LSDL Lecture 01
How to use a pre-trained model?

Ildus Sadrtdinov, 09.09.2024

Large datasets

1) Impossible to collect large amounts of high-quality data

image link

2) Too expensive to annotate → **self-supervised learning**

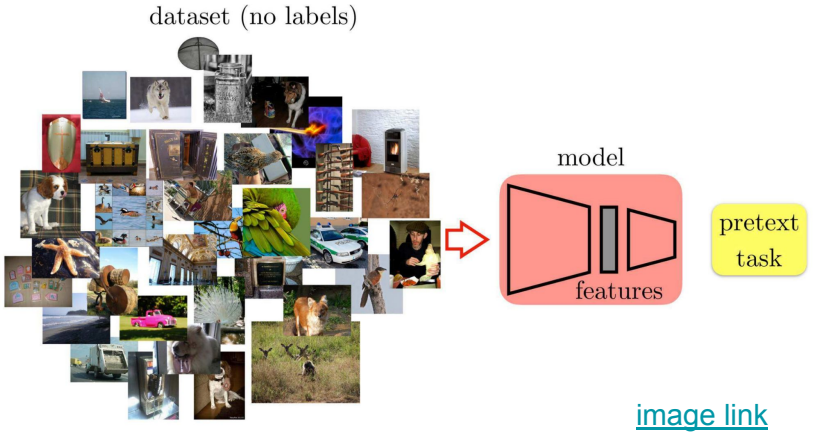


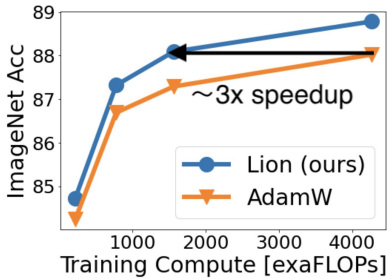
image link

Large models

1) What training algorithms to use?

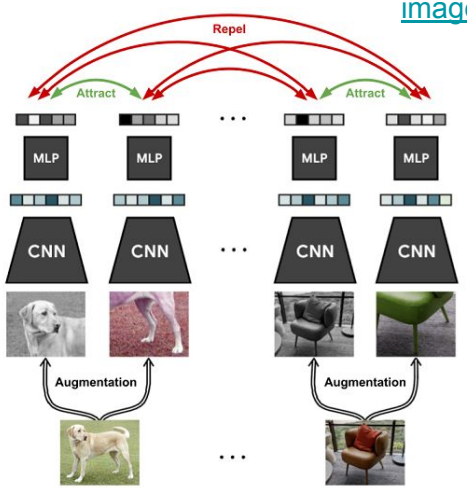
2) Too large to fit in a single GPU

image link

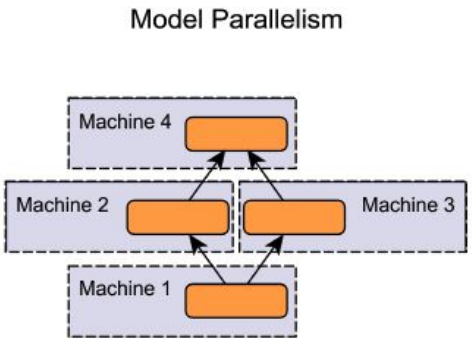


optimizers

image link



training objectives



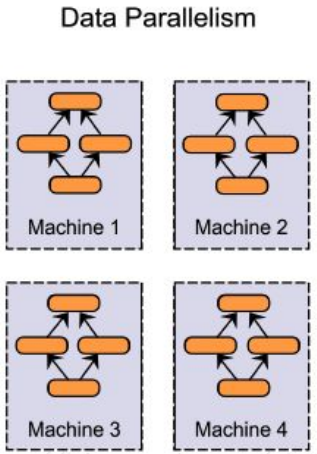
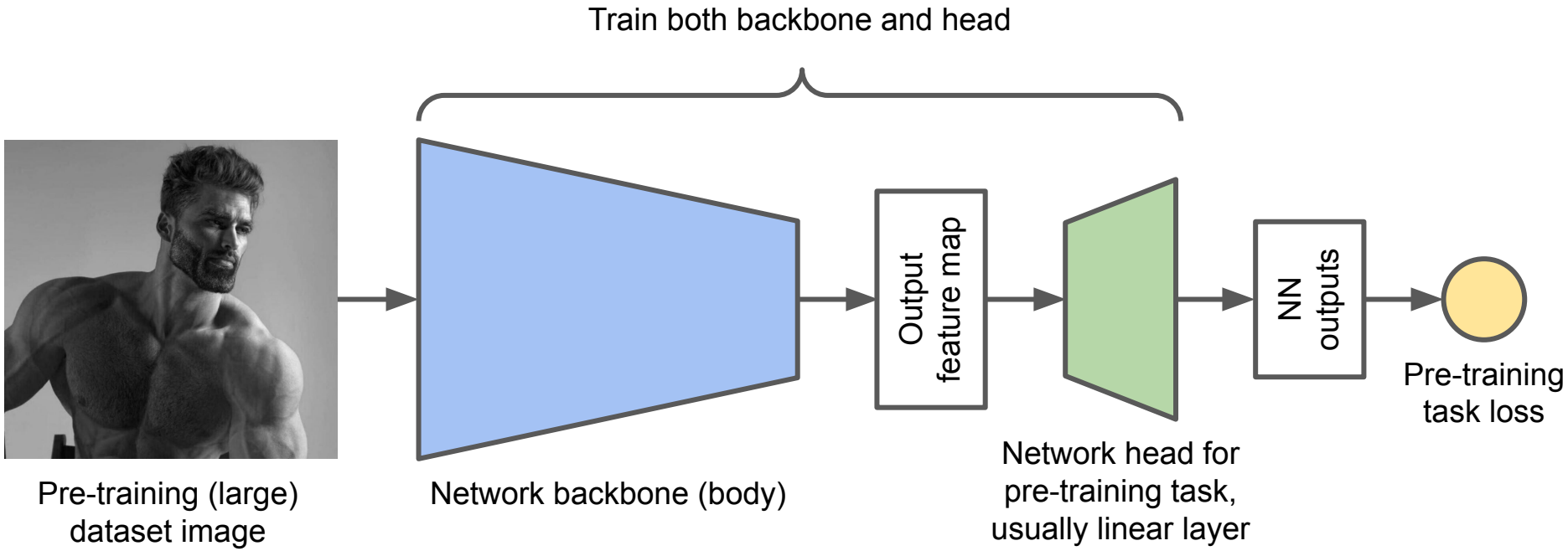


image link

Course overview

- **Vision:** classic pre-text tasks, contrastive learning,
  masked image modelling
- **NLP:** Parameter-Efficient Fine-Tuning (PEFT),
  Retrieval Augmented Generation (RAG),
  Reinforcement Learning with Human Feedback (RLHF)
- **Misc:** ensembling, weight averaging,
  pruning, quantization, distillation
- **Audio:** contrastive & masked modelling approaches
- **Modern optimizers (?)**

Transfer learning: pre-training



Transfer learning: fine-tuning

Downstream (small) dataset image

Train both backbone and head

Network backbone (body)

