Development of a Sky Condition Classifier Using All-Sky Images

Mingyu Jeon, Rizchel Masong, Yi-Chieh Chang

meme explainer: <u>the sky when there is a cool astronomical event</u>.

A3N 2025 (18-22 Aug 2025)
KIAS, Seoul, South Korea.

Motivation

Higashi-Hiroshima Observatory

Photo: https://dive-hiroshima.com/en/explore/1685/

DESI

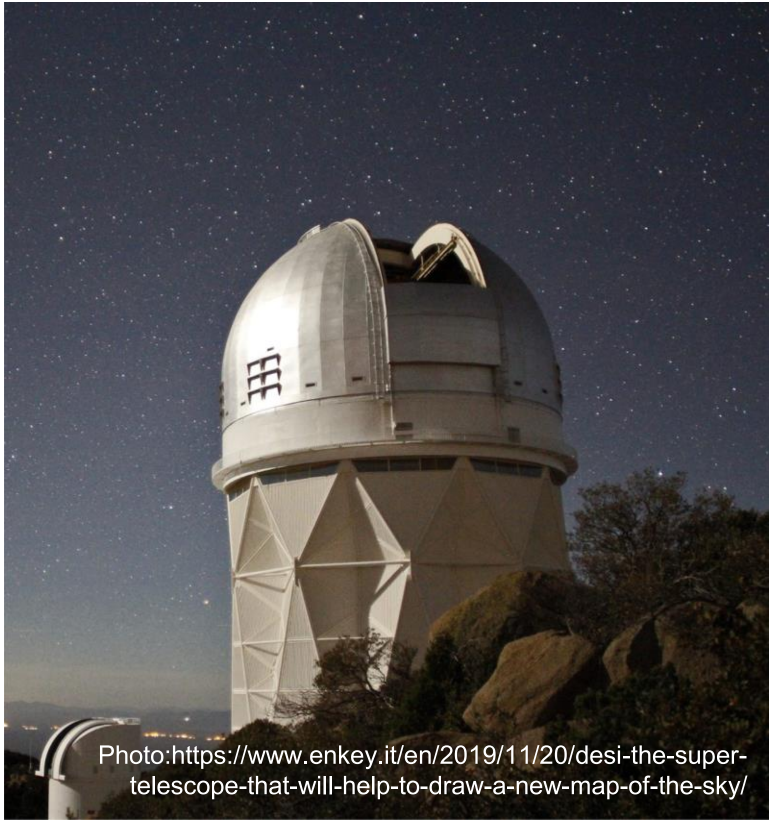


Photo:https://www.enkey.it/en/2019/11/20/desi-the-super-telescope-that-will-help-to-draw-a-new-map-of-the-sky/

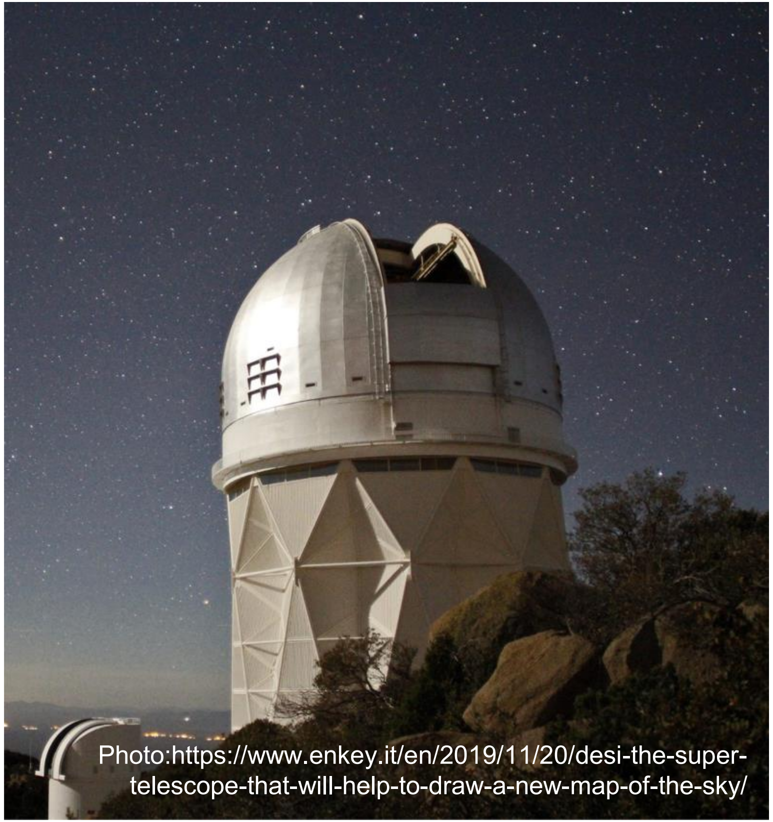
Motivation

Our data

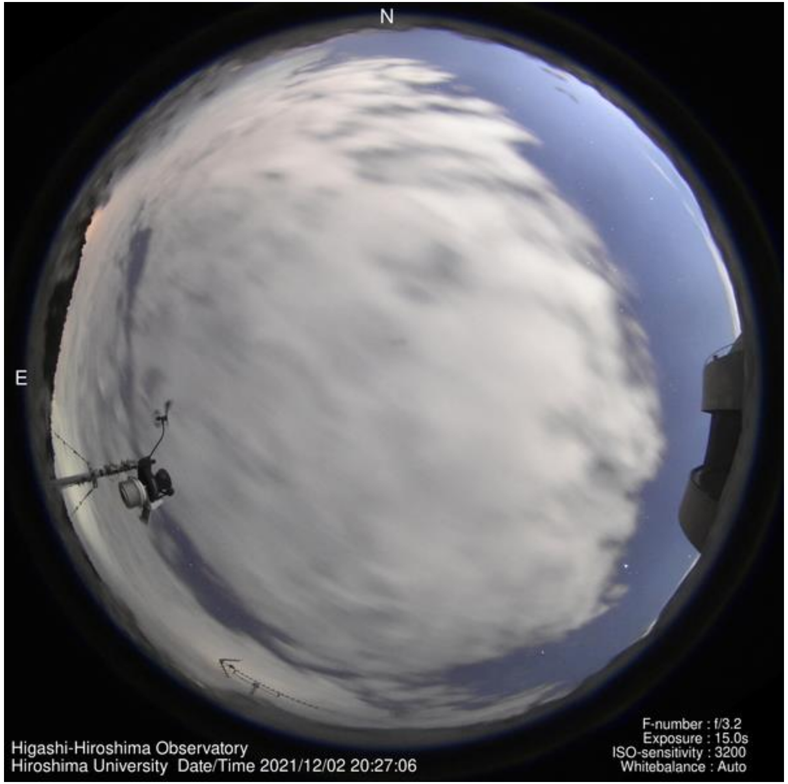
Higashi-Hiroshima Observatory

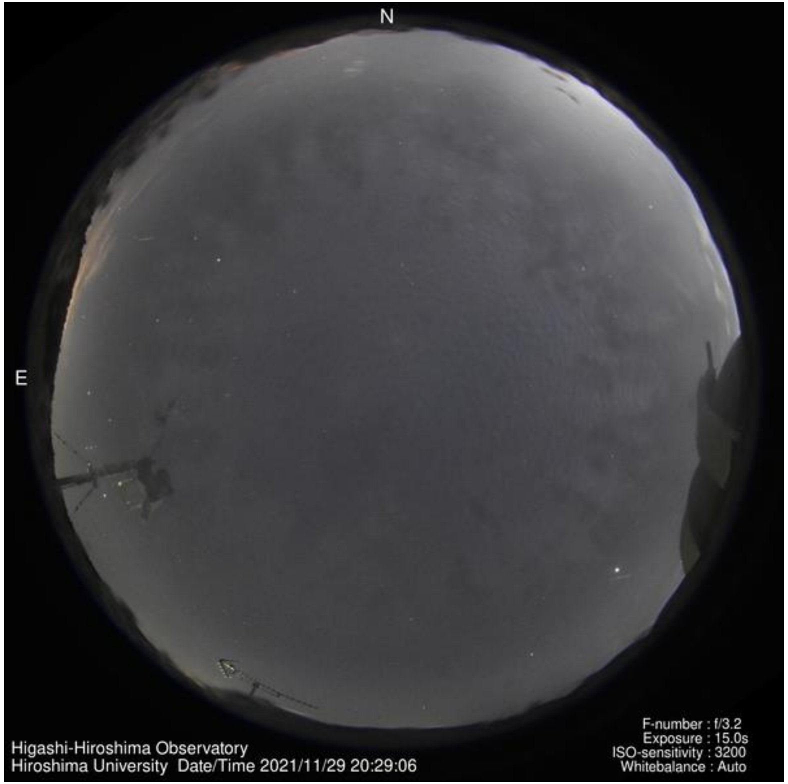
Photo: https://dive-hiroshima.com/en/explore/1685/

