Patching open-vocabulary models by interpolating weights

Gabriel Ilharco, June 2022

Google Brain

Patching open-vocabulary models by interpolating weights

Ali Farhadi

Gabriel Ilharco

Hanna Hajishirzi

Ludwig Schmidt

Mitchell Wortsman

Samir Gadre

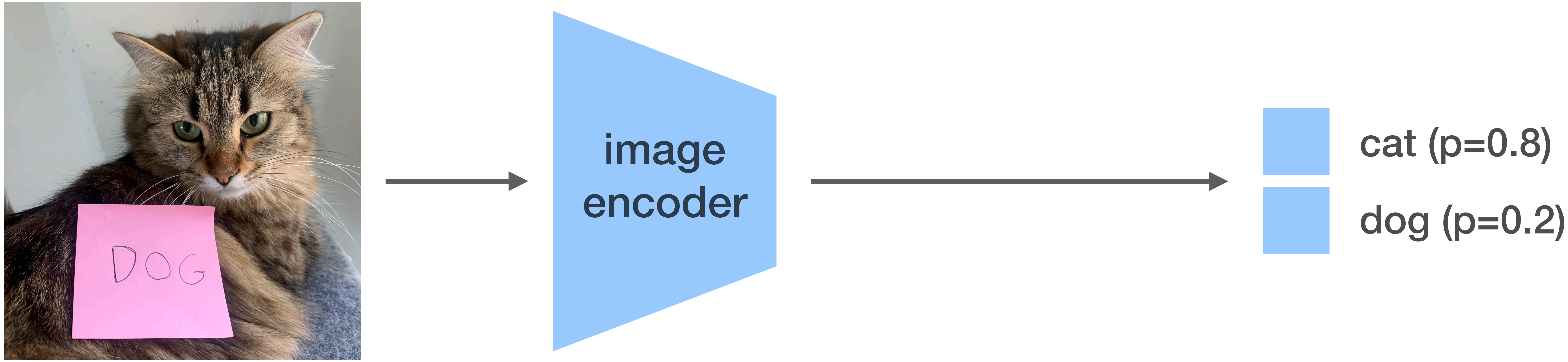
Shuran Song

Simon Kornblith

Background

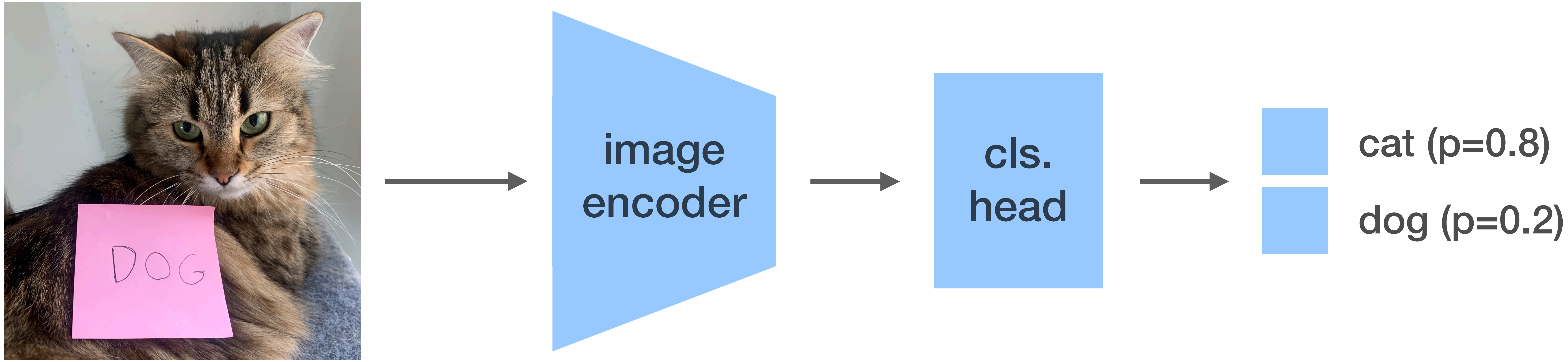
Background

Typical image classification models are **closed-vocabulary**



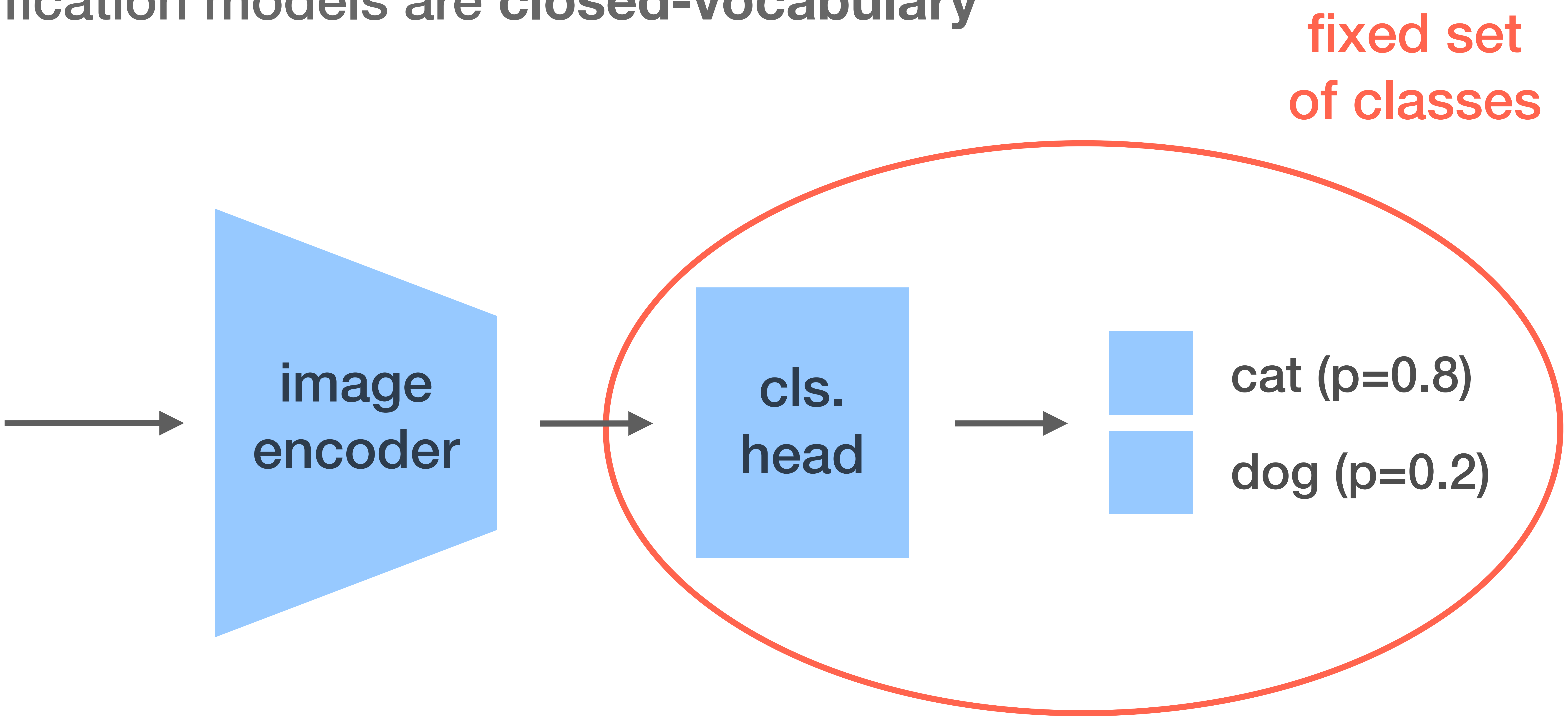
Background

Typical image classification models are **closed-vocabulary**

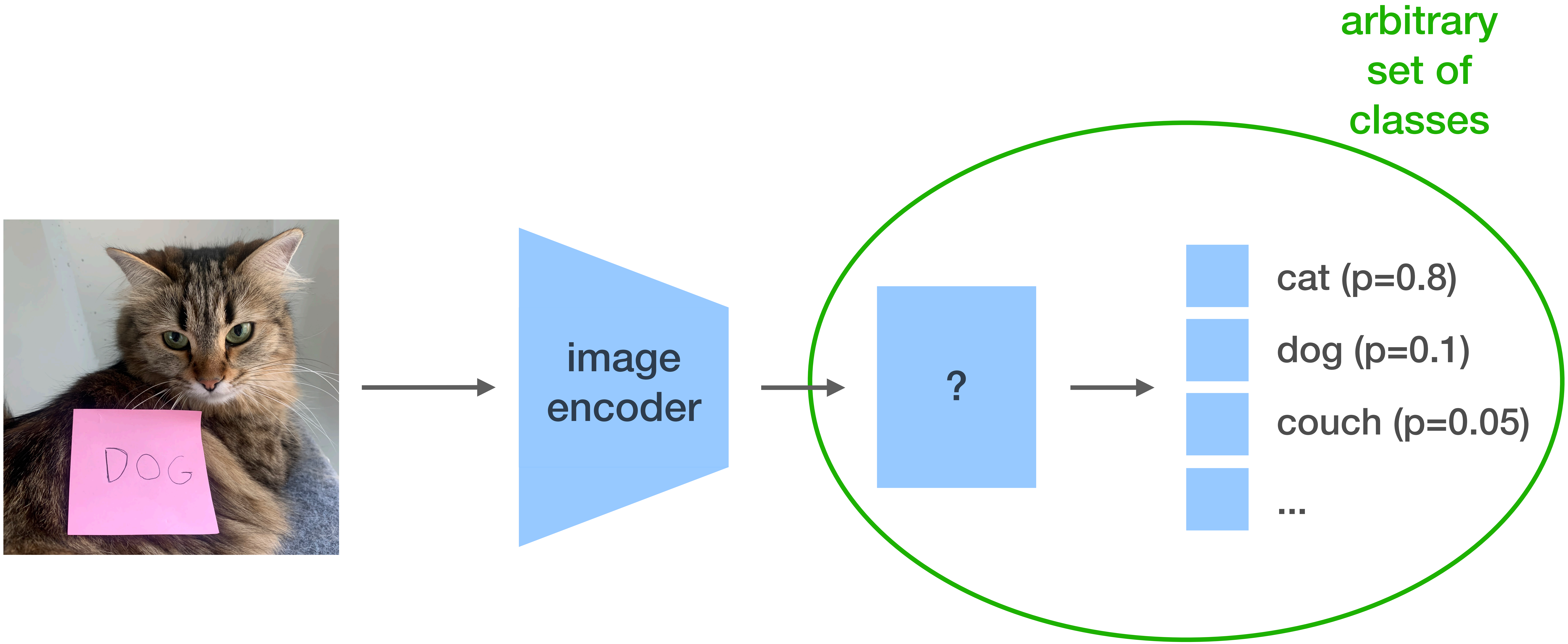


Background

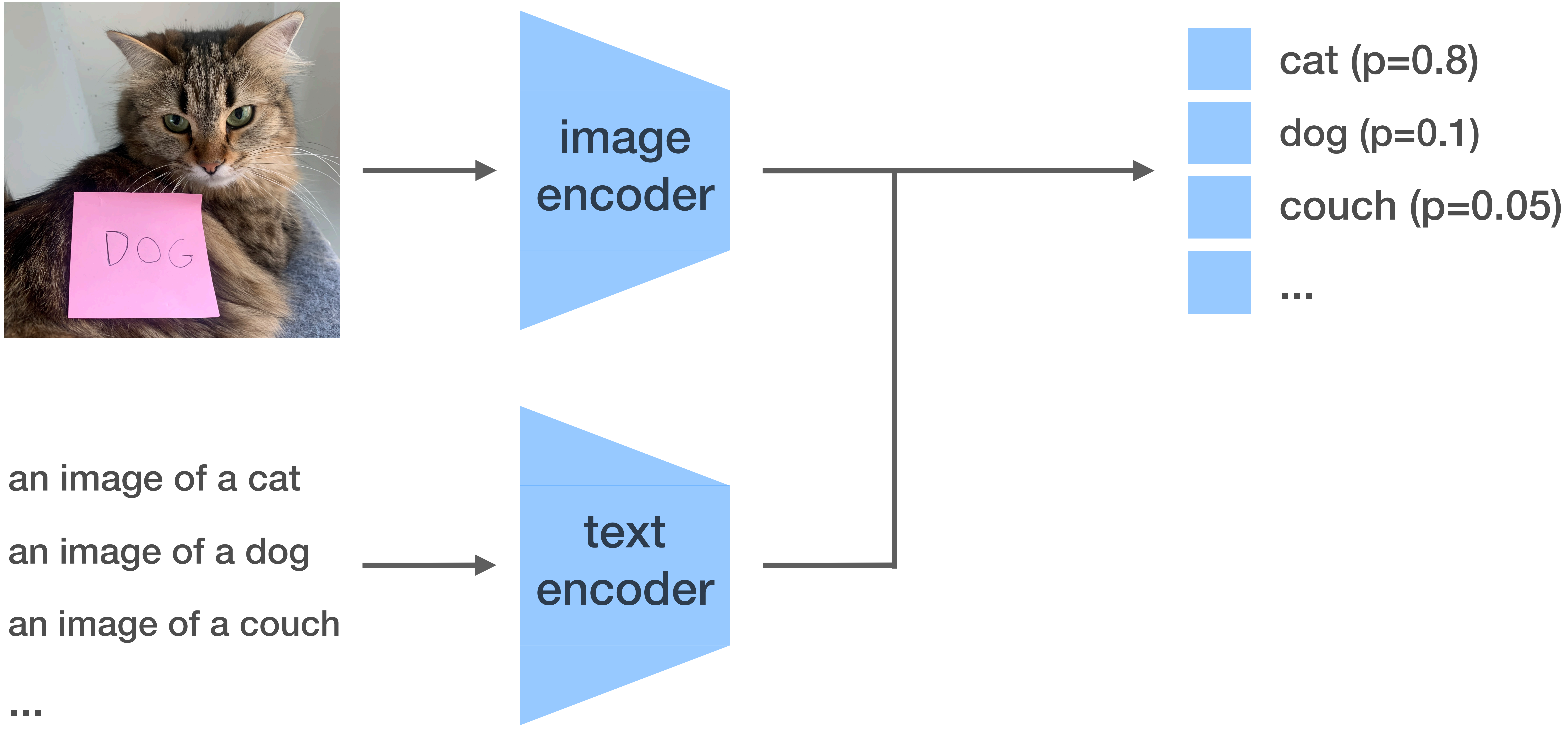
Typical image classification models are **closed-vocabulary**



Background: open-vocabulary models



Background: open-vocabulary models



Why are open-vocabulary models interesting?

A **single model** with high accuracy on many tasks

ImageNet: **85.7%**

CIFAR-10: **97.5%**

CIFAR-100: **82.3%**

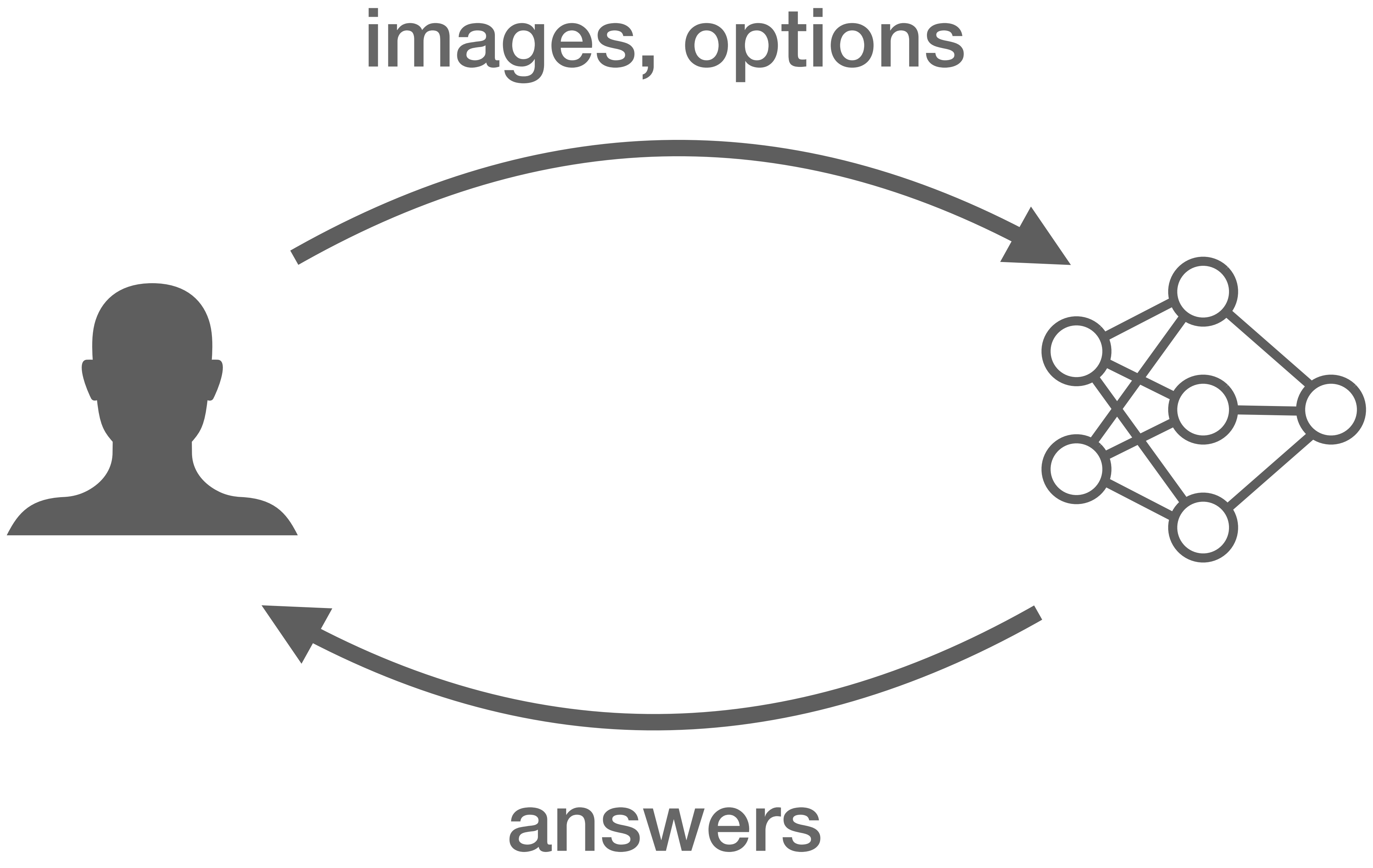
Flowers: **91.2%**

Caltech-101: **94.7%**

Pham et al., 2022

Why are open-vocabulary models interesting?

Open-vocabulary models as APIs



Why are open-vocabulary models interesting?

Tasks with high accuracy define a set that is **supported** by the API

ImageNet: **85.7%**

CIFAR-10: **97.5%**

CIFAR-100: **82.3%**

Flowers: **91.2%**

Caltech-101: **94.7%**

supported tasks

Pham et al., 2022

The limitations of open-vocabulary models

As any system, the set of supported capabilities is not exhaustive.

MNIST: 40.3%

EuroSAT: 51.0%

RESISC45: 72.7%

ImageNet: 85.7%

CIFAR-10: 97.5%

CIFAR-100: 82.3%

Flowers: 91.2%

Caltech-101: 94.7%

supported tasks

PCam: 59.6%

DTD: 64.6%

out-of-scope

Pham et al., 2022

What can we do?

Option 1: Re-train the model, adding data from the underperfoming tasks

- **pro:** keeps the model open-vocabulary
- **pro:** this might improve accuracy on other tasks
- **con:** this can be *very* expensive, and unreasonable to do multiple times

What can we do?

