Robot Perception and Control

Robot Perception in 2D

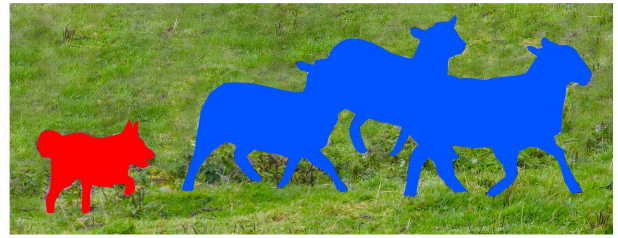
Last updated: Jul / 25 /2024

Kashu Yamazaki

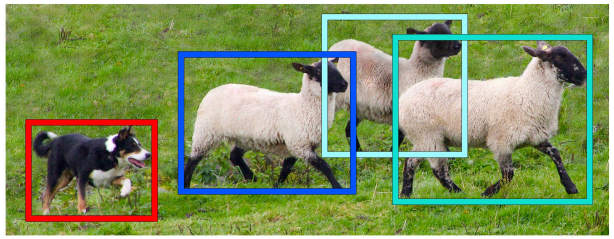
kyamazak@andrew.cmu.edu

Image Perception Tasks

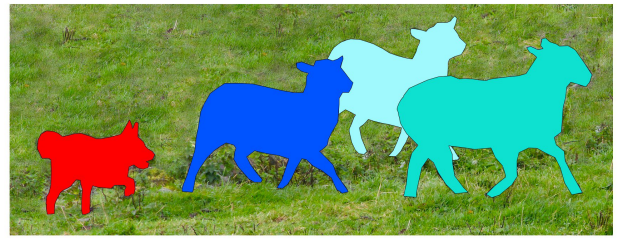
Image Recognition



Semantic Segmentation



Object Detection



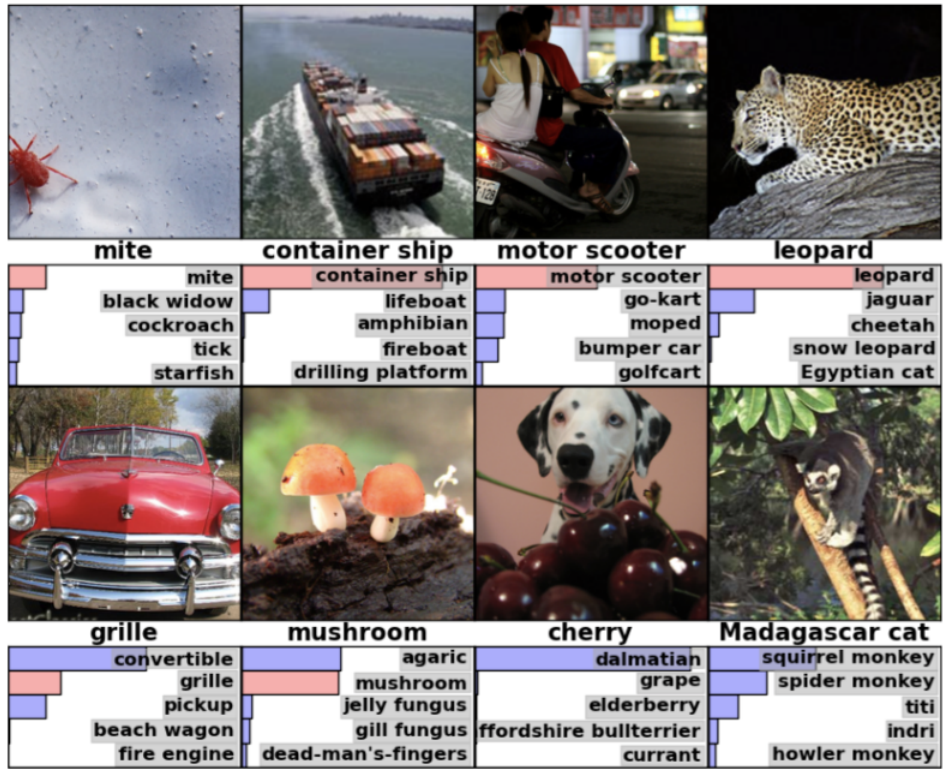
Instance Segmentation

Image Classification

ImageNet

ImageNet is a large-scale dataset for image classification, known for its use in Large Scale Visual Recognition Challenge (ILSVRC).

- (2009): ImageNet was released by Dr. *Li Fei-Fei* team.
- (2012): **AlexNet** made a breakthrough achieving 16% top-5 error rate - kickstarting the boom in deep learning.
- (2017): 95% accuracy reached and ILSVRC concluded.