DESI

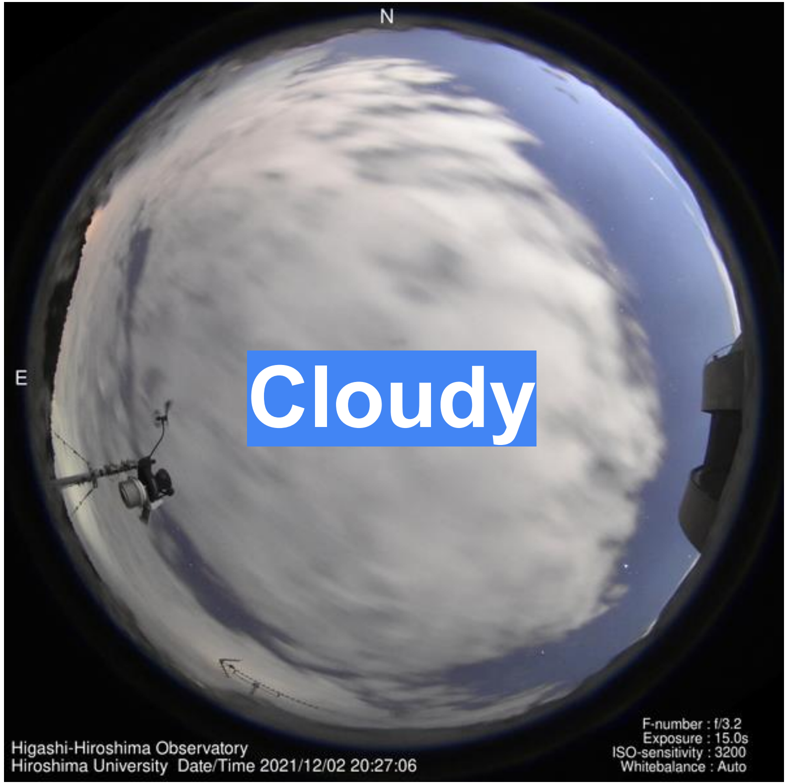


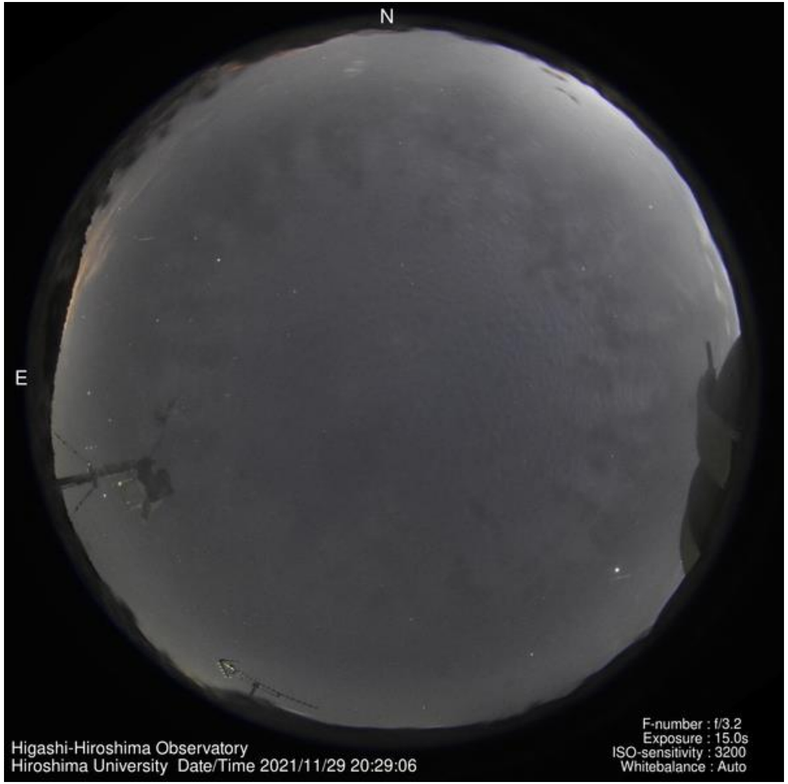
Motivation


N
E
F-number : f/3.2
Exposure : 15.0s
ISO-sensitivity : 3200
Whitebalance : Auto
Higashi-Hiroshima Observatory
Hiroshima University Date/Time 2021/12/02 20:27:06

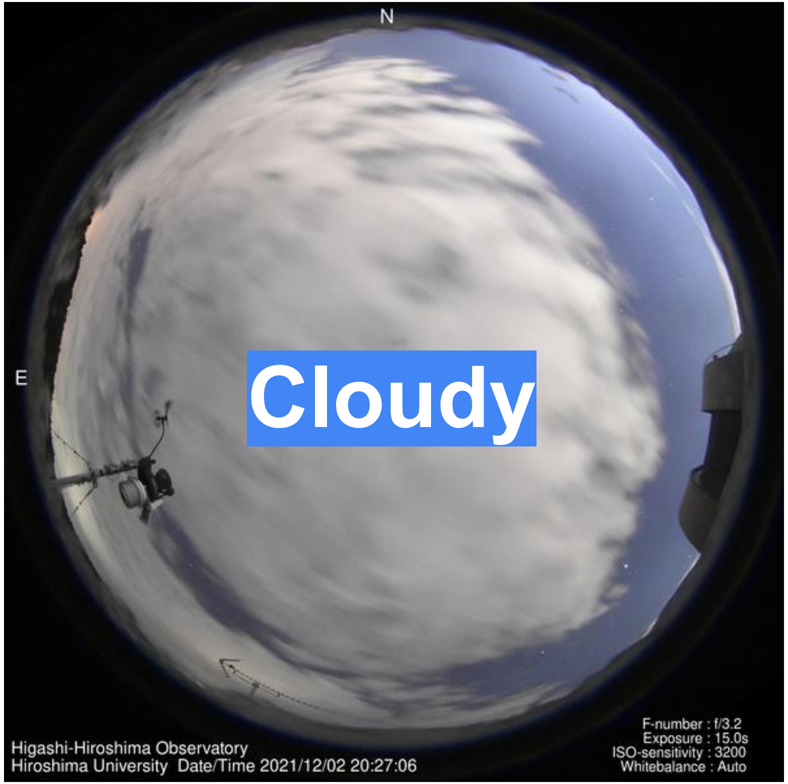

N
E
F-number : f/3.2
Exposure : 15.0s
ISO-sensitivity : 3200
Whitebalance : Auto
Higashi-Hiroshima Observatory
Hiroshima University Date/Time 2021/11/29 20:29:06

