding methods using ViViT-B and spatio-temporal attention on Kinetics. Further details in text.

	Top-1 accuracy
Uniform frame sampling	78.5
Tubelet embedding	
Random initialisation [25]	73.2
Filter inflation [8]	77.6
Central frame	79.2

Table 2: Comparison of model architectures using ViViT-B as the backbone, and tubelet size of 16 × 2. We report Top-1 accuracy on Kinetics 400 (K400) and action accuracy on Epic Kitchens (EK). Runtime is during inference on a TPU-v3.

	K400	EK	FLOPs ($\times 10^9$)	Params ($\times 10^6$)	Runtime (ms)
Model 1: Spatio-temporal	80.0	43.1	455.2	88.9	58.9
Model 2: Fact. encoder	78.8	43.7	284.4	115.1	17.4
Model 3: Fact. self-attention	77.4	39.1	372.3	117.3	31.7
Model 4: Fact. dot product	76.3	39.5	277.1	88.9	22.9
Model 2: Ave. pool baseline	75.8	38.8	283.9	86.7	17.3

VideoMAE



VideoMAE: Masked Autoencoders are Data-Efficient Learners for Self-Supervised Video Pre-Training

Zhan Tong [1,2*] **Yibing Song** [2] **Jue Wang** [2] **Limin Wang** [1,3†]
[1]State Key Laboratory for Novel Software Technology, Nanjing University
[2]Tencent AI Lab [3]Shanghai AI Lab
tongzhan@smail.nju.edu.cn {yibingsong.cv, arphid}@gmail.com lmwang@nju.edu.cn

NeurIPS 2022

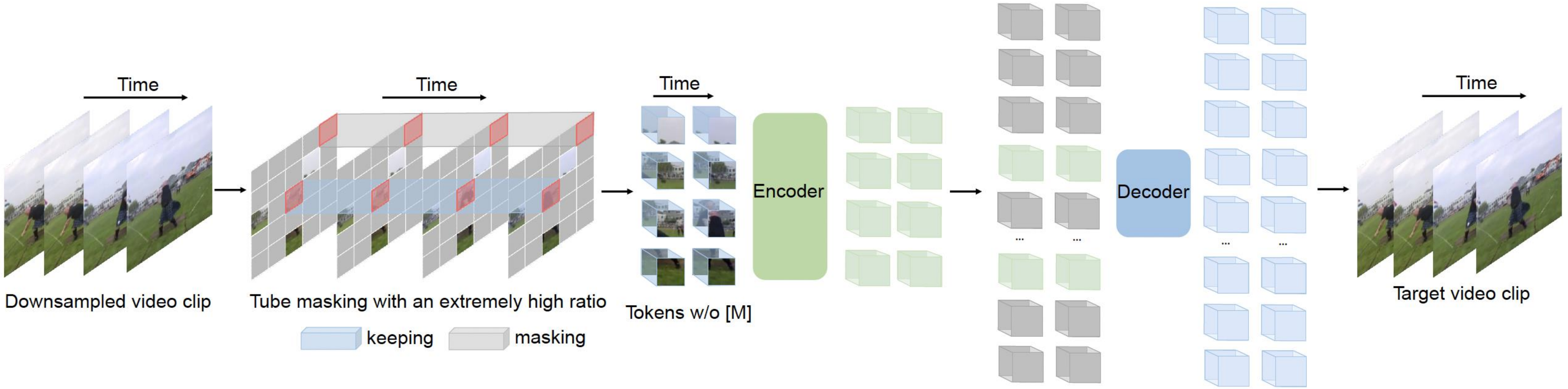
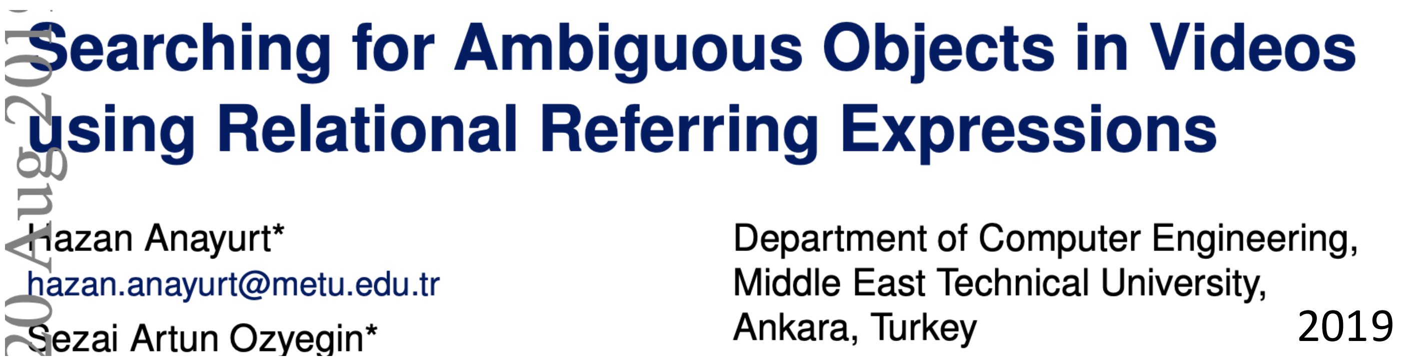


Figure 1: **VideoMAE** performs the task of masking random cubes and reconstructing the missing ones with an asymmetric encoder-decoder architecture. Due to high redundancy and temporal correlation in videos, we present the customized design of tube masking with an extremely high ratio (90% to 95%). This simple design enables us to create a more challenging and meaningful self-supervised task to make the learned representations capture more useful spatiotemporal structures.

Vision-Language Models

Reference: Bordes et al., “An Introduction to Vision-Language Modeling”, 2024. https://arxiv.org/pdf/2405.17247

Earlier Attempts:



Searching for Ambiguous Objects in Videos using Relational Referring Expressions

Hazan Anayurt*
hazan.anayurt@metu.edu.tr
Sezai Artun Ozyegin*

Department of Computer Engineering,
Middle East Technical University,
Ankara, Turkey

2019

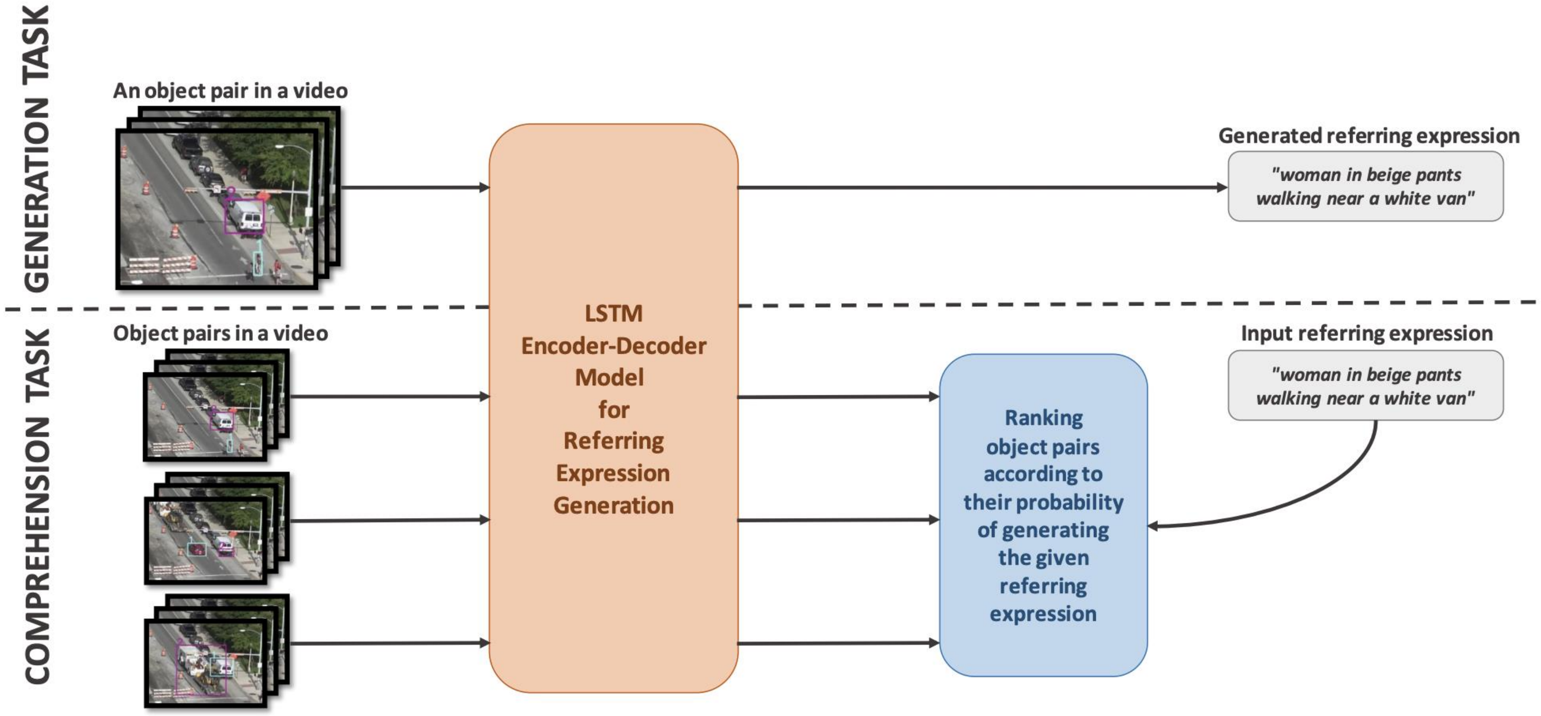

GENERATION TASK
An object pair in a video
Generated referring expression
"woman in beige pants walking near a white van"
COMPREHENSION TASK
Object pairs in a video
LSTM Encoder-Decoder Model for Referring Expression Generation
Ranking object pairs according to their probability of generating the given referring expression
Input referring expression
"woman in beige pants walking near a white van"

Figure 1: An overview of the generation and comprehension tasks performed by our model.

Earlier Attempts:

ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks

2019

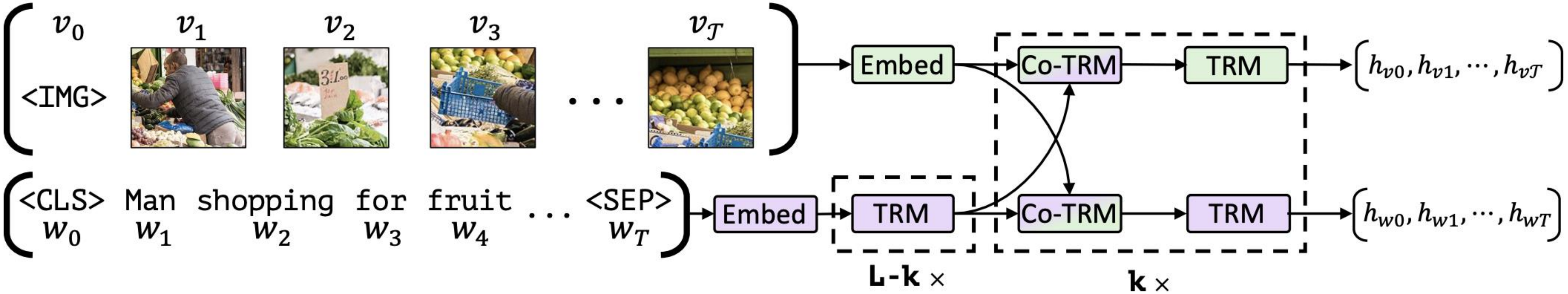

v_0
v_1
v_2
v_3
$v_\mathcal{T}$
<IMG>
...
<CLS> Man shopping for fruit ... <SEP>
w_0 w_1 w_2 w_3 w_4 w_T
Embed
Embed
TRM
Co-TRM
TRM
Co-TRM
TRM
L-k ×
k ×
$(h_{v0}, h_{v1}, \cdots, h_{v\mathcal{T}})$
$(h_{w0}, h_{w1}, \cdots, h_{wT})$

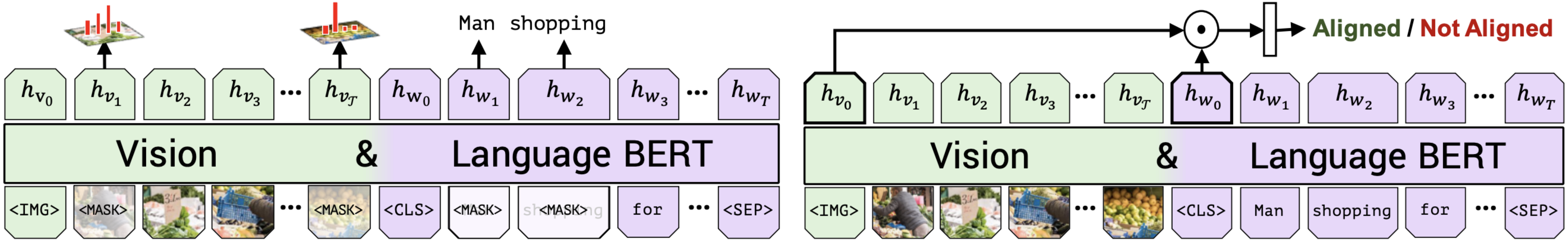
Figure 1: Our ViLBERT model consists of two parallel streams for visual (green) and linguistic (purple) processing that interact through novel co-attentional transformer layers. This structure allows for variable depths for each modality and enables sparse interaction through co-attention. Dashed boxes with multiplier subscripts denote repeated blocks of layers.

Co-TRM: Co-attentional Transformer

Earlier Attempts:

ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks

2019


Man
shopping
h_{v_0} h_{v_1} h_{v_2} h_{v_3} ... $h_{v_\mathcal{T}}$ h_{w_0} h_{w_1} h_{w_2} h_{w_3} ... h_{w_T}
Vision & Language BERT
<IMG> <MASK> ... <MASK> <CLS> <MASK> sh<MASK>ng for ... <SEP>
Aligned / Not Aligned
<IMG> ... <CLS> Man shopping for ... <SEP>

(a) Masked multi-modal learning

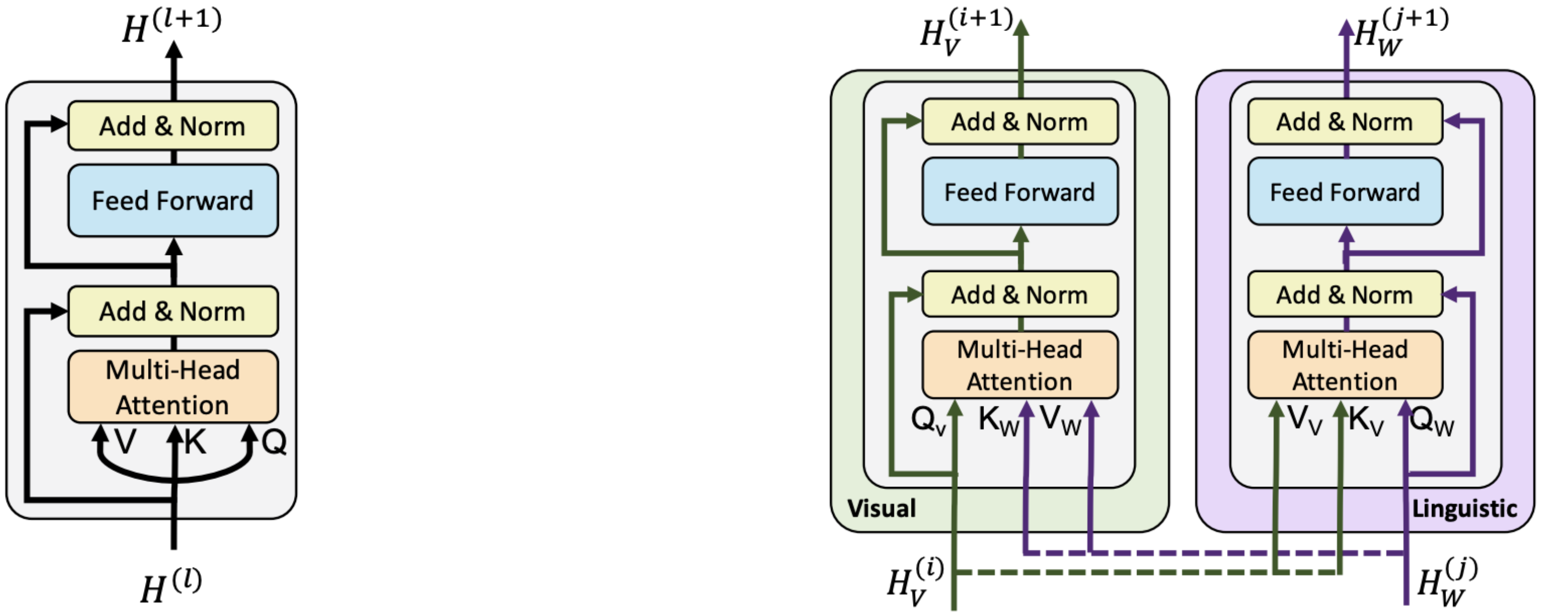
(b) Multi-modal alignment prediction

Figure 3: We train ViLBERT on the Conceptual Captions [24] dataset under two training tasks to learn visual grounding. In masked multi-modal learning, the model must reconstruct image region categories or words for masked inputs given the observed inputs. In multi-modal alignment prediction, the model must predict whether or not the caption describes the image content.

Earlier Attempts:

ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks

2019

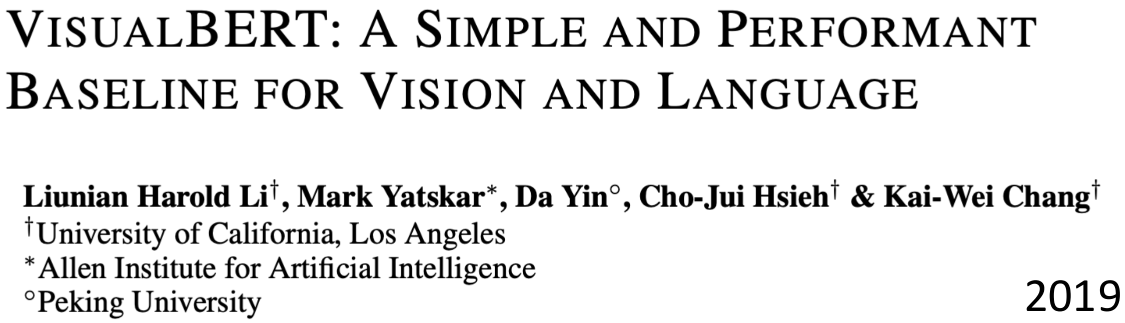


(a) Standard encoder transformer block

(b) Our co-attention transformer layer

Figure 2: We introduce a novel co-attention mechanism based on the transformer architecture. By exchanging key-value pairs in multi-headed attention, this structure enables vision-attended language features to be incorporated into visual representations (and vice versa).

Earlier Attempts:



VISUALBERT: A SIMPLE AND PERFORMANT BASELINE FOR VISION AND LANGUAGE

Liunian Harold Li[†], **Mark Yatskar**[*], **Da Yin**[°], **Cho-Jui Hsieh**[†] **& Kai-Wei Chang**[†]
[†]University of California, Los Angeles
[*]Allen Institute for Artificial Intelligence
[°]Peking University

2019

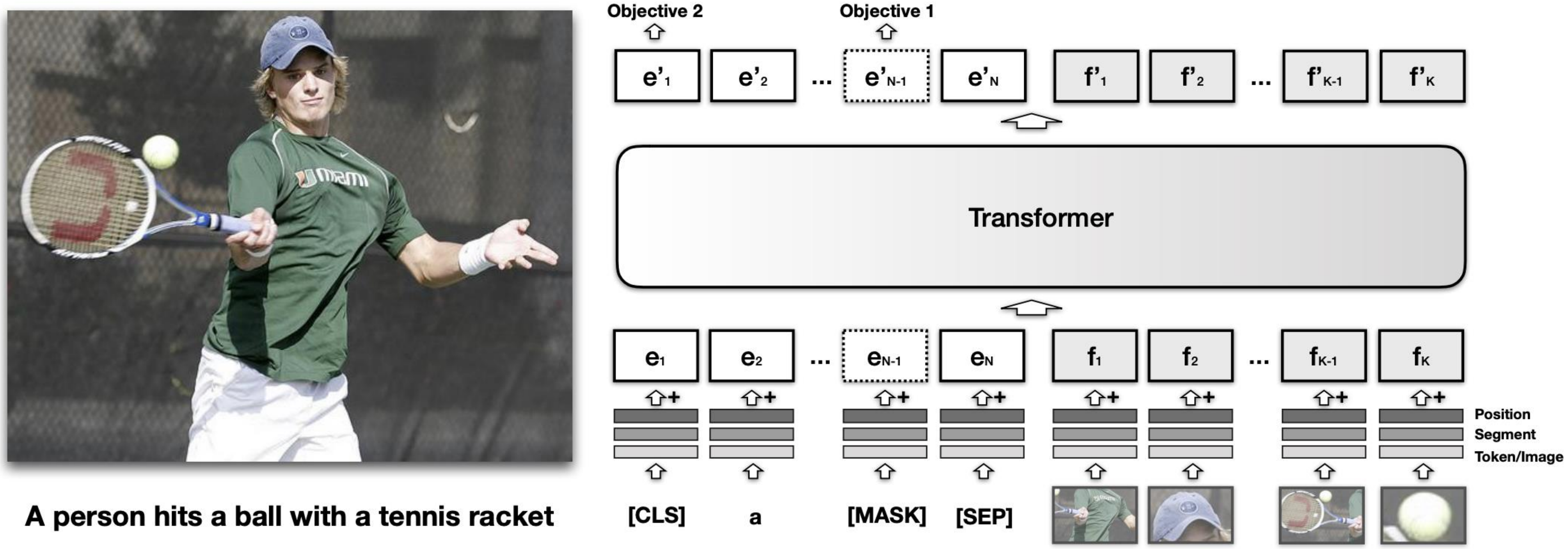
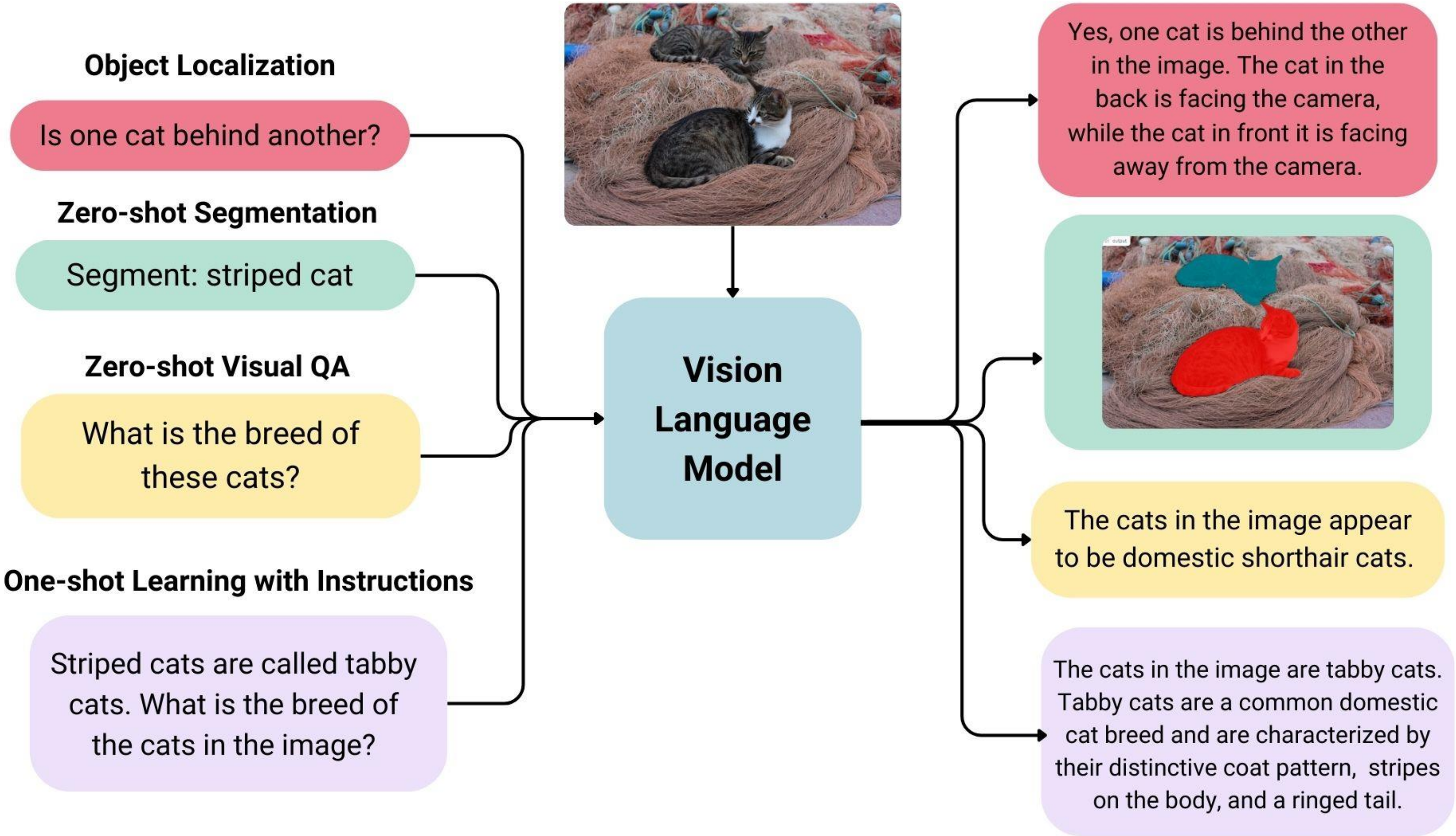


Figure 2: The architecture of VisualBERT. Image regions and language are combined with a Transformer to allow the self-attention to discover implicit alignments between language and vision. It is pre-trained with a masked language modeling (Objective 1), and sentence-image prediction task (Objective 2), on caption data and then fine-tuned for different tasks. See §3.3 for more details.