Output feature map

Network head for downstream task

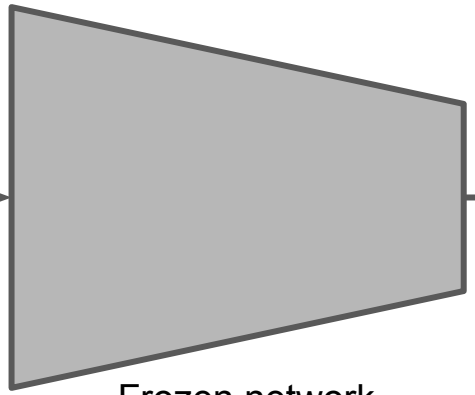
Drop pre-training head, replace with randomly init for fine-tuning

NN outputs

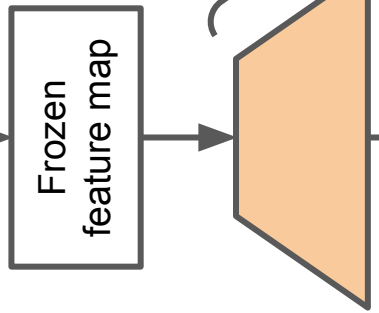
Downstream task loss

Transfer learning: linear probing

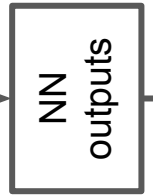
Downstream (small) dataset image

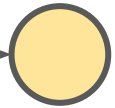


Frozen network backbone (body)

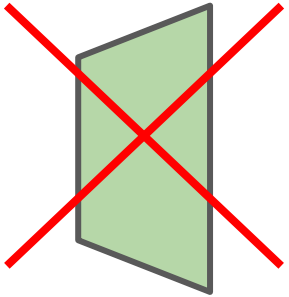


Network head for downstream task





Downstream task loss



Drop pre-training head, replace with randomly init for linear probing

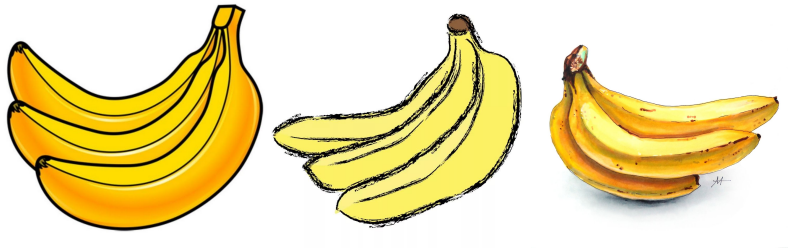
Train only head

When to use fine-tuning or linear probing?

- Fine-tuning → better **in-distribution (ID)** accuracy
- Linear probing → better **out-of-distribution (OOD)** accuracy

In-distribution

Out-of-distribution



Fine-tuning distorts pre-trained features

$$\widehat{L}(v,B) = \|XB^\top v - Y\|_2^2$$

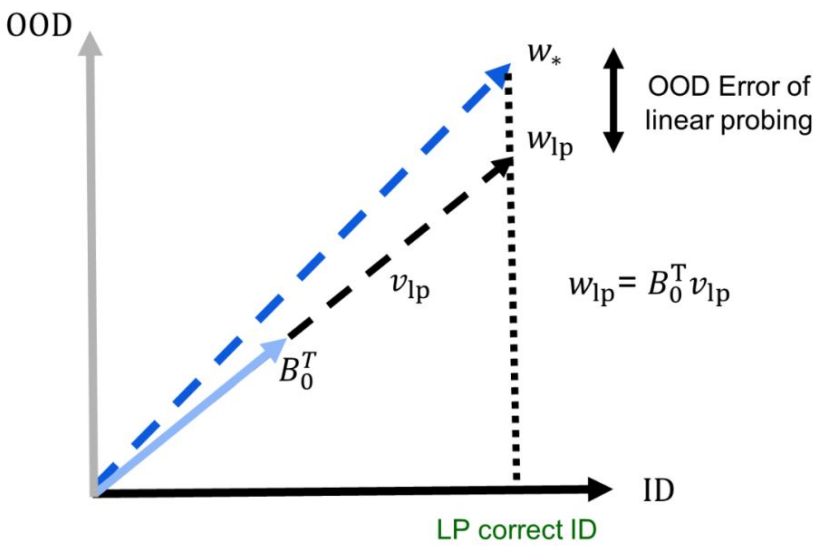
$$\nabla_B \widehat{L}(v,B) = 2v(Y - XB^\top v)^\top X$$

$$\nabla_B \widehat{L}(v,B)u = 0$$

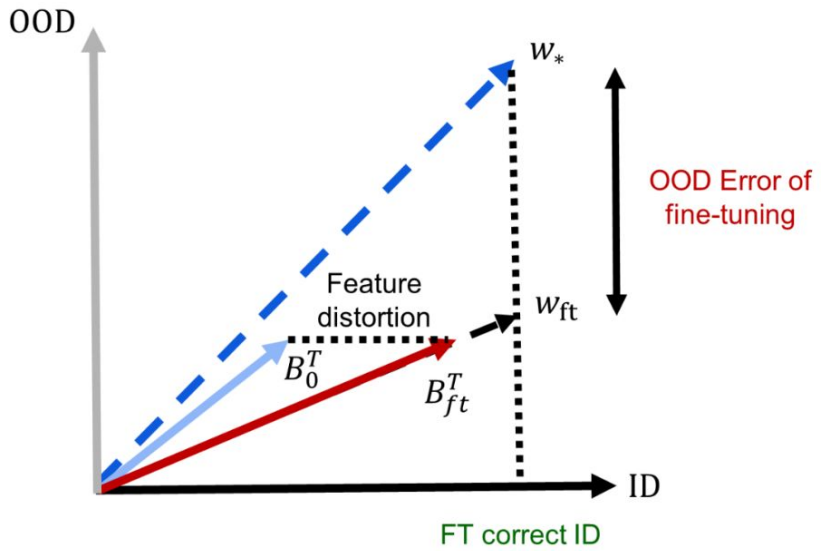
$$B^{k+1}u = \left(B^k - \eta \nabla_{B^k} \hat{L}(v, B^k)\right)u = B^k u$$

Kumar et al., Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution, 2022, https://arxiv.org/pdf/2202.10054

Fine-tuning distorts pre-trained features

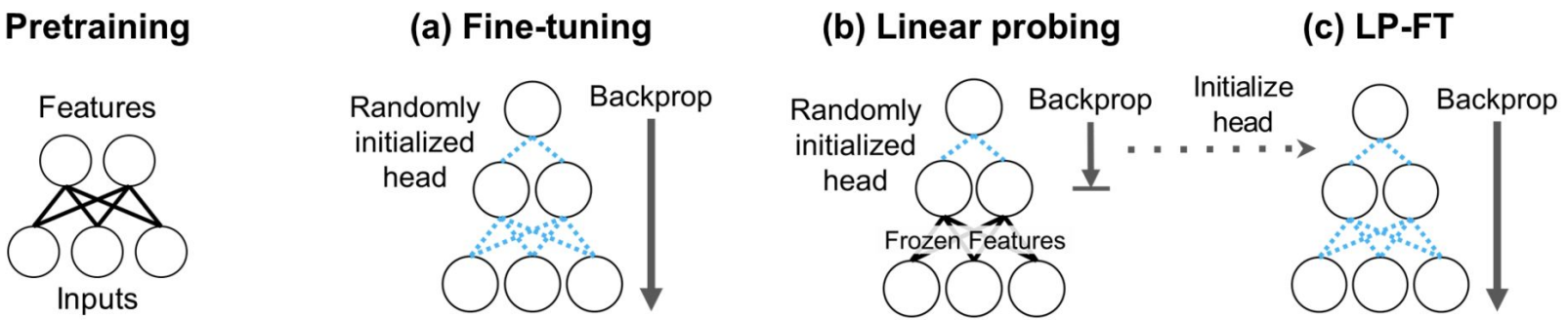


(a) Toy example (Linear probing)



