

CS388: Natural Language Processing

Lecture 16: Intro to Vision



TEXAS

The University of Texas at Austin

Administration

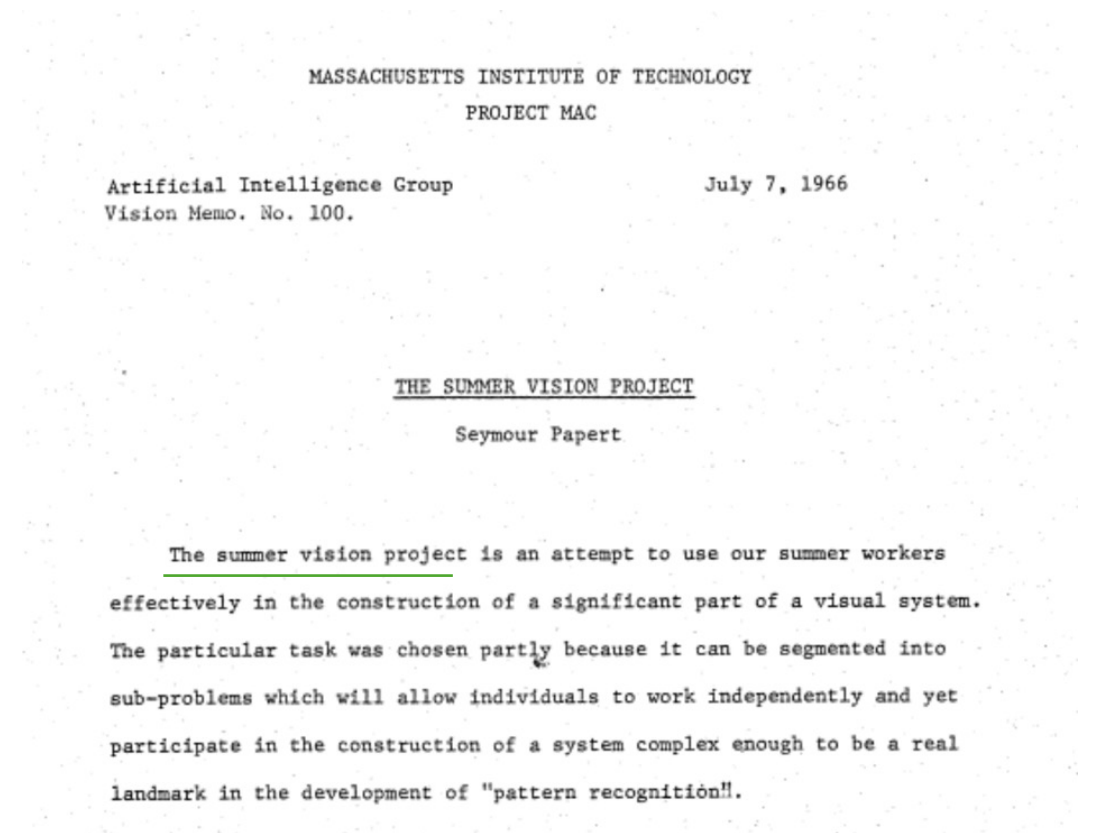
- HW due today
- Proposals due tomorrow
- Quiz 3: 10/27

Today

- What is computer vision?
- CNNs, ResNet
- ViT
- What is grounding?
- Early grounded LLMs (LXMERT, ViLBERT)
- CLIP and its variants

What is computer vision?

- **1966:** MIT Summer Vision Project
 - Goal: build a large part of a visual system in one summer
- 60 years later: still an open research area

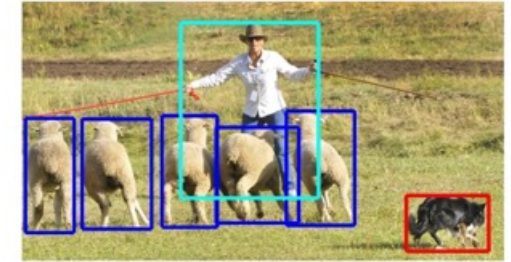


Goals of computer vision

- **Classification:** what is in the image?
- **Detection:** where is it?
- **Segmentation:** which pixels belong to it?
- **3D and motion:** shape, depth, video
- **Description:** captions, answers to questions



