Lecture 6

EMPIRICAL PHENOMENA IN ROBUST GENERALIZATION

CS329D

Goals for today

3 major themes

From domain adaptation to generalization

How should we measure robustness to distribution shifts?

What kinds of robustness interventions seem to work well?

Roadmap

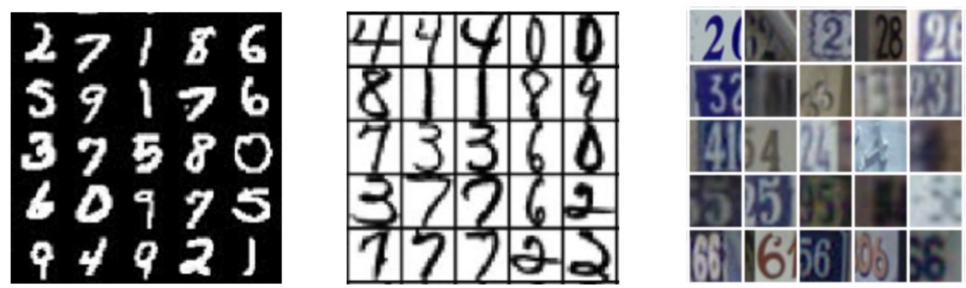
- Intro to Generalization
- Representation Learning
- Evaluating Generalization
- Measuring Robustness
    - Absolute, effective, and relative robustness
- Robustness Interventions
    - Model architectures, more/better data, adversarial robustness, pre-training, self-supervised learning
- Zero-shot Learning
    - Motivation
    - CLIP
    - NLP (through ChatGPT)

Our setting until now: Unsupervised Domain Adaptation

Task setup:
labeled source data +
unlabeled target data

Training data — Test domains

MNIST — USPS — SVHN



Key structure:
we have information about the target in the form of unlabeled data

Training data (GTA)

Test data (real world)

The dream: generalization to unknown test distributions

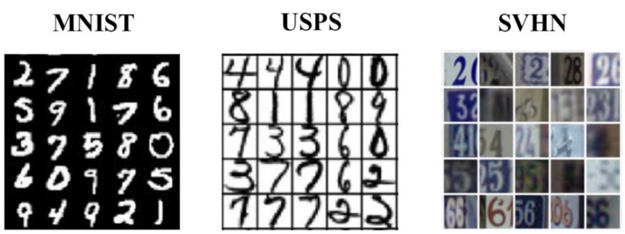
Humanlike robustness: more general, doesn't need specific target domain data

Input: a diverse range of input examples (possibly from many environments)

Test distributions: a range of related, but not identical tasks

Domain generalization examples

- **Zero shot / transfer :** Imagenet to Imagenet-sketch
- **Causal:** Generalizing to an intervention (e.g. deleting a gene from an organism)
- **Multi-environment:** We observe multiple domains and generalize to a new one



- **Known family of targets**: – adversarial examples

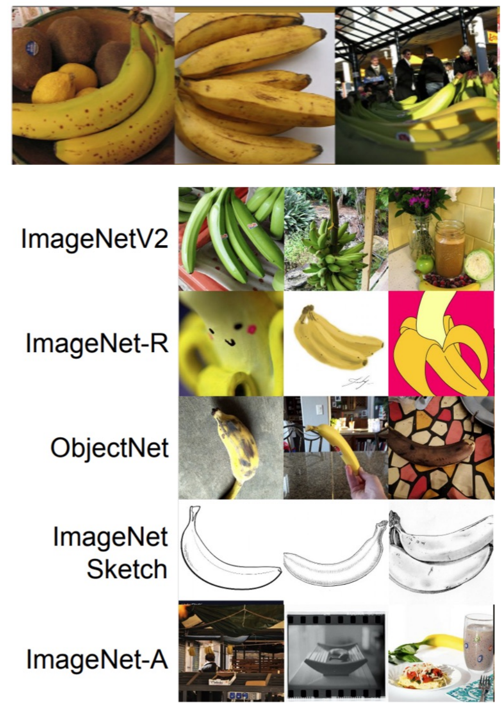
Shared in all these cases: no explicit data from the target

Focus today: zero-shot generalization

Zero shot generalization:

Training: train on some i.i.d data from p_{train} (e.g. Imagenet)

Test: generalize to 'reasonable' tasks in the same modality



What is 'reasonable'? Who knows!

Representation Learning

Learning transformations of the data that make it easier to extract useful information for performing a wide range of downstream tasks

In deep learning, usually:

→representation = last layer before classifier

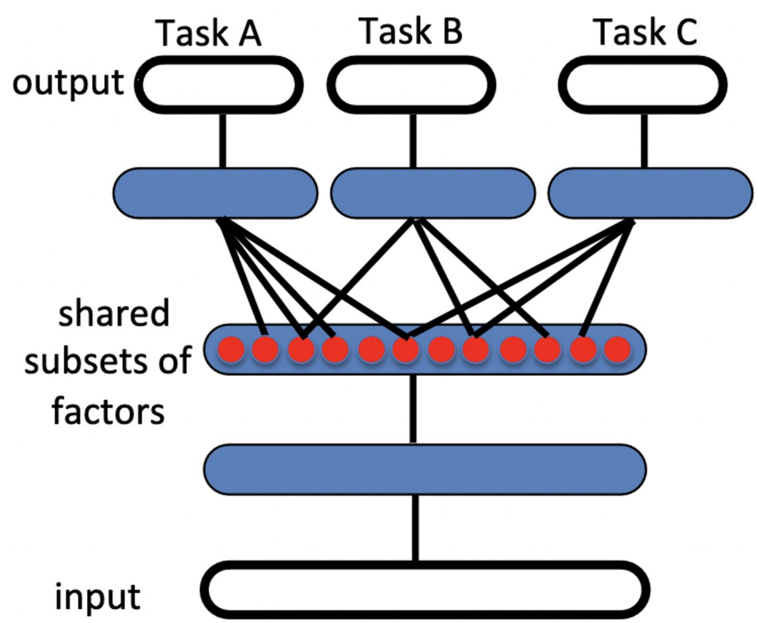
Desirable traits:

Compression

Distributed

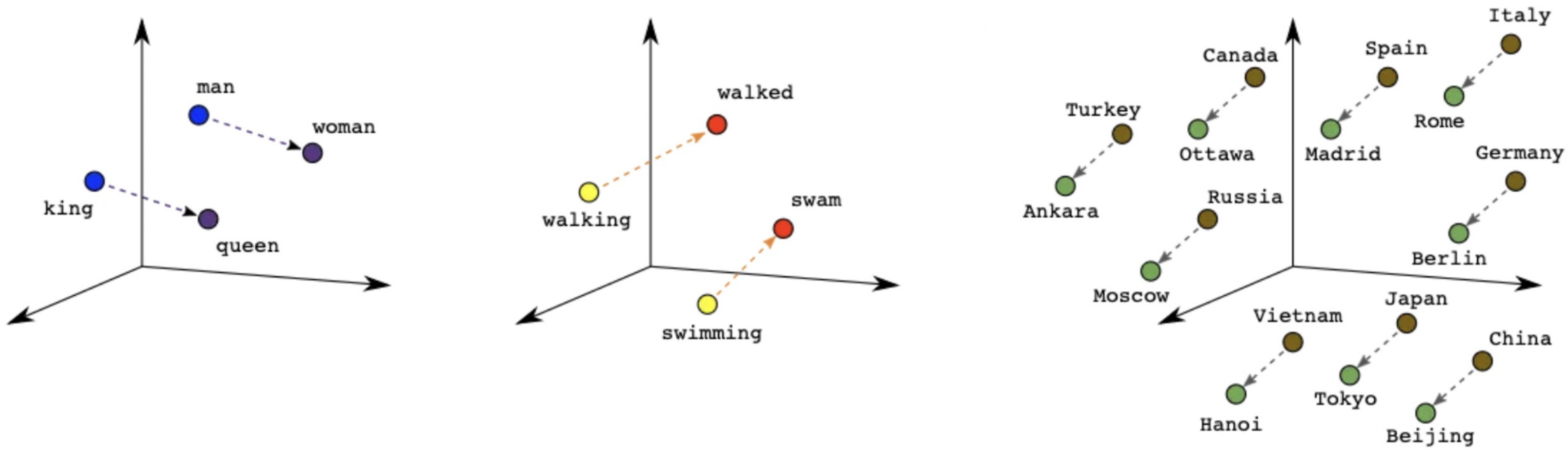
Clustered

Invariant



Bengio et al. (2013)

Representation Learning



Male-Female Verb Tense Country-Capital

Image Source: (Embeddings: Translating to a Lower-Dimensional Space) by Google.

Learning Robust Representations

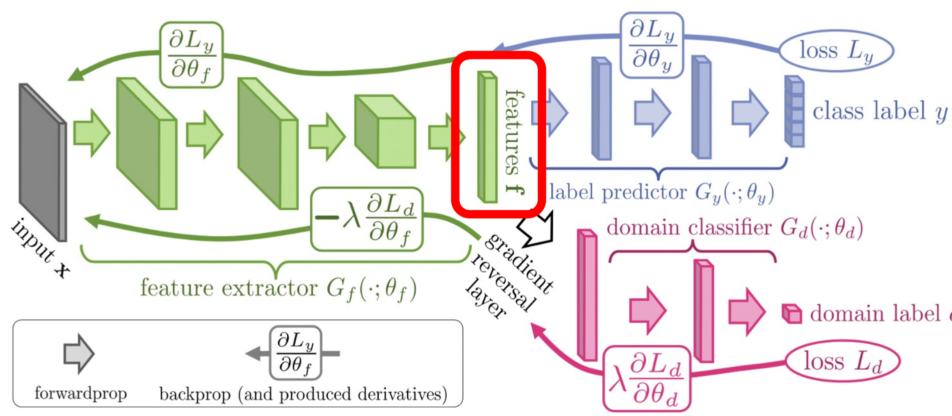
Domain Adversarial Neural Networks

Goal: P(y|f, x~X_{source}) = P(y|f, x~X_{test})

Knowledge of domain does not give information about label ⟺ same optimal classifier

(A) **Cow: 0.99**, Pasture: 0.99, Grass: 0.99, No Person: 0.98, Mammal: 0.98

(B) No Person: 0.99, Water: 0.98, Beach: 0.97, Outdoors: 0.97, Seashore: 0.97



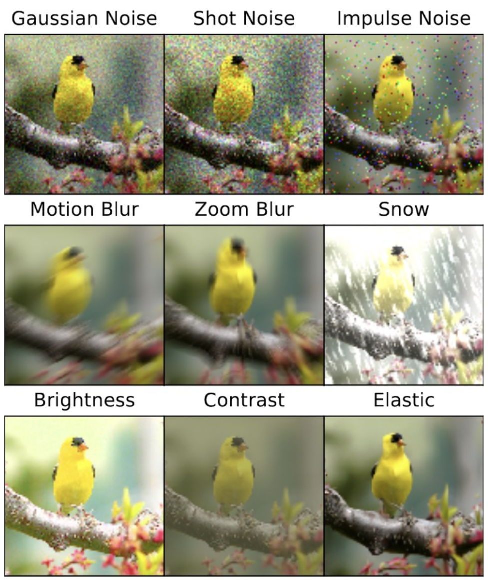
Beery 2018

Ganin 2015

Evaluating Generalization

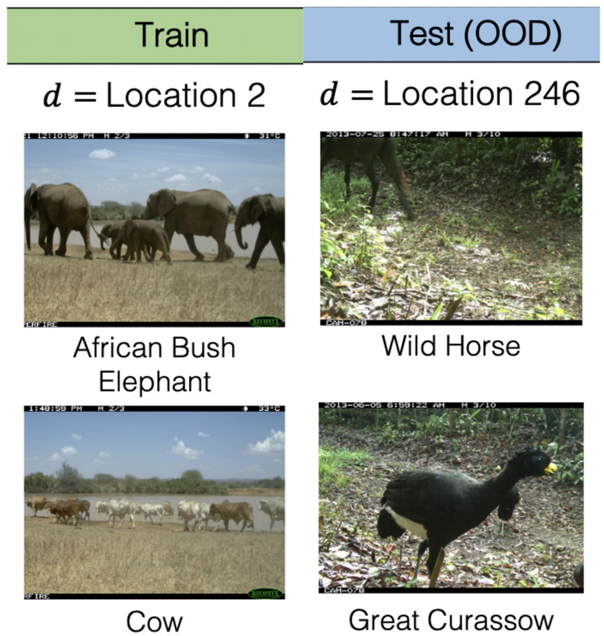
There are different types of distribution shifts that we can face in deployment, including:

Synthetic



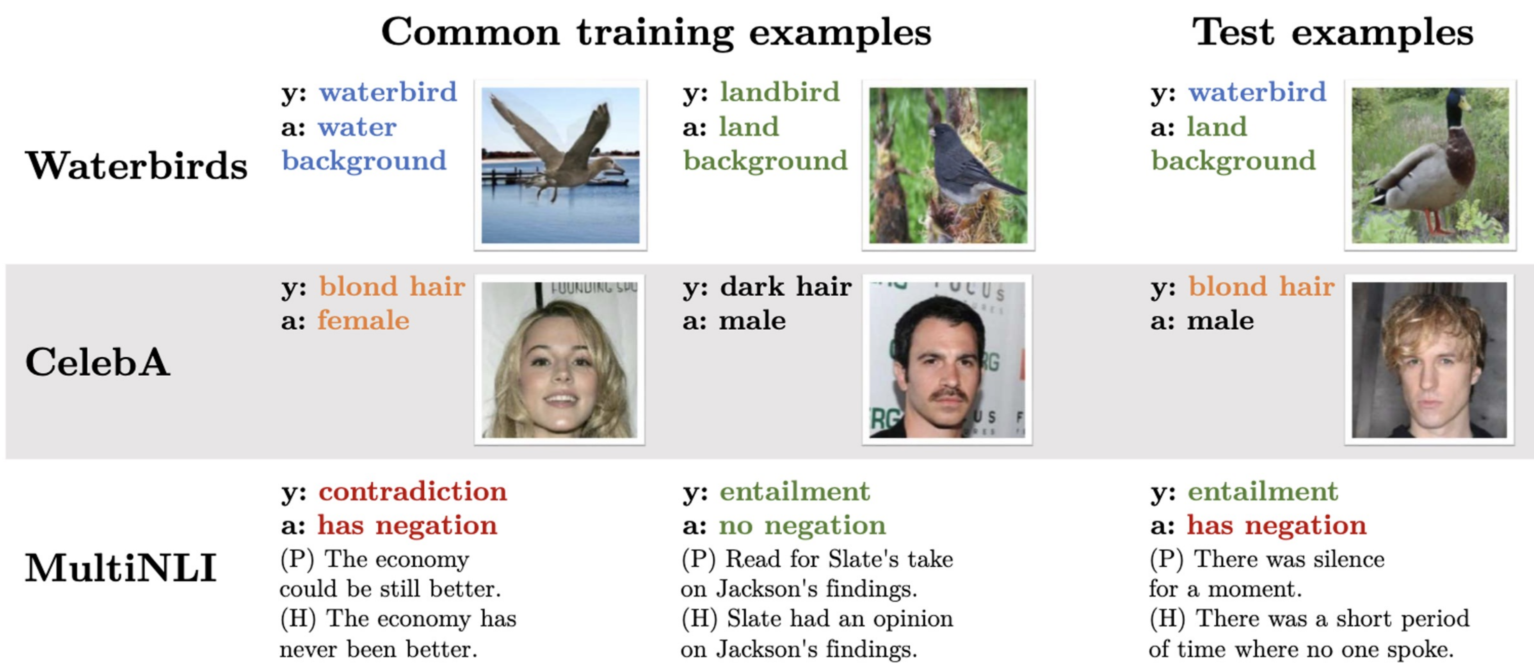
ImageNet-C

Natural



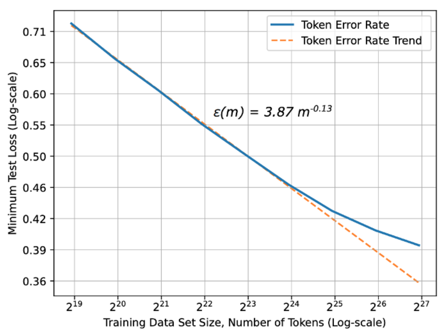
Adversarial

Robustness to Spurious Correlations

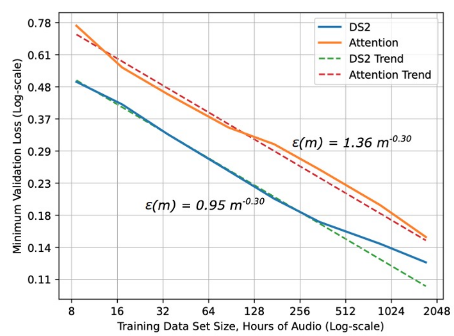


Sagawa 2020

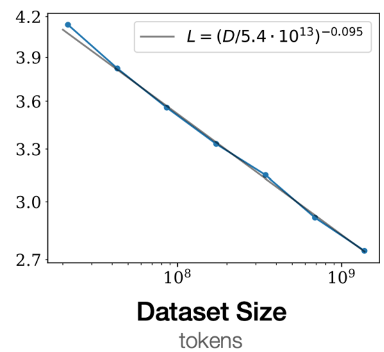
Don’t we already know more data helps?