(b) Toy example (fine-tuning)

Fine-tuning distorts pre-trained features



ID test	85.1%	82.9%	**85.7%**
OOD test	59.3%	66.2%	**68.9%**

Average accuracies (10 distribution shifts)

Fine-tuning distorts pre-trained features

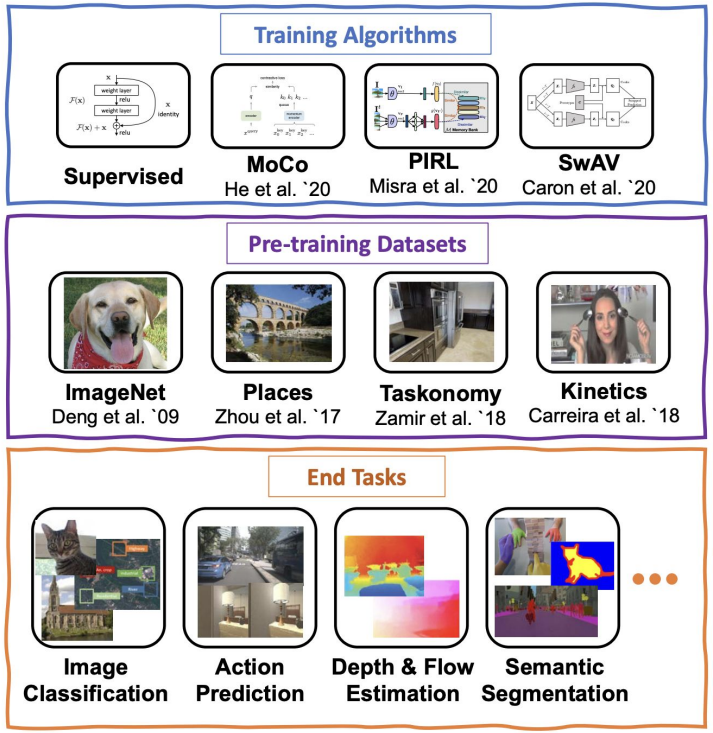
- Early stopping does not mitigate feature distortion
- ID features are distorted more than OOD features

	ID	**OOD**
FT	0.0188 ± 0.0001	0.0167 ± 0.0001
LP-FT	0.0011 ± 0.0001	0.0009 ± 0.0001

- Pre-trained features must be suitable for the task
- If ID ≈ OOD, then FT is better than LP

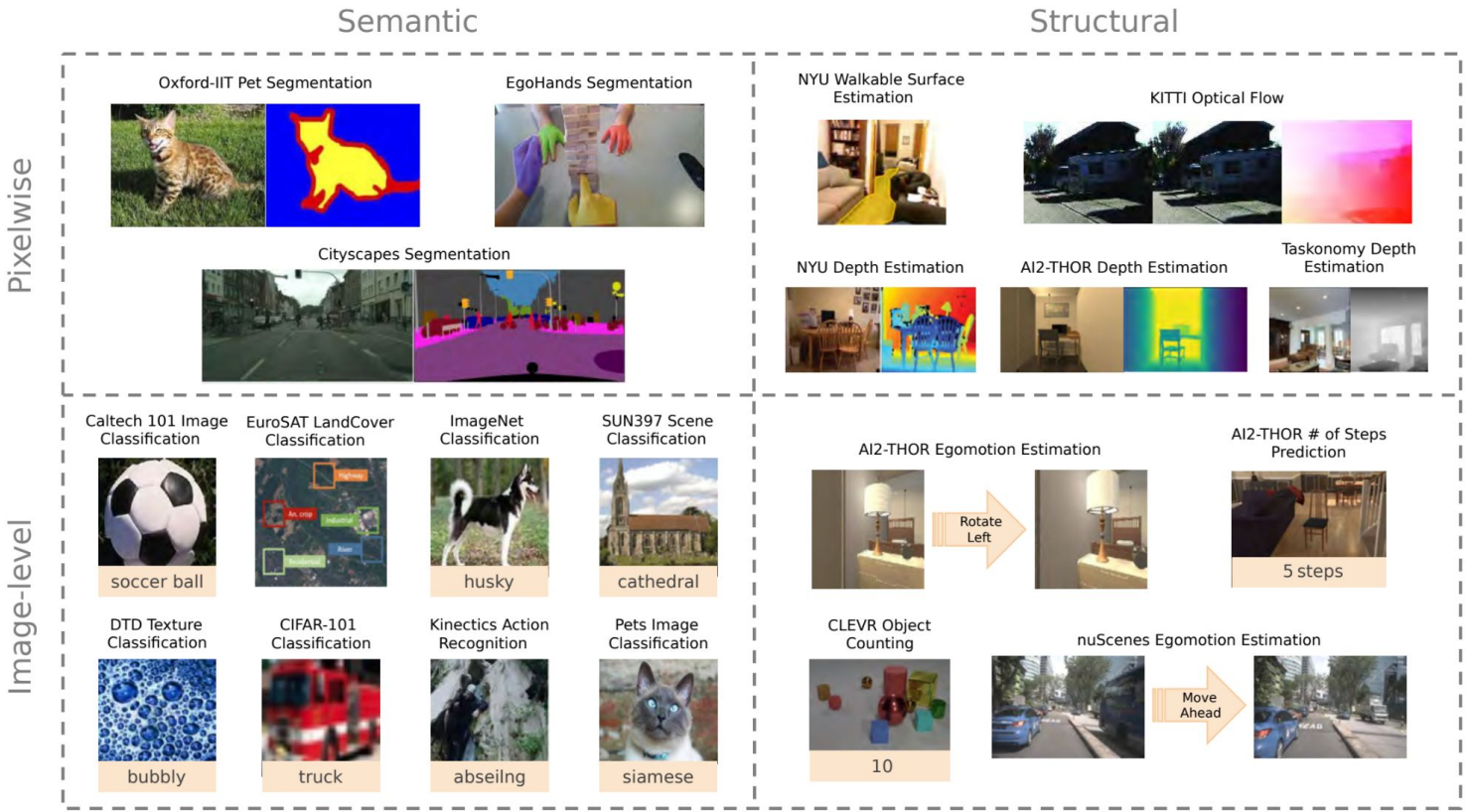
Which pre-training task to choose?

