Efficient machine learning with resource constraints

Paolo Didier Alfano

Ph.D. Candidate

UniGe | MaLGa

Machine Learning revolution

"Any sufficiently advanced technology

is indistinguishable from magic"

Arthur C. Clarke

Machine Learning revolution

“Any sufficiently advanced technology is indistinguishable from magic”

Arthur C. Clarke

- Text-to-image generation:

An astronaut riding a horse in photorealistic style.

Machine Learning revolution

"Any sufficiently advanced technology is indistinguishable from magic"

Arthur C. Clarke

- Text-to-image generation:

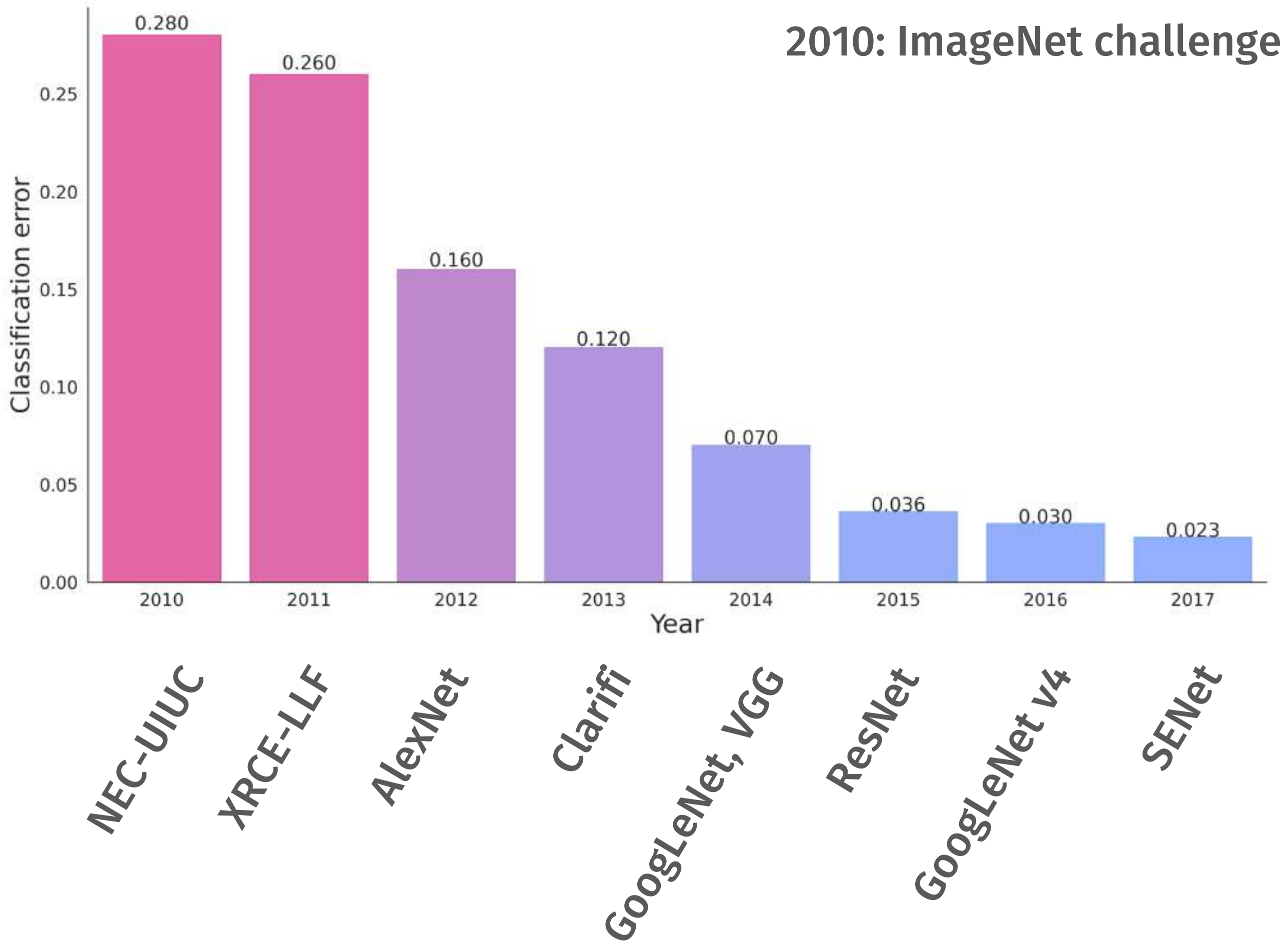
An astronaut riding a horse in photorealistic style.

- Language generation:

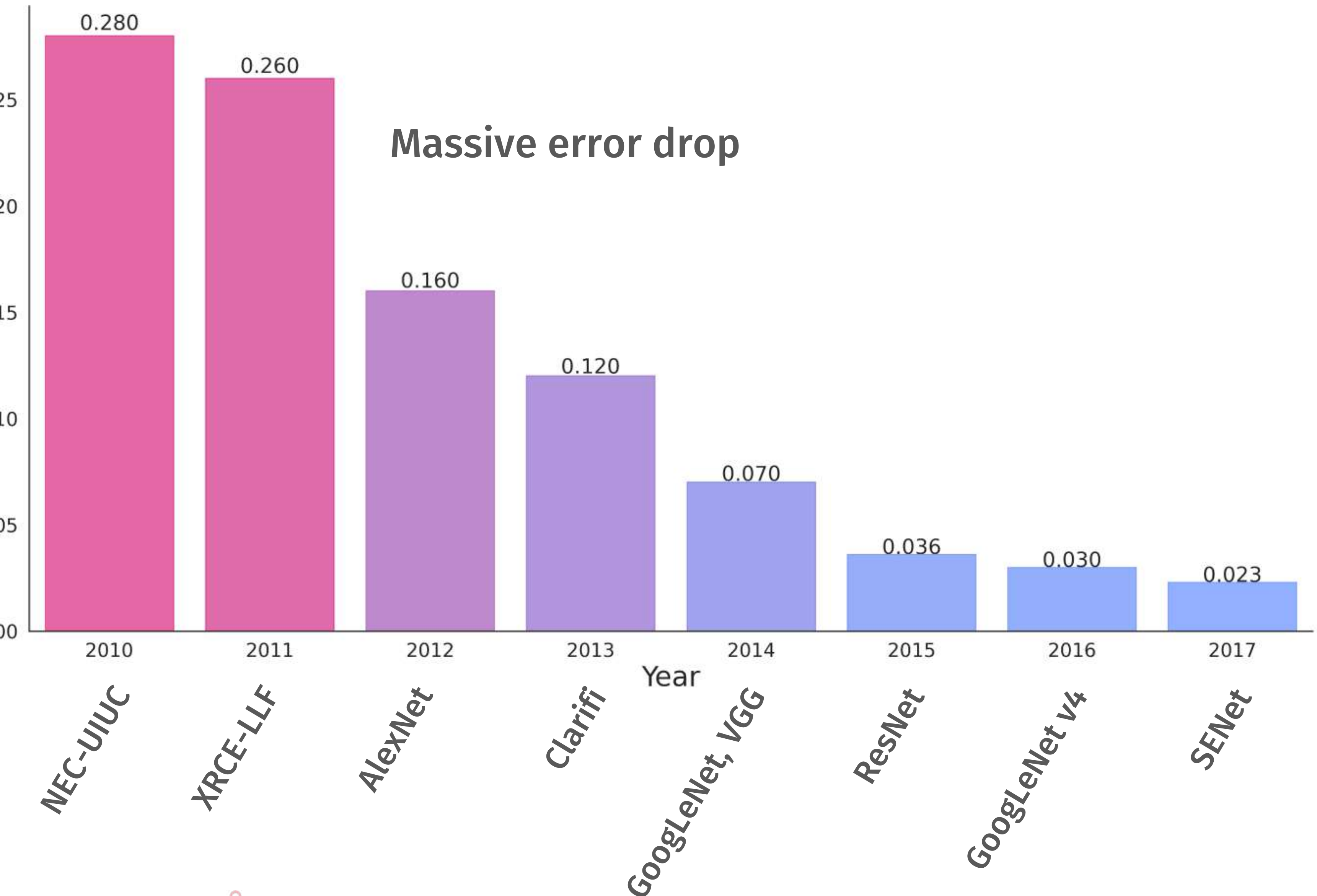
Write a dialogue by Plato where he criticizes the use of language models

```
Socrates: My friend, I am troubled by the
recent rise of these so-called autoregressive
language models.

Aristotle: What do you mean, Socrates?

Socrates: I mean that these models are being
```

Beginning the revolution



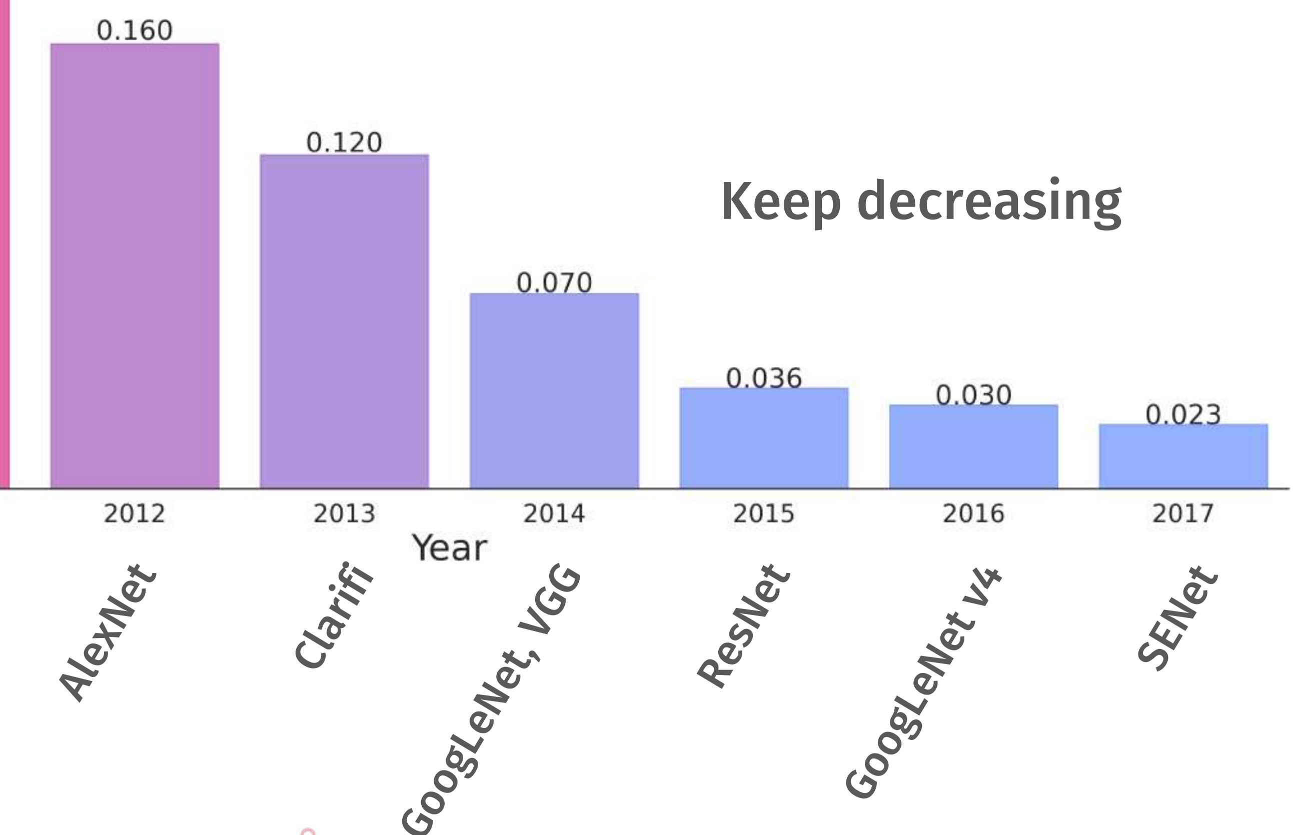
Beginning the revolution



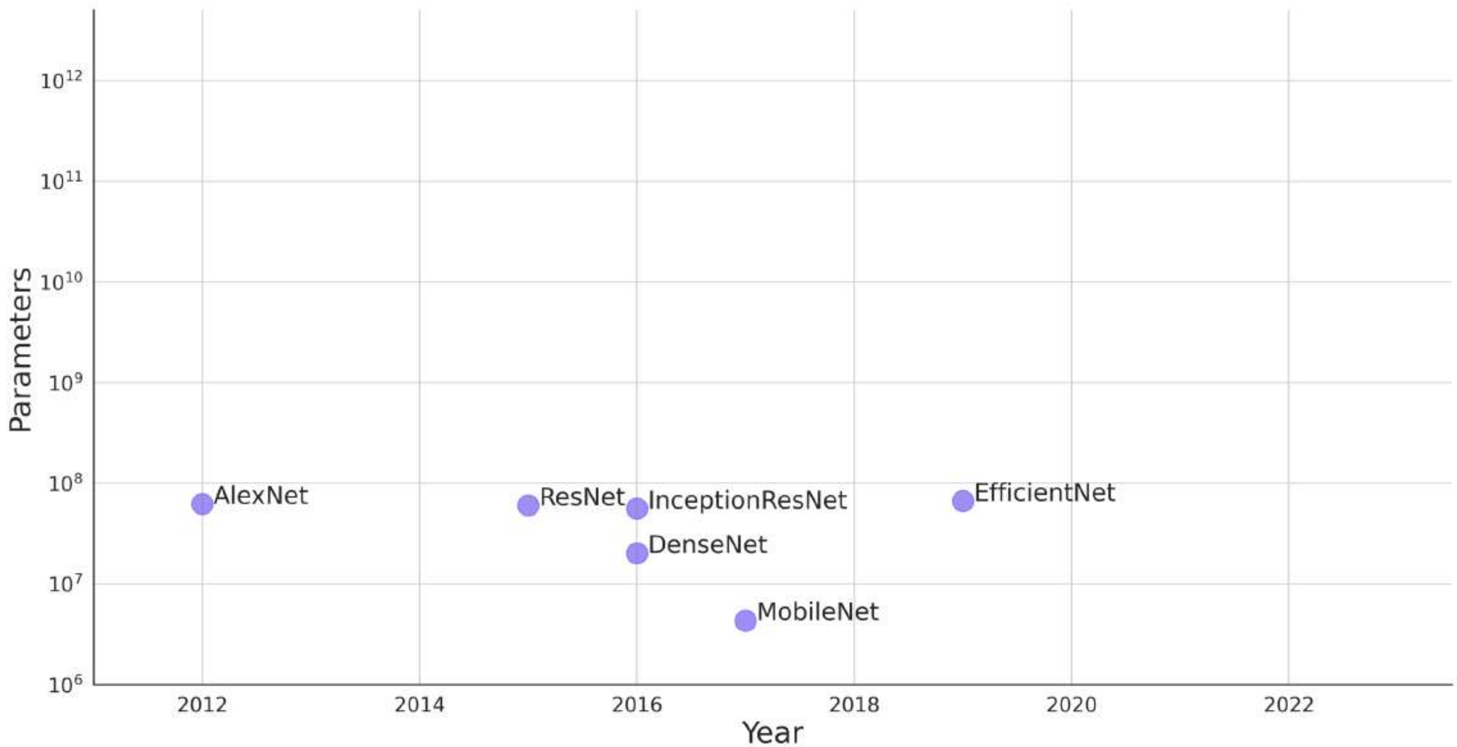
Beginning the revolution

Massive error drop

Keep decreasing

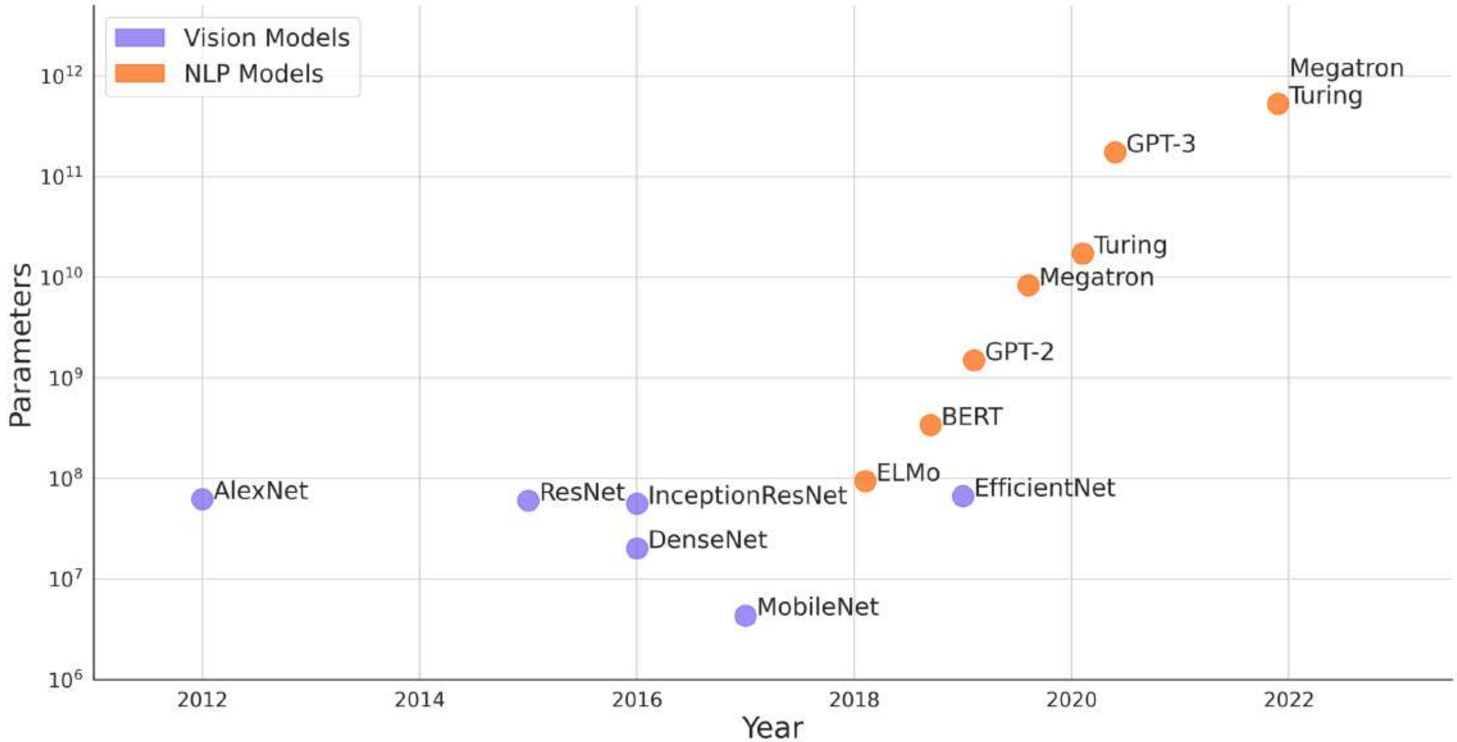


Big models

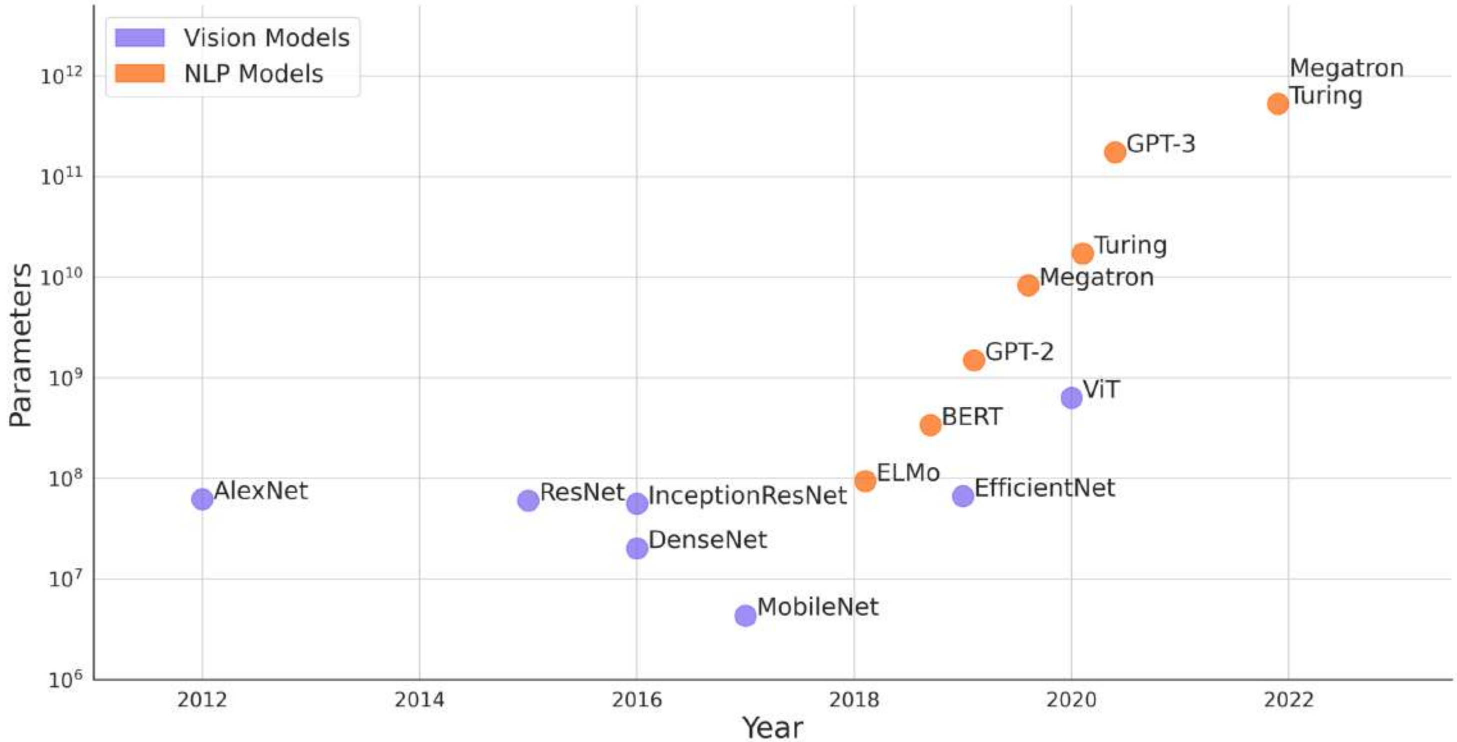


Parameters
10^{12}
10^{11}
10^{10}
10^{9}
10^{8}
10^{7}
10^{6}
AlexNet
ResNet
InceptionResNet
DenseNet
MobileNet
EfficientNet
2012
2014
2016
2018
2020
2022
Year