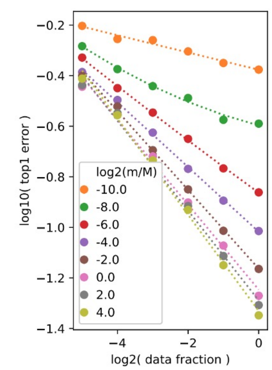
Machine translation



Speech



Language modeling



Object recognition

Hestness et al 2017.

Kaplan et al 2020.

Rosenfeld 2020.

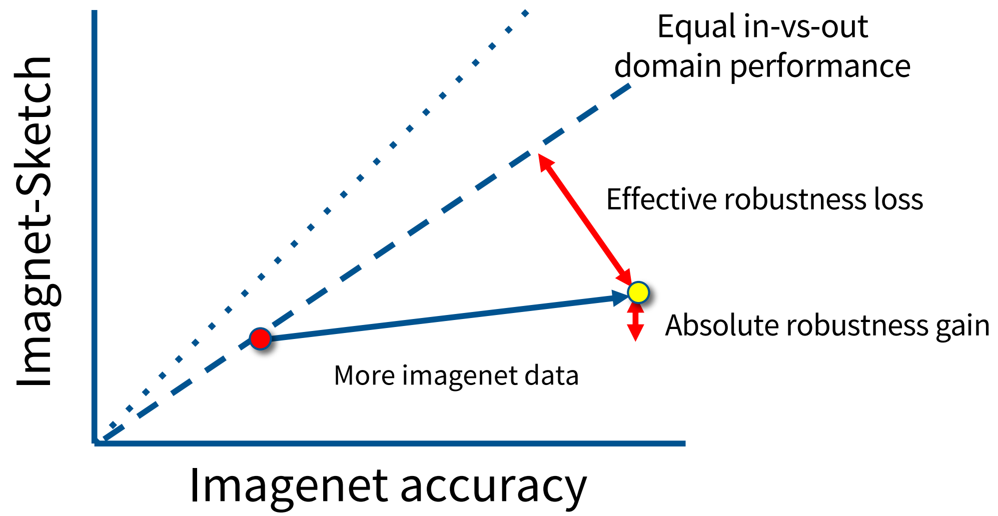
More data always helps! But are we really gaining “robustness”

Analyzing absolute vs effective robustness

Absolute: OOD performance

Effective: OOD performance beyond what can be predicted by ID performance

Relative: OOD performance gained by applying robustness intervention



-Adding data may increase *absolute robustness* but decrease *effective robustness*

-Robustness intervention may increase *effective robustness* but decrease *absolute robustness*

Arguments for studying effective and relative robustness

In this lecture we will study relative / effective robustness

Why study absolute robustness?

Why study effective and relative robustness?

Arguments for studying effective and relative robustness

In this lecture we will study relative / effective robustness

Why study absolute robustness?

- This is what we care about (performance out of domain)

Why study effective and relative robustness?

- Decouple robustness from general performance research (just combine them!)
- Helps identify promising directions to push on
- Differential treatment (fairness)

In many cases: effective and relative robustness isolate effects of robustness interventions and build intuition to improve absolute robustness

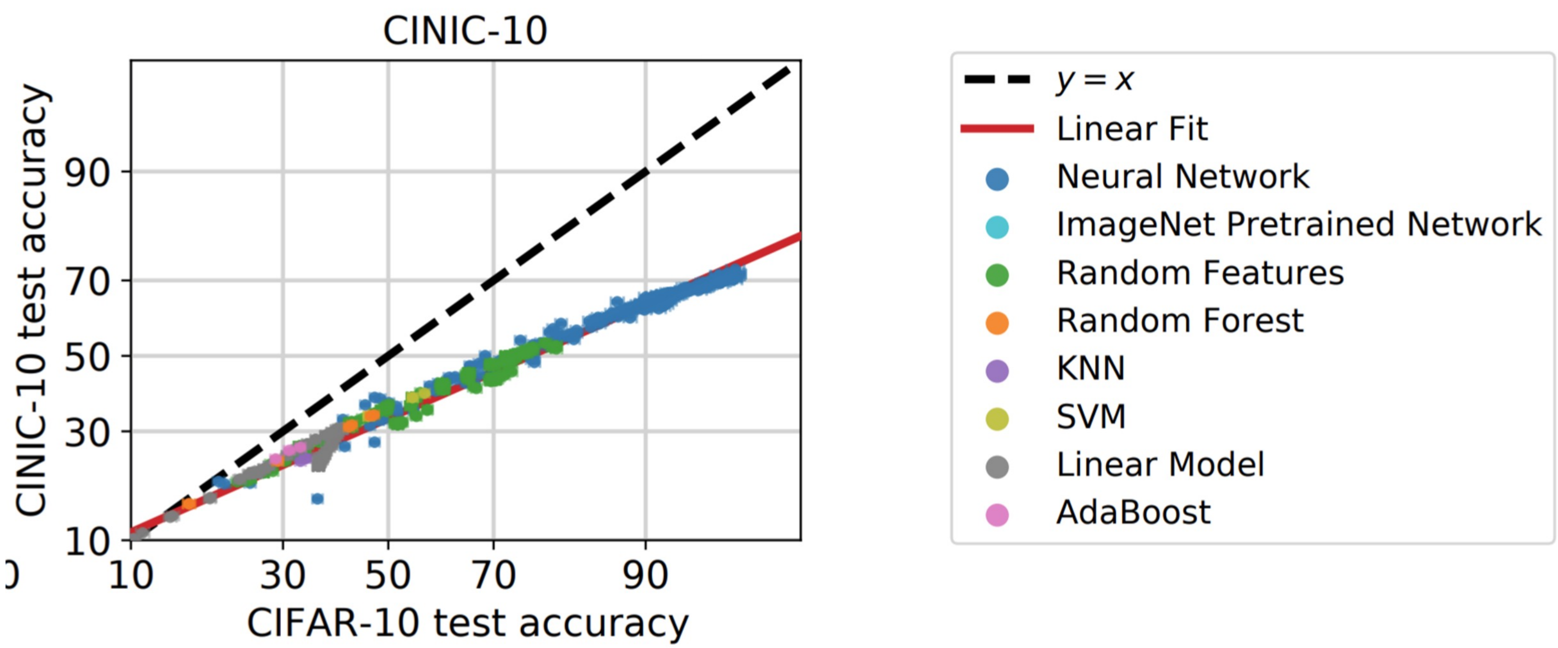
Quick poll

Which of these models has the highest effective robustness?

1. Neural nets + pretraining
2. Neural nets
3. Random forest
4. Linear models
5. No differences in effective robustness

Existing high level observations about relative robustness

Answer: no real difference.

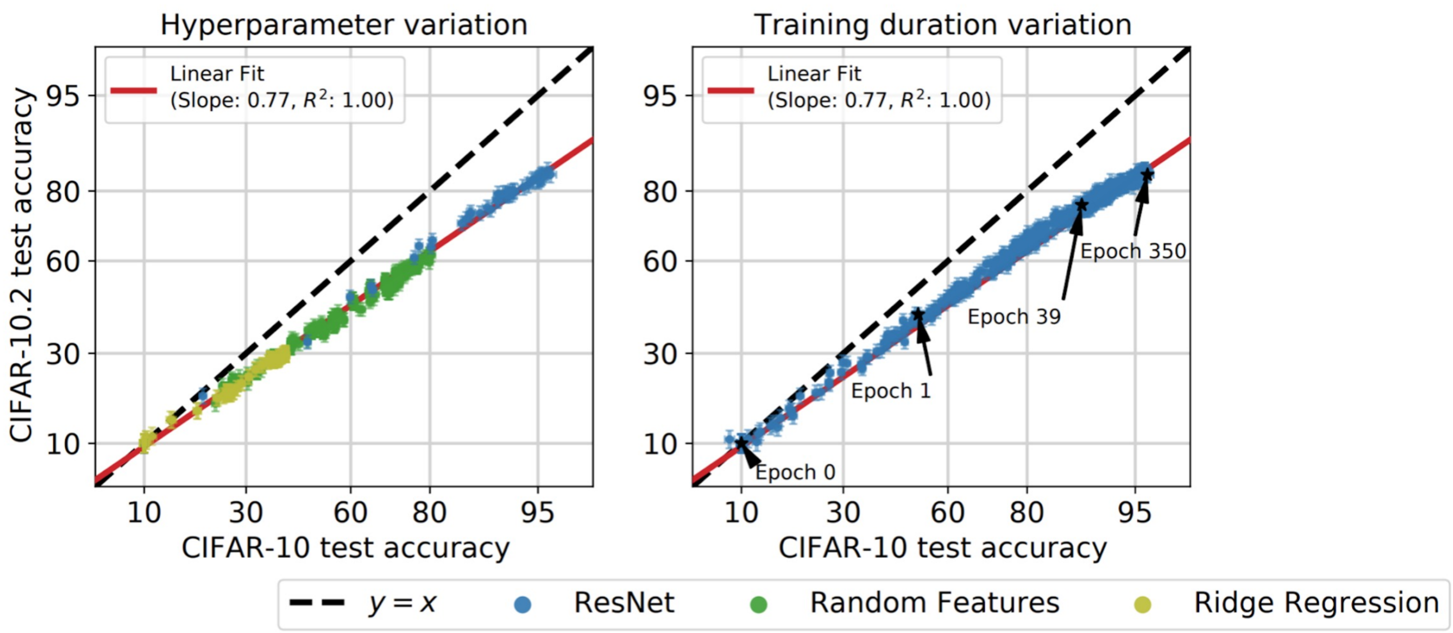


What we see: most progress has been on in-domain accuracy!

[Accuracy On The Line, Miller+ 2020]

Building some intuition about effective robustness

Effective robustness trends hold across different hyperparams, training iterations



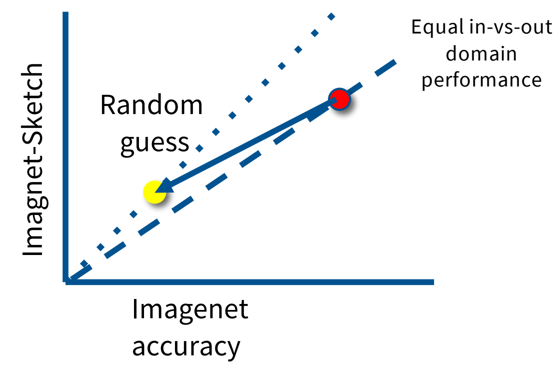
Some caveats with effective and relative robustness

Before we dive in…

- Not all relative robustness gains lead to *absolute* robustness gains.

  Examples: adversarial robustness, zero-shot learning

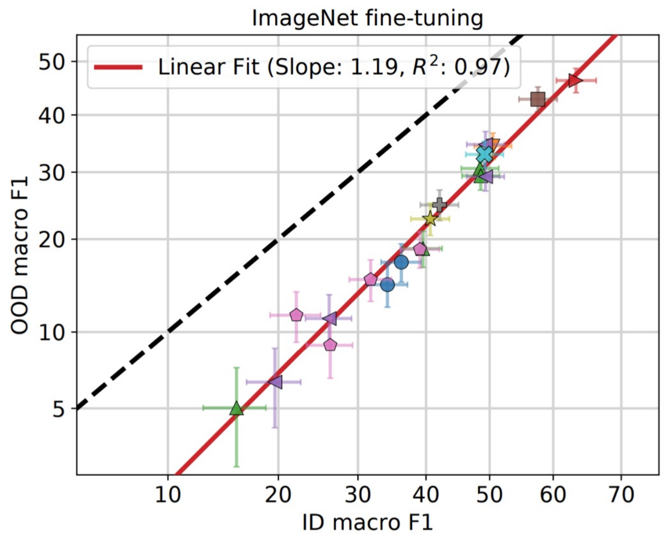
- Baselines are difficult to assess – random interpolation can give robustness gains!



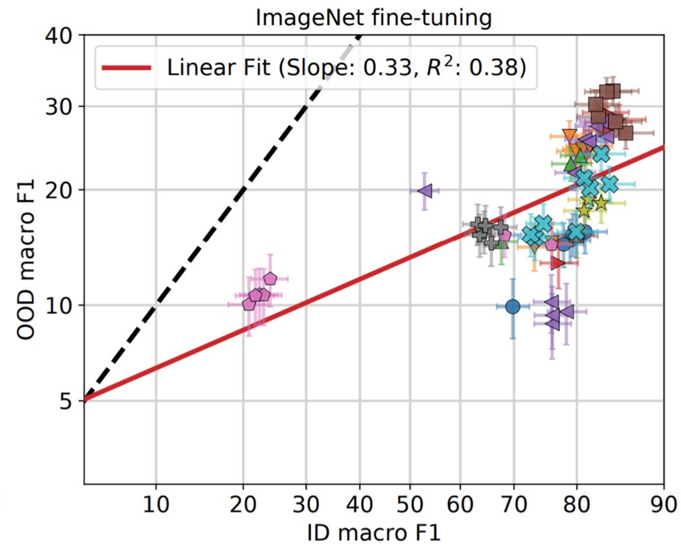
Goal (Roughly): Get on a better effective robustness trend with reasonable interventions, then higher ID accuracy will lift all boats

Also, not all datasets cleanly fit the line

We'll mostly cover cases where the fit is good, but that's not always the case..



Iwildcam 2.0



Iwildcam 1.0

An overview of different robustness phenomena

Does… help?

- Different model architectures?
- More data? Better data?
- Adversarial robustness?
- Pre-training?
- Zero-shot learning?

Model architectures: the premise

Is the latest and greatest image classifier more robust than AlexNet?
(Current iteration of this is visual transformers)

Vision Transformers are Robust Learners

Sayak Paul*
PyImageSearch
s.paul@pyimagesearch.com

Pin-Yu Chen*
IBM Research
pin-yu.chen@ibm.com

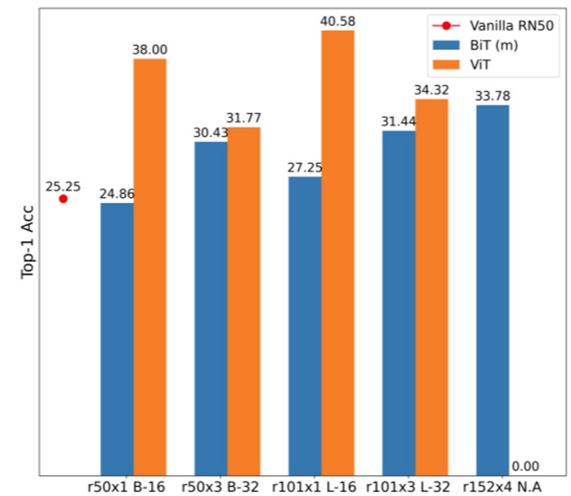


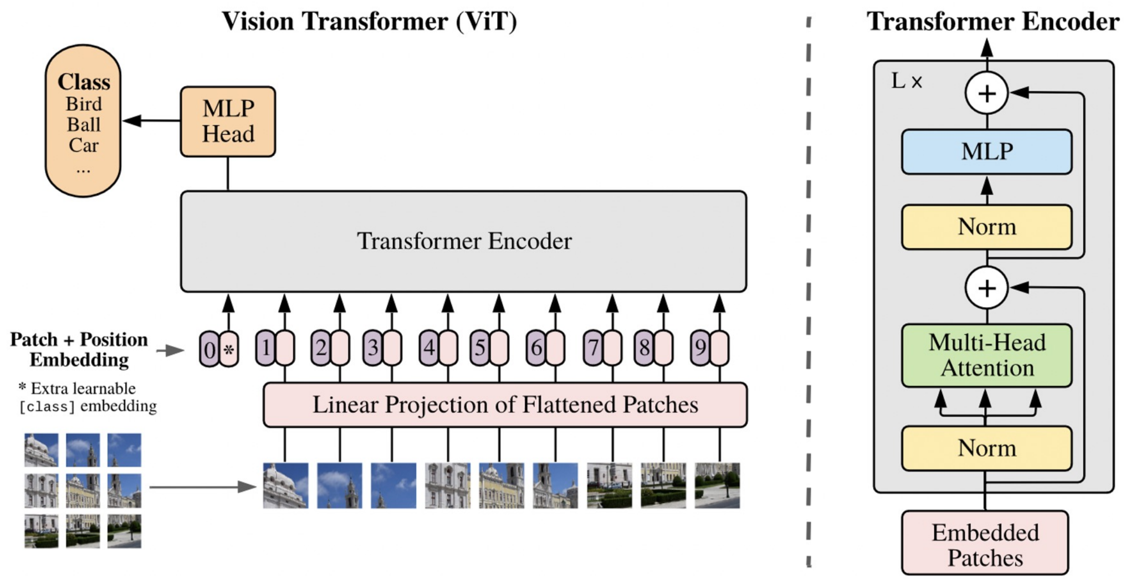
Figure 3: Top-1 accuracy scores (%) on ImageNet-R dataset [14].