- Supervised or self-supervised?
- Which dataset to use for pre-training?
- Which network architecture to use?*

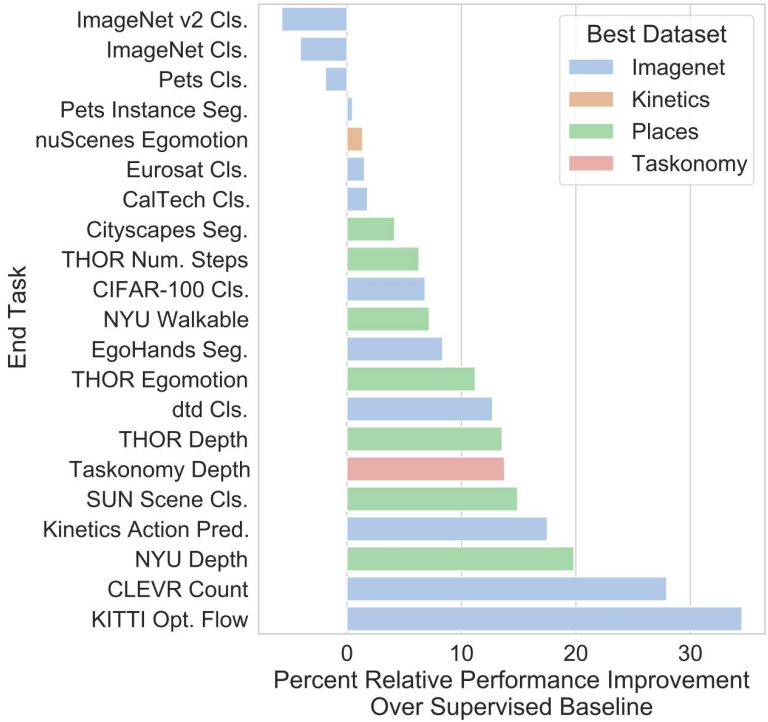


Kotar et al., Contrasting Contrastive Self-Supervised Representation Learning Pipelines, 2021, https://arxiv.org/pdf/2103.14005

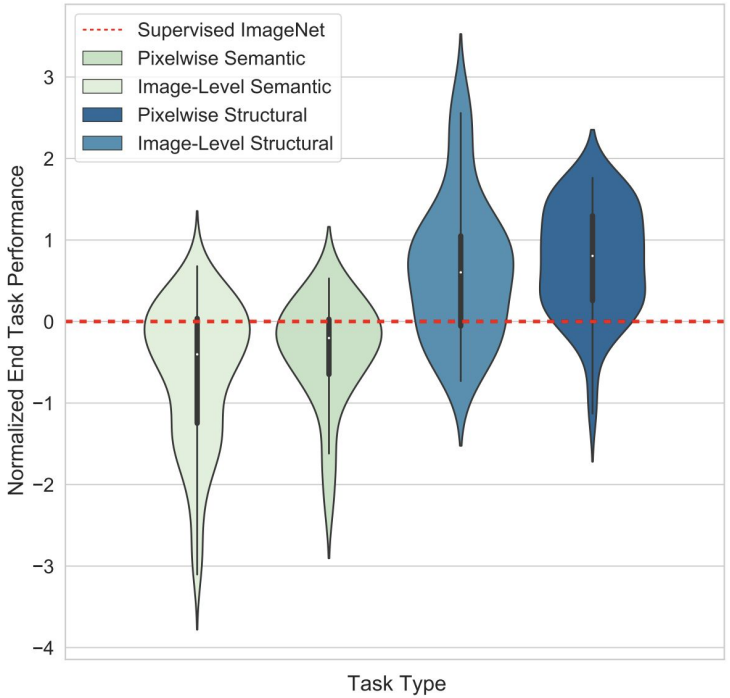
Categorization of downstream tasks



Supervised vs. self-supervised

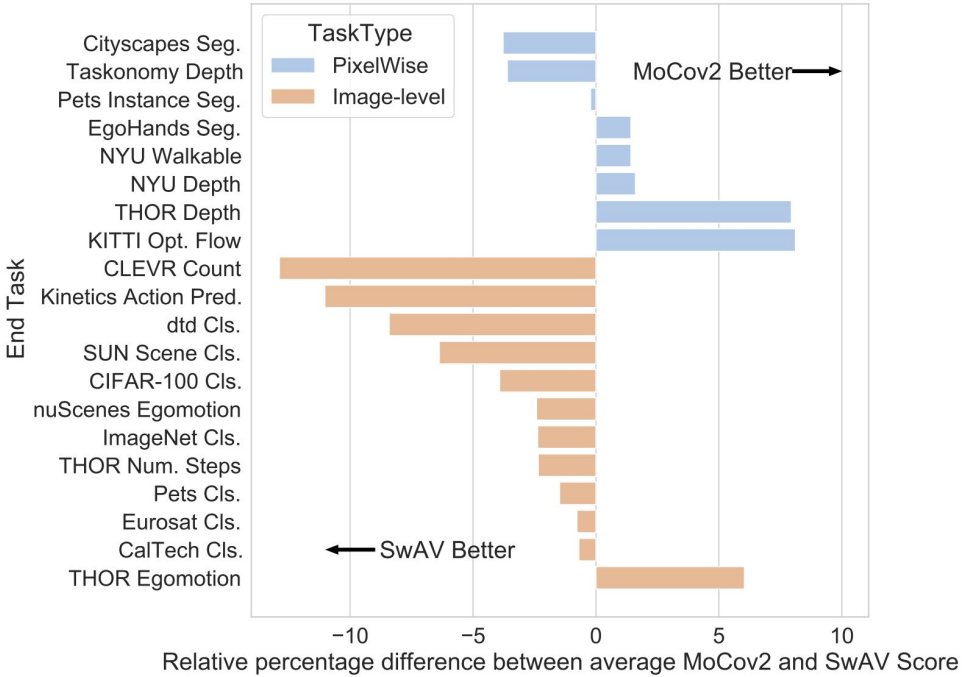


Comparison on different datasets



Comparison on different tasks

Self-supervised vs. self-supervised



Comparison of SwAV and MoCov2 (two self-supervised algorithms)

Is PT accuracy a good proxy for downstream quality?

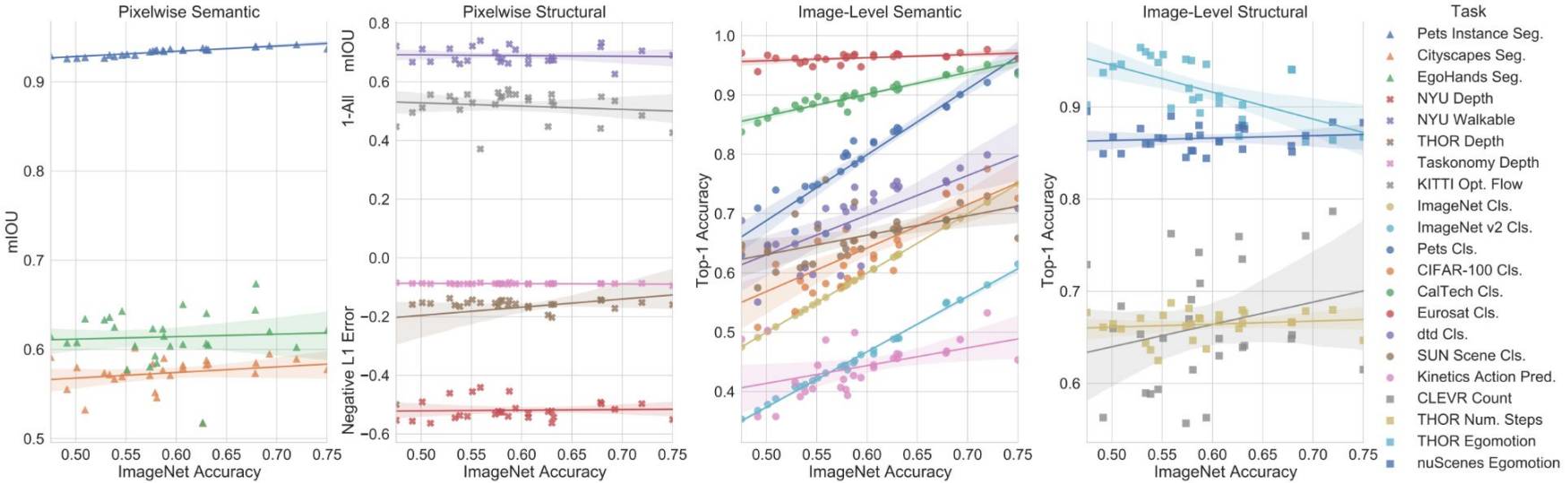


Figure 4. **Correlation of end task performances with ImageNet classification accuracy.** The plots show the end task performance against the ImageNet top-1 accuracy for all end tasks and encoders. Each point represents a different encoder trained with different algorithms and datasets. This reveals the lack of a strong correlation between the performance on ImageNet classification and tasks from other categories.