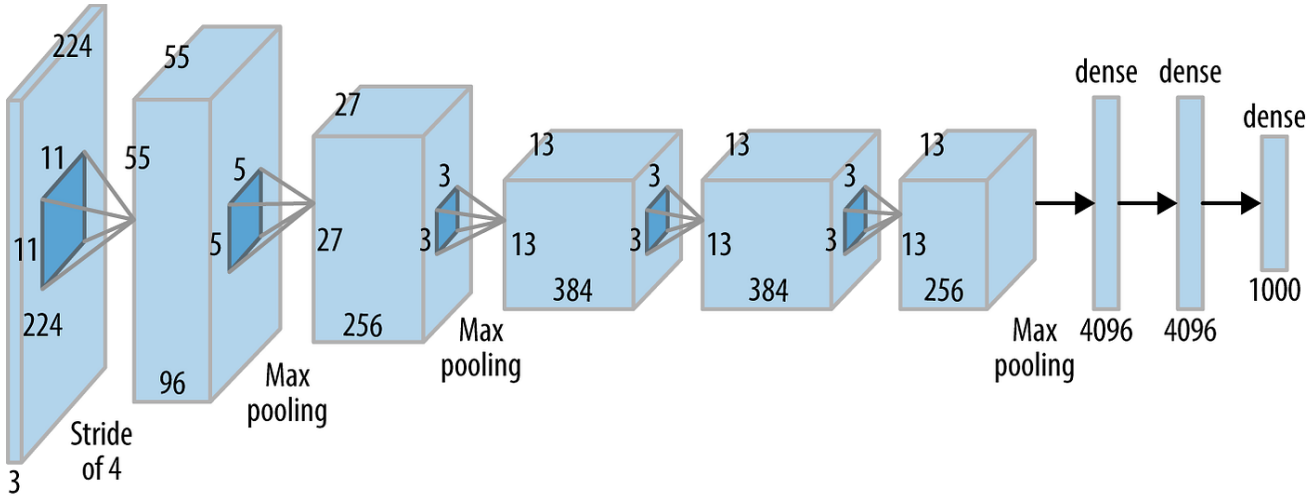


🕮 After ILSVRC Challenge

ImageNet continues to be a valuable resource for researchers for pre-training image encoder (backbone) models.

AlexNet

AlexNet is a **convolutional neural network** (CNN) architecture, designed by Alex Krizhevsky that contains eight layers: the first five are convolutional layers, some of them followed by max-pooling layers, and the last three are fully connected layers.



Influence

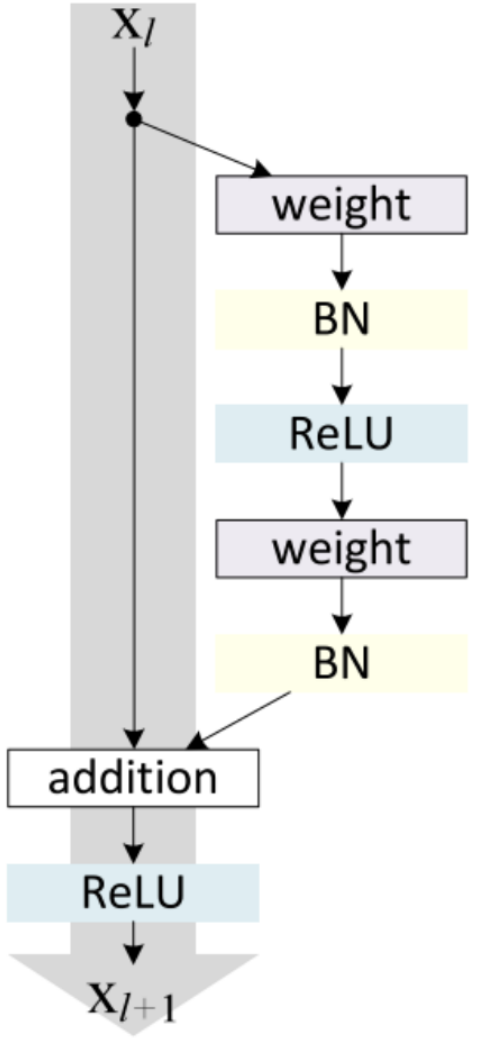
AlexNet is considered one of the most influential papers published in computer vision, having spurred many more papers published employing CNNs and GPUs to accelerate deep learning.

ResNet arxiv

ResNet layers learn **residual functions** with respect to the layer inputs.
A residual function $F_\theta(x) = H_\phi(x) - x$ represents difference between the underlying function $y = H_\phi(x)$ and the input x. A general form of ResNet layer is written as:

$$y = F_\theta(x) + x$$

The operation of $+x$ is called **residual connection** that performs an identity mapping to connect the input of the subnetwork F_θ with its output.



Influence

Learning the residual function can make the training of very deep networks easier. It helps in addressing the **vanishing gradient problem**.