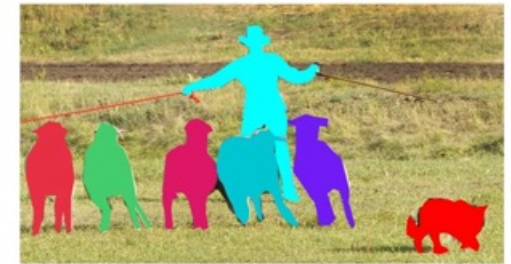
(a) Image classification



(b) Object localization

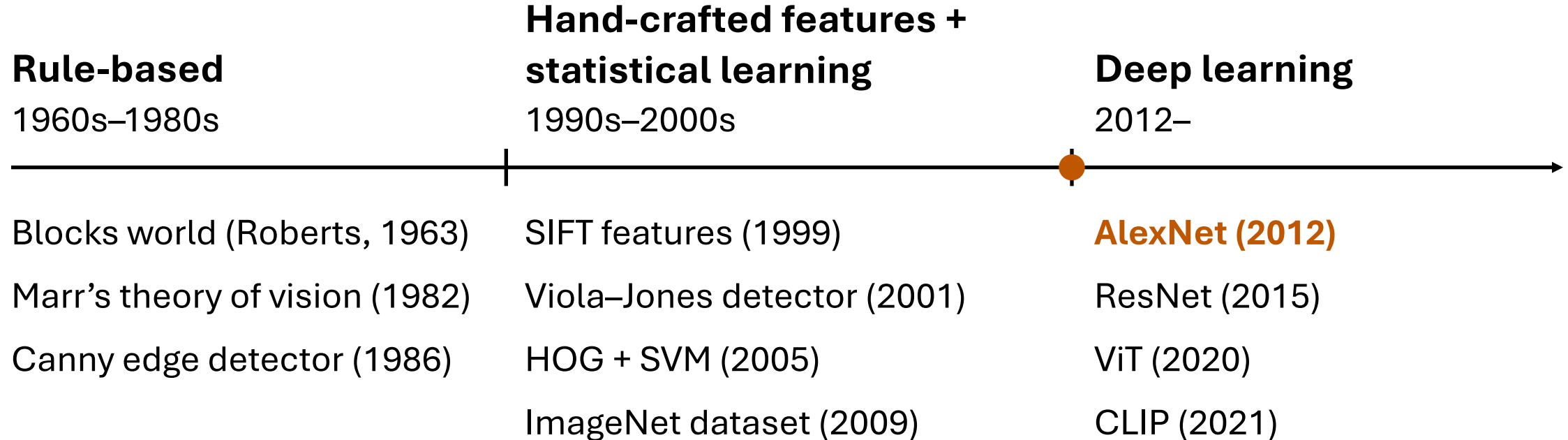


(c) Semantic segmentation



(d) This work

Computer vision timeline



ImageNet and AlexNet

ImageNet challenge (ILSVRC), winning top-5 error

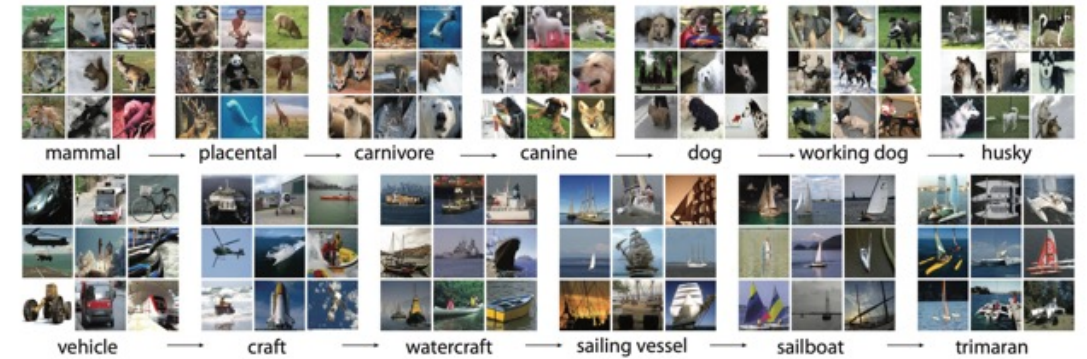
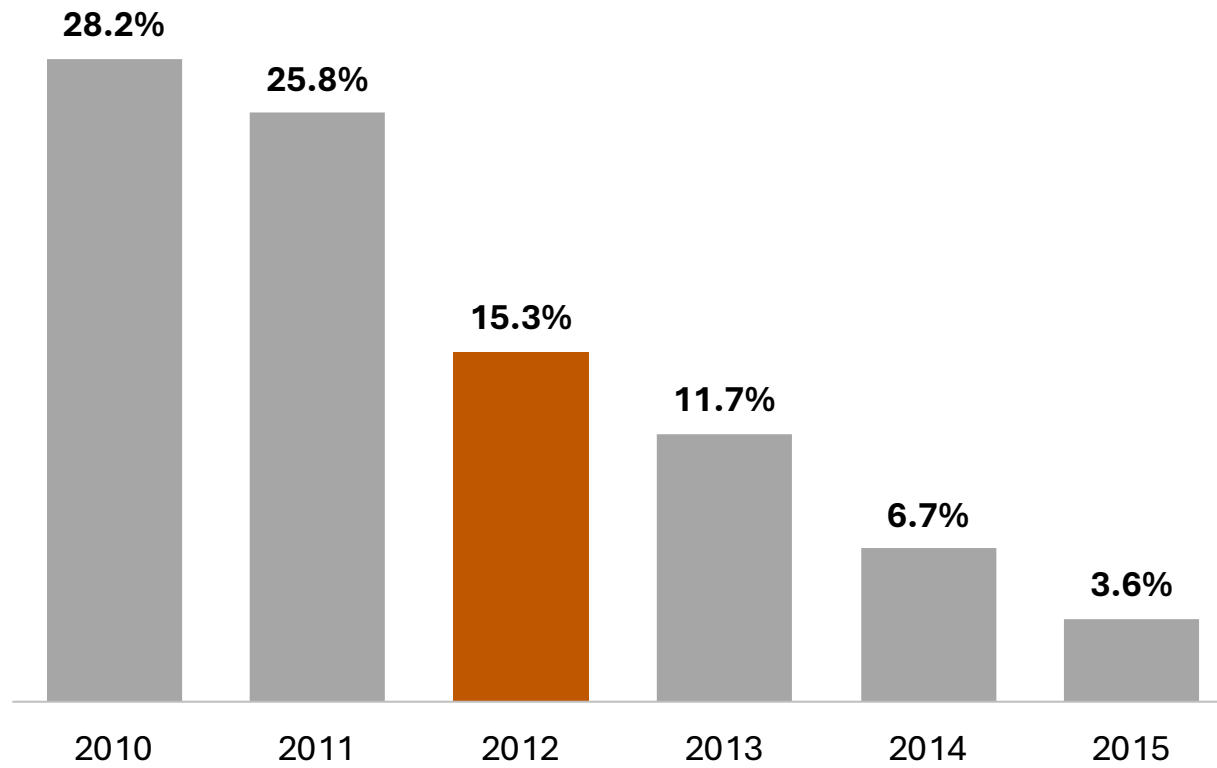


Figure 1: A snapshot of two root-to-leaf branches of ImageNet: the **top** row is from the mammal subtree; the **bottom** row is from the vehicle subtree. For each synset, 9 randomly sampled images are presented.

Computer vision today: open problems

- **Robustness:** accuracy drops under distribution shift
- **Long tail:** rare objects and settings
- **3D and physics:** depth, geometry, how objects interact
- **Video:** reasoning over time
- **Vision-language models:** counting, spatial relations, hallucinated objects

How images are represented

R			G			B		
255	248	31	120	115	90	40	38	160
240	236	27	118	110	84	35	33	171
201	190	22	96	92	70	30	28	180

One $H \times W$ grid per color channel

$$x \in \mathbb{R}^{H \times W \times C}$$

- $C = 3$ (RGB)
- Each entry: 0–255
- $224 \times 224 \times 3 = 150,528$ values

Why not flatten the image and use an MLP?

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

↓
 $\text{vec}(x)$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----

Pixel 6's neighbors 2 and 10 end up 8 positions apart

$$h = \sigma(W \text{vec}(x) + b)$$

- $224 \times 224 \times 3$ input, 1,000 hidden units: **150M weights**
- **Spatial structure lost**
- **No sharing:** a cat in the corner is learned separately from a cat in the center
- **Fixed input size**

Convolution: slide a small filter over the image

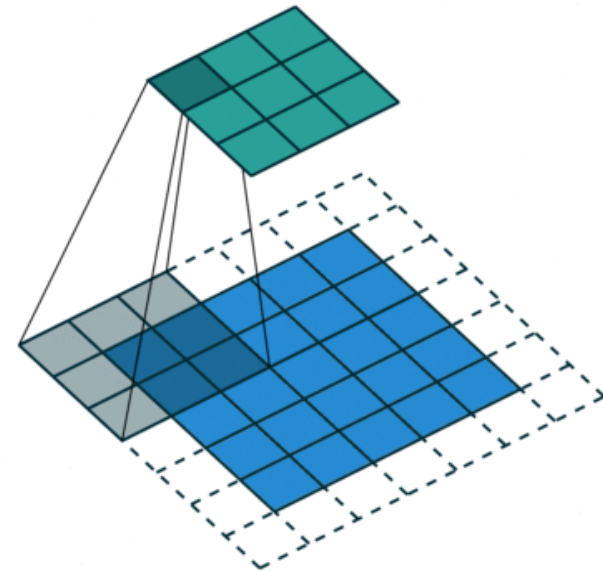
$$(x * K)[i, j] = \sum_m \sum_n x[i + m, j + n] K[m, n]$$

K

-1	0	1
-2	0	2
-1	0	1

Vertical edge filter (Sobel)

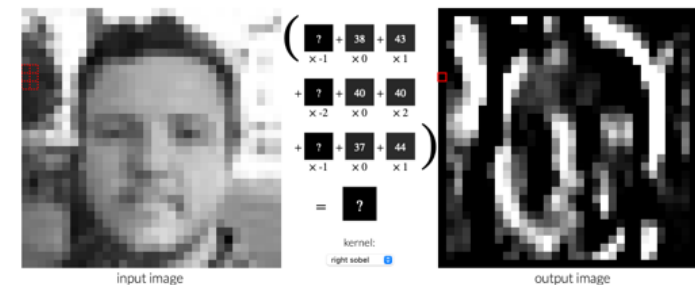
- Large output where brightness changes left to right
- Same weights at every position



right sobel

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Below, for each 3x3 block of pixels in the image on the left, we multiply each pixel by the corresponding entry of the kernel and then take the sum. That sum becomes a new pixel in the image on the right. Hover over a pixel in either image to see how its value is computed.



Sources: Dumoulin & Visin (2016), A Guide to Convolution Arithmetic for Deep Learning (top right); Powell (2015), Image Kernels Explained Visually, Setosa (bottom right)

How do we learn filters?