Does the pre-training dataset matter?

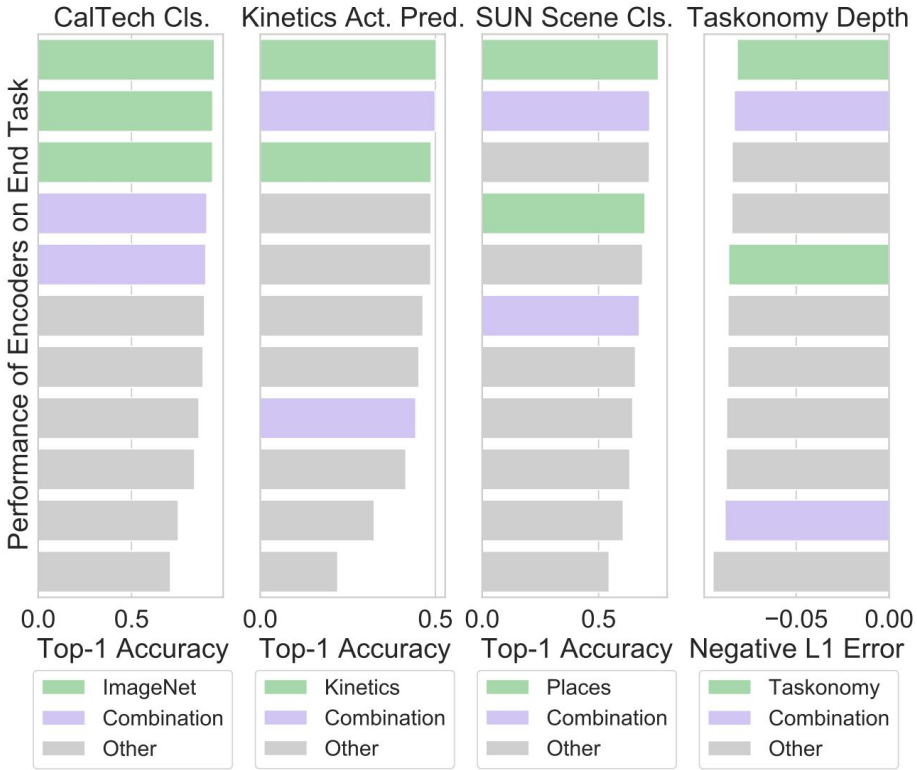
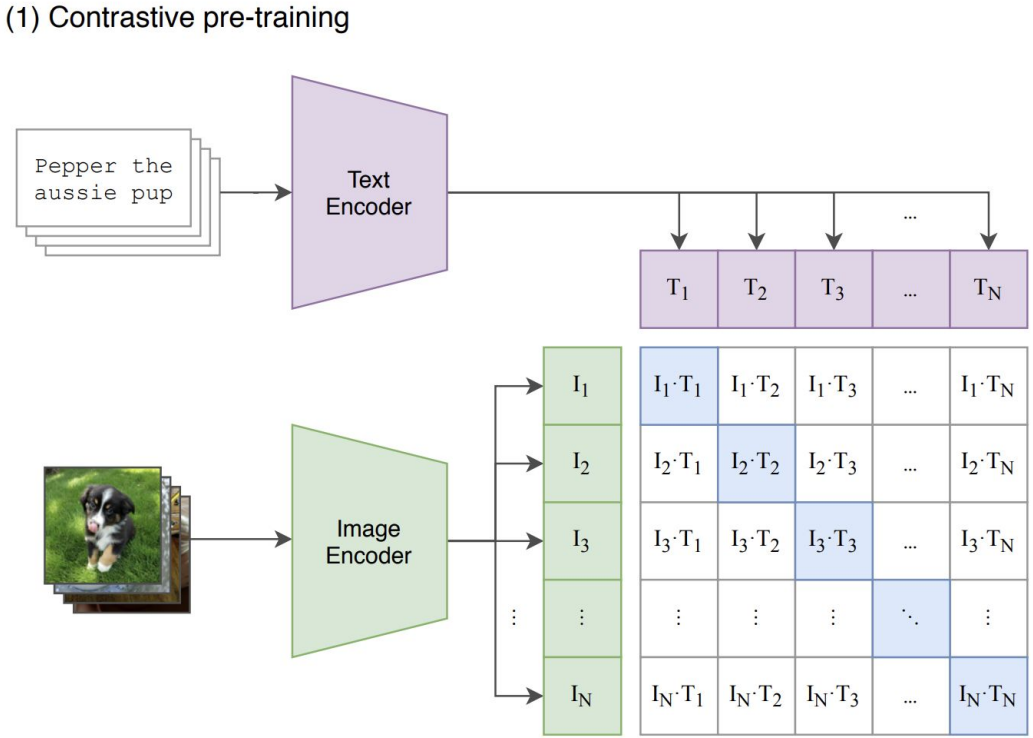


Figure 7. **Similarity of the pre-training datasets and end tasks.**

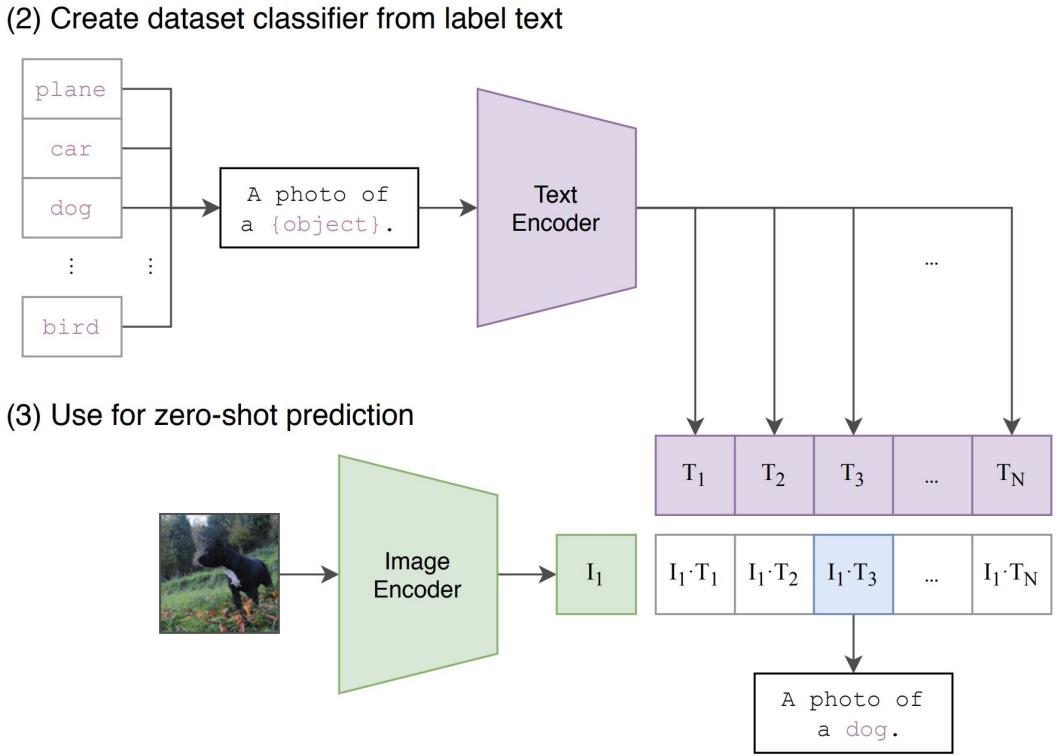
CLIP: Contrastive Language-Image Pre-training