Big models

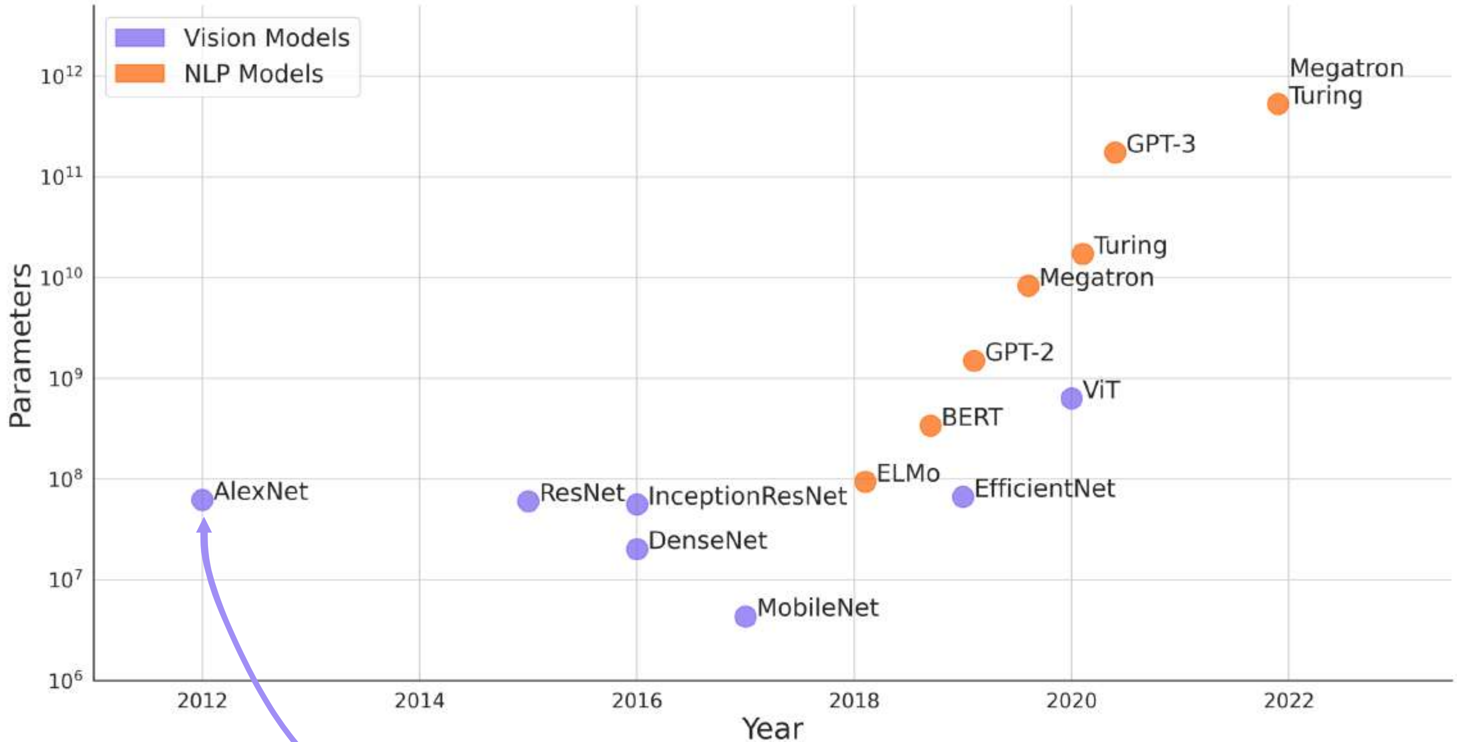

Vision Models
NLP Models
Parameters
Year
10^{12}
10^{11}
10^{10}
10^{9}
10^{8}
10^{7}
10^{6}
2012
2014
2016
2018
2020
2022
AlexNet
ResNet
InceptionResNet
DenseNet
MobileNet
ELMo
EfficientNet
BERT
GPT-2
Megatron
Turing
GPT-3
Megatron
Turing

Big models



Vision Models
NLP Models
Parameters
10^{12}
10^{11}
10^{10}
10^{9}
10^{8}
10^{7}
10^{6}
Year
2012
2014
2016
2018
2020
2022
AlexNet
ResNet
InceptionResNet
DenseNet
MobileNet
ELMo
EfficientNet
BERT
ViT
GPT-2
Megatron
Turing
GPT-3
Megatron Turing

Big models

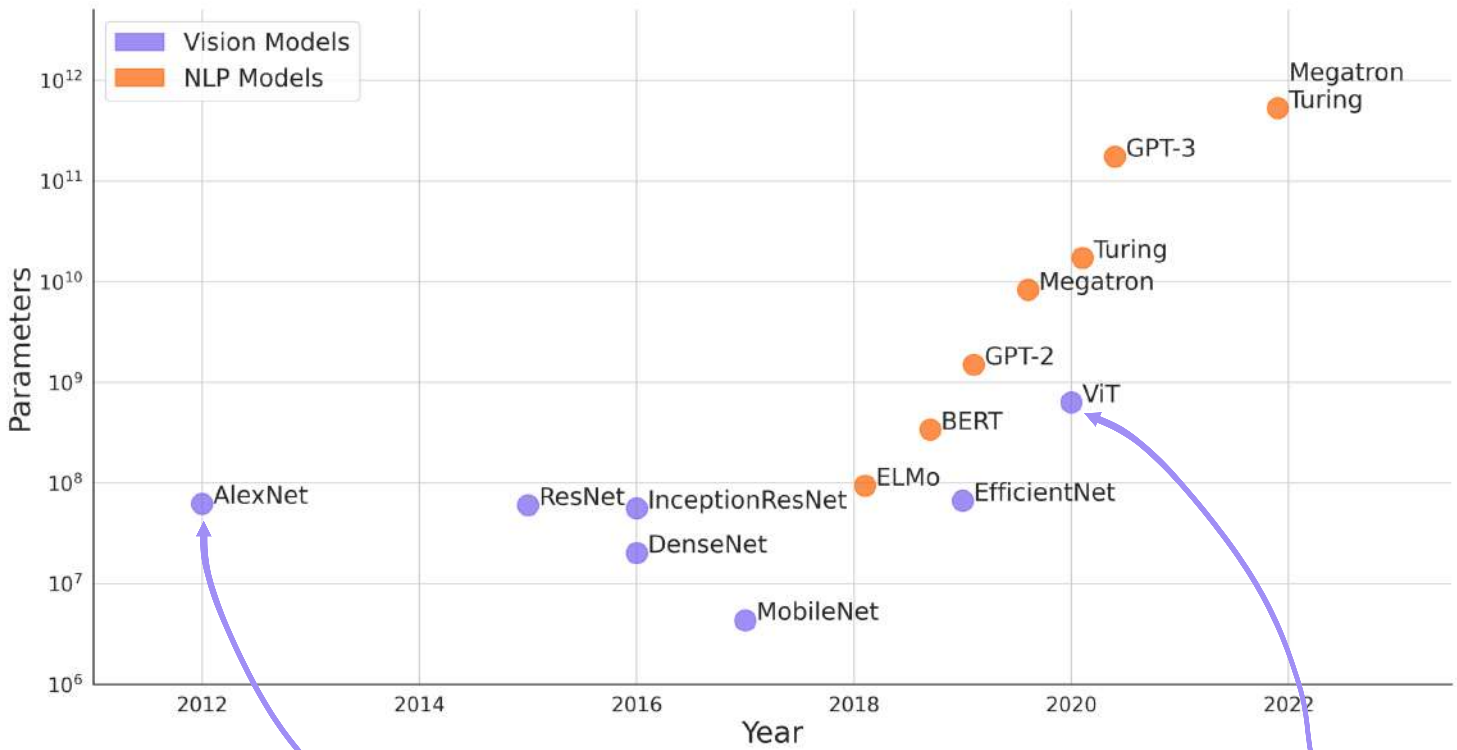


12 GPU days

GTX 580 3GB

[Krizhevsky et al. 2012]

Big models



12 GPU days

GTX 580 3GB

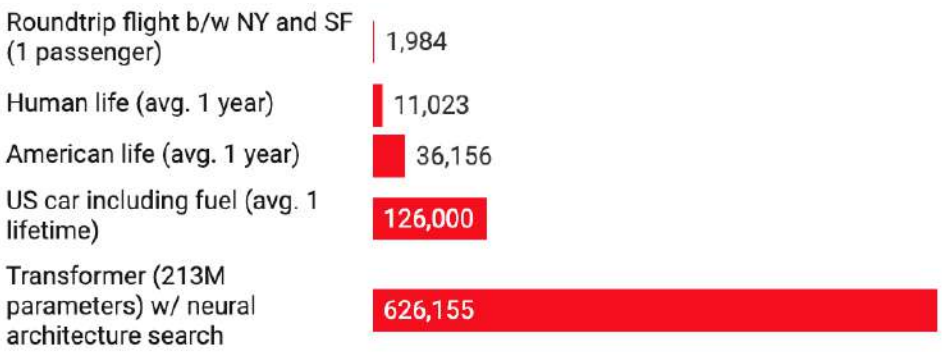
[Krizhevsky et al. 2012]

2500 TPUv3 days

[Dosovitskiy et al. 2020]

What about costs?

What about costs?



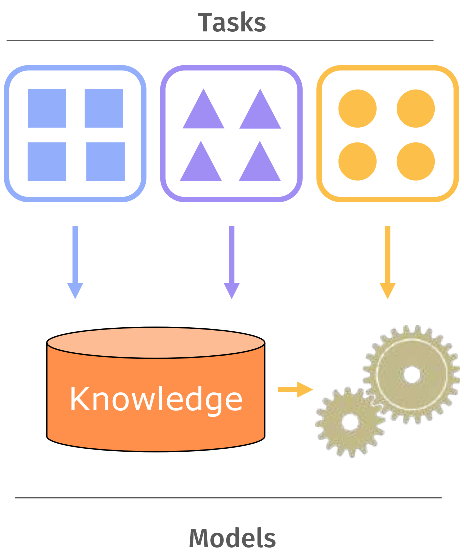
[Strubell et al. 2019]

Tackle costs

Tackle costs

Training time efficiency:

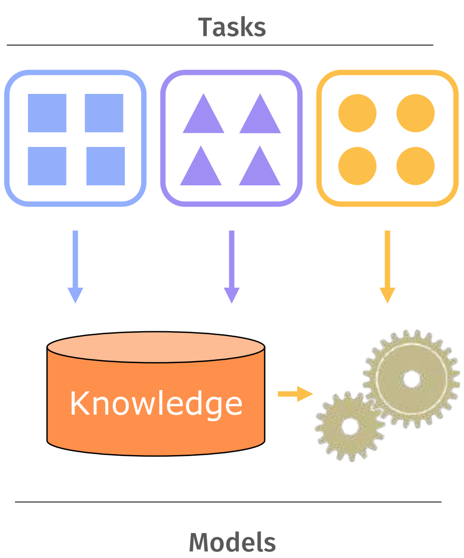
Transfer Learning



Tackle costs

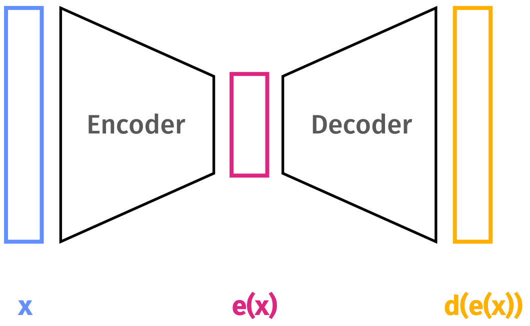
Training time efficiency:

Transfer Learning



Representation efficiency:

Dimensionality reduction



Outline

Introduction

Training time efficiency

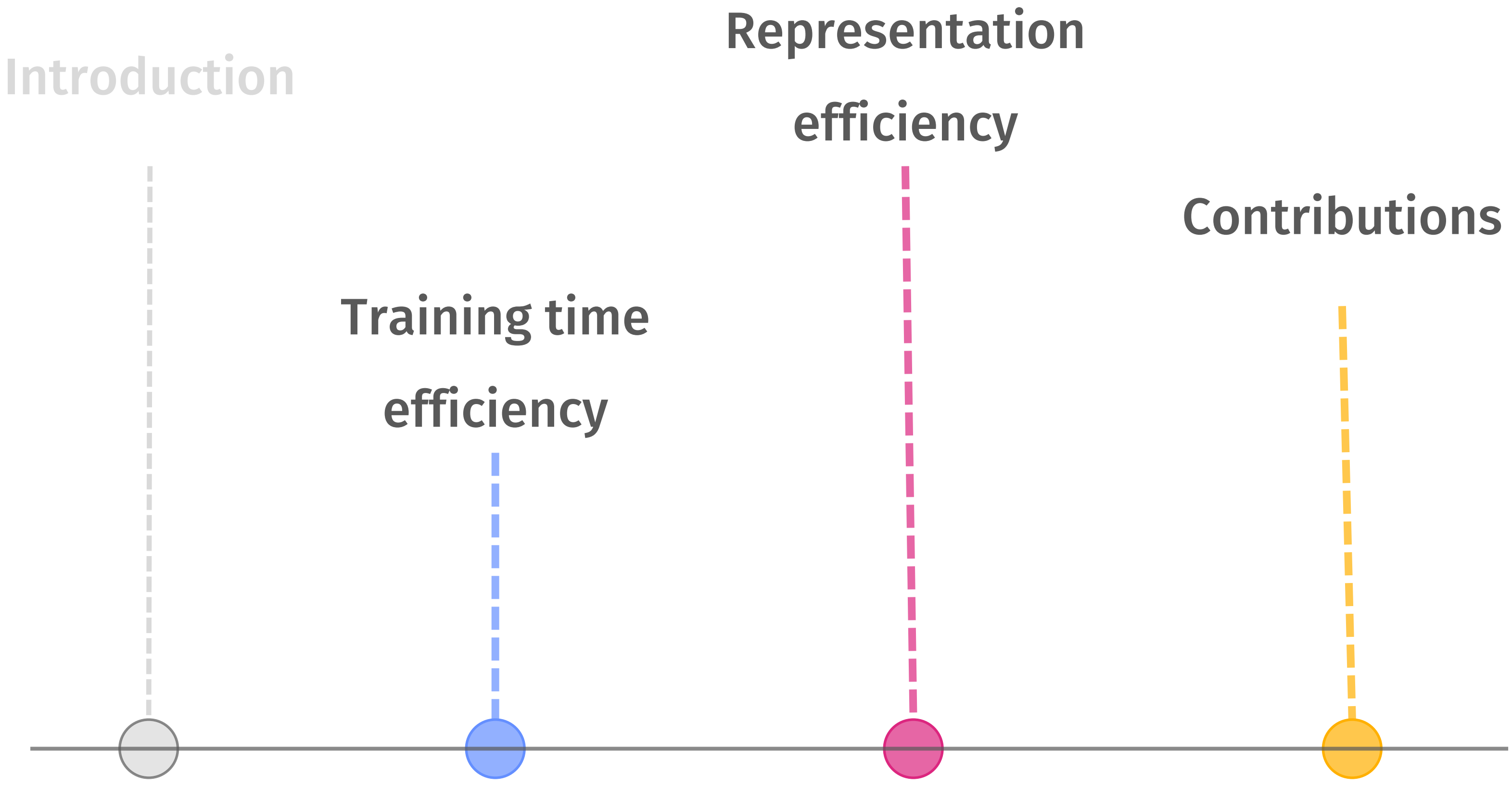
Representation efficiency

Datasets similarity

Contributions

UniGe | MaLGa

Outline



Introduction
Training time efficiency
Representation efficiency
Contributions

Training time efficiency

Fine-tuning or top-tuning? A study on transfer learning with image pre-trained features and fast kernel methods, Alfano, Pastore, Rosasco, Odone
Submitted @IMAVIS Journal

Supervised learning

[Russel and Norvig 2020]

Data:

$$X = \{x_1, \ldots, x_n\}$$

$$Y = \{y_1, \ldots, y_n\}$$

Supervised learning

[Russel and Norvig 2020]

Data:

$$X = \{x_1, \ldots, x_n\}$$

$$Y = \{y_1, \ldots, y_n\}$$

Domain:

$$\mathcal{D} = \{X, Y\}$$

Supervised learning

[Russel and Norvig 2020]

Data:

$$X = \{x_1, \ldots, x_n\}$$

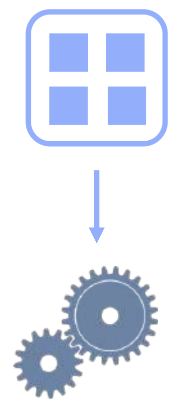
$$Y = \{y_1, \ldots, y_n\}$$

Domain:

$$\mathcal{D} = \{X, Y\}$$

Predictive function:

$$f : \mathcal{X} \rightarrow \mathcal{Y}$$



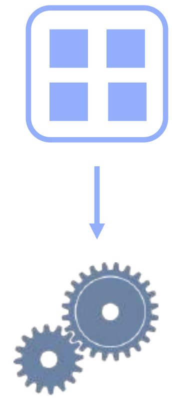
Models

Transfer learning

[Zhuang et al 2021]

Domain: $$\mathcal{D} = \{X, Y\}$$

Predictive function: $$f : \mathcal{X} \rightarrow \mathcal{Y}$$



Models

Transfer learning

[Zhuang et al 2021]

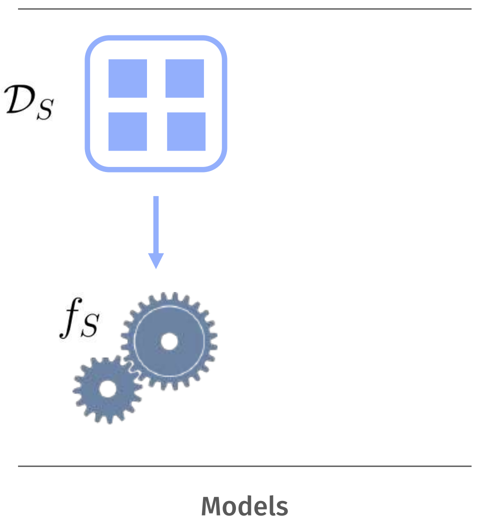
Domain: $\mathcal{D} = \{X, Y\}$

Predictive function: $f : \mathcal{X} \to \mathcal{Y}$

Source (big):

$\mathcal{D}_S$

f_S



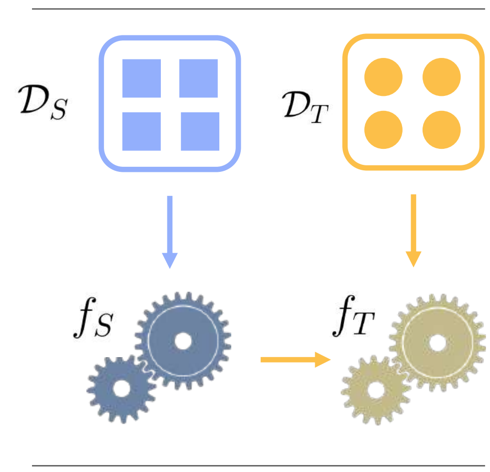
Transfer learning

[Zhuang et al 2021]

Domain: $\mathcal{D} = \{X, Y\}$

Predictive function: $f : \mathcal{X} \to \mathcal{Y}$

Source (big):	Target (small):
$\mathcal{D}_S$	$\mathcal{D}_T$
f_S	f_T **?**



Models

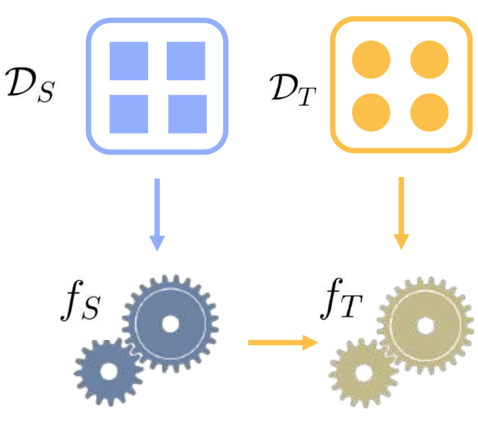
Transfer learning

[Zhuang et al 2021]

Domain: $\mathcal{D} = \{X, Y\}$

Predictive function: $f : \mathcal{X} \to \mathcal{Y}$

Source (big):	Target (small):
$\mathcal{D}_S$	$\mathcal{D}_T$
f_S	f_T **?**



Models

Can we exploit f_S ?