- Don't hand-design K : make filter weights **parameters**
- Learn them with backprop on the task loss
- Early layers learn **edge and color** detectors
- Deeper layers combine them into **textures, parts, objects**

Sources: Krizhevsky et al. (2012), ImageNet Classification with Deep CNNs (top); Zeiler & Fergus (2014), Visualizing and Understanding Convolutional Networks (bottom)

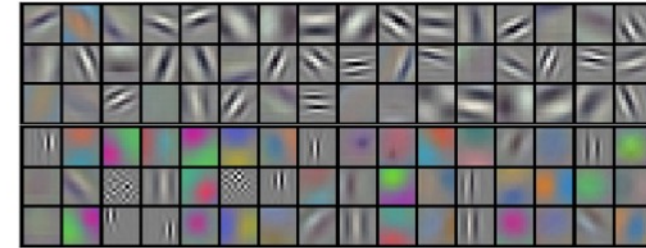
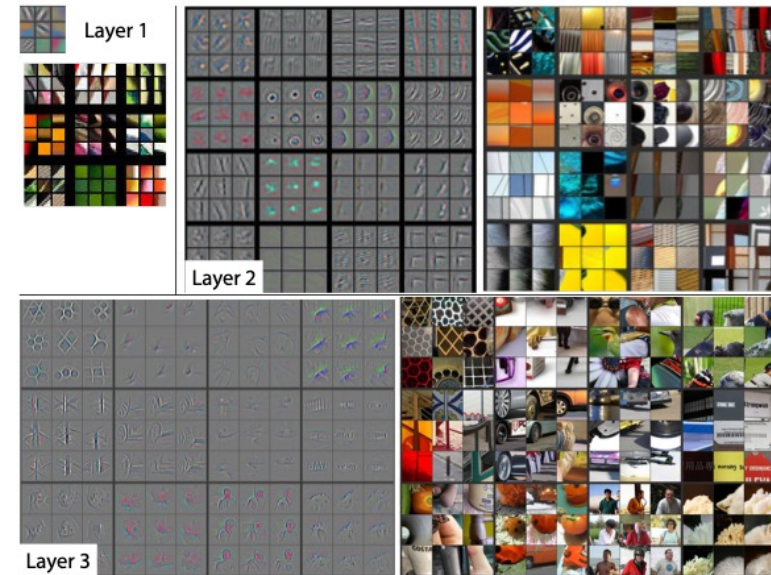


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

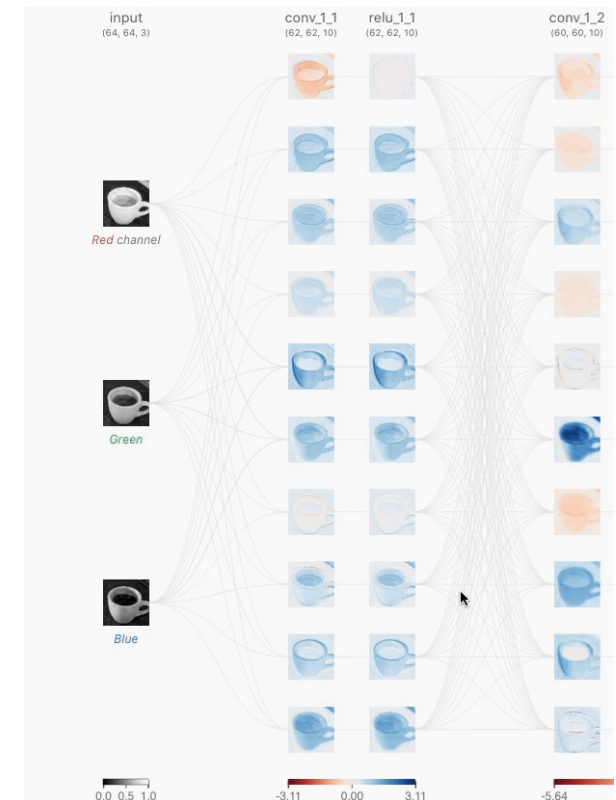


Convolutional neural networks



Add something showing the increasing number of channels decreasing w/h thing down to the MLP layer

- **Local connected** small patch
- **Weight sharing** channels = 1, 7
- **Translation equivariance:** shift the input, the output shifts
- Conv layers work on **any image size**



ResNet: deeper networks were harder to train

- Deeper helped: AlexNet 8 layers → VGG 19 layers
- But a **56-layer** plain net has **higher training error** than a 20-layer one
- Not overfitting: an **optimization** problem
- Extra layers could just learn the identity, but SGD doesn't find it

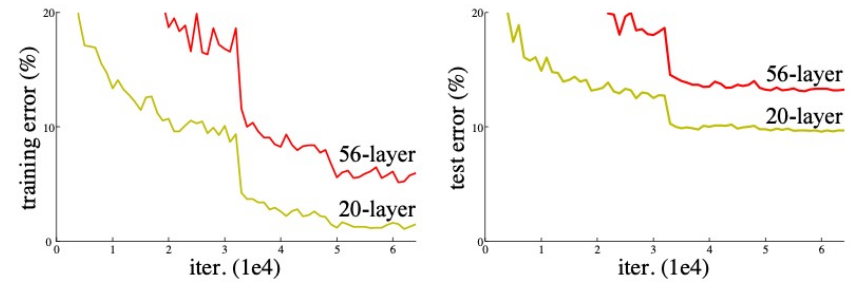


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer “plain” networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Residual connections

$$h_{l+1} = h_l + F(h_l; \theta_l)$$

- Each block learns only the **residual** F
- Identity is easy: set $F = 0$
- Gradients flow straight through the skip path
- Same idea as the **residual stream** in Transformers

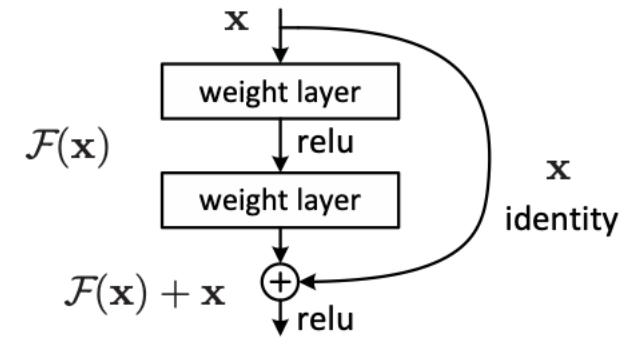
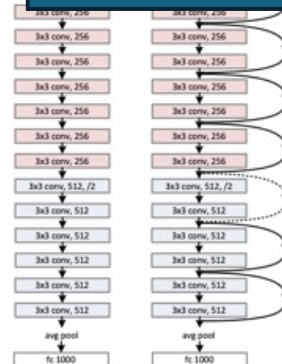
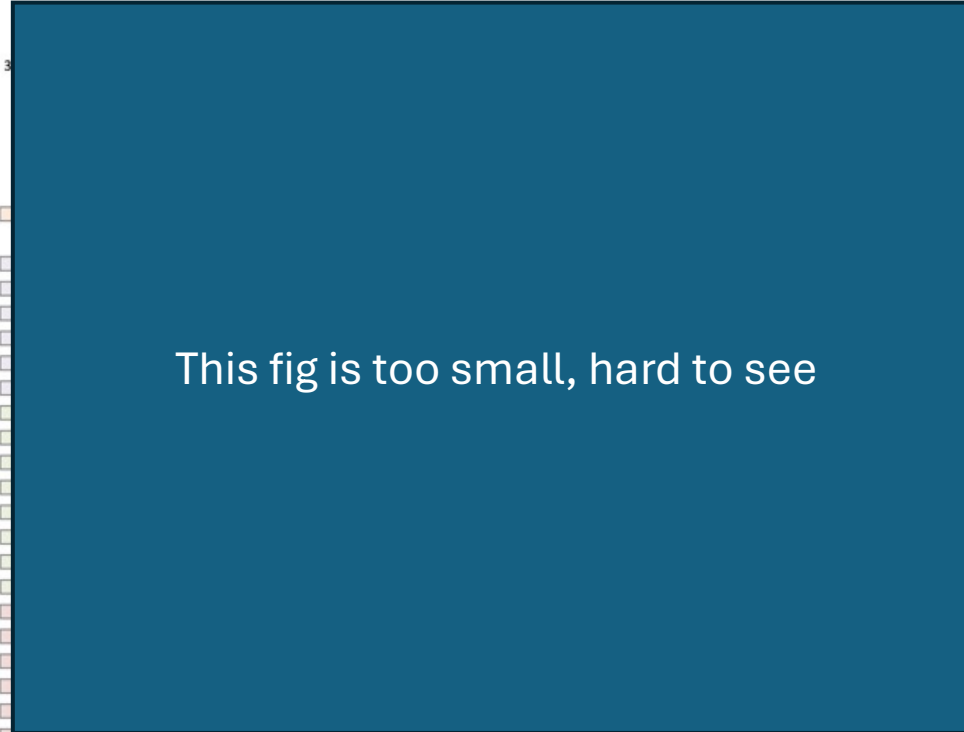
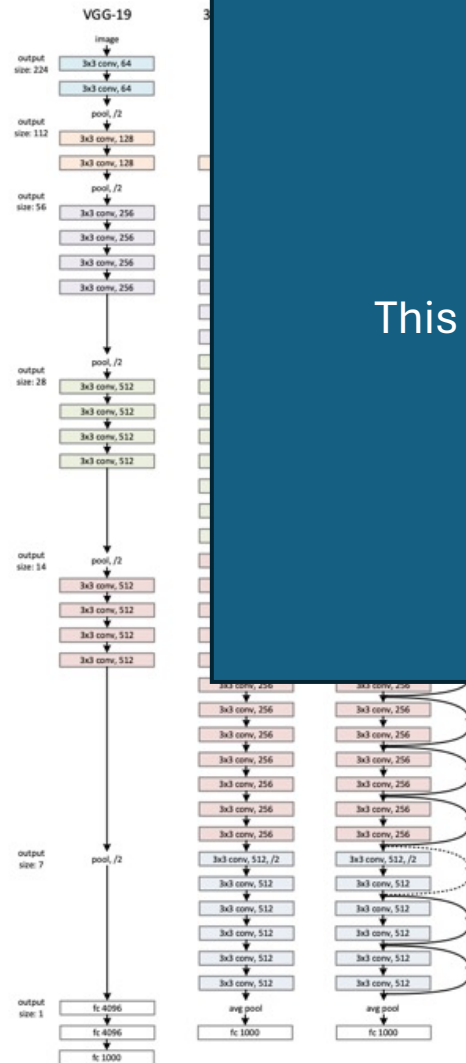