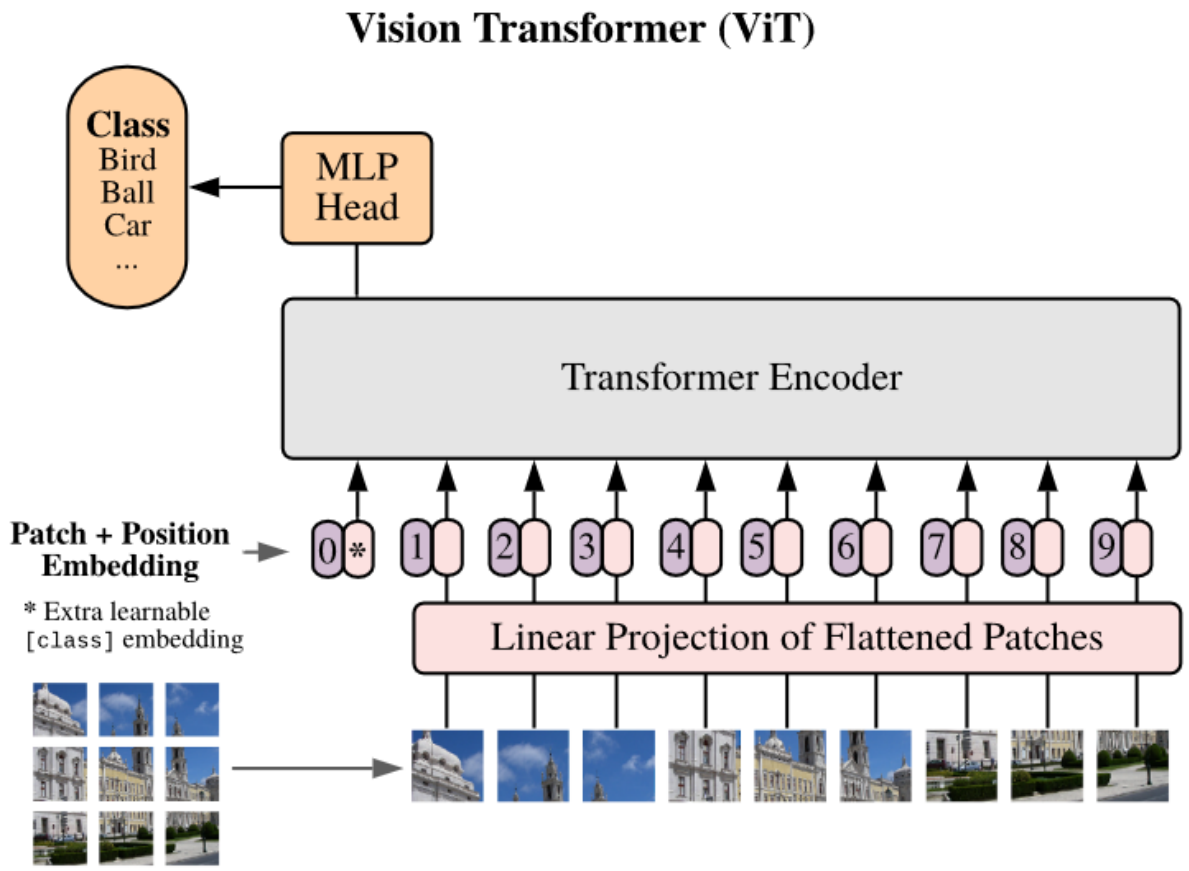
ViT arxiv

Applied pure transformer directly to sequences of *image patches*.

- An image patch is a 16×16 pixel crop of the image that will be treated as an image token.
- The patchify stem is implemented by a stride-p, $p \times p$ convolution ($p = 16$ by default) applied to the input image [1].

Influence

Performance scales with dataset size and becomes a new de facto for image backbone.



Representing Image as Patches

For an input image $x \in \mathbb{R}^{H \times W \times C}$ and patch size p, the $N = \frac{HW}{p^2}$ image patches $x_p \in \mathbb{R}^{N \times (p^2 C)}$ will be created.

- $p = 16$ for original ViT; thus 256×256 image is represented as 16×16 *image tokens*.

Sample implementation:

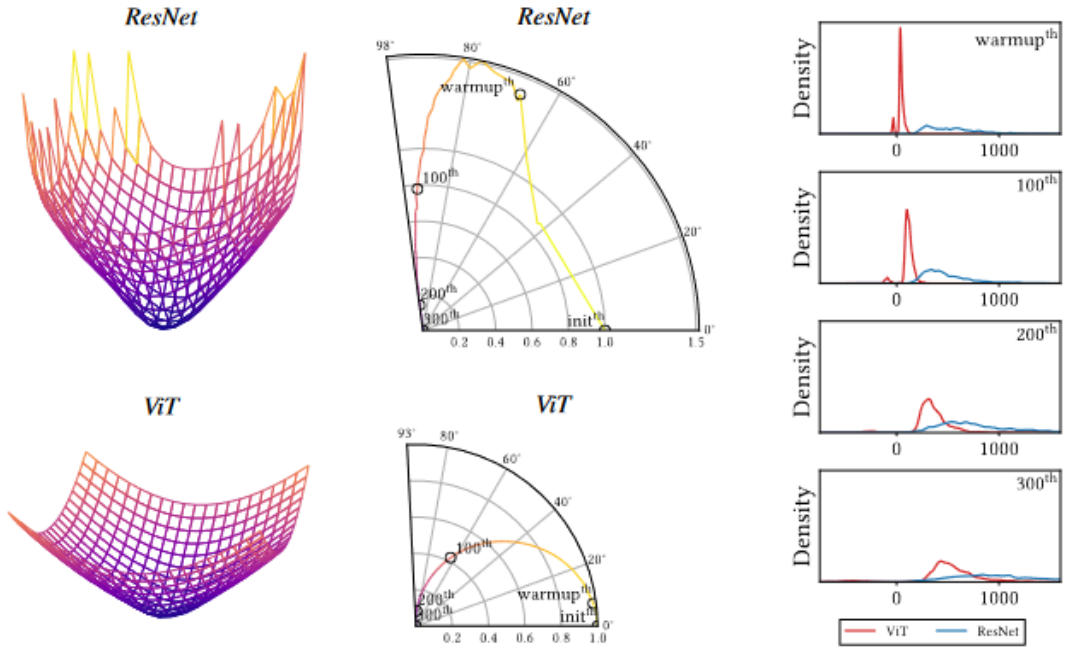
```
from einops import rearrange
proj = nn.Linear((patch_size**2)*channels, dim)
x_p = rearrange(img, 'b c (h p) (w p) -> b (h w) (p p c)', p = patch_size)
embedding = proj(x_p)
```

or equivalently:

```
conv = nn.Conv2d(channels, dim, kernel_size=patch_size, stride=patch_size)
embedding = rearrange(conv(img), 'b c h w -> b (h w) c')
```

ViT vs ResNet arxiv

- MHSAs and Convs exhibit opposite behaviors. **MHSAs are low-pass filters**, but **Convs are high-pass filters**.
- MHSAs improve not only accuracy but also generalization by flattening the loss landscapes.



(a) Loss landscape visualizations (b) Trajectories in polar coordinates (c) Hessian max eigenvalue spectra

Figure 1: **Three different aspects consistently show that MSAs flatten loss landscapes.** *Left*: Loss landscape visualizations show that ViT has a flatter loss (NLL + ℓ_2 regularization) than ResNet. *Middle*: ViT converges to the optimum along a smooth trajectory. In the polar coordinate, $r_t = \frac{||\Delta \boldsymbol{w}_t||}{||\Delta \boldsymbol{w}_{\text{init}}||}$ and $\theta_t = \cos^{-1}\left(\frac{\Delta \boldsymbol{w}_t \cdot \Delta \boldsymbol{w}_{\text{init}}}{||\Delta \boldsymbol{w}_t|| ||\Delta \boldsymbol{w}_{\text{init}}||}\right)$ where $\Delta \boldsymbol{w}_t = \boldsymbol{w}_t - \boldsymbol{w}_{\text{optim}}$ and $\boldsymbol{w}_t$ is NN weight at t^{th} epoch. $\boldsymbol{w}_{\text{init}}$ and $\boldsymbol{w}_{\text{optim}}$ are the weights at initialization and optimum. *Right*: The magnitude of the Hessian eigenvalues of ViT is smaller than that of ResNet. See Fig. 4 for a more detailed analysis.