Vision Transformers

-Split image into patches, flatten, project

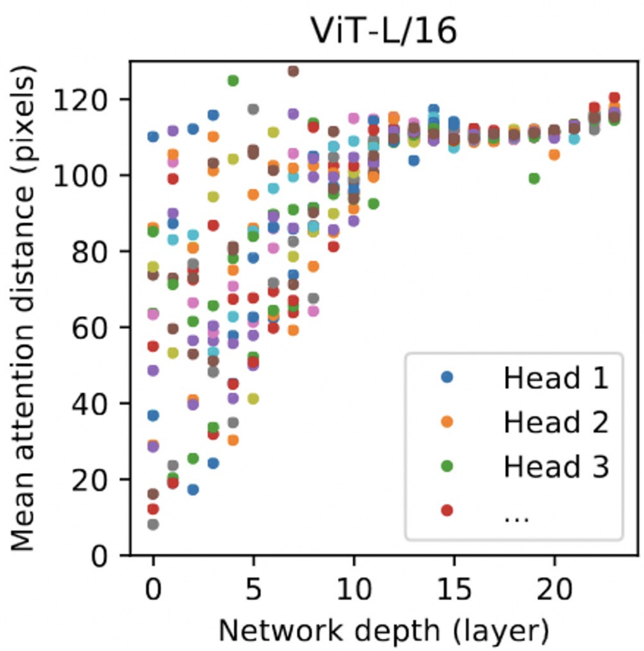
-Encode with transformers

→just like text/BERT



Vision Transformers

Hypothesis: CNN's use local context; ViT uses global context, so more robust



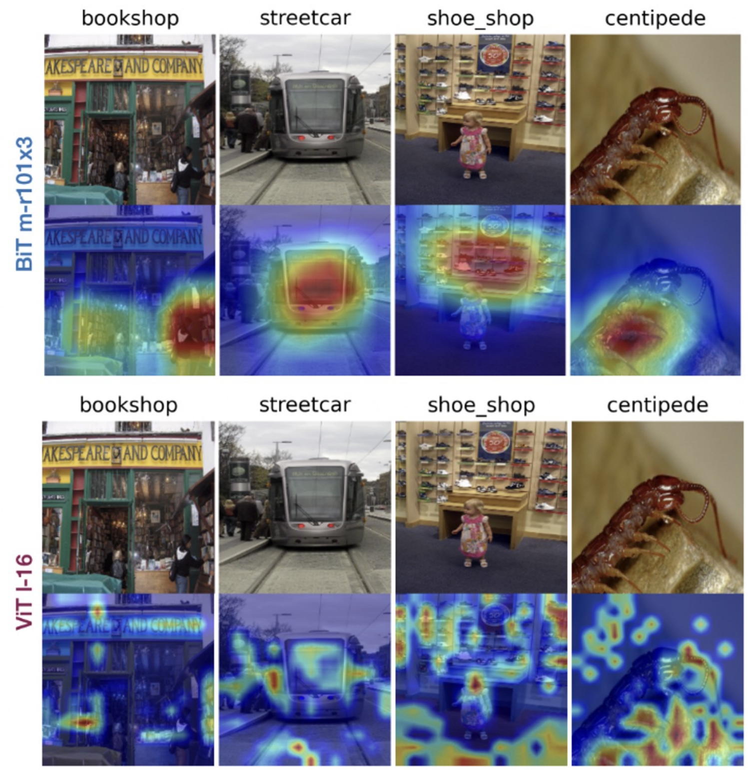
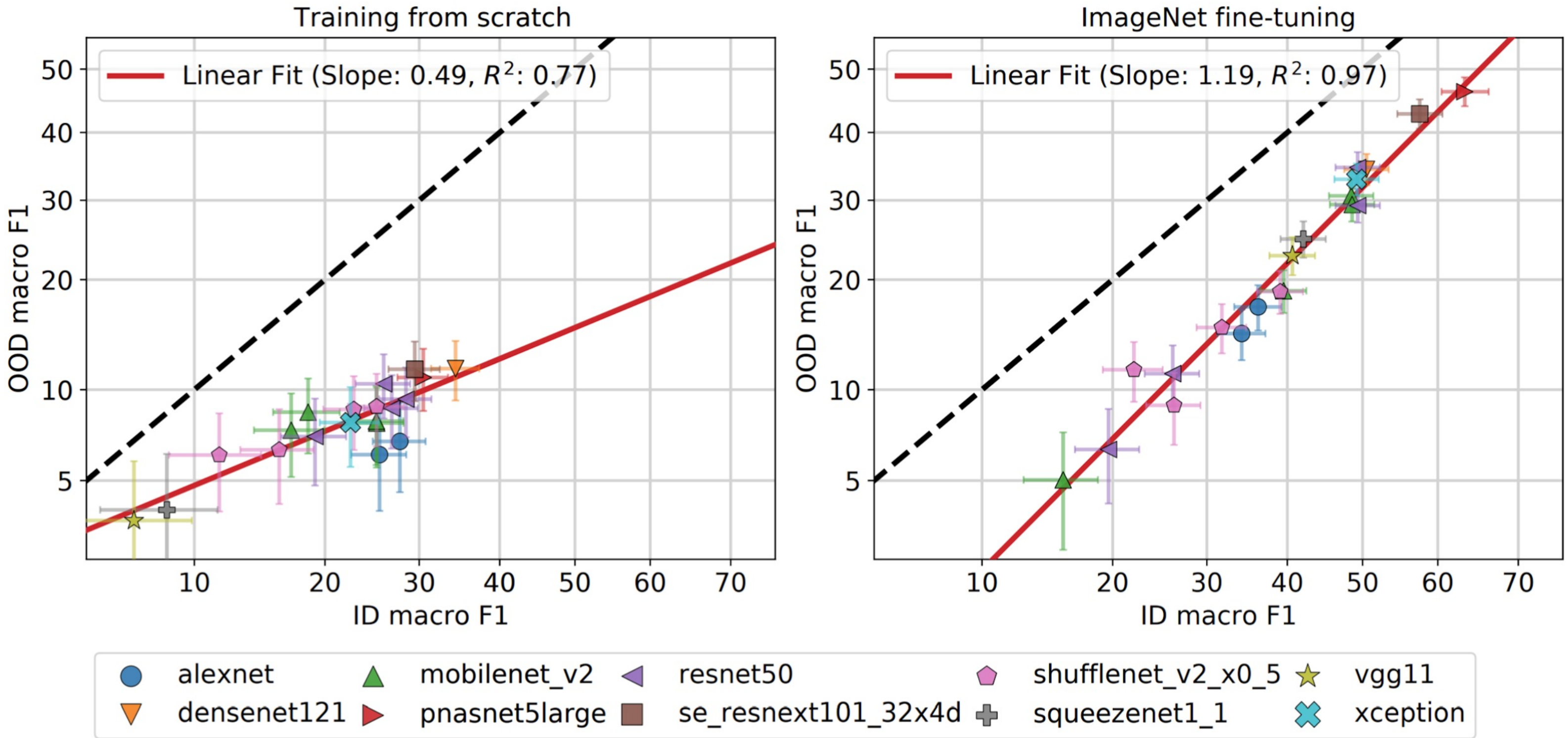


Figure 11: Grad-CAM results for the images where both BiT and ViT give correct predictions.

It’s hard to get off the effective robustness line

Answer: No – example from iWildCam-WILDS from scratch (left) or pretrained (right)

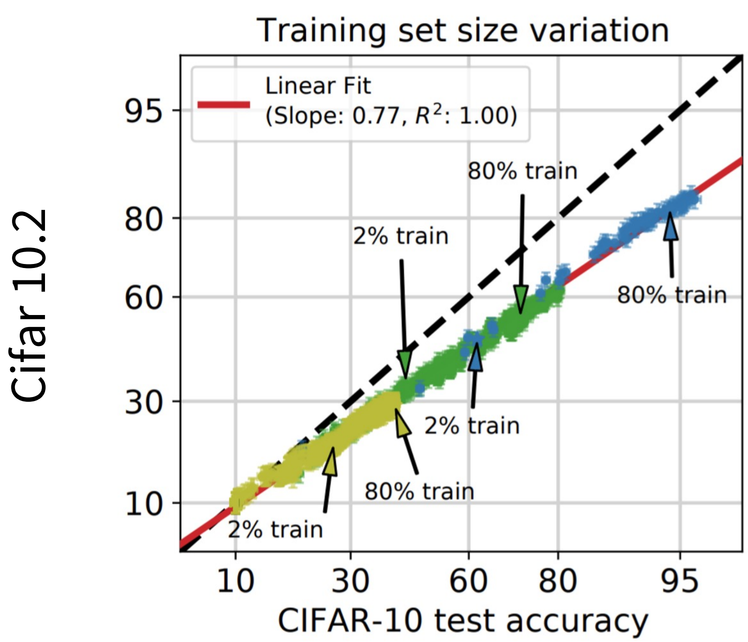


ViT included in Shi 2023 follow-up study

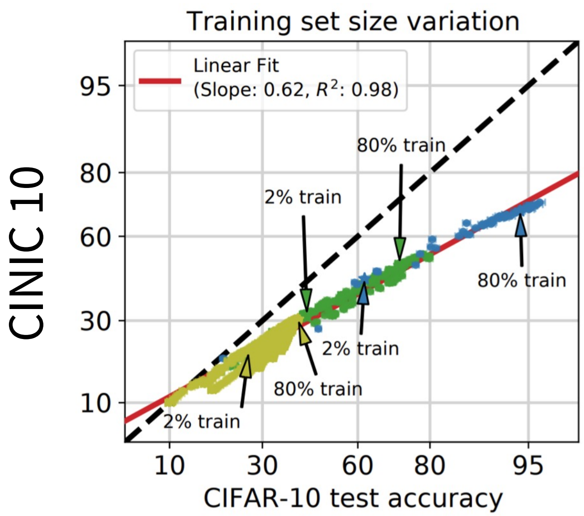
Does more data help?

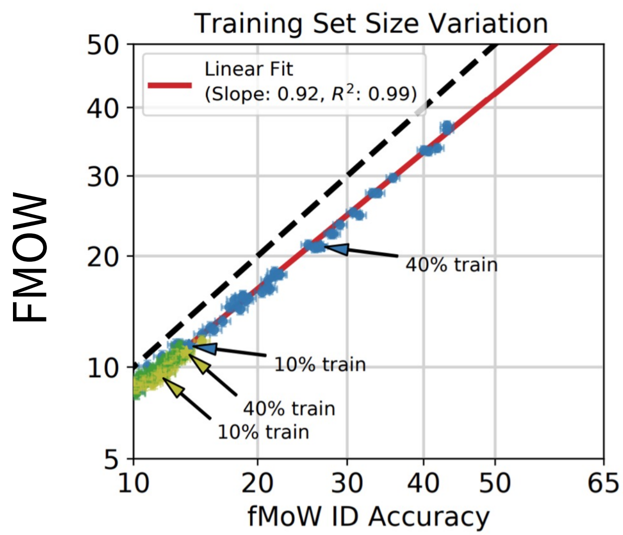
Obviously more data helps for absolute robustness

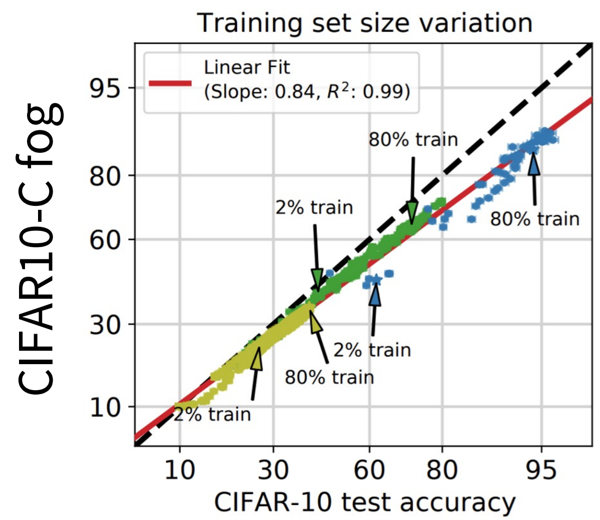
Does getting data help for effective robustness?



Collecting data that's i.i.d doesn't help



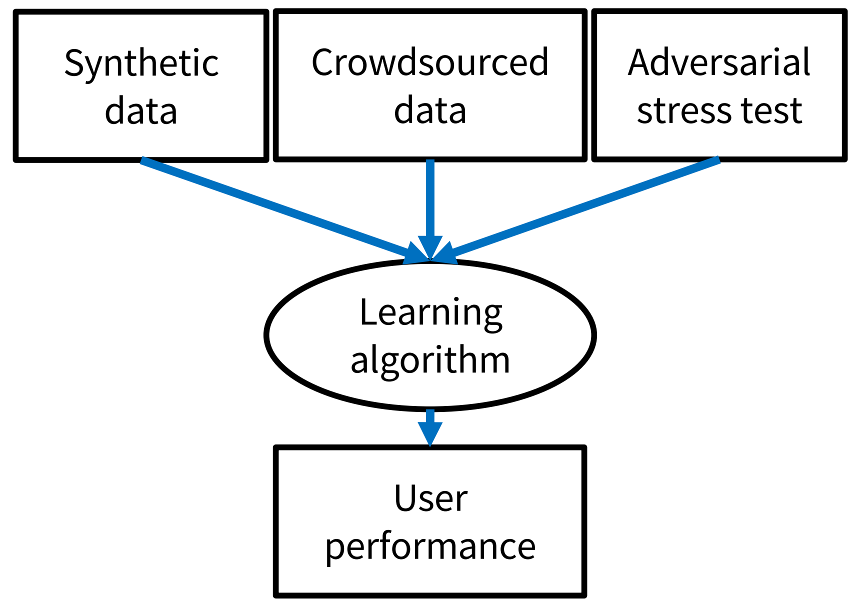




Conclusion: more in-domain data does not improve effective robustness

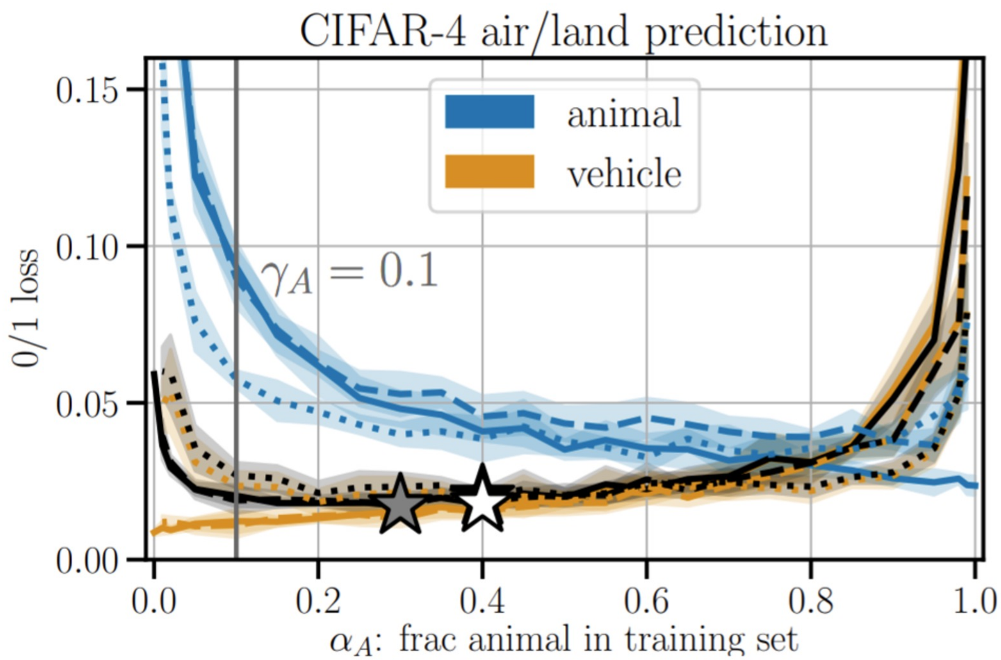
Quantity doesn't help. Does quality?

- In practice we may have more than one data source
- Maybe we can mix up multiple sources of data to build a more robust model
- How does data composition (p) and size (n) affect performance?



Optimizing data collection mixtures

Picking the right 'mix' of data sources can lead to substantial improvements.



[Rolf+ 2021]

Takeaways: If we want similar performance across groups, not having any animals/vehicles = catastrophic. Want > 50% animals.