Figure 2. Residual learning: a building block.

ResNet architecture



of 3x3 conv residual blocks
average pool + one FC layer
2 layers

ResNet results

- Won ILSVRC 2015: **3.57%** top-5 error
- 152 layers: 8× deeper than VGG-19, fewer FLOPs
- Plain nets get worse with depth; ResNets get better
- Started a trend of “deeper” deep learning

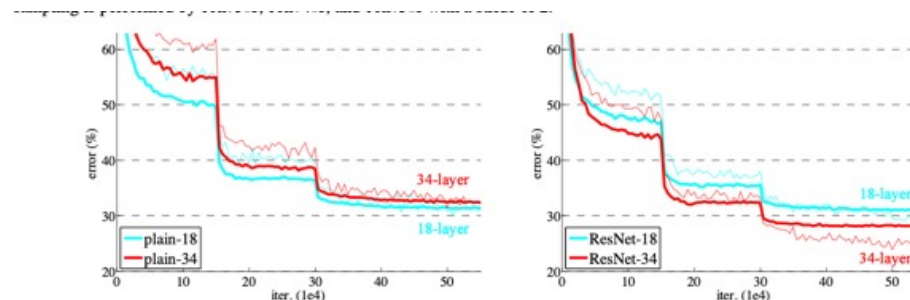


Figure 4. Training on ImageNet. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

method	top-1 err.	top-5 err.
VGG [41] (ILSVRC' 14)	-	8.43 [†]
GoogLeNet [44] (ILSVRC' 14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	19.38	4.49

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except [†] reported on the test set).

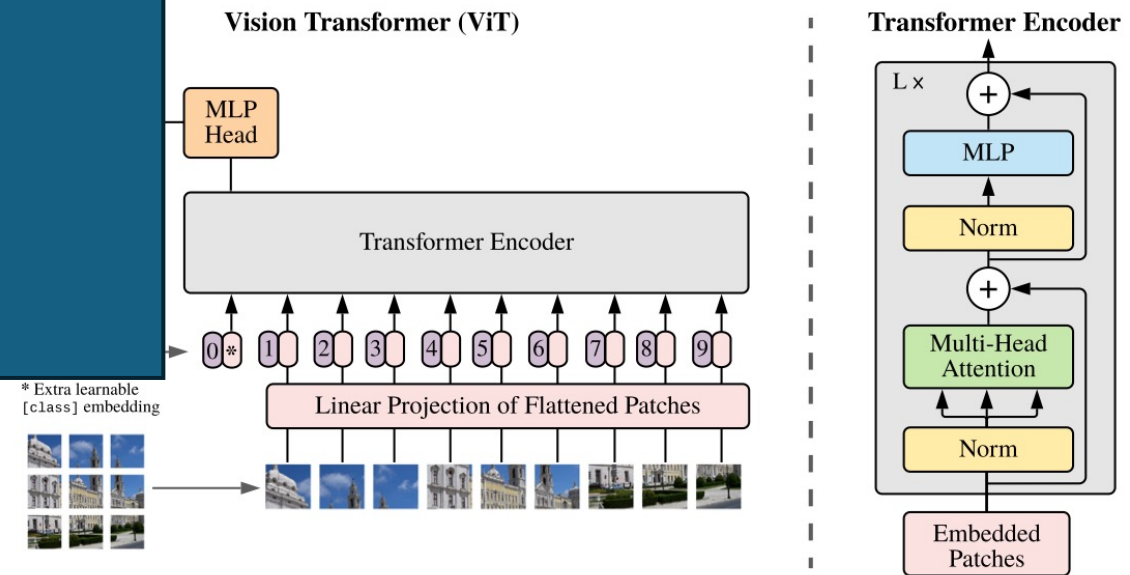
Vision Transformer (ViT)

- **Back to flattened**
- Split image into
- Flatten + project
- Add [CLS] token
- Standard Trans

This fig is too small, hard to see
Get an animation of tiling
Clarify that tiles get flattened

$$N = \frac{HW}{p^2}$$

$$z_0 = [x_{\text{cls}}; x_p^1 E; \dots; x_p^N E] + E_{\text{pos}}$$



Dosovitskiy et al. (2020), An Image is Worth 16x16 Words

ViT performance

- No convolutions: weak inductive bias
- Pretrained on ImageNet (1.3M images): **worse** than ResNets
- Pretrained on JFT-300M (303M images): **better**
- ViT-H/14: **88.55%** ImageNet top-1, with $\sim 4\times$ less pretraining compute than bigger/better version of ResNet

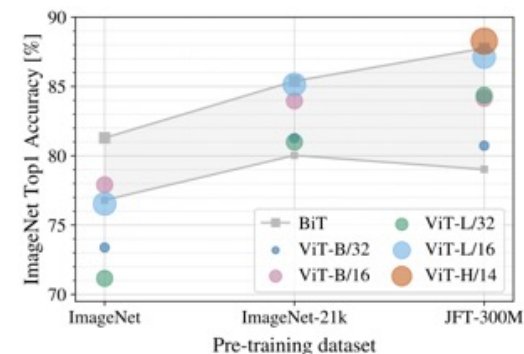


Figure 3: Transfer to ImageNet. While large ViT models perform worse than BiT ResNets (shaded area) when pre-trained on small datasets, they shine when pre-trained on larger datasets. Similarly, larger ViT variants overtake smaller ones as the dataset grows.

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	88.55 ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	90.72 ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	99.50 ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	—
CIFAR-100	94.55 ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	—
Oxford-IIIT Pets	97.56 ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	—
Oxford Flowers-102	99.68 ± 0.02	99.74 ± 0.00	99.61 ± 0.02	99.63 ± 0.03	—
VTAB (19 tasks)	77.63 ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	—
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Isn't this class about
language?