- Encode images and texts into the same embedding space
- Bring together embeddings of pairs (image, caption)
- Push apart embeddings from different pairs

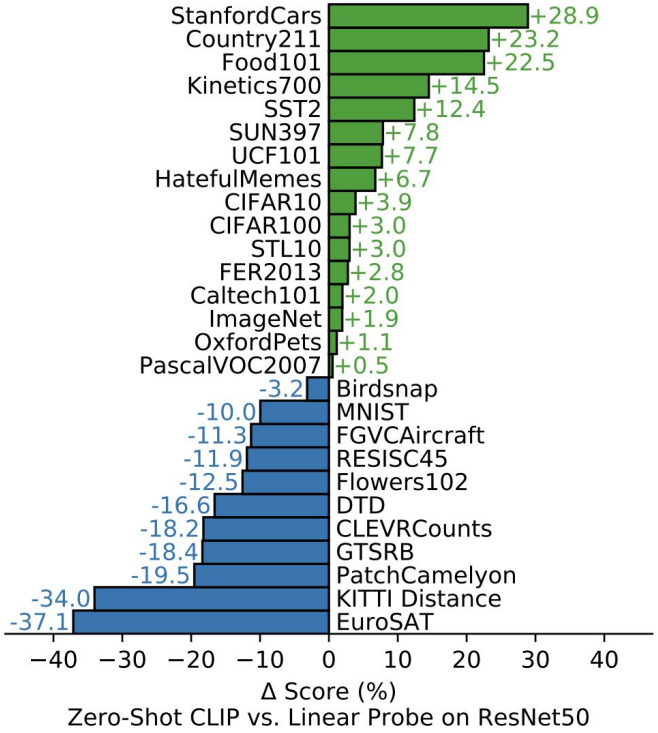
Radford et al., Learning Transferable Visual Models From Natural Language Supervision, 2021, https://arxiv.org/pdf/2103.00020

CLIP: Contrastive Language-Image Pre-training

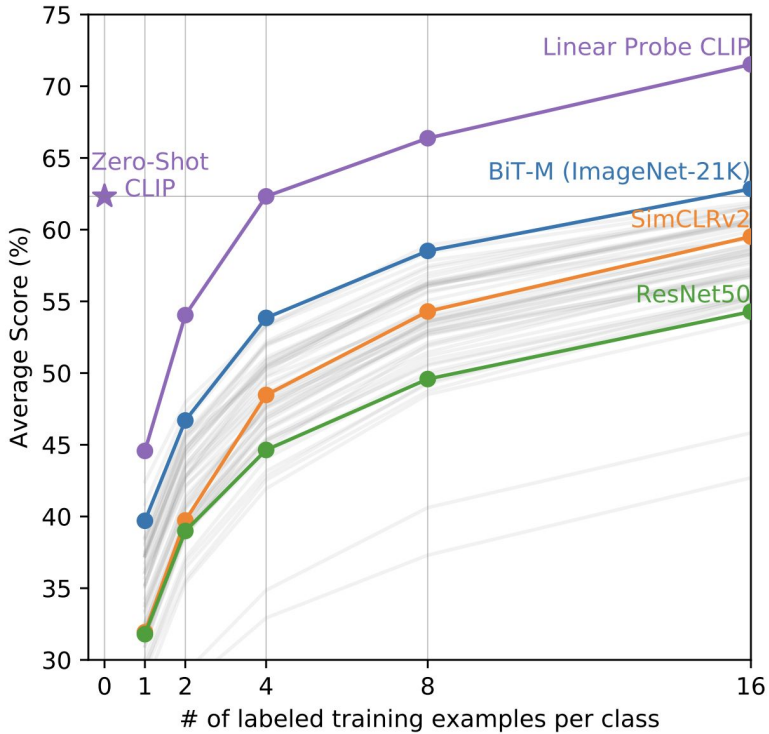


- Get image embedding
- Calculate similarity to embeddings of class texts
- Use similarity scores as classification logits

CLIP is a strong zero-shot predictor

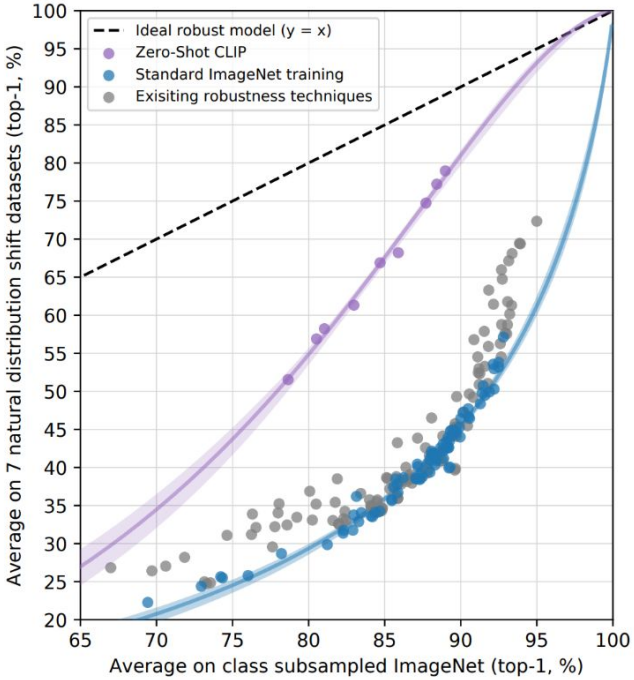


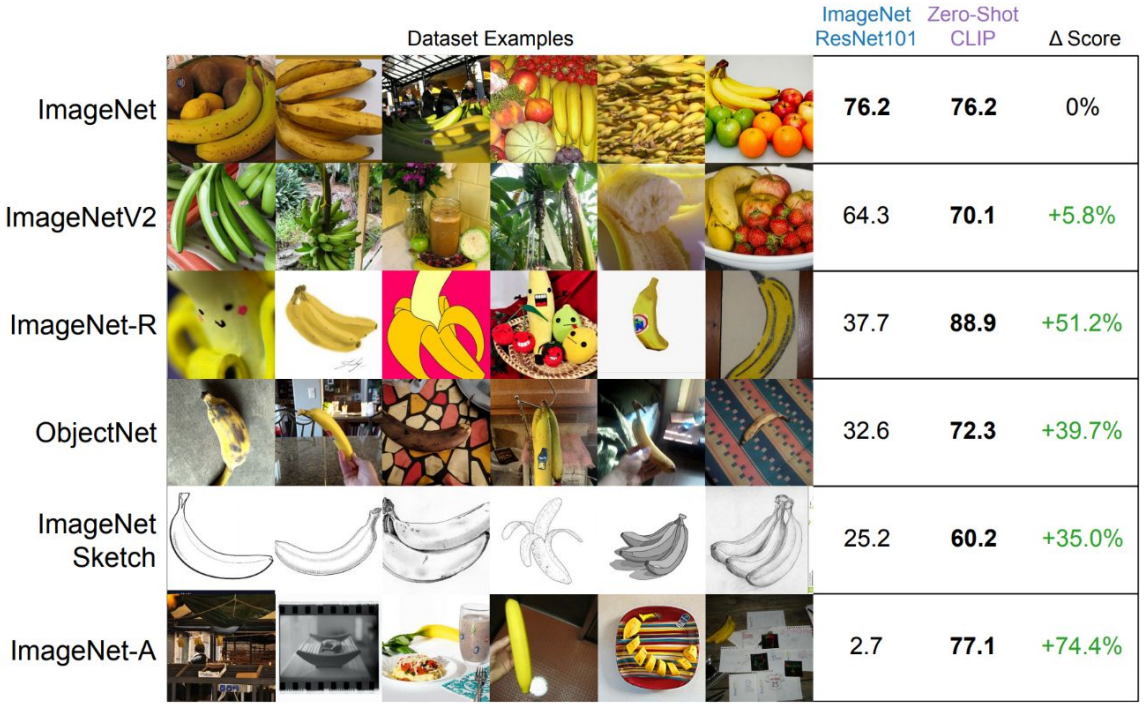
Zero-shot CLIP is competitive with a fully supervised baseline