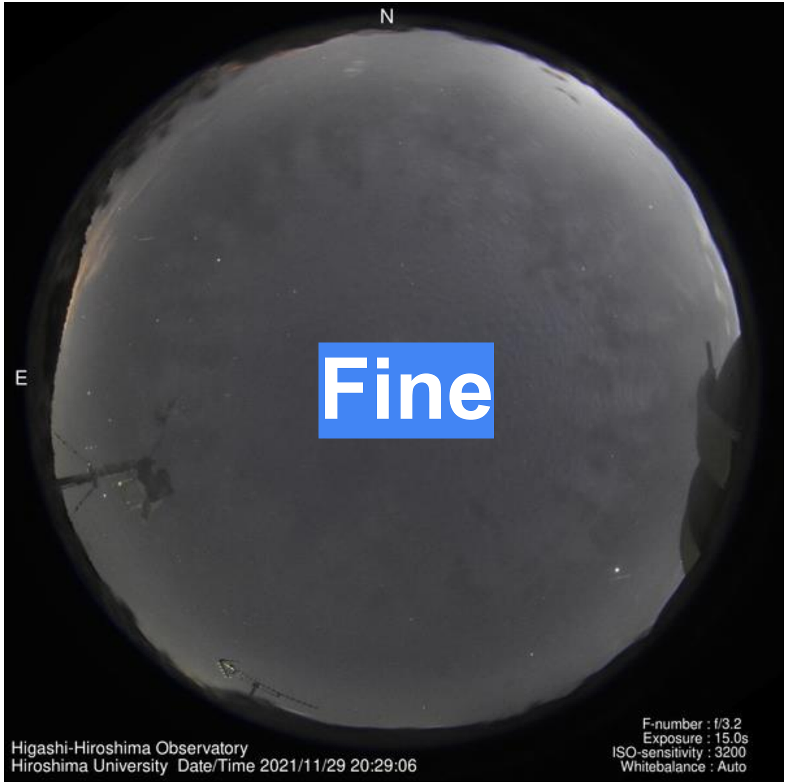
Motivation


N
E
Cloudy
Higashi-Hiroshima Observatory
Hiroshima University Date/Time 2021/12/02 20:27:06
F-number : f/3.2
Exposure : 15.0s
ISO-sensitivity : 3200
Whitebalance : Auto


N
E
Higashi-Hiroshima Observatory
Hiroshima University Date/Time 2021/11/29 20:29:06
F-number : f/3.2
Exposure : 15.0s
ISO-sensitivity : 3200
Whitebalance : Auto

Motivation


N
E
Cloudy
Higashi-Hiroshima Observatory
Hiroshima University Date/Time 2021/12/02 20:27:06
F-number : f/3.2
Exposure : 15.0s
ISO-sensitivity : 3200
Whitebalance : Auto


N
E
Fine
Higashi-Hiroshima Observatory
Hiroshima University Date/Time 2021/11/29 20:29:06
F-number : f/3.2
Exposure : 15.0s
ISO-sensitivity : 3200
Whitebalance : Auto

https://hasc.hiroshima-u.ac.jp/environ/skymon_summary.html

Environment Monitor at Higashi-Hiroshima Observatory

Current Time (This PC)
2025/08/21 (Thu) 12:27:28

Local Sidereal Time
-

METEOROGRAM						
	Time	Temperature (degs C)	Humidity (%)	Wind Dir.	Wind Sp. (m/s)	Rain (mm/5min)
Outside	12:10	+28.1	77	SSE	2.2	0.0
Dome Inside	12:05	+22.5	47	-	-	-

RAINDROP SENSOR (at 12:31:04)				TELESCOPE Temp (at 12:24:00)		
North		South		TopRingS	CenSectS	PillMidE
Level 0	DRY	Level 0	DRY	+25.2	+22.6	+23.6

TELESCOPE (at 12:16:02)			
Control PC	Position	GRB mode	GRB obs status
OFF	Azimuth - Altitude -	OFF	-

DOME		
DomeSlit	MirrorCover	Azimuth
CLOSED	CLOSED	+0.0 (-)

Link

- [Meteogram of HHO]
- [Hiroshima Astrophysical Science Center]
- Observatory monitoring camera
  [Rikubetsu(Ginganomori)] [Nayoro(Hokkaido Univ.)] [Gunma(GAO)] [Saitama Univ.] [Akeno(Tokyo-kogyo Univ.)] [Kyoto(KAO)] [Osaka(Osaka Kyoiku Univ.)] [Sayo(NHAO)] [Okayama(OAO)] [Bisei(JSGA)] [Iriki(Kagoshima Univ.)] [Sutherland(SuperWASP group)]

Method

- Data : 4,984 (Cloudy) + 3,822 (Fine) = 8,806 Images
- Binary Classification
- CNN

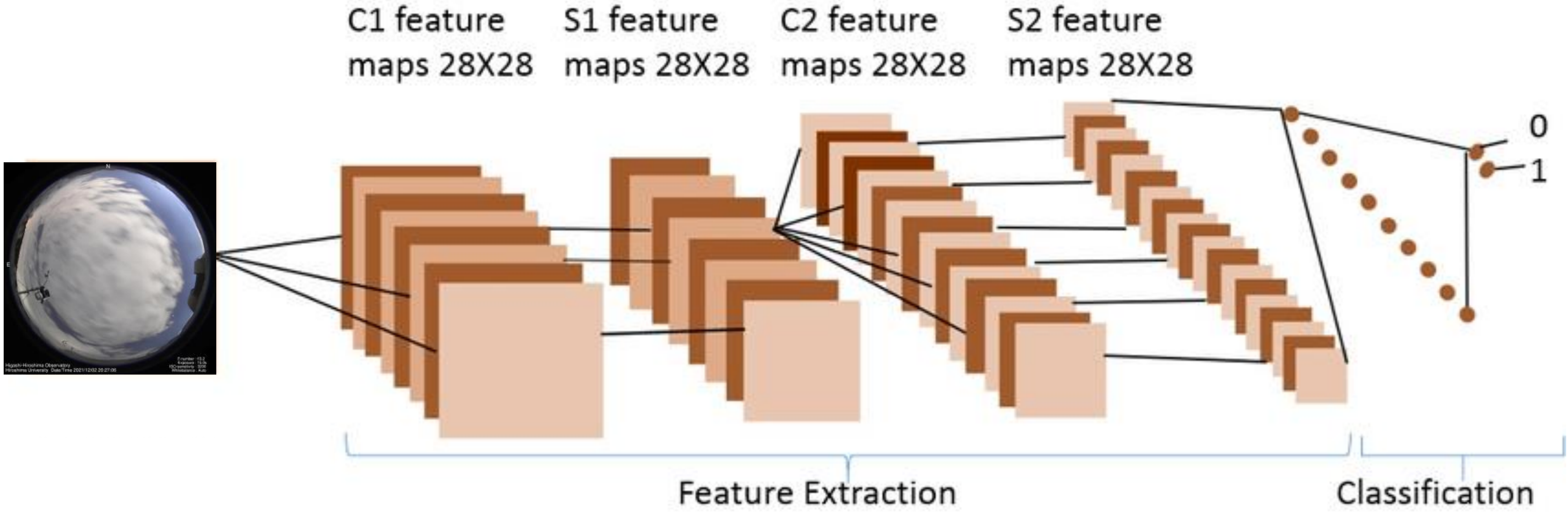


Photo: Bahram et al. (2017)

Method

- Data : 4,984 (Cloudy) + 3,822 (Fine) = 8,806 Images
- Binary Classification
- CNN **<u>= Feature Extractor + Classifier</u>**