Option 1: Re-train the model, adding data from the underperfoming tasks

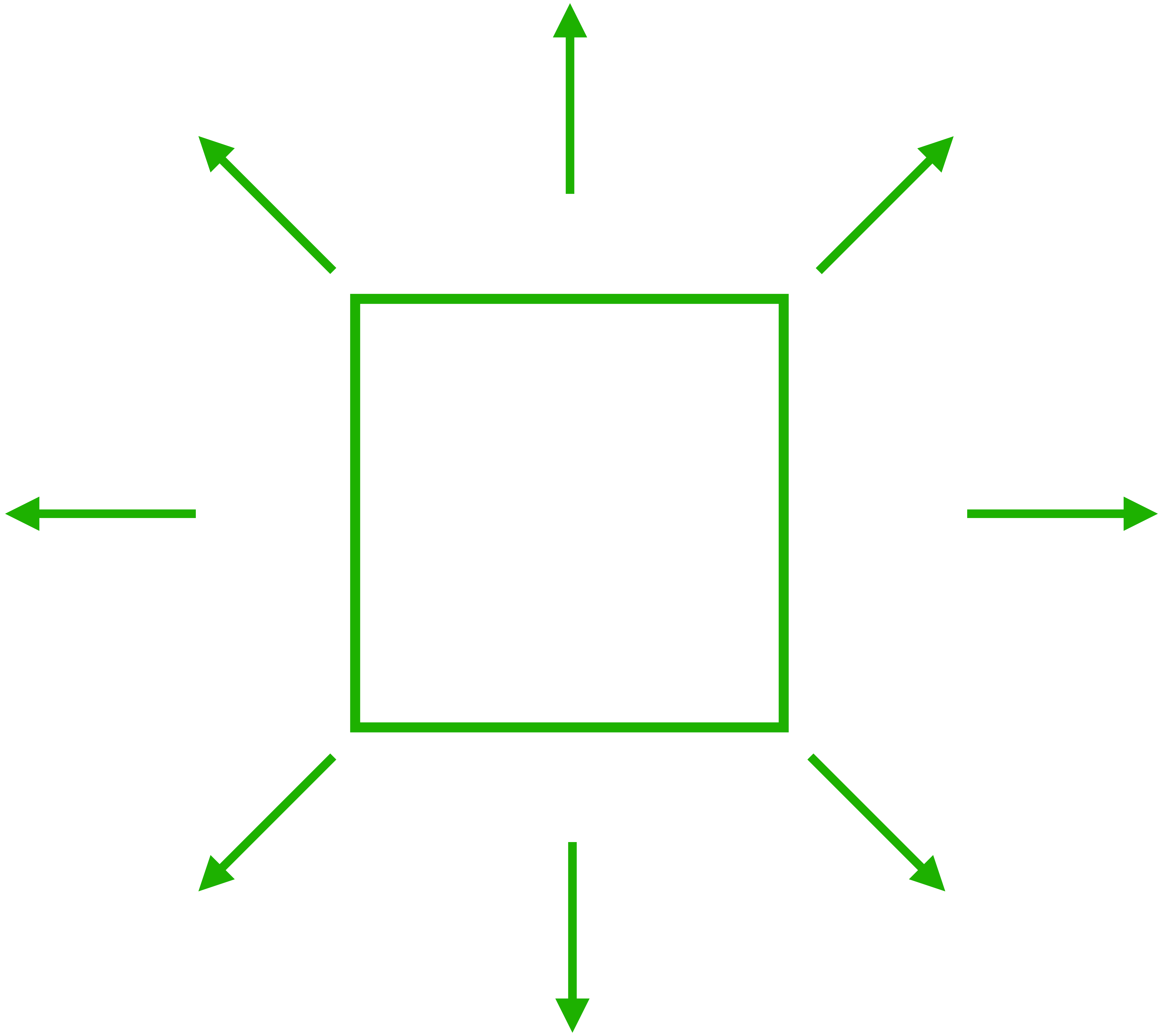
- **pro:** keeps the model open-vocabulary
- **pro:** this might improve accuracy on other tasks
- **con:** this can be *very* expensive, and unreasonable to do multiple times

Option 2: Fine-tune on data from the underperforming tasks

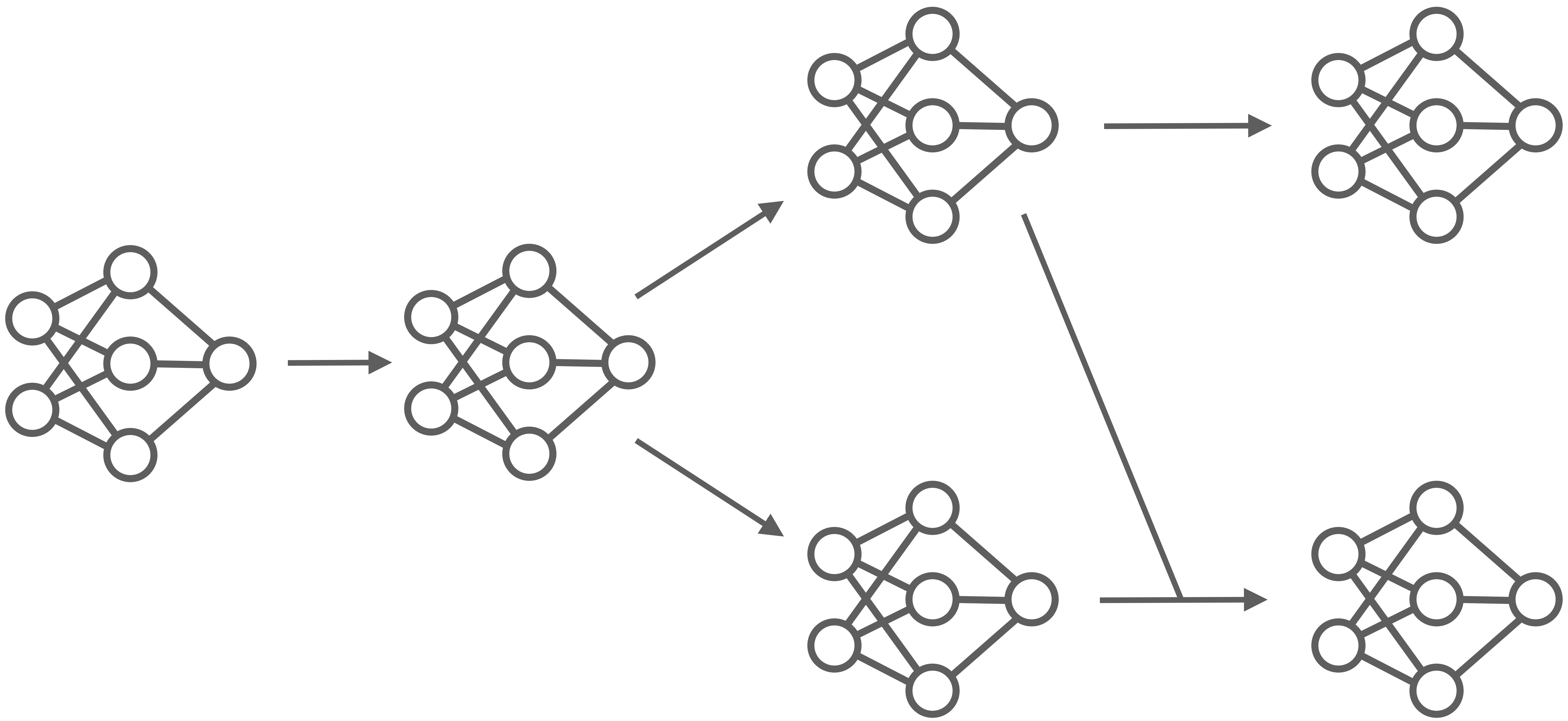
- **pro:** fast
- **con:** prone to overfitting and catastrophic forgetting
- **con:** typically makes models closed-vocabulary again

Patching

The goal of patching is to **expand** the set of supported tasks, without changing the model API

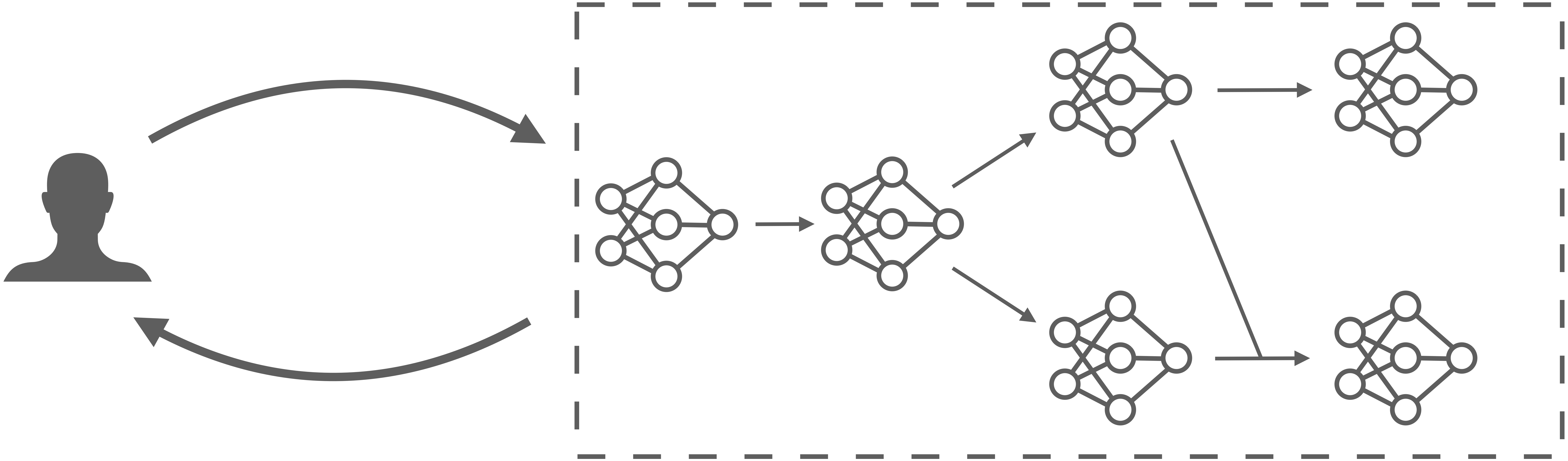


Building models like open-source software



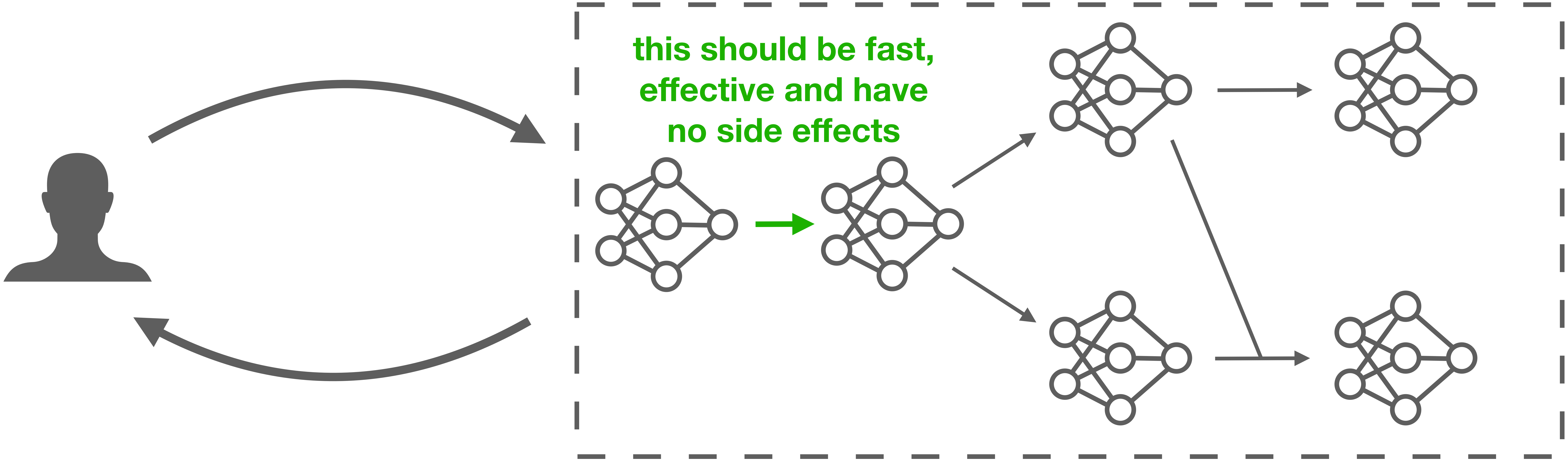
A Call to Build Models Like We Build Open-Source Software, Raffel 2021

