Vision–Language Models

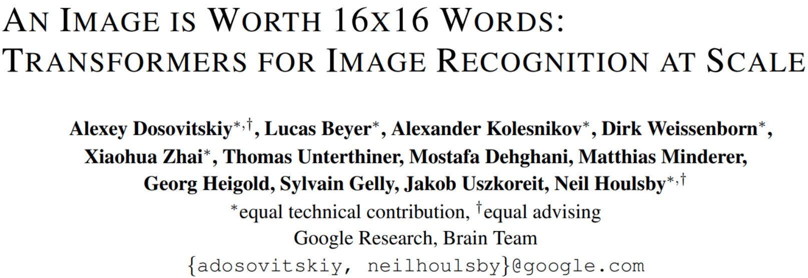
Jordan Boyd-Graber

University of Maryland

Visual Transformers

Slides from Mohit Iyyer, Vicente Ordonez, Fei-Fei Li, Justin Johnson, and Jacob Andreas

Using Patches for Transformers

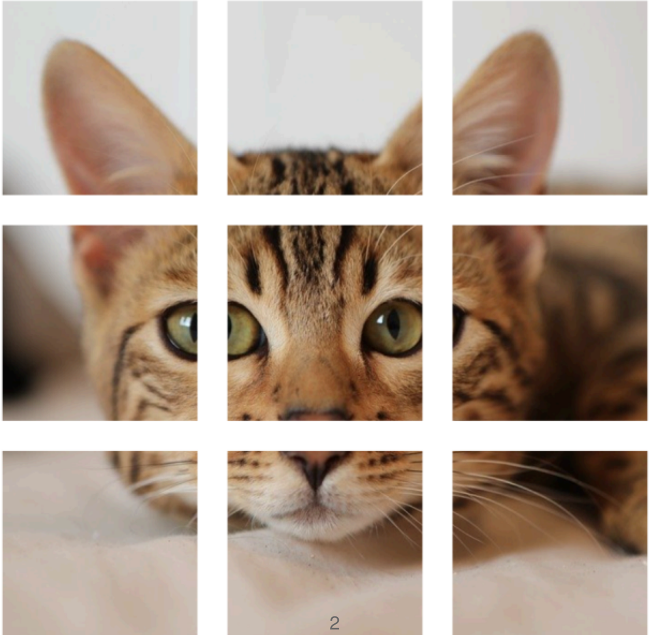


An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

Alexey Dosovitskiy[*,†], **Lucas Beyer**[*], **Alexander Kolesnikov**[*], **Dirk Weissenborn**[*],
Xiaohua Zhai[*], **Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer,**
Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby[*,†]
[*]equal technical contribution, [†]equal advising
Google Research, Brain Team
{adosovitskiy, neilhoulsby}@google.com

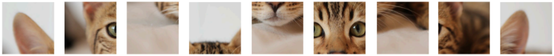
Using Patches for Transformers

Using Patches for Transformers

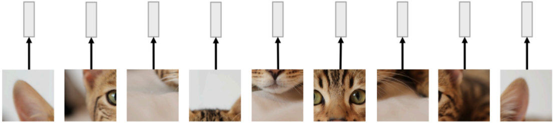


Using Patches for Transformers

N input patches, each
of shape 3x16x16



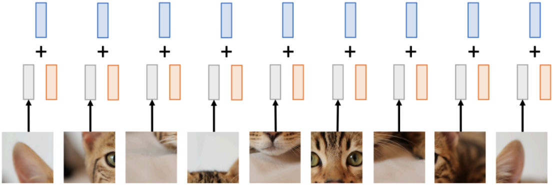
Using Patches for Transformers



Linear projection to D-dimensional vector

N input patches, each of shape 3x16x16

Using Patches for Transformers

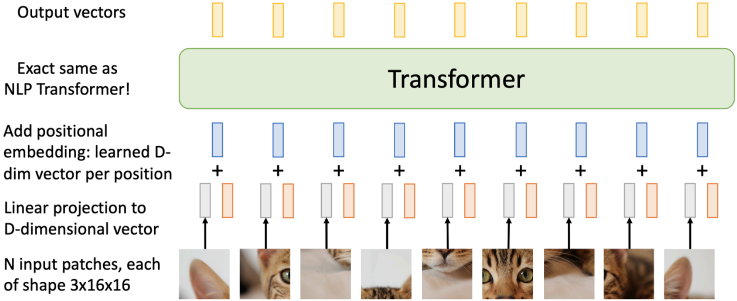


Add positional embedding: learned D-dim vector per position

Linear projection to D-dimensional vector

N input patches, each of shape 3x16x16

Using Patches for Transformers



Output vectors

Exact same as NLP Transformer!

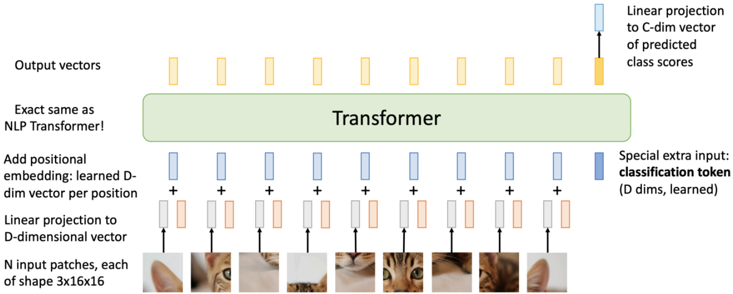
Transformer

Add positional embedding: learned D-dim vector per position

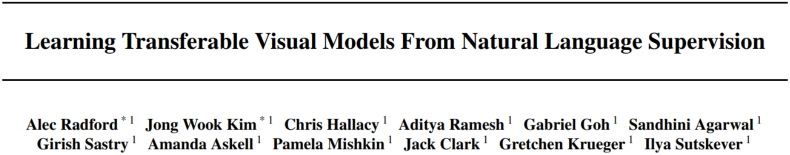
Linear projection to D-dimensional vector