ViT vs ResNet

ViT models are less effective in capturing the high-frequency components (related to local structures) of images than CNN models [1↗].
For ViTs to capture high-frequency components:

- knowledge distillation using a CNN teacher model [2↗].
- utilizing convolutional-like operation or multi-scale feature maps.
- use **RandAugment** [3↗].

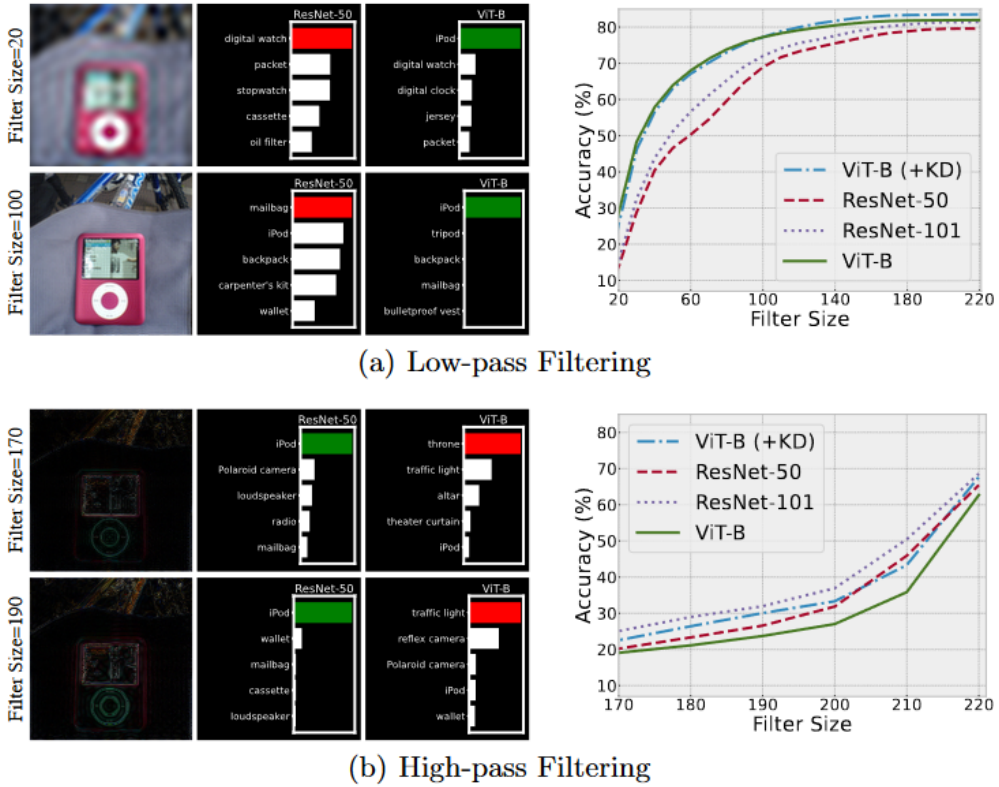
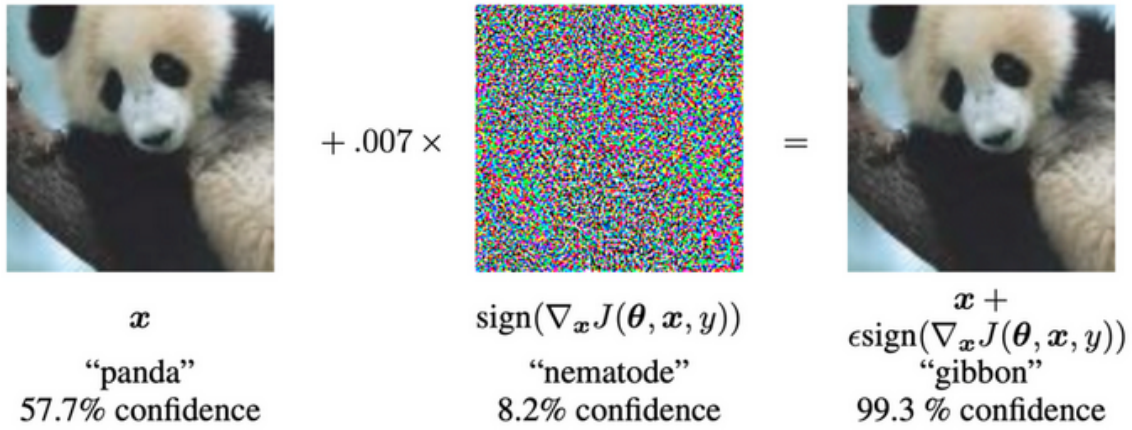