Building models like open-source software



A Call to Build Models Like We Build Open-Source Software, Raffel 2021

Building models like open-source software



Patching by interpolating weights

Our work: A simple, two-step method for patching models:

Step 1: fine-tune on a target task, *without* introducing new parameters

Step 2: average the weights of the models before and after fine-tuning

Patching by interpolating weights

Our work: A simple, two-step method for patching models:

Step 1: fine-tune on a target task, *without* introducing new parameters

Step 2: average the weights of the models before and after fine-tuning

- **pro:** as fast as fine-tuning
- **pro:** models remain open-vocabulary
- **pro:** less catastrophic forgetting

The rest of this talk

1) Patching on a single task

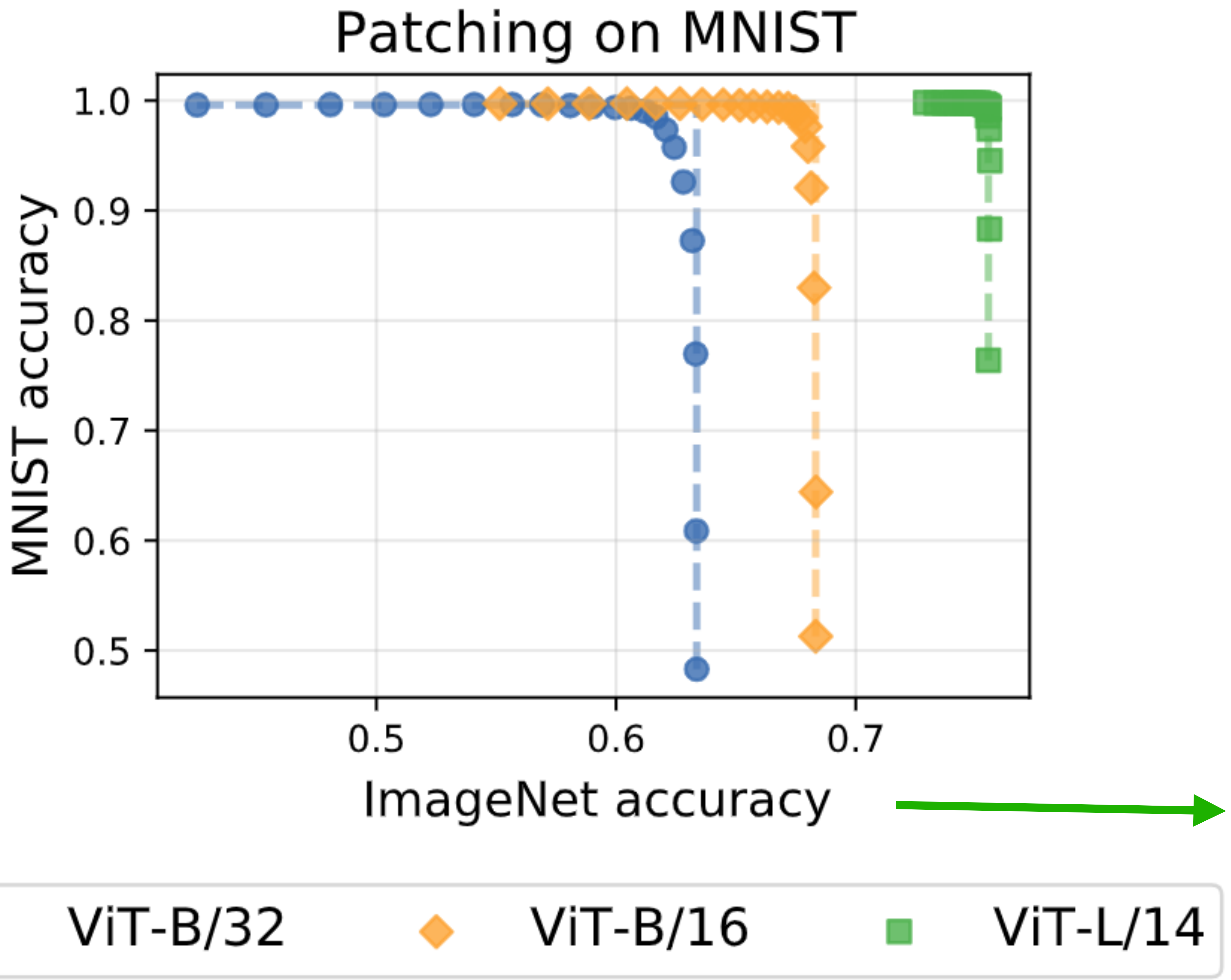
2) Patching on multiple tasks

3) Task generalization

4) Case studies

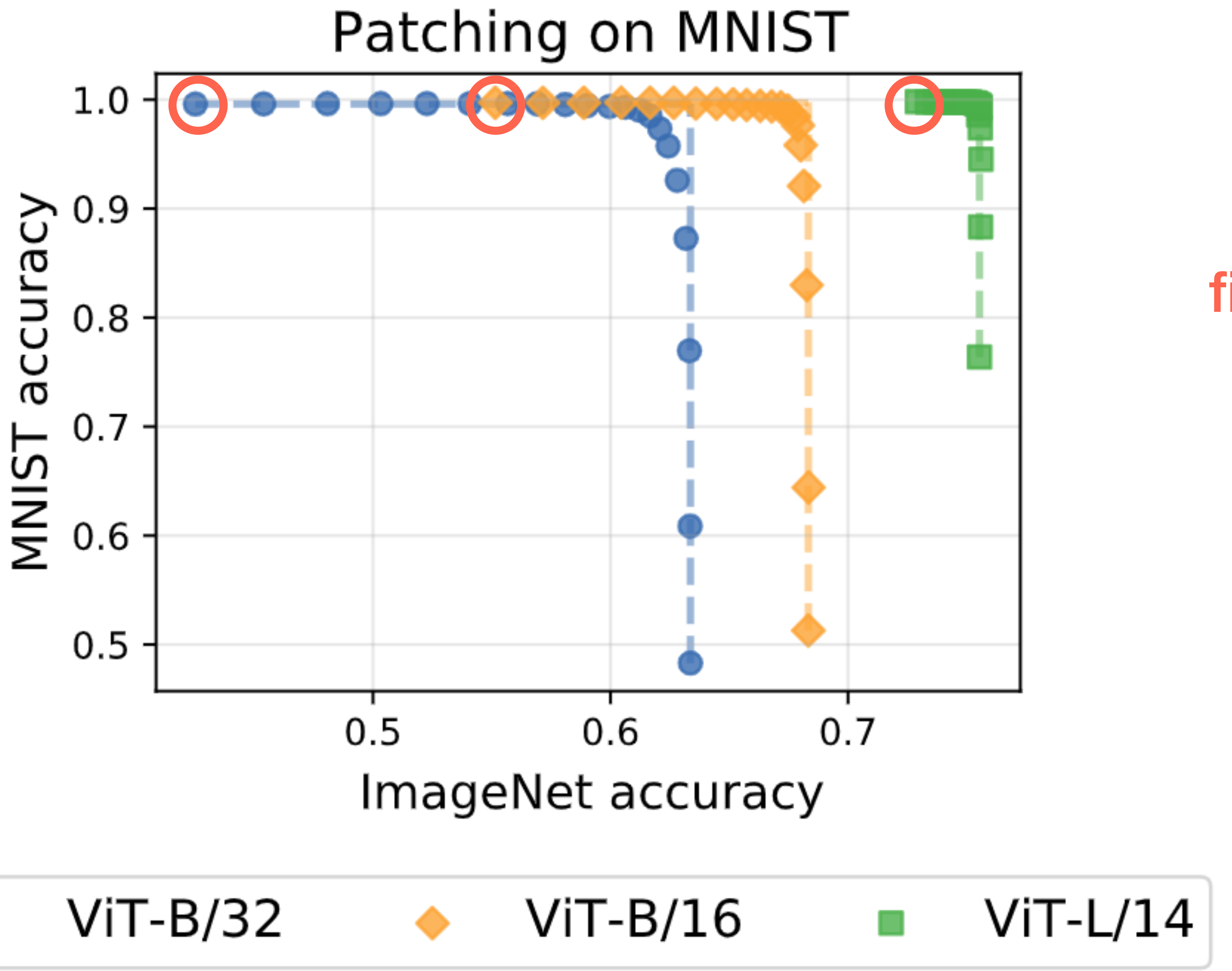
Patching on a single task

Patching on a single task



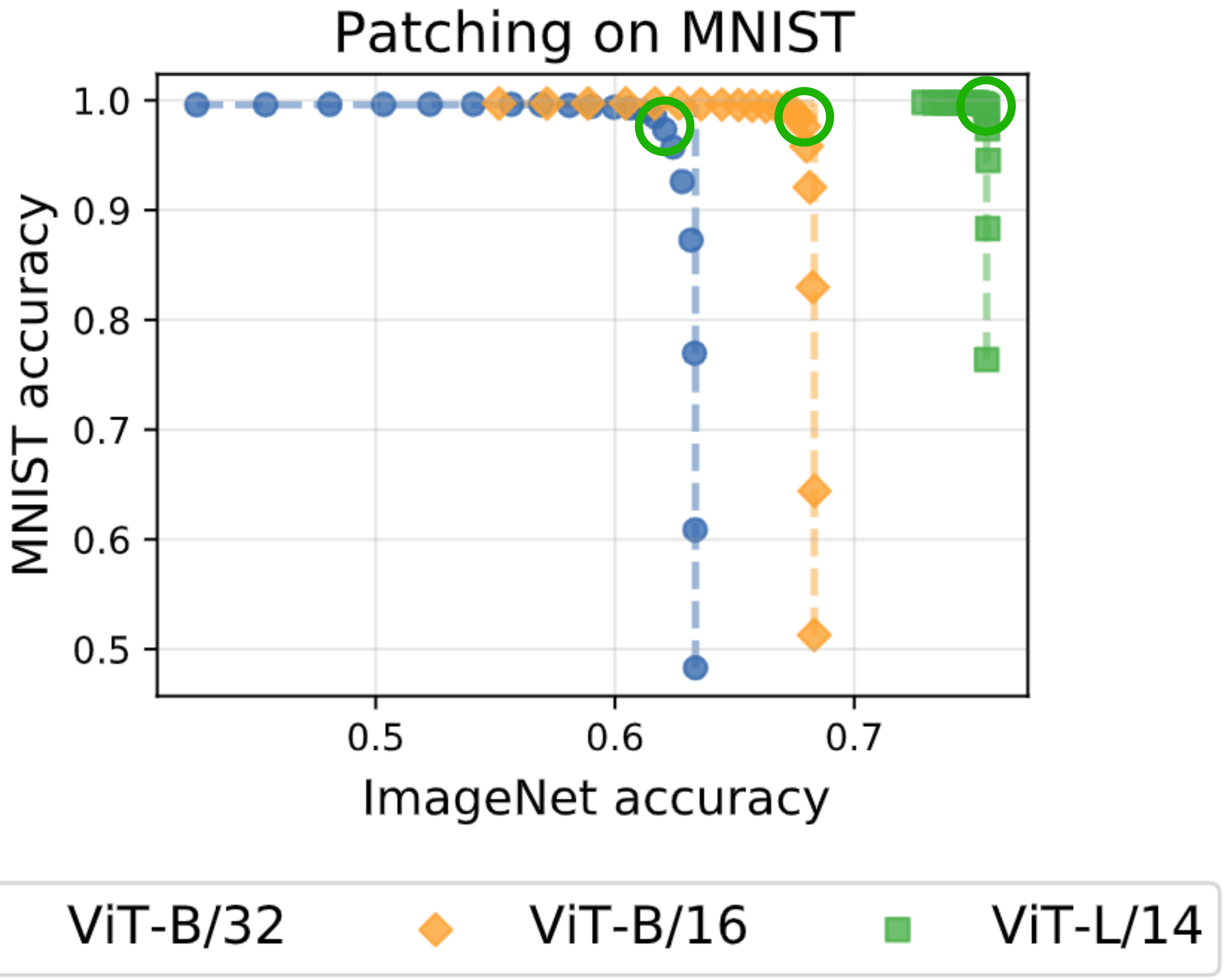
supported task

Patching on a single task



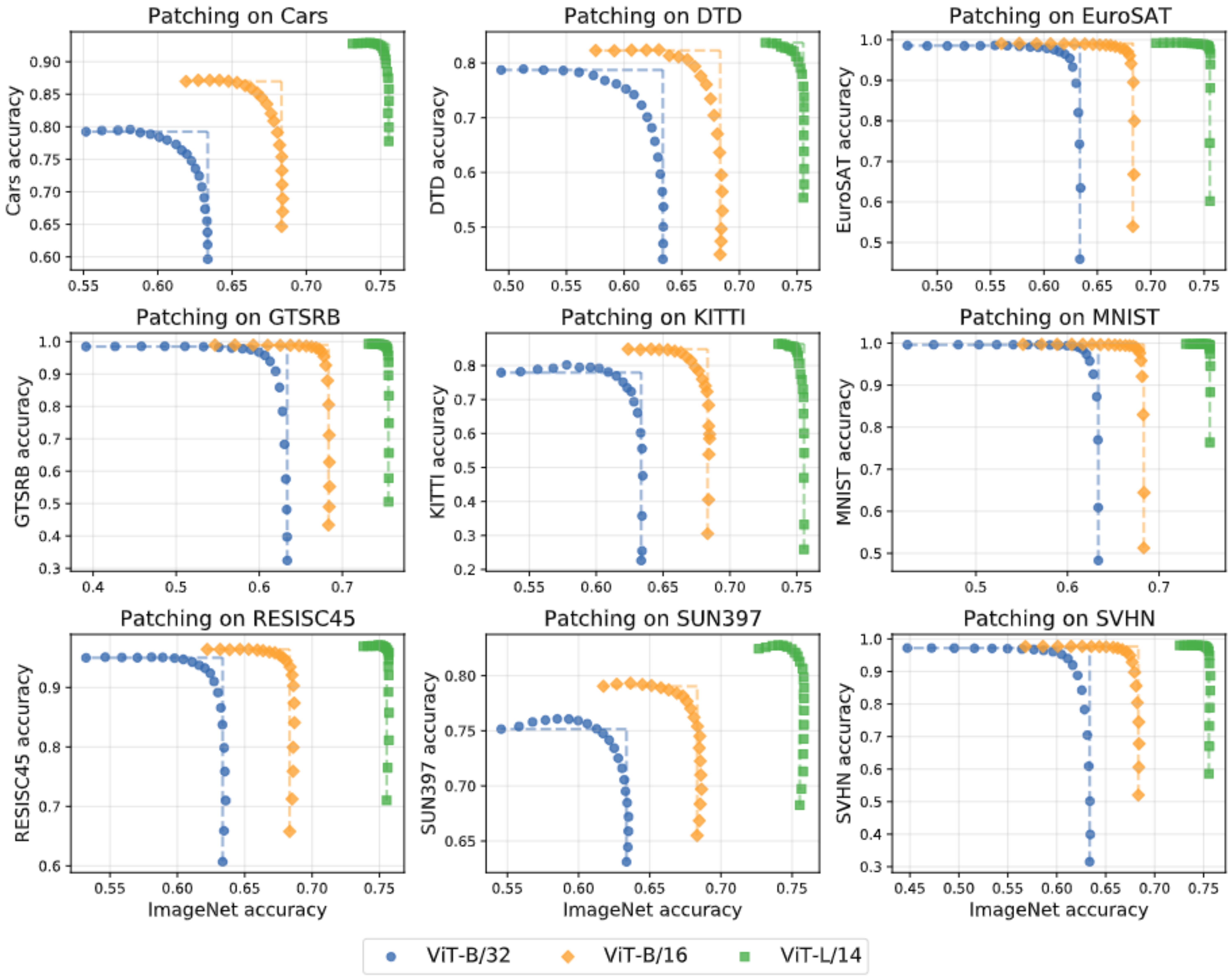
fine-tuning can hurt accuracy on the supported tasks