Using better data gives robustness gains

Using scaling laws to predict 'optimal' data collection can improve robustness

Task: predicting book review ratings from good reads
Train vs test: history vs fantasy proportions

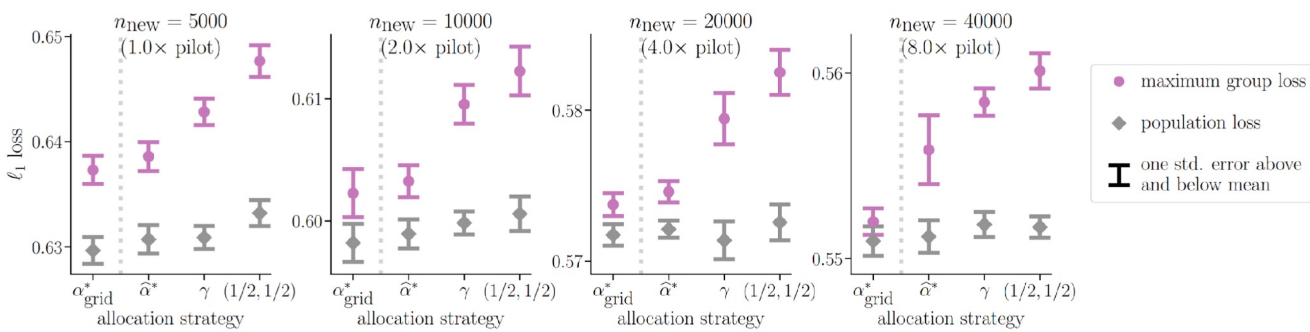
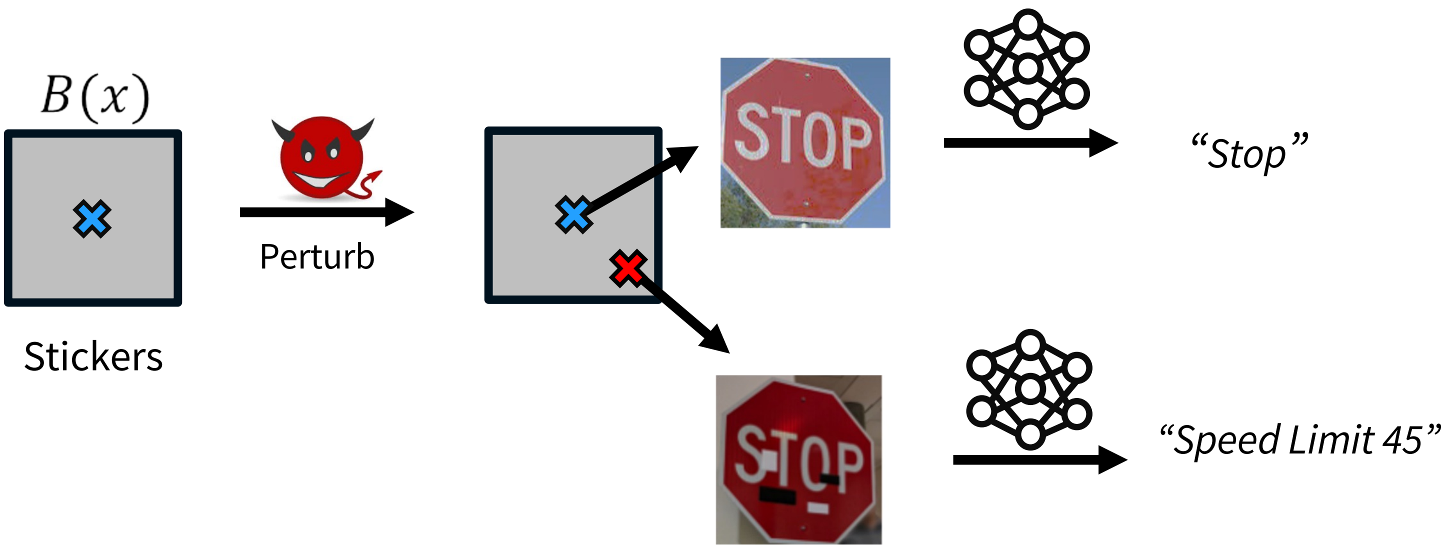


Figure 2: Pilot sample experiment. Panels show the result of the three allocations $\vec{\alpha} \in [\hat{\alpha}^*_{\text{minmax}}, \vec{\gamma}, (1/2, 1/2)]$ for different sizes of the new training sets compared with an α^*_{grid} baseline that minimizes the maximum group loss over a grid of resolution 0.01, averaged over the random trials. Purple circles indicate average maximum error over groups and grey diamonds indicate average population error. Ranges denote standard errors taken over the 10 trials.

[Rolf+ 2021]

Does adversarial robustness help?

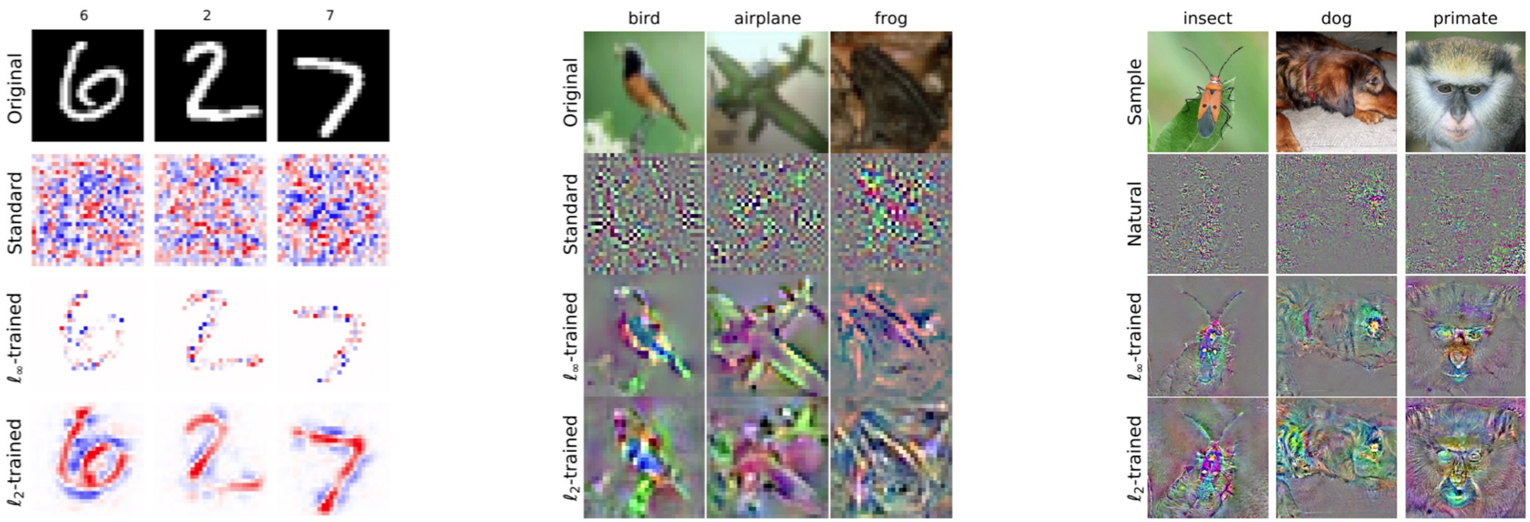
One major class of robustness interventions:
Adversarial robustness to perturbations



[Eykholt+ 2018]

Why might adversarial examples help?

Adversarially robust models have more 'humanlike' loss gradients



(a) MNIST (b) CIFAR-10 (c) Restricted ImageNet

(Shown: gradients of examples taken with respect to input)

[Tsipras+ 2019]

How does adversarial robustness affect performance?

On adversarial attacks: dramatic (50%!) error decrease
On standard error: *decrease* in performance of 3x.

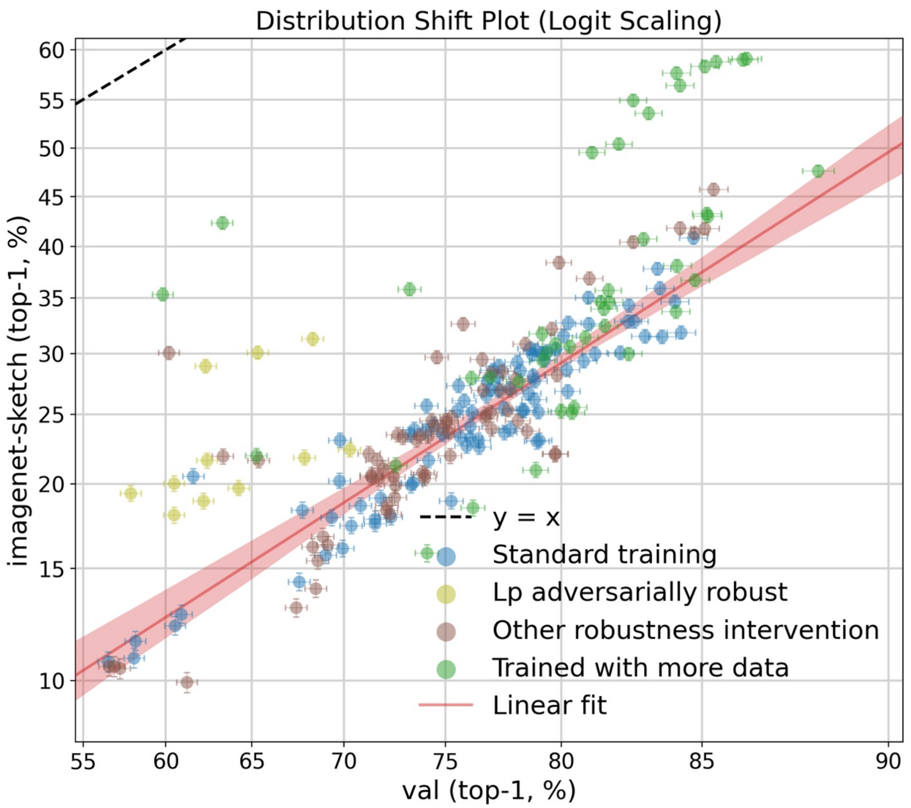
Model	Robust error	Standard error
Standard training	100	4
Adversarial training [Madry et al. 2018]	56	13
TRADES [Zhang et al. 2019]	47	15
Adv training ++ [Rice et al. 2020]	46	15
Fast adv training [Zhang et al. 2019]	55	15
MART[Wang et al. 2019]	45	17

Relative robustness gains

This leads to substantial effective robustness gains

- Drop in standard accuracy shifts points to the left
- Increase in robust accuracy shift points off the line

Adversarial examples improve effective (but not absolute) robustness.



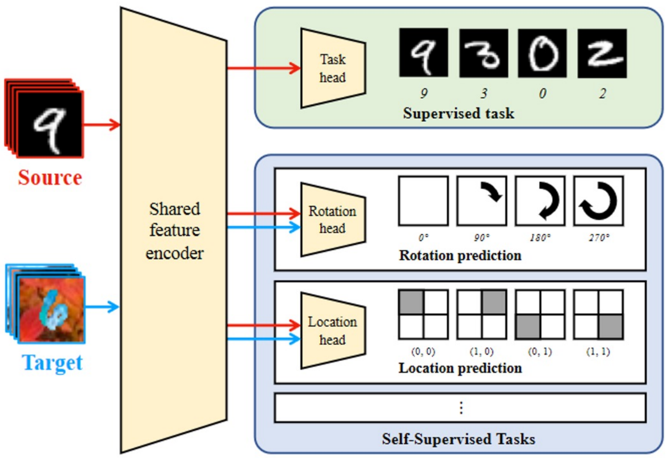
Recap So Far…

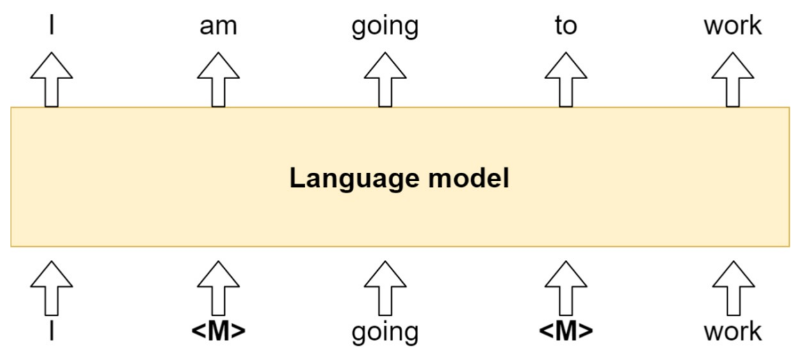
Q: Does …. help with effective robustness?

- Model architectures: **Not really** (even neural vs not neural)
- Data: **Not for i.i.d** , **a little for non-i.i.d. (i.e. smart collection strategies)**
- Adversarial robustness: **Yes, but at a great cost**

Does pre-training help?

We know that self-supervision with unlabeled target data can help (UDA-SS, TAPT etc)

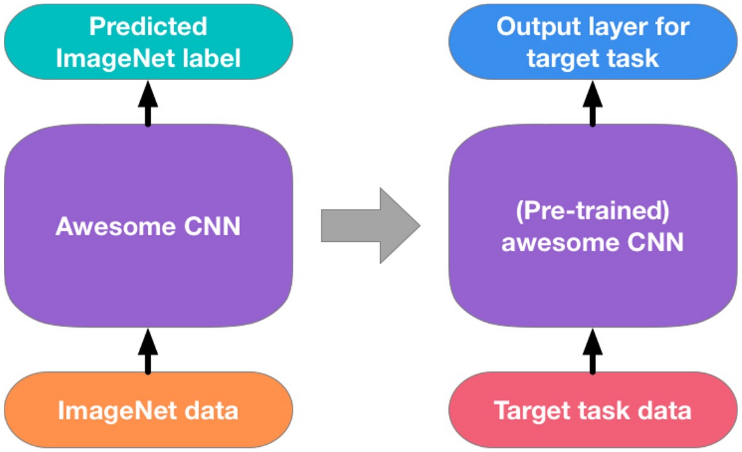




Can this help even without target domain data?

Pre-training

Imagenet pre-training is one of the basic building blocks of modern image classifiers.



For robustness, we know it can improve several things..

- Adversarial robustness
- Resistance to label noise
- Performance to label shift

Let's look at each of these in turn..

Robustness to adversaries

Adversarial robustness against (weak) adversaries improve.

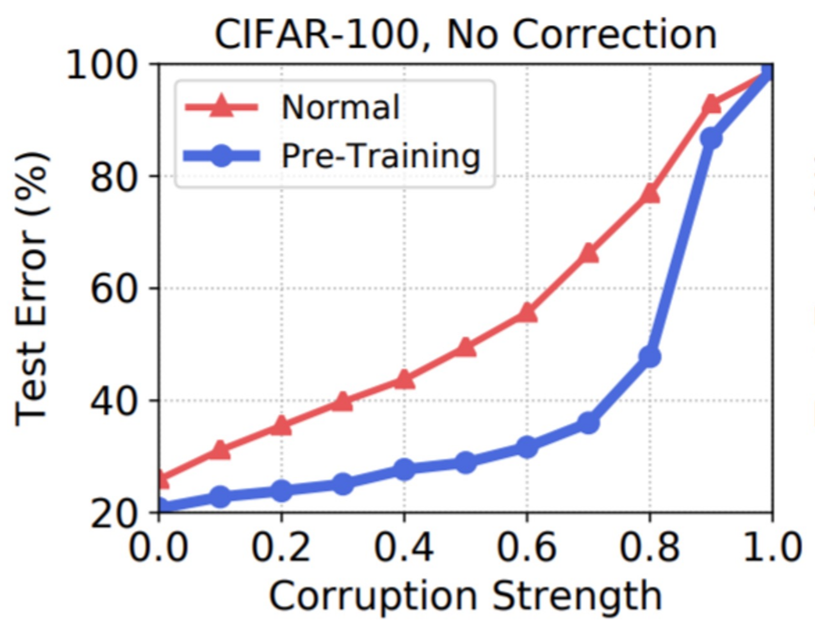
Table 1. Adversarial accuracies of models trained from scratch, with adversarial training, and with adversarial training with pre-training. All values are percentages. The pre-trained models have comparable clean accuracy to adversarially trained models from scratch, as implied by He et al. (2018), but pre-training can markedly improve adversarial accuracy.

	CIFAR-10		CIFAR-100	
	Clean	Adversarial	Clean	Adversarial
Normal Training	96.0	0.0	81.0	0.0
Adversarial Training	87.3	45.8	59.1	24.3
Adv. Pre-Training and Tuning	87.1	57.4	59.2	33.5

Hendrycks 2019

Improvements in performance under label noise

As label noise increases: both normal and pre-training performance degrades, but pre-trained model performance degrades *less*



The increase in red-blue gap is a form of 'effective robustness'

Robustness under label shift

Right to left increases imbalance ratio.