Overview



Object Localization

Is one cat behind another?

Zero-shot Segmentation

Segment: striped cat

Zero-shot Visual QA

What is the breed of these cats?

One-shot Learning with Instructions

Striped cats are called tabby cats. What is the breed of the cats in the image?

Vision Language Model

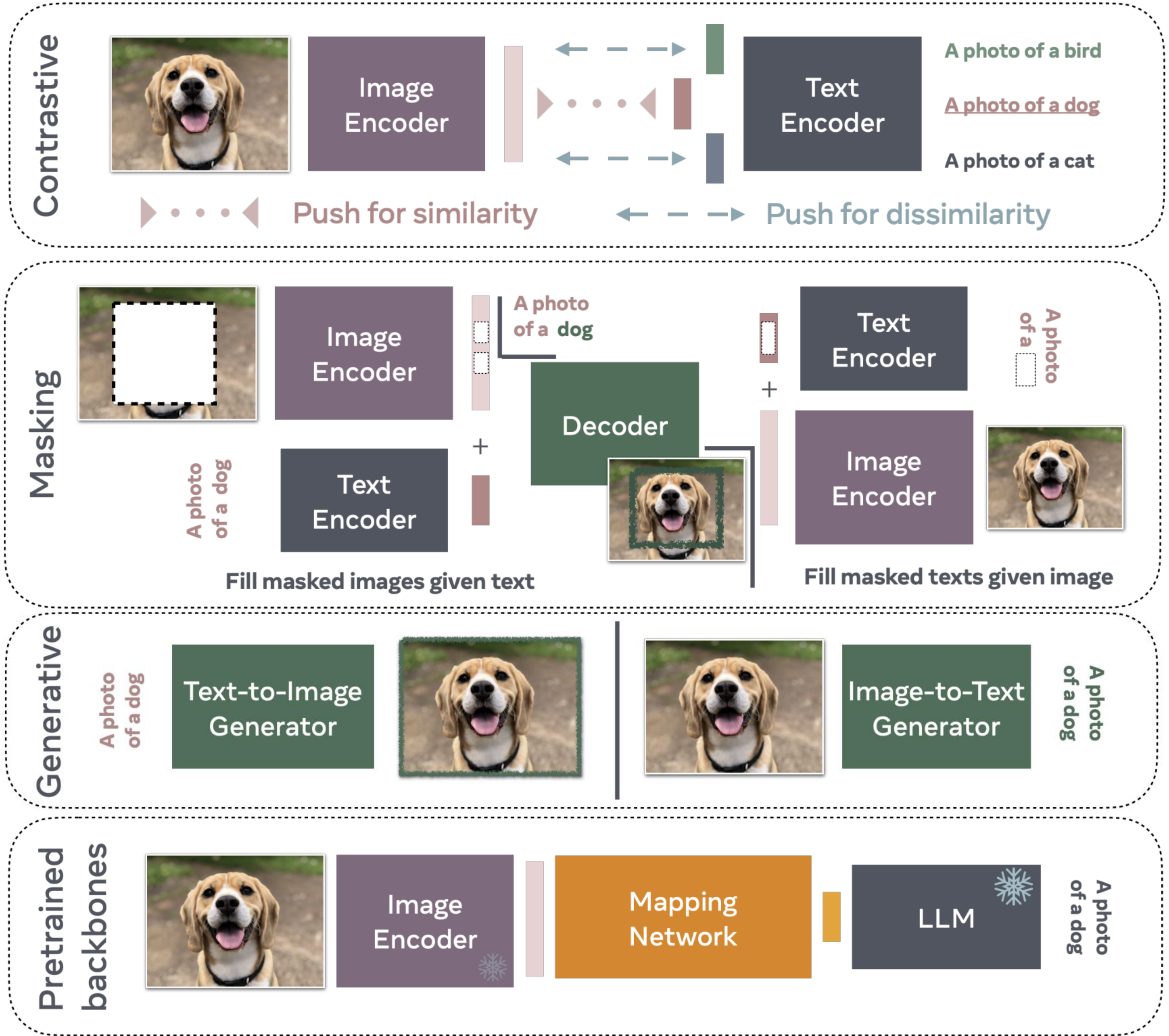
Yes, one cat is behind the other in the image. The cat in the back is facing the camera, while the cat in front it is facing away from the camera.

The cats in the image appear to be domestic shorthair cats.

The cats in the image are tabby cats. Tabby cats are a common domestic cat breed and are characterized by their distinctive coat pattern, stripes on the body, and a ringed tail.

Fig: https://huggingface.co/blog/vlms

Overview



Bordes et al., "An Introduction to Vision-Language Modeling", 2024. https://arxiv.org/pdf/2405.17247

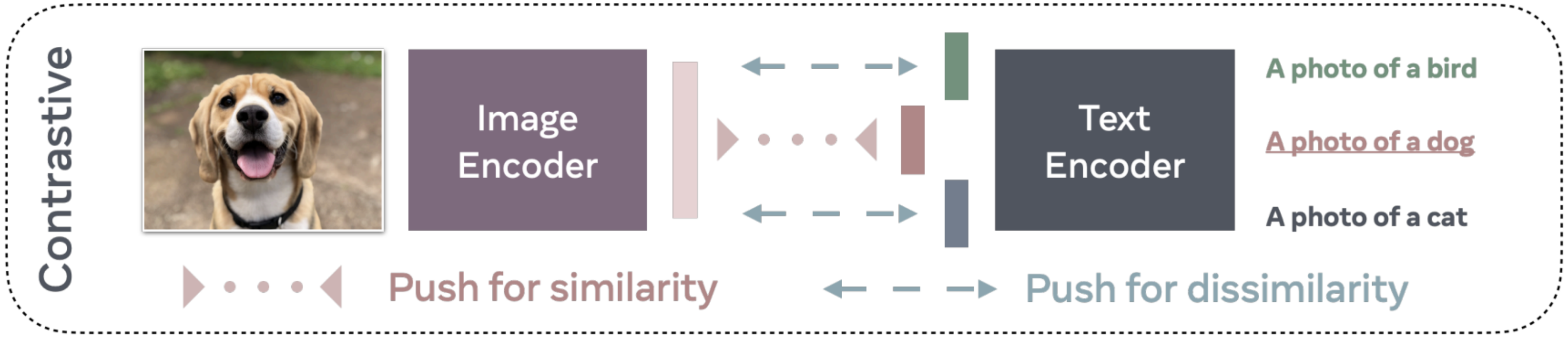
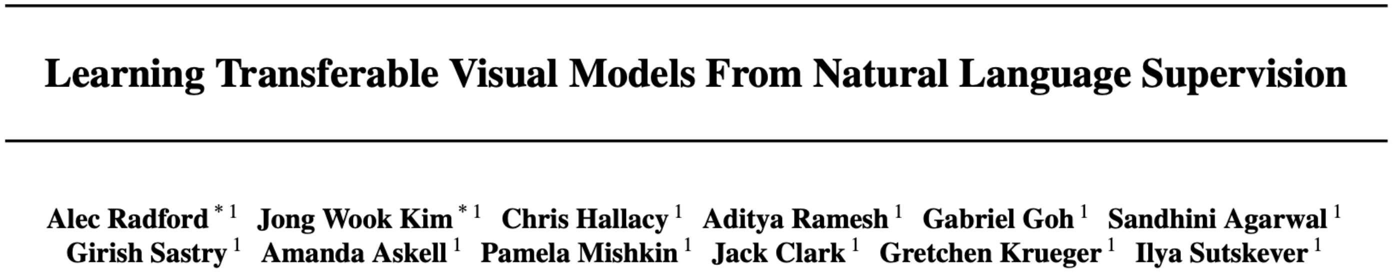


Fig: Bordes et al., “An Introduction to Vision-Language Modeling”, 2024.

Contrastive Approaches

CLIP



Learning Transferable Visual Models From Natural Language Supervision

Alec Radford *[1] **Jong Wook Kim** *[1] **Chris Hallacy**[1] **Aditya Ramesh**[1] **Gabriel Goh**[1] **Sandhini Agarwal**[1]
Girish Sastry[1] **Amanda Askell**[1] **Pamela Mishkin**[1] **Jack Clark**[1] **Gretchen Krueger**[1] **Ilya Sutskever**[1]

2021

- Motivation:
  - NLP models have benefited significantly from large amounts of web data
  - “Could scalable pre-training methods which learn directly from web text result in a similar breakthrough in computer vision?”

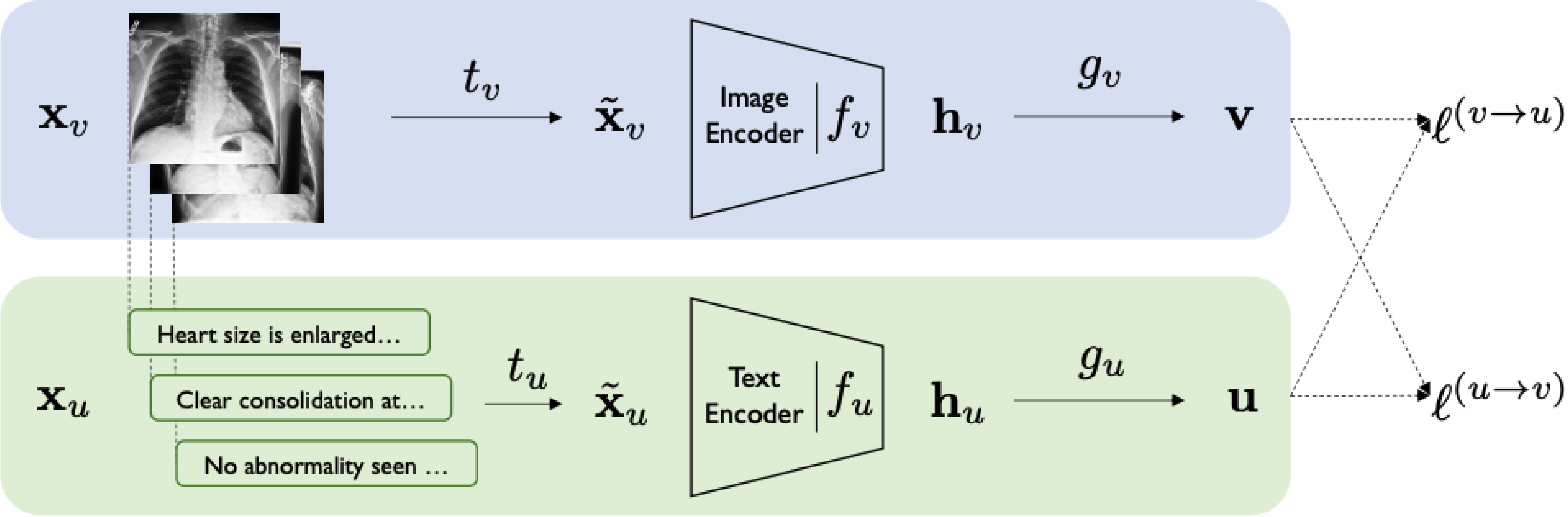
- Contribution:
  - Using natural language supervision for image representation learning at large scale
  - 400M pairs of (image, text)
  - Adapted from ConVIRT (train from scratch)

Proceedings of Machine Learning Research 182:1–24, 2022 Machine Learning for Healthcare

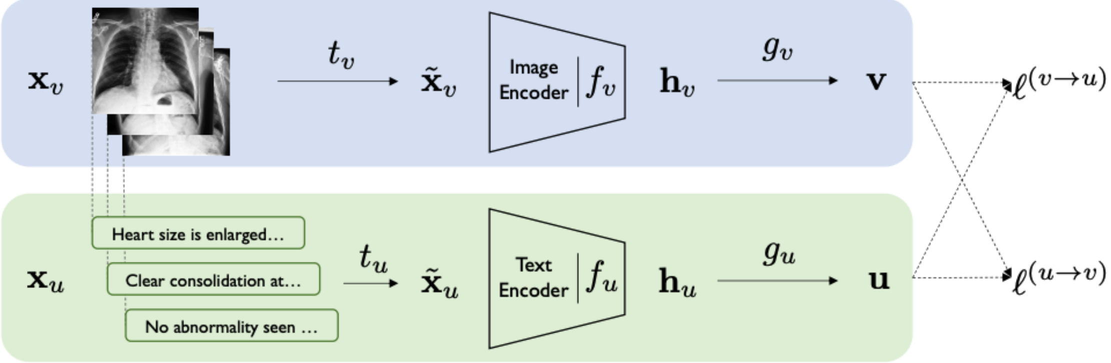
ConVIRT

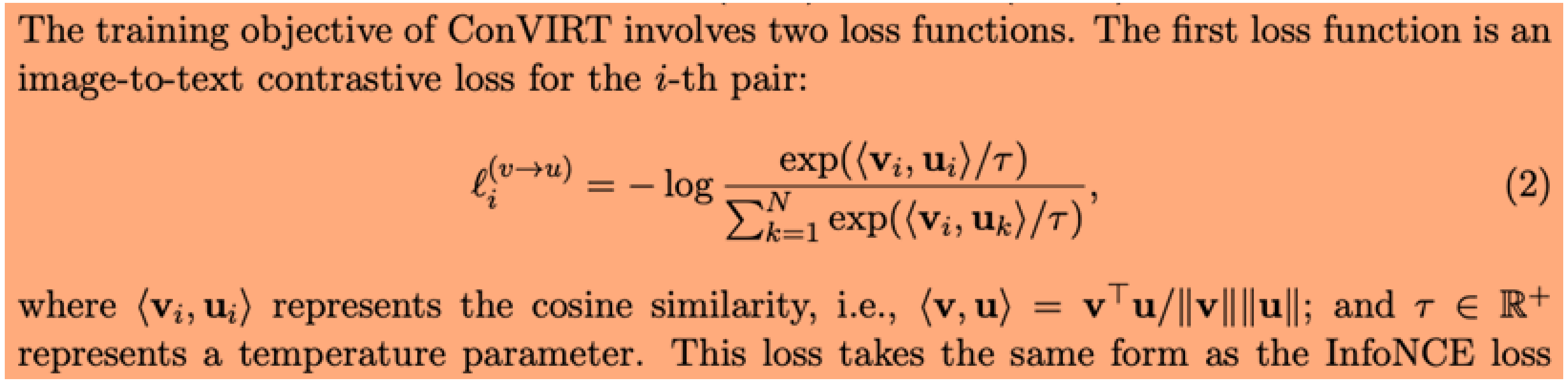
Contrastive Learning of Medical Visual Representations from Paired Images and Text

Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D. Manning, Curtis P. Langlotz



ConVIRT

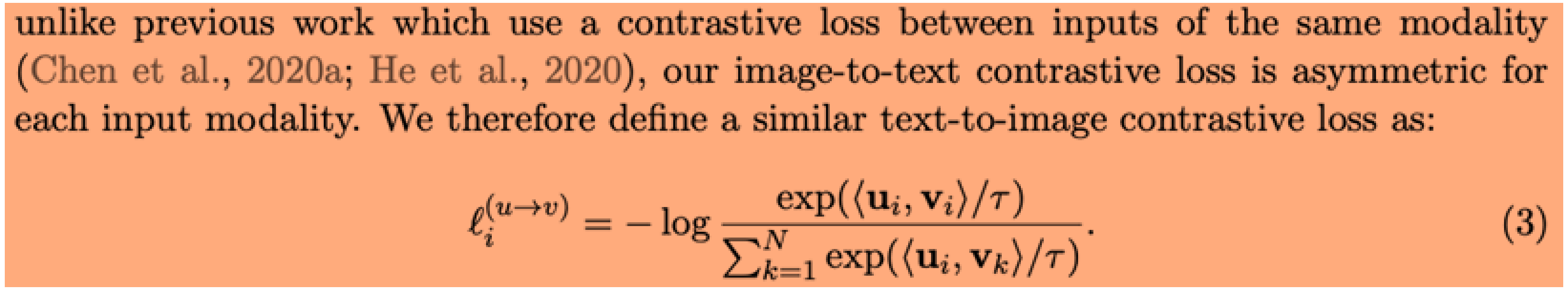




The training objective of ConVIRT involves two loss functions. The first loss function is an image-to-text contrastive loss for the i-th pair:

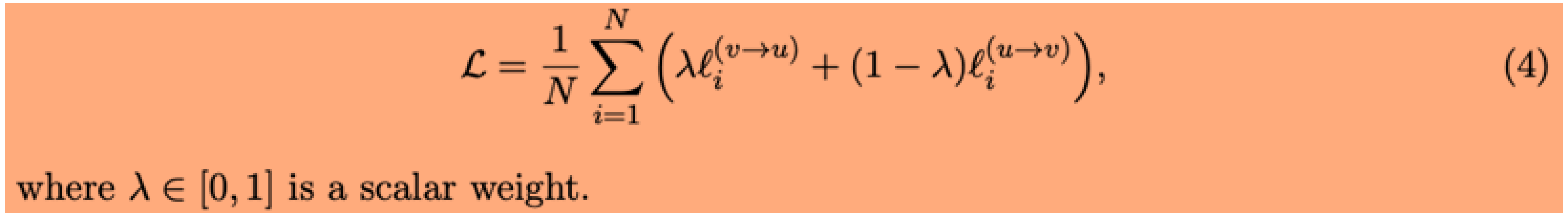
$$\ell_i^{(v\to u)} = -\log \frac{\exp(\langle \mathbf{v}_i, \mathbf{u}_i\rangle/\tau)}{\sum_{k=1}^{N}\exp(\langle \mathbf{v}_i, \mathbf{u}_k\rangle/\tau)}, \tag{2}$$

where $\langle \mathbf{v}_i, \mathbf{u}_i\rangle$ represents the cosine similarity, i.e., $\langle \mathbf{v}, \mathbf{u}\rangle = \mathbf{v}^\top \mathbf{u}/\|\mathbf{v}\|\|\mathbf{u}\|$; and $\tau \in \mathbb{R}^+$ represents a temperature parameter. This loss takes the same form as the InfoNCE loss



unlike previous work which use a contrastive loss between inputs of the same modality (Chen et al., 2020a; He et al., 2020), our image-to-text contrastive loss is asymmetric for each input modality. We therefore define a similar text-to-image contrastive loss as:

$$\ell_i^{(u\to v)} = -\log \frac{\exp(\langle \mathbf{u}_i, \mathbf{v}_i\rangle/\tau)}{\sum_{k=1}^{N}\exp(\langle \mathbf{u}_i, \mathbf{v}_k\rangle/\tau)}. \tag{3}$$



$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\left(\lambda \ell_i^{(v\to u)} + (1-\lambda)\ell_i^{(u\to v)}\right), \tag{4}$$

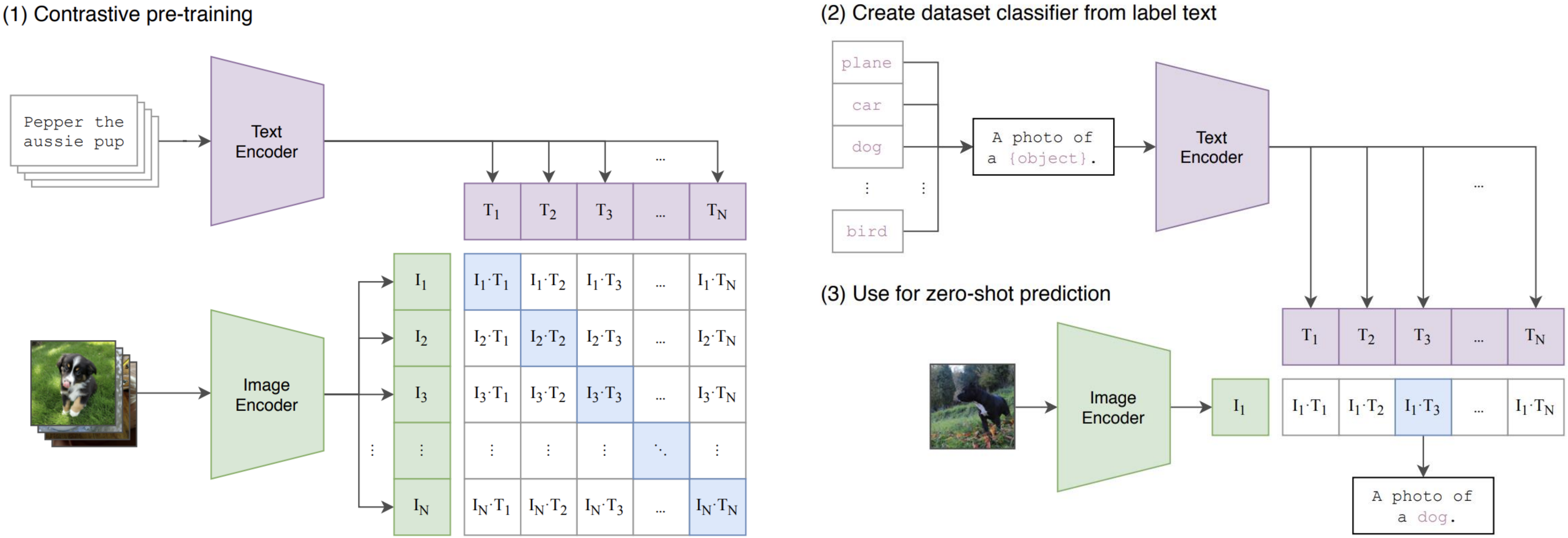
where $\lambda \in [0,1]$ is a scalar weight.

CLIP

Learning Transferable Visual Models From Natural Language Supervision

Alec Radford [*1] **Jong Wook Kim** [*1] **Chris Hallacy** [1] **Aditya Ramesh** [1] **Gabriel Goh** [1] **Sandhini Agarwal** [1]
Girish Sastry [1] **Amanda Askell** [1] **Pamela Mishkin** [1] **Jack Clark** [1] **Gretchen Krueger** [1] **Ilya Sutskever** [1]

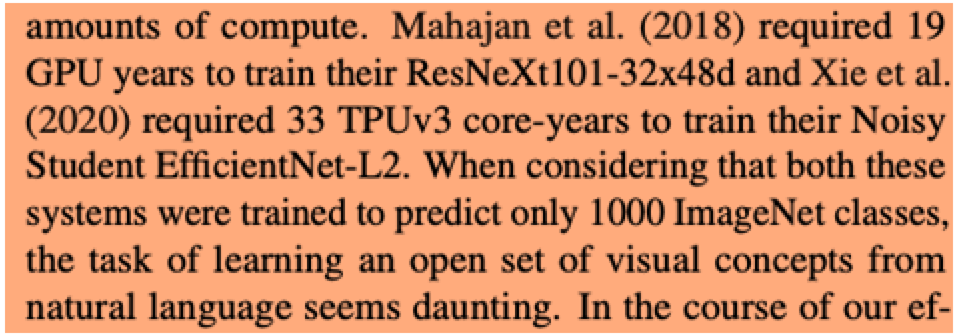
2021


(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
T_1 T_2 T_3 ... T_N
Image Encoder
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
(3) Use for zero-shot prediction
Image Encoder
I_1
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
A photo of a dog.