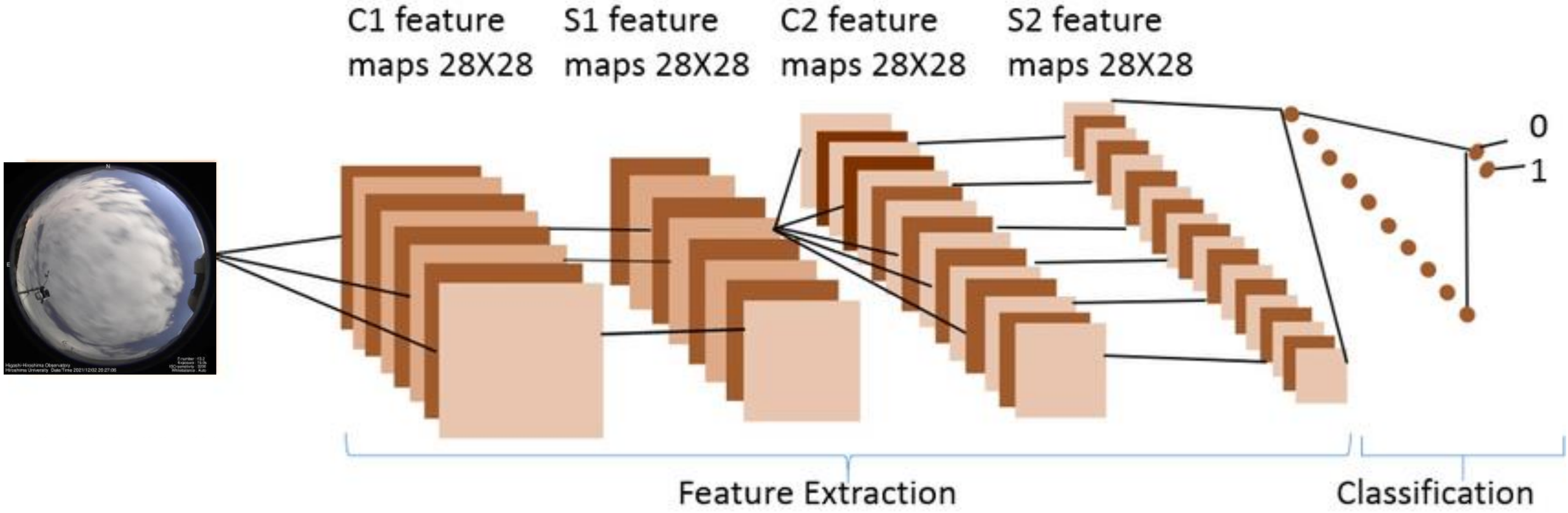
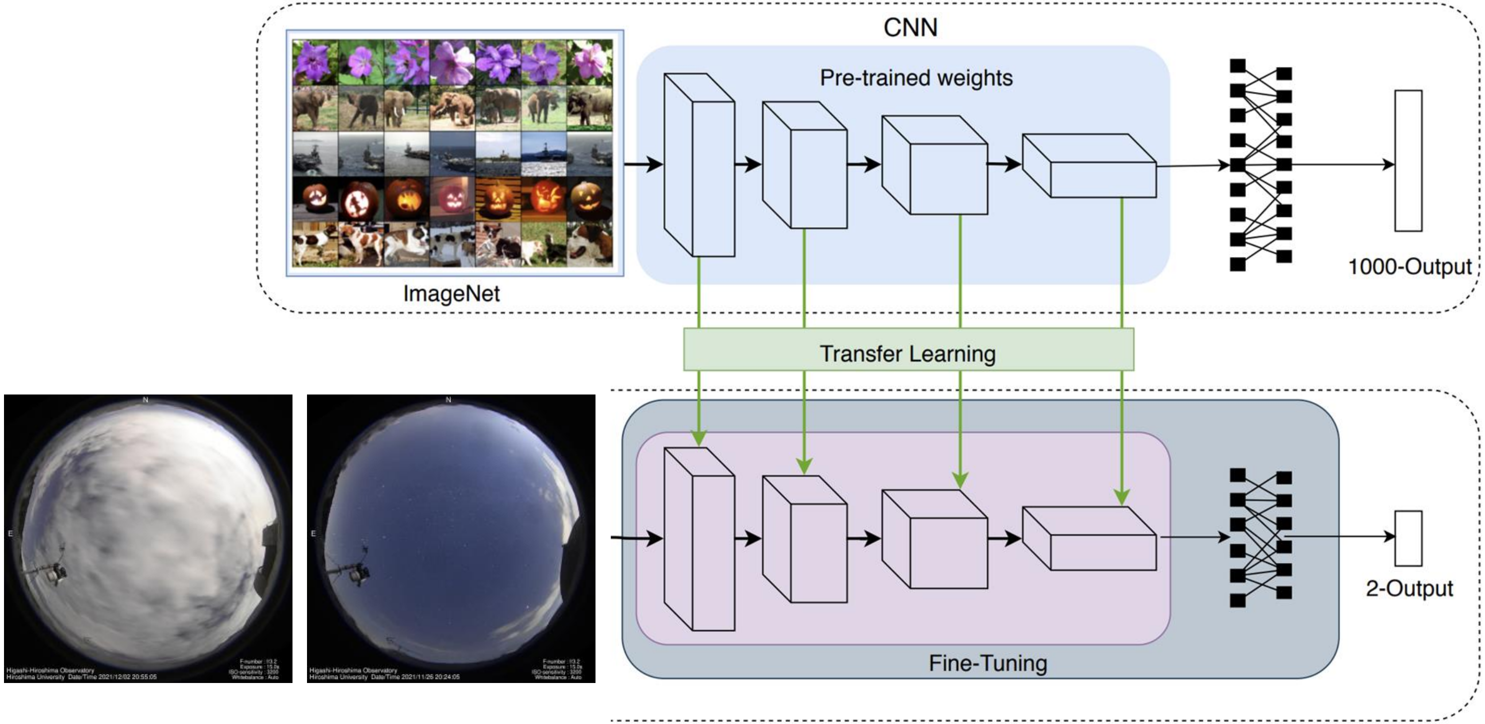


Photo: Bahram et al. (2017)

Transfer Learning

Idea: Pre-trained CNN layers can extract useful features from (almost) any image



Cloudy or Fine

Vogado et al., 2021 Sensors

A PyTorch library for pre-trained vision models

pytorch image models, 0.99

timm

pytorch image models

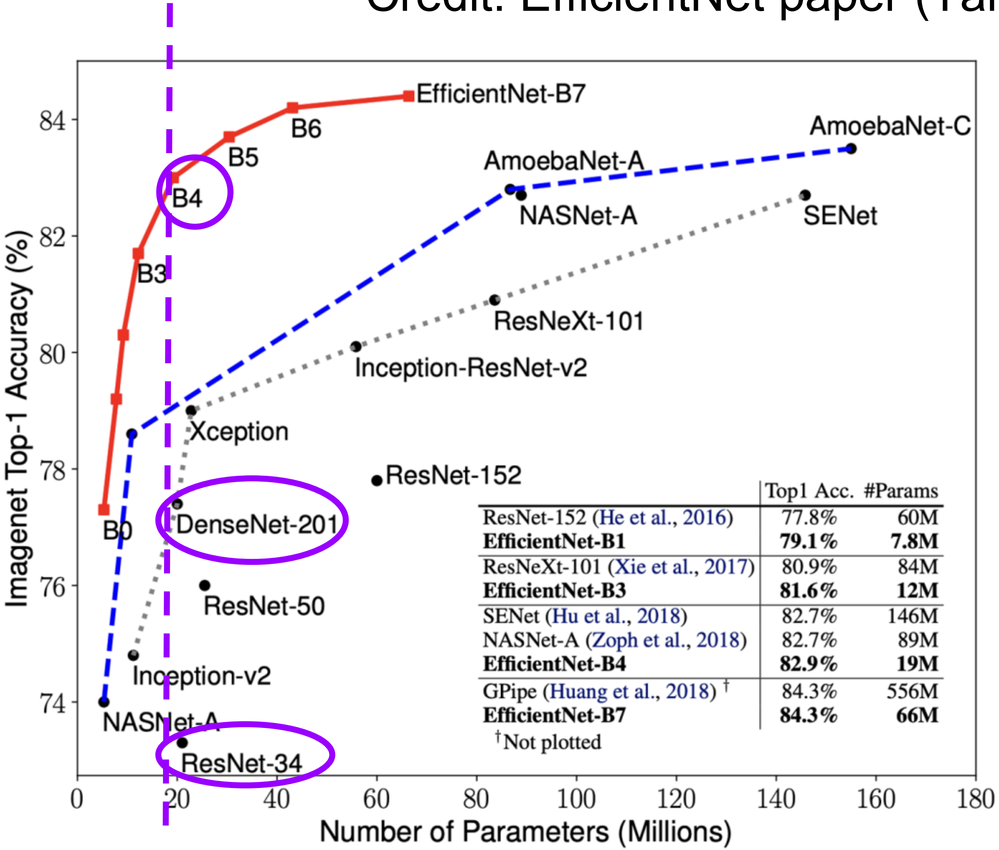
pre-trained vision models, collections and more!