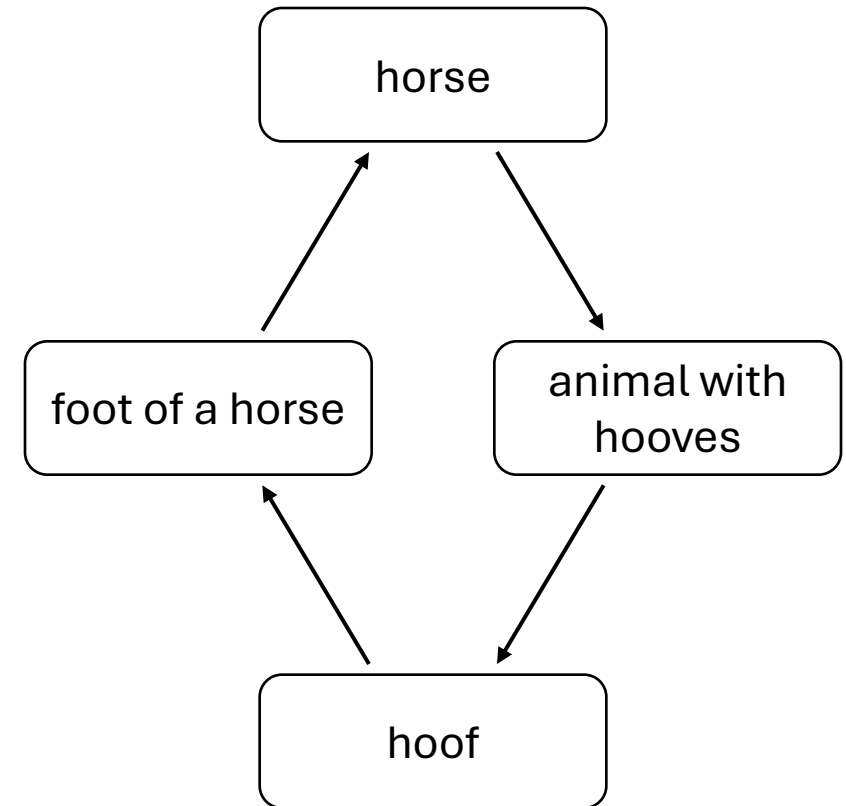
Yes! Grounding:

- We want to connect images and language
- Language happens in the real world (which can be represented visually)
- Lots of important tasks require both

The symbol grounding problem



- How do we know what language means?
 - Need to “ground” language symbols to something
- Harnad (1990): how do symbols get meaning if they are only defined by other symbols?
 - Imagine: learning Chinese from a Chinese–Chinese dictionary
- Answer: symbols must connect to **perception**
- Text-only LMs learn from symbols alone



Grounding in this class

- Connect text to image
- **representations**
- Phrases $\leftrightarrow$ image
- Sentences $\leftrightarrow$ image

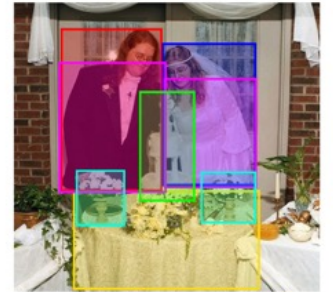
This fig is too small, hard to see



... is wearing **glasses** and an orange hat.
... wearing a beer can crocheted hat.
... **glasses** is wearing a Blitz hat.
... starring at something.
... hat and **glasses**.



During a gay pride parade in an Asian city, **some people** hold up **rainbow flags** to show their support.
A group of youths march down a **street** waving **flags** showing a color spectrum.
Oriental people with **rainbow flags** walking down a city street.
A group of people walk down a **street** waving **rainbow flags**.
People are outside waving **flags**.



A couple in **their wedding attire** stand behind **a table** with a **wedding cake** and flowers.
A bride and groom are standing in front of **their wedding cake** at their reception.
A bride and groom smile as they view **their wedding cake** at a reception.
A couple stands behind **their wedding cake**.
Man and **woman** cutting **wedding cake**.

Plummer et al. (2015), Flickr30k Entities

Why is grounding useful?

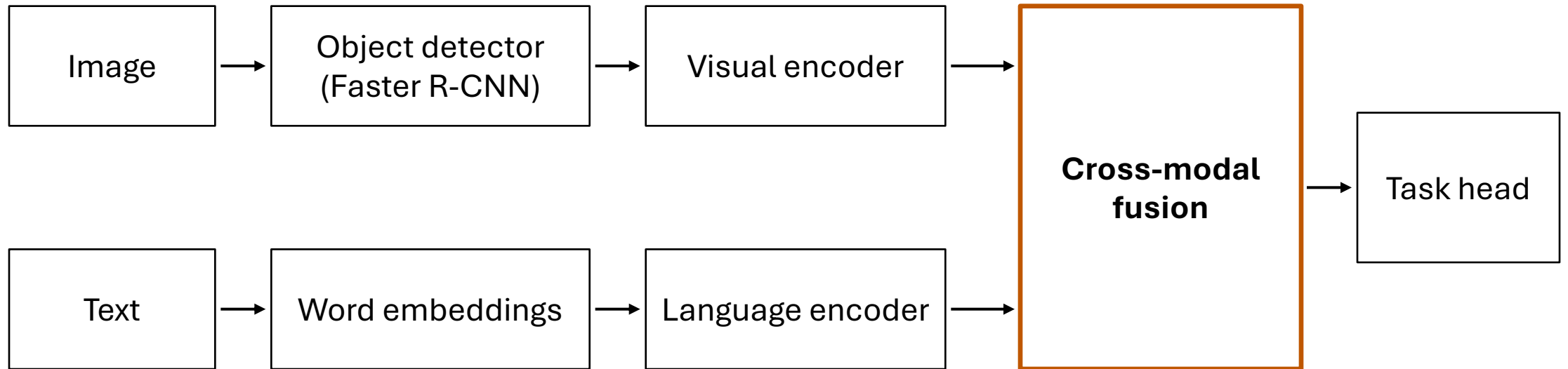
Many tasks require it

- Visual question answering
- Image captioning
- Text–image retrieval
- Referring expressions
- Visual reasoning
- Following instructions (robotics)

A signal for learning

- We don't learn language in a vacuum
- Words refer to things in the world
- Maybe grounding helps models learn meaning
 - Bisk et al. (2020), Experience Grounds Language

Early grounded models: dual-stream



- One encoder per modality, fused late with cross-attention
- Image = a set of detected region features

LXMERT (Tan & Bansal, 2019)

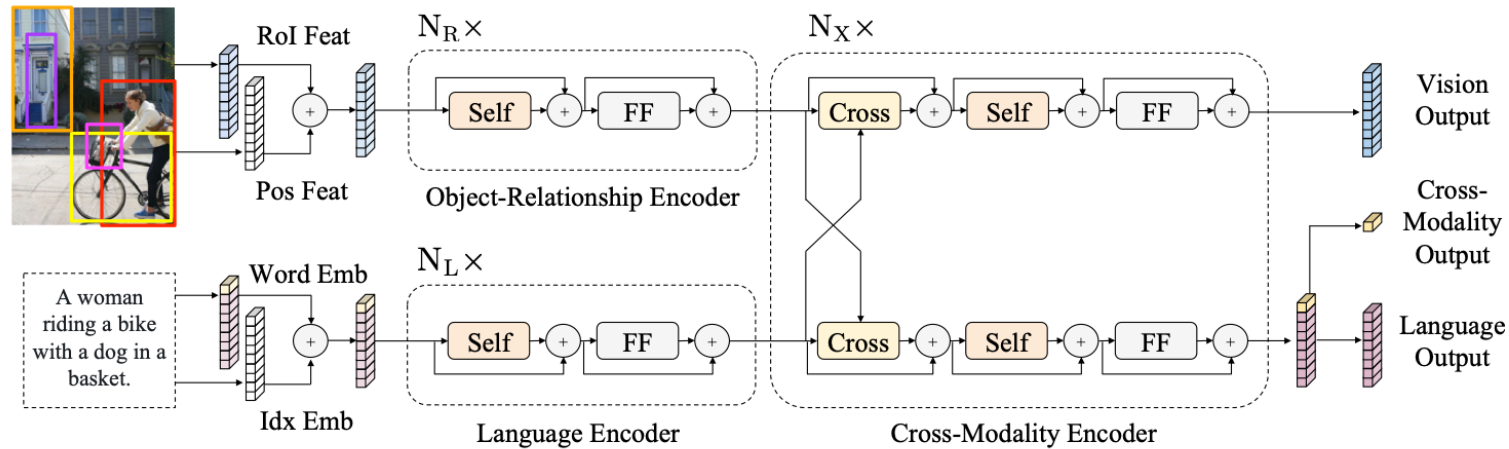


Figure 1: The LXMERT model for learning vision-and-language cross-modality representations. ‘Self’ and ‘Cross’ are abbreviations for self-attention sub-layers and cross-attention sub-layers, respectively. ‘FF’ denotes a feed-forward sub-layer.