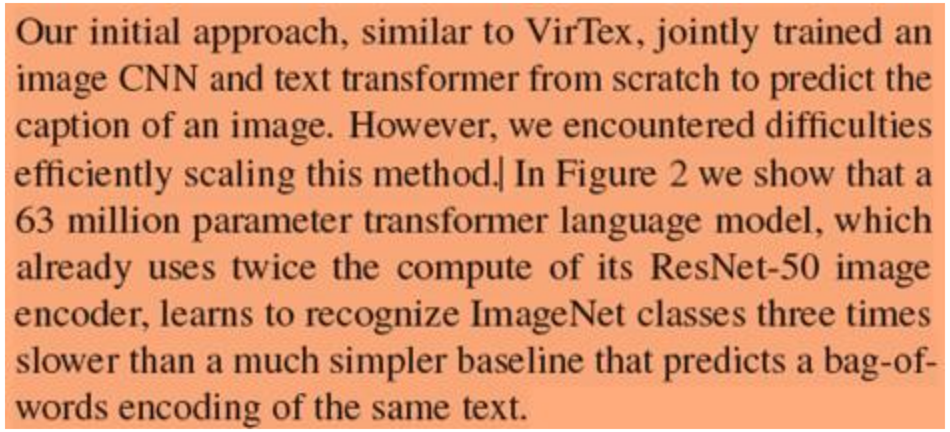
CLIP: Dataset

- MS-COCO, Visual Genome, .. are limited in size
- YFCC100M: Includes 100M images but metadata for each image is sparse and low quality. When filtered, the # of images is around the size of ImageNet
- NL supervision requires a large-scale dataset
- Search (image, text) pairs with text including 500K base queries:
  - "The base query list is all words occurring at least 100 times in the English version of Wikipedia. This is augmented with bi-grams with high pointwise mutual information as well as the names of all Wikipedia articles above a certain search volume. Finally all WordNet synsets not already in the query list are added."
- To balance, 20K (image, text) pairs are included for each query.
- 400M is similar to the size of WebText used to train GPT-2

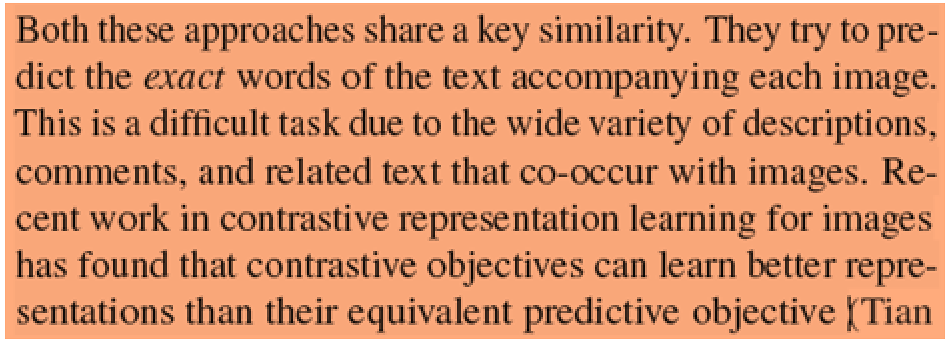
CLIP: Efficiency



amounts of compute. Mahajan et al. (2018) required 19 GPU years to train their ResNeXt101-32x48d and Xie et al. (2020) required 33 TPUv3 core-years to train their Noisy Student EfficientNet-L2. When considering that both these systems were trained to predict only 1000 ImageNet classes, the task of learning an open set of visual concepts from natural language seems daunting. In the course of our ef-



Our initial approach, similar to VirTex, jointly trained an image CNN and text transformer from scratch to predict the caption of an image. However, we encountered difficulties efficiently scaling this method. In Figure 2 we show that a 63 million parameter transformer language model, which already uses twice the compute of its ResNet-50 image encoder, learns to recognize ImageNet classes three times slower than a much simpler baseline that predicts a bag-of-words encoding of the same text.



Both these approaches share a key similarity. They try to predict the *exact* words of the text accompanying each image. This is a difficult task due to the wide variety of descriptions, comments, and related text that co-occur with images. Recent work in contrastive representation learning for images has found that contrastive objectives can learn better representations than their equivalent predictive objective (Tian

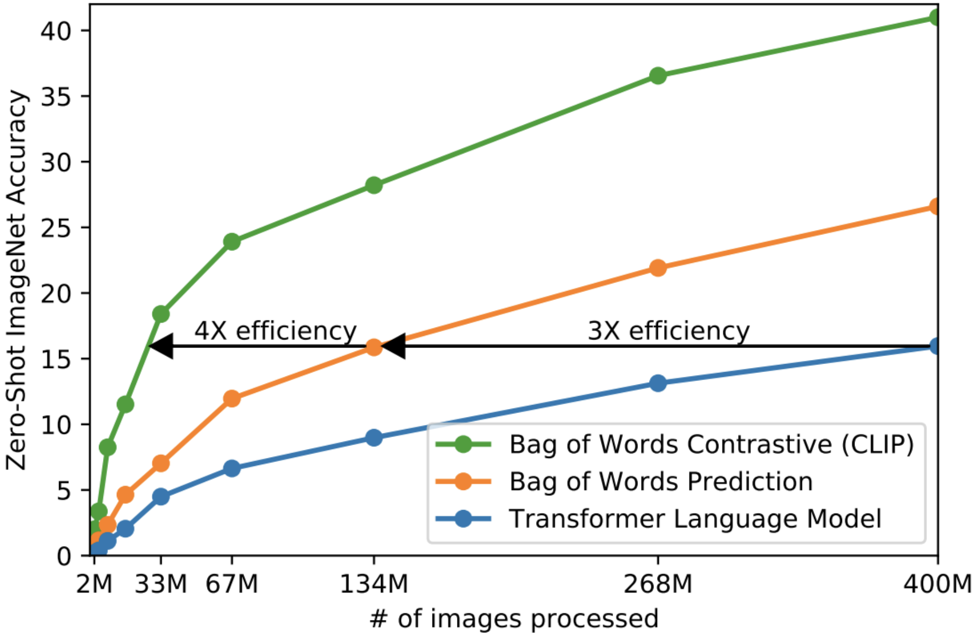
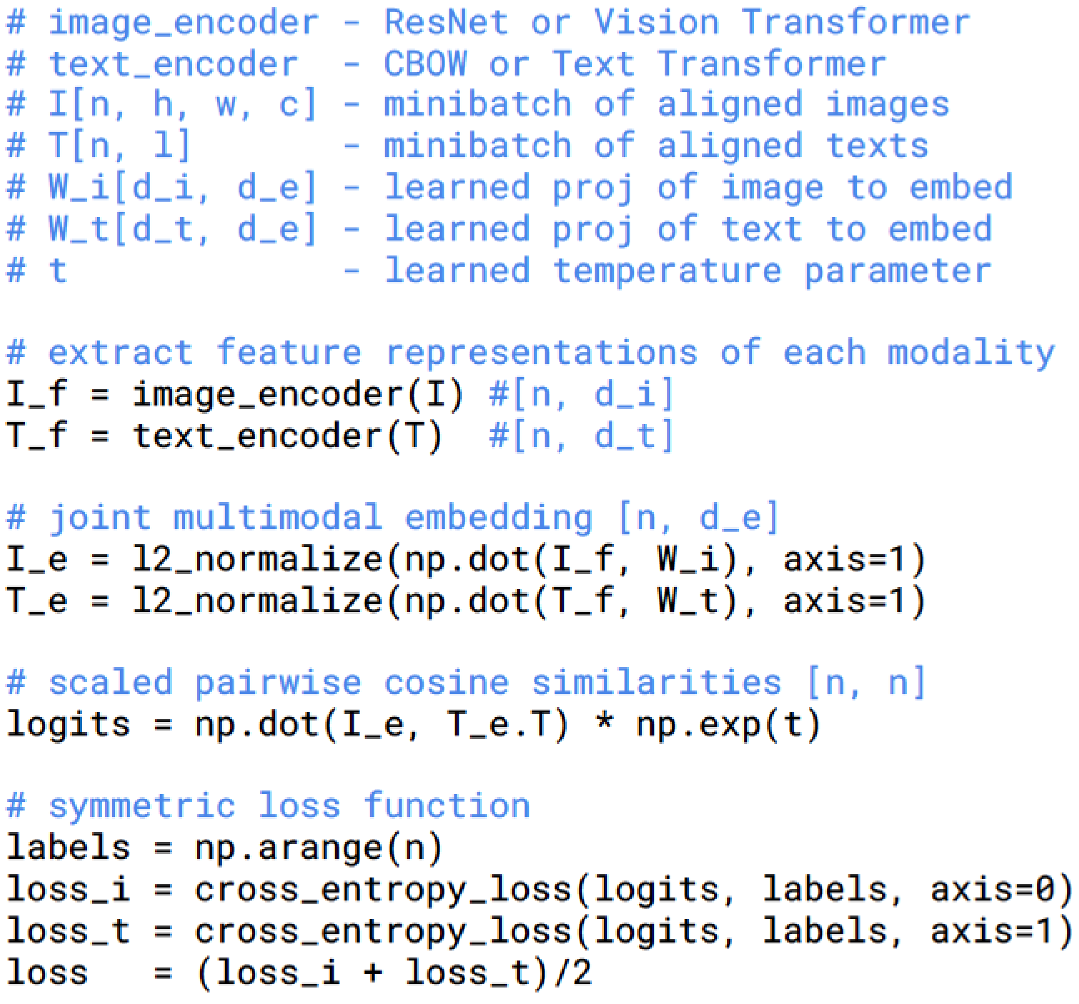


Figure 2. **CLIP is much more efficient at zero-shot transfer than our image caption baseline.** Although highly expressive, we found that transformer-based language models are relatively weak at zero-shot ImageNet classification. Here, we see that it learns 3x slower than a baseline which predicts a bag-of-words (BoW) encoding of the text (Joulin et al., 2016). Swapping the prediction objective for the contrastive objective of CLIP further improves efficiency another 4x.