Hugging Face

- Aggregating Nested Transformers - https://arxiv.org/abs/2105.12723
- BEiT - https://arxiv.org/abs/2106.08254
- BEiT-V2 - https://arxiv.org/abs/2208.06366
- BEiT3 - https://arxiv.org/abs/2208.10442
- Big Transfer ResNetV2 (BiT) - https://arxiv.org/abs/1912.11370
- Bottleneck Transformers - https://arxiv.org/abs/2101.11605
- CaiT (Class-Attention in Image Transformers) - https://arxiv.org/abs/2103.17239
- CoaT (Co-Scale Conv-Attentional Image Transformers) - https://arxiv.org/abs/21(
- CoAtNet (Convolution and Attention) - https://arxiv.org/abs/2106.04803
- ConvNeXt - https://arxiv.org/abs/2201.03545
- ConvNeXt-V2 - http://arxiv.org/abs/2301.00808
- ConViT (Soft Convolutional Inductive Biases Vision Transformers)- https://arxiv.o
- CspNet (Cross-Stage Partial Networks) - https://arxiv.org/abs/1911.11929
- DeiT - https://arxiv.org/abs/2012.12877
- DeiT-III - https://arxiv.org/pdf/2204.07118.pdf
- DenseNet - https://arxiv.org/abs/1608.06993
- DLA - https://arxiv.org/abs/1707.06484
- DPN (Dual-Path Network) - https://arxiv.org/abs/1707.01629
- EdgeNeXt - https://arxiv.org/abs/2206.10589
- EfficientFormer - https://arxiv.org/abs/2206.01191
- EfficientFormer-V2 - https://arxiv.org/abs/2212.08059
- EfficientNet (MBConvNet Family)
  - EfficientNet NoisyStudent (B0-B7, L2) - https://arxiv.org/abs/1911.04252
  - EfficientNet AdvProp (B0-B8) - https://arxiv.org/abs/1911.09665
  - EfficientNet (B0-B7) - https://arxiv.org/abs/1905.11946
  - EfficientNet-EdgeTPU (S, M, L) - https://ai.googleblog.com/2019/08/efficient

Model

Credit: EfficientNet paper (Tan & Le, 2019 arXiv)



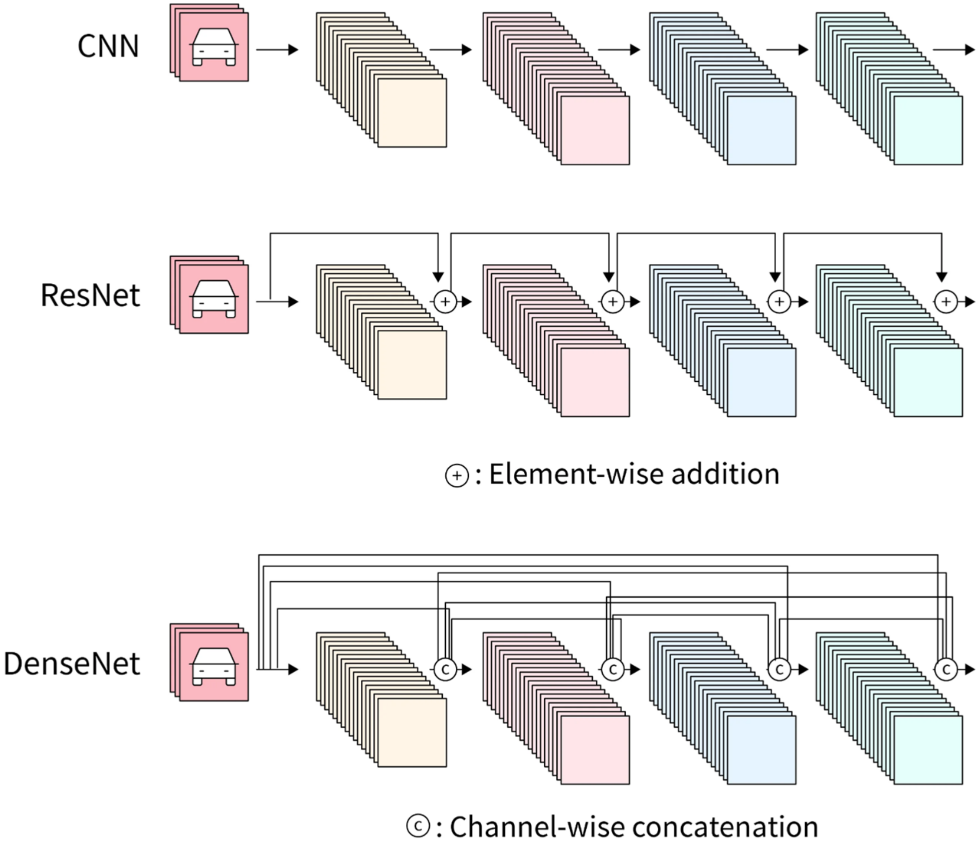
Similar #parameters !

We use 3 models with ~20M params

EfficientNet-B4	17.6M
DenseNet-201	18.1M
ResNet-34	21.3M

※ params of only CNN layers

Model Explanation: ResNet & DenseNet

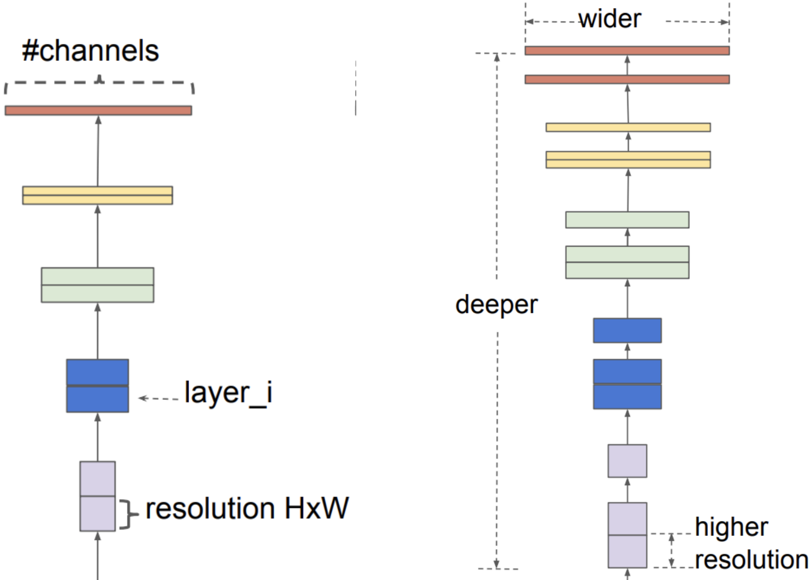


ResNet-34
= ResNet with 34 layers

DenseNet-201
= DenseNet with 201 layers

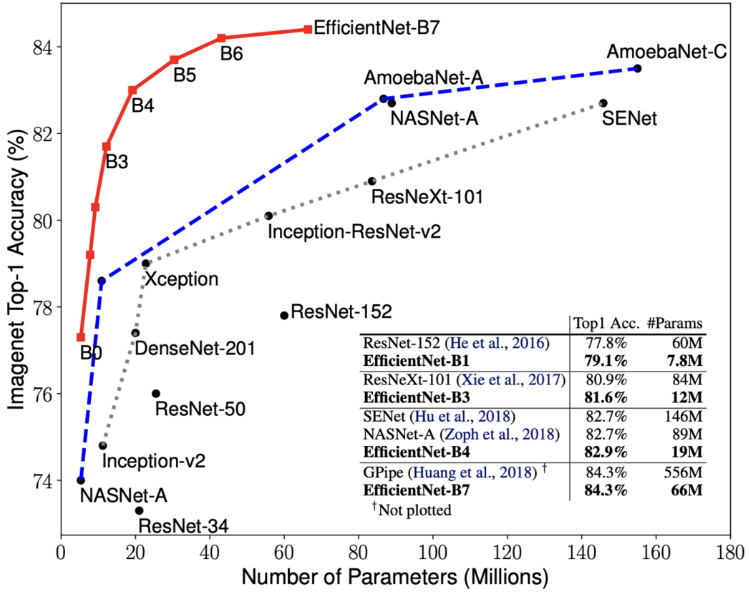
Model Explanation: EfficientNet

(1) Find a baseline architecture using Neural Architecture Search
(2) Scale up the baseline architecture using Compound Scaling