- Three encoders: **object-relationship, language, cross-modality**
- Image: 36 detected objects (Faster R-CNN), region features + box positions
- Pretrained on 9.18M image–sentence pairs from 180K images

LXMERT pretraining

- **Masked word**
- **Masked object**
predict detected
- **Matching:** do
this image?
- **Image QA:** an
- NLVR2 accur

animate



The left image contains twice the number of dogs as the right image, and at least two dogs in total are standing.



One image shows exactly two brown acorns in back-to-back caps on green foliage.

Suhr et al, (2019)

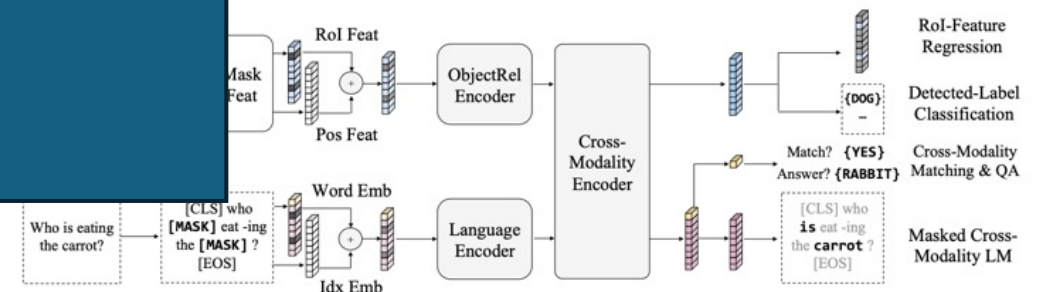


Figure 2: Pre-training in LXMERT. The object RoI features and word tokens are masked. Our five pre-training tasks learn the feature representations based on these masked inputs. Special tokens are in brackets and classification labels are in braces.

ViLBERT (Lu et al., 2019)

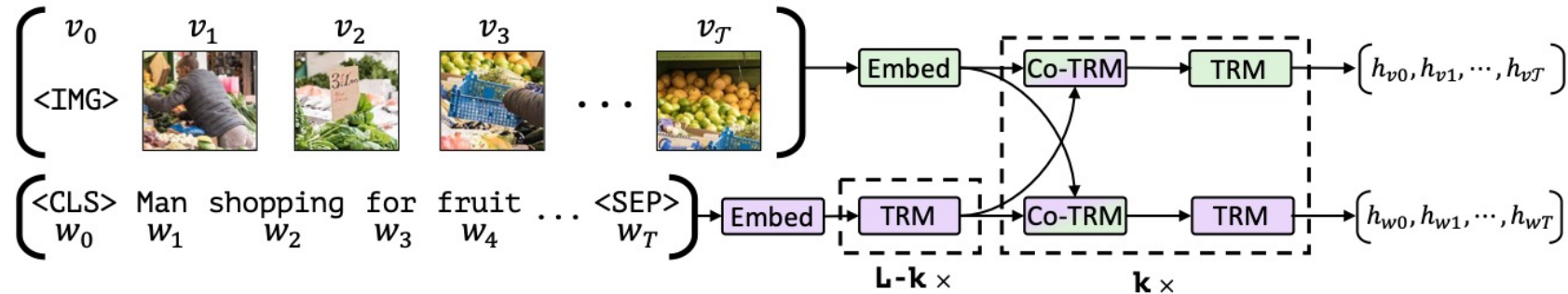
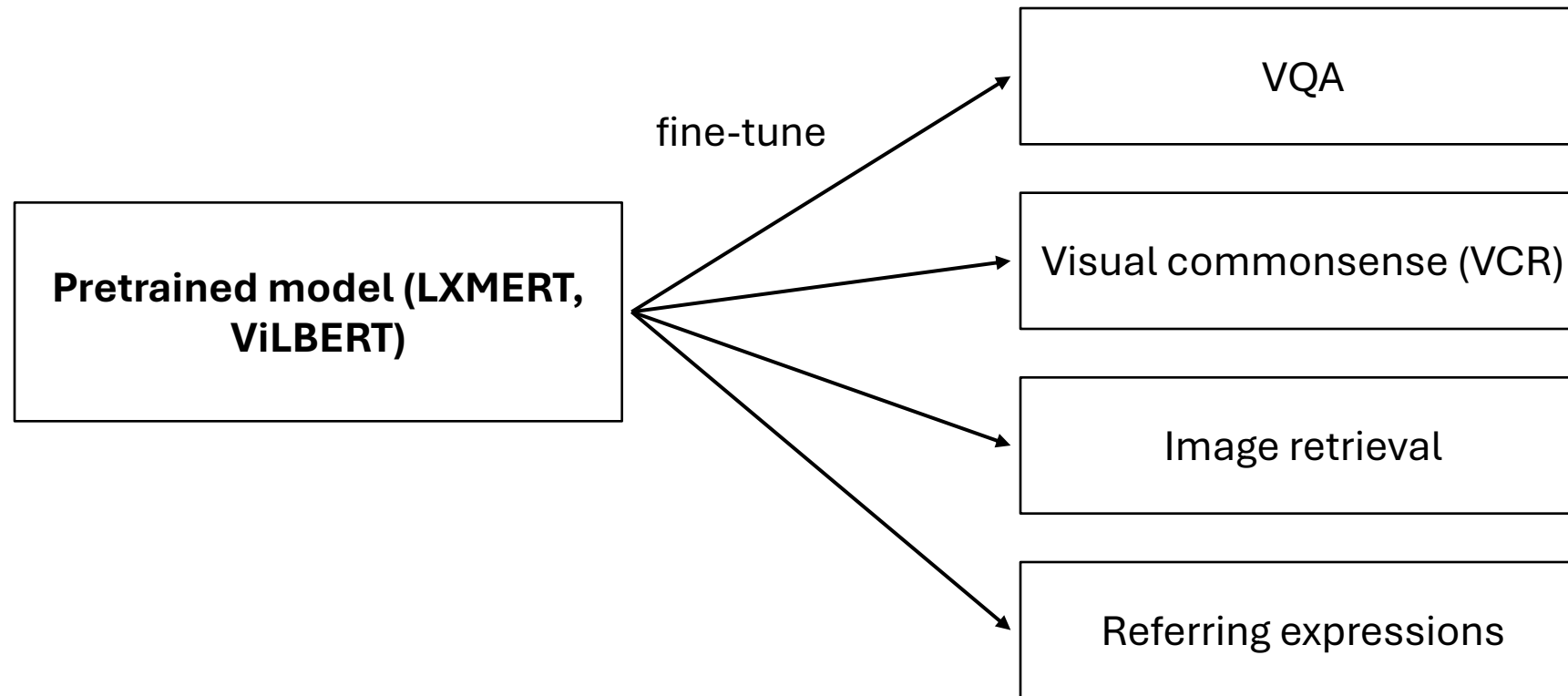


Figure 1: Our ViLBERT model consists of two parallel streams for visual (green) and linguistic (purple) processing that interact through novel co-attentional transformer layers. This structure allows for variable depths for each modality and enables sparse interaction through co-attention. Dashed boxes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

- Two BERT streams linked by **co-attention**
- Pretrained on Conceptual Captions (3.1M image–caption pairs)
- Transferred to VQA, VCR, referring expressions, retrieval

Both follow the BERT paradigm



How many slices of pizza are there?
Is this a vegetarian pizza?