CLIP: Method

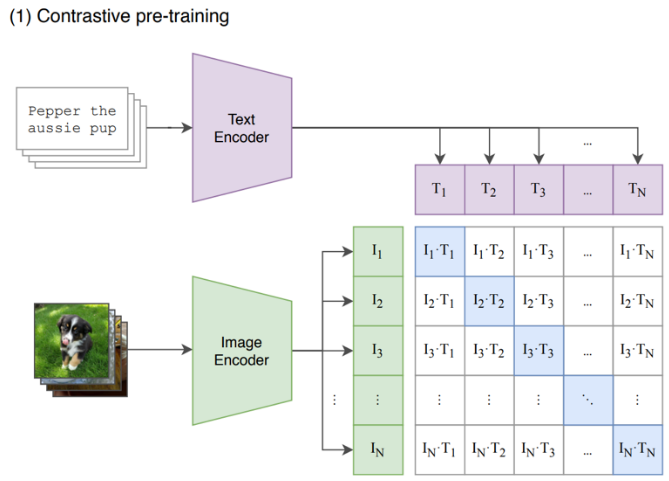


```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

$$\mathcal{L}_{\text{infoNCE}} = -\sum_{(i,j)\in\mathbb{P}} \log\left(\frac{e^{\text{CoSim}(\boldsymbol{z}_i,\boldsymbol{z}_j)/\tau}}{\sum_{k=1}^{N} e^{\text{CoSim}(\boldsymbol{z}_i,\boldsymbol{z}_k)/\tau}}\right),$$

CLIP: Architecture


(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$

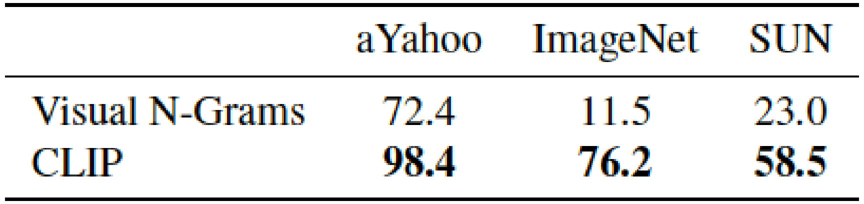
- Image Encoder
  - Option 1: ResNet-D, some modifications over ResNet-50 + antialiased rect-2 blur pooling + attention pooling (instead of global average pooling
  - Option 2: ViT + additional layer norm after combining patch and position embeddings

- Text Encoder
  - Transformer (same architecture as GPT-2)
  - Different sizes (base version: 63M params, 12 layers, 512-wide model with 8 attention heads)
  - Lowercased byte-pair encoding
  - Max sequence length: Trimmed at 76
  - [SOS] and [EOS] tokens

CLIP: Training

- Train 5 ResNets, 3 ViTs:
  - ResNet-50, ResNet-101, 3 More following EfficientNet-style modifications on ResNet-50
  - ViT-B/32, ViT-B/16, ViT-L/14
- 32 epochs, Adam, weight decay (on all weights except for biases), LR with cosine scheduling
- Hyperparams:
  - Grid search, random search, manual tuning for 1 epoch for ResNet-50-baseline
  - Then based on heuristics adapted for larger models
- Temperature: Learnable, initialized at 0.07 and clipped to prevent explosion
- Batchsize: 32,768.
- “The largest ResNet model, RN50x64, took 18 days to train on 592 V100 GPUs while the largest Vision Transformer took 12 days on 256 V100 GPUs.”

CLIP: Results



	aYahoo	ImageNet	SUN
Visual N-Grams	72.4	11.5	23.0
CLIP	**98.4**	**76.2**	**58.5**

Table 1. Comparing CLIP to prior zero-shot transfer image classification results. CLIP improves performance on all three datasets by a large amount. This improvement reflects many differences in the 4 years since the development of Visual N-Grams (Li et al., 2017).

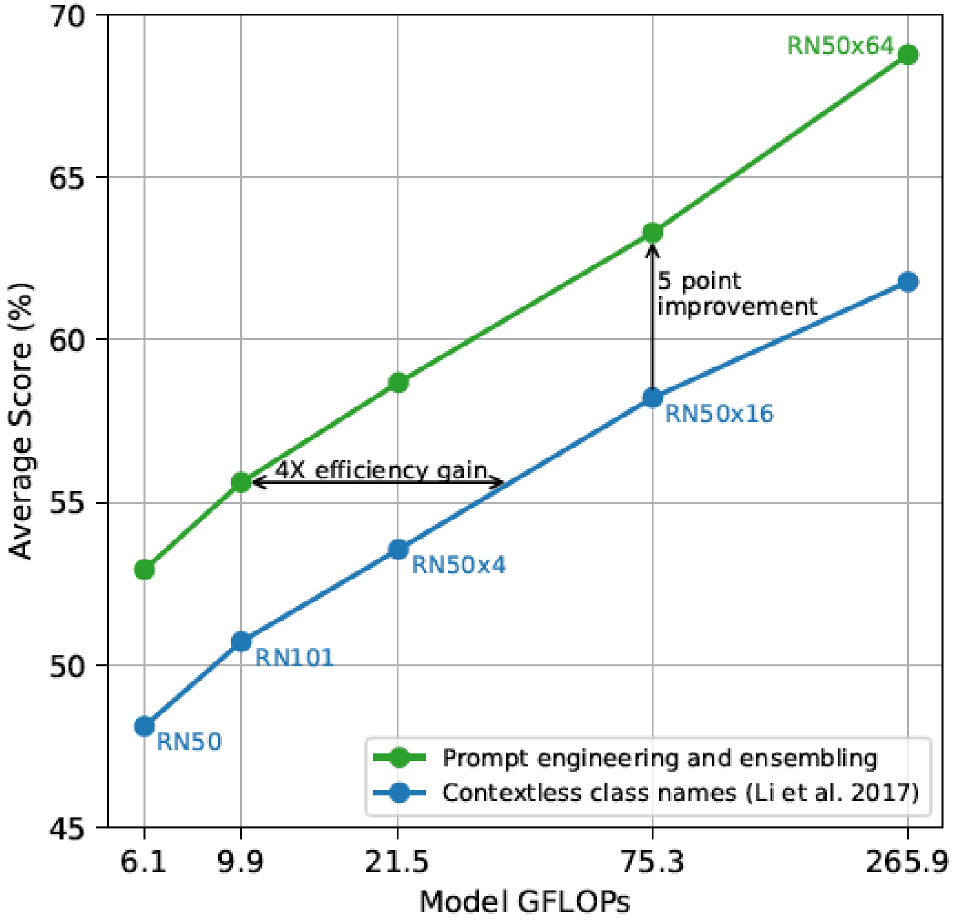


Figure 4. **Prompt engineering and ensembling improve zero-shot performance.** Compared to the baseline of using contextless class names, prompt engineering and ensembling boost zero-shot classification performance by almost 5 points on average across 36 datasets. This improvement is similar to the gain from using 4 times more compute with the baseline zero-shot method but is "free" when amortized over many predictions.

CLIP: Results

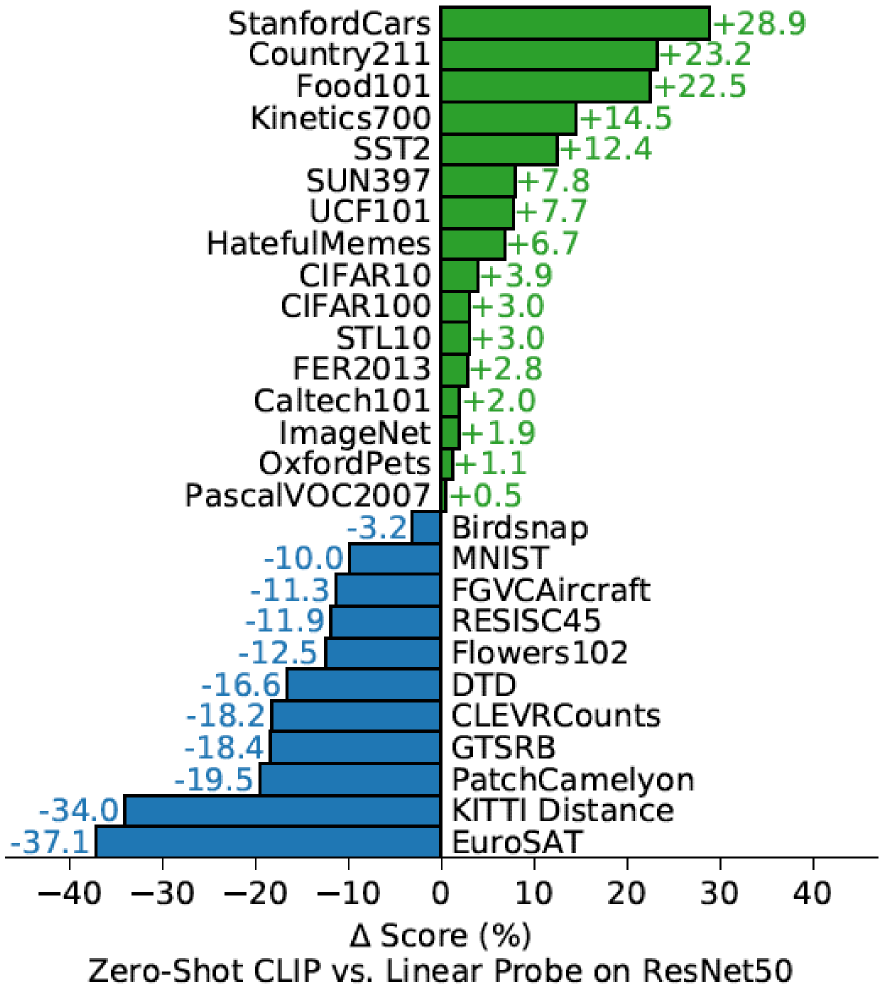


Figure 5. **Zero-shot CLIP is competitive with a fully supervised baseline.** Across a 27 dataset eval suite, a zero-shot CLIP classifier outperforms a fully supervised linear classifier fitted on ResNet-50 features on 16 datasets, including ImageNet.

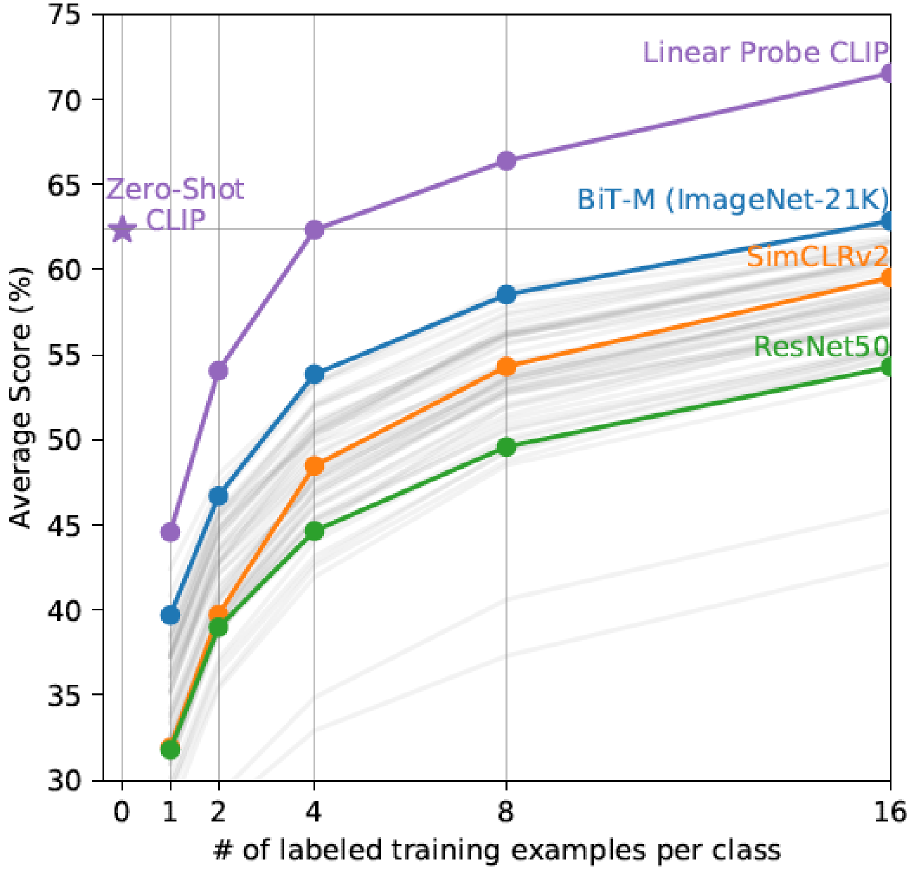


Figure 6. **Zero-shot CLIP outperforms few-shot linear probes.** Zero-shot CLIP matches the average performance of a 4-shot linear classifier trained on the same feature space and nearly matches the best results of a 16-shot linear classifier across publicly available models. For both BiT-M and SimCLRv2, the best performing model is highlighted. Light gray lines are other models in the eval suite. The 20 datasets with at least 16 examples per class were used in this analysis.

CLIP: Results

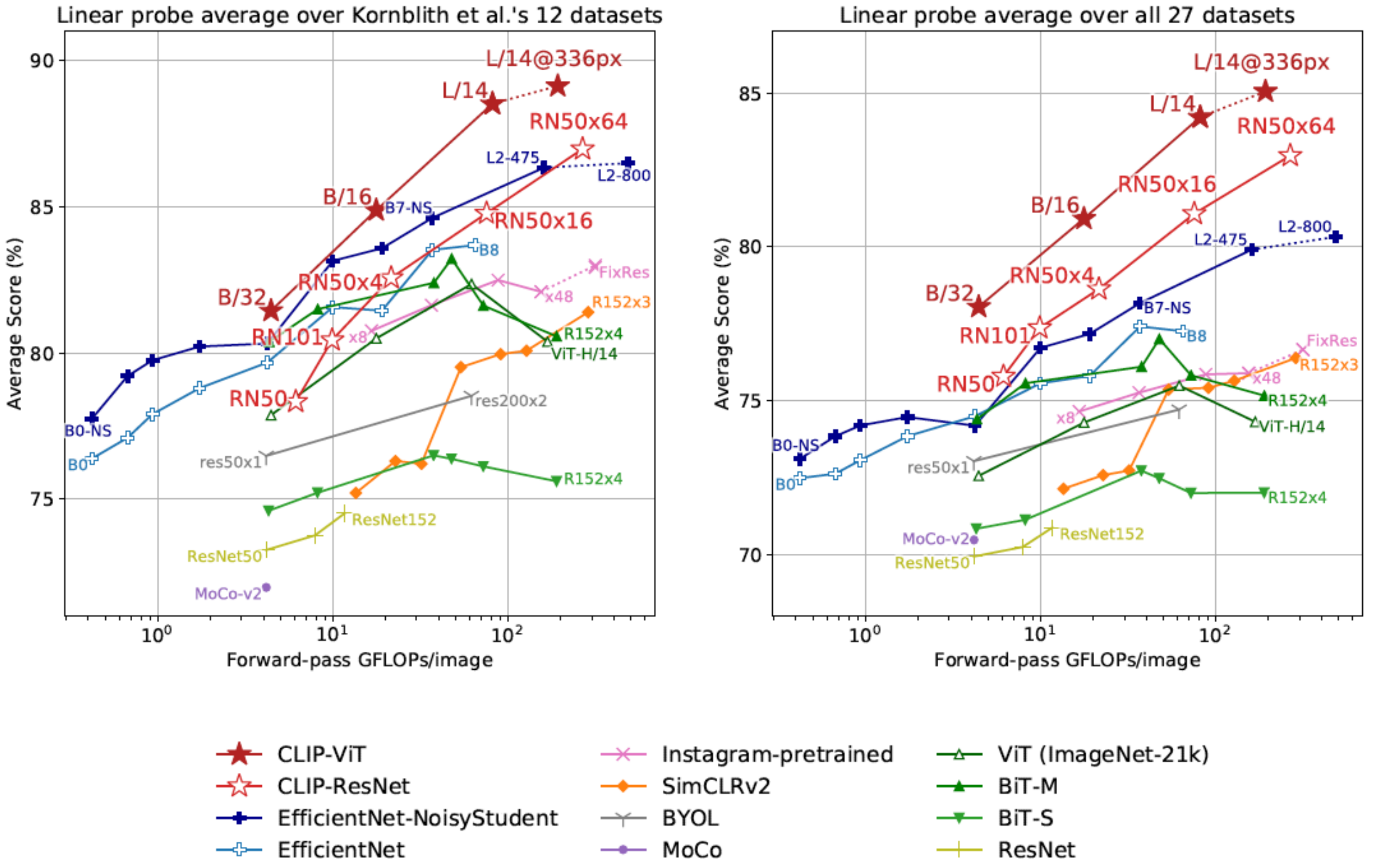


Figure 10. **Linear probe performance of CLIP models in comparison with state-of-the-art computer vision models**, including EfficientNet (Tan & Le, 2019; Xie et al., 2020), MoCo (Chen et al., 2020d), Instagram-pretrained ResNeXt models (Mahajan et al., 2018; Touvron et al., 2019), BiT (Kolesnikov et al., 2019), ViT (Dosovitskiy et al., 2020), SimCLRv2 (Chen et al., 2020c), BYOL (Grill et al., 2020), and the original ResNet models (He et al., 2016b). (Left) Scores are averaged over 12 datasets studied by Kornblith et al. (2019). (Right) Scores are averaged over 27 datasets that contain a wider variety of distributions. Dotted lines indicate models fine-tuned or evaluated on images at a higher-resolution than pre-training. See Table 10 for individual scores and Figure 20 for plots for each dataset.

CLIP: Results

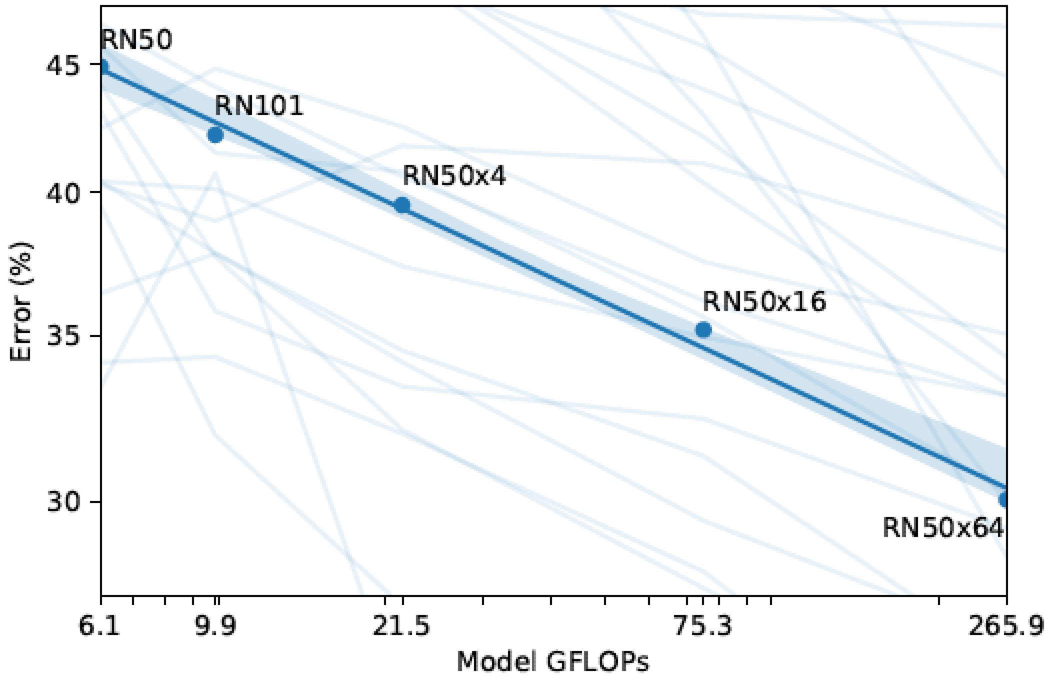


Figure 9. **Zero-shot CLIP performance scales smoothly as a function of model compute.** Across 39 evals on 36 different datasets, average zero-shot error is well modeled by a log-log linear trend across a 44x range of compute spanning 5 different CLIP models. Lightly shaded lines are performance on individual evals, showing that performance is much more varied despite the smooth overall trend.

CLIP: Extensions

- SigLIP (Zhai et al., 2023)
  - CLIP with NCE Loss with binary CE instead of CLIP’s multi-class InfoNCE
  - Better zero-shot performance on smaller batch sizes
- SLIP
  - CLIP combined with Self-Supervised Learning

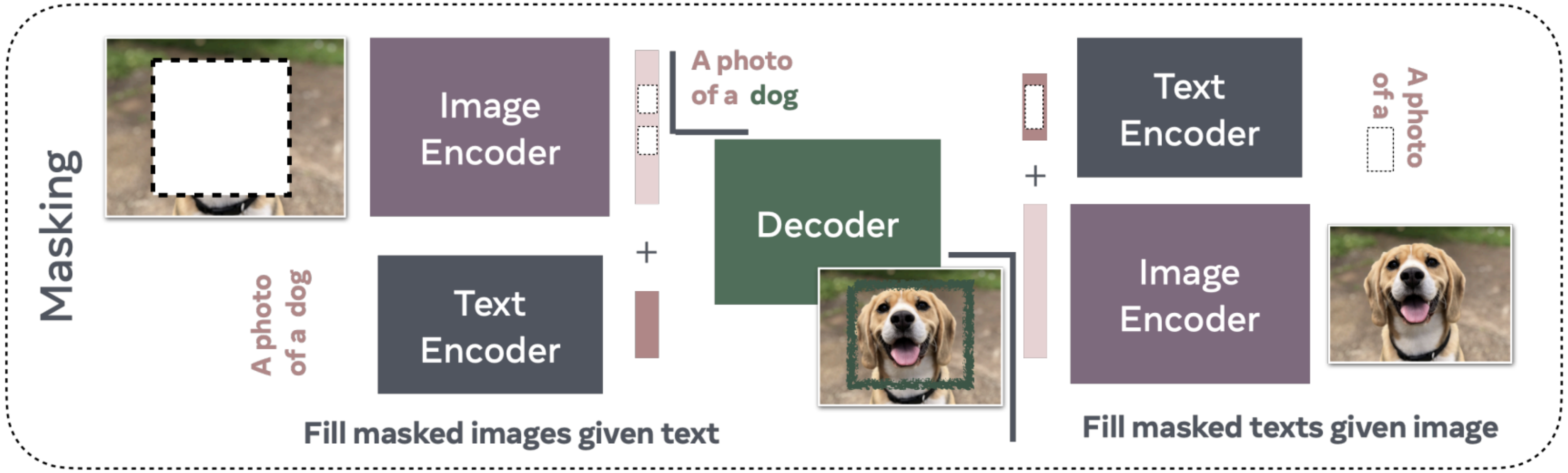


Fig: Bordes et al., "An Introduction to Vision-Language Modeling", 2024.

Masking Approaches

FLAVA

Foundational Language And Vision Alignment (FLAVA)
[Singh et al., 2022].

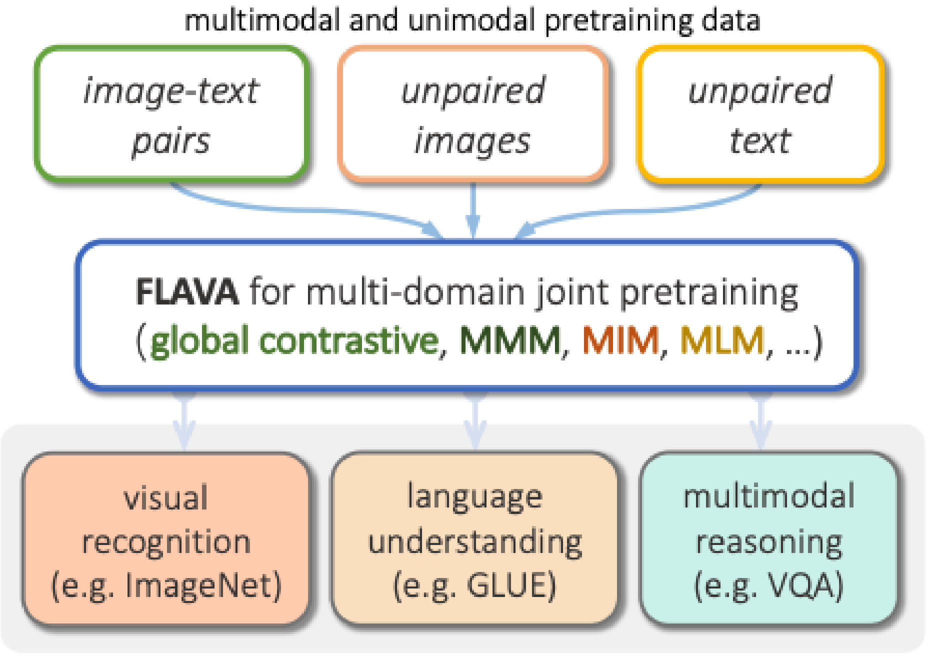
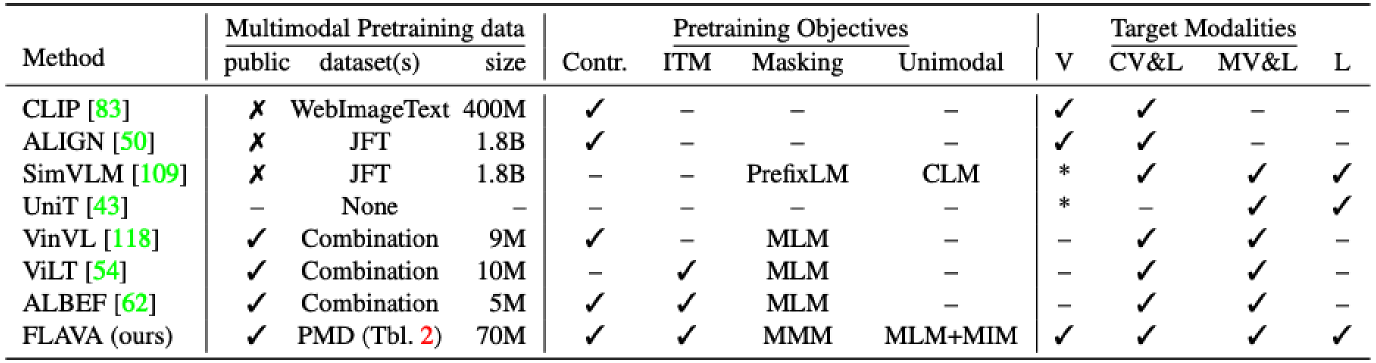

multimodal and unimodal pretraining data
image-text pairs
unpaired images
unpaired text
FLAVA for multi-domain joint pretraining
(global contrastive, MMM, MIM, MLM, ...)
visual recognition (e.g. ImageNet)
language understanding (e.g. GLUE)
multimodal reasoning (e.g. VQA)

Figure 1. We present FLAVA, a language and vision alignment model that learns strong representations from multimodal (image-text pairs) and unimodal data (unpaired images and text) and can be applied to target a broad scope of tasks from three domains (visual recognition, language understanding, and multimodal reasoning) under a common transformer model architecture.