[Garcia-Gasulla et al 2018]

[Kornblith et al 2018]

ImageNet (ILSVRC)

[Russakovsky et al 2015]

1.3	million labeled images
1.000	different labels

ImageNet (ILSVRC)

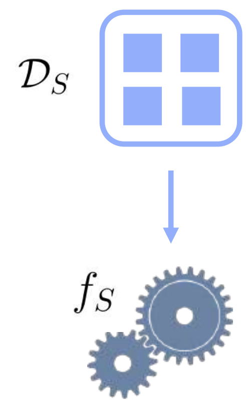
[Russakovsky et al 2015]

1.3 million labeled images

1.000 different labels

ImageNet as source domain

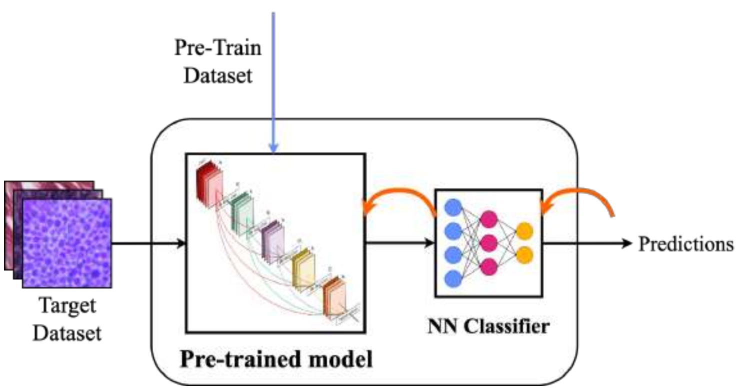
Best models adapted to it



Fine-Tuning vs Top-tuning

Fine-Tuning vs Top-tuning

1) Fine Tuning

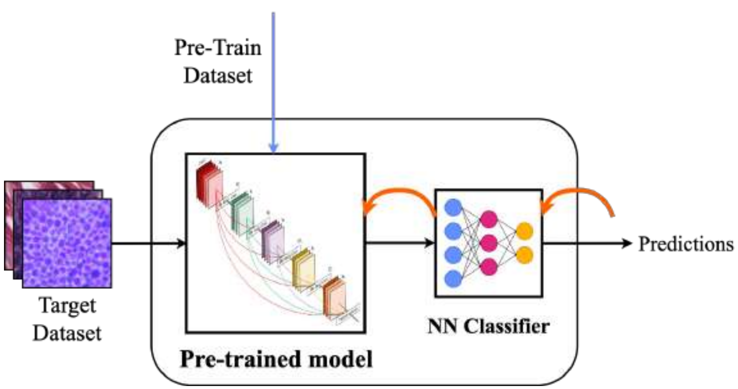


[Goodfellow et al 2016]

Fine-Tuning vs Top-tuning

1) Fine Tuning

$$\Phi_{FT} = \qquad\qquad \circ \underbrace{\Phi_C \circ \ldots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$

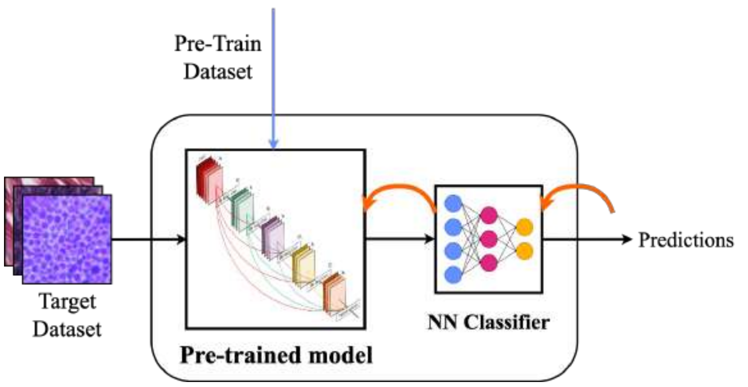


[Goodfellow et al 2016]

Fine-Tuning vs Top-tuning

1) Fine Tuning

$$\Phi_{FT} = \underbrace{\Phi_{C+L} \circ \ldots \Phi_{C+1}}_{\text{Fully connected layers}} \circ \underbrace{\Phi_C \circ \ldots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$

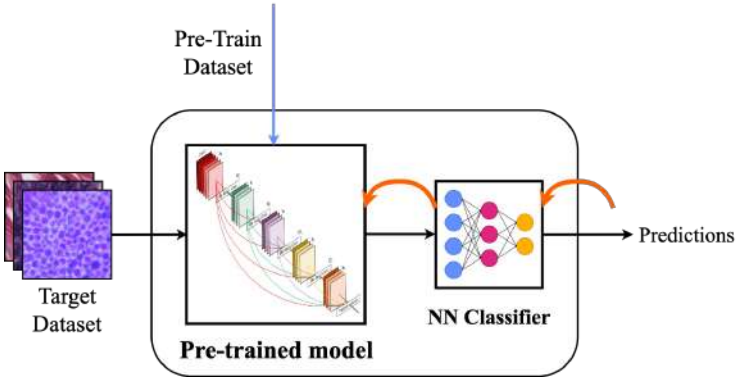


[Goodfellow et al 2016]

Fine-Tuning vs Top-tuning

1) Fine Tuning

$$\Phi_{FT} = \underbrace{\Phi_{C+L} \circ \dots \Phi_{C+1}}_{\text{Fully connected layers}} \circ \underbrace{\Phi_C \circ \dots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$



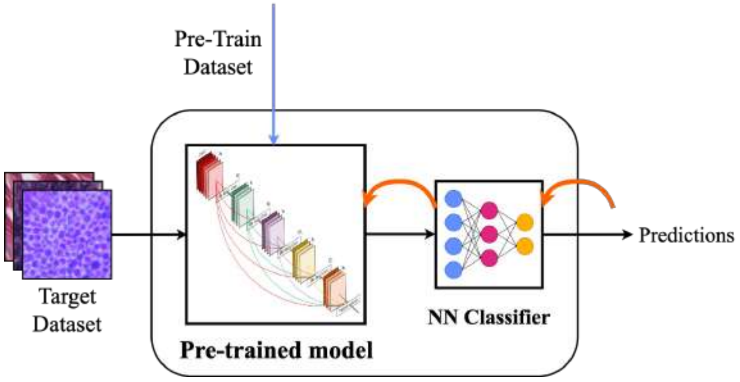
[Goodfellow et al 2016]

- All parameters updated

Fine-Tuning vs Top-tuning

1) Fine Tuning

$$\Phi_{FT} = \underbrace{\Phi_{C+L} \circ \ldots \Phi_{C+1}}_{\text{Fully connected layers}} \circ \underbrace{\Phi_C \circ \ldots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$



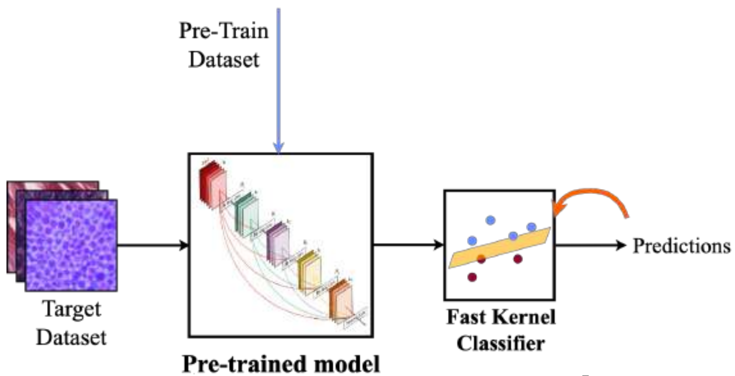
[Goodfellow et al 2016]

- All parameters updated

- Adaptive

Fine-Tuning vs Top-tuning

2) Top-Tuning

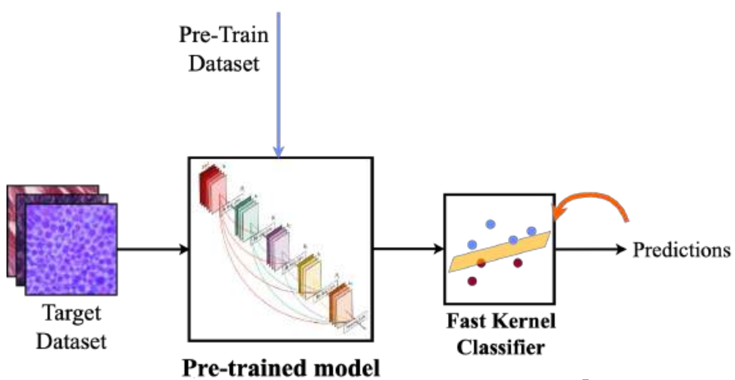


[Meanti et al 2020]

Fine-Tuning vs Top-tuning

2) Top-Tuning

$$\Phi_{TT} = \underbrace{\Phi_C \circ \ldots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$

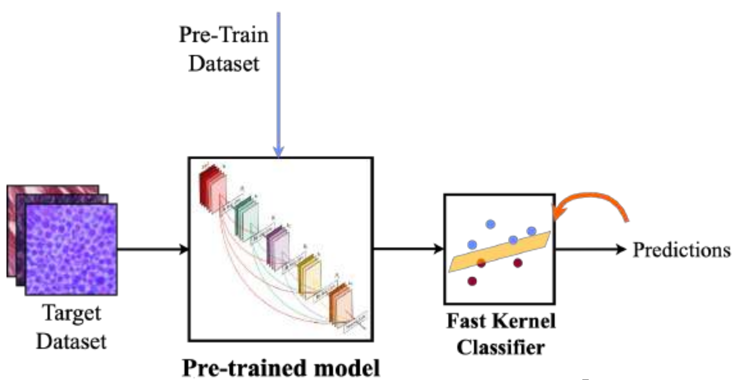


[Meanti et al 2020]

Fine-Tuning vs Top-tuning

2) Top-Tuning

$$\Phi_{TT} = \underbrace{\Psi}_{\text{Kernel feature map}} \circ \underbrace{\Phi_C \circ \ldots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$



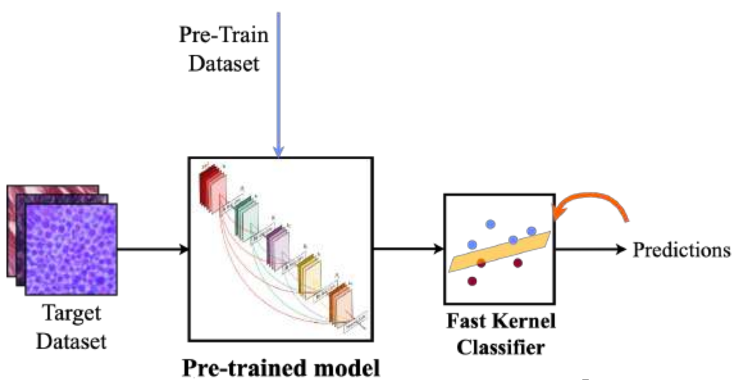
[Meanti et al 2020]

- Only Fast Kernel updated

Fine-Tuning vs Top-tuning

2) Top-Tuning

$$\Phi_{TT} = \underbrace{\Psi}_{\text{Kernel feature map}} \circ \underbrace{\Phi_C \circ \ldots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$



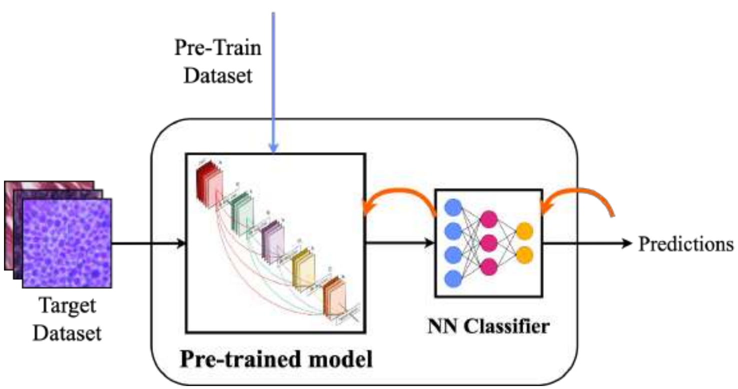
[Meanti et al 2020]

- Only Fast Kernel updated
- Faster

Fine-Tuning vs Top-tuning

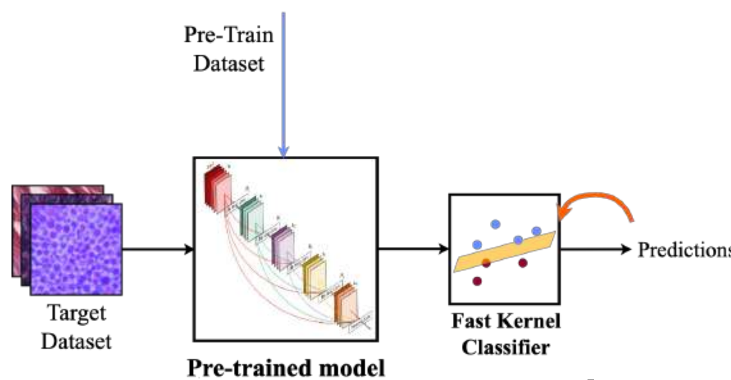
1) Fine Tuning

$$\Phi_{FT} = \underbrace{\Phi_{C+L} \circ \ldots \Phi_{C+1}}_{\text{Fully connected layers}} \circ \underbrace{\Phi_C \circ \ldots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$



2) Top-Tuning

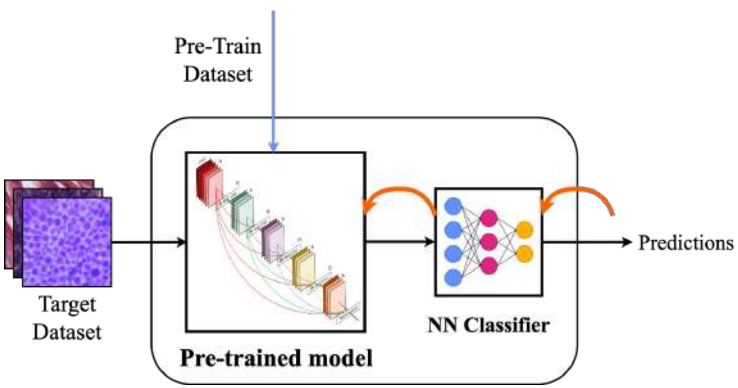
$$\Phi_{TT} = \underbrace{\Psi}_{\text{Kernel feature map}} \circ \underbrace{\Phi_C \circ \ldots \circ \Phi_1(x)}_{\text{Convolutional layers}}$$



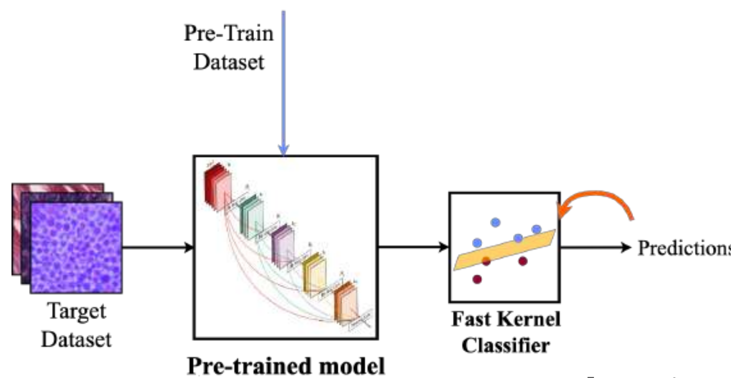
[Meanti et al 2020]

Fine-Tuning vs Top-tuning

1) Fine Tuning



2) Top-Tuning



[Meanti et al 2020]

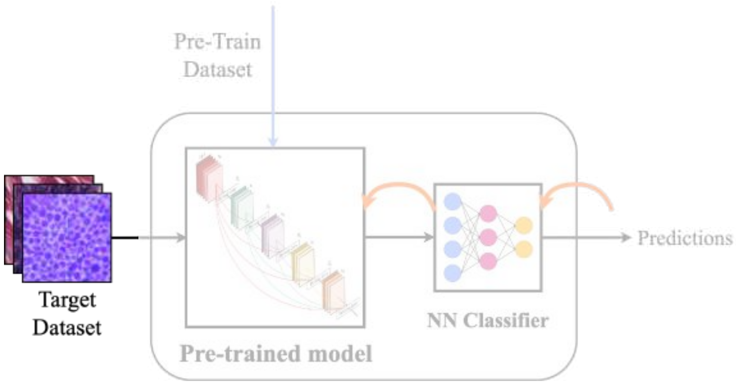
Accuracy

Training time

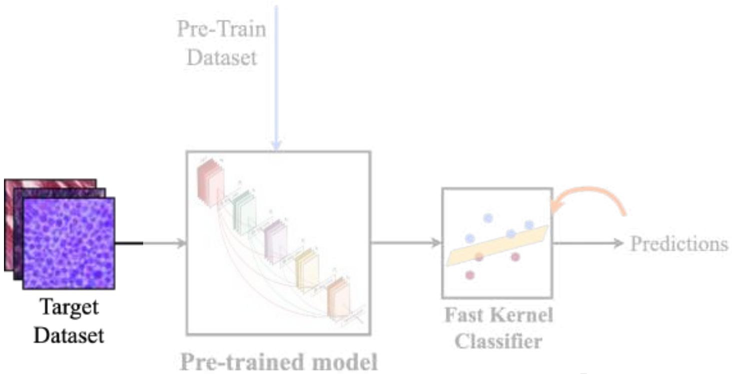
Best model?

Target dataset

1) Fine Tuning



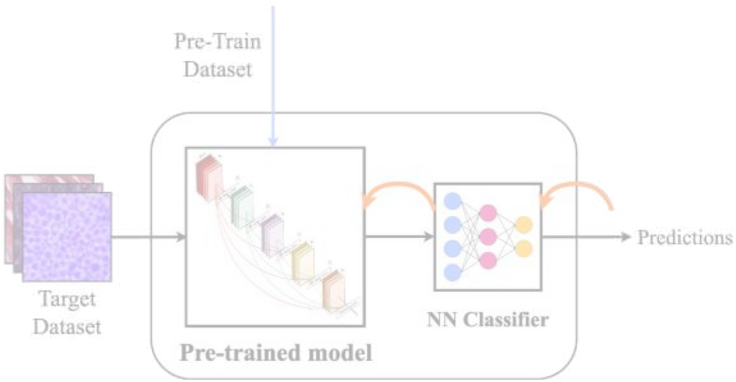
2) Top-Tuning



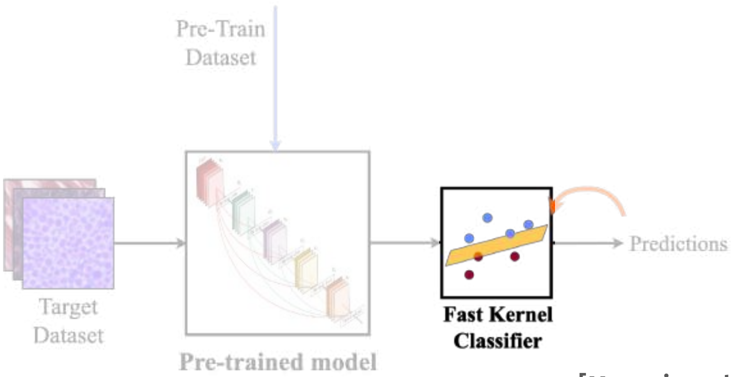
[Meanti et al 2020]

Classifier

1) Fine Tuning



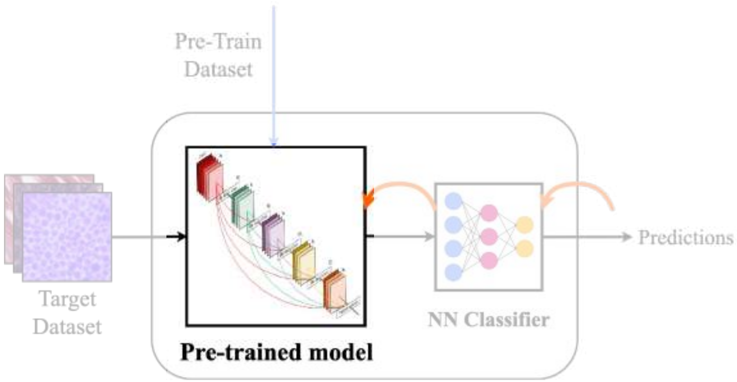
2) Top-Tuning



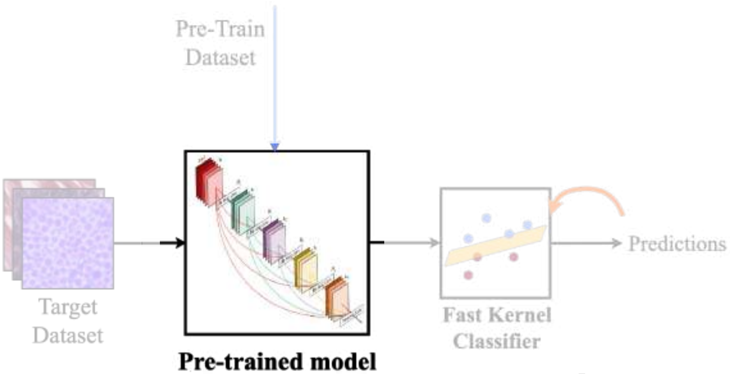
[Meanti et al 2020]

Pre-trained model

1) Fine Tuning



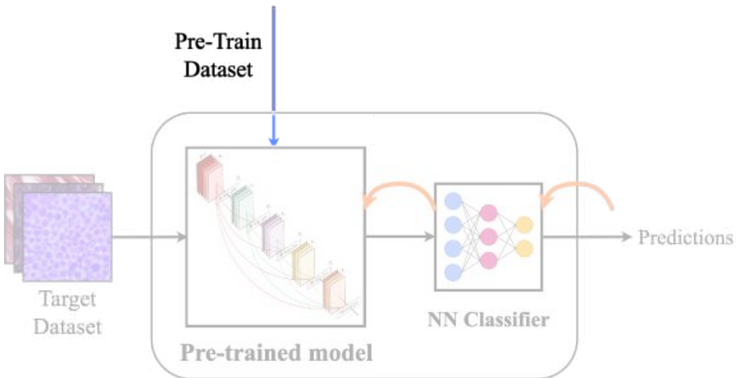
2) Top-Tuning



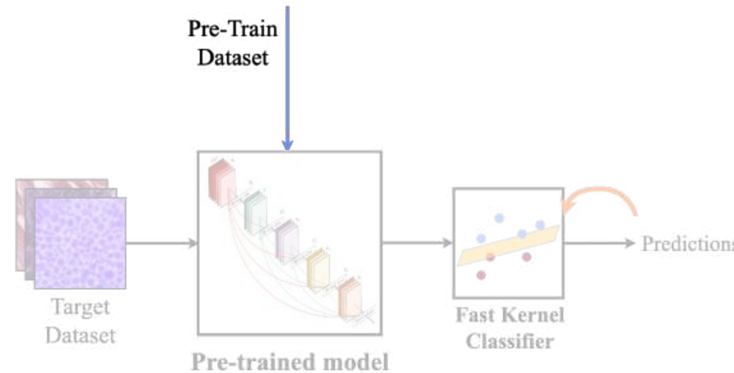
[Meanti et al 2020]

Pre-train source

1) Fine Tuning



2) Top-Tuning



[Meanti et al 2020]