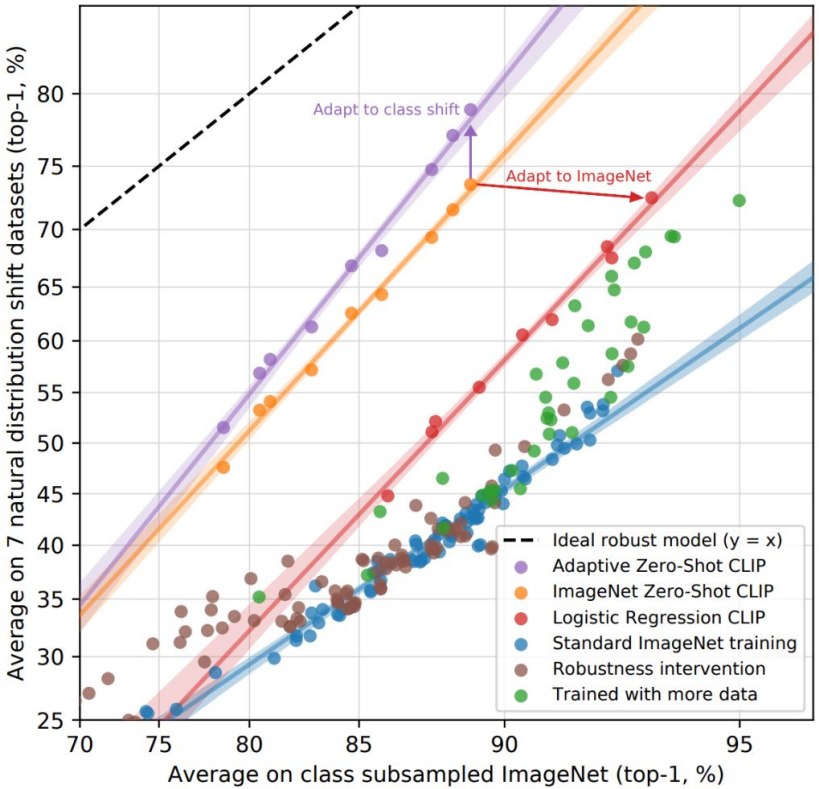
Zero-shot CLIP outperforms few-shot linear probes