- Every task needs **labeled data** and its own fine-tuned copy

Problem: vision and language loosely aligned

- **Frank et al. (2021):** removing the image hurts word prediction; removing text barely hurts object prediction
 - Vision is used for language, not the reverse
- **Cao et al. (2020):** attention goes mostly to text
- **Winoground (Thrush et al., 2022):** swap words in a caption → models near chance

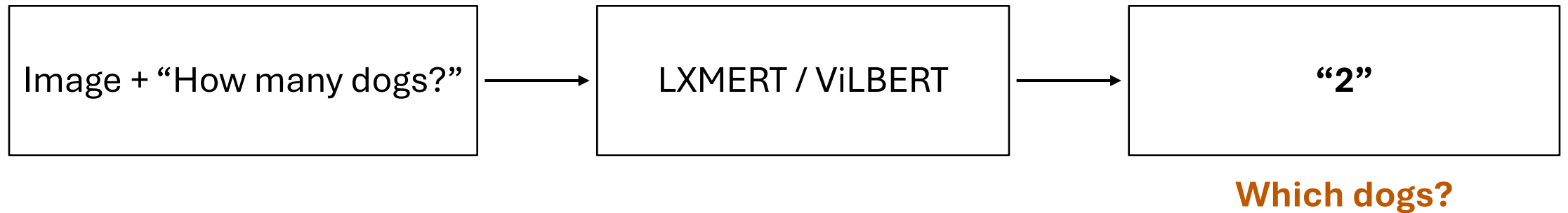


(a) some plants surrounding a lightbulb



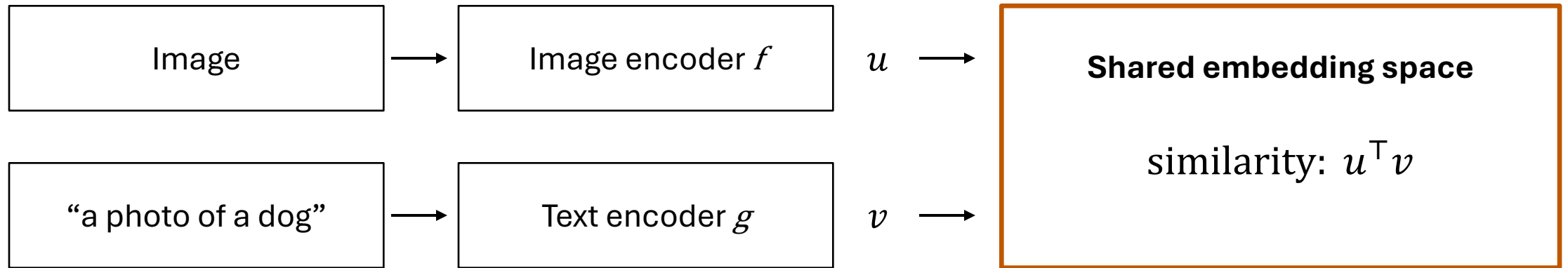
(b) a lightbulb surrounding some plants

Problem: “grounding” is vague



- Output is a label, with no pointer to the image regions behind it
- Regions come from a frozen, separately trained detector
- Attention weights are not a reliable explanation
- **Hypothesis-only baseline** (like NLI)

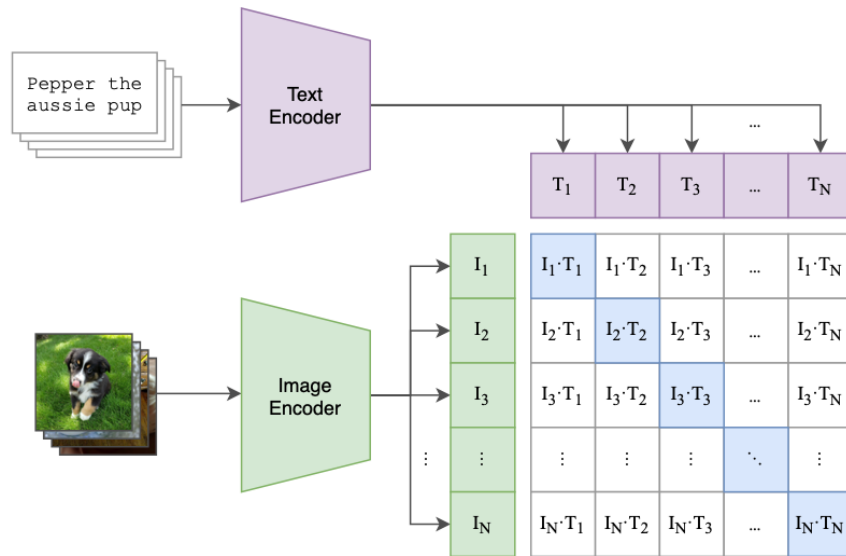
CLIP: align images and text directly



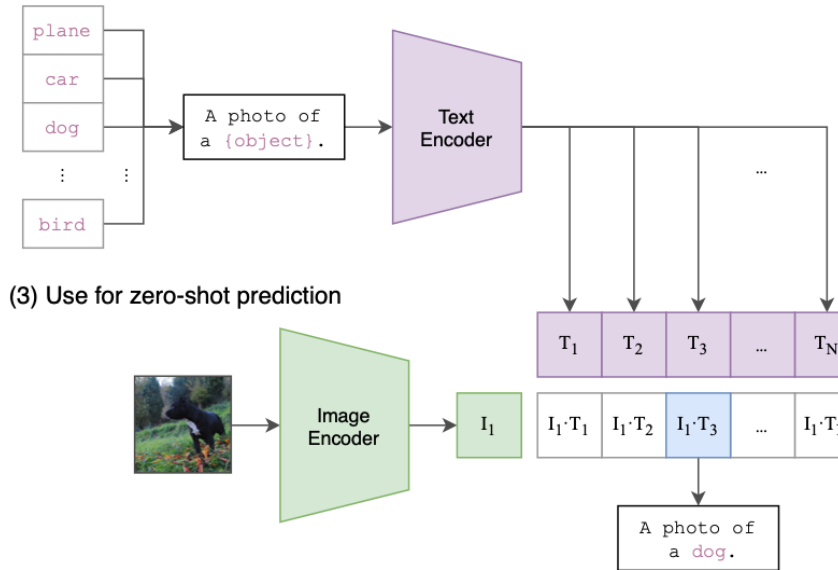
- Matching image–caption pairs close, mismatched pairs far apart
- **Contrastive learning** on 400M web image–text pairs
- Aligns whole images with whole captions, not regions

CLIP overview (Radford et al., 2021)

(1) Contrastive pre-training



(2) Create dataset classifier from label text



- 400M image-text pairs collected from the web
- Image encoder: ResNet or ViT; text encoder: Transformer
- Batch of 32,768 pairs

CLIP training objective

$$u_i = \frac{f(x_i)}{\|f(x_i)\|}, \quad v_j = \frac{g(t_j)}{\|g(t_j)\|}, \quad s_{ij} = \frac{u_i^\top v_j}{\tau}$$

$$\mathcal{L} = -\frac{1}{2B} \sum_{i=1}^B \left[\log \frac{\exp(s_{ii})}{\sum_j \exp(s_{ij})} + \log \frac{\exp(s_{ii})}{\sum_j \exp(s_{ji})} \right]$$

- $B \times B$ similarity matrix: B positives on the diagonal, the rest negatives
- Cross-entropy both ways: image $\rightarrow$ text, text $\rightarrow$ image
- Temperature τ is learned

```
# 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
```

Zero-shot classification with CLIP

$$t_y = \text{“a photo of a } \{y\} \text{.”}, \quad v_y = g(t_y)$$

$$p(y \mid x) = \frac{\exp(u^\top v_y / \tau)}{\sum_{y' \in \mathcal{Y}} \exp(u^\top v_{y'} / \tau)}$$