Target datasets

32 Target datasets

Small to medium size

Dataset name	#images (Tr/Te)	Img. size mean	#classes
AFHQ (AF)[58]	13.167/1.463	512×512	3
Beans (BE)[59]	1.167/128	500×500	3
Best artworks (BA)[60]	7.896/878	980×921	50
Boat types (BT)[61]	1.315/147	905×1234	9
Caltech-101 (C101)[62]	3.060/6.084	251×282	102
Cassava (CSV)[63]	7.545/1.885	573×611	5
Cats vs Dogs (CVSD) [64]	20.935/2.327	365×410	2
Chest xray (CXRAY) [65]	4.708/524	968×1321	2
CIFAR10 (CIF10) [66]	50.000/10.000	32×32	10
CIFAR100 (CIF100) [66]	50.000/10.000	32×32	100
Citrus leaves (CLV) [67]	534/60	256×256	4
Colorectal hist (COL) [68]	4.500/500	150×150	8
Deep weeds (DW) [69]	15.758/1.751	256×256	9
DTD (DTD)[70]	3.760/1.880	453×500	47
EuroSAT (ES) [71]	24.300/2.700	64×64	10
FGVC Aircraft (AIR)[72]	6.667/3.333	353×1056	100
Footb vs Rugby (FVSR) [73]	2.203/245	618×788	2
Gemstones (GEM) [74]	2.571/286	330×335	87
Hors or Hum (HVSH) [75]	1.027/256	300×300	2
iCubWorld subset (ICUB)[38]	86.400/9.600	256×256	10
Indian Food (IF) [76]	3.600/400	550×610	80
Make No Make(MVSN)[77]	1.355/151	211×246	2
Malaria (MAL) [78]	24.802/2.756	133×132	2
Meat quality (MQA) [79]	1.706/190	720×1280	2
Oxford Flowers (OF) [80]	2.040/6.149	538×624	102
Oxford-IIIT Pets (OP) [81]	3.680/3.669	383×431	37
Plankton (PL) [82]	4.500/500	106×120	10
Sars Covid (SCOV) [83]	2.232/249	260×350	2
Stanford Cars (SC) [84]	8.144/8.041	308×573	196
Stanford Dogs (SD) [85]	12.000/8.580	386×443	120
Tensorflow Flowers(TFF) [86]	3.303/367	272×365	5
Weather (MW) [87]	1.012/113	335×506	4

Target datasets

32 Target datasets

Small to medium size

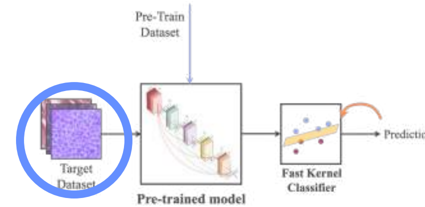
On average

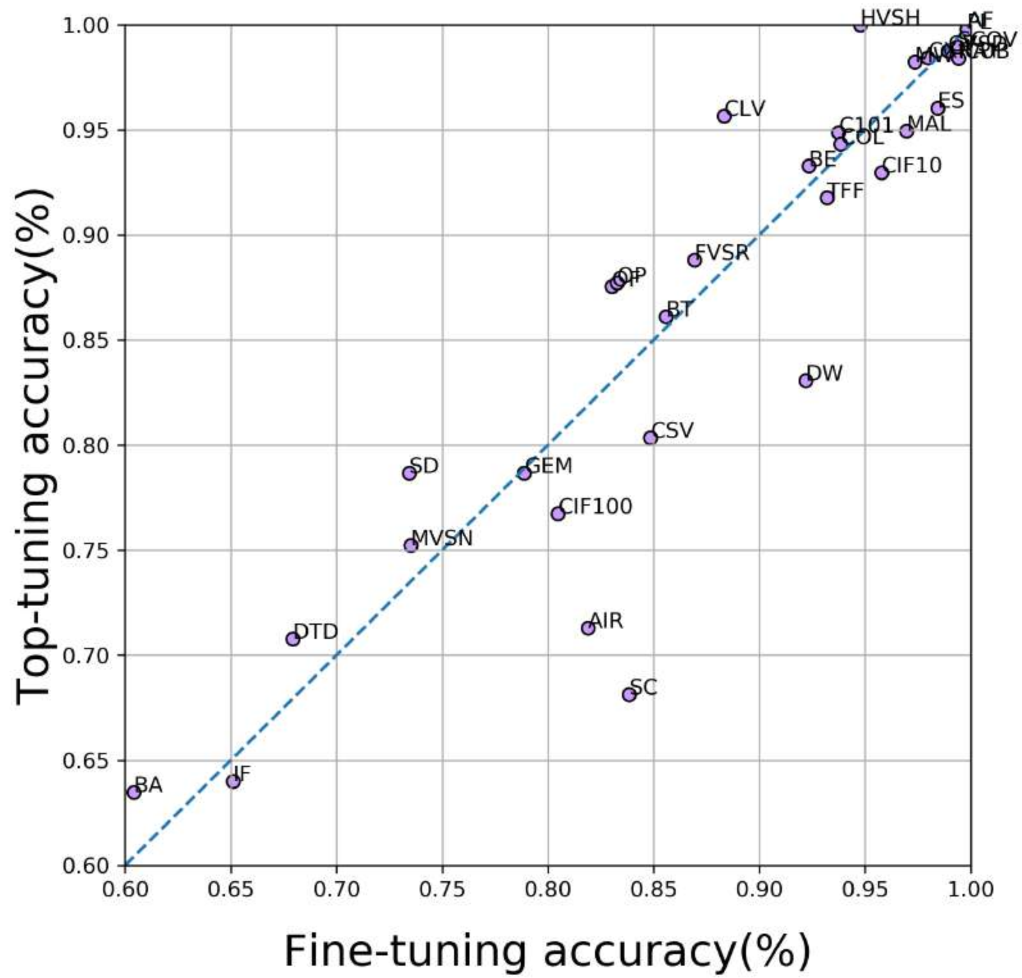
11.746 images

35 classes

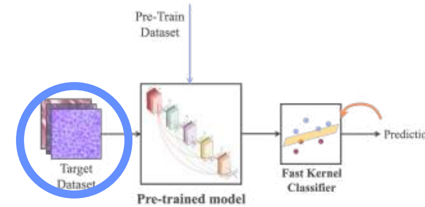
Dataset name	#images (Tr/Te)	Img. size mean	#classes
AFHQ (AF)[58]	13.167/1.463	512×512	3
Beans (BE)[59]	1.167/128	500×500	3
Best artworks (BA)[60]	7.896/878	980×921	50
Boat types (BT)[61]	1.315/147	905×1234	9
Caltech-101 (C101)[62]	3.060/6.084	251×282	102
Cassava (CSV)[63]	7.545/1.885	573×611	5
Cats vs Dogs (CVSD) [64]	20.935/2.327	365×410	2
Chest xray (CXRAY) [65]	4.708/524	968×1321	2
CIFAR10 (CIF10) [66]	50.000/10.000	32×32	10
CIFAR100 (CIF100) [66]	50.000/10.000	32×32	100
Citrus leaves (CLV) [67]	534/60	256×256	4
Colorectal hist (COL) [68]	4.500/500	150×150	8
Deep weeds (DW) [69]	15.758/1.751	256×256	9
DTD (DTD)[70]	3.760/1.880	453×500	47
EuroSAT (ES) [71]	24.300/2.700	64×64	10
FGVC Aircraft (AIR)[72]	6.667/3.333	353×1056	100
Footb vs Rugby (FVSR) [73]	2.203/245	618×788	2
Gemstones (GEM) [74]	2.571/286	330×335	87
Hors or Hum (HVSH) [75]	1.027/256	300×300	2
iCubWorld subset (ICUB)[38]	86.400/9.600	256×256	10
Indian Food (IF) [76]	3.600/400	550×610	80
Make No Make(MVSN)[77]	1.355/151	211×246	2
Malaria (MAL) [78]	24.802/2.756	133×132	2
Meat quality (MQA) [79]	1.706/190	720×1280	2
Oxford Flowers (OF) [80]	2.040/6.149	538×624	102
Oxford-IIIT Pets (OP) [81]	3.680/3.669	383×431	37
Plankton (PL) [82]	4.500/500	106×120	10
Sars Covid (SCOV) [83]	2.232/249	260×350	2
Stanford Cars (SC) [84]	8.144/8.041	308×573	196
Stanford Dogs (SD) [85]	12.000/8.580	386×443	120
Tensorflow Flowers(TFF) [86]	3.303/367	272×365	5
Weather (MW) [87]	1.012/113	335×506	4

Marginal fine-tuning benefits





Marginal fine-tuning benefits

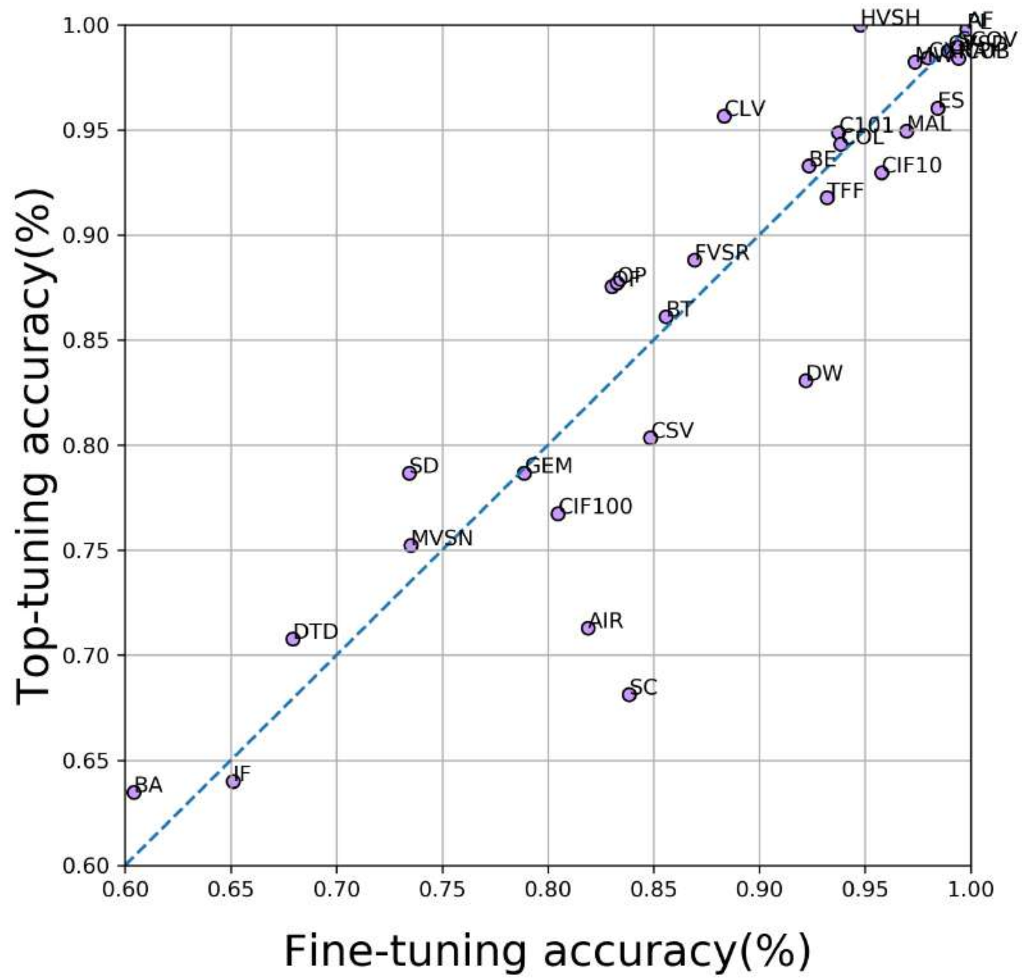


Accuracy comparison:

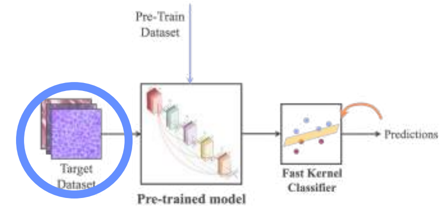
11/32: 'same' [±1.0%]

10/32: fine-tuning better

11/32: top-tuning better



Marginal fine-tuning benefits



Accuracy comparison:

11/32: ‘same’ [±1.0%]

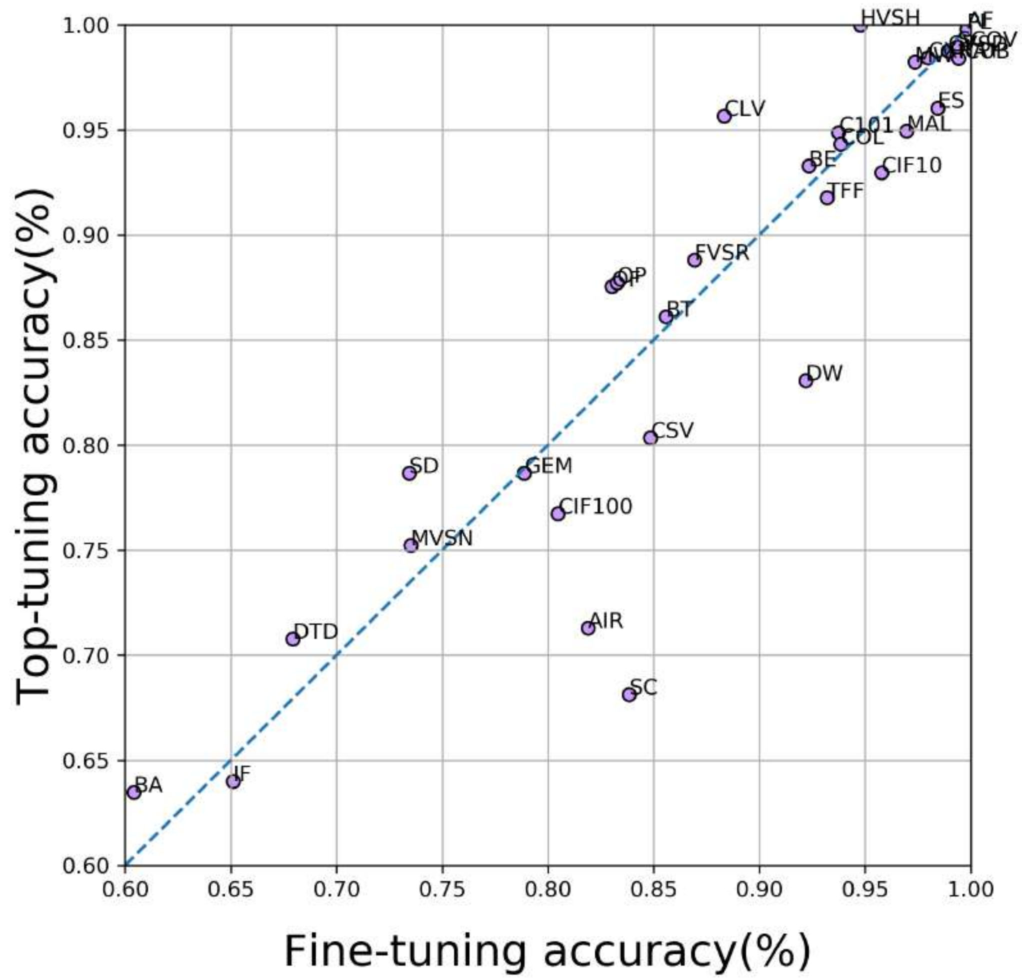
10/32: fine-tuning better

11/32: top-tuning better

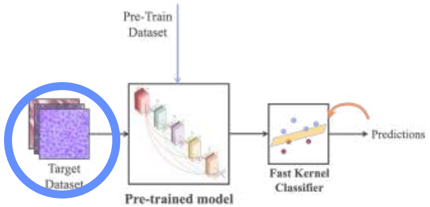
Aircraft, Stanford Cars?

- Fine-grained
- Few data
- Not represented in ImageNet

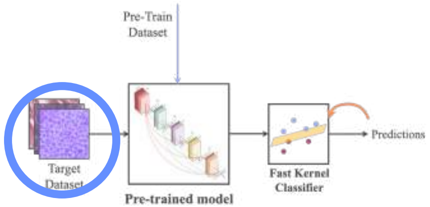
[Kornblith et al 2018]

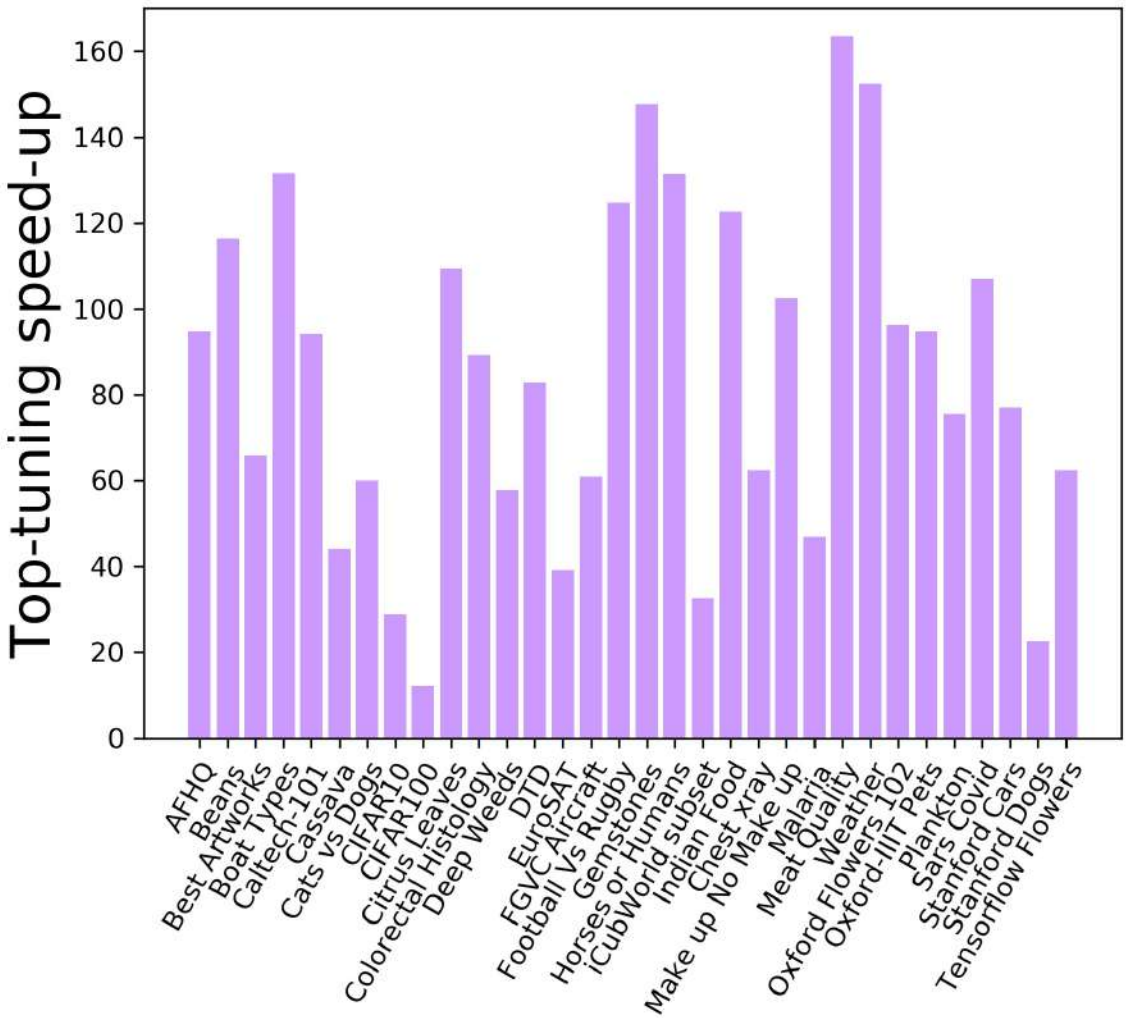


Top-tuning: hours to minutes

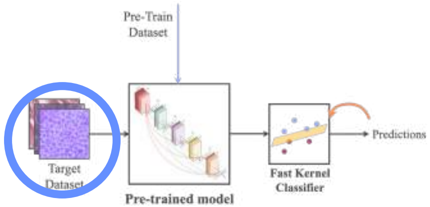


Top-tuning: hours to minutes



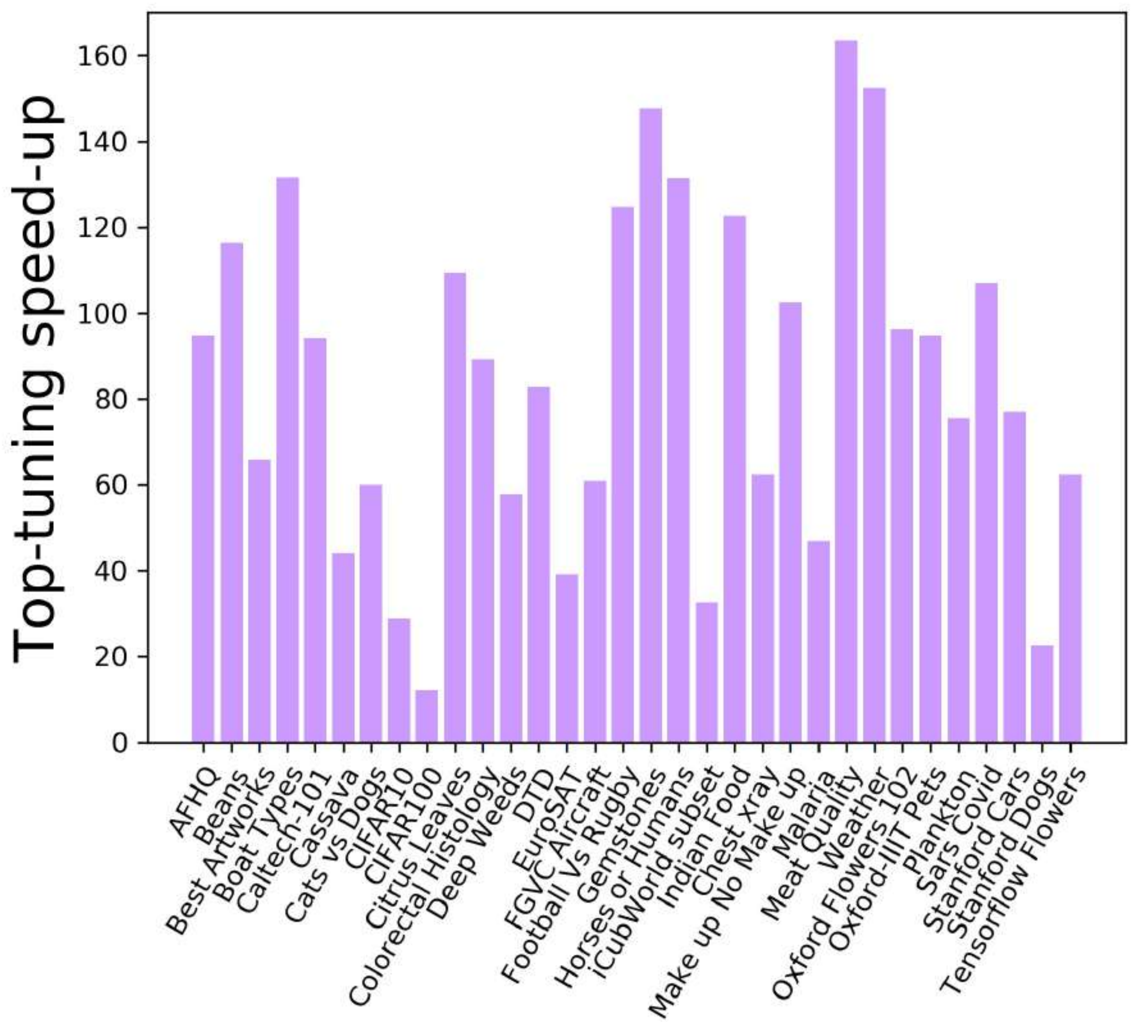


Top-tuning: hours to minutes

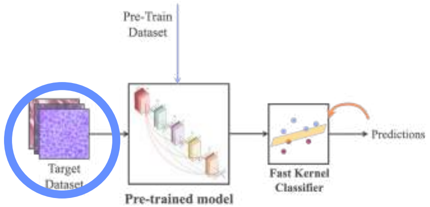


Massive speed-up:

~[10, 150]x



Top-tuning: hours to minutes



Massive speed-up:

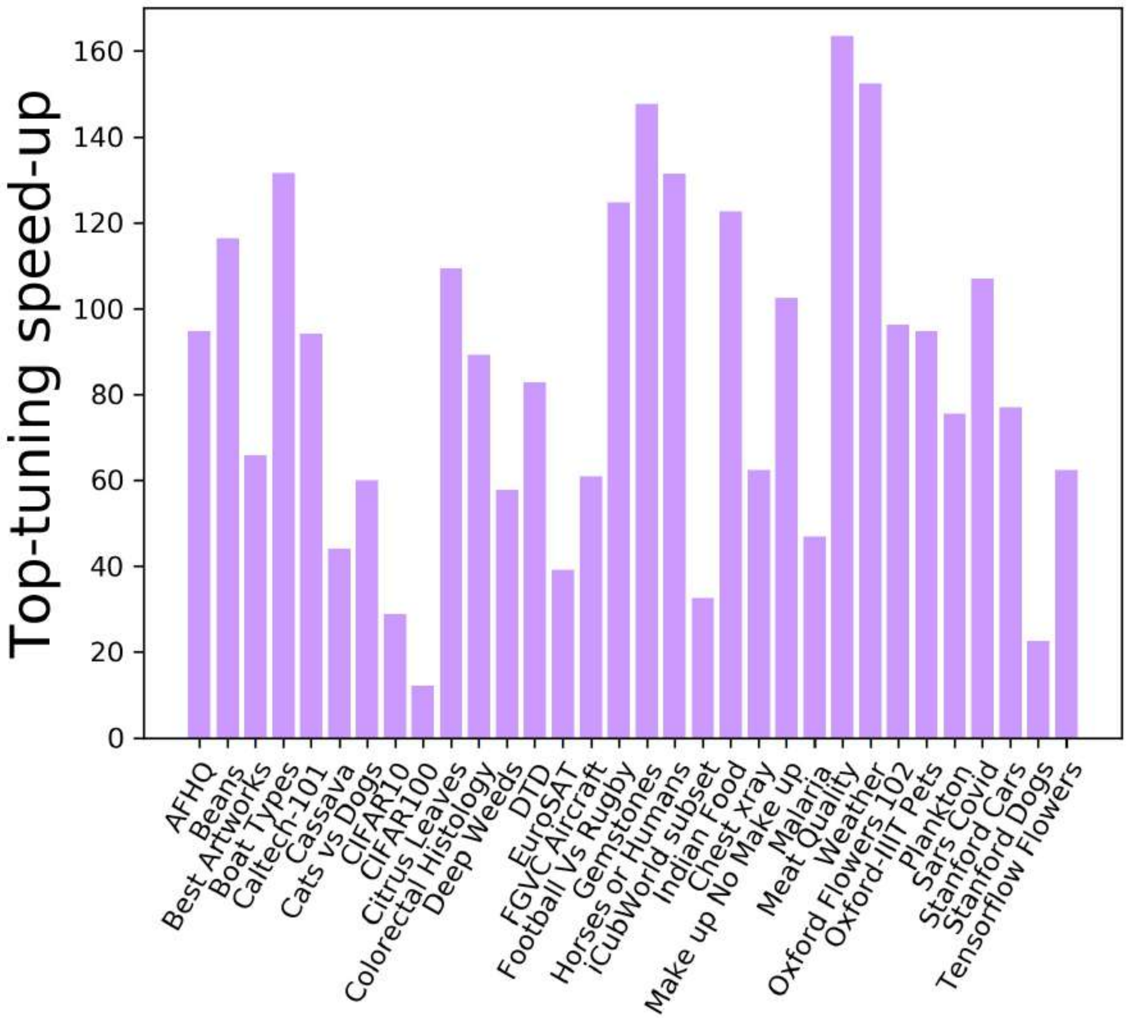
~[10, 150]x

Avg training time:

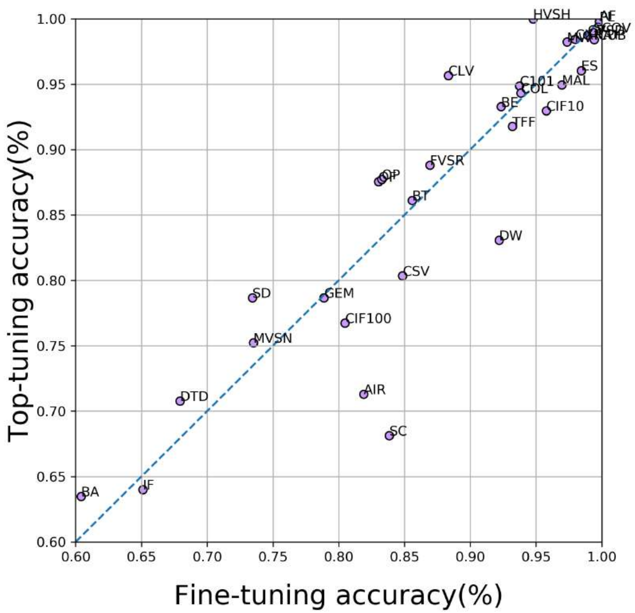
Fine-tuning ~48 mins

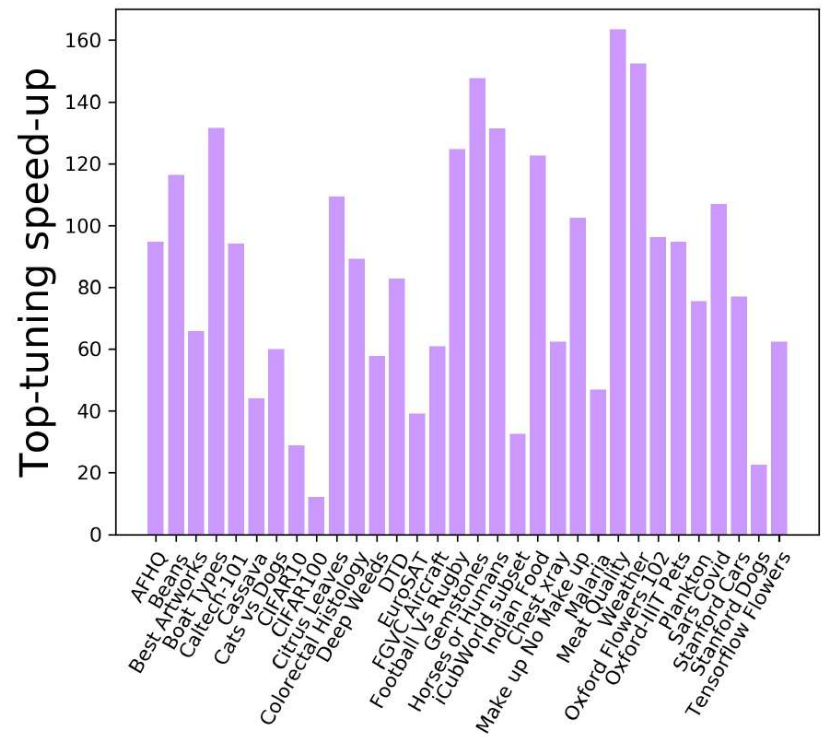
Top-tuning ~1 min

Quadro RTX 6000, 24Gb

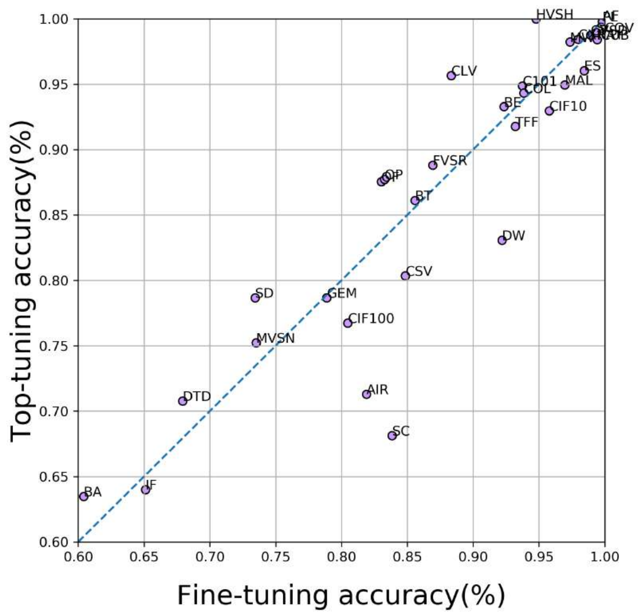


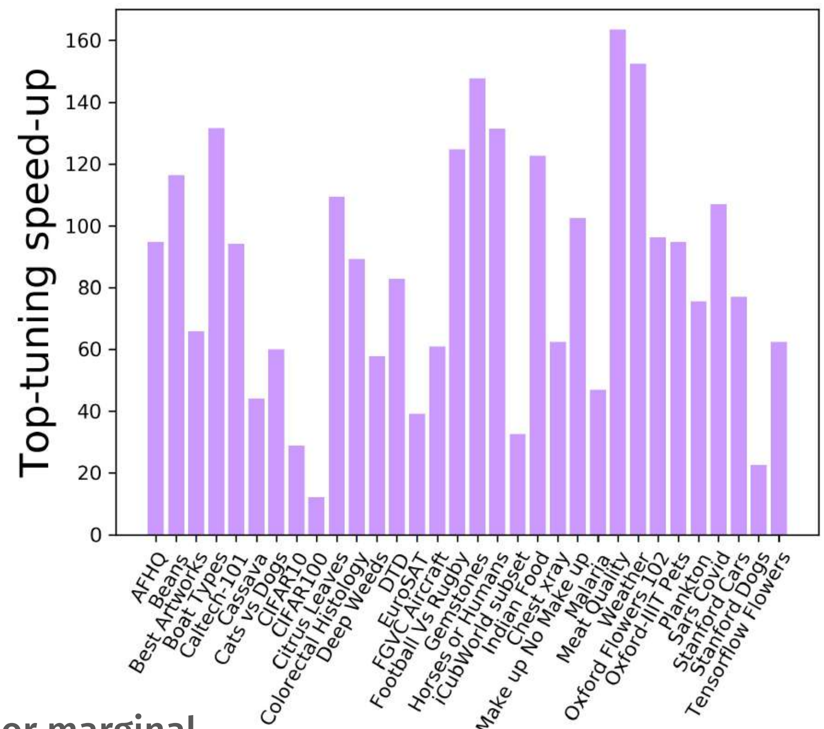
Take home messages





Take home messages

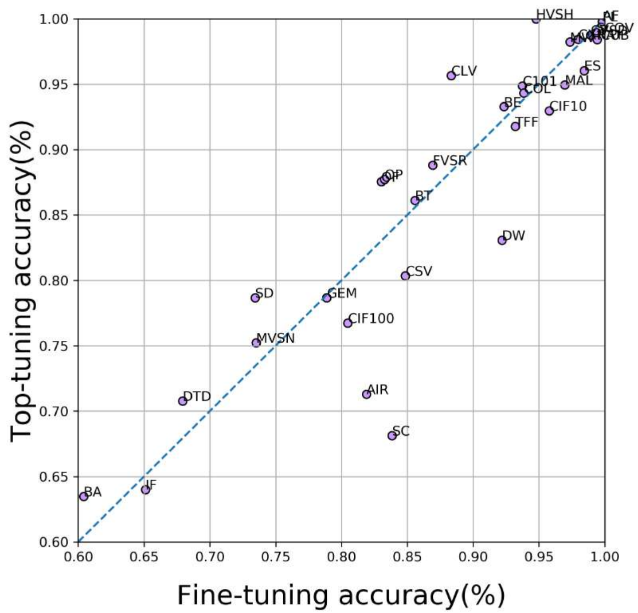


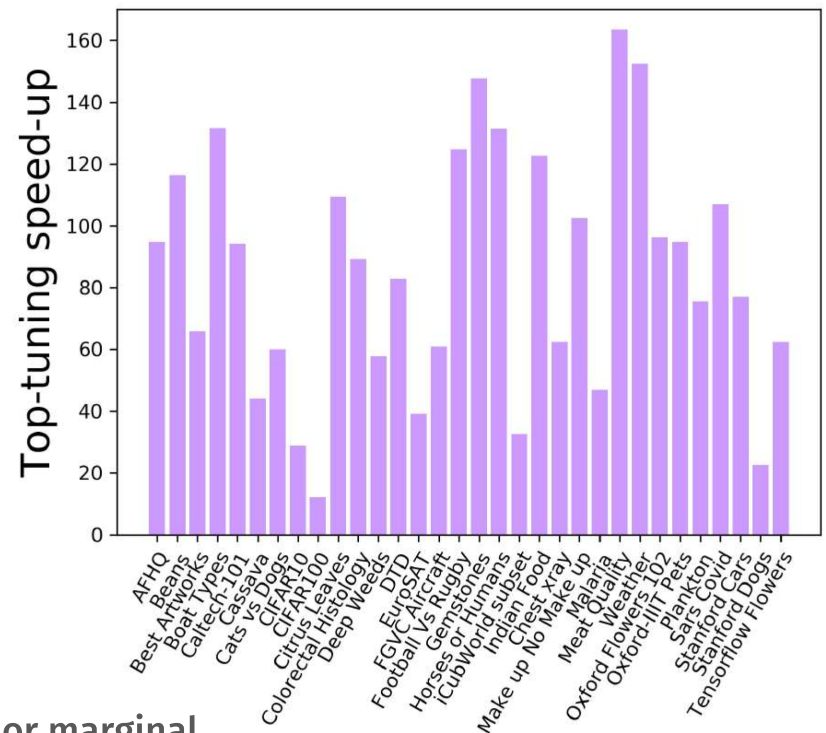


1. **Accuracy benefit of fine-tuning: absent or marginal**

2. **Top-tuning massive time saving: hours to minutes**

Take home messages



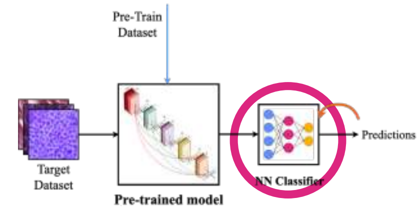


1. **Accuracy benefit of fine-tuning: absent or marginal**
2. **Top-tuning massive time saving: hours to minutes**

Results robustness?