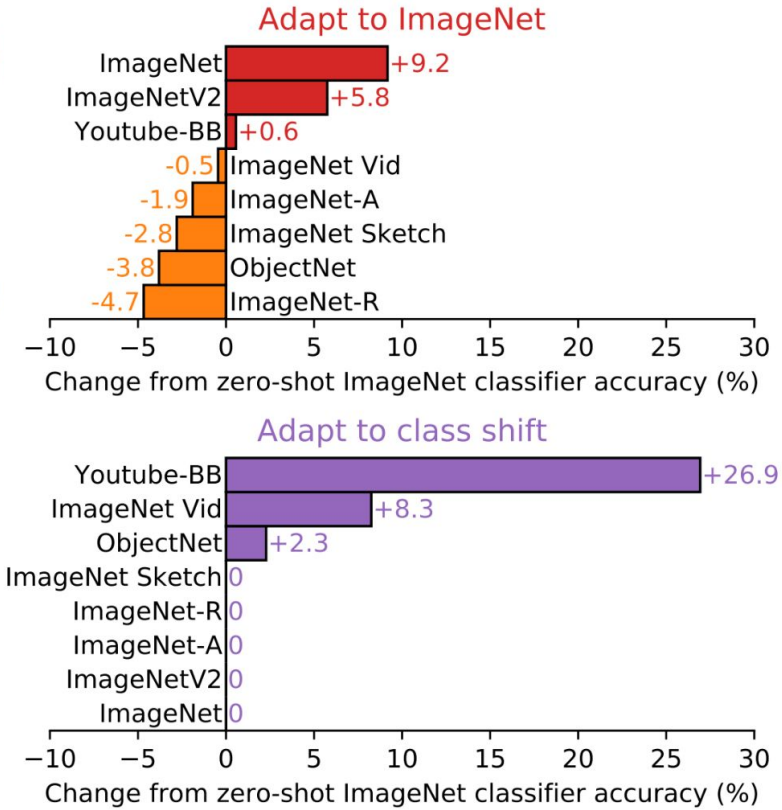
Zero-shot CLIP is a robust model



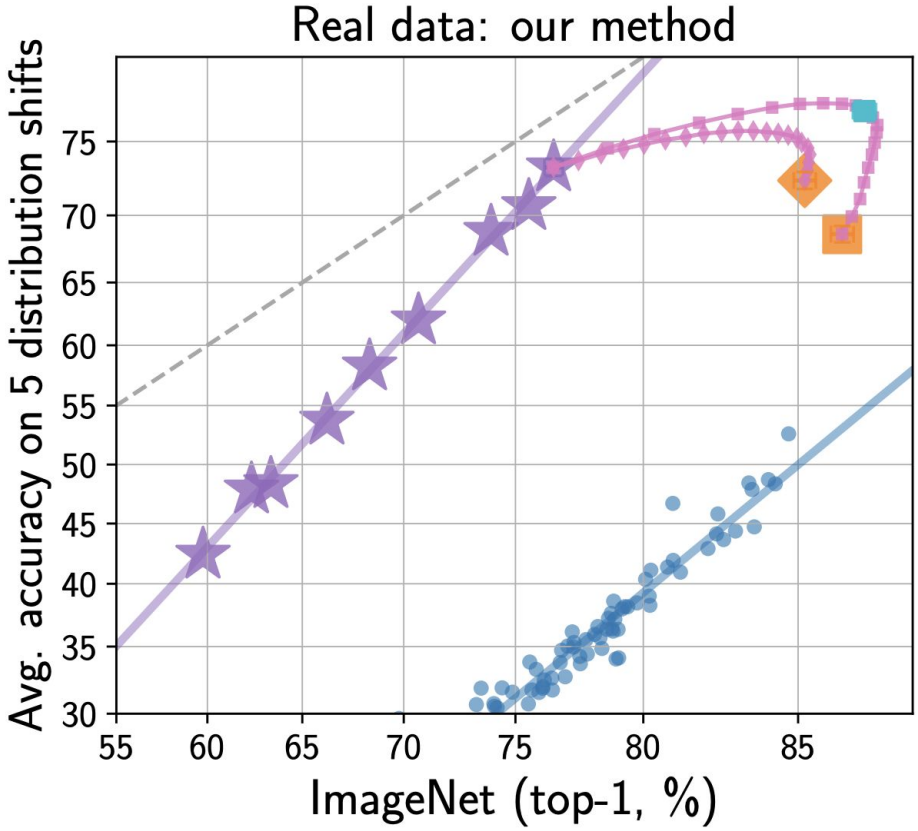


ID adaptation hurts robustness





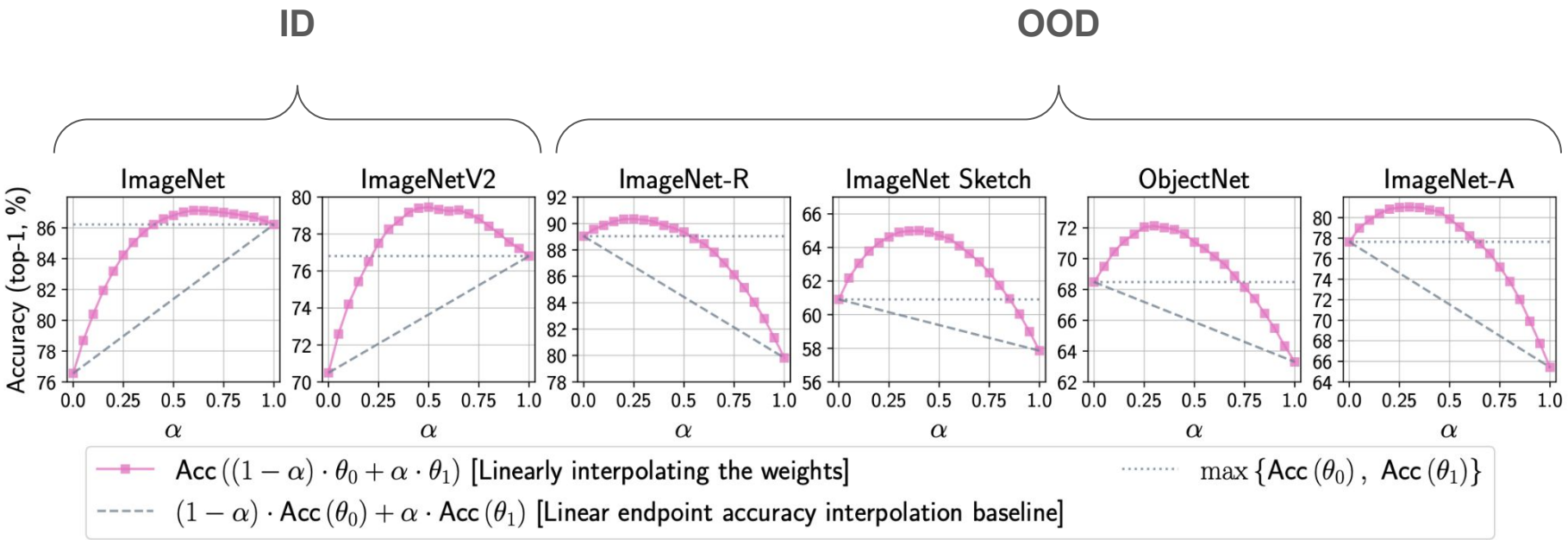
How to get both strong and robust model?



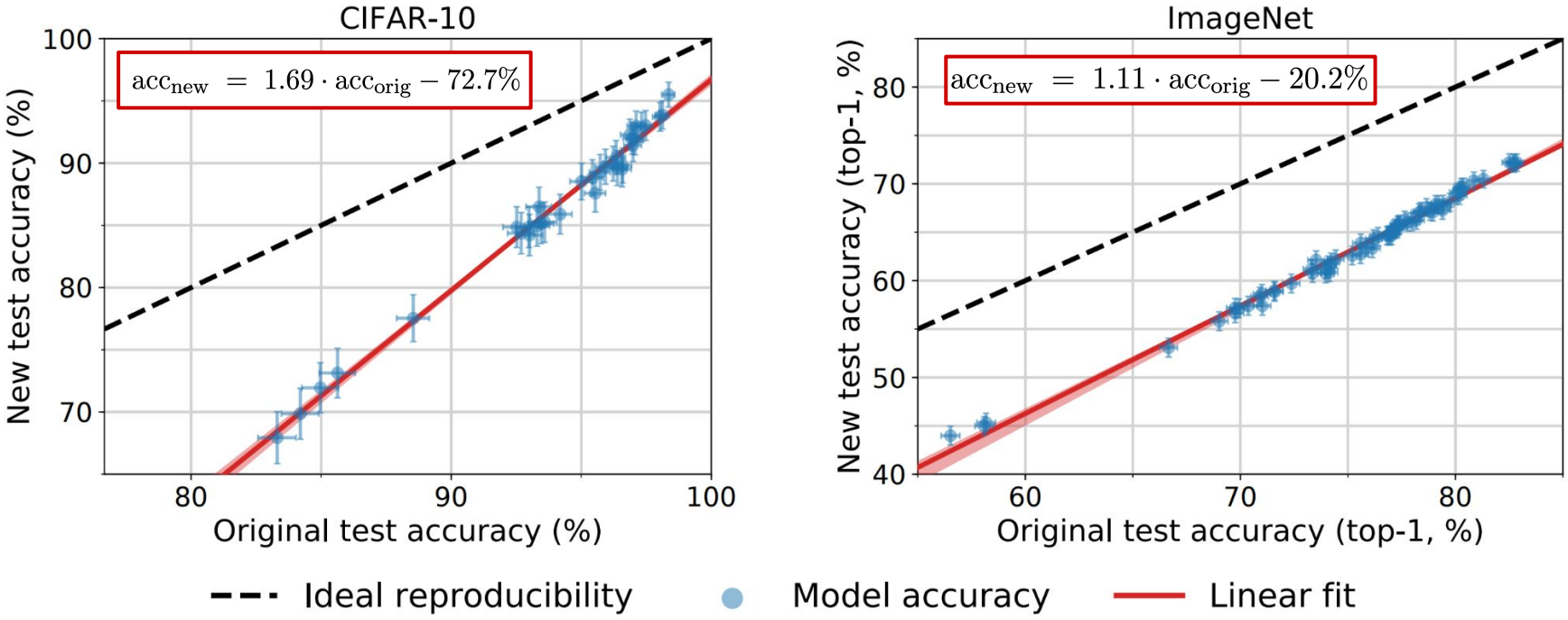
$$\text{wse}(x, \alpha) = f(x, (1-\alpha)\cdot\theta_0 + \alpha\cdot\theta_1)$$

Wortsman et al., Robust fine-tuning of zero-shot models, 2021, https://arxiv.org/pdf/2109.01903

Interpolating between fine-tuned and zero-shot solutions



Do ImageNet Classifiers Generalize to ImageNet?



Recht et al., Do ImageNet Classifiers Generalize to ImageNet?, 2019, https://arxiv.org/pdf/1902.10811

Do ImageNet Classifiers Generalize to ImageNet?

Validation set: 10 images/class

- **Matched frequency:** original selection frequency distribution for each class
- **Threshold0.7:** sample images with selection frequency > 0.7
- **TopImages:** select images with top frequency

Sampling Strategy	Average MTurk Selection Freq.	Average Top-1 Accuracy Change	Average Top-5 Accuracy Change
MatchedFrequency	0.73	-11.8%	-8.2%
Threshold0.7	0.85	-3.2%	-1.2%
TopImages	0.93	+2.1%	+1.8%