(a) baseline

(e) compound scaling



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

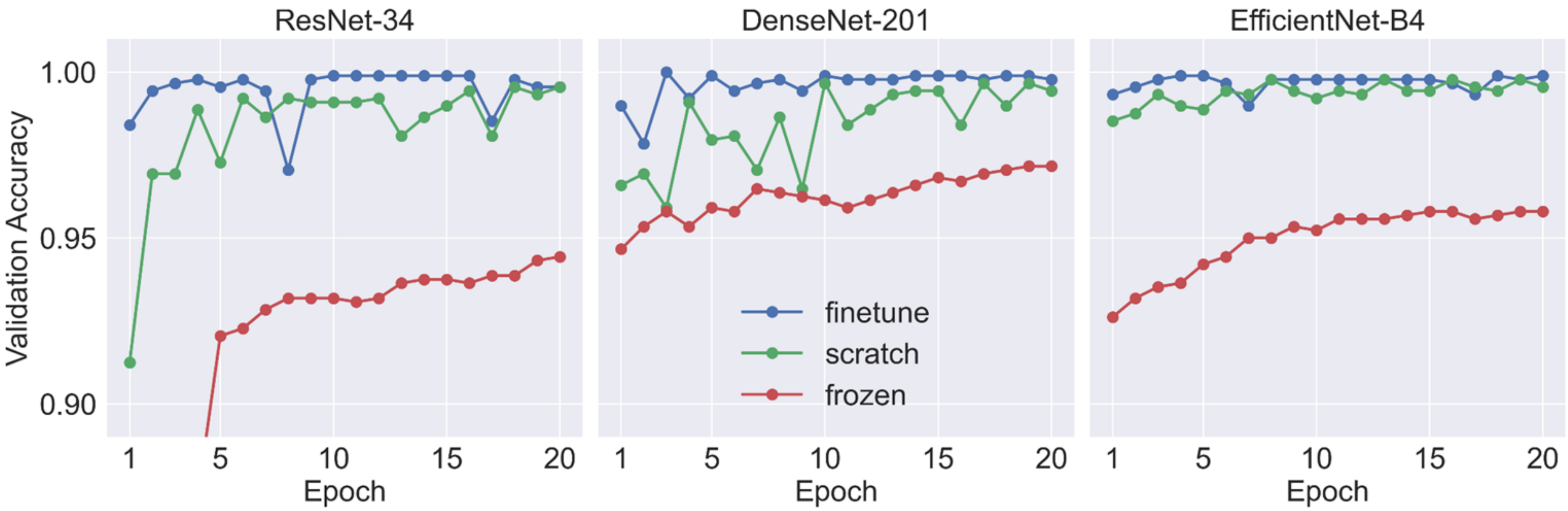
†Not plotted

EfficientNet-B4
= EfficientNet with compound scaling stage 4

$$\text{depth: } d = \alpha^\phi$$
$$\text{width: } w = \beta^\phi$$
$$\text{resolution: } r = \gamma^\phi$$
$$\text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$$
$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

Results (1): Finetune > Scratch > Frozen

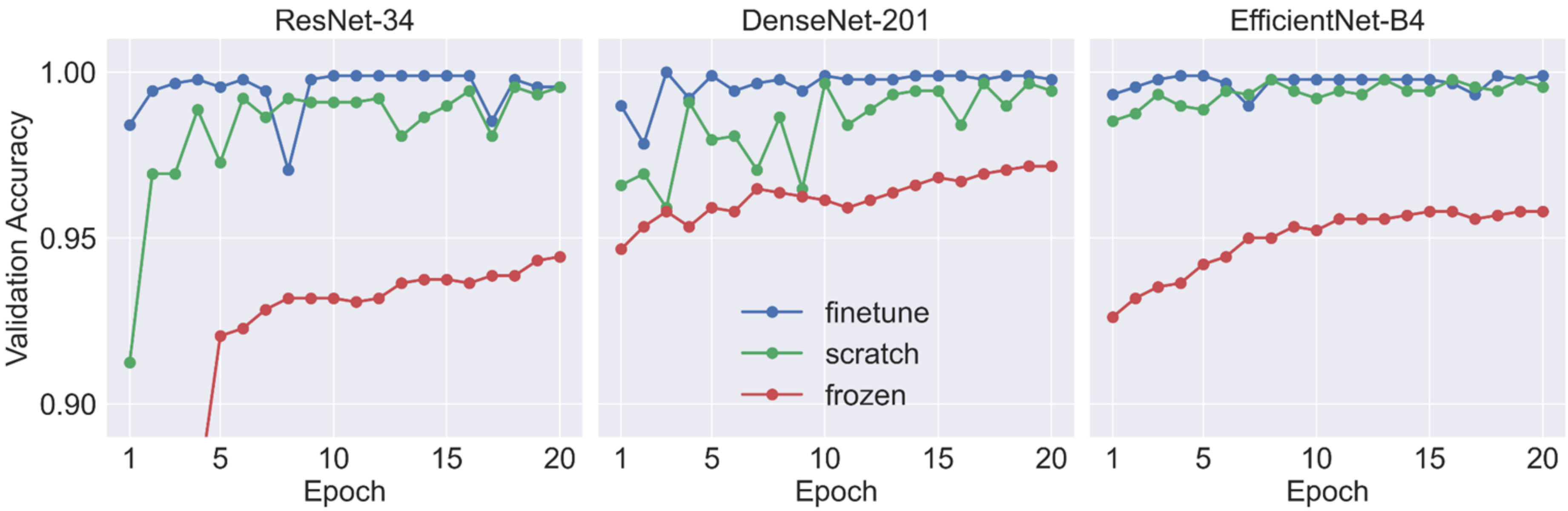
Scratch	Train the model from random initialization
Frozen	Freeze feature extractor layers of a pre-trained model and only train the final classifier
Finetune	Start with a pre-trained model and update all layers on the new dataset



Results (1): Finetune > Scratch > Frozen

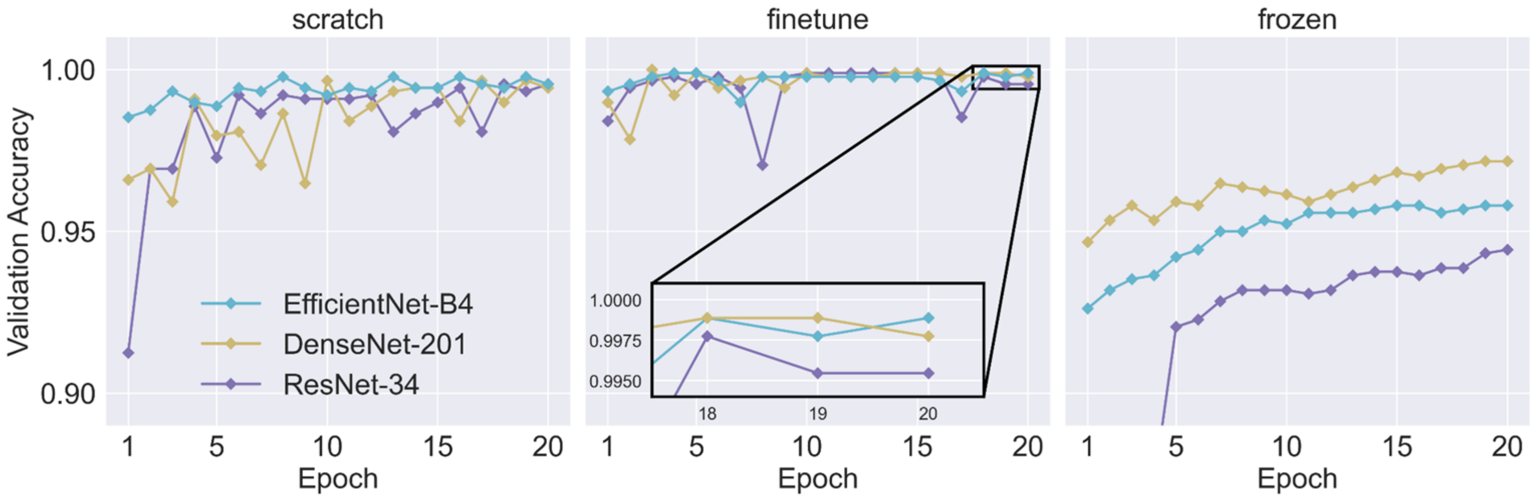
Why 'Frozen' is worst?