Ablation study

Ablation study



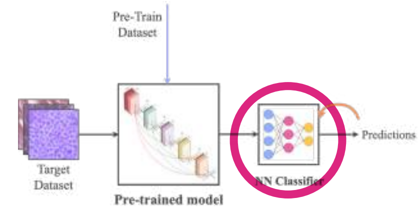
Target Dataset

Classifier

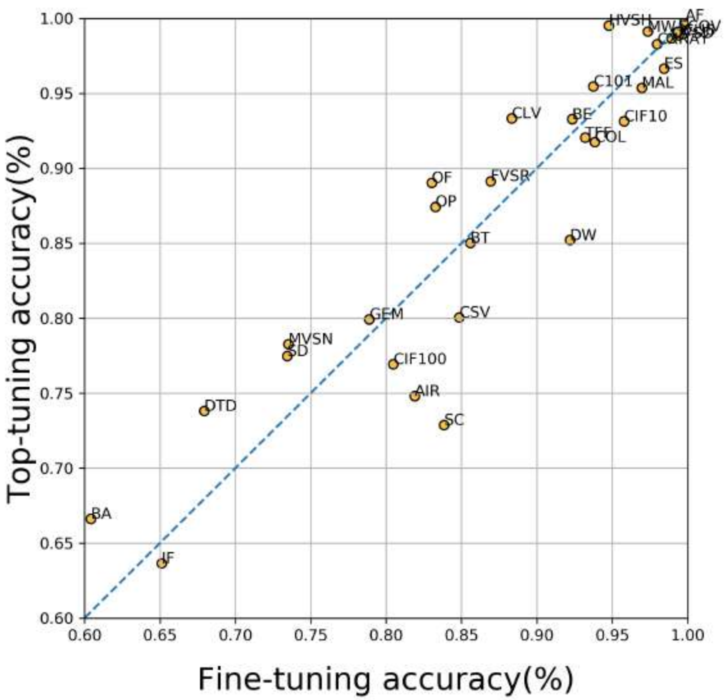
Pre-trained Model

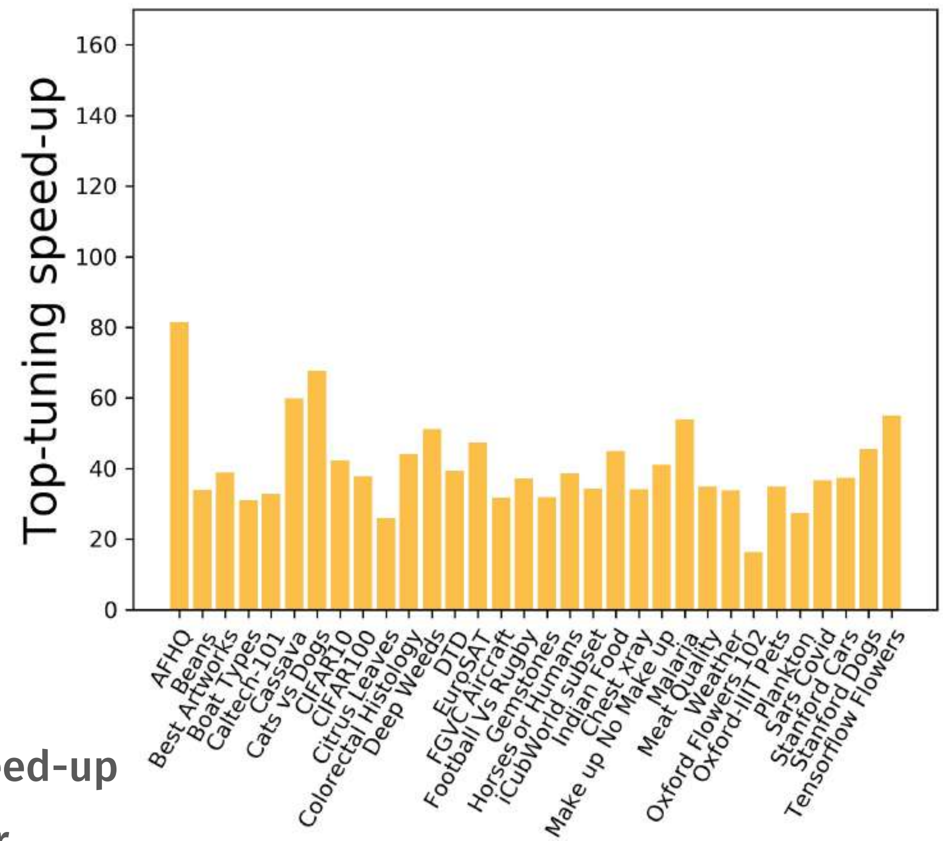
Pre-train Source

Classifier: low dependency



Fully connected Neural Network

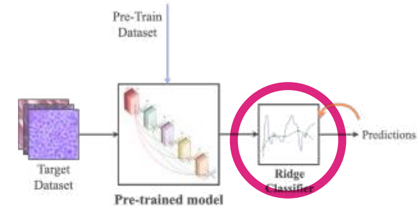




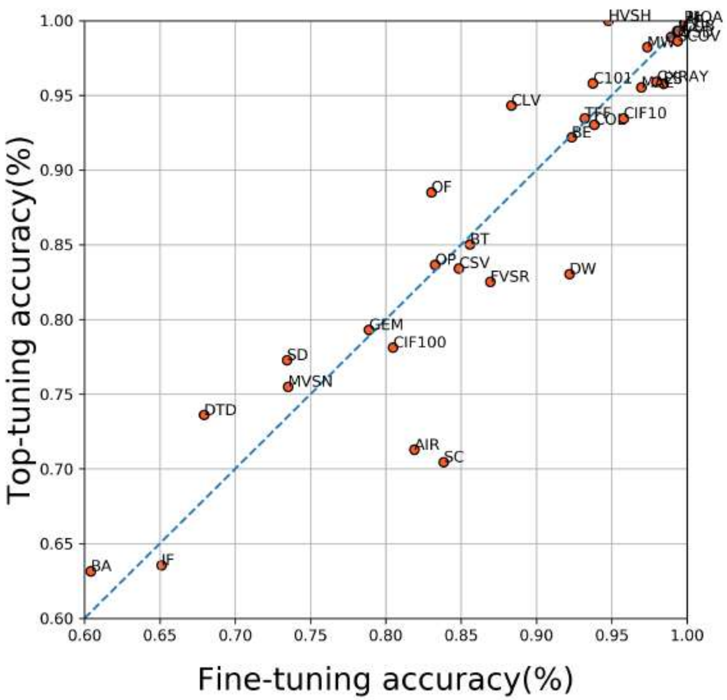
Similar trend in accuracy and speed-up

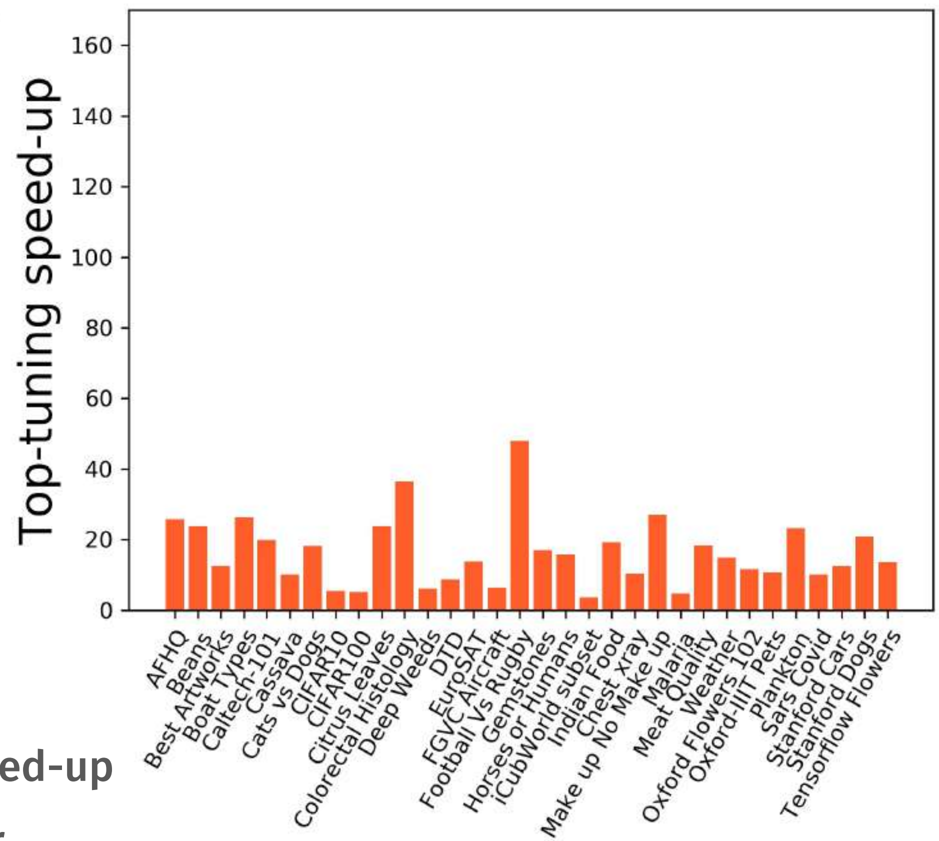
Slower w.r.t. Fast Kernel classifier

Classifier: low dependency



Ridge Regression Classifier

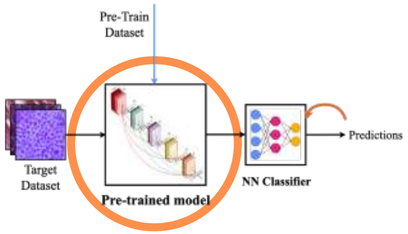




Similar trend in accuracy and speed-up

Slower w.r.t. Fast Kernel classifier

Ablation study



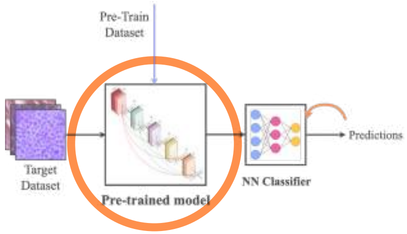
Target Dataset

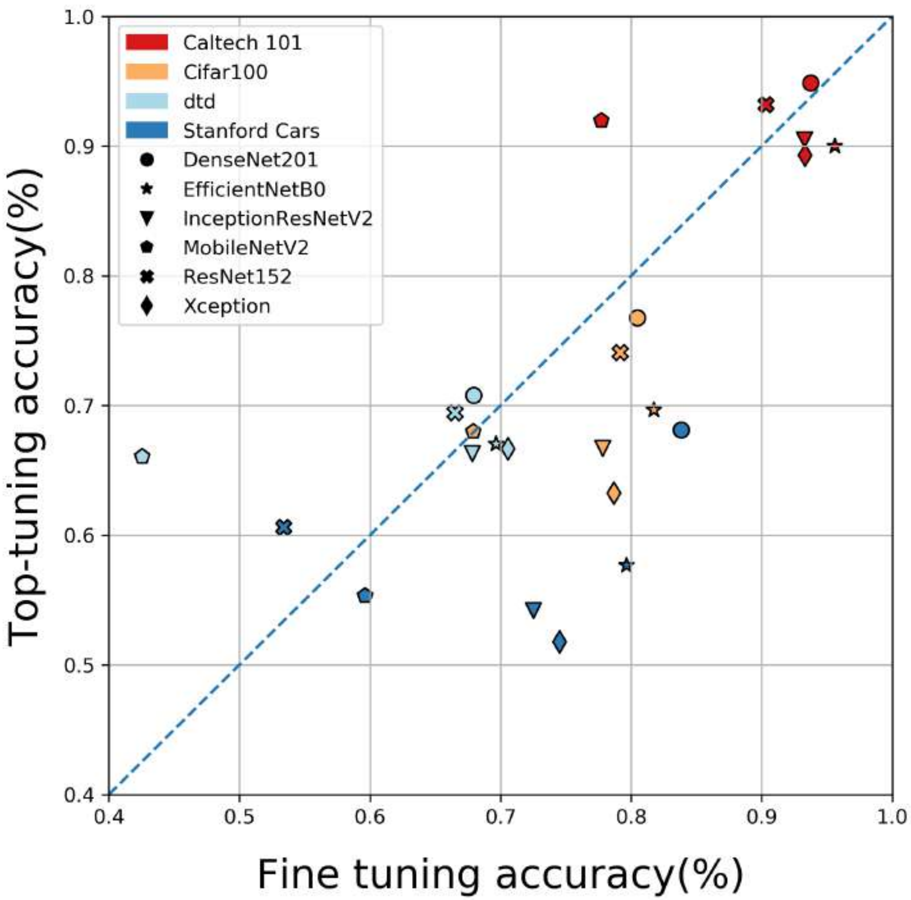
Classifier

Pre-trained Model

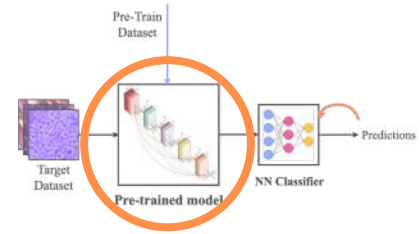
Pre-train Source

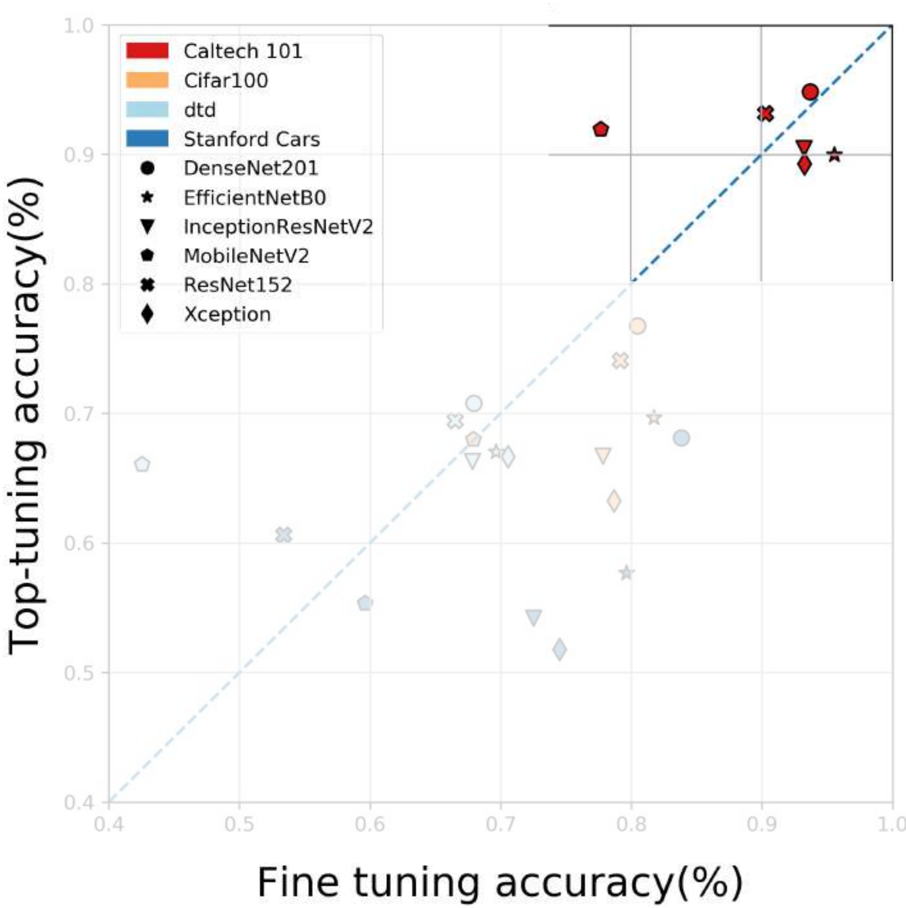
Low impact of pre-trained model



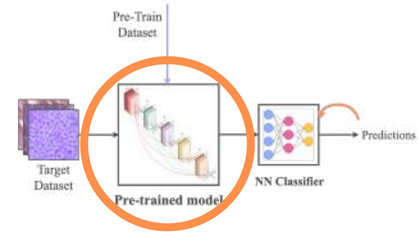


Low impact of pre-trained model



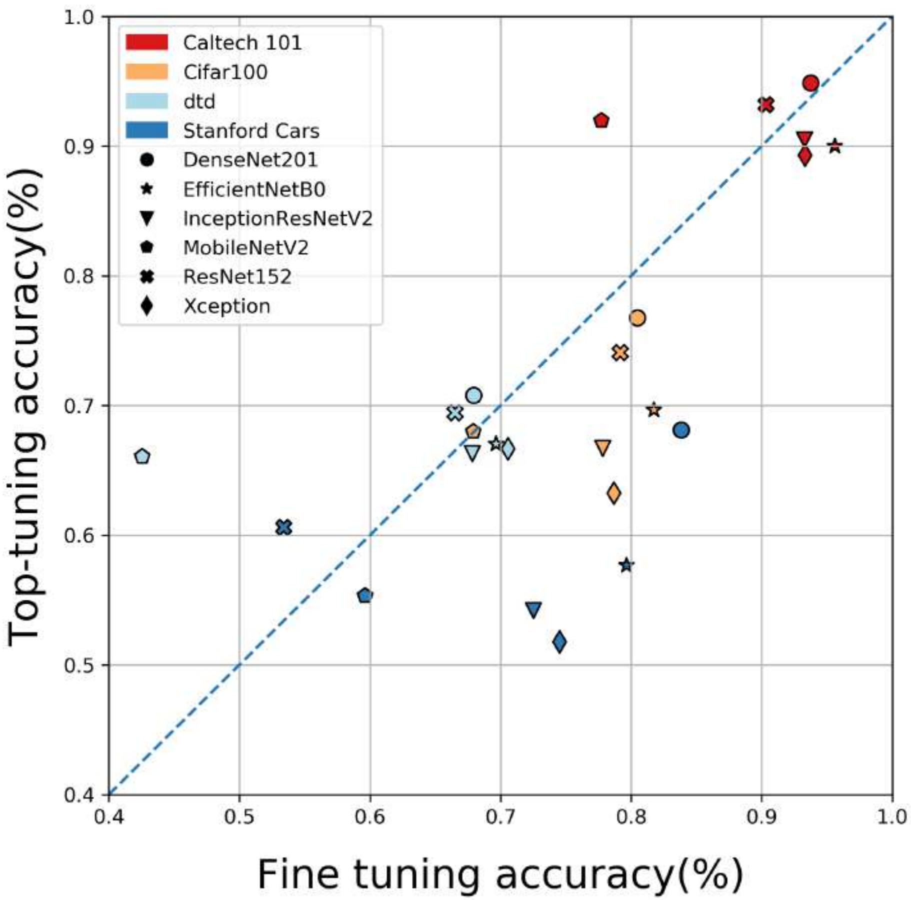


Low impact of pre-trained model

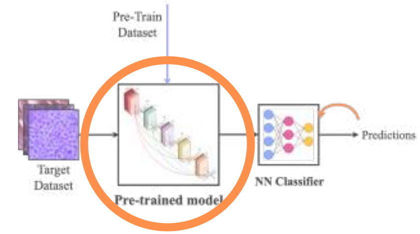


Similar trend

Low dependency from pre-trained model

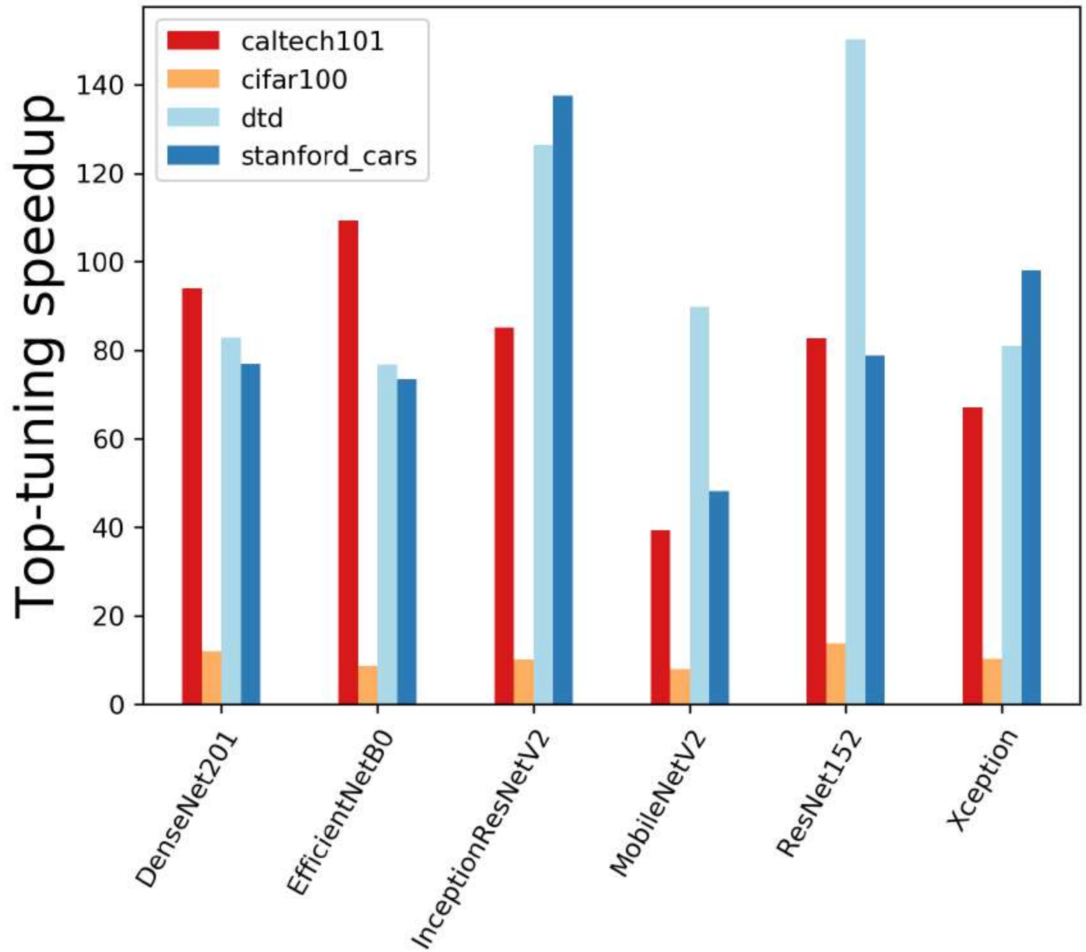


Low impact of pre-trained model

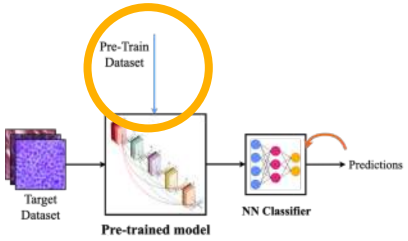


Similar speed-up

Low dependency from pre-trained model



Ablation study



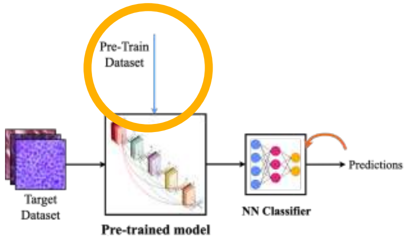
Target Dataset

Classifier

Pre-trained Model

Pre-train Source

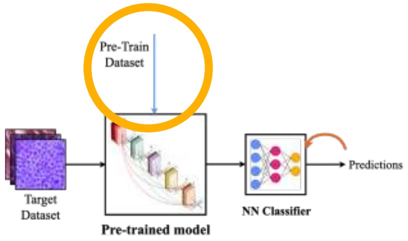
Pre-train, general infos



3 additional pre-trains with same #images:

Cifar100, ImageNet100, ImageNet50k

Pre-train, general infos



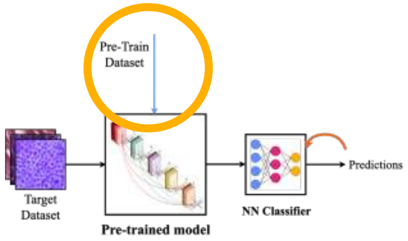
3 additional pre-trains with same #images:

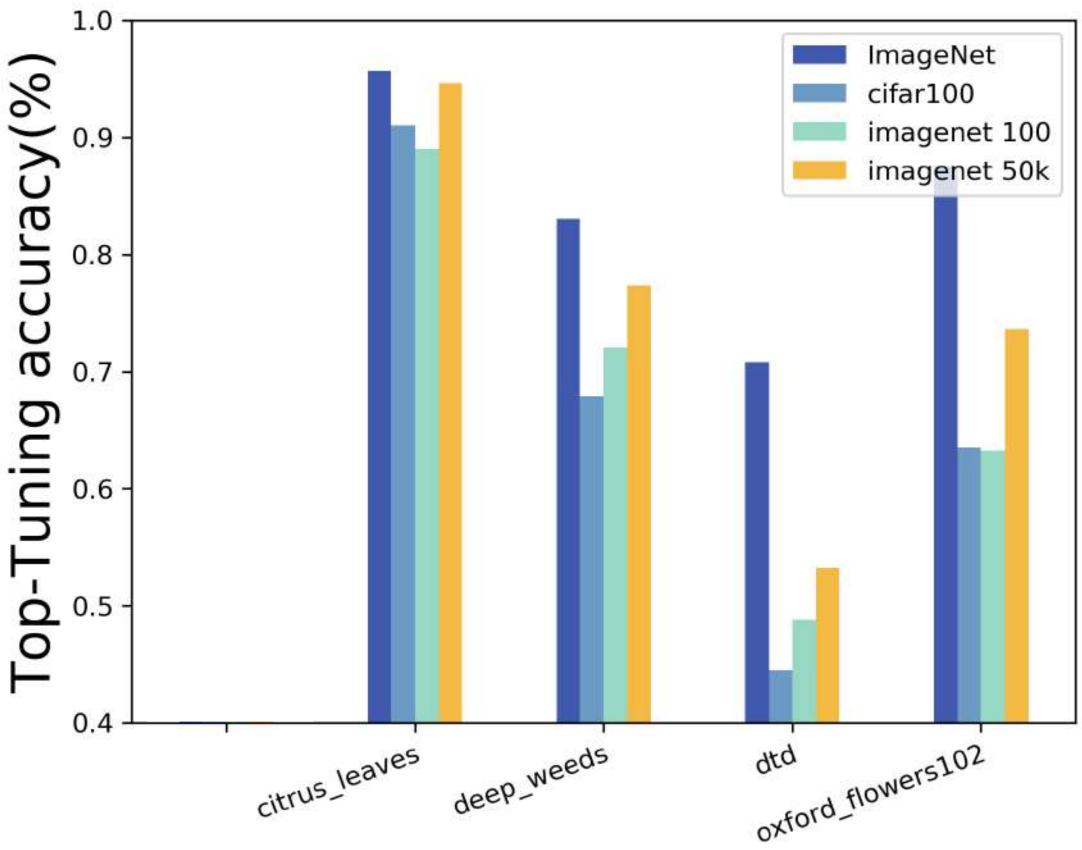
Cifar100, ImageNet100, ImageNet50k

W.r.t. ImageNet:

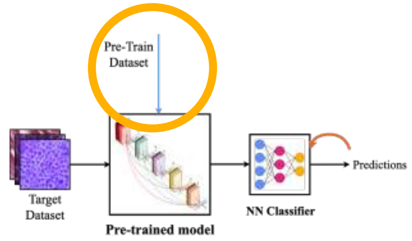
Cifar100:	low amount of classes	many samples per class
ImageNet100:	low amount of classes	many samples per class
ImageNet50k:	high amount of classes	few samples per class