N input patches, each of shape 3x16x16

Using Patches for Transformers


Output vectors
Exact same as NLP Transformer!
Transformer
Add positional embedding: learned D-dim vector per position
Linear projection to D-dimensional vector
N input patches, each of shape 3x16x16
Linear projection to C-dim vector of predicted class scores
Special extra input: **classification token** (D dims, learned)
+

CLIP

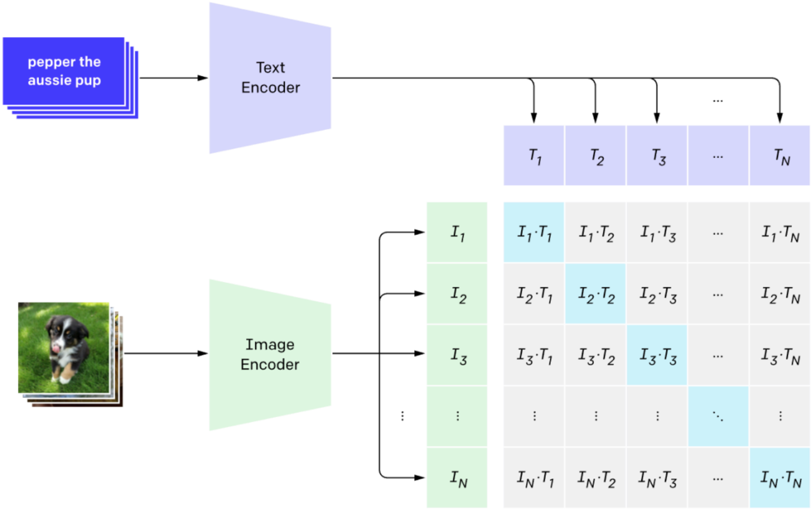


Learning Transferable Visual Models From Natural Language Supervision

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]

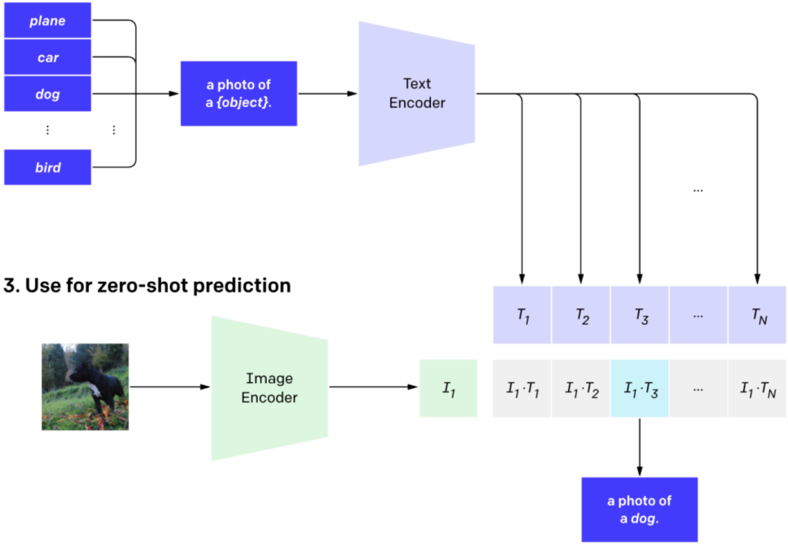
- OpenAI collect 400 million (image, text) pairs from the web
- Then, they train an image encoder and a text encoder with a simple contrastive loss: given a collection of images and text, predict which (image, text) pairs actually occurred in the dataset

Joint Training

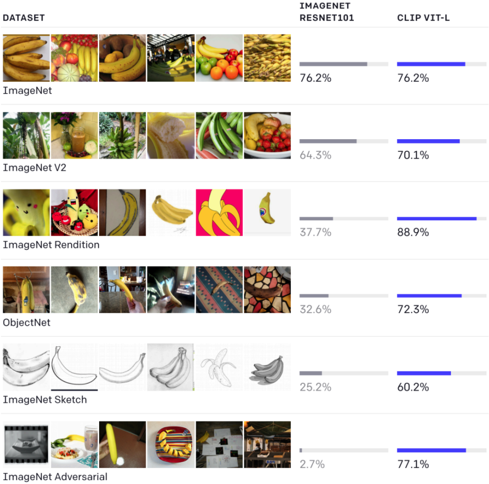


	T_1	T_2	T_3	...	T_N
I_1	$I_1 \cdot T_1$	$I_1 \cdot T_2$	$I_1 \cdot T_3$	...	$I_1 \cdot T_N$
I_2	$I_2 \cdot T_1$	$I_2 \cdot T_2$	$I_2 \cdot T_3$	...	$I_2 \cdot T_N$
I_3	$I_3 \cdot T_1$	$I_3 \cdot T_2$	$I_3 \cdot T_3$	...	$I_3 \cdot T_N$
⋮	⋮	⋮	⋮	⋱	⋮
I_N	$I_N \cdot T_1$	$I_N \cdot T_2$	$I_N \cdot T_3$	...	$I_N \cdot T_N$

Joint Training



Joint Training



DATASET	IMAGENET RESNET101	CLIP VIT-L
ImageNet	76.2%	76.2%
ImageNet V2	64.3%	70.1%
ImageNet Rendition	37.7%	88.9%
ObjectNet	32.6%	72.3%
ImageNet Sketch	25.2%	60.2%
ImageNet Adversarial	2.7%	77.1%

Generating text is one thing, but what about image generation?

- Could do autoregressive model pixel by pixel (people have tried)
- But better to learn higher-order structure