- Train only a linear classifier (output = W * feature vector + b)
- Test if features extracted from pre-trained CNN are linearly separable for our data
- Perform worst since features extracted from pre-trained CNN are not necessarily linearly separable for our data

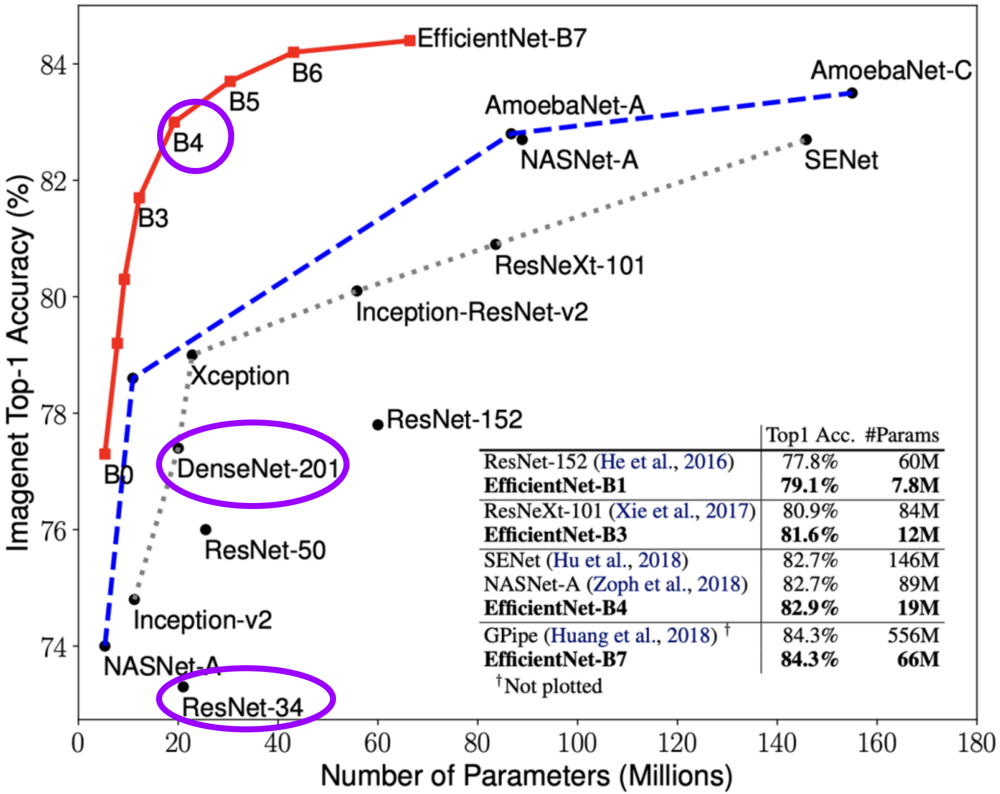


Results (2): EfficientNet > DenseNet > ResNet

Model (# params)	Test Accuracy (best epoch) best epoch = best validation accuracy		
	scratch	finetune	frozen
EfficientNet-B4 (17.6M)	**1.0000**	**0.9989**	0.9524
DenseNet-201 (18.1M)	0.9966	0.9977	**0.9705**
ResNet-34 (21.3M)	0.9966	0.9977	0.9388



Results (2): EfficientNet > DenseNet > ResNet



EfficientNet-B4	17.6M
DenseNet-201	18.1M
ResNet-34	21.3M

※ params of only CNN layers

License MIT | Tests passing | Monthly Downloads 36k | Total Downloads 1M

Advanced AI explainability for PyTorch

`pip install grad-cam`

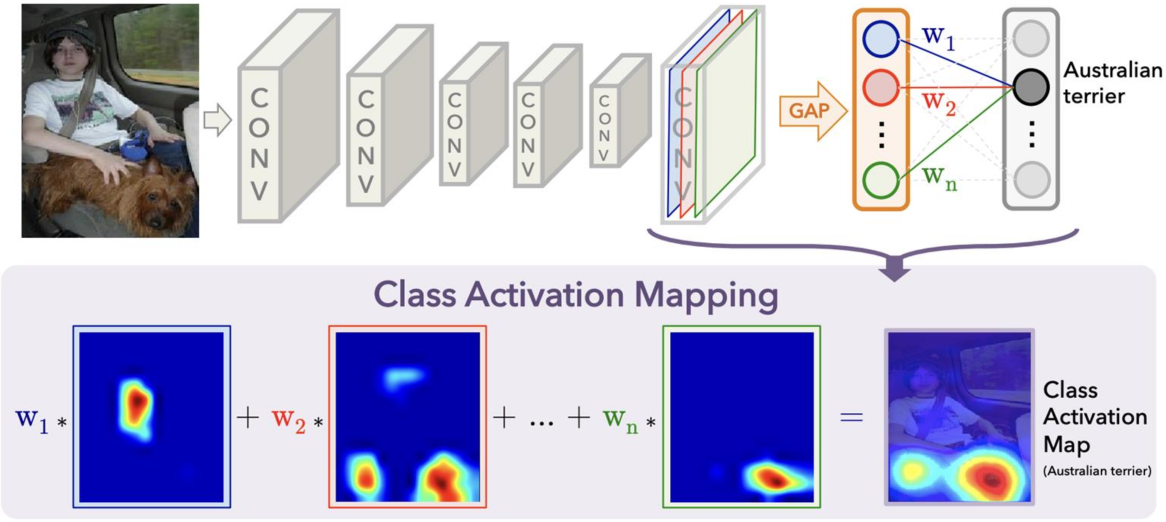
Documentation with advanced tutorials: https://jacobgil.github.io/pytorch-gradcam-book

Explainable AI: CAM-based methods

Class Activation Map (CAM)

Idea: If a CNN classifier works well, the last convolutional feature maps usually contain class-specific spatial information.

→ CAM-based methods (e.g., Grad-CAM, Eigen-CAM, etc.) visualize this information in various ways.



Zhou et al., 2015 arXiv

Method
GradCAM
HiResCAM
GradCAMElementWise
GradCAM++
XGradCAM
AblationCAM
ScoreCAM
EigenCAM
EigenGradCAM
LayerCAM
FullGrad
Deep Feature Factorizations
KPCA-CAM

Grad-CAM (Gradient-weighted Class Activation Map)

Grad-CAM for a class c

ReLU → only positive influence… pixels whose intensity is increased to increase score y^c

$$L^c_{\text{Grad-CAM}} = \text{ReLU}\underbrace{\left(\sum_k \alpha^c_k A^k\right)}_{\text{linear combination}}$$

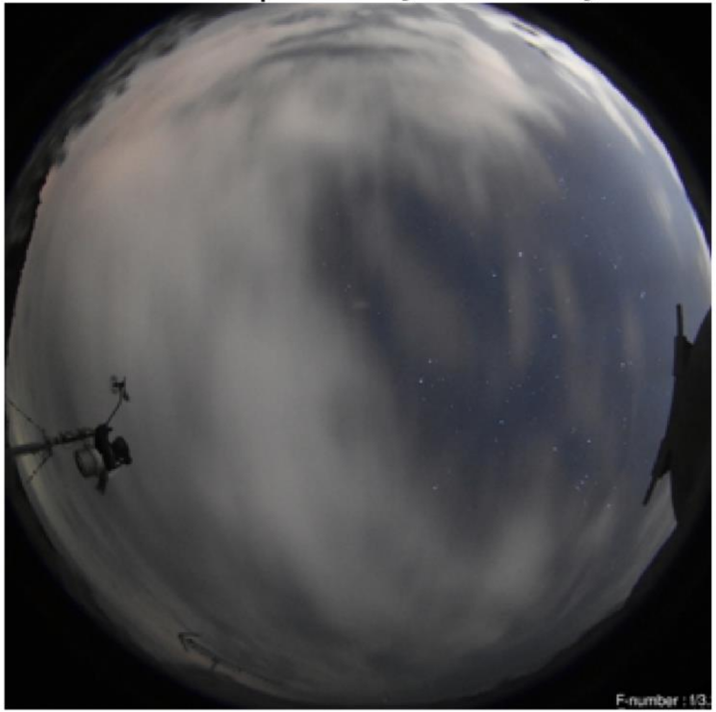
α^c_k is the "importance" of feature map k, A^k, for the class c

$$\alpha^c_k = \overbrace{\frac{1}{Z}\sum_i \sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A^k_{ij}}}_{\text{gradients via backprop}}$$