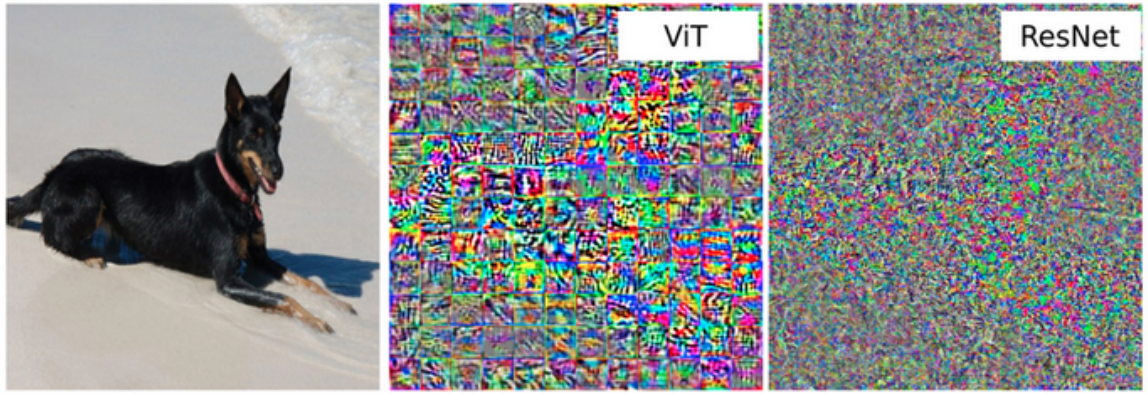
Fig. 1: Comparison of ViT-B, ResNet-50, ResNet-101, and ViT-B(+KD) on low- and high-pass filtered validation set with different filter sizes. ViT-B is the base ViT model taking as input a sequence of 16×16 patches. KD denotes knowledge distillation, where the teacher model is a RegNetY-16GF [46] following [54]. The top-1 accuracy of ViT-B, ResNet-101, ResNet-50, and ViT-B (+KD) on the ImageNet validation set is 82.0%, 79.8%, , 81.6%, and 83.6%, respectively.

ViT vs ResNet

Robustness to input perturbations:

- **ResNet**: noise has a high frequency component and localized structure [1↑]
- **ViT**: relatively low frequency component and a large structure (The border is clearly visible in the 16×16 size patch.)
  - When pre-trained with a sufficient amount of data, ViT are at least as robust as the ResNet counterparts on a broad range of perturbations [2↑].





ViT vs ResNet arxiv

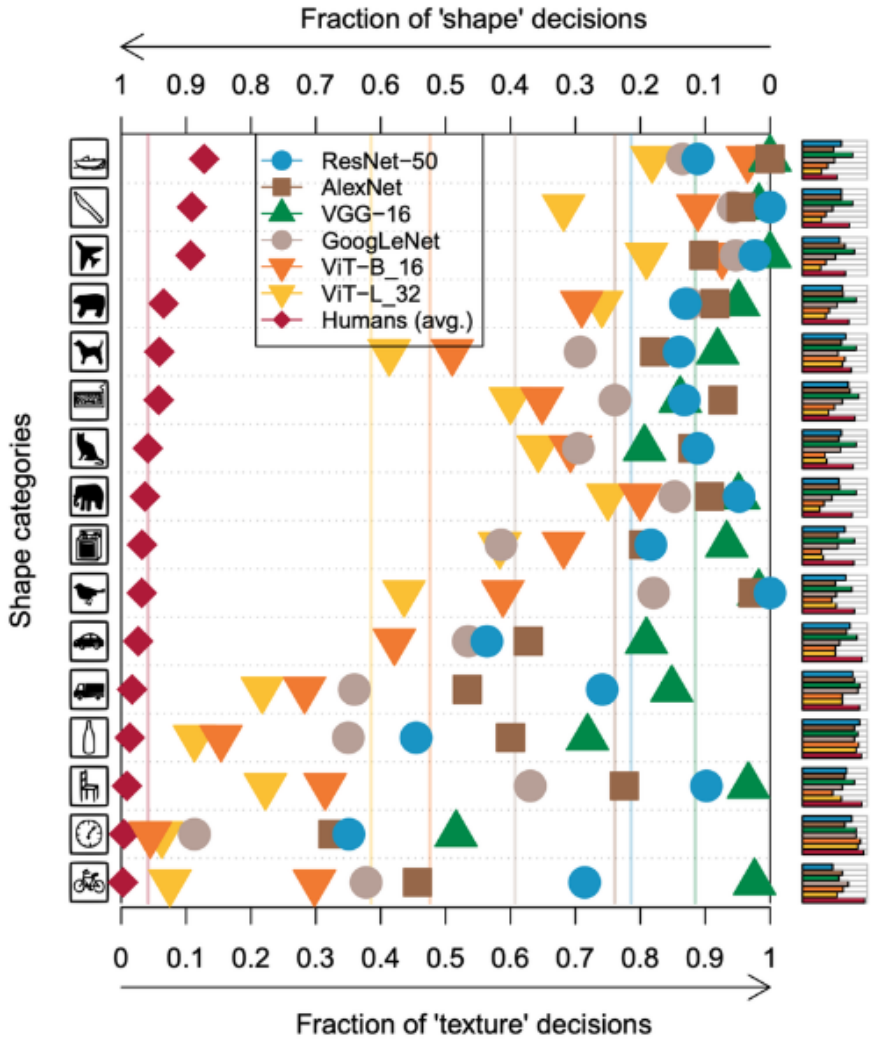
Is dicision based on texture or shape?

- **ResNet**: relies on texture rather than shape [1]
- **ViT**: little more robust to texture perturbation
- **Human**: much robust to texture perturbation