Semantic variability matters



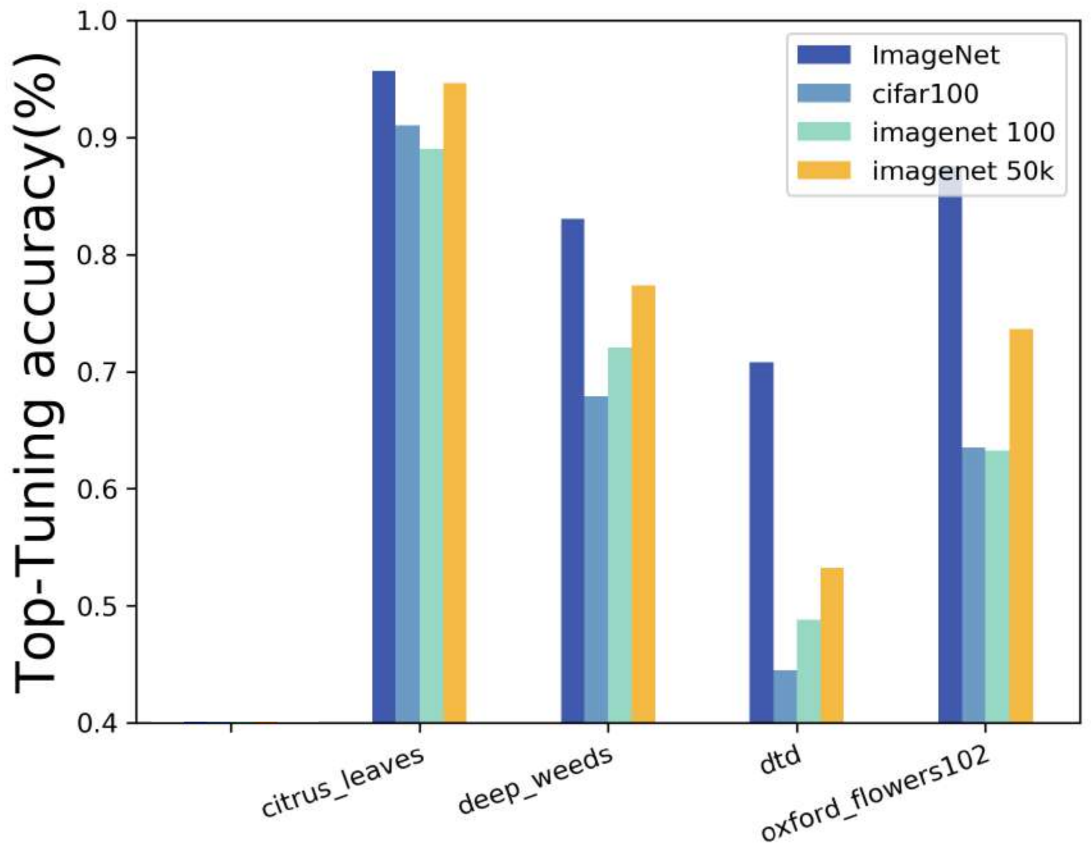


Semantic variability matters

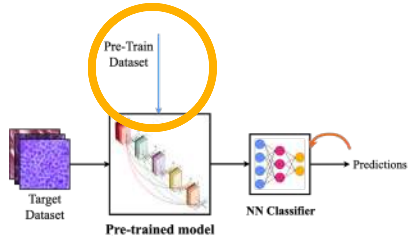


Whole ImageNet

always better



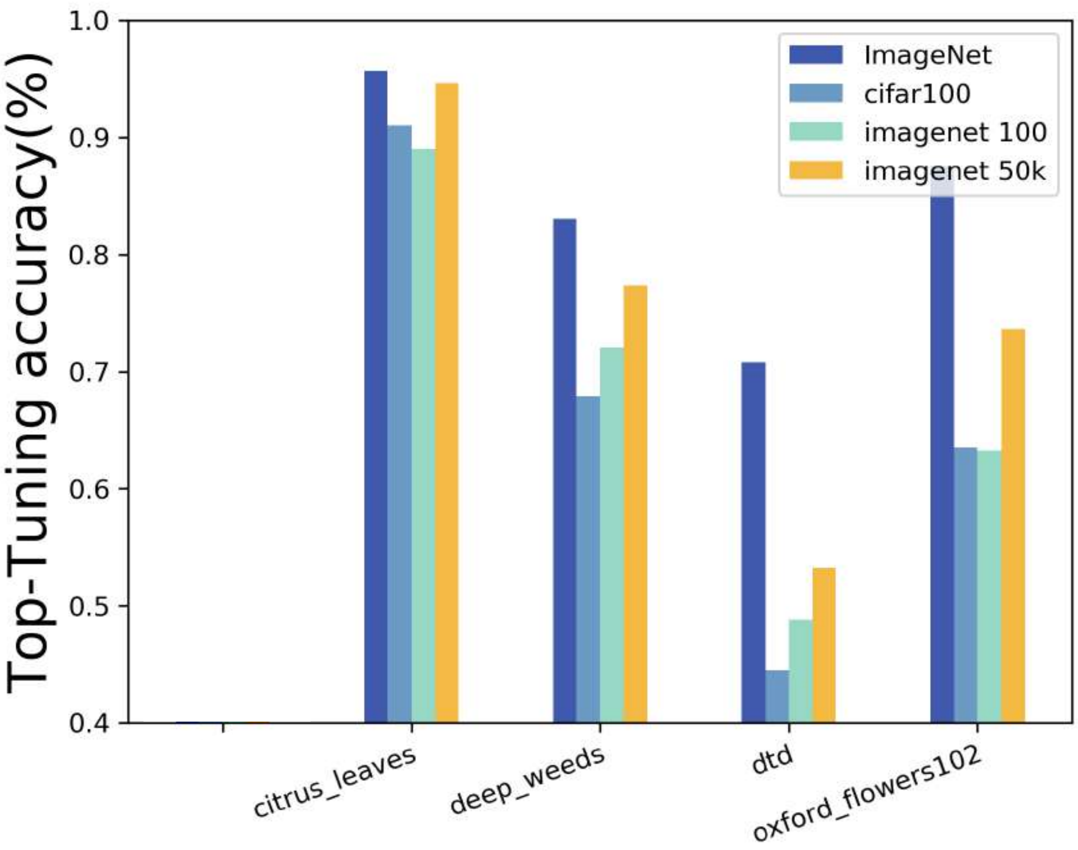
Semantic variability matters



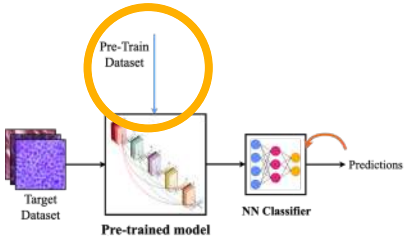
Whole ImageNet

always better

ImageNet50k 2° best choice..



Semantic variability matters

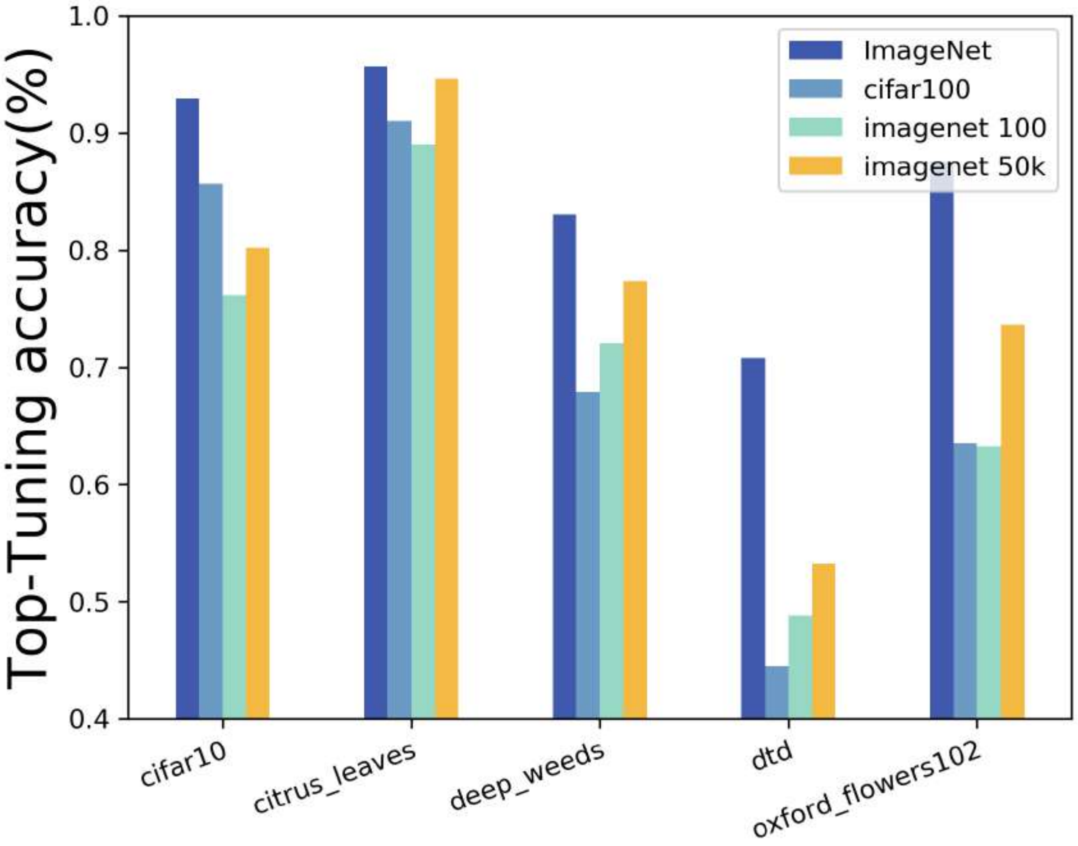


Whole ImageNet

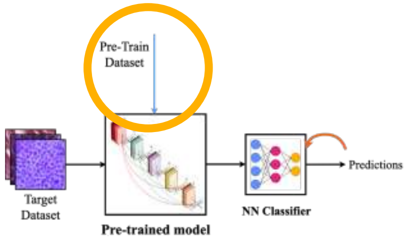
always better

ImageNet50k 2° best choice..

..except on cifar10 target



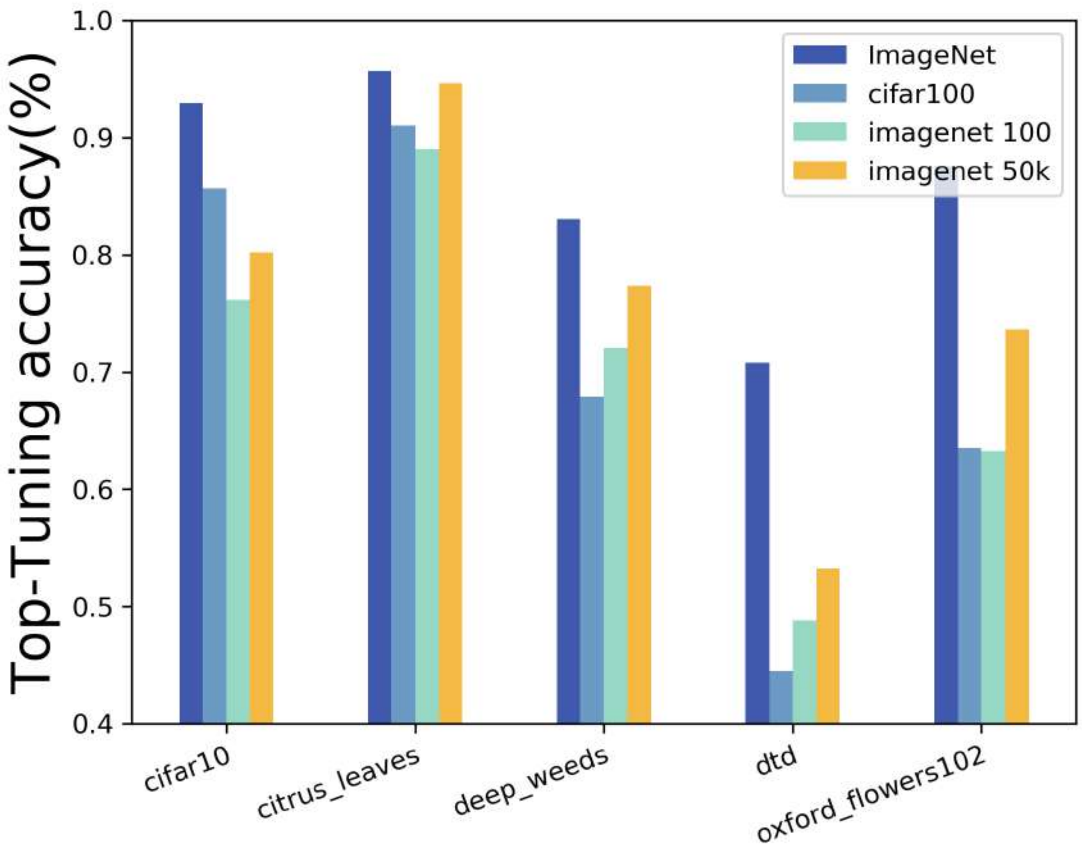
Semantic variability matters



Whole ImageNet
always better

ImageNet50k 2° best choice..
..except on cifar10 target

Semantic variability matters!



Training time efficiency, conclusions

Training time efficiency, conclusions

- Accuracy benefit of fine-tuning: absent or marginal

Training time efficiency, conclusions

- Accuracy benefit of fine-tuning: absent or marginal
- Top-tuning massive time saving: hours to minutes

Training time efficiency, conclusions

- Accuracy benefit of fine-tuning: absent or marginal
- Top-tuning massive time saving: hours to minutes
- Consistency across architectural design choices

Pre-trained features role

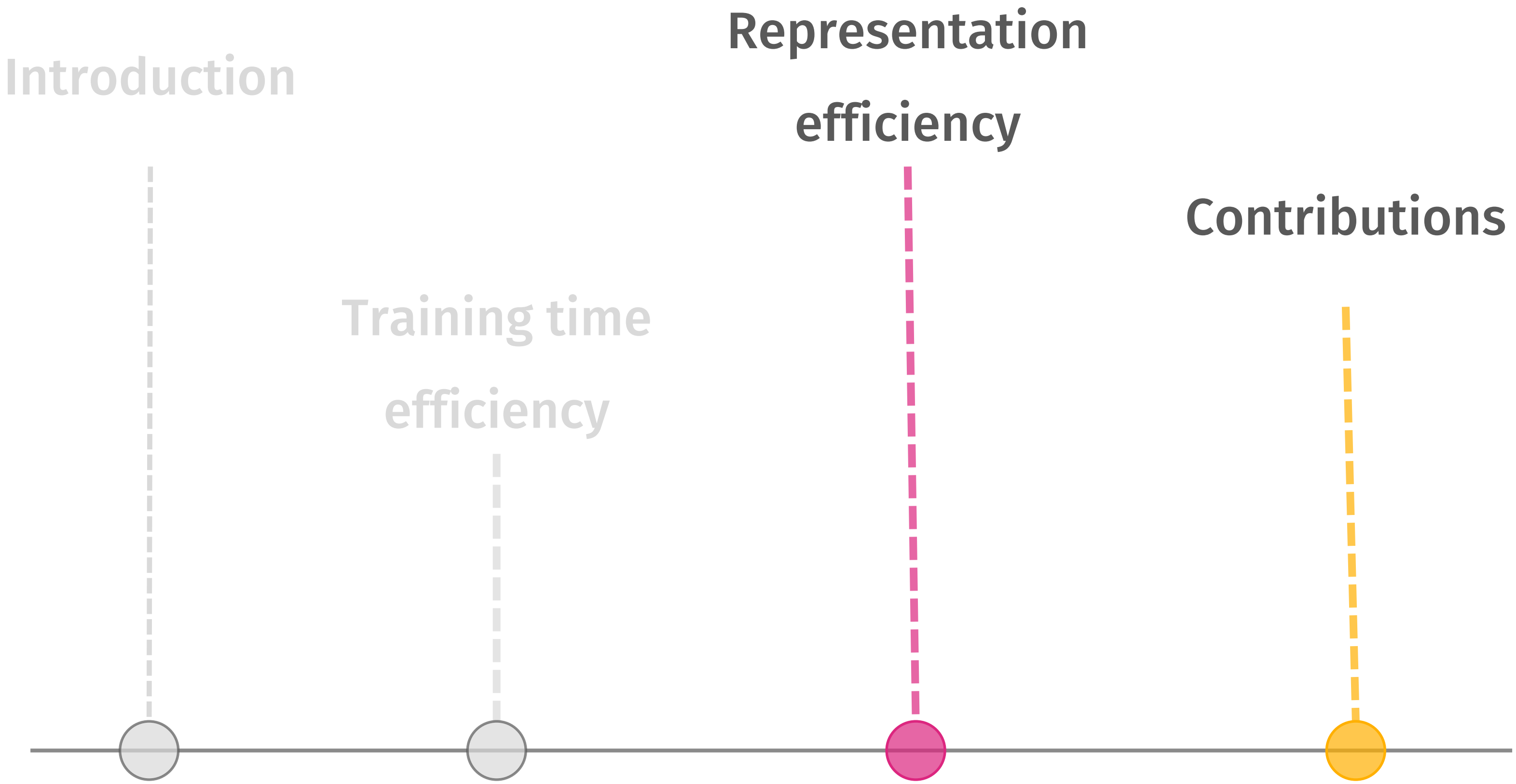
“Universal” representation?

Beyond image classification?

[Maiettini et al 2018]

[Ceola et al 2022]

Outline


Introduction
Training time efficiency
Representation efficiency
Contributions

Representation efficiency

Efficient Unsupervised Learning for Plankton Images, Alfano, Rando, Letizia, Pastore, Rosasco, Odone

Published @ICPR 2022

Clustering plankton images

Clustering plankton images

5000 images

10 classes

Plankton domain:

- Many unlabeled data
- Many classes
- Embedded device, marine microscopy

Clustering plankton images

5000 images

10 classes

Plankton domain:

- Many unlabeled data
- Many classes
- Embedded device, marine microscopy

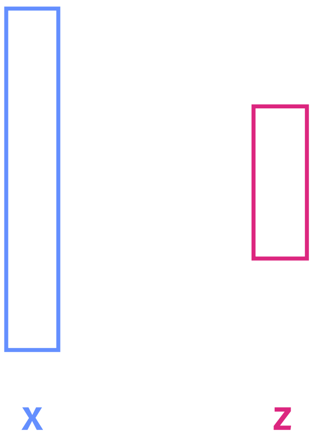
Image clustering via features extraction:

- Pre-trained features, too big!

Variational Auto Encoders

[Kingma and Welling 2014]

Unsupervised model, no labels



Variational Auto Encoders

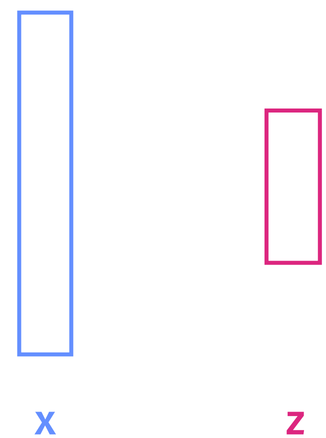
[Kingma and Welling 2014]

Unsupervised model, no labels

Aim: informative encoding

$\mathbf{x}$ ~ 10^4 elements

$\mathbf{z}$ ~ 10^2 elements



Variational Auto Encoders

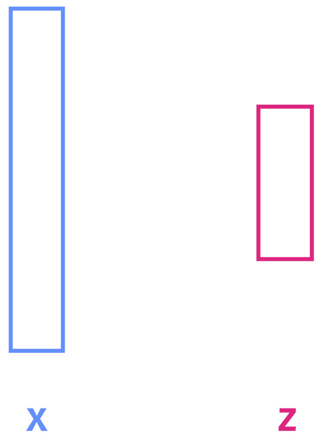
[Kingma and Welling 2014]

Unsupervised model, no labels

Aim: informative encoding

$\mathbf{x} \sim 10^4$ elements

$\mathbf{z} \sim 10^2$ elements

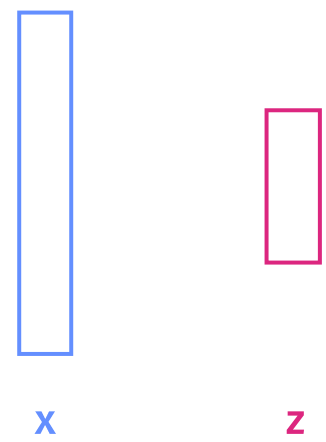


Bottleneck: only main info go through

Variational Auto Encoders

[Kingma and Welling 2014]

How to compress?



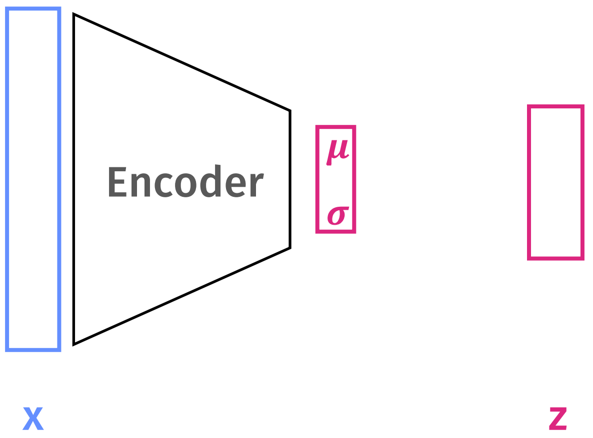
Variational Auto Encoders

[Kingma and Welling 2014]

How to compress?

3 parts model:

- Encode (compression)



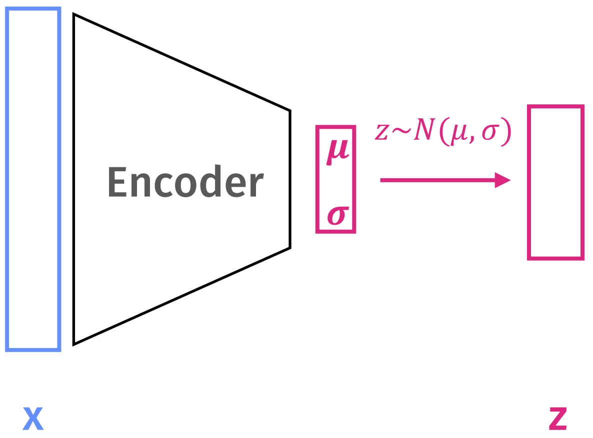
Variational Auto Encoders

[Kingma and Welling 2014]

How to compress?

3 parts model:

- Encode (compression)
- Sampling



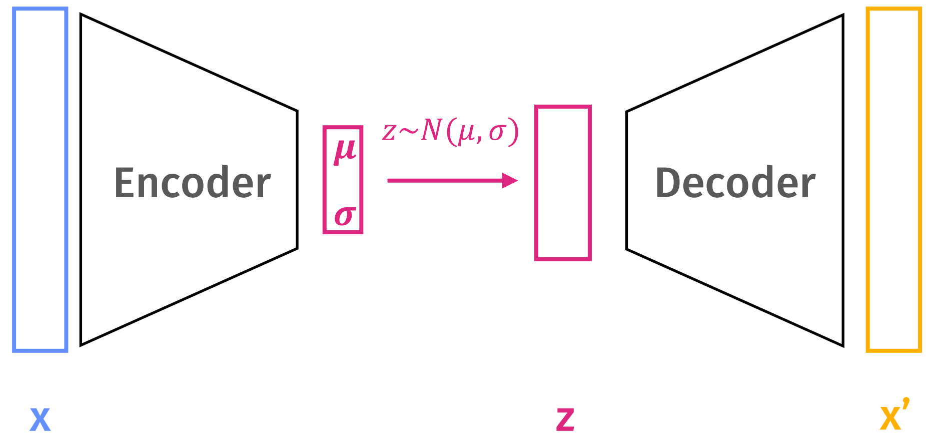
Variational Auto Encoders

[Kingma and Welling 2014]

How to compress?