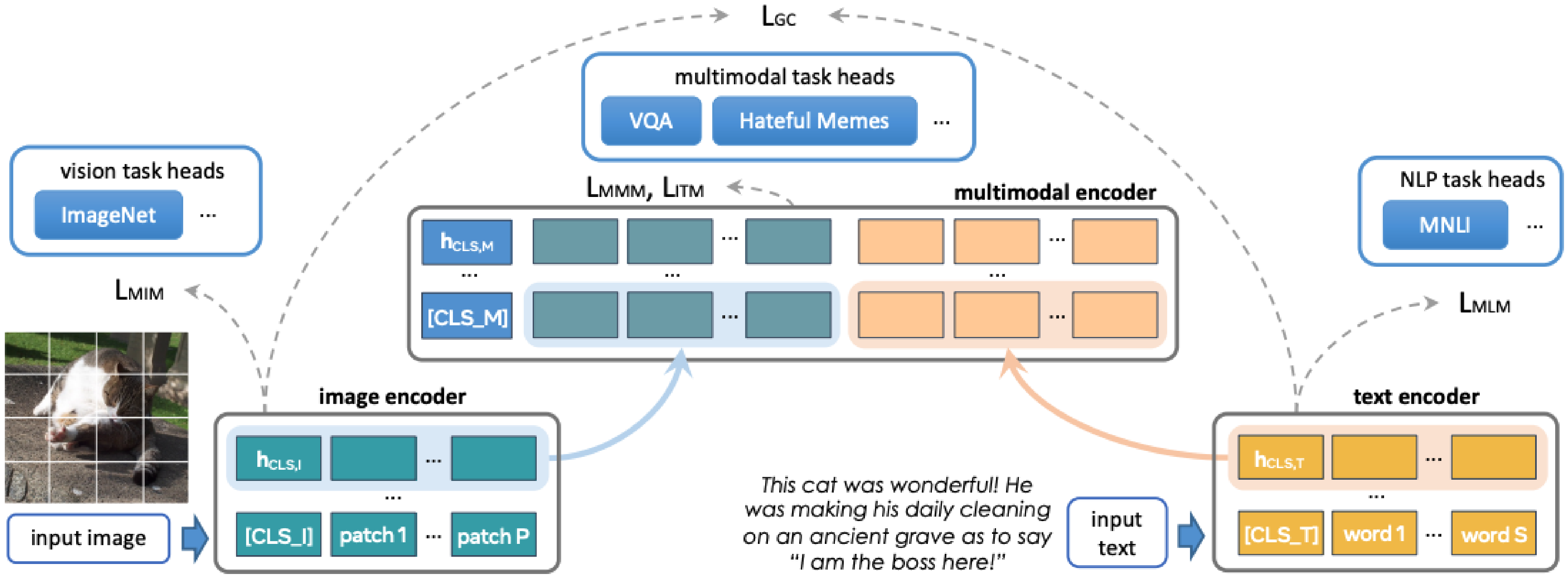


Method	Multimodal Pretraining data			Pretraining Objectives				Target Modalities			
	public	dataset(s)	size	Contr.	ITM	Masking	Unimodal	V	CV&L	MV&L	L
CLIP [83]	✗	WebImageText	400M	✓	–	–	–	✓	✓	–	–
ALIGN [50]	✗	JFT	1.8B	✓	–	–	–	✓	✓	–	–
SimVLM [109]	✗	JFT	1.8B	–	–	PrefixLM	CLM	*	✓	✓	✓
UniT [43]	–	None	–	–	–	–	–	*	–	✓	✓
VinVL [118]	✓	Combination	9M	✓	–	MLM	–	–	✓	✓	–
ViLT [54]	✓	Combination	10M	–	✓	MLM	–	–	✓	✓	–
ALBEF [62]	✓	Combination	5M	✓	✓	MLM	–	–	✓	✓	–
FLAVA (ours)	✓	PMD (Tbl. 2)	70M	✓	✓	MMM	MLM+MIM	✓	✓	✓	✓

Table 1. Comparison of recent models in different modalities. CV&L and MV&L stands for cross-modal and multi-modal vision-and-language. * means the modality is partially targeted (SimVLM [109] and UniT [43] include ImageNet and object detection, respectively).

GC: Global Contrastive Loss (same as CLIP)

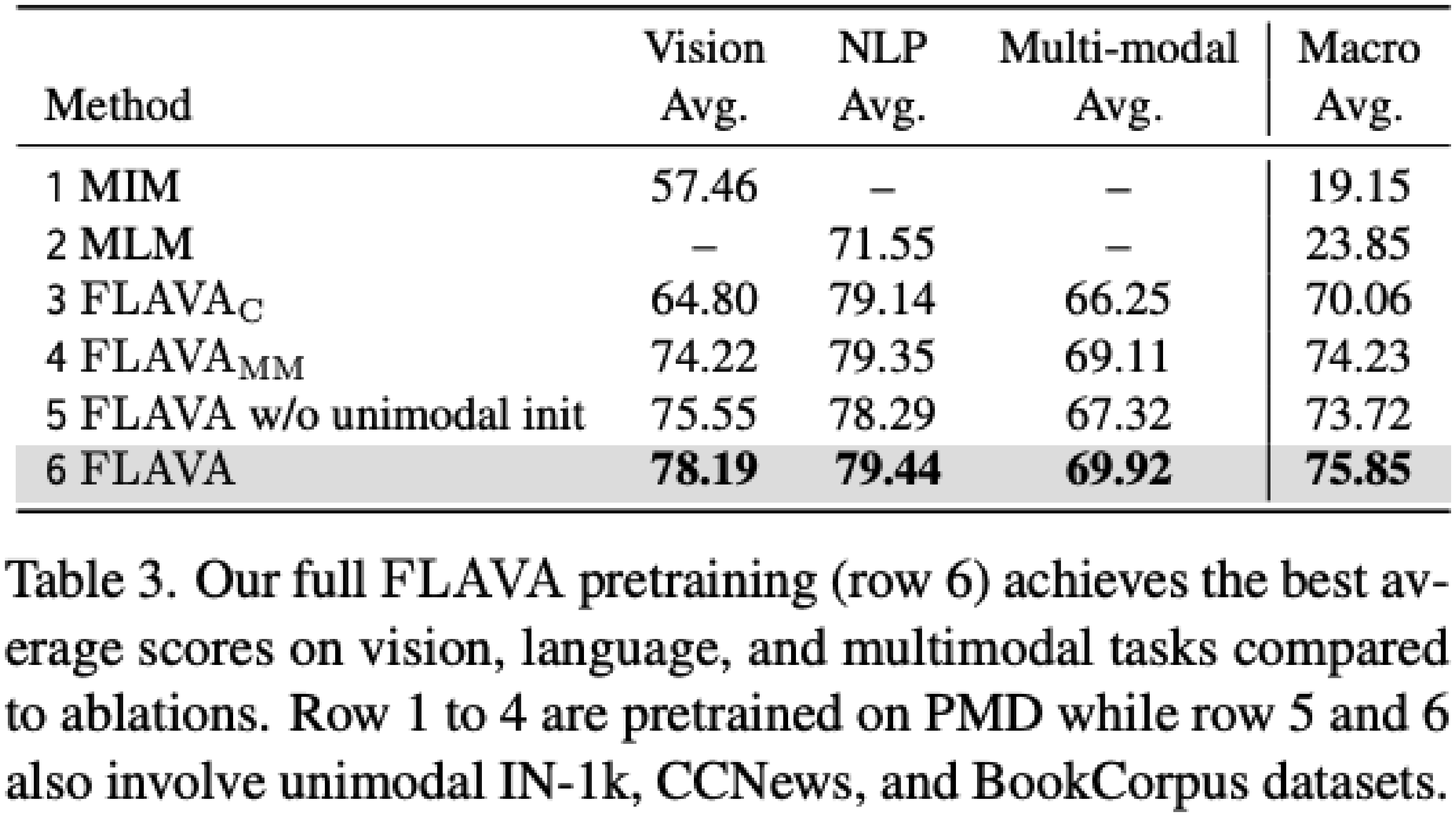
FLAVA



This cat was wonderful! He was making his daily cleaning on an ancient grave as to say "I am the boss here!"

Figure 2. **An overview of our FLAVA model**, with an image encoder transformer to capture unimodal image representations, a text encoder transformer to process unimodal text information, and a multimodal encoder transformer that takes as input the encoded unimodal image and text and integrates their representations for multimodal reasoning. **During pretraining**, masked image modeling (MIM) and mask language modeling (MLM) losses are applied onto the image and text encoders over a single image or a text piece, respectively, while contrastive, masked multimodal modeling (MMM), and image-text matching (ITM) loss are used over paired image-text data. **For downstream tasks**, classification heads are applied on the outputs from the image, text, and multimodal encoders respectively for visual recognition, language understanding, and multimodal reasoning tasks.

FLAVA: Results



Method	Vision Avg.	NLP Avg.	Multi-modal Avg.	Macro Avg.
1 MIM	57.46	–	–	19.15
2 MLM	–	71.55	–	23.85
3 FLAVA$_{\text{C}}$	64.80	79.14	66.25	70.06
4 FLAVA$_{\text{MM}}$	74.22	79.35	69.11	74.23
5 FLAVA w/o unimodal init	75.55	78.29	67.32	73.72
6 FLAVA	**78.19**	**79.44**	**69.92**	**75.85**

Table 3. Our full FLAVA pretraining (row 6) achieves the best average scores on vision, language, and multimodal tasks compared to ablations. Row 1 to 4 are pretrained on PMD while row 5 and 6 also involve unimodal IN-1k, CCNews, and BookCorpus datasets.

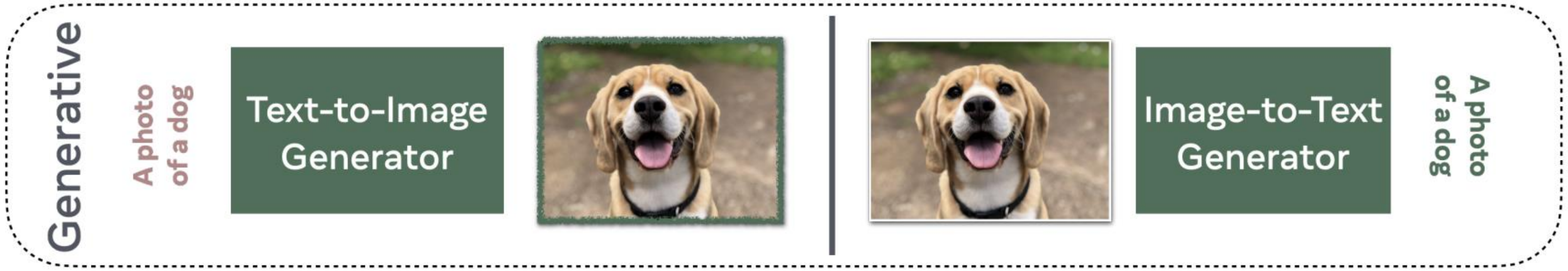
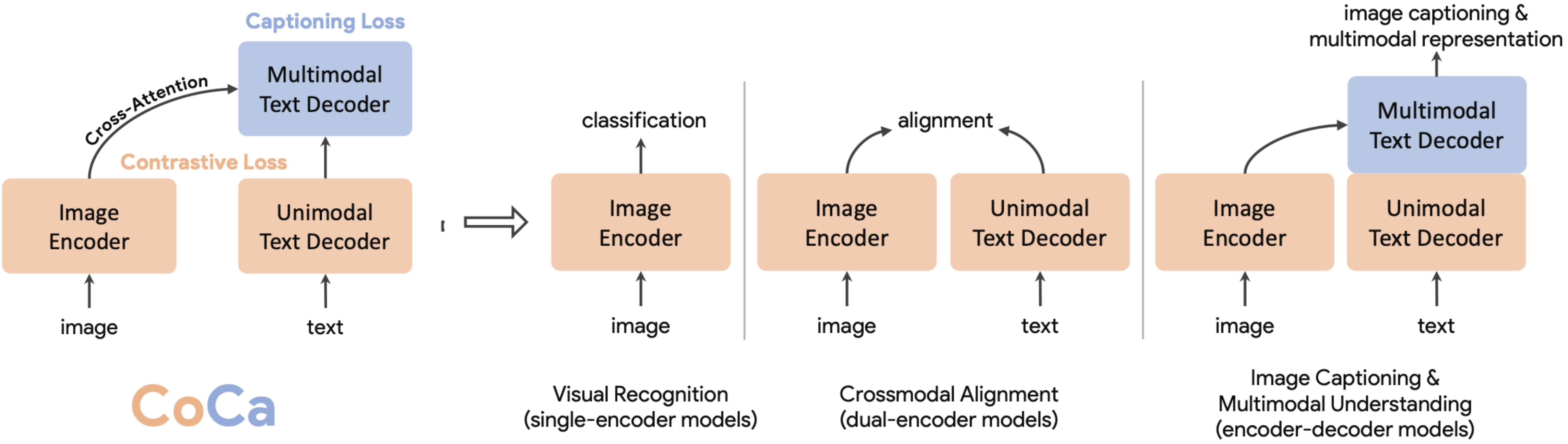


Fig: Bordes et al., “An Introduction to Vision-Language Modeling”, 2024.

Generative Approaches

CoCa

Contrastive Captioner (CoCa),
Yu et al., 2022.

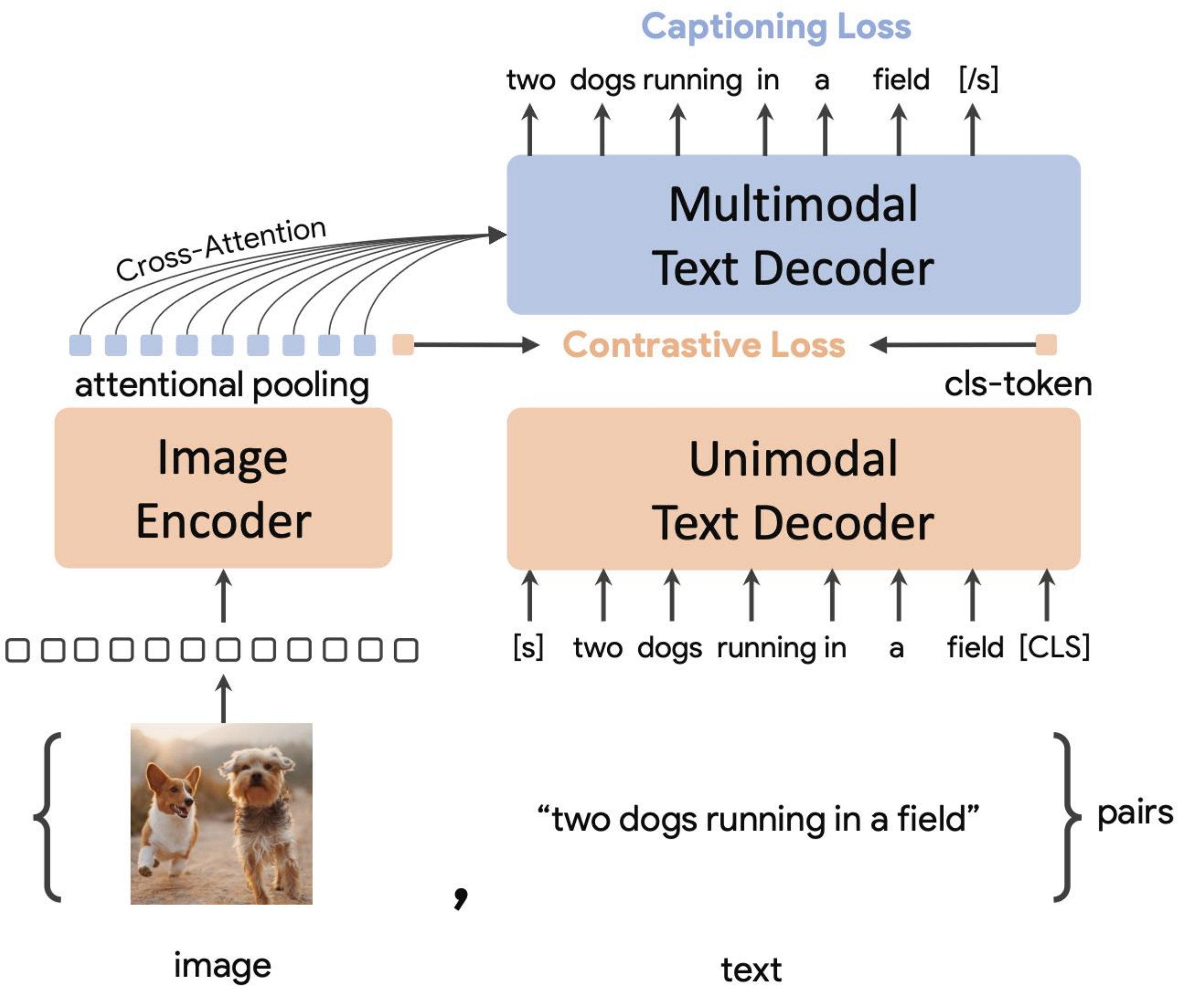


Pretraining

Zero-shot, frozen-feature or finetuning

CoCa

Contrastive Captioner (CoCa),
Yu et al., 2022.



Single-Encoder Classification. The classic single-encoder approach pretrains a visual encoder through image classification on a large crowd-sourced image annotation dataset (*e.g.*, ImageNet [9], Instagram [20] or JFT [21]), where the vocabulary of annotation texts is usually fixed. These image annotations are usually mapped into discrete class vectors to learn with a cross-entropy loss as

$$\mathcal{L}_{\text{Cls}} = -p(y)\log q_\theta(x), \tag{1}$$

Dual-Encoder Contrastive Learning. Compared to pretraining with single-encoder classification, which requires human-annotated labels and data cleaning, the dual-encoder approach exploits noisy web-scale text descriptions and introduces a learnable text tower to encode free-form texts. The two encoders are jointly optimized by contrasting the paired text against others in the sampled batch:

$$\mathcal{L}_{\text{Con}} = -\frac{1}{N}\Big(\underbrace{\sum_i^N \log\frac{\exp(x_i^\top y_i/\sigma)}{\sum_{j=1}^N \exp(x_i^\top y_j/\sigma)}}_{\text{image-to-text}} + \underbrace{\sum_i^N \log\frac{\exp(y_i^\top x_i/\sigma)}{\sum_{j=1}^N \exp(y_i^\top x_j/\sigma)}}_{\text{text-to-image}}\Big), \tag{2}$$

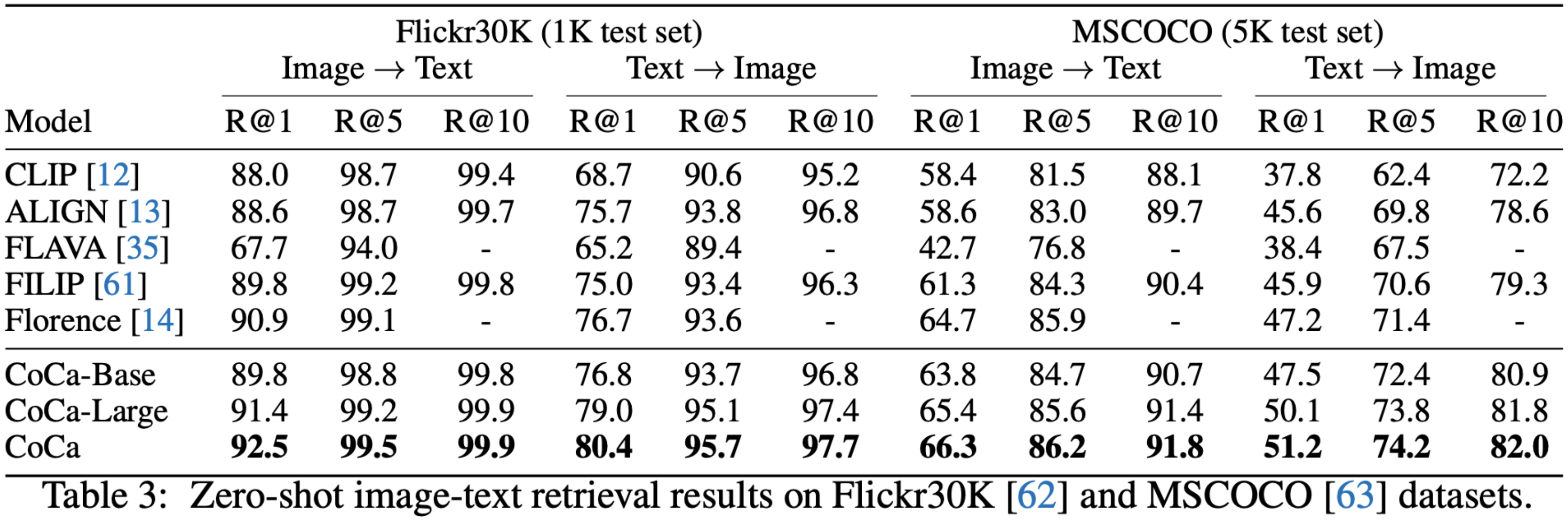
where x_i and y_j are normalized embeddings of the image in the i-th pair and that of the text in the j-th pair. N is the batch size, and σ is the temperature to scale the logits. In addition to the image

Encoder-Decoder Captioning. While the dual-encoder approach encodes the text as a whole, the generative approach (*a.k.a.* captioner) aims for detailed granularity and requires the model to predict the exact tokenized texts of y autoregressively. Following a standard encoder-decoder architecture, the image encoder provides latent encoded features (*e.g.*, using a Vision Transformer [39] or ConvNets [40]) and the text decoder learns to maximize the conditional likelihood of the paired text y under the forward autoregressive factorization:

$$\mathcal{L}_{\text{Cap}} = -\sum_{t=1}^{T}\log P_\theta(y_t|y_{<t}, x). \tag{3}$$

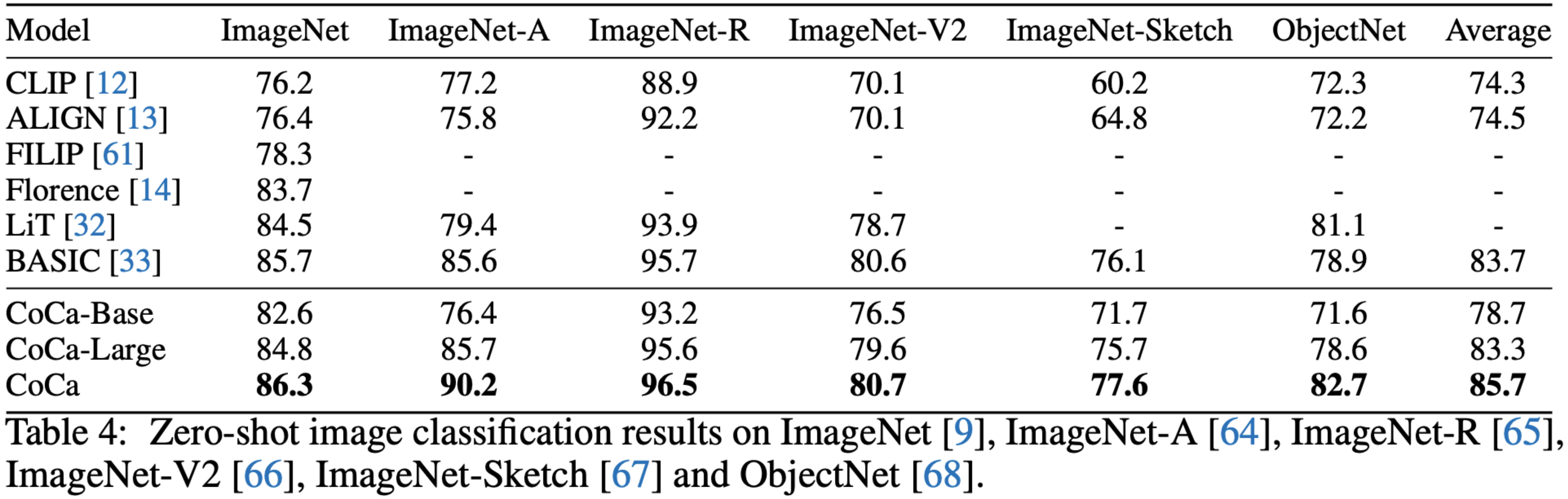
CoCa

Contrastive Captioner (CoCa),
Yu et al., 2022.



	Flickr30K (1K test set)						MSCOCO (5K test set)					
	Image → Text			Text → Image			Image → Text			Text → Image		
Model	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10
CLIP [12]	88.0	98.7	99.4	68.7	90.6	95.2	58.4	81.5	88.1	37.8	62.4	72.2
ALIGN [13]	88.6	98.7	99.7	75.7	93.8	96.8	58.6	83.0	89.7	45.6	69.8	78.6
FLAVA [35]	67.7	94.0	-	65.2	89.4	-	42.7	76.8	-	38.4	67.5	-
FILIP [61]	89.8	99.2	99.8	75.0	93.4	96.3	61.3	84.3	90.4	45.9	70.6	79.3