(a) Texture image
81.4% **Indian elephant**
10.3% indri
8.2% black swan

(b) Content image
71.1% **tabby cat**
17.3% grey fox
3.3% Siamese cat

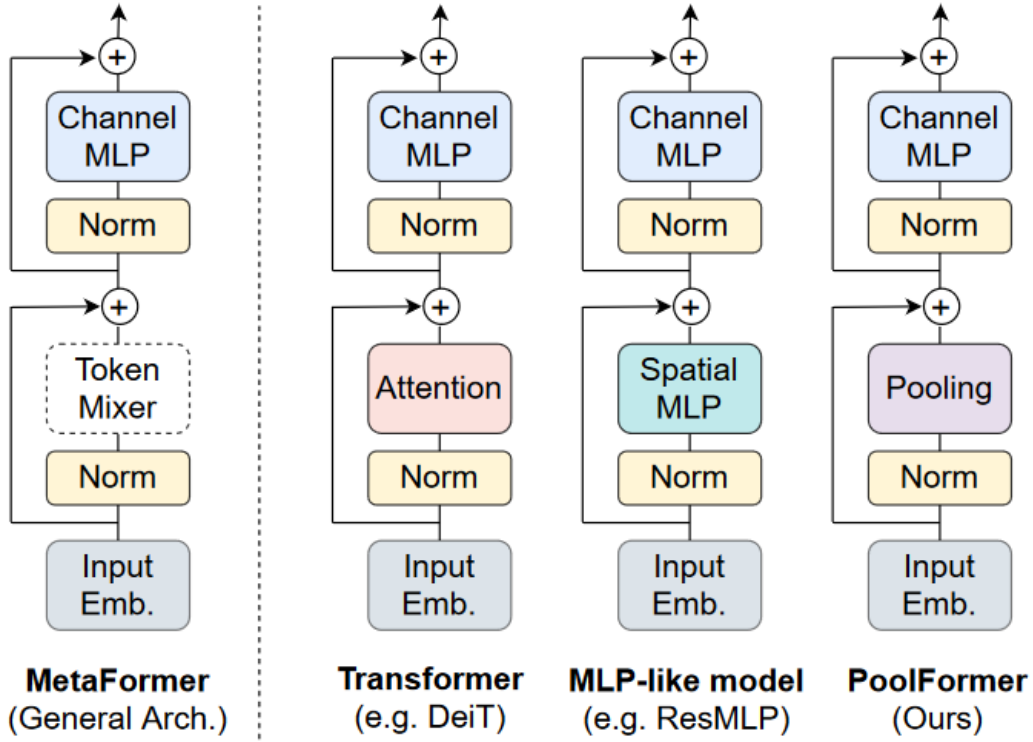
(c) Texture-shape cue conflict
63.9% **Indian elephant**
26.4% indri
9.6% black swan



Do We Really Need Attention? arxiv

> It turns out that it is **not** *"Attention is All You Need"*.

- As long as the tokens can be mixed, MetaFormer architecture can achieve the similar performance as the Transformer.
- Meta architecture of transformer layer can be viewed as a *special case of ResNet* layer with following componets:
  - Token mixers (self-attention, etc.)
  - Position Encoding
  - Channel MLP (1x1 convolution)
  - Normalization (LayerNorm, etc.)



CLIP arxiv github

Learning directly from raw text about images leverages *a broader source of supervision* compared to using a fixed set of predetermined object categories. CLIP introduced a simple yet effective contrastive objective to learn a vision-language embedding space, where similar concepts are pulled together and different concepts are pushed apart.

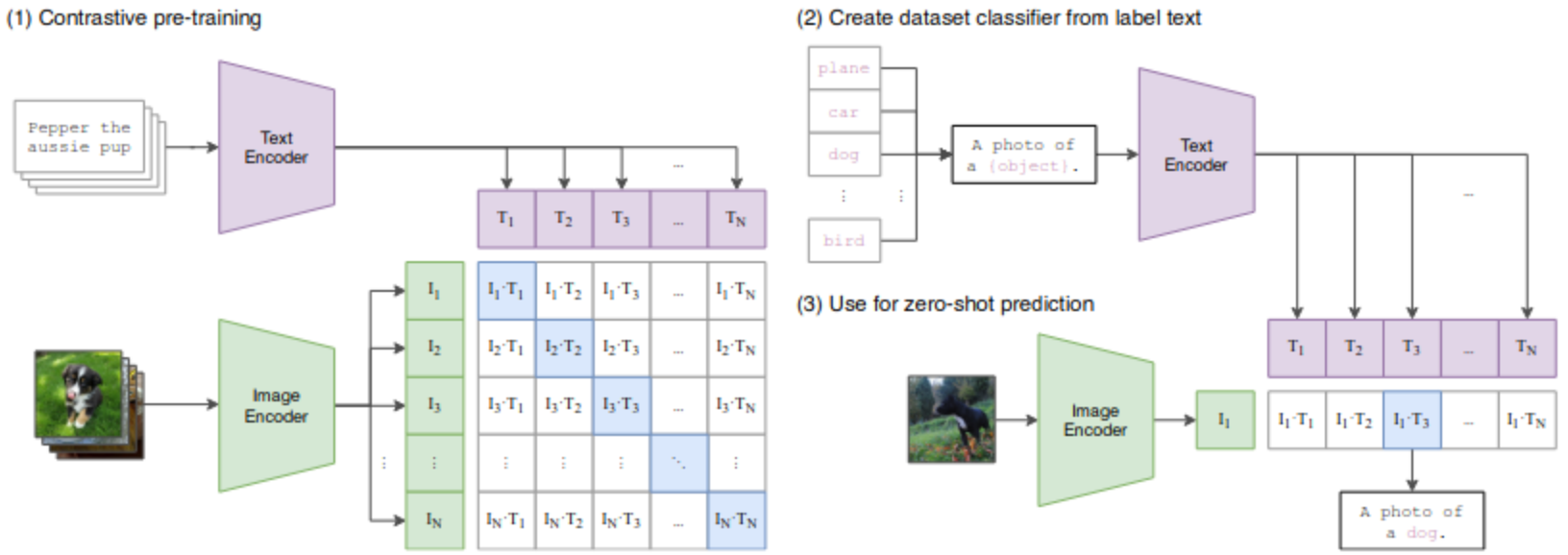
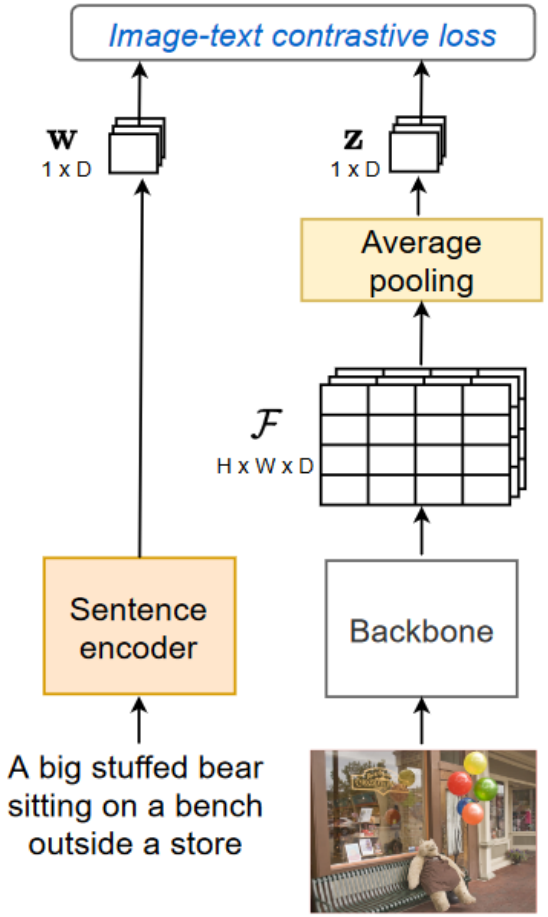