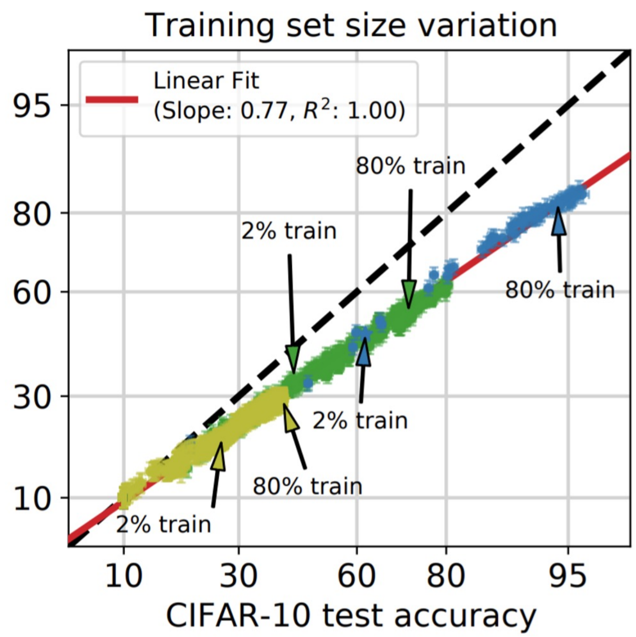
Table 3. Experimental results on the imbalanced CIFAR-10 and CIFAR-100 datasets.

Dataset	Method \ Imbalance Ratio	0.2	0.4	0.6	0.8	1.0	1.5	2.0
		Total Test Error Rate / Minority Test Error Rate (%)						
CIFAR-10	Normal Training	23.7 / 26.0	21.8 / 26.5	21.1 / 25.8	20.3 / 24.7	20.0 / 24.5	18.3 / 23.1	15.8 / 20.2
	Cost Sensitive	22.6 / 24.9	21.8 / 26.2	21.1 / 25.7	20.2 / 24.3	20.2 / 24.6	18.1 / 22.9	16.0 / 20.1
	Oversampling	21.0 / 23.1	19.4 / 23.6	19.0 / 23.2	18.2 / 22.2	18.3 / 22.4	17.3 / 22.2	15.3 / 19.8
	SMOTE	19.7 / 21.7	19.7 / 24.0	19.2 / 23.4	19.2 / 23.4	18.1 / 22.1	17.2 / 22.1	15.7 / 20.4
	Pre-Training	8.0 / 8.8	7.9 / 9.5	7.6 / 9.2	8.0 / 9.7	7.4 / 9.1	7.4 / 9.5	7.2 / 9.4

Relevant comparison is top row (normal) and bottom row (pre-trained)

Does pre-training help relative robustness?

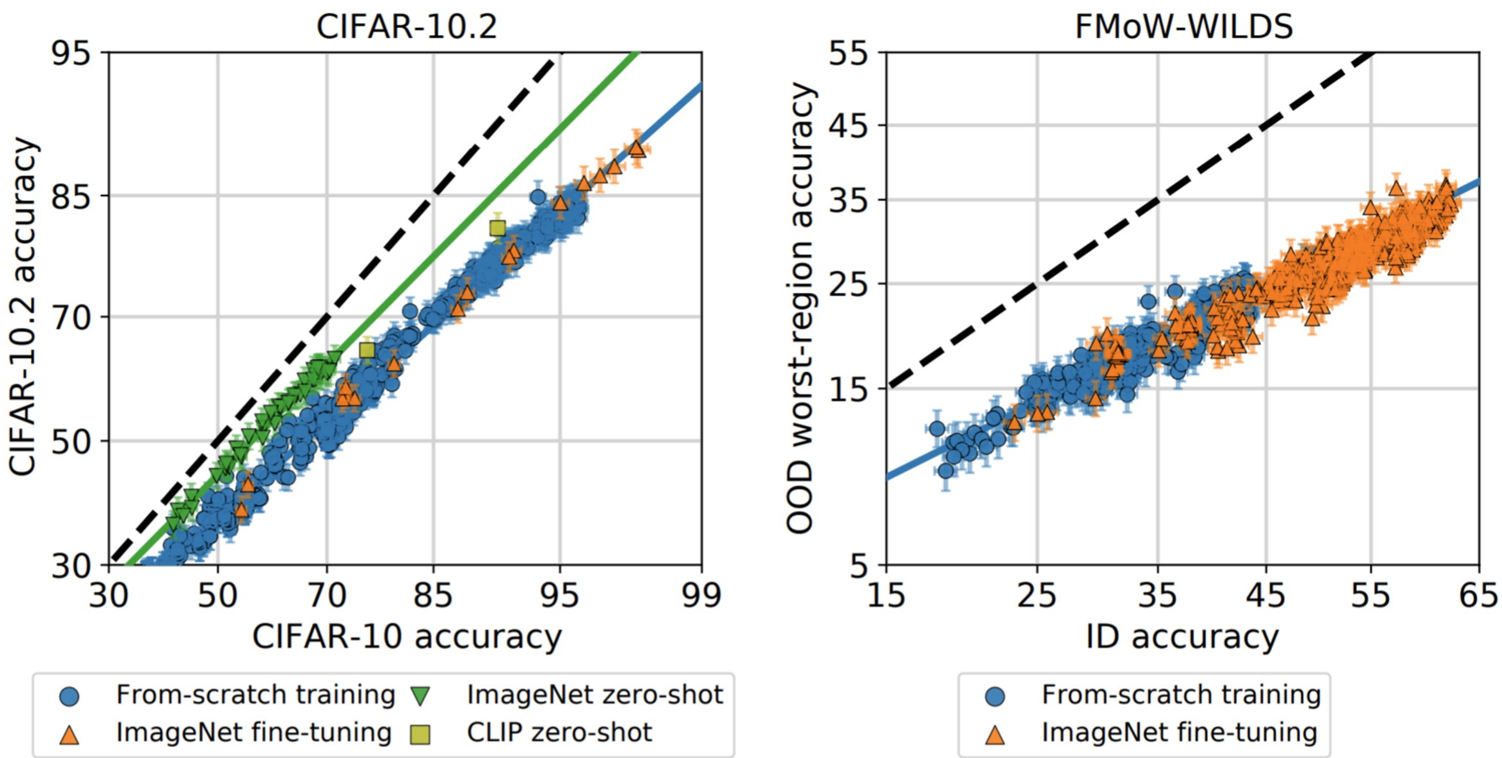
Pre-training seems great, but is this all absolute robustness?



Is this just like getting more data, or are we getting ‘effective robustness’?

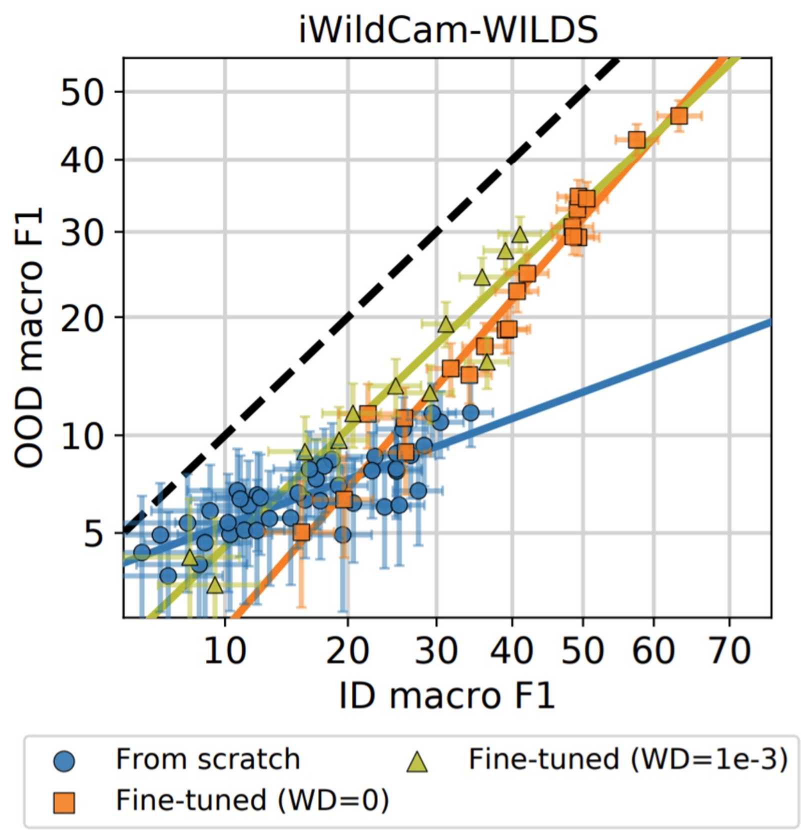
Pre-trained models and effective robustness

Of course, not all pre-training is complex. Fine-tuning alone sometimes isn't enough.



Sometimes this can help

But for some datasets, fine-tuning can have fairly dramatic effects



Roadmap

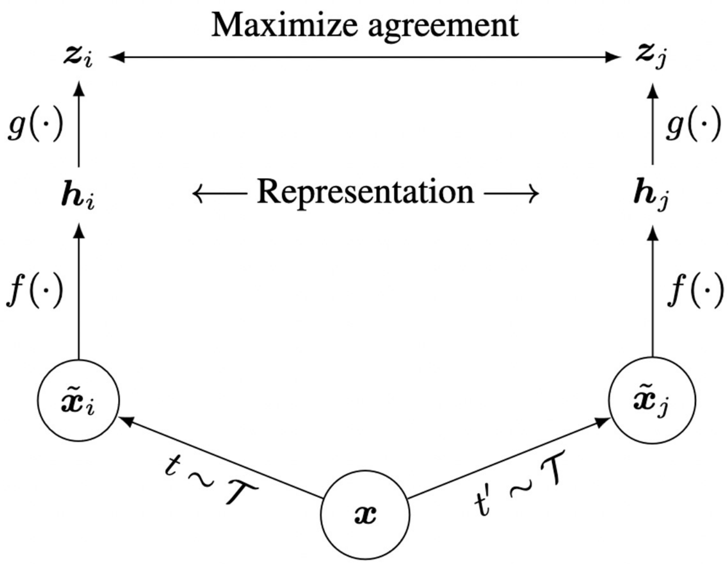
- Intro to Generalization
- Representation Learning
- Evaluating Generalization
- Measuring Robustness
  - Absolute, effective, and relative robustness
- Robustness Interventions
  - Model architectures, more/better data, adversarial robustness, pre-training
  - **self-supervised learning**
- **Zero-shot Learning**
  - **Motivation**
  - **CLIP**
  - **NLP (through ChatGPT)**

Self-Supervised Vision Learning

Take a (massive) unlabeled dataset and create a supervised learning problem

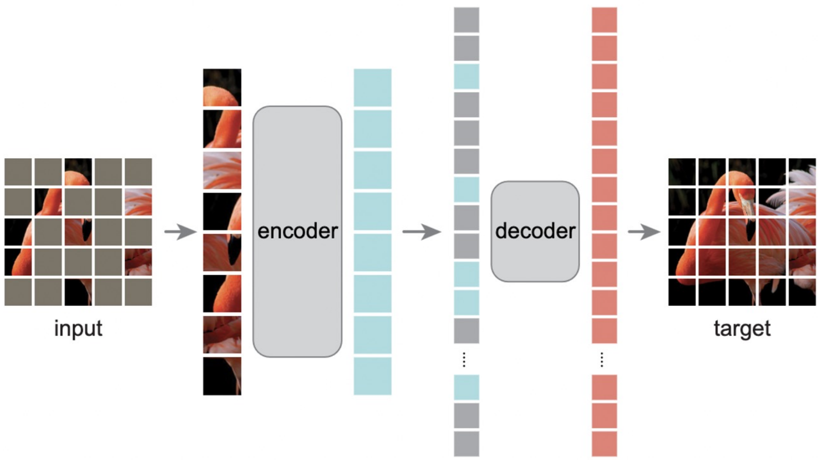
SimCLR

Contrastive learning - predict whether views are derived from same image



VIT-MAE

Masked auto encoder - predict missing pixels



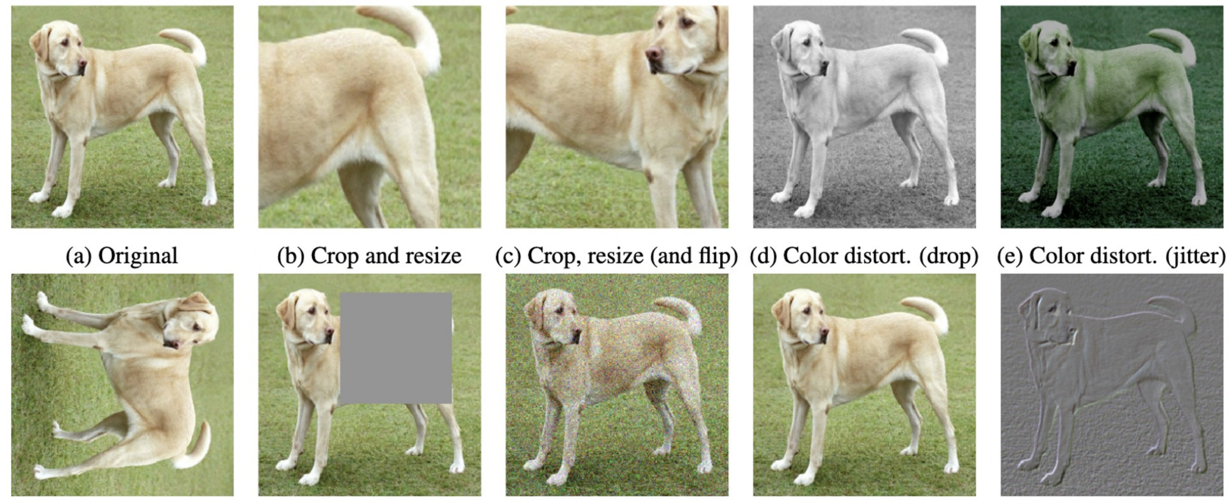
Self-Supervised Learning - SimCLR

1. For each image in a batch, create positive example from augmented view

1. Treat all other images in the batch as negative examples

1. Calculate contrastive loss

1. Backpropogate, repeat, etc., etc.



(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

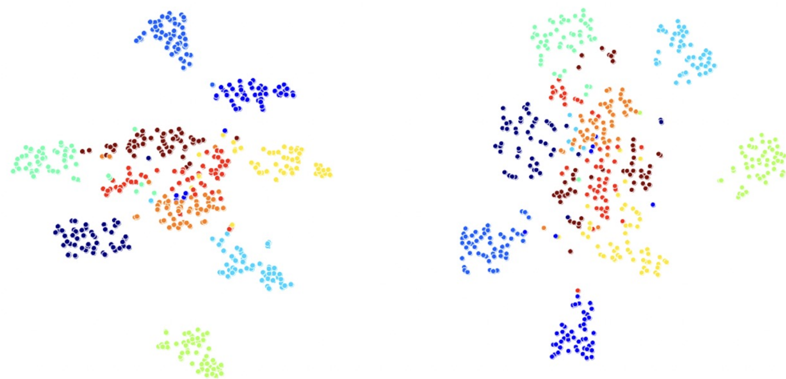
(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

Name	Negative loss function	Gradient w.r.t. $\boldsymbol{u}$
NT-Xent	$\boldsymbol{u}^T\boldsymbol{v}^+/\tau - \log\sum_{\boldsymbol{v}\in\{\boldsymbol{v}^+,\boldsymbol{v}^-\}}\exp(\boldsymbol{u}^T\boldsymbol{v}/\tau)$	$(1-\frac{\exp(\boldsymbol{u}^T\boldsymbol{v}^+/\tau)}{Z(\boldsymbol{u})})/\tau\boldsymbol{v}^+ - \sum_{\boldsymbol{v}^-}\frac{\exp(\boldsymbol{u}^T\boldsymbol{v}^-/\tau)}{Z(\boldsymbol{u})}/\tau\boldsymbol{v}^-$
NT-Logistic	$\log\sigma(\boldsymbol{u}^T\boldsymbol{v}^+/\tau) + \log\sigma(-\boldsymbol{u}^T\boldsymbol{v}^-/\tau)$	$(\sigma(-\boldsymbol{u}^T\boldsymbol{v}^+/\tau))/\tau\boldsymbol{v}^+ - \sigma(\boldsymbol{u}^T\boldsymbol{v}^-/\tau)/\tau\boldsymbol{v}^-$
Margin Triplet	$-\max(\boldsymbol{u}^T\boldsymbol{v}^- - \boldsymbol{u}^T\boldsymbol{v}^+ + m, 0)$	$\boldsymbol{v}^+ - \boldsymbol{v}^-$ if $\boldsymbol{u}^T\boldsymbol{v}^+ - \boldsymbol{u}^T\boldsymbol{v}^- < m$ else $\boldsymbol{0}$

Table 2. Negative loss functions and their gradients. All input vectors, i.e. $\boldsymbol{u}, \boldsymbol{v}^+, \boldsymbol{v}^-$, are ℓ_2 normalized. NT-Xent is an abbreviation for "Normalized Temperature-scaled Cross Entropy". Different loss functions impose different weightings of positive and negative examples.