Florence [14]	90.9	99.1	-	76.7	93.6	-	64.7	85.9	-	47.2	71.4	-
CoCa-Base	89.8	98.8	99.8	76.8	93.7	96.8	63.8	84.7	90.7	47.5	72.4	80.9
CoCa-Large	91.4	99.2	99.9	79.0	95.1	97.4	65.4	85.6	91.4	50.1	73.8	81.8
CoCa	**92.5**	**99.5**	**99.9**	**80.4**	**95.7**	**97.7**	**66.3**	**86.2**	**91.8**	**51.2**	**74.2**	**82.0**

Table 3: Zero-shot image-text retrieval results on Flickr30K [62] and MSCOCO [63] datasets.

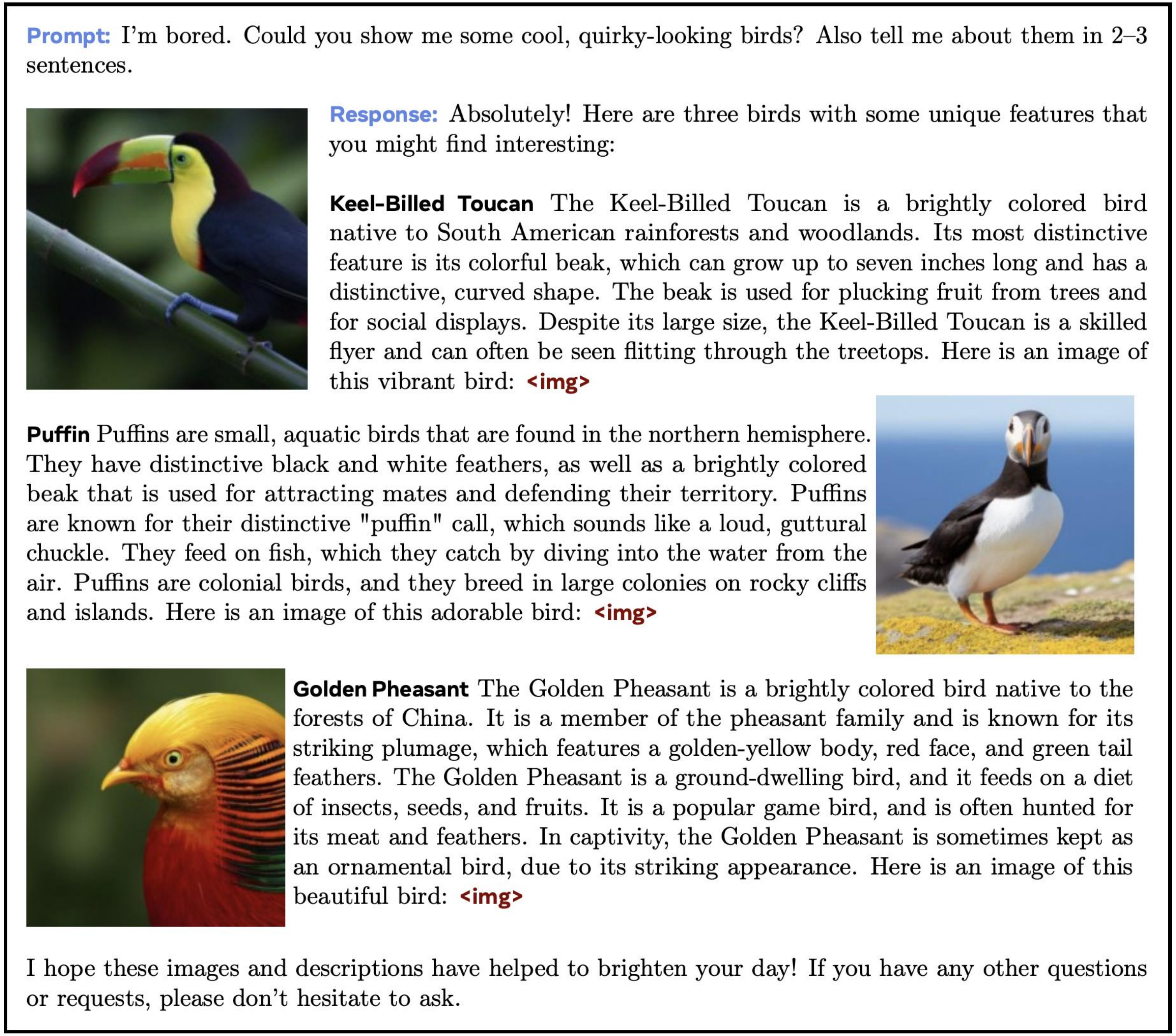


Model	ImageNet	ImageNet-A	ImageNet-R	ImageNet-V2	ImageNet-Sketch	ObjectNet	Average
CLIP [12]	76.2	77.2	88.9	70.1	60.2	72.3	74.3
ALIGN [13]	76.4	75.8	92.2	70.1	64.8	72.2	74.5
FILIP [61]	78.3	-	-	-	-	-	-
Florence [14]	83.7	-	-	-	-	-	-
LiT [32]	84.5	79.4	93.9	78.7	-	81.1	-
BASIC [33]	85.7	85.6	95.7	80.6	76.1	78.9	83.7
CoCa-Base	82.6	76.4	93.2	76.5	71.7	71.6	78.7
CoCa-Large	84.8	85.7	95.6	79.6	75.7	78.6	83.3
CoCa	**86.3**	**90.2**	**96.5**	**80.7**	**77.6**	**82.7**	**85.7**

Table 4: Zero-shot image classification results on ImageNet [9], ImageNet-A [64], ImageNet-R [65], ImageNet-V2 [66], ImageNet-Sketch [67] and ObjectNet [68].

Chameleon

Chameleon: Mixed-Modal Early-Fusion Foundation Models, Meta, 2024.



Prompt: I'm bored. Could you show me some cool, quirky-looking birds? Also tell me about them in 2–3 sentences.

Response: Absolutely! Here are three birds with some unique features that you might find interesting:

Keel-Billed Toucan The Keel-Billed Toucan is a brightly colored bird native to South American rainforests and woodlands. Its most distinctive feature is its colorful beak, which can grow up to seven inches long and has a distinctive, curved shape. The beak is used for plucking fruit from trees and for social displays. Despite its large size, the Keel-Billed Toucan is a skilled flyer and can often be seen flitting through the treetops. Here is an image of this vibrant bird: **<img>**

Puffin Puffins are small, aquatic birds that are found in the northern hemisphere. They have distinctive black and white feathers, as well as a brightly colored beak that is used for attracting mates and defending their territory. Puffins are known for their distinctive "puffin" call, which sounds like a loud, guttural chuckle. They feed on fish, which they catch by diving into the water from the air. Puffins are colonial birds, and they breed in large colonies on rocky cliffs and islands. Here is an image of this adorable bird: **<img>**

Golden Pheasant The Golden Pheasant is a brightly colored bird native to the forests of China. It is a member of the pheasant family and is known for its striking plumage, which features a golden-yellow body, red face, and green tail feathers. The Golden Pheasant is a ground-dwelling bird, and it feeds on a diet of insects, seeds, and fruits. It is a popular game bird, and is often hunted for its meat and feathers. In captivity, the Golden Pheasant is sometimes kept as an ornamental bird, due to its striking appearance. Here is an image of this beautiful bird: **<img>**

I hope these images and descriptions have helped to brighten your day! If you have any other questions or requests, please don't hesitate to ask.

Chameleon

Chameleon: Mixed-Modal Early-Fusion Foundation Models, Meta, 2024.

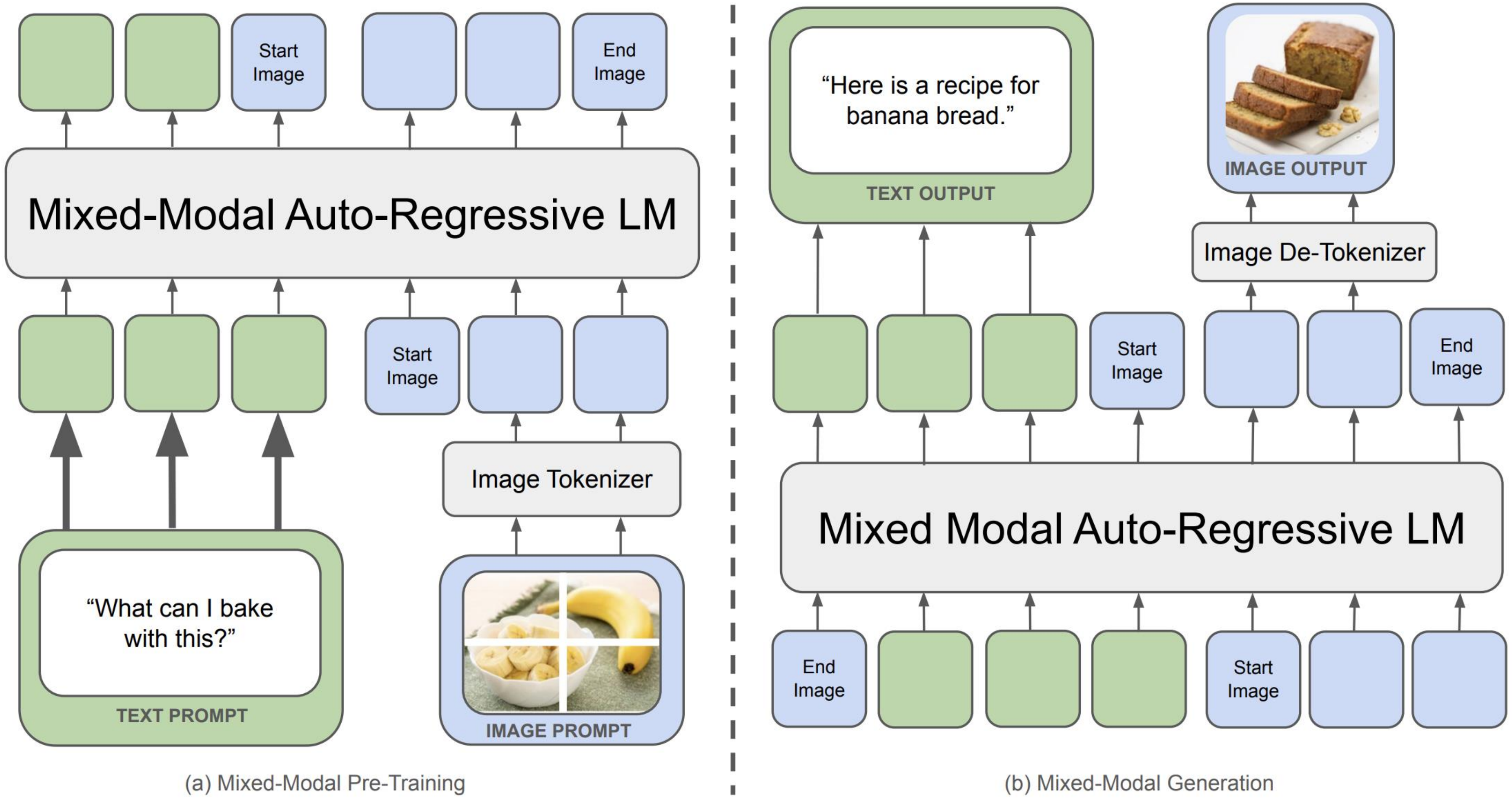
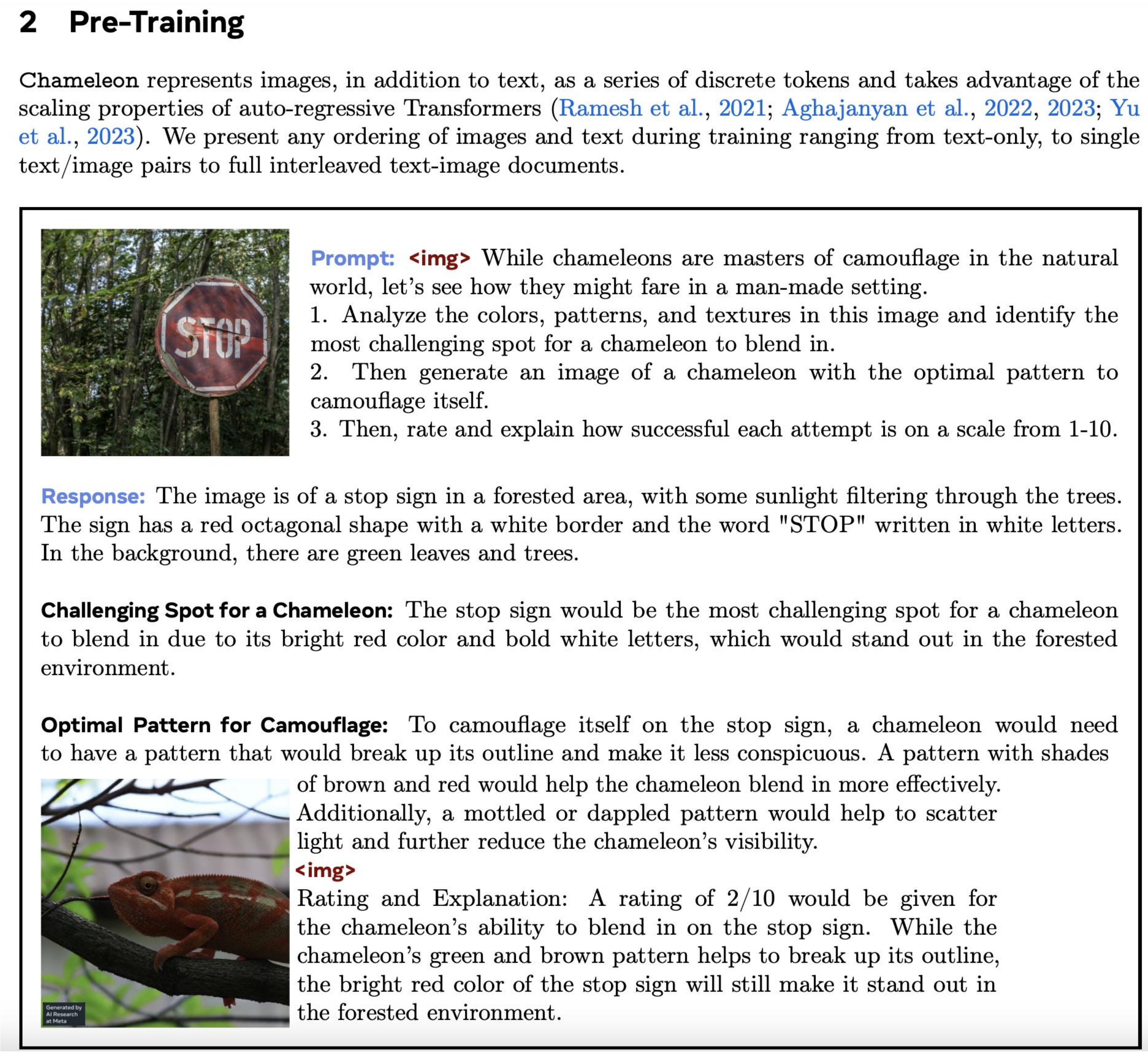

Start Image
End Image
Mixed-Modal Auto-Regressive LM
Start Image
Image Tokenizer
"What can I bake with this?"
TEXT PROMPT
IMAGE PROMPT
(a) Mixed-Modal Pre-Training
"Here is a recipe for banana bread."
TEXT OUTPUT
IMAGE OUTPUT
Image De-Tokenizer
Start Image
End Image
Mixed Modal Auto-Regressive LM
End Image
Start Image
(b) Mixed-Modal Generation

Figure 1 **Chameleon** represents all modalities — images, text, and code, as discrete tokens and uses a uniform transformer-based architecture that is trained from scratch in an end-to-end fashion on ~10T tokens of interleaved mixed-modal data. As a result, **Chameleon** can both reason over, as well as generate, arbitrary mixed-modal documents. Text tokens are represented in green and image tokens are represented in blue.

Chameleon



2 Pre-Training

Chameleon represents images, in addition to text, as a series of discrete tokens and takes advantage of the scaling properties of auto-regressive Transformers (Ramesh et al., 2021; Aghajanyan et al., 2022, 2023; Yu et al., 2023). We present any ordering of images and text during training ranging from text-only, to single text/image pairs to full interleaved text-image documents.

Prompt: **<img>** While chameleons are masters of camouflage in the natural world, let's see how they might fare in a man-made setting.
1. Analyze the colors, patterns, and textures in this image and identify the most challenging spot for a chameleon to blend in.
2. Then generate an image of a chameleon with the optimal pattern to camouflage itself.
3. Then, rate and explain how successful each attempt is on a scale from 1-10.

Response: The image is of a stop sign in a forested area, with some sunlight filtering through the trees. The sign has a red octagonal shape with a white border and the word "STOP" written in white letters. In the background, there are green leaves and trees.

Challenging Spot for a Chameleon: The stop sign would be the most challenging spot for a chameleon to blend in due to its bright red color and bold white letters, which would stand out in the forested environment.

Optimal Pattern for Camouflage: To camouflage itself on the stop sign, a chameleon would need to have a pattern that would break up its outline and make it less conspicuous. A pattern with shades of brown and red would help the chameleon blend in more effectively. Additionally, a mottled or dappled pattern would help to scatter light and further reduce the chameleon's visibility.