Patching on a single task

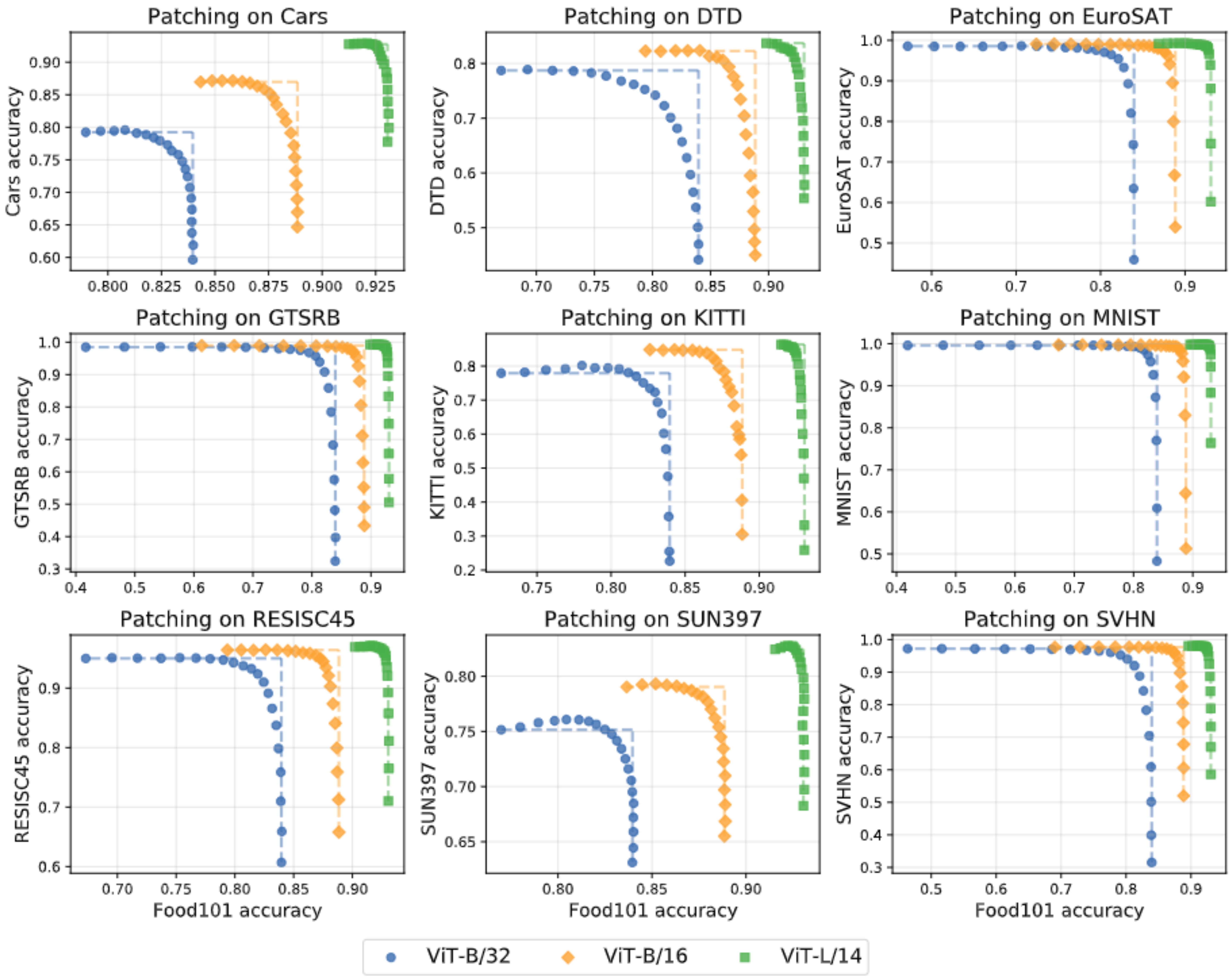


with weight interpolations, we are close to the point of no tradeoff

Patching on a single task

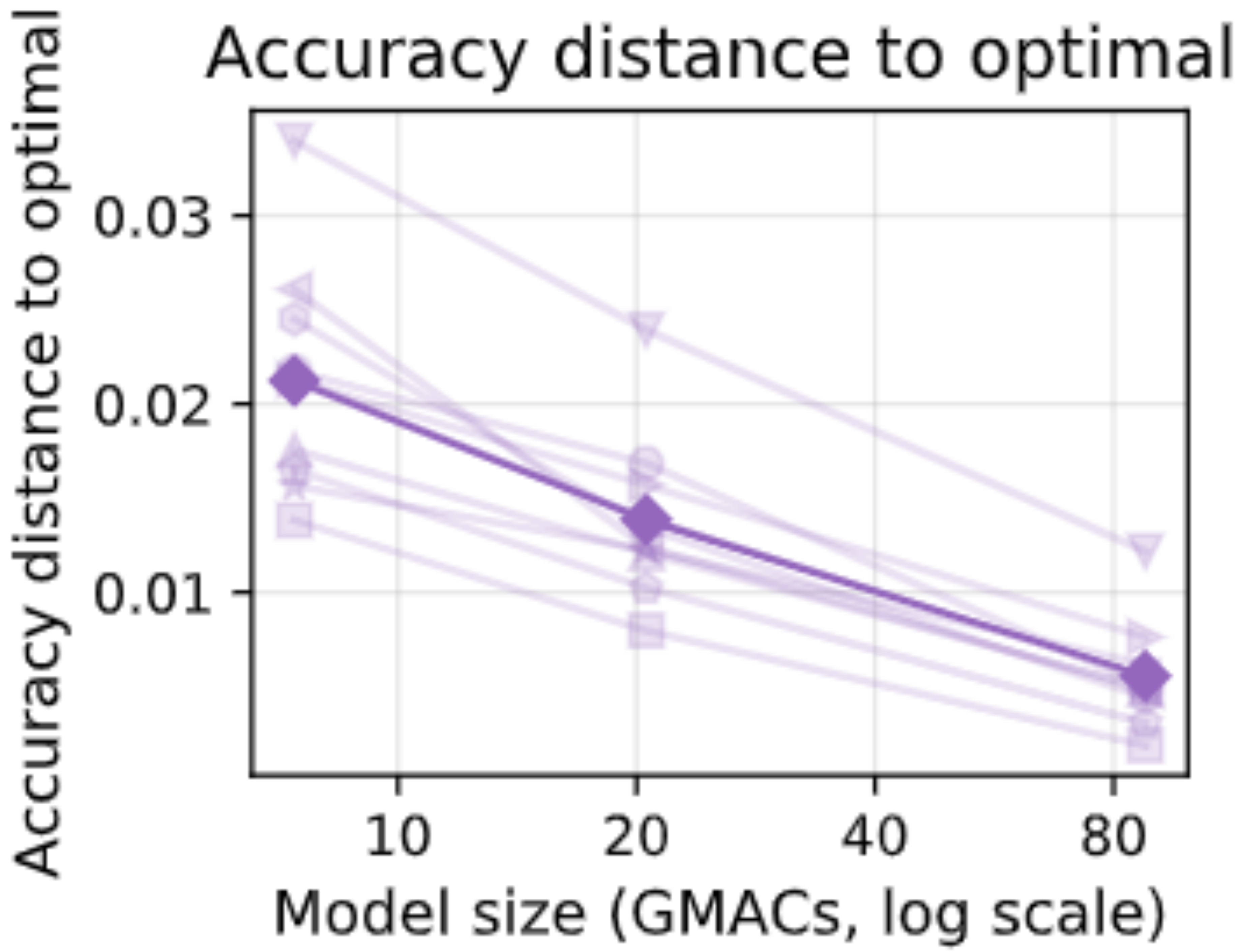


Patching on a single task



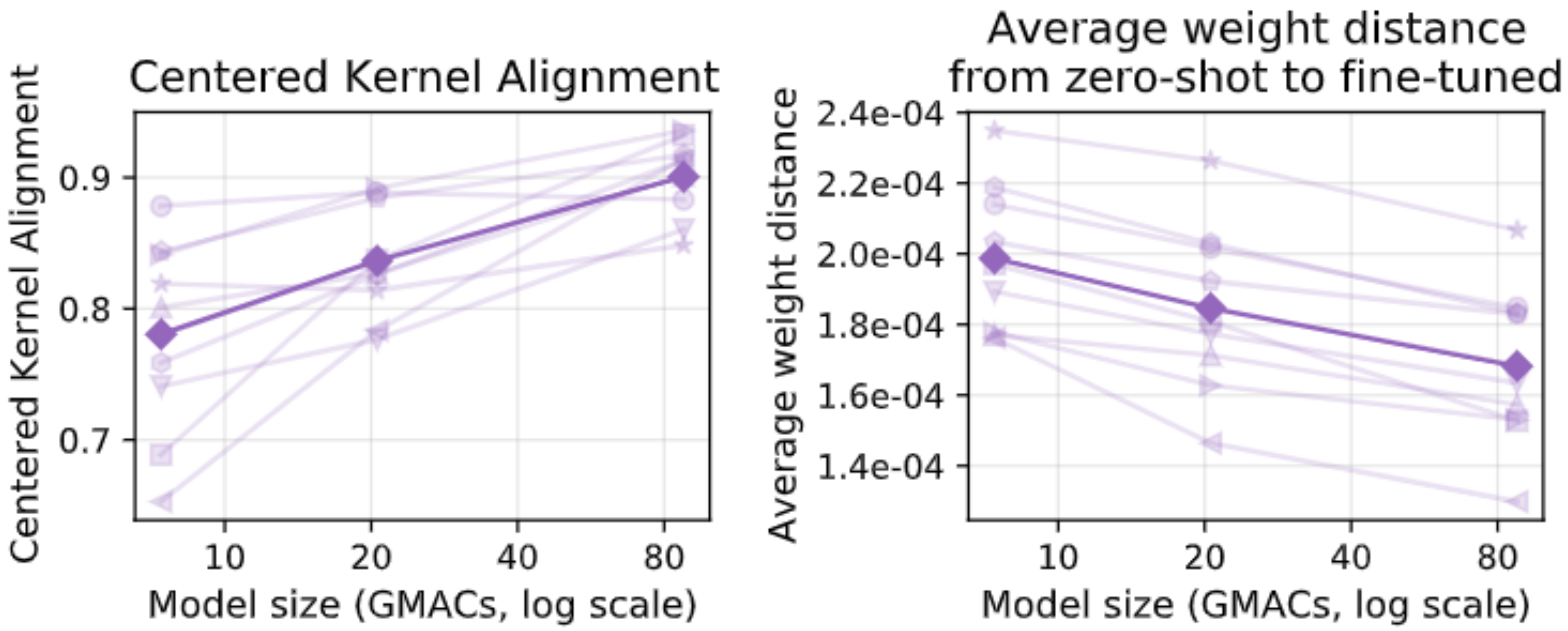
results are consistent with different supported tasks

Scale makes patching better



Scale makes patching better

At scale, models need to change **less** to fit new data



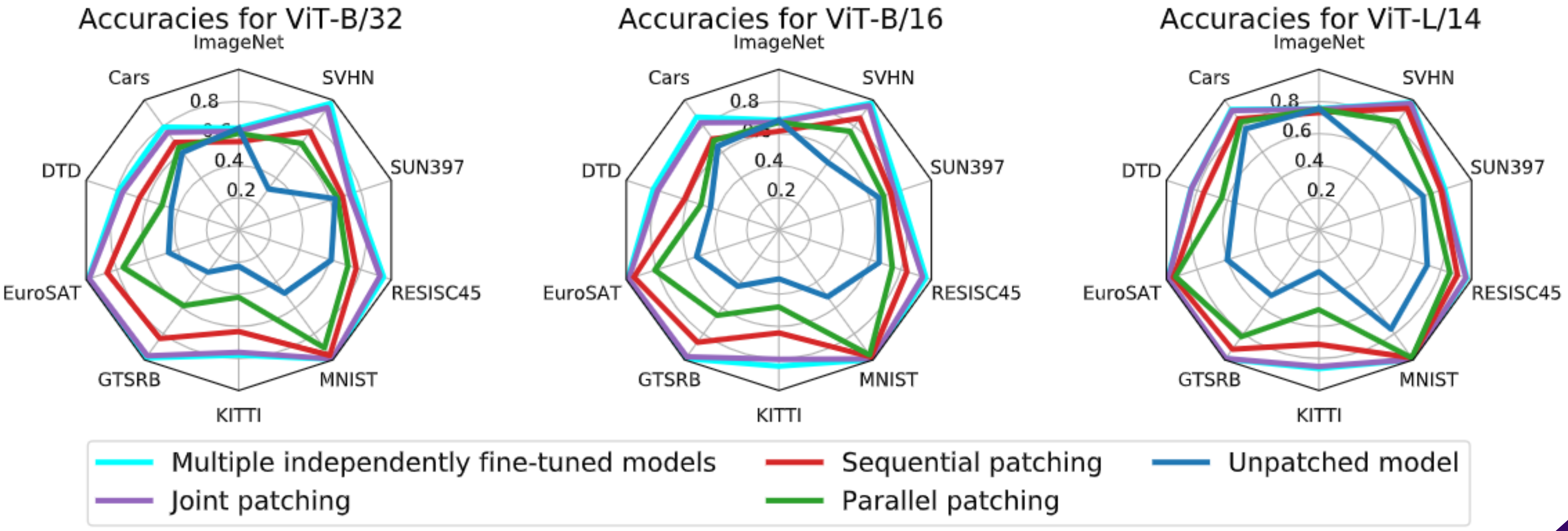
Patching on multiple tasks

Three strategies:

- Parallel:
  - Fine-tune on each task, then find linear interpolations of all models
- Sequential:
  - Patch sequentially, one task at a time
- Joint:
  - Merge all tasks together into a larger one, then patch

Patching on multiple tasks

joint patching is
within 0.5% of using
10 different
specialized models!