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

CLIP arxiv github

Sample implementation:

```
# 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
```



(a) ALIGN / CLIP

Object Detection

YOLO

DETR

Segmentation

LSeg arxiv github

The text embeddings provide a flexible label representation and help generalize to previously unseen categories at test time, without retraining or even requiring a single additional training sample.

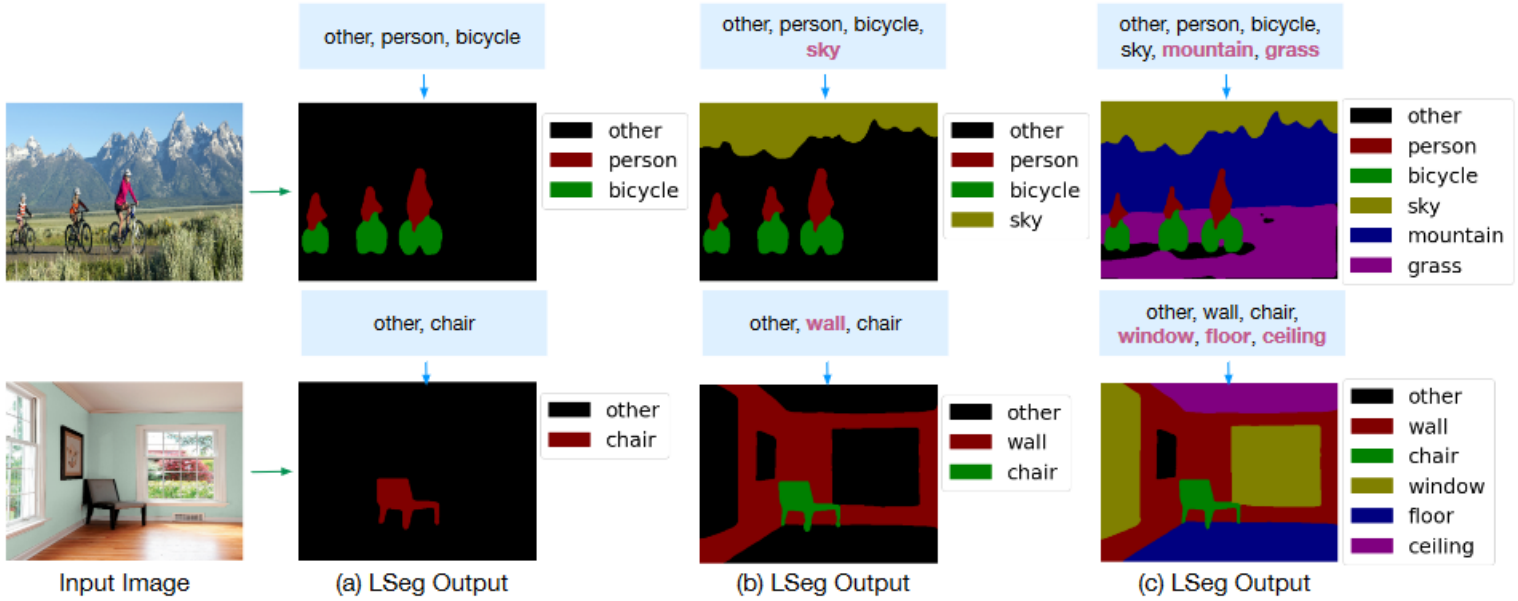


Figure 1: Example results. LSeg is able to handle unseen labels as well as label sets of arbitrary length and order. This enables flexible synthesis of zero-shot semantic segmentation models on the fly. From left to right, labels that are removed between runs are underlined, whereas labels that are added are marked in **bold red**.

LSeg arxiv github

Language-driven Semantic Segmentation (LSeg) embeds text labels and image pixels into a common space, and assigns the closest label to each pixel.