Self-Supervised Learning - SimCLR

SimCLR/SSL give well-separated classes without any labels!
→ Avoid (bad) shortcut learning



(a) $\boldsymbol{h}$ (b) $\boldsymbol{z} = g(\boldsymbol{h})$

Figure B.4. t-SNE visualizations of hidden vectors of images from a randomly selected 10 classes in the validation set.

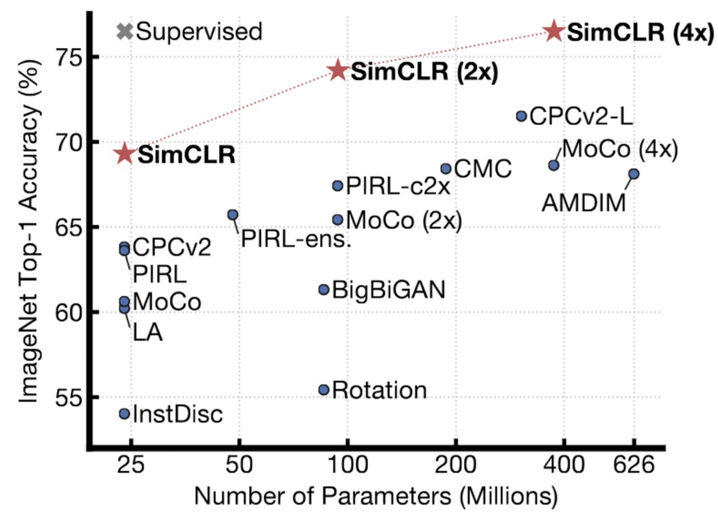


Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

Emerging trends: zero-shot and multitasking

Next up: emerging modern trends in generalization

Common theme: using language as a 'glue' to bridge tasks

- **Zero-shot learning:** We are inherently robust if we don't use any training data
- **Multitasking:** train on so many tasks that we don't pick up biases from any task
  - **Examples:** CLIP, GPT-3 variants

Logic behind few-shot robustness

Q: why do we have better in-domain than out-of-domain accuracy?

Logic behind few-shot robustness

Q: why do we have better in-domain than out-of-domain accuracy?

A: because we learned non-generalizable predictors from in-domain data.

What if we don't use training data..?

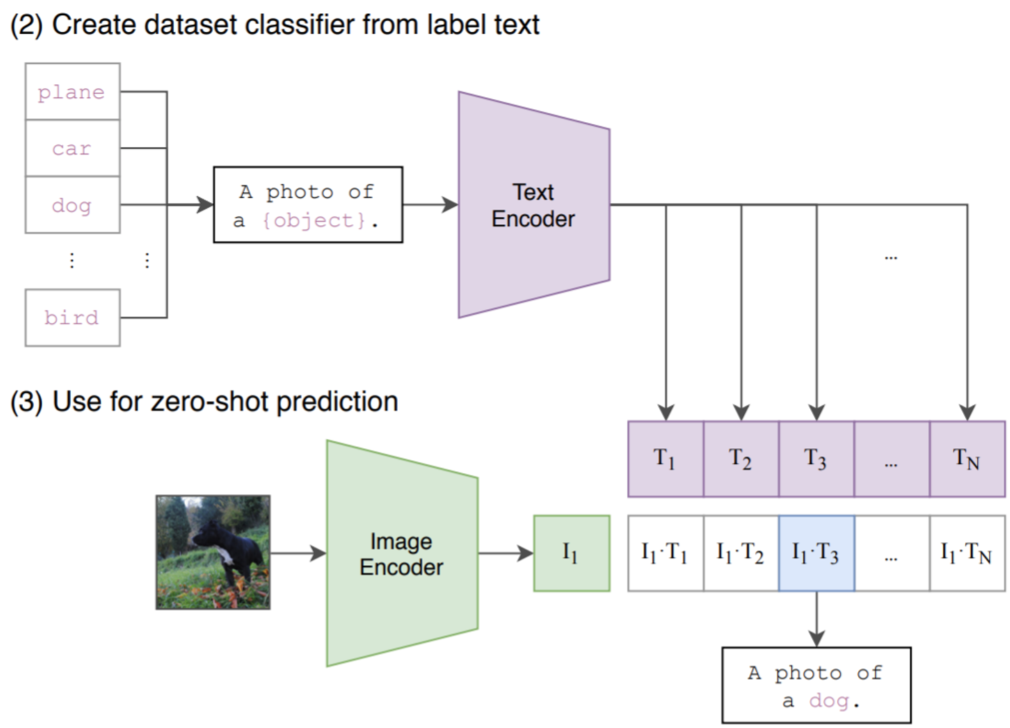
- No data → no ability to learn spurious in-domain correlations.
- Very little data → harder to learn spurious correlations (?)

$$\underbrace{\boxed{E_{p(t),p(\theta)}[l(\theta,t)]}}_{\text{cross-domain loss}} \leq \underbrace{\boxed{E_{p(t),p(\theta|t)}[l(\theta,t)]}}_{\text{in-domain loss}} + \underbrace{\boxed{\mathcal{O}(\sqrt{(}I(\theta,t)))}}_{\text{information used}}$$

From adapting a bound by Xu and Raginsky 2017

Image classification via zero-shot learning (CLIP)

Say we can **jointly embed** images and text into the same space..
Then we can perform object detection by checking if "A photo of a dog" is a valid caption



How does CLIP work? (1)

How is this thing trained?

Scrape caption data from the internet

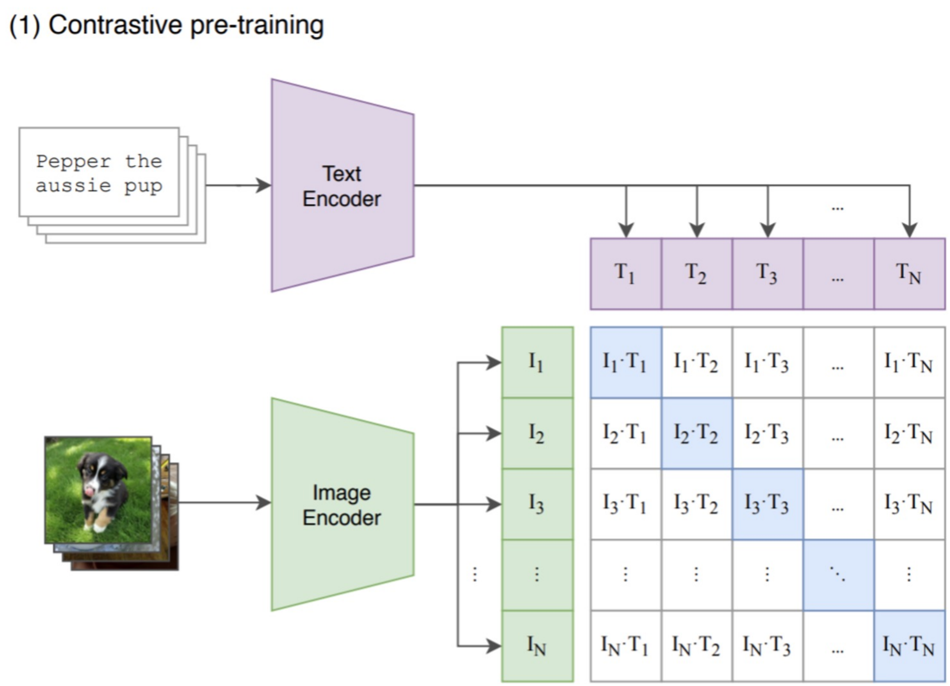
- (image, text pairs filtered)
- 400,000,000!!!

Encoders

- Image: ResNet, ViT
- Text: Transformer

Train ‘contrastively’

- large batches (32K)
- positive example: paired caption
- negative example: all other captions