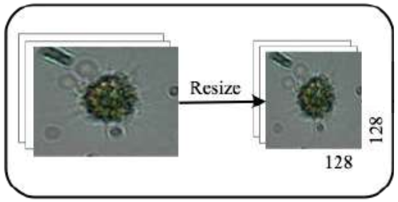
3 parts model:

- Encode (compression)
- Sampling
- Decode (decompression)



Pipeline

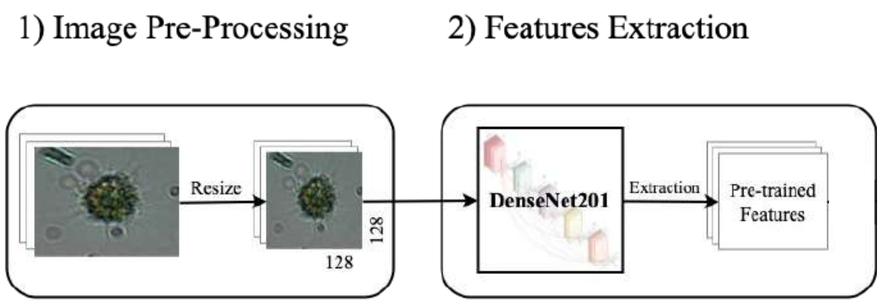
1) Image Pre-Processing



Resize
+
Normalization

Input ready

Pipeline



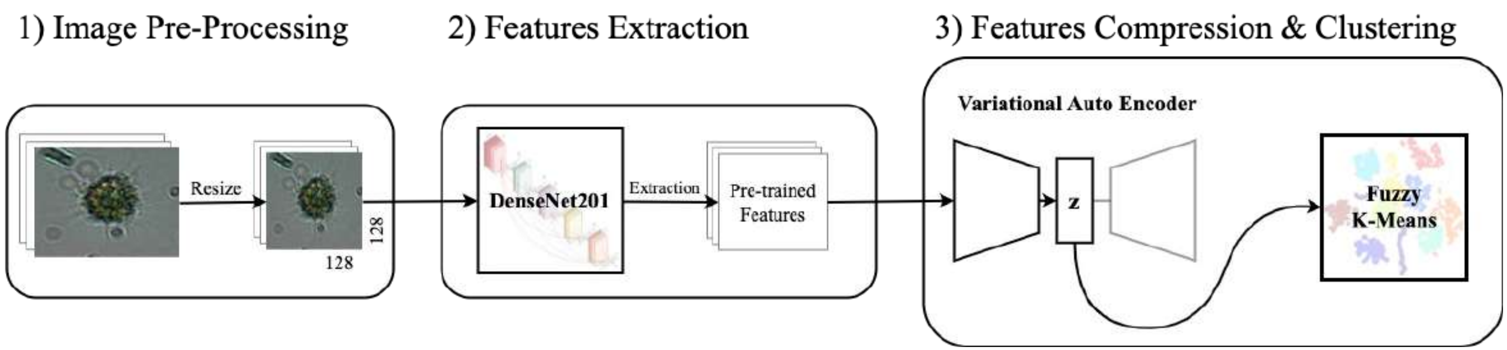
Resize
+
Normalization

Input ready

Pre-trained arch

Pre-trained Features

Pipeline



Resize
+
Normalization

Input ready

Pre-trained arch

Pre-trained Features

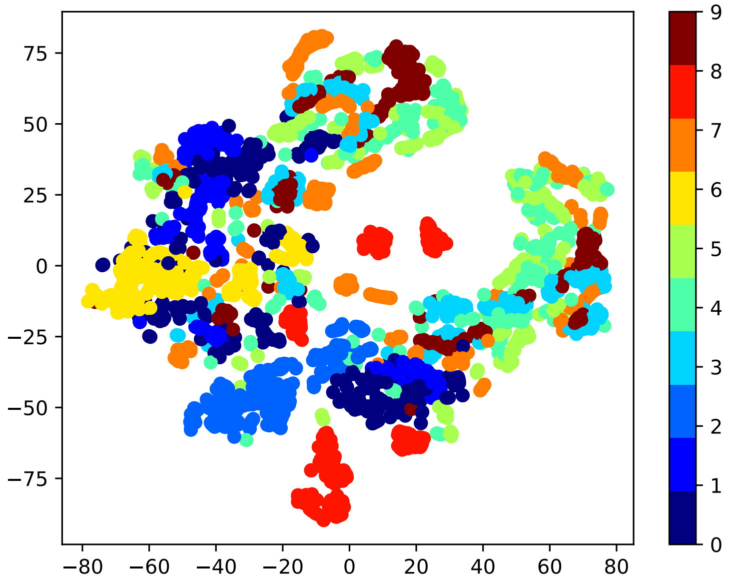
Variational Auto Encoder

Embedding in latent space

Clustering

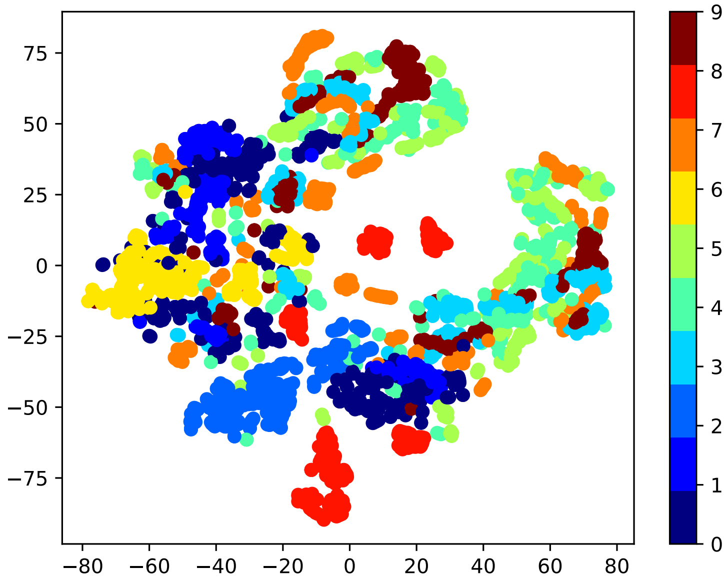
Qualitative results

Qualitative results

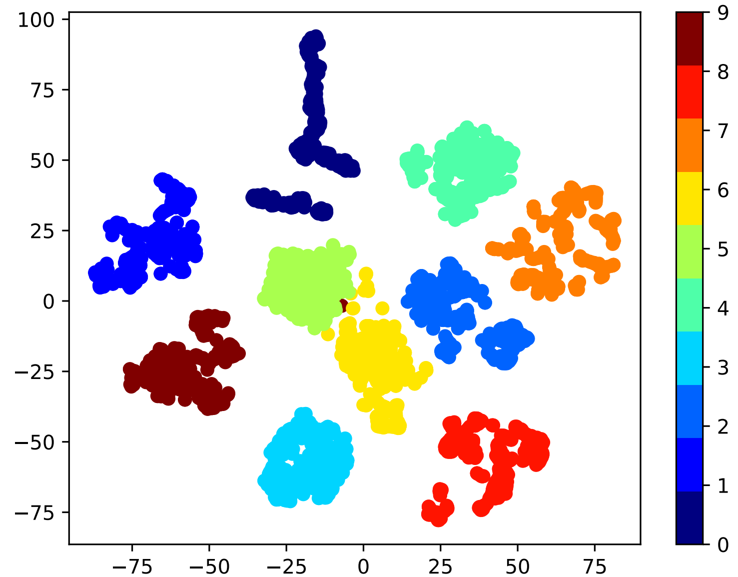


Input: images

Qualitative results



Input: images



Input: pre-trained features

UniGe | MaLGa

Quantitative metric

Evaluation by *purity* and *overlaps*

Quantitative metric

Evaluation by *purity* and *overlaps*

Purity: given N data point, a set of clusters M, a set of classes D:

$$\frac{1}{N}\sum_{m\in M}\max_{d\in D}|m\cap d|$$

Quantitative metric

Evaluation by *purity* and *overlaps*

Purity: given N data point, a set of clusters M, a set of classes D:

$$\frac{1}{N}\sum_{m\in M}\max_{d\in D}|m\cap d|$$

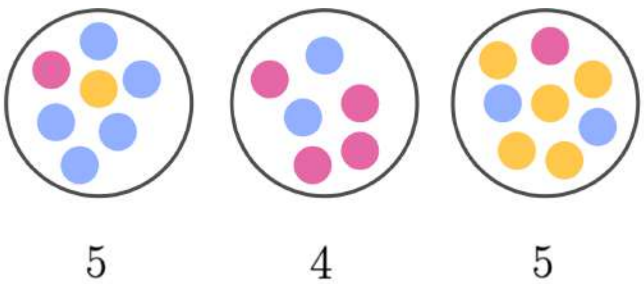
5

Quantitative metric

Evaluation by *purity* and *overlaps*

Purity: given N data point, a set of clusters M, a set of classes D:

$$\frac{1}{N}\sum_{m\in M}\max_{d\in D}|m\cap d|$$

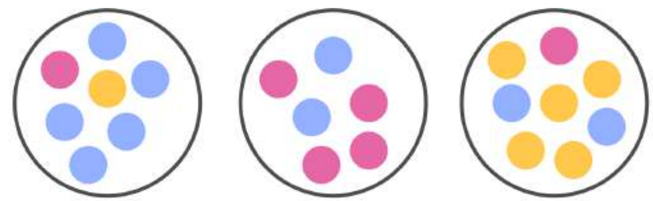


Quantitative metric

Evaluation by *purity* and *overlaps*

Purity: given N data point, a set of clusters M, a set of classes D:

$$\frac{1}{N}\sum_{m\in M}\max_{d\in D}|m\cap d|$$



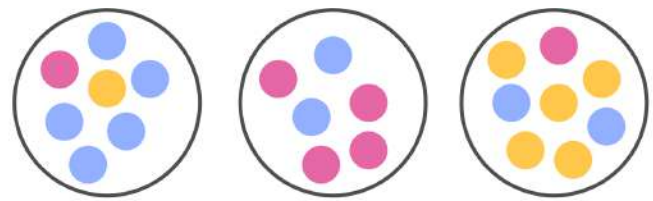
5 4 5 $\frac{14}{21}=\frac{2}{3}$

Quantitative metric

Evaluation by *purity* and *overlaps*

Purity: given N data point, a set of clusters M, a set of classes D:

$$\frac{1}{N}\sum_{m\in M}\max_{d\in D}|m\cap d|$$



5 4 5 $\frac{14}{21}=\frac{2}{3}$

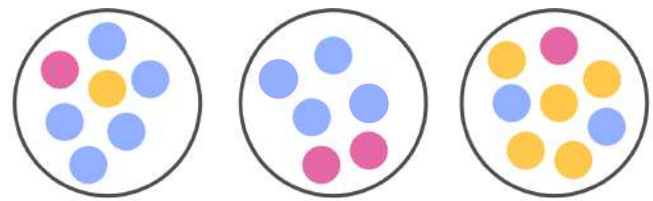
Overlaps: #classes lost

Quantitative metric

Evaluation by *purity* and *overlaps*

Purity: given N data point, a set of clusters M, a set of classes D:

$$\frac{1}{N}\sum_{m\in M}\max_{d\in D}|m\cap d|$$



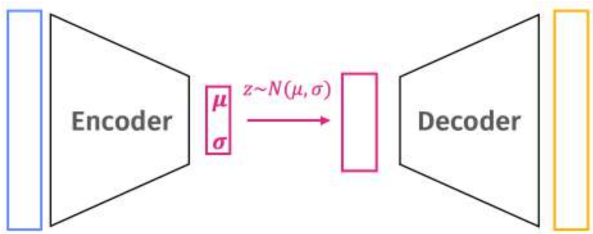
5 4 5 $\frac{14}{21}=\frac{2}{3}$

Overlaps: #classes lost

Results

Algorithm/Z	10	30	50	100	500
image-VAE	0.53 ± 0.017 (1.4 ± 0.5)	0.55 ± 0.04 (1.6 ± 0.49)	0.58 ± 0.01 (2.0 ± 0.63)	0.59 ± 0.01 (1.6 ± 0.48)	0.62 ± 0.01 (2.0 ± 0.0)
FE_{r_2}-VAE	0.98 ± 0.01 (0.0 ± 0.0)	0.98 ± 0.03 (0.0 ± 0.0)	0.98 ± 0.01 (0.0 ± 0.0)	0.98 ± 0.02 (0.0 ± 0.0)	0.98 ± 0.02 (0.0 ± 0.0)

Z: latent space dimension

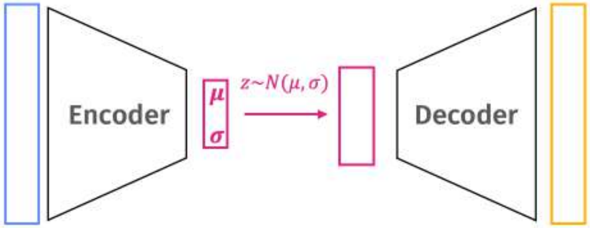


Results

Algorithm/Z	10	30	50	100	500
image-VAE	0.53 ± 0.017 (1.4 ± 0.5)	0.55 ± 0.04 (1.6 ± 0.49)	0.58 ± 0.01 (2.0 ± 0.63)	0.59 ± 0.01 (1.6 ± 0.48)	0.62 ± 0.01 (2.0 ± 0.0)
FE_{r_2}-VAE	0.98 ± 0.01 (0.0 ± 0.0)	0.98 ± 0.03 (0.0 ± 0.0)	0.98 ± 0.01 (0.0 ± 0.0)	0.98 ± 0.02 (0.0 ± 0.0)	0.98 ± 0.02 (0.0 ± 0.0)

Z: latent space dimension

Huge difference image-features

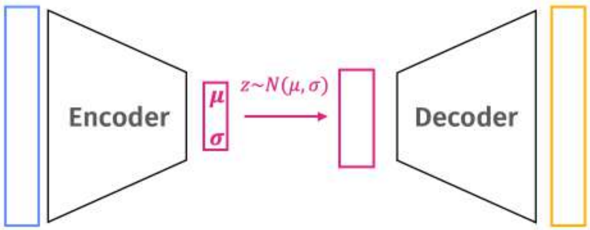


Results

Algorithm/Z	10	30	50	100	500
image-VAE	0.53 ± 0.017 (1.4 ± 0.5)	0.55 ± 0.04 (1.6 ± 0.49)	0.58 ± 0.01 (2.0 ± 0.63)	0.59 ± 0.01 (1.6 ± 0.48)	0.62 ± 0.01 (2.0 ± 0.0)
FE_{r_2}-VAE	0.98 ± 0.01 (0.0 ± 0.0)	0.98 ± 0.03 (0.0 ± 0.0)	0.98 ± 0.01 (0.0 ± 0.0)	0.98 ± 0.02 (0.0 ± 0.0)	0.98 ± 0.02 (0.0 ± 0.0)

Z: latent space dimension

Huge difference image-features



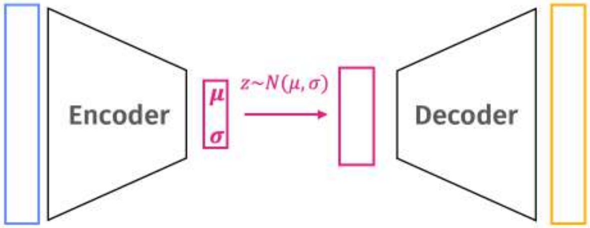
Z relevant?

Results

Algorithm/Z	10	30	50	100	500
image-VAE	0.53 ± 0.017 (1.4 ± 0.5)	0.55 ± 0.04 (1.6 ± 0.49)	0.58 ± 0.01 (2.0 ± 0.63)	0.59 ± 0.01 (1.6 ± 0.48)	0.62 ± 0.01 (2.0 ± 0.0)
FE_{r_2}-VAE	0.98 ± 0.01 (0.0 ± 0.0)	0.98 ± 0.03 (0.0 ± 0.0)	0.98 ± 0.01 (0.0 ± 0.0)	0.98 ± 0.02 (0.0 ± 0.0)	0.98 ± 0.02 (0.0 ± 0.0)

Z: latent space dimension

Huge difference image-features



Z relevant? Yes, in fine-grained datasets

Representation efficiency, conclusions

Representation efficiency, conclusions

- Pretrained features & Variational Auto Encoders, effective tool

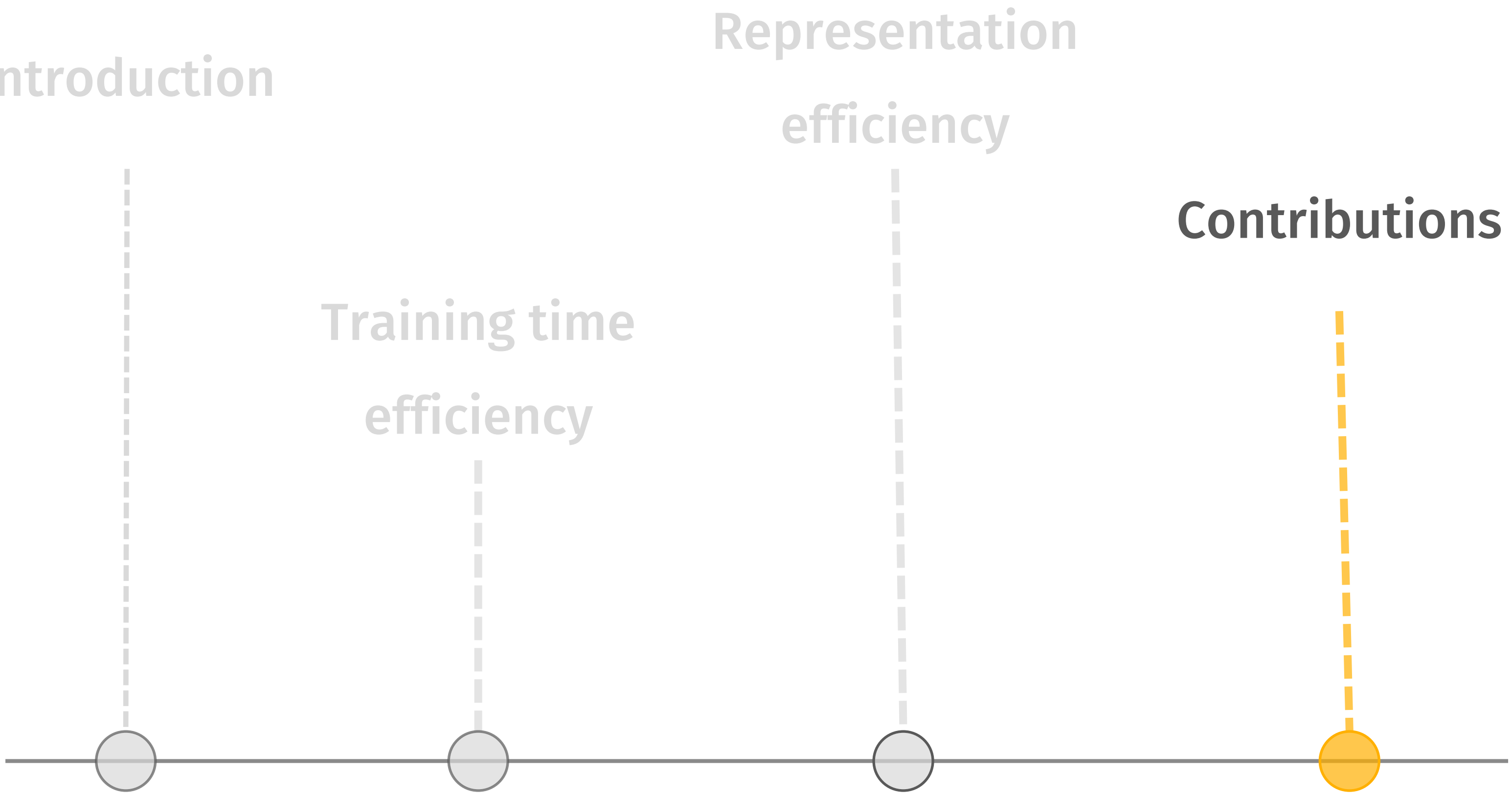
Representation efficiency, conclusions

- Pretrained features & Variational Auto Encoders, effective tool
- Reduced size, good for embedded devices

Representation efficiency, conclusions

- Pretrained features & Variational Auto Encoders, effective tool
- Reduced size, good for embedded devices
- Unsupervised pipeline

Outline


Introduction
Training time efficiency
Representation efficiency
Contributions

Contributions

Contributions

- Training time efficiency:

  Top-tuning outperforming fine-tuning

Contributions

- Training time efficiency:

  Top-tuning outperforming fine-tuning

- Representation efficiency:

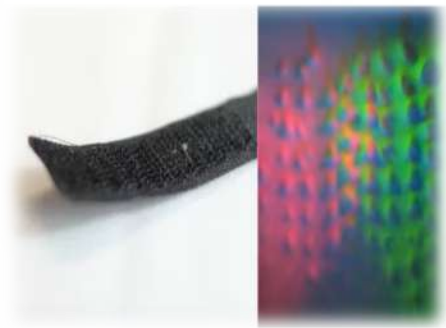
  Clustering for embedded devices

Developments

Real-time

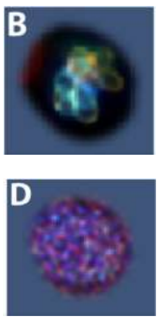
touch via vision

[Lambeta et al. 2020]



Scalable synthetic

cells engineering



Embedded pose and

action recognition

[Hachiuma et al. 2023]

Publications

Fine-tuning or top-tuning? A study on transfer learning with image pre-trained features and fast kernel methods, Alfano, Pastore, Rosasco, Odone
Under revision @IMAVIS Journal

Efficient Unsupervised Learning for Plankton Images, Alfano, Rando, Letizia, Pastore, Rosasco, Odone
Published @ICPR 2022

An unsupervised learning approach to resolve phenotype to genotype mapping in budding yeasts vacuoles, Alfano, Pastore
Under revision @ICIAP conference 2023

UniGe

MaLGa

References

[Dosovitskiy et al. 2020]: *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*

[Hachiuma et al. 2023]: *Unified Keypoint-based Action Recognition Framework via Structured Keypoint Pooling*

[Kingma and Welling 2014]: *Auto-Encoding Variational Bayes*

[Kornblith et al 2018]: *Do better ImageNet models transfer better?*

[Krizhevsky et al. 2012]: *ImageNet Classification with Deep Con- volutional Neural Networks*

[Lambeta et al. 2020]: *DIGIT: A Novel Design for a Low-Cost Compact High-Resolution Tactile Sensor with Application to In-Hand Manipulation*

[Moro et al. 2022]: *Markerless vs. Marker-Based Gait Analysis: A Proof of Concept Study*

[Strubell et al. 2019]: *Energy and Policy Considerations for Deep Learning in NLP*

[Russakovsky et al 2015]: *Imagenet large scale visual recognition challenge*

[Russel and Norvig 2020]: *Artificial Intelligence: A Modern Approach, fourth edition*

[Zhuang et al 2021]: *A Comprehensive Survey on Transfer Learning*