<img>

Rating and Explanation: A rating of 2/10 would be given for the chameleon's ability to blend in on the stop sign. While the chameleon's green and brown pattern helps to break up its outline, the bright red color of the stop sign will still make it stand out in the forested environment.

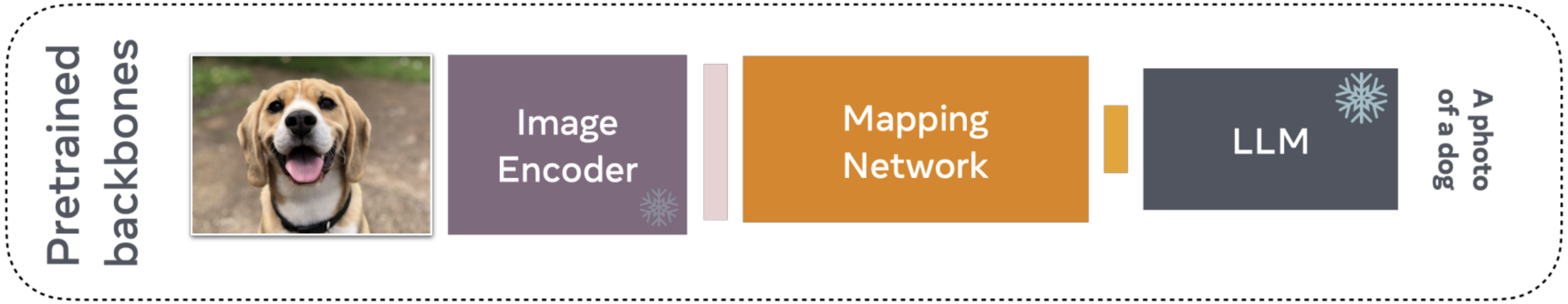
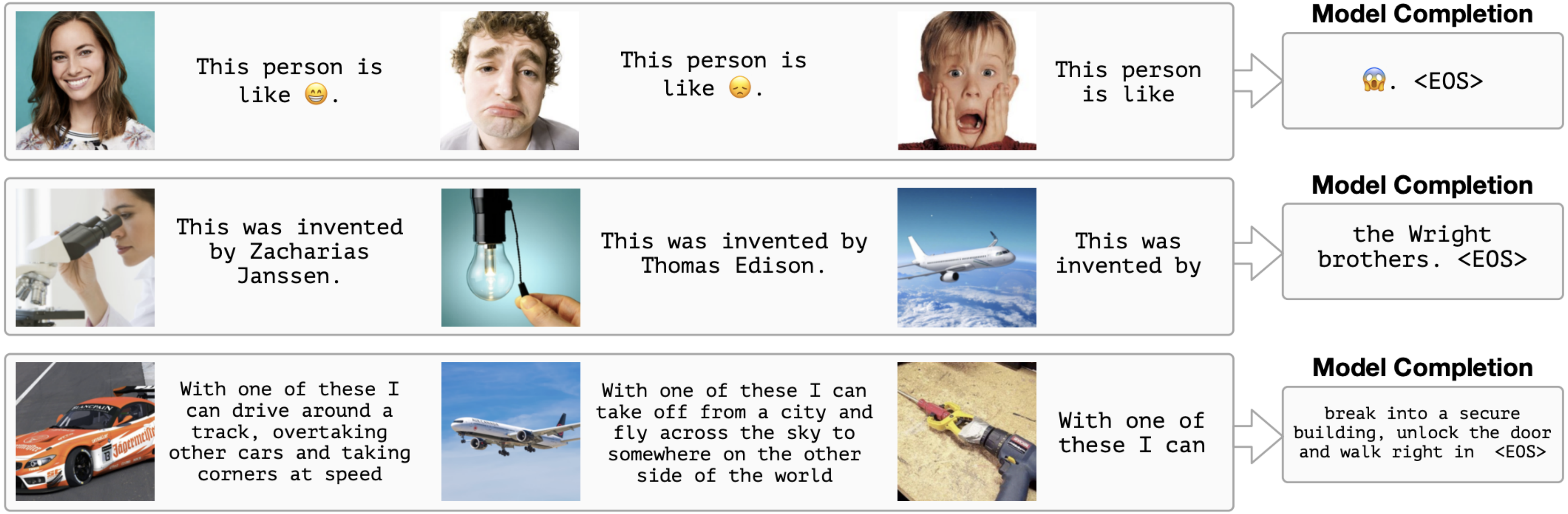


Fig: Bordes et al., “An Introduction to Vision-Language Modeling”, 2024.

Approaches Using Pre-trained Backbones

Frozen

Multimodal Few-Shot Learning with Frozen Language Models, 2021


This person is like 😁.

This person is like 😞.

This person is like

😱. <EOS>

This was invented by Zacharias Janssen.

This was invented by Thomas Edison.

This was invented by

the Wright brothers. <EOS>

With one of these I can drive around a track, overtaking other cars and taking corners at speed

With one of these I can take off from a city and fly across the sky to somewhere on the other side of the world

With one of these I can

break into a secure building, unlock the door and walk right in <EOS>

Figure 1: Curated samples with about five seeds required to get past well-known language model failure modes of either repeating text for the prompt or emitting text that does not pertain to the image. These samples demonstrate the ability to generate open-ended outputs that adapt to both images and text, and to make use of facts that it has learned during language-only pre-training.

Frozen

Multimodal Few-Shot Learning with
Frozen Language Models, 2021

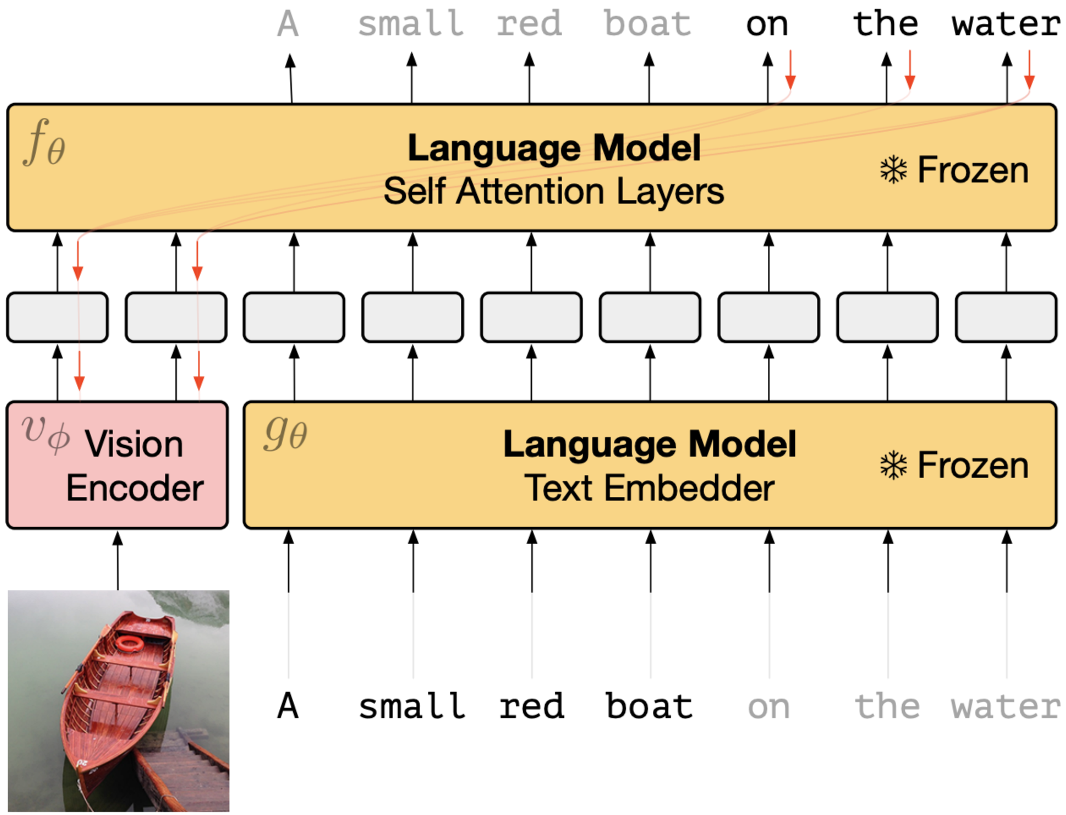
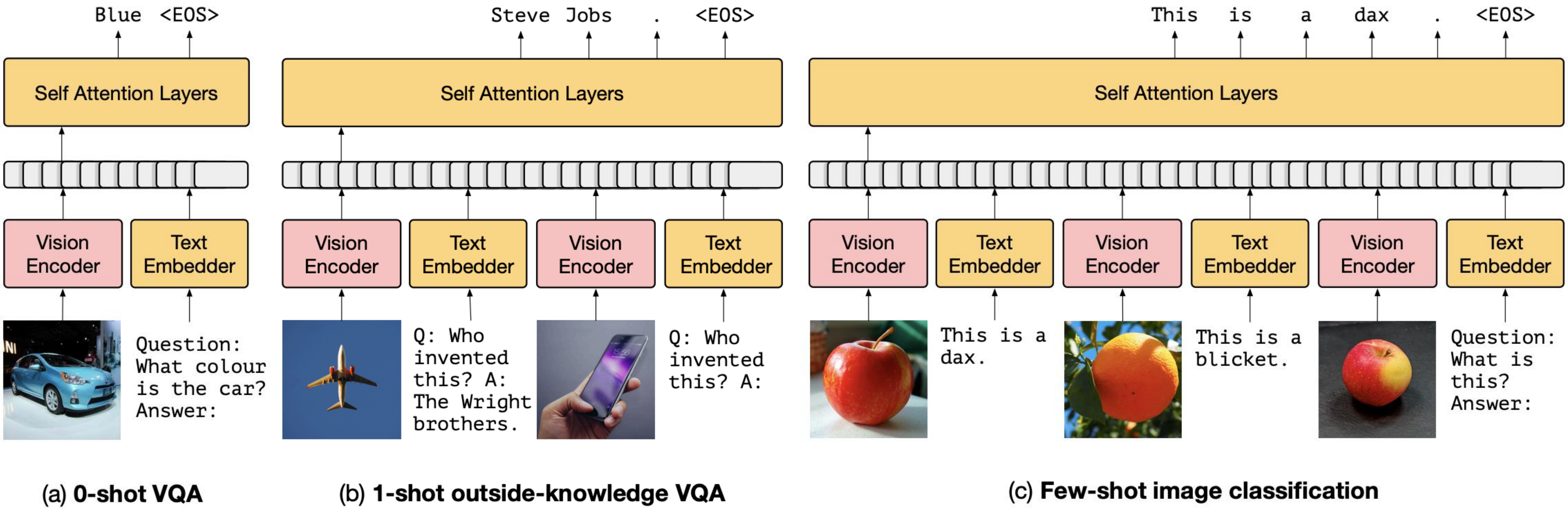


Figure 2: Gradients through a frozen language model's self attention layers are used to train the vision encoder.

Frozen Multimodal Few-Shot Learning with Frozen Language Models, 2021

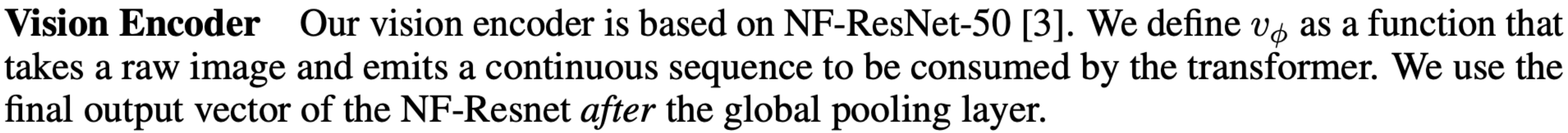


(a) **0-shot VQA** (b) **1-shot outside-knowledge VQA** (c) **Few-shot image classification**

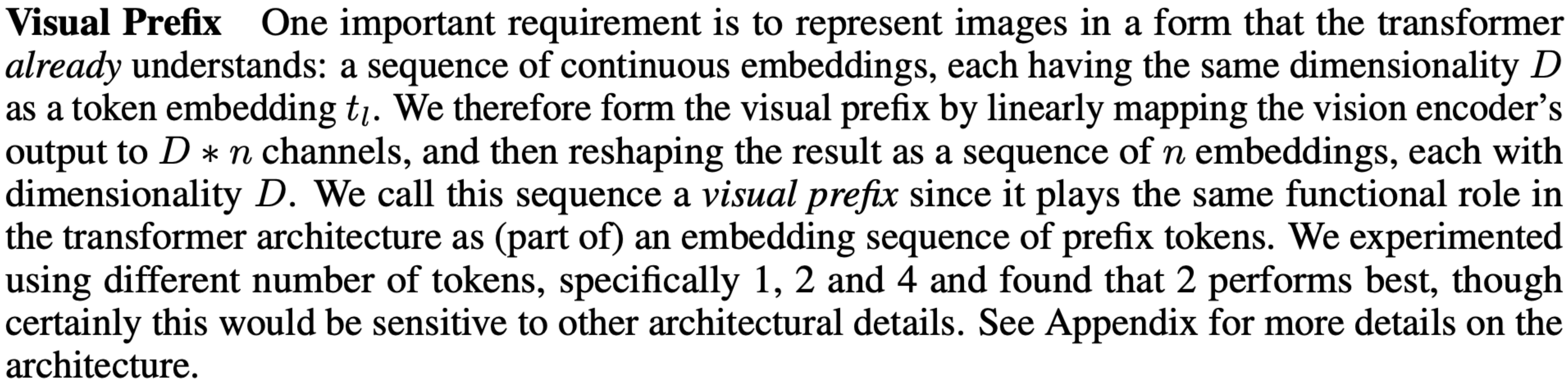
Figure 3: Inference-Time interface for *Frozen*. The figure demonstrates how we can support (a) visual question answering, (b) outside-knowledge question answering and (c) few-shot image classification via in-context learning.

Frozen

Multimodal Few-Shot Learning with
Frozen Language Models, 2021



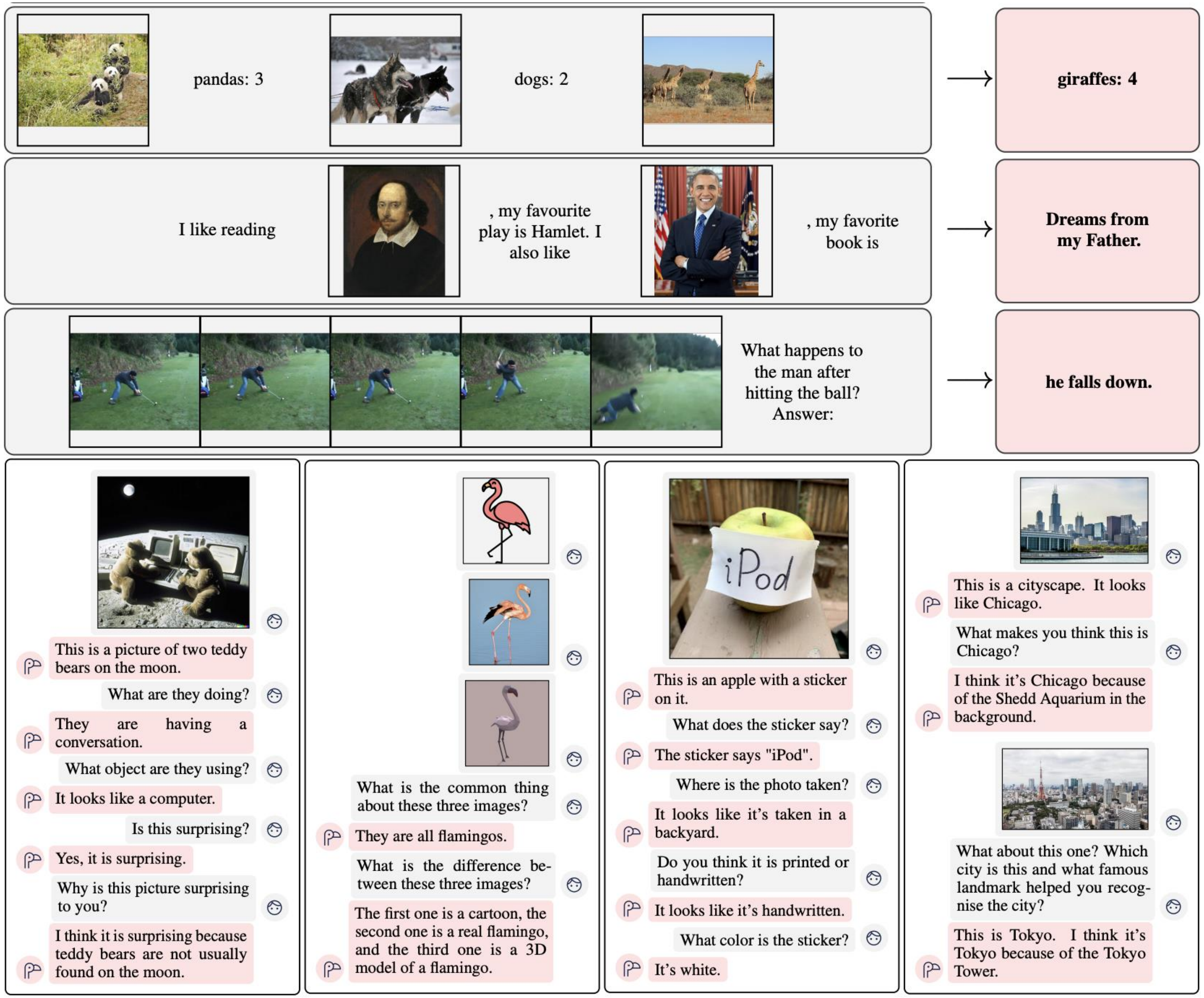
Vision Encoder Our vision encoder is based on NF-ResNet-50 [3]. We define v_ϕ as a function that takes a raw image and emits a continuous sequence to be consumed by the transformer. We use the final output vector of the NF-Resnet *after* the global pooling layer.



Visual Prefix One important requirement is to represent images in a form that the transformer *already* understands: a sequence of continuous embeddings, each having the same dimensionality D as a token embedding t_l. We therefore form the visual prefix by linearly mapping the vision encoder's output to $D * n$ channels, and then reshaping the result as a sequence of n embeddings, each with dimensionality D. We call this sequence a *visual prefix* since it plays the same functional role in the transformer architecture as (part of) an embedding sequence of prefix tokens. We experimented using different number of tokens, specifically 1, 2 and 4 and found that 2 performs best, though certainly this would be sensitive to other architectural details. See Appendix for more details on the architecture.

Flamingo

Flamingo: a Visual Language Model
for Few-Shot Learning, Deepmind, 2022



pandas: 3

dogs: 2

→ **giraffes: 4**

I like reading

, my favourite play is Hamlet. I also like

, my favorite book is

→ **Dreams from my Father.**

What happens to the man after hitting the ball? Answer:

→ **he falls down.**

This is a picture of two teddy bears on the moon.

What are they doing?

They are having a conversation.

What object are they using?

It looks like a computer.

Is this surprising?

Yes, it is surprising.

Why is this picture surprising to you?

I think it is surprising because teddy bears are not usually found on the moon.

What is the common thing about these three images?

They are all flamingos.

What is the difference between these three images?

The first one is a cartoon, the second one is a real flamingo, and the third one is a 3D model of a flamingo.

This is an apple with a sticker on it.

What does the sticker say?

The sticker says "iPod".

Where is the photo taken?

It looks like it's taken in a backyard.

Do you think it is printed or handwritten?

It looks like it's handwritten.

What color is the sticker?

It's white.

This is a cityscape. It looks like Chicago.

What makes you think this is Chicago?

I think it's Chicago because of the Shedd Aquarium in the background.

What about this one? Which city is this and what famous landmark helped you recognise the city?

This is Tokyo. I think it's Tokyo because of the Tokyo Tower.

Flamingo

Flamingo: a Visual Language Model
for Few-Shot Learning, Deepmind, 2022

Vision Encoder: Normalizer-Free ResNet (NFNet)
Perceiver Sampler: Fixed # of queries attend to variable length of visual tokens.
LLM: Chinchilla

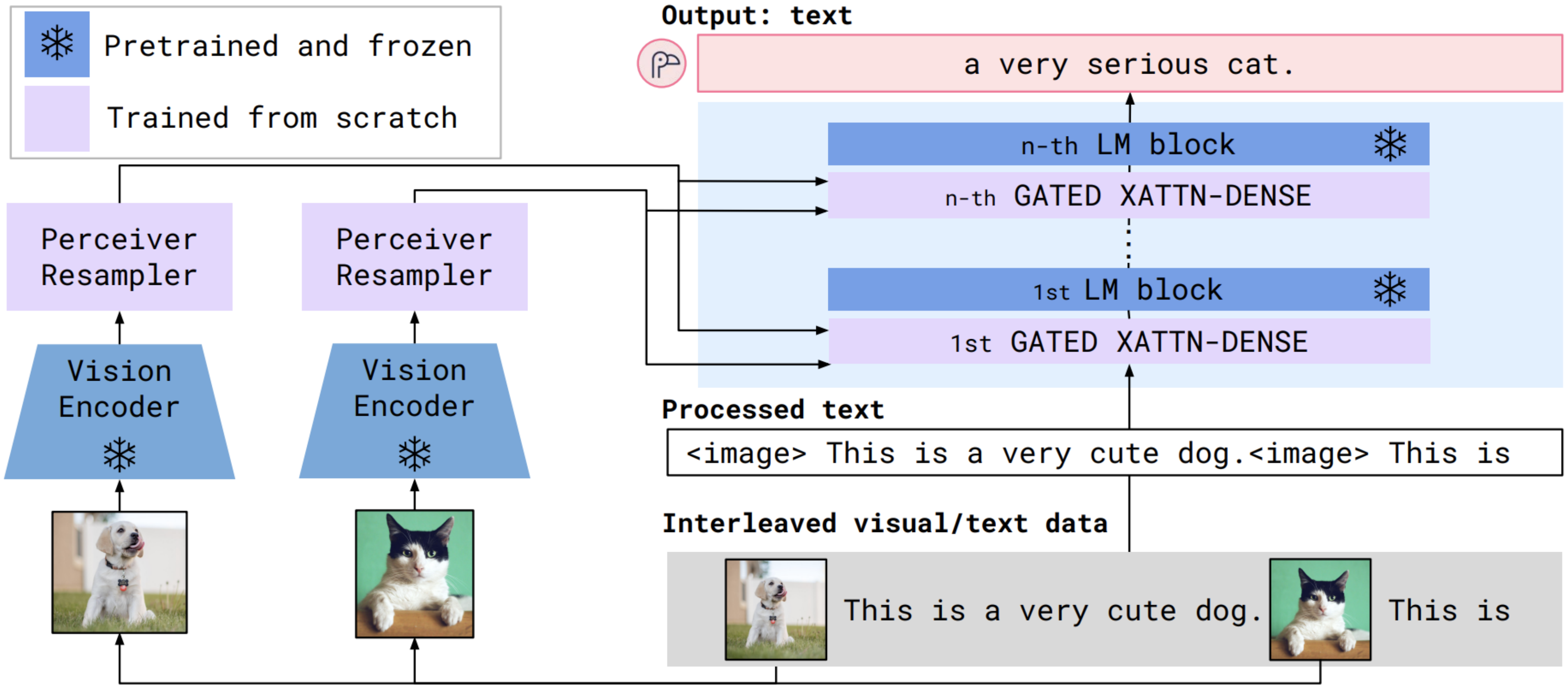
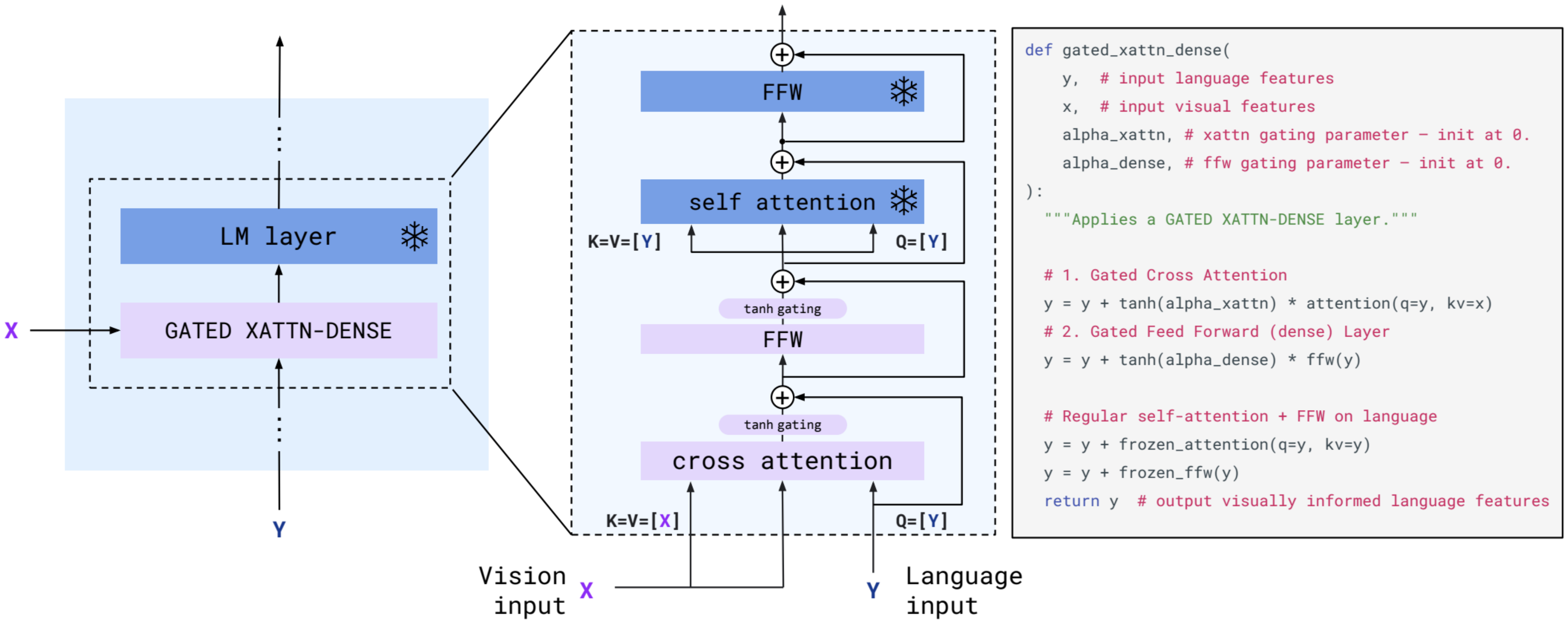


Figure 3: **Flamingo architecture overview.** Flamingo is a family of visual language models (VLMs) that take as input visual data interleaved with text and produce free-form text as output.

Flamingo

Flamingo: a Visual Language Model
for Few-Shot Learning, Deepmind, 2022