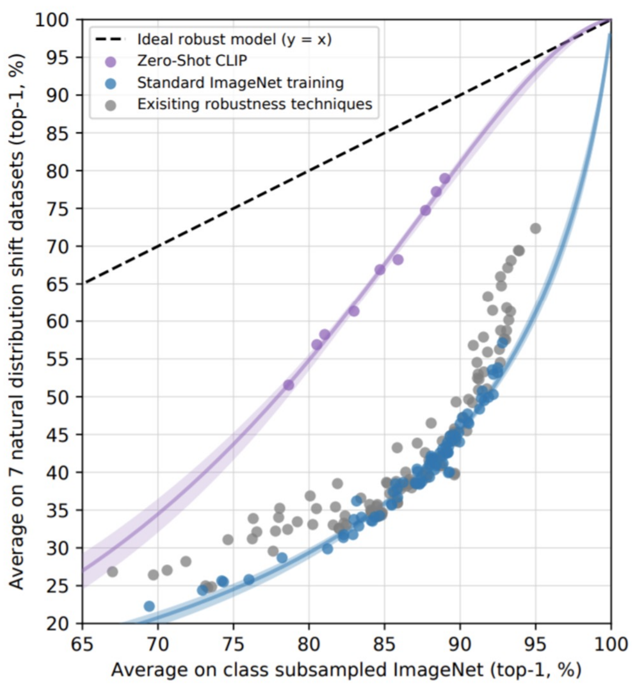


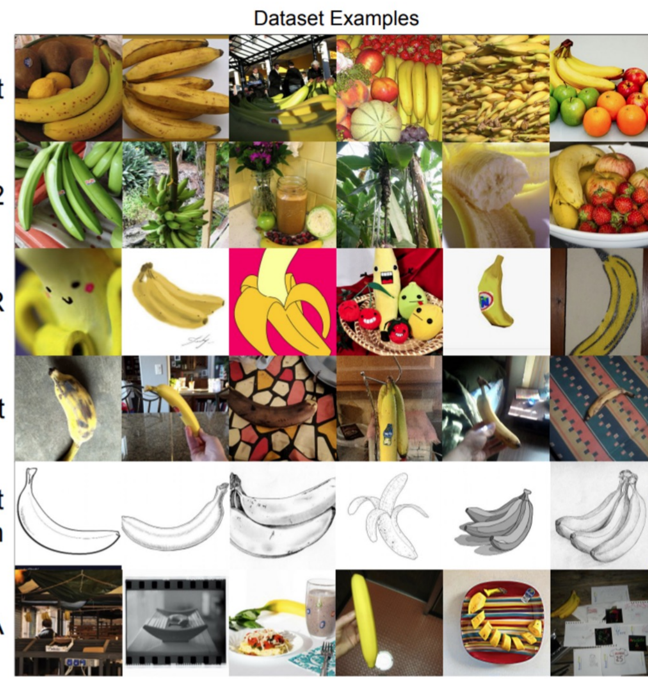
How does CLIP work (2)?

```
# 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.

Observations from a zero-shot model (CLIP)

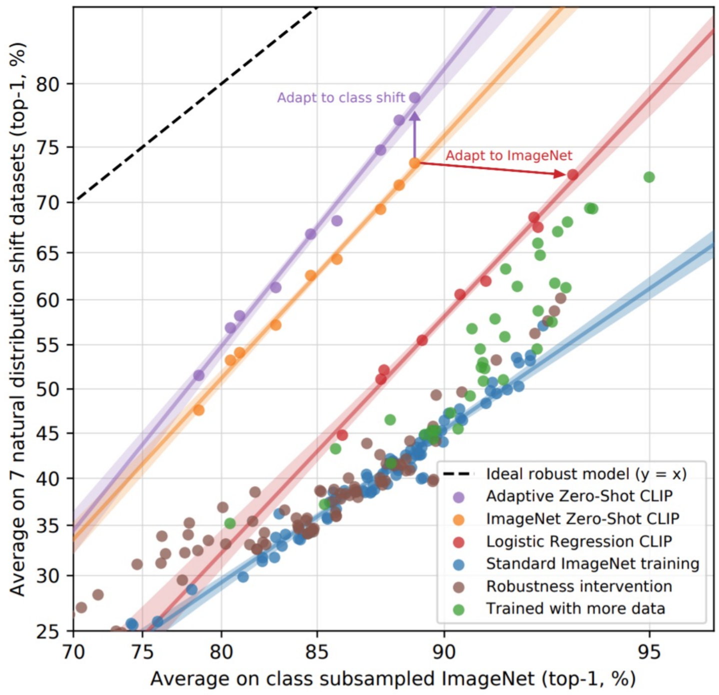


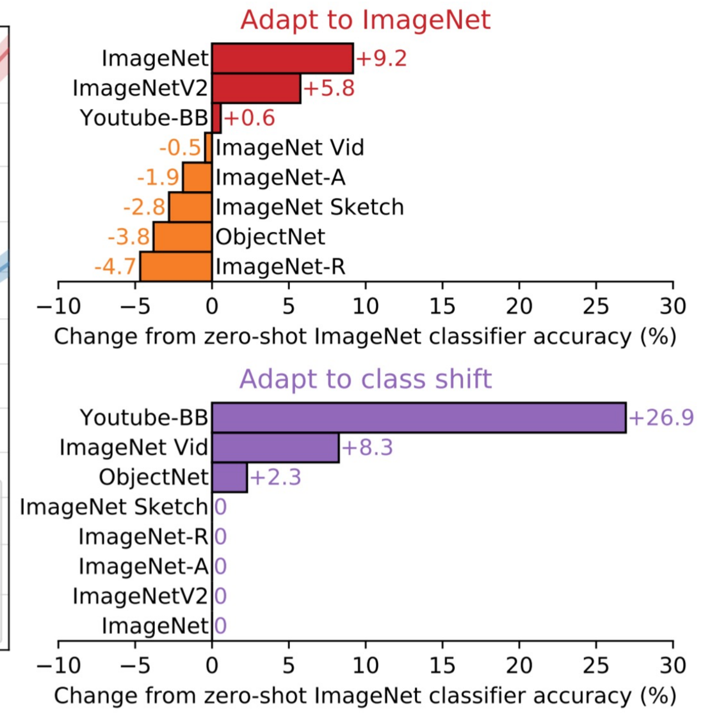


	ImageNet ResNet101	Zero-Shot CLIP	Δ Score
ImageNet	**76.2**	**76.2**	0%
ImageNetV2	64.3	**70.1**	+5.8%
ImageNet-R	37.7	**88.9**	+51.2%
ObjectNet	32.6	**72.3**	+39.7%
ImageNet Sketch	25.2	**60.2**	+35.0%
ImageNet-A	2.7	**77.1**	+74.4%

More robustness observations

Fine-tuning on imagenet data kills these robustness gains (red line)





Problems are not a lack of data!

Few shot robustness

Few-shot performance also shows similar trends.

As we add data (1-shot to 128-shot to all)

- absolute robustness increases.
- relative robustness decreases.

‘Zero shot and few shot models are inherently robust’

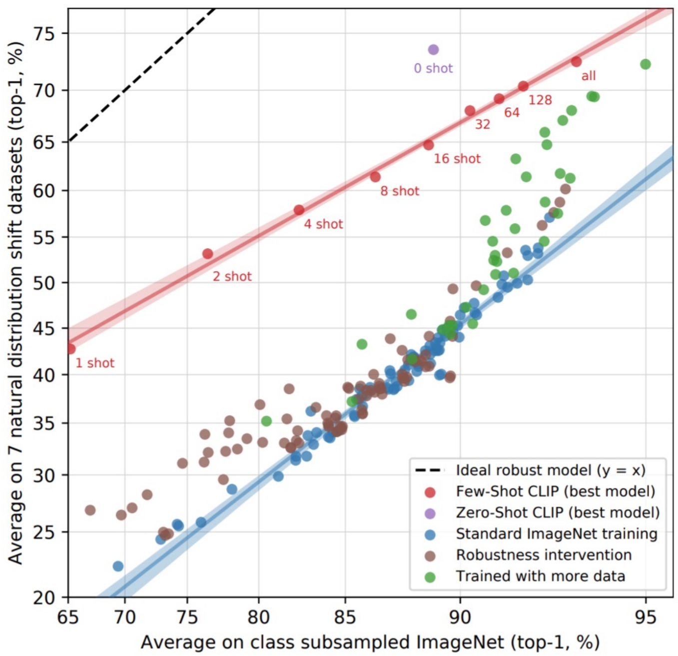


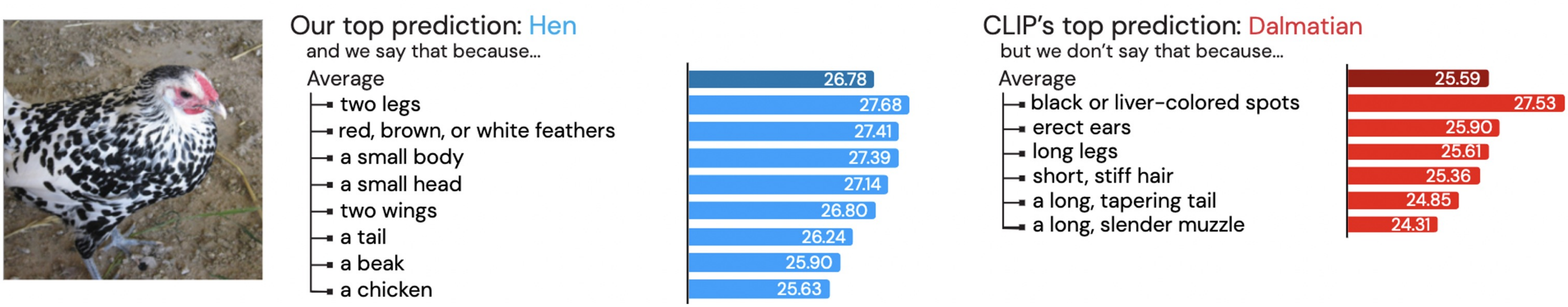
Figure 15. **Few-shot CLIP also increases effective robustness compared to existing ImageNet models but is less robust than zero-shot CLIP.** Minimizing the amount of ImageNet training data used for adaption increases effective robustness at the cost of decreasing relative robustness. 16-shot logistic regression CLIP matches zero-shot CLIP on ImageNet, as previously reported in Figure 7, but is less robust.

Visual Classification via Description from LLM

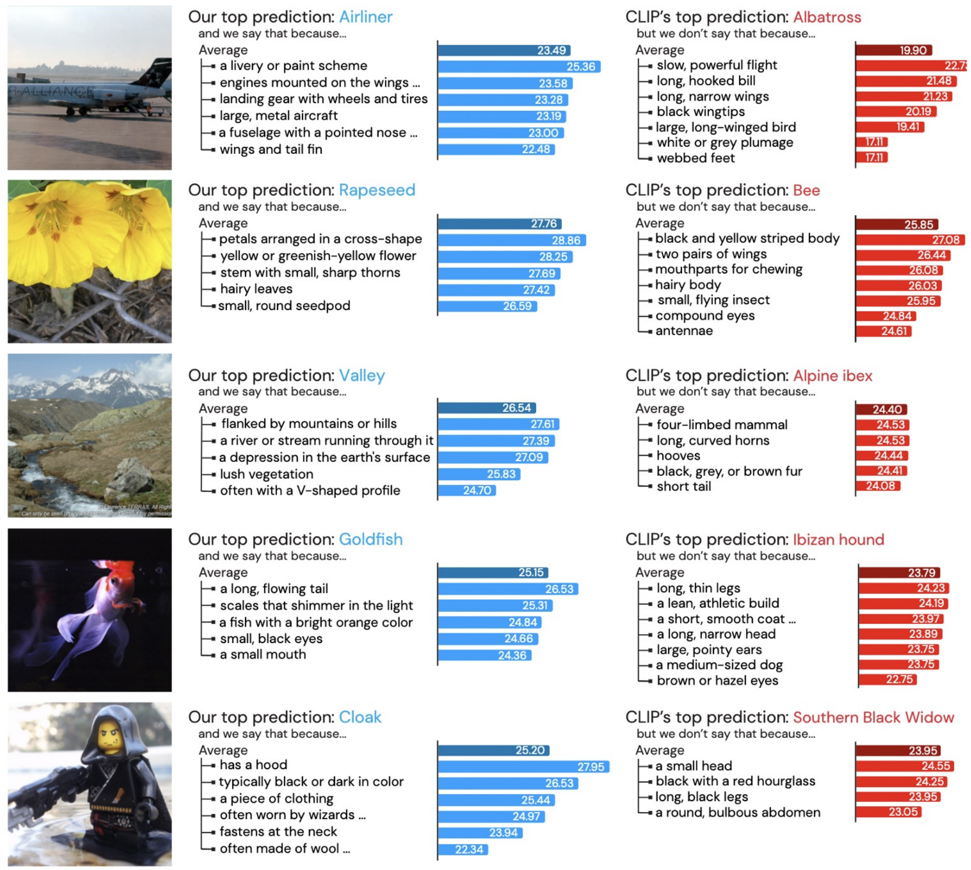
By only using the category name, FSL w/ CLIP neglects to use rich context information available via language

- Gives no intermediate understanding of why a category is chosen
- Provides no mechanism for adjusting the criteria used towards this decision.

Menon & Vondrick (2022) use class descriptions from LLMs classify based on descriptive features

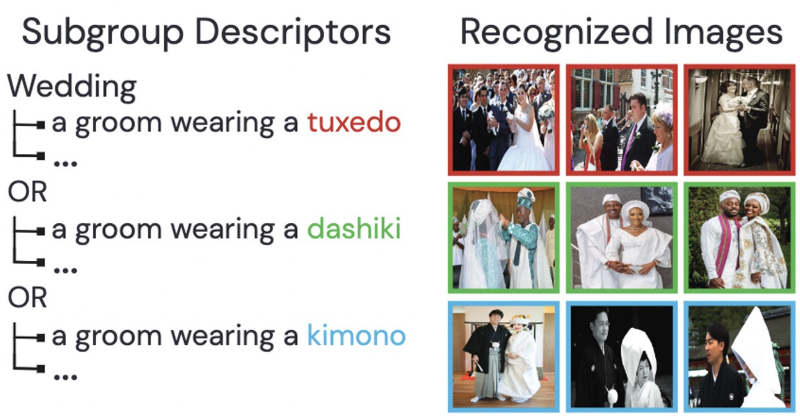


Visual Classification via Description from LLM



Visual Classification via Description from LLM

Richer class descriptions can help mitigate bias!



Sub-group	Ours	CLIP
Western African	100%	40%
Chinese	100%	20%
Japanese	100%	0%
North Indian	100%	60%

Figure 6: (left) CLIP only compares to the word 'wedding', yielding biased results – it only correctly recognizes the first row. The descriptor-based approach provides a way to address the bias, by expanding the initial set of descriptors (only the top) to be more inclusive with prior knowledge. (right) Modifying the descriptors to be more inclusive causes accuracy to significant improve on sub-groups.

Robustness in Modern NLP

Up until now, we have focused on robustness in modern computer vision

→What about Natural Language Processing?

Modern NLP is focused on zero-shot and few-shot generalization via a paradigm called **In-Context Learning** applied to **large language models**

→popularized by GPT-3 (Brown 2021)

→language model can perform arbitrary tasks!

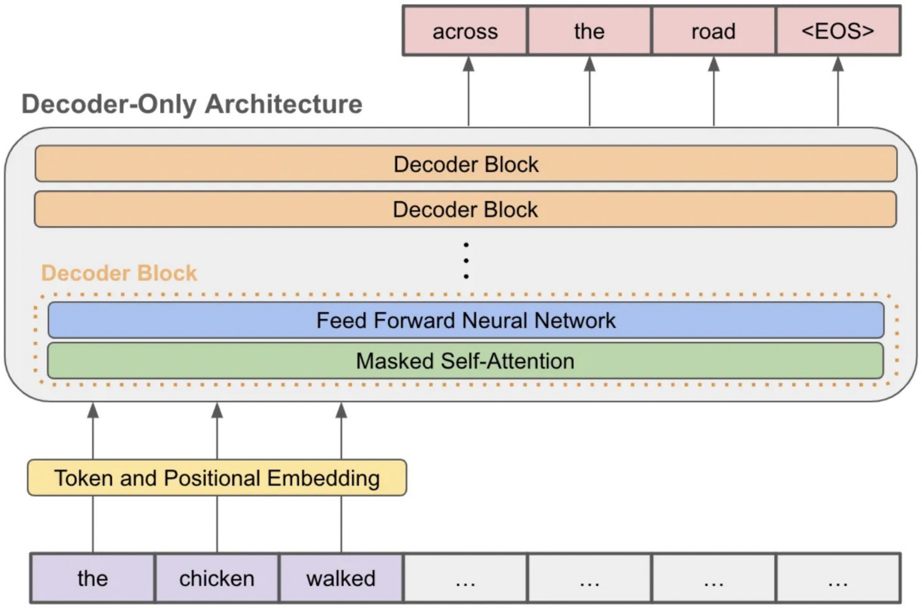
Language Modeling

Objective: Predict most likely word conditioned on some input string

$$p(x) = \prod_{i=1}^{n} p(s_n | s_1, ..., s_{n-1})$$

Generative language models are trained on massive corpora to predict the next word

Language is generated left-to-right, one word at a time



In Context Learning

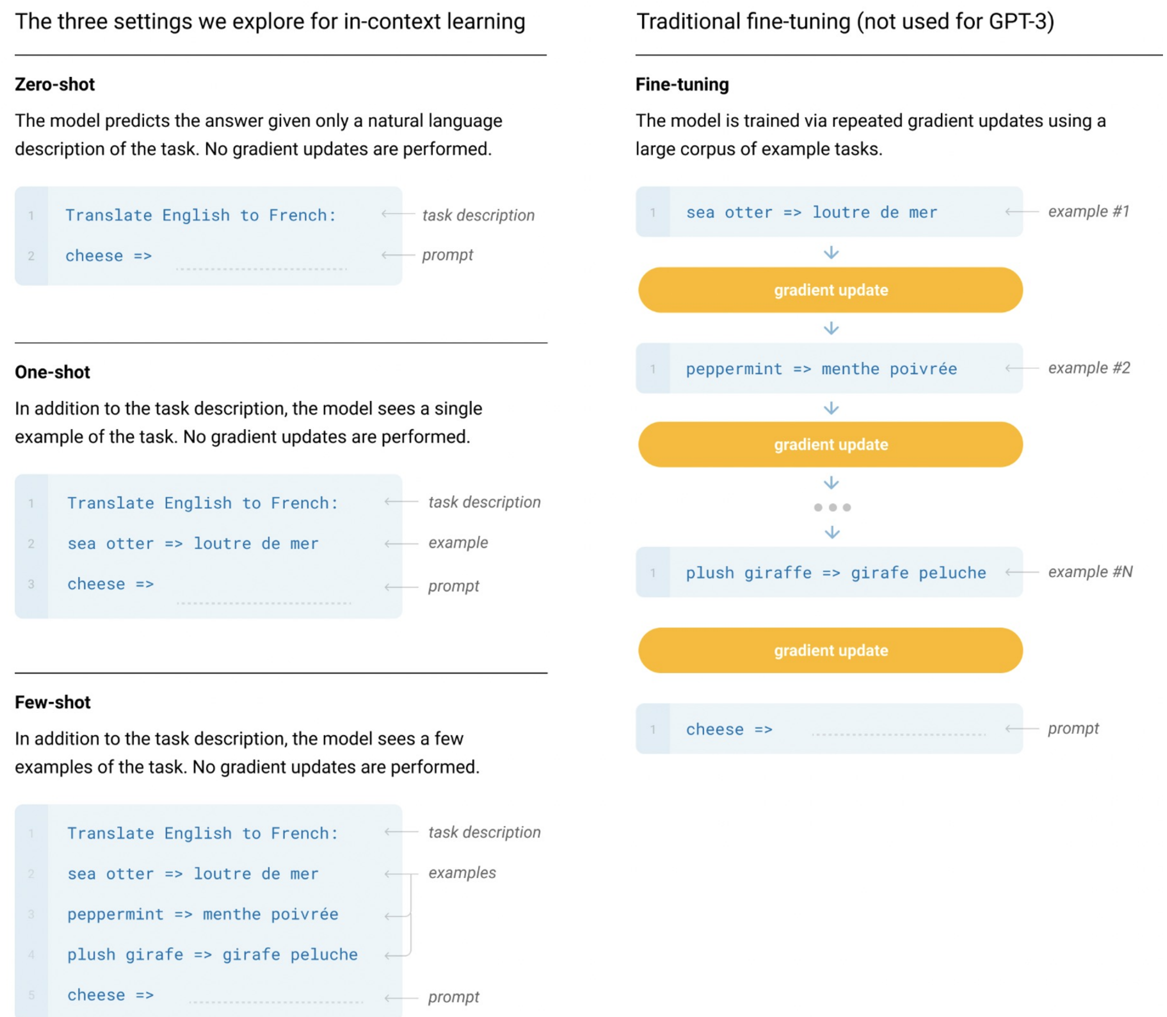
Predictions are generated by conditioning on a task-relevant prompt

Prompt components:

- task description
- examples
- query

“Learn” the task being performed from in-context examples

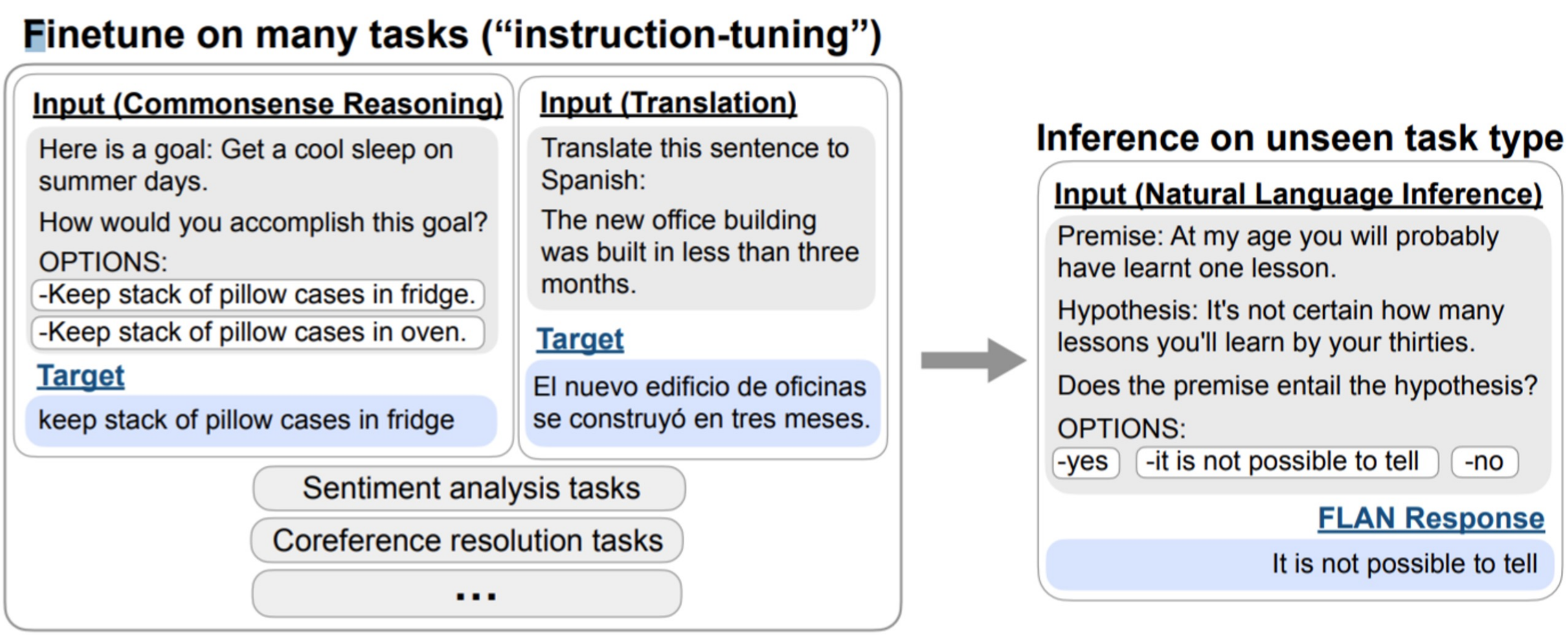
- Relevant context
- Label space
- Answer format
- Input-output correspondence?



Instruction Tuning

CLIP: Zero-shot across different object classes via language embedding.

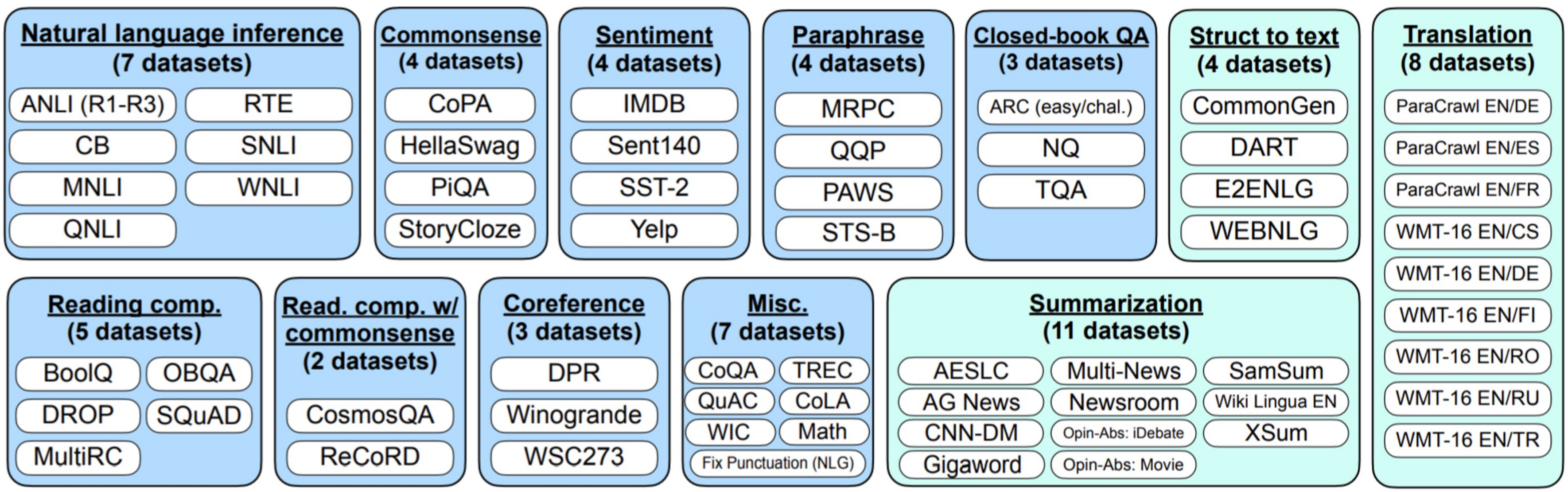
Instruction Tuning: Zero-shot across different *tasks* via language.



How does this relate to robustness?

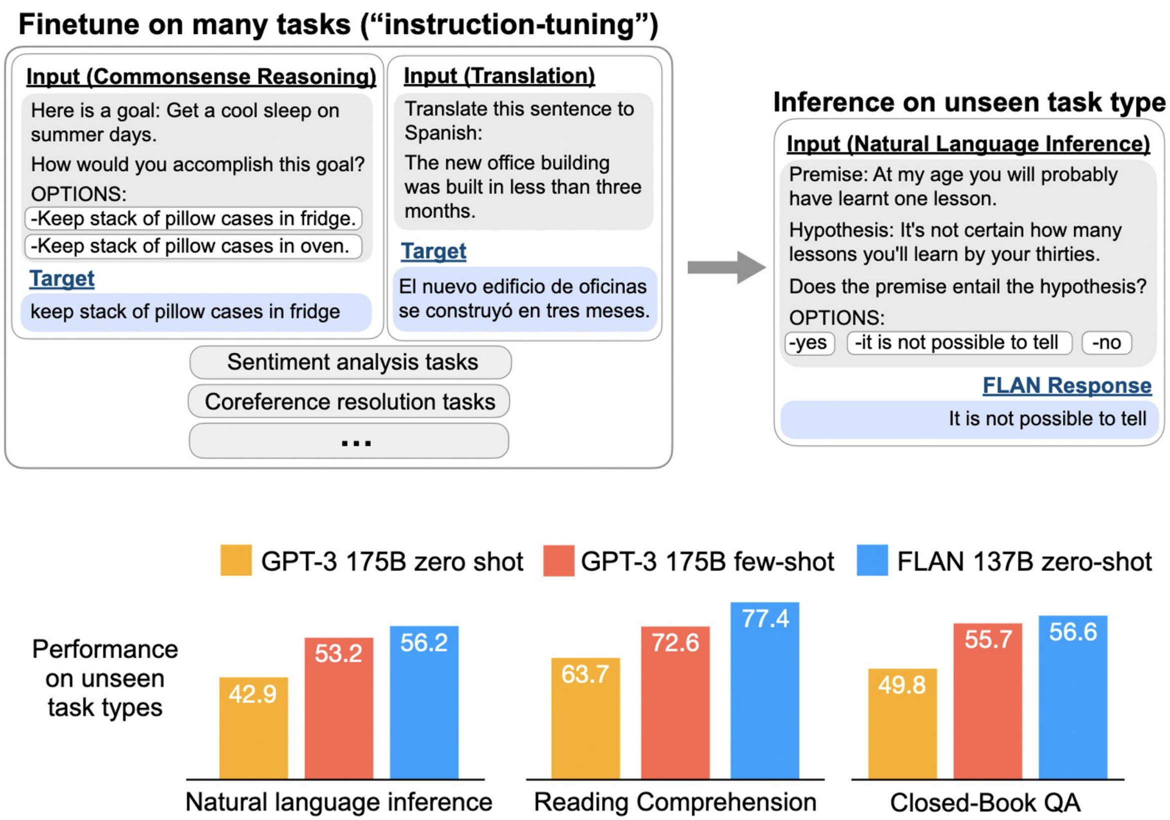