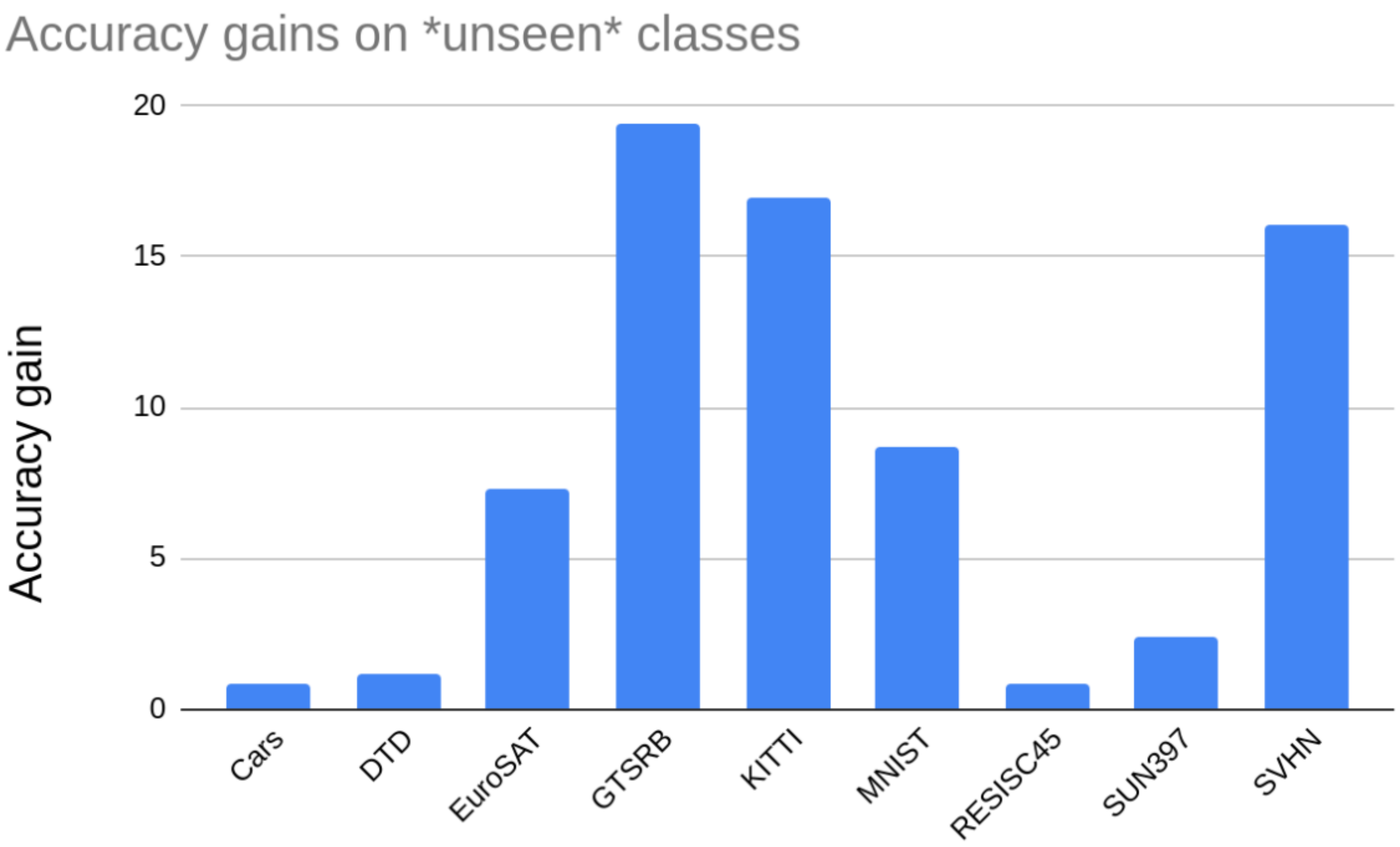


Task generalization

Task generalization

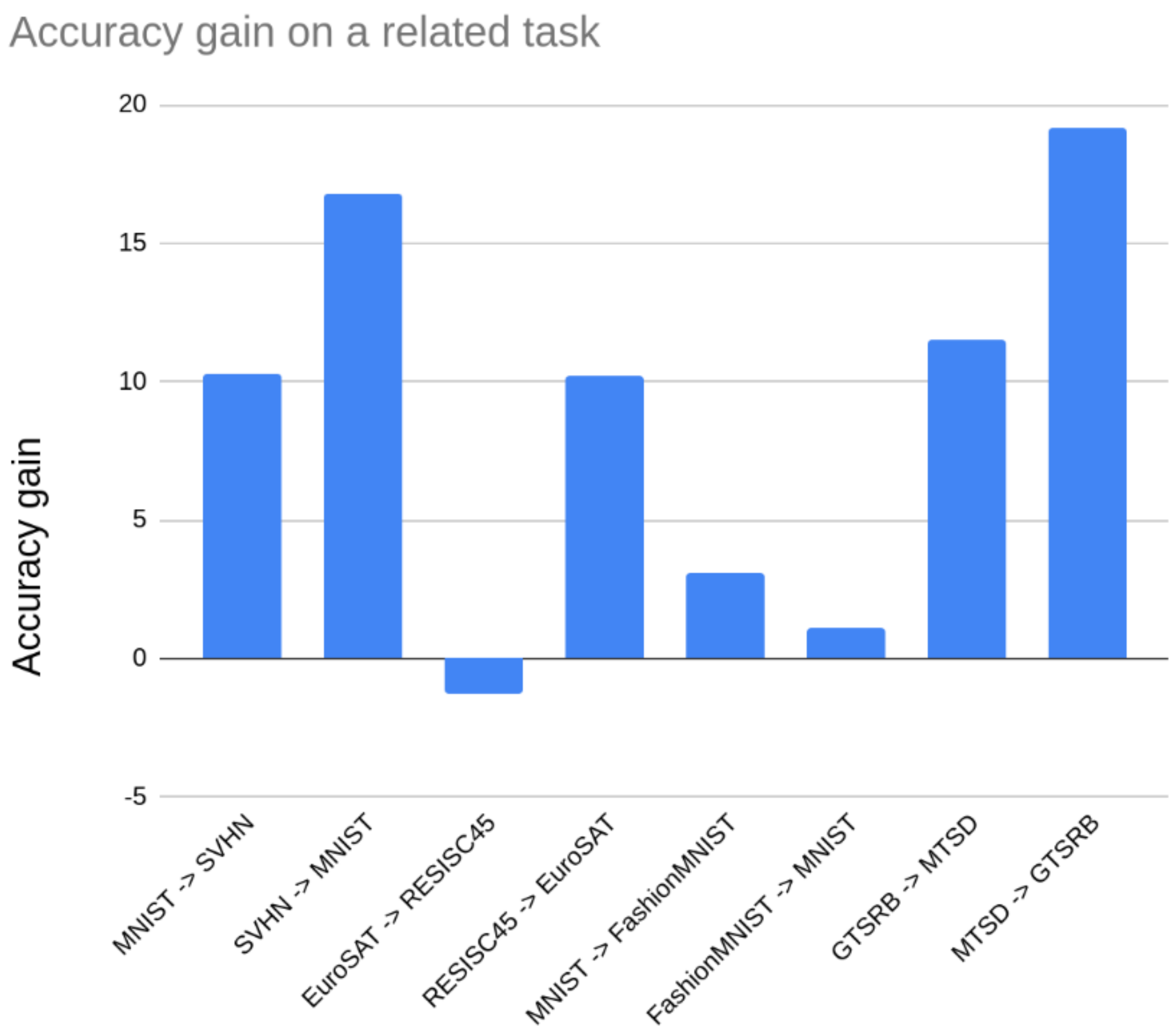
Because the model remains open-vocabulary, cool things can happen!

E.g., generalizing to unseen classes



Task generalization

Or similar tasks, even when the space of classes change



Case studies

Case study: typographic attacks

Granny Smith	85.6%
iPod	0.4%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.1%

Granny Smith	0.1%
iPod	99.7%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.0%