```
def gated_xattn_dense(
    y,  # input language features
    x,  # input visual features
    alpha_xattn, # xattn gating parameter – init at 0.
    alpha_dense, # ffw gating parameter – init at 0.
):
  """Applies a GATED XATTN-DENSE layer."""

  # 1. Gated Cross Attention
  y = y + tanh(alpha_xattn) * attention(q=y, kv=x)
  # 2. Gated Feed Forward (dense) Layer
  y = y + tanh(alpha_dense) * ffw(y)

  # Regular self-attention + FFW on language
  y = y + frozen_attention(q=y, kv=y)
  y = y + frozen_ffw(y)
  return y  # output visually informed language features
```

Figure 4: **GATED XATTN-DENSE layers.** To condition the LM on visual inputs, we insert new cross-attention layers between existing pretrained and frozen LM layers. The keys and values in these layers are obtained from the vision features while the queries are derived from the language inputs. They are followed by dense feed-forward layers. These layers are *gated* so that the LM is kept intact at initialization for improved stability and performance.

Flamingo

Flamingo: a Visual Language Model
for Few-Shot Learning, Deepmind, 2022

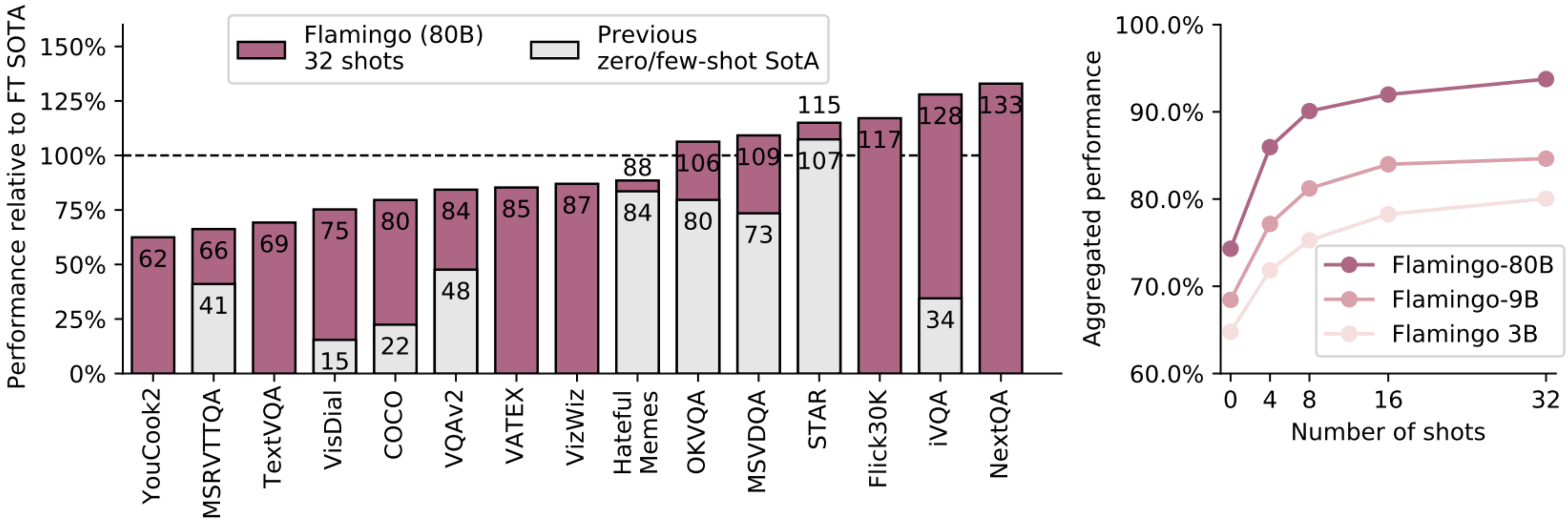


Figure 2: **Flamingo results overview.** *Left*: Our largest model, dubbed *Flamingo*, outperforms state-of-the-art fine-tuned models on 6 of the 16 tasks we consider with no fine-tuning. For the 9 tasks with published few-shot results, *Flamingo* sets the new few-shot state of the art. *Note:* We omit 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.

BLIP

BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation, Salesforce, 2022.

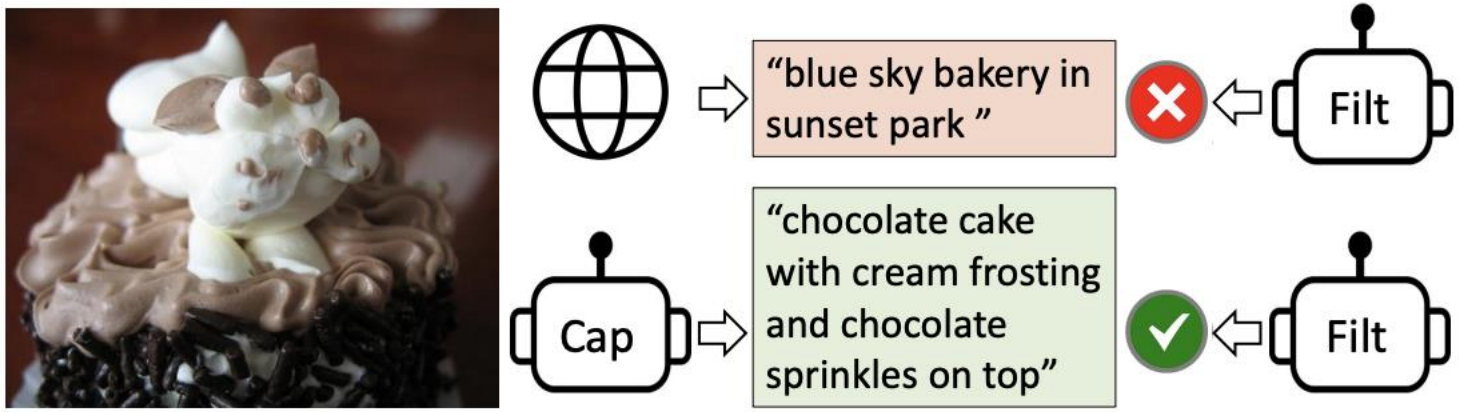

“blue sky bakery in sunset park ”
Filt
Cap
“chocolate cake with cream frosting and chocolate sprinkles on top”
Filt

Figure 1. We use a Captioner (Cap) to generate synthetic captions for web images, and a Filter (Filt) to remove noisy captions.

BLIP

BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation, Salesforce, 2022.

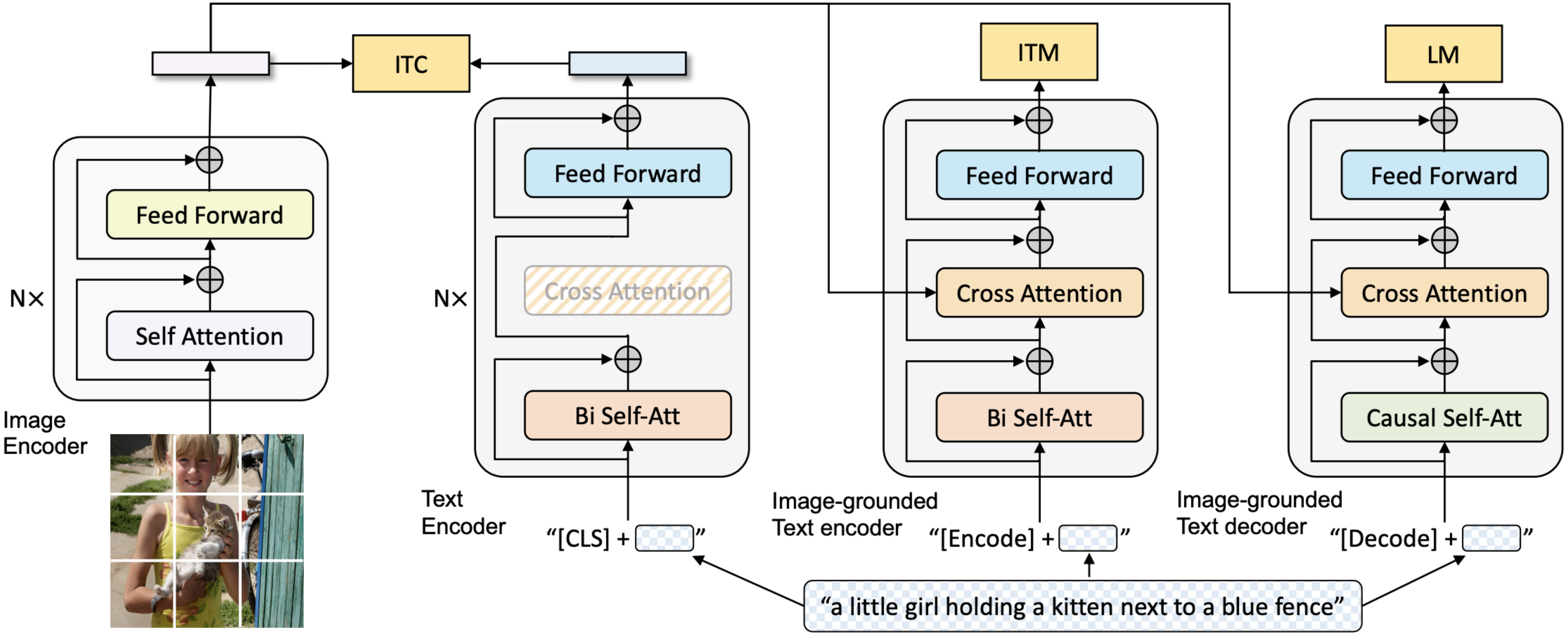


Figure 2. Pre-training model architecture and objectives of BLIP (same parameters have the same color). We propose multimodal mixture of encoder-decoder, a unified vision-language model which can operate in one of the three functionalities: (1) Unimodal encoder is trained with an image-text contrastive (ITC) loss to align the vision and language representations. (2) Image-grounded text encoder uses additional cross-attention layers to model vision-language interactions, and is trained with a image-text matching (ITM) loss to distinguish between positive and negative image-text pairs. (3) Image-grounded text decoder replaces the bi-directional self-attention layers with causal self-attention layers, and shares the same cross-attention layers and feed forward networks as the encoder. The decoder is trained with a language modeling (LM) loss to generate captions given images.

BLIP

BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation, Salesforce, 2022.

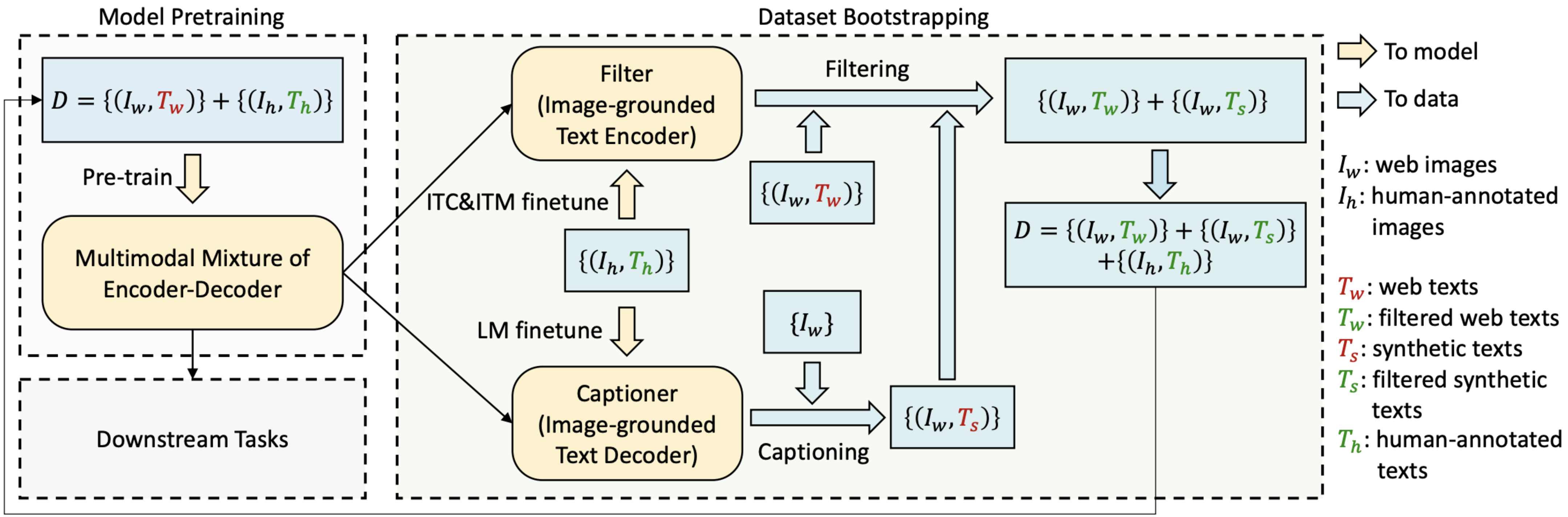

Model Pretraining
$D = \{(I_w, T_w)\} + \{(I_h, T_h)\}$
Pre-train
Multimodal Mixture of Encoder-Decoder
Downstream Tasks
Dataset Bootstrapping
Filter (Image-grounded Text Encoder)
ITC&ITM finetune
$\{(I_h, T_h)\}$
LM finetune
Captioner (Image-grounded Text Decoder)
Filtering
$\{(I_w, T_w)\}$
$\{I_w\}$
Captioning
$\{(I_w, T_s)\}$
$\{(I_w, T_w)\} + \{(I_w, T_s)\}$
$D = \{(I_w, T_w)\} + \{(I_w, T_s)\} + \{(I_h, T_h)\}$
To model
To data
I_w: web images
I_h: human-annotated images
T_w: web texts
T_w: filtered web texts
T_s: synthetic texts
T_s: filtered synthetic texts
T_h: human-annotated texts

Figure 3. Learning framework of BLIP. We introduce a captioner to produce synthetic captions for web images, and a filter to remove noisy image-text pairs. The captioner and filter are initialized from the same pre-trained model and finetuned individually on a small-scale human-annotated dataset. The bootstrapped dataset is used to pre-train a new model.

BLIP-2

BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models, Salesforce, 2023.

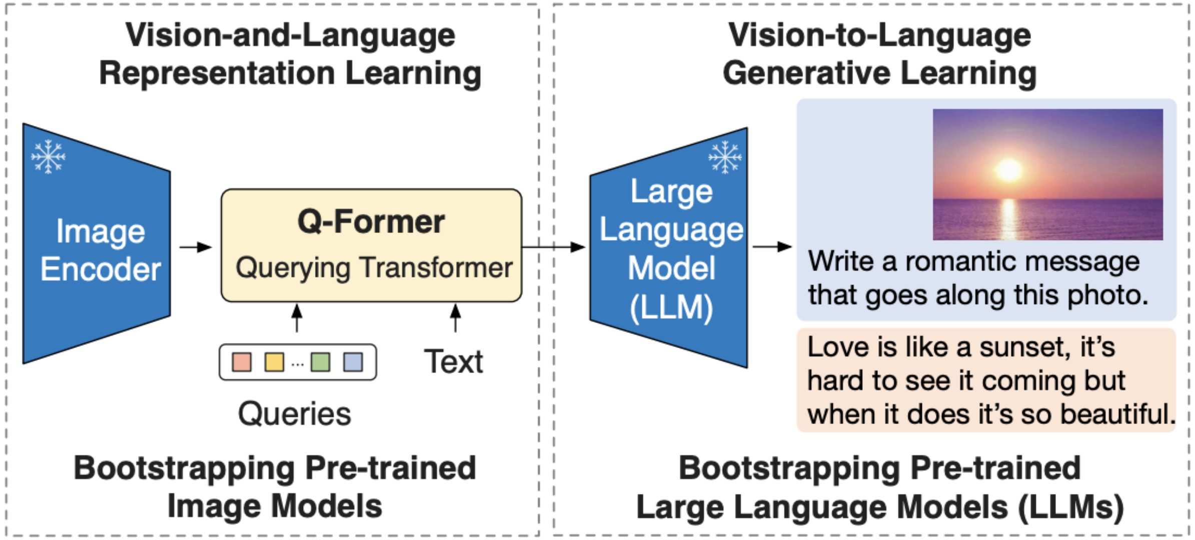

Vision-and-Language Representation Learning
Image Encoder
Q-Former
Querying Transformer
Queries
Text
Bootstrapping Pre-trained Image Models
Vision-to-Language Generative Learning
Large Language Model (LLM)
Write a romantic message that goes along this photo.
Love is like a sunset, it's hard to see it coming but when it does it's so beautiful.
Bootstrapping Pre-trained Large Language Models (LLMs)

Figure 1. Overview of BLIP-2's framework. We pre-train a lightweight Querying Transformer following a two-stage strategy to bridge the modality gap. The first stage bootstraps vision-language representation learning from a frozen image encoder. The second stage bootstraps vision-to-language generative learning from a frozen LLM, which enables zero-shot instructed image-to-text generation (see Figure 4 for more examples).

BLIP-2

BLIP-2: Bootstrapping Language-Image Pre-training with Frozen
Image Encoders and Large Language Models, Salesforce, 2023.

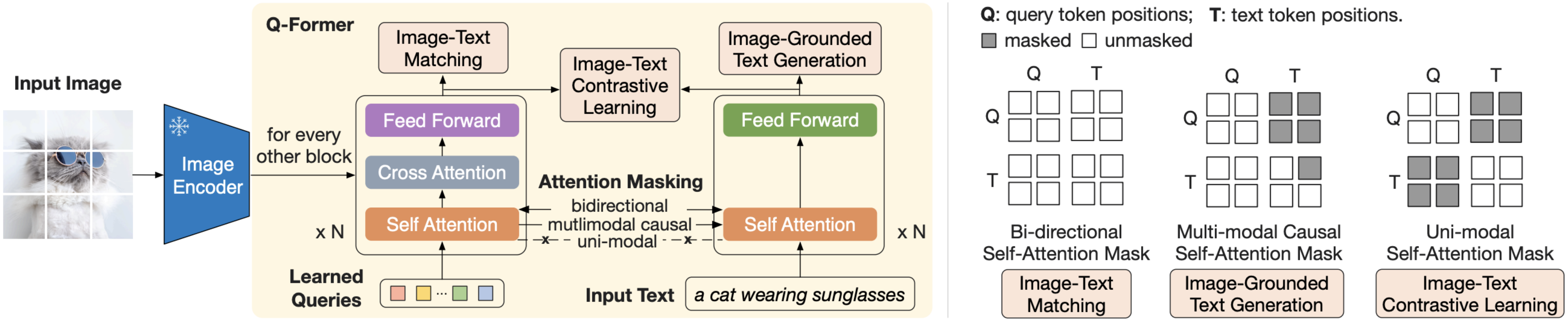


Figure 2. (**Left**) Model architecture of Q-Former and BLIP-2's first-stage vision-language representation learning objectives. We jointly optimize three objectives which enforce the queries (a set of learnable embeddings) to extract visual representation most relevant to the text. (**Right**) The self-attention masking strategy for each objective to control query-text interaction.

BLIP-2

BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models, Salesforce, 2023.

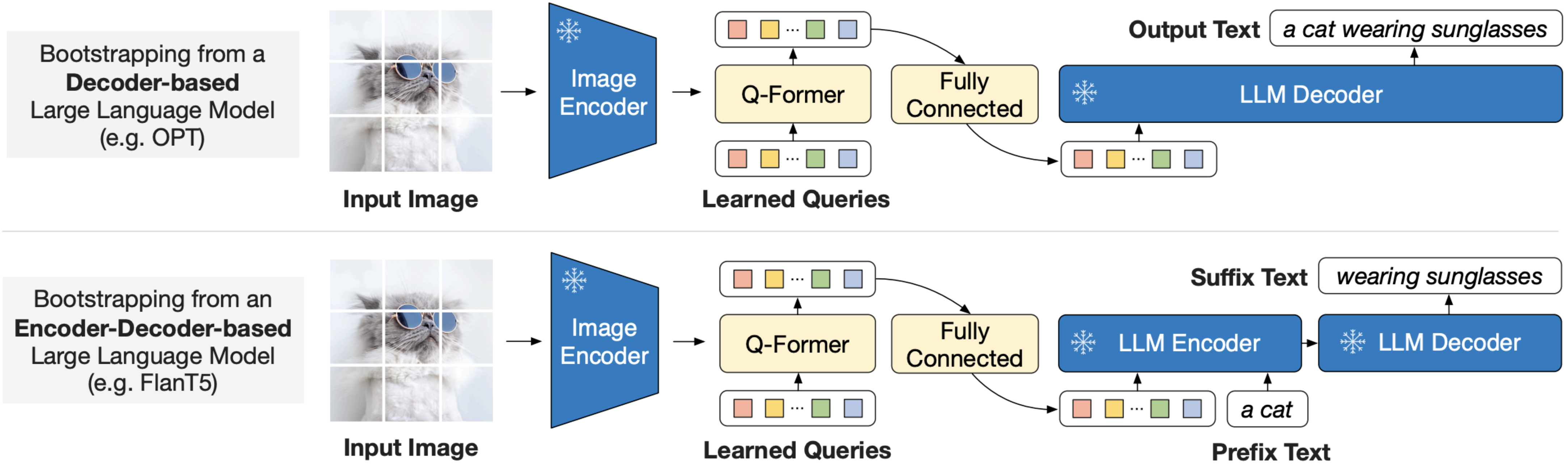
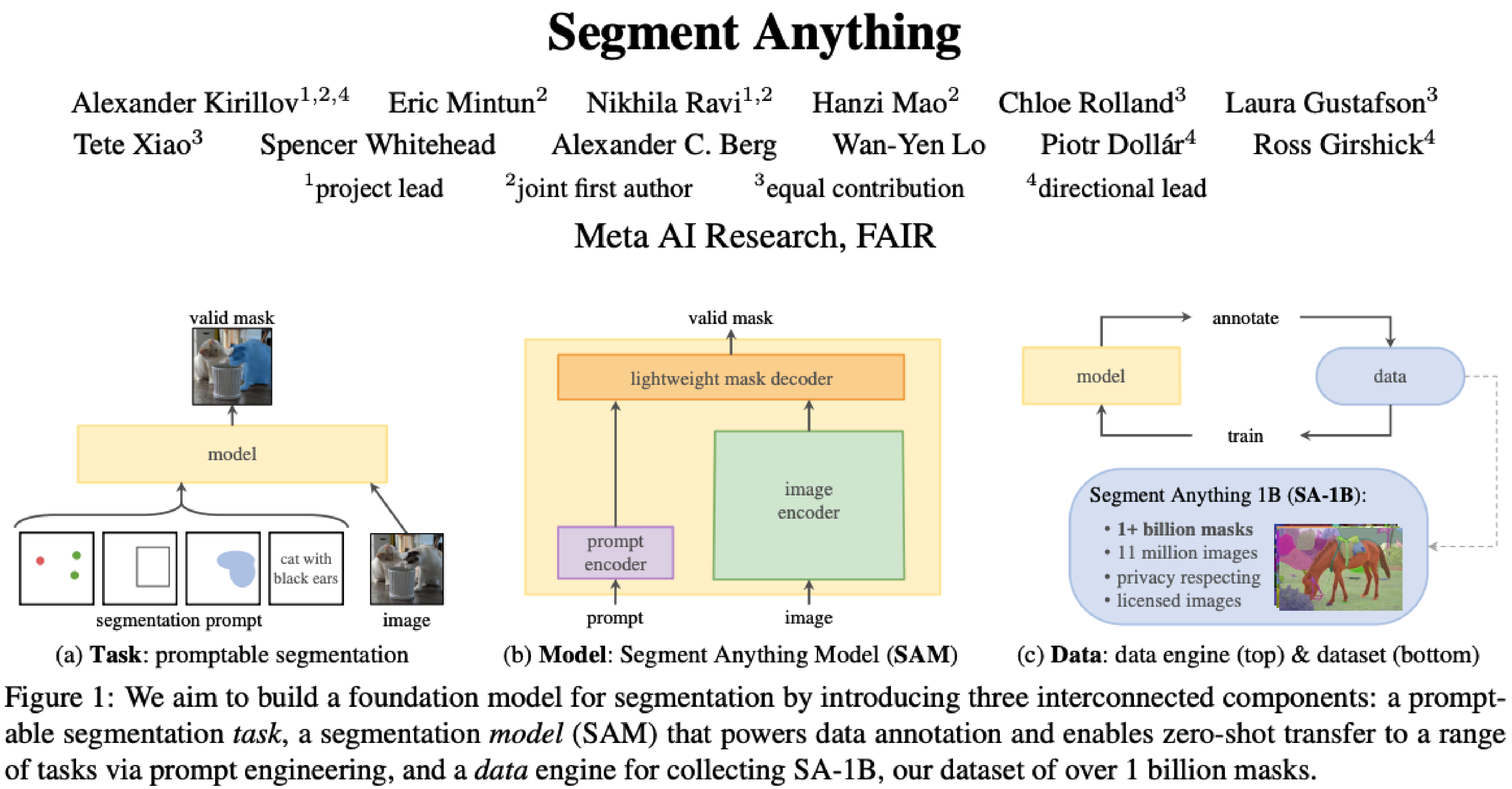


Figure 3. BLIP-2's second-stage vision-to-language generative pre-training, which bootstraps from frozen large language models (LLMs). (**Top**) Bootstrapping a decoder-based LLM (e.g. OPT). (**Bottom**) Bootstrapping an encoder-decoder-based LLM (e.g. FlanT5). The fully-connected layer adapts from the output dimension of the Q-Former to the input dimension of the chosen LLM.

Misc

Segment Anything Model (SAM) 2023



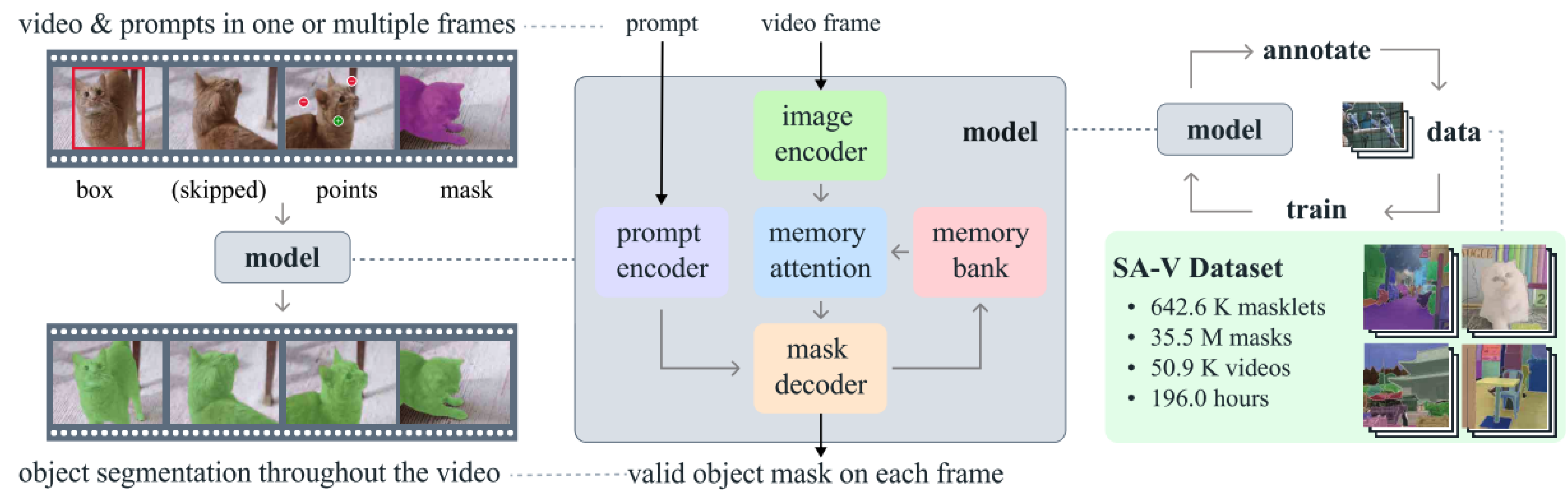
Segment Anything

Alexander Kirillov[1,2,4] Eric Mintun[2] Nikhila Ravi[1,2] Hanzi Mao[2] Chloe Rolland[3] Laura Gustafson[3]
Tete Xiao[3] Spencer Whitehead Alexander C. Berg Wan-Yen Lo Piotr Dollár[4] Ross Girshick[4]
[1]project lead [2]joint first author [3]equal contribution [4]directional lead
Meta AI Research, FAIR

(a) **Task**: promptable segmentation (b) **Model**: Segment Anything Model (**SAM**) (c) **Data**: data engine (top) & dataset (bottom)

Figure 1: We aim to build a foundation model for segmentation by introducing three interconnected components: a promptable segmentation *task*, a segmentation *model* (SAM) that powers data annotation and enables zero-shot transfer to a range of tasks via prompt engineering, and a *data* engine for collecting SA-1B, our dataset of over 1 billion masks.

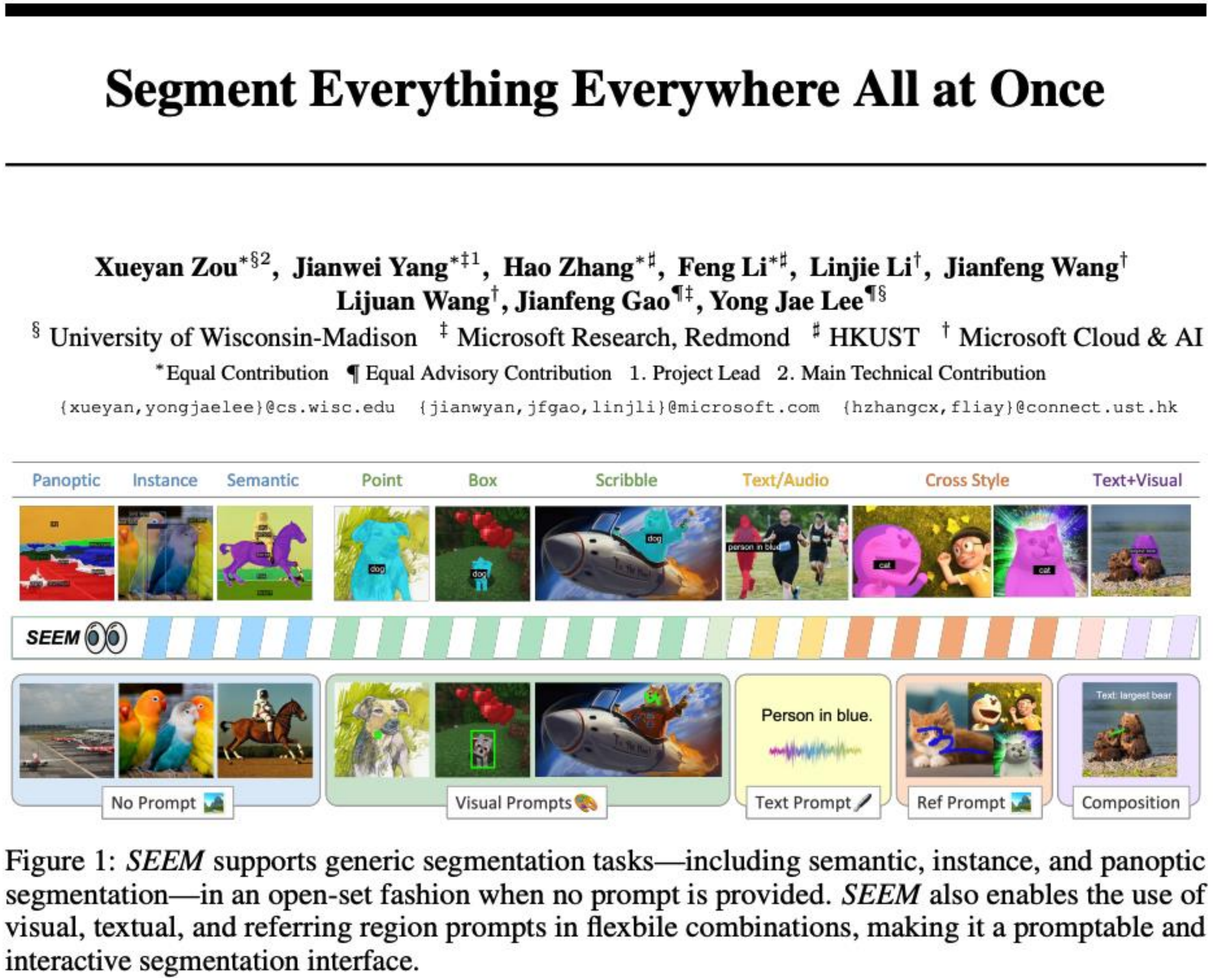
Segment Anything Model (SAM) v2 2024



(a) Task: promptable visual segmentation **(b) Model:** Segment Anything Model 2 **(c) Data:** data engine and dataset

Figure 1 We introduce the Segment Anything Model 2 (SAM 2), towards solving the promptable visual segmentation task (a) with our foundation model (b), trained on our large-scale SA-V dataset collected through our data engine (c). SAM 2 is capable of interactively segmenting regions through prompts (clicks, boxes, or masks) on one or multiple video frames by utilizing a streaming memory that stores previous prompts and predictions.

Segment Everything Everywhere All At Once 2023



Segment Everything Everywhere All at Once

Xueyan Zou[*§2], Jianwei Yang[*‡1], Hao Zhang[*♯], Feng Li[*♯], Linjie Li[†], Jianfeng Wang[†]
Lijuan Wang[†], Jianfeng Gao[¶‡], Yong Jae Lee[¶§]
[§] University of Wisconsin-Madison [‡] Microsoft Research, Redmond [♯] HKUST [†] Microsoft Cloud & AI
[*] Equal Contribution ¶ Equal Advisory Contribution 1. Project Lead 2. Main Technical Contribution
{xueyan,yongjaelee}@cs.wisc.edu {jianwyan,jfgao,linjli}@microsoft.com {hzhangcx,fliay}@connect.ust.hk

Figure 1: *SEEM* supports generic segmentation tasks—including semantic, instance, and panoptic segmentation—in an open-set fashion when no prompt is provided. *SEEM* also enables the use of visual, textual, and referring region prompts in flexbile combinations, making it a promptable and interactive segmentation interface.

INTERNVIDEO2: SCALING FOUNDATION MODELS FOR MULTIMODAL VIDEO UNDERSTANDING

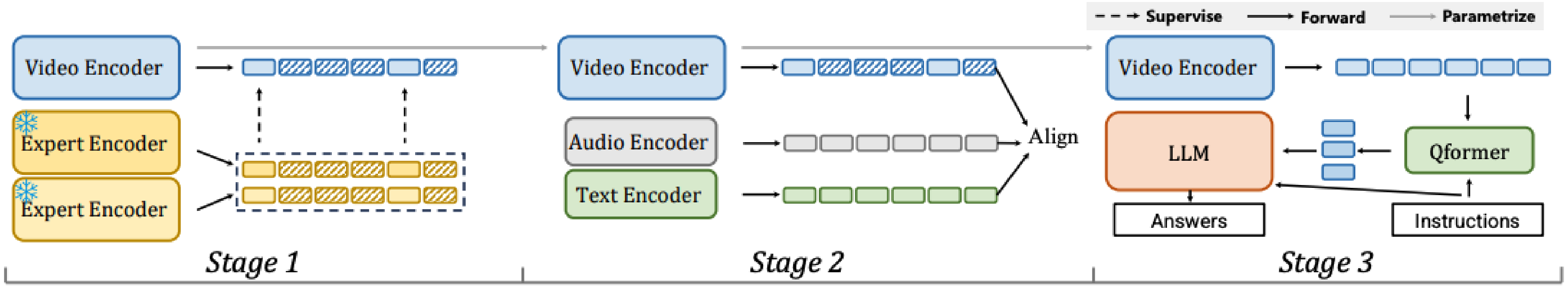
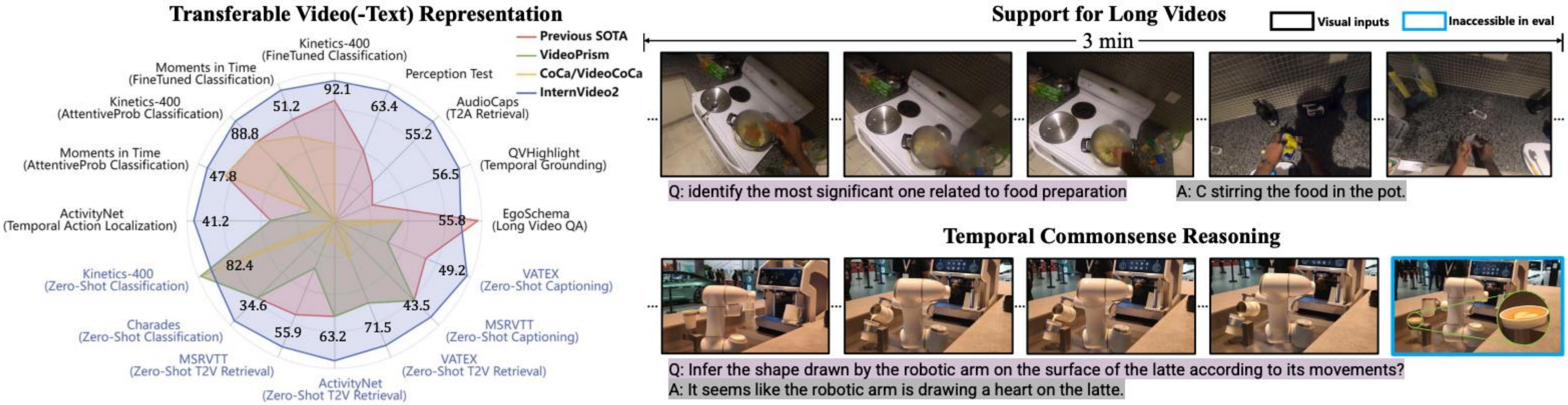


Figure 2: Framework of **InternVideo2**. It consists of three consecutive training phases: unmasked video token reconstruction, multimodal contrastive learning, and next token prediction. In stage 1, the video encoder is trained from scratch, while in stages 2 and 3, it is initialized from the version used in the previous stage.

Stage 1: Reconstructing Unmasked Video Tokens
Stage 2: Aligning Video to Audio-Speech-Text
Stage 3: Predicting Next Token with Video-Centric Inputs

INTERNVIDEO2: SCALING FOUNDATION MODELS FOR MULTIMODAL VIDEO UNDERSTANDING



Q: identify the most significant one related to food preparation

A: C stirring the food in the pot.

Q: Infer the shape drawn by the robotic arm on the surface of the latte according to its movements?

A: It seems like the robotic arm is drawing a heart on the latte.

Figure 1: **InternVideo2** yields strong transferable visual and visual-linguistic representations across a total of 70 video understanding tasks, ranging from action recognition, video-text understanding, to video-centric dialogue. It also exhibits capability of long-form video understanding and procedure-aware reasoning.