CLIP: zero-shot learning to avoid dataset biases
Instruction-tuning: zero-shot learning to avoid task biases

Define a **task** with a set of datasets, split into **train and test tasks**



Instruction Tuning

These zero shot models are inherently robust. The key is to make them perform well



Benefits of massive multitasking + zero-shot learning

Remarkably good zero-shot performance now achievable: within 10% of supervised.

	READING COMPREHENSION			CLOSED-BOOK QA			
	BoolQ acc.	MultiRC F1	OBQA acc.	ARC-e acc.	ARC-c acc.	NQ EM	TQA EM
Supervised model	91.2[a]	88.2[a]	85.4[a]	92.6[a]	81.1[a]	36.6[a]	60.5[a]
Base LM 137B zero-shot	81.0	60.0	41.8	76.4	42.0	3.2	21.9
· few-shot	79.7	59.6	50.6	80.9	49.4	22.1	63.3
GPT-3 175B zero-shot	60.5	72.9	57.6	68.8	51.4	14.6	64.3
· few-shot	77.5	74.8	65.4	70.1	51.5	29.9	71.2
FLAN 137B zero-shot							
- average template	80.2 ▲2.7 std=3.1	74.5 ↑2.4 std=3.7	77.4 ▲12.0 std=1.3	79.5 ▲8.6 std=0.8	61.7 ▲10.2 std=1.4	18.6 ↑4.0 std=2.7	66.5 ↑2.2 std=2.6
- best dev template	82.9 ▲5.4	77.5 ▲2.7	78.4 ▲13.0	79.6 ▲8.7	63.1 ▲11.6	20.7 ↑6.1	68.1 ↑3.8

Table 2: Results on reading comprehension and closed-book question answering. For FLAN, we report both the average of up to ten templates, as well as the best dev template. The triangle ▲ indicates improvement over few-shot GPT-3. The up-arrow ↑ indicates improvement only over zero-shot GPT-3. [a]T5-11B.

Key commonalities between CLIP and instruction-tuning

Key takeaways

- Zero-shot models are *inherently* robust.
- One path to building effective robust models is to build effective zero-shot ones
- Language is a common interface across tasks
  - Progress in large language models is causing an explosion in zero-shot learning progress across vision, robotics, etc.

Chain Of Thought Prompting

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models - Wei et al. (2022)

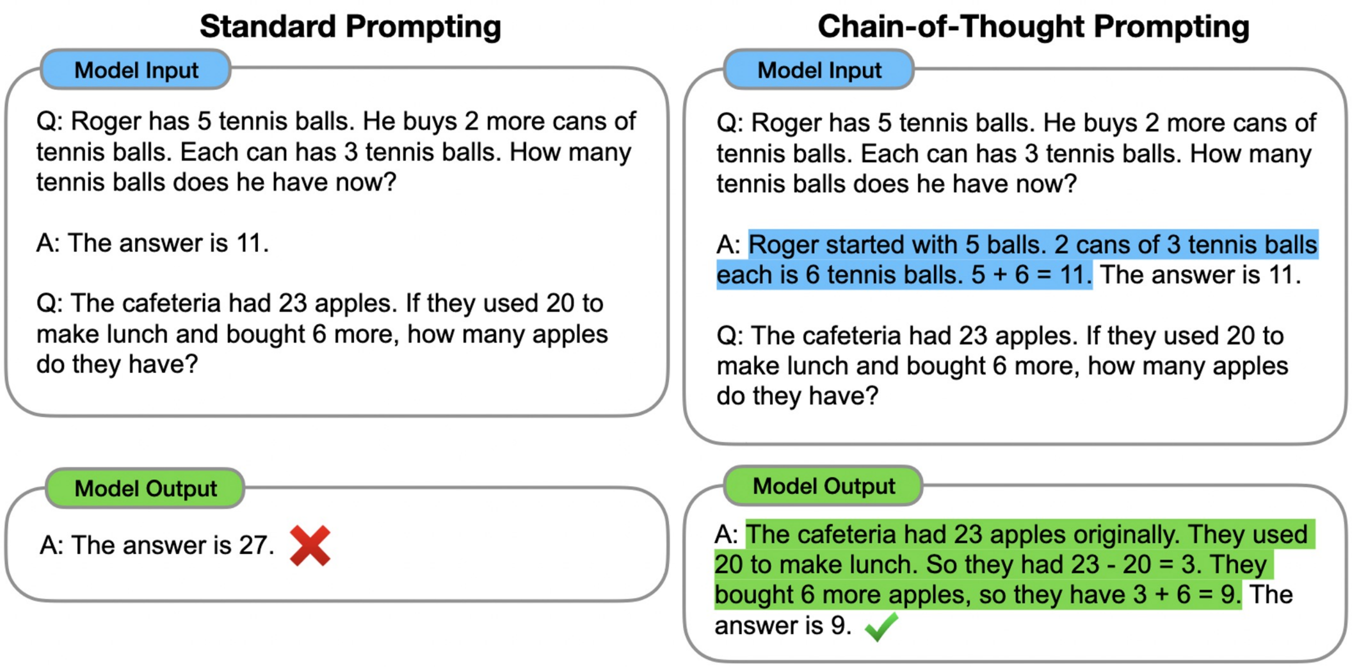


Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

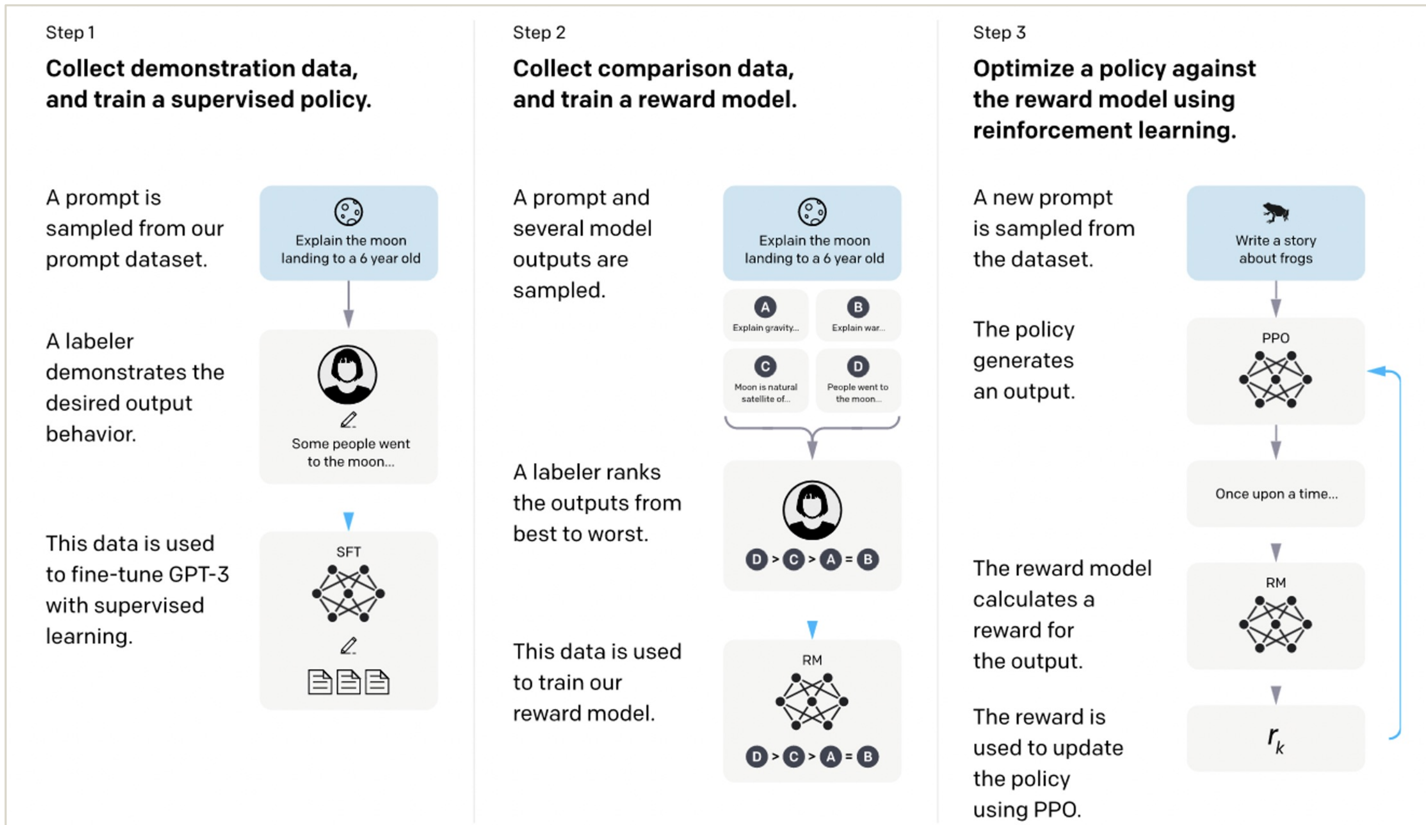
Reinforcement Learning From Human Feedback

Instruction tuning relies on typical NLP datasets to generate ICL examples

Under RLHF, collect prompts and desired outputs from humans
→ Align with human preferences

Is RL necessary?

Ouyang 2022



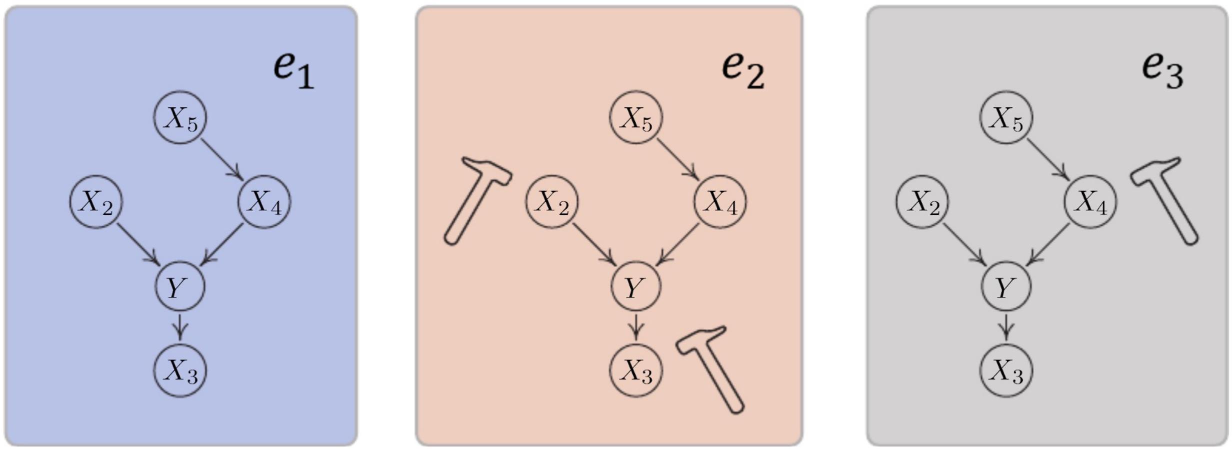
Big recap slide

So.. what helps for transfer?

- Model architectures: **Not really** (even neural vs not neural)
- Data: **Not for i.i.d** , **a little for non-iid**
- Pre-training: **Yes, both finetuning and more generally**
- Adversarial robustness: **Yes, but at a great cost**
- Zero-shot/multitask: **Yes**

Direction 1: get more similar environments

How else can we make progress on generalization to new domains?



In the multitask approaches: observe many tasks (environments), embed them into a common space, learn a single predictor

A related, causal view: observe many environments (for a single task), learn a predictor that works well across all environments.

Direction 2: constraining the target distribution

Today – we operated on zero knowledge of the target. What if we know a bit more?

test loss **worst-case loss**

$$\mathbb{E}_{p_{\mathrm{x}}^{\text{test}}}[\ell(x;\theta)] \leq \sup_{p_{\mathrm{x}} \in \mathcal{P}} \mathbb{E}_{p_{\mathrm{x}}}[\ell(x;\theta)]$$

Cannot be computed because the **test distribution is unknown.**

Upper bound holds whenever $\mathcal{P}$ contains p^{test}

If we can identify the target distribution up to a 'neighborhood' we can use worst-case optimization to ensure good performance.

This lets us incorporate our knowledge of the test distribution without data.

Conclusion and reminders

Empirical (effective) robustness

- **Things that (surprisingly) don't help**: better models, more (iid) data
- **Things you might do for robustness:** better data, pre-training
- **Emerging idea**: zero-shot learning for robustness

Reminder

Project proposal due next Monday!