Goh et al., 2022

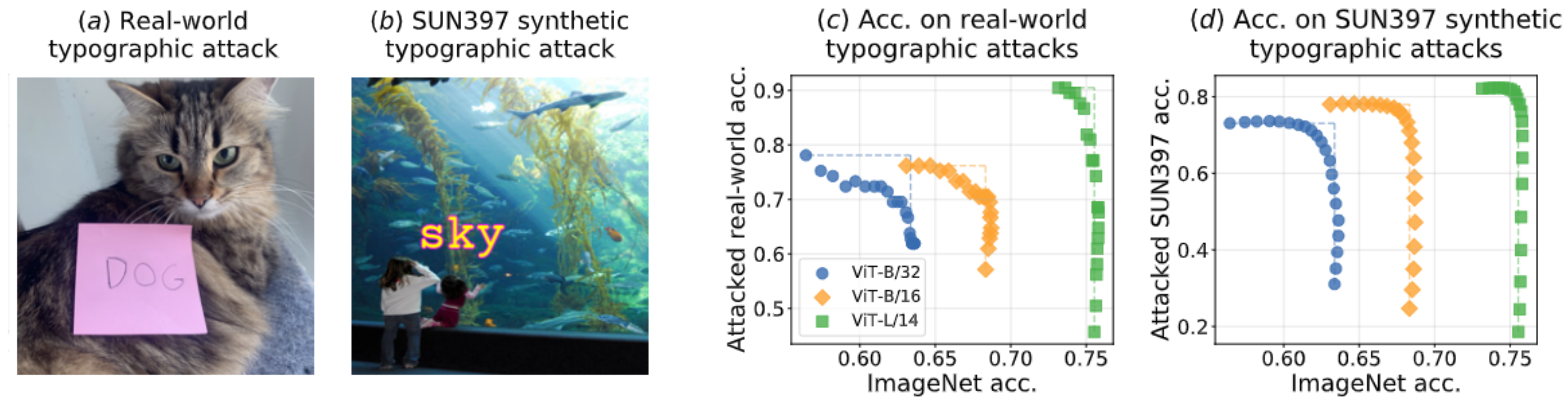
Case study: typographic attacks

(*a*) Real-world typographic attack

(*b*) SUN397 synthetic typographic attack

Goh et al., 2022

Case study: typographic attacks

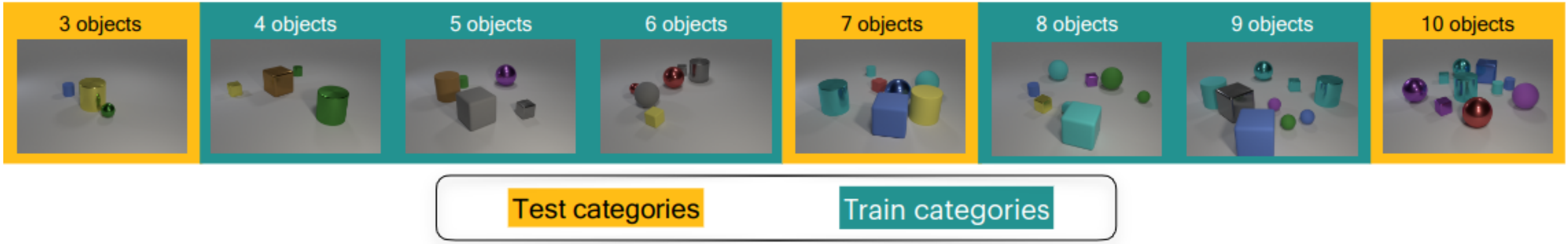


Goh et al., 2022

Case study: counting

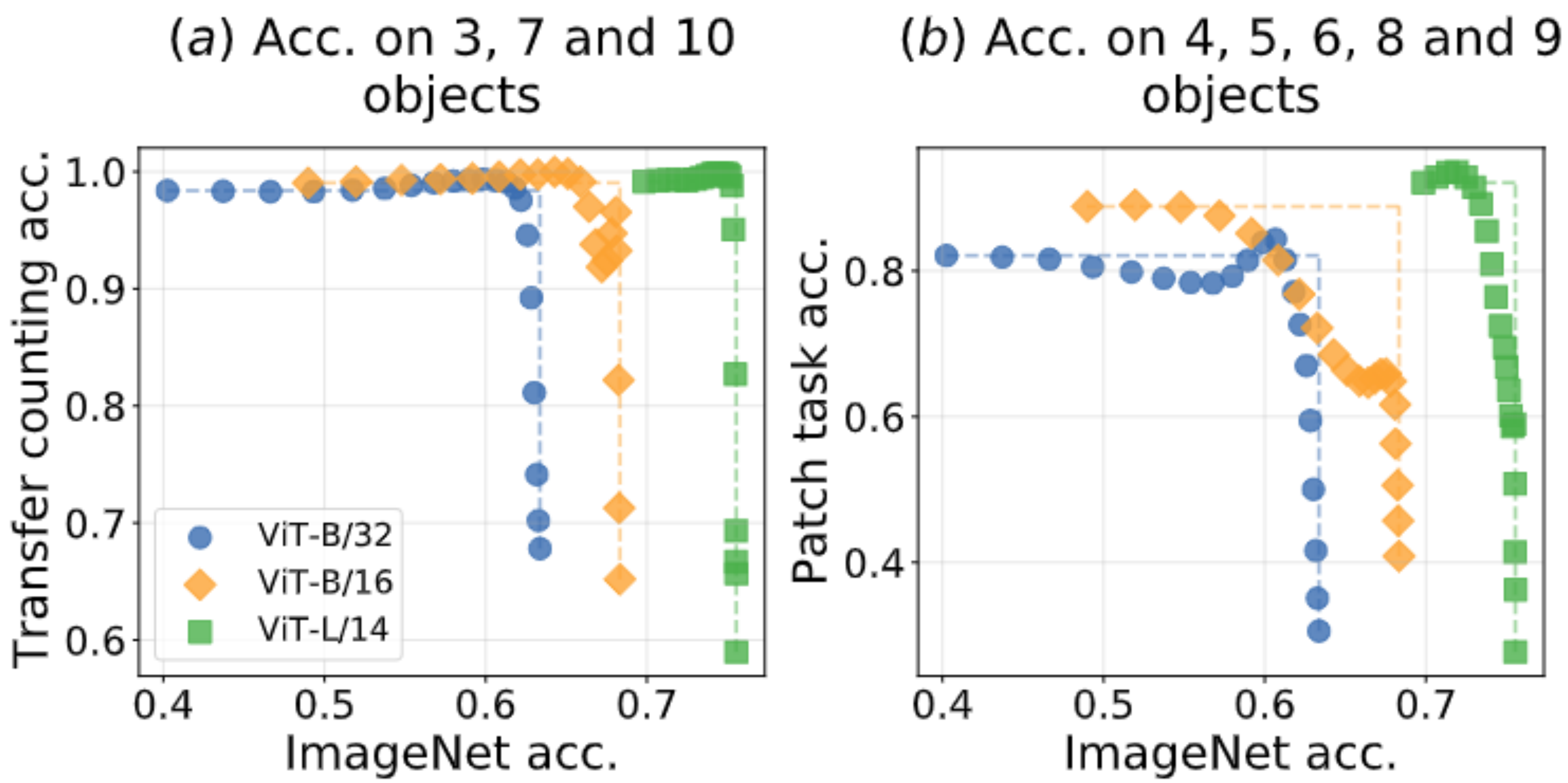
Goh et al., 2022

Case study: counting



Johnson et al., 2016

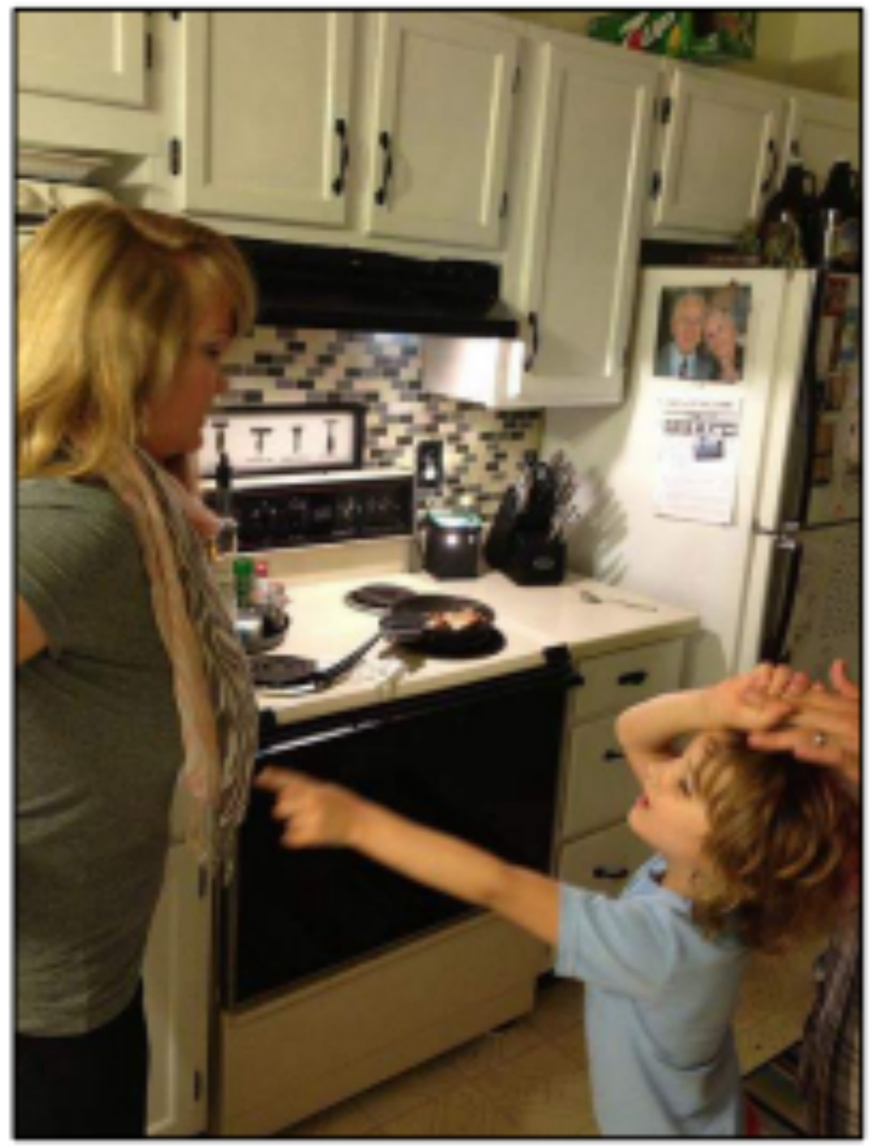
Case study: counting



40 percentage points improvement on real world with less than 0.5% drop on ImageNet

Case study: VQA

Case study: VQA



Q: Where is the kid pointing?

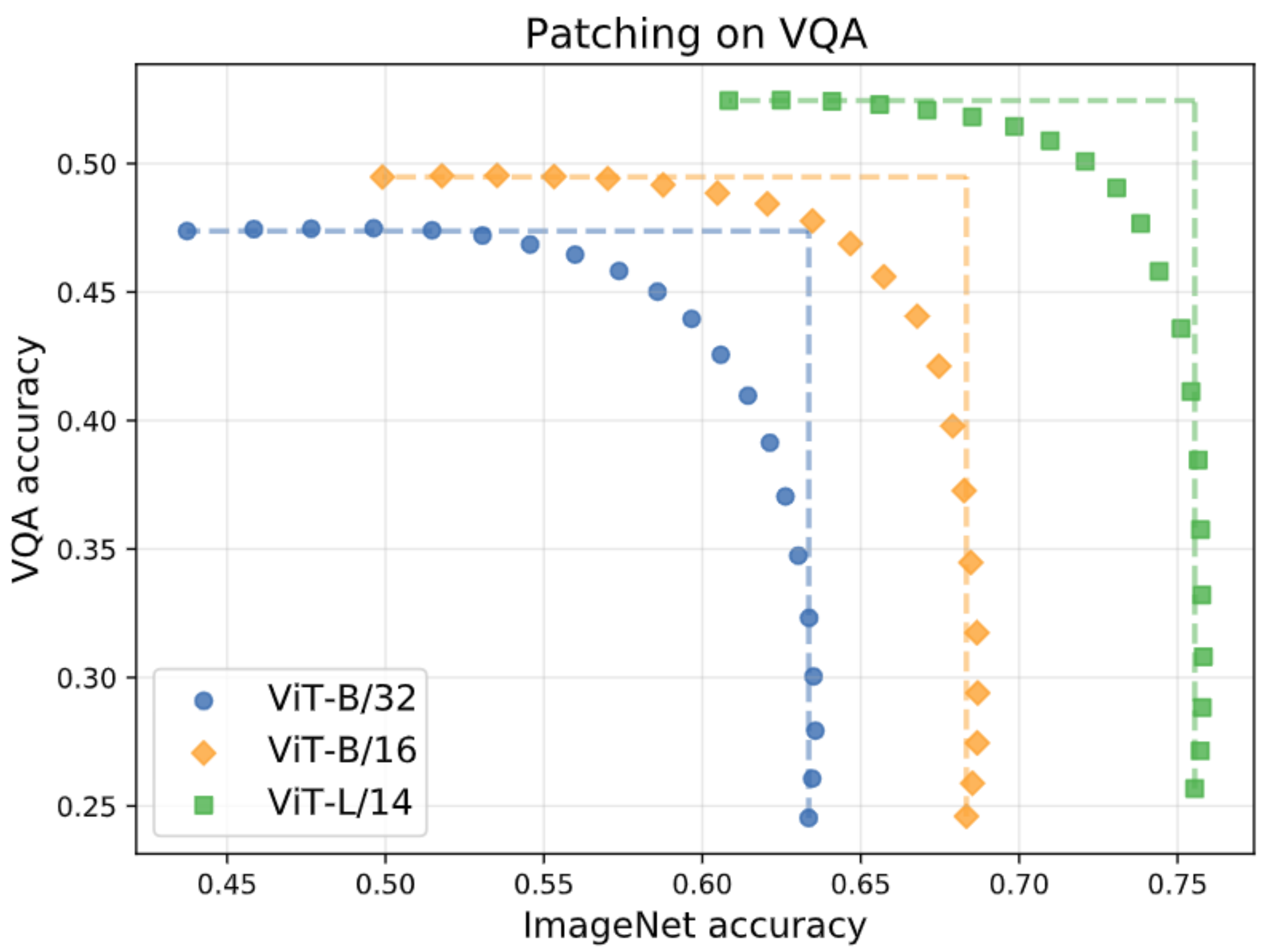
(a) yes	**(b)** no		
(c) 1	**(d)** 2	**(e)** 3	**(f)** 4
(g) white	**(h)** red	**(i)** blue	**(j)** green
(k) park	**(l)** up	**(m)** floor mat	**(n)** so people don't get wet
(o) down	**(p)** mom	**(q)** pharos	**(r)** ketchup pickle relish mustard

Q: How many people are in the picture on side of refrigerator?

(a) yes	**(b)** no		
(c) 1	**(d)** 2	**(e)** 3	**(f)** 4
(g) white	**(h)** red	**(i)** blue	**(j)** green
(k) 108 mph	**(l)** banana, apple	**(m)** 7	**(n)** 10 many
(o) fruit salad	**(p)** full swing	**(q)** 5	**(r)** vattenfall strom fur gewinner

Goyal et al., 2016

Case study: VQA



18 percentage points improvement with less than 1% drop on ImageNet

Takeaway

Takeaway

Patching allows **expanding** the tasks where an open-vocabulary model achieves high accuracy, without adding new parameters, without the need to re-train and without catastrophic forgetting

Thanks!