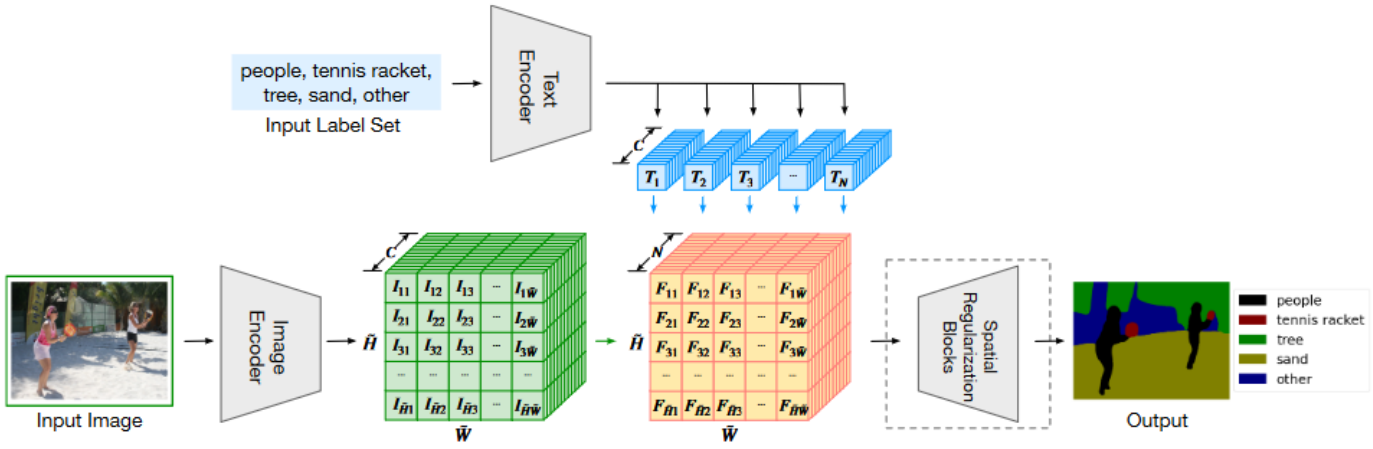
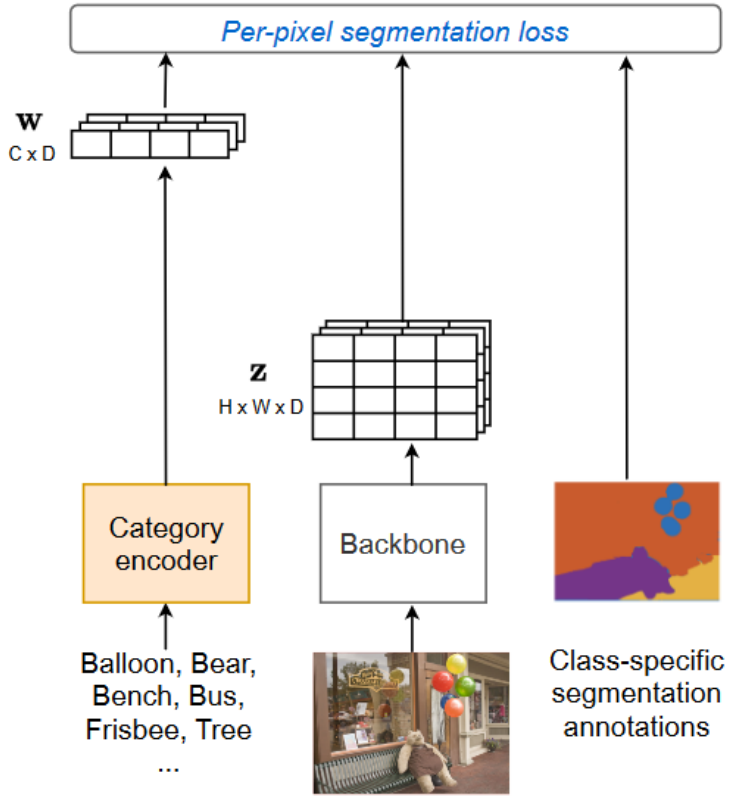


Figure 2: Overview. A text encoder embeds labels into a vector space. An image encoder extracts per-pixel embeddings from the image and correlates the feature of each pixel to all label embeddings. The image encoder is trained to maximize the correlation between the text embedding and the image pixel embedding of the ground-truth class of the pixel. A final spatial regularization block spatially regularizes and cleans up the predictions.



(b) Per-pixel segmentation

SAM arxiv

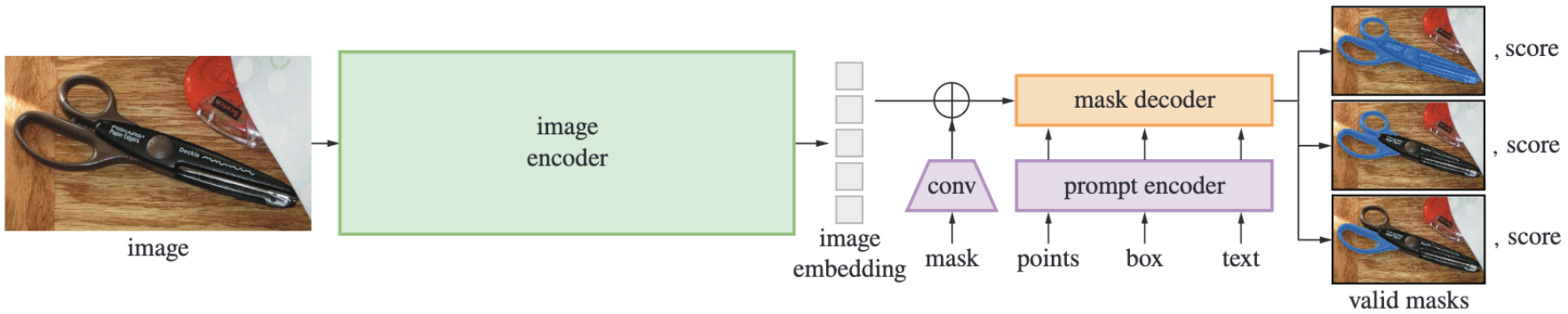


Figure 4: Segment Anything Model (SAM) overview. A heavyweight image encoder outputs an image embedding that can then be efficiently queried by a variety of input prompts to produce object masks at amortized real-time speed. For ambiguous prompts corresponding to more than one object, SAM can output multiple valid masks and associated confidence scores.