- Turn each class name into a caption
- Predict the class whose caption is closest to the image
- **No labeled training data** for the new task
- Prompt engineering + ensembling: about +5 points on ImageNet

What CLIP can be used for

Out of the box

- Zero-shot classification
- Text ↔ image retrieval
- Filtering web data by CLIP score
- Grounding, once extended to regions (OWL-ViT)

As a building block

- Vision encoder for VLMs (LLaVA)
- Text encoder for image generation (Stable Diffusion)
- Image embeddings for DALL·E 2

Can't generate text by itself, but a strong **pretraining objective**

CLIP results

- **76.2%** zero-shot on ImageNet: matches a ResNet-50 trained on 1.28M labeled images
- Much more **robust to distribution shift** (sketches, renditions)

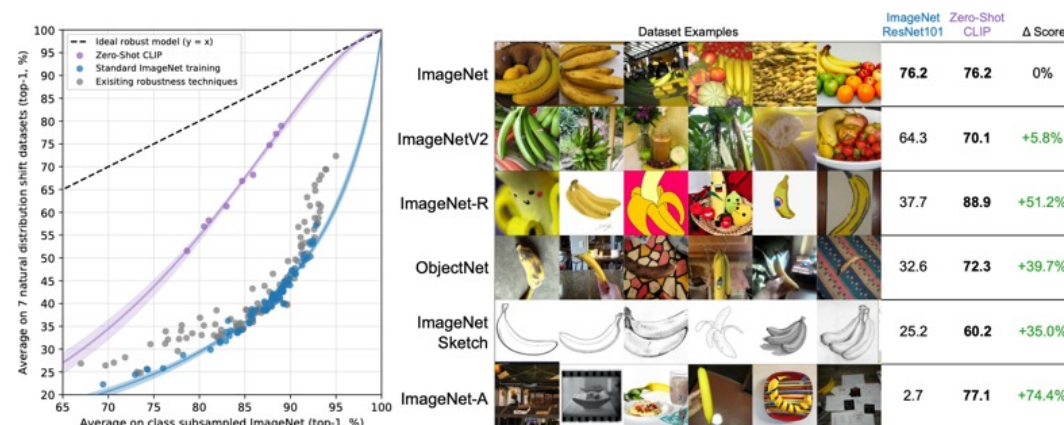
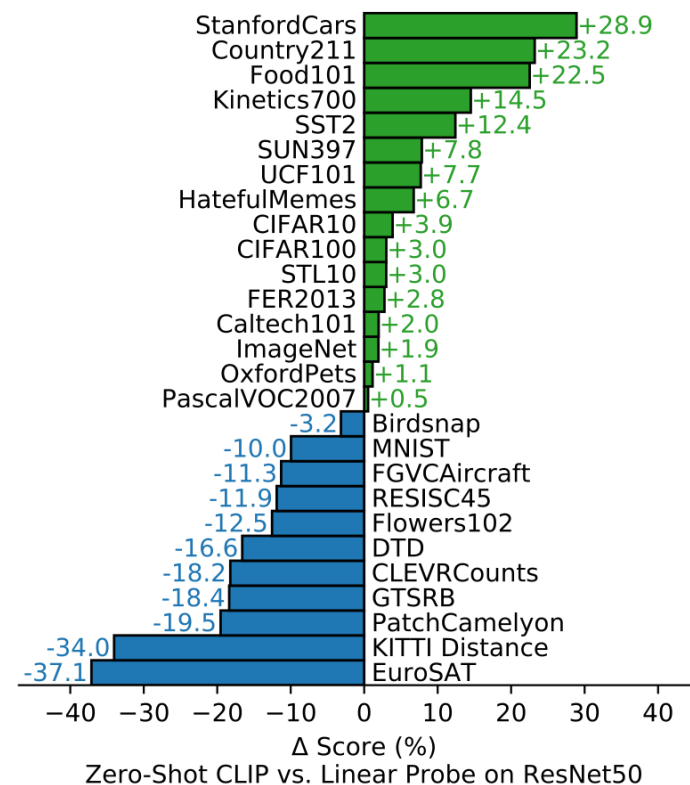


Figure 13. Zero-shot CLIP is much more robust to distribution shift than standard ImageNet models. (Left) An ideal robust model

CLIP Variants

- **SigLIP:** swaps CLIP's softmax over the batch for a per-pair sigmoid loss, so it trains well with smaller batches
- **EVA-CLIP:** keeps the CLIP loss but starts from a pretrained EVA image encoder, reaching higher accuracy with fewer training samples

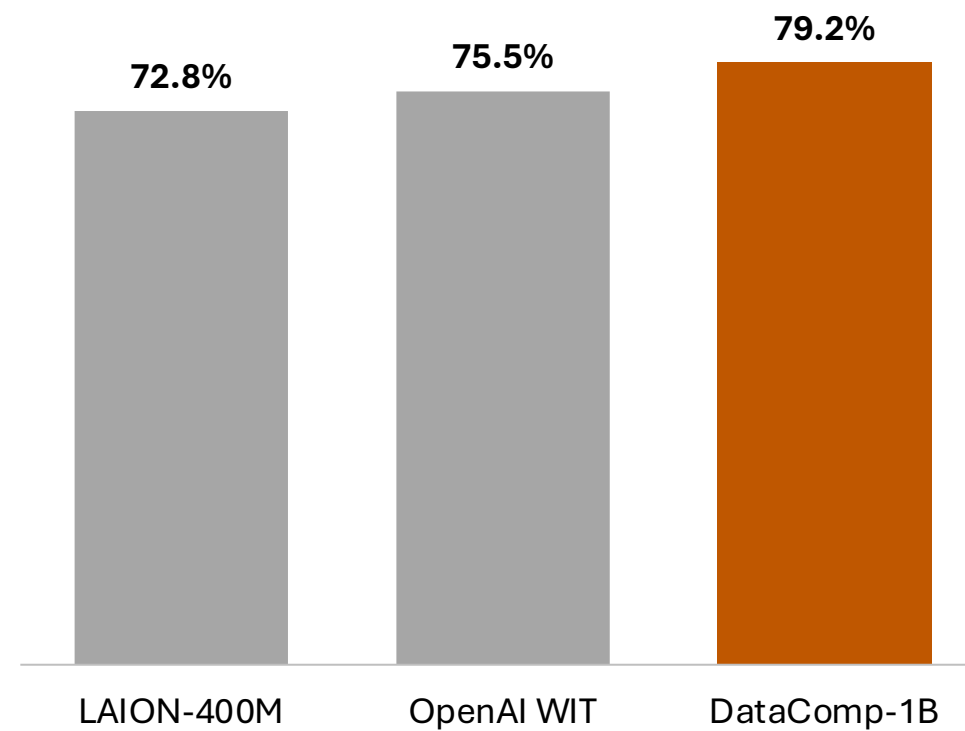
Note on data quality

- Data quality matters here, as it did in LLM pretraining
- Web image–text data is noisy and full of content you don’t want to train on
- **LAION-400M**: explicit sexual content, racist and ethnic slurs, malignant stereotypes (Birhane et al., 2021)
- **LAION-5B**: child sexual abuse material found (2023); pulled, re-released as Re-LAION-5B (2024)
- Filtering at this scale is hard

Better data, better CLIP

- DataComp (Gadre et al., 2023): fix the model and compute, change only the data
- Filter a 12.8B-pair web pool by CLIP score + image clustering
- DataComp-1B beats OpenAI's data by **3.7 points**

Zero-shot ImageNet, ViT-L/14, same compute



Takeaways

- **CNNs:** local filters with shared weights, learned end to end
- **ResNets:** residual connections make very deep nets trainable
- **ViT:** images as sequences of patch tokens; wins with enough data
- **Grounding:** connecting language to what it refers to
- **LXMERT, ViLBERT:** BERT-style, fine-tuned per task, loosely aligned
- **CLIP:** contrastive image–text training enables zero-shot transfer
- **Data quality** matters as much as in LLM pretraining