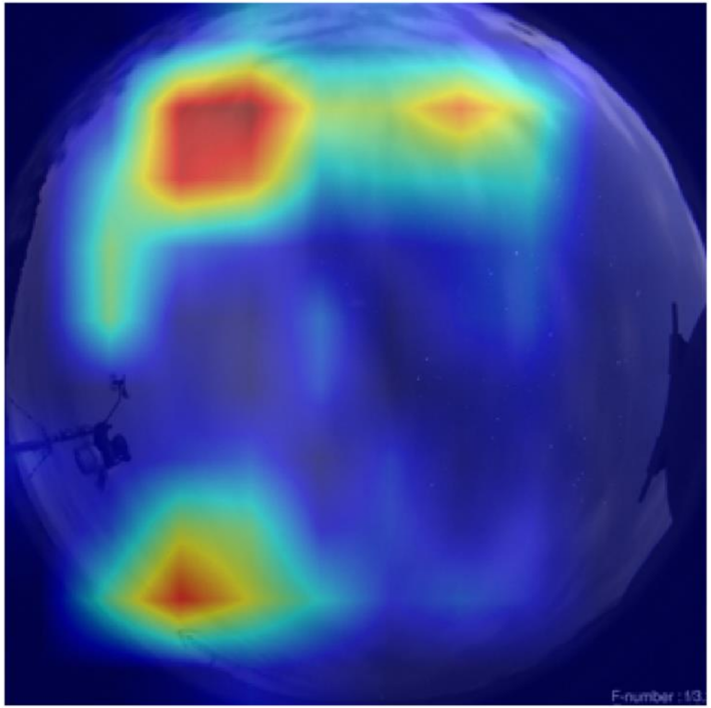
Selvaraju et al., 2016 arXiv / 2020 IJCV

Red pixels = higher contribution to the model’s prediction
Blue pixels = lower contribution to the model’s prediction

Actual / Prediction / Probability
EfficientNet-B4 | cloudy / cloudy / 100%

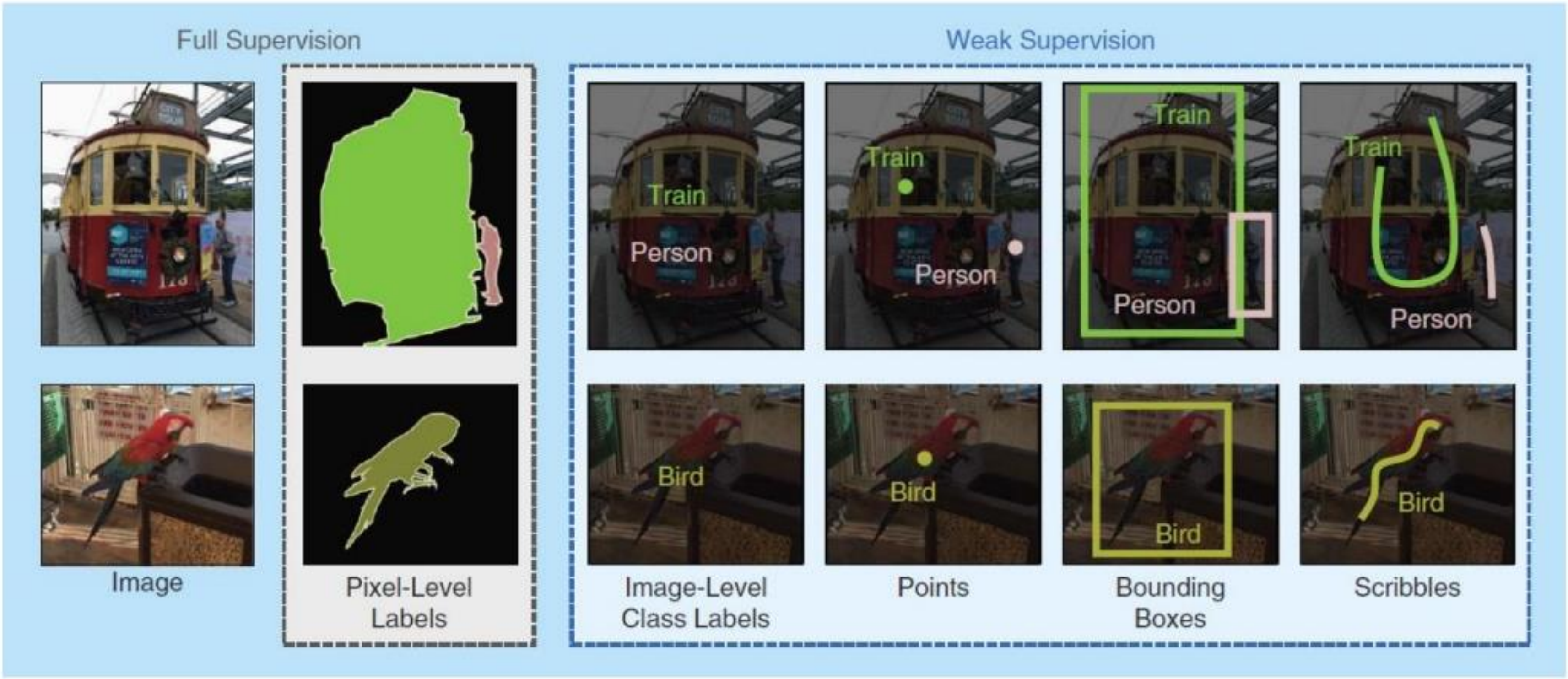


cloudy
Grad-CAM



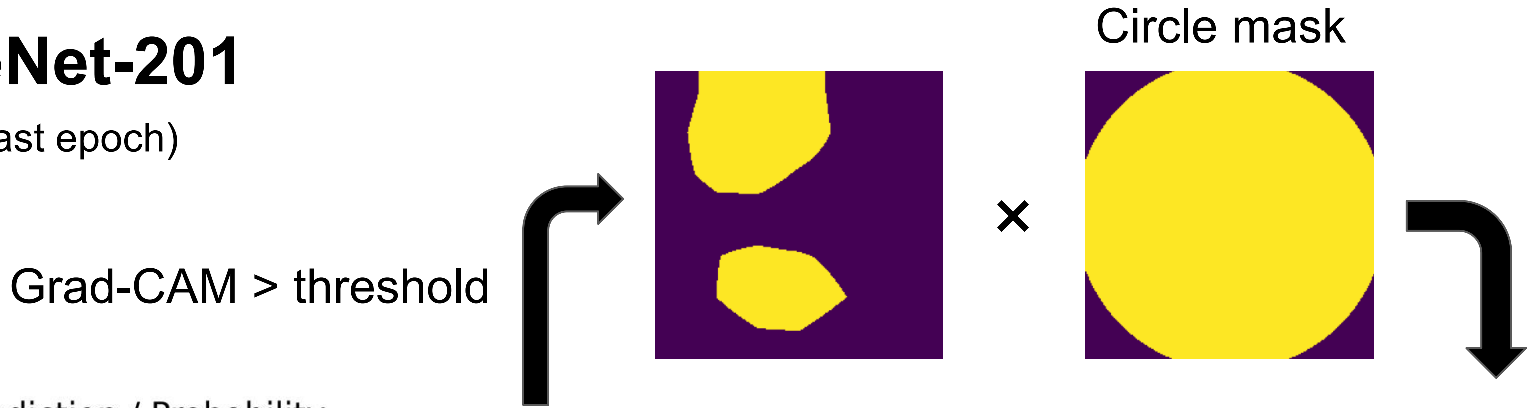
Weakly-supervised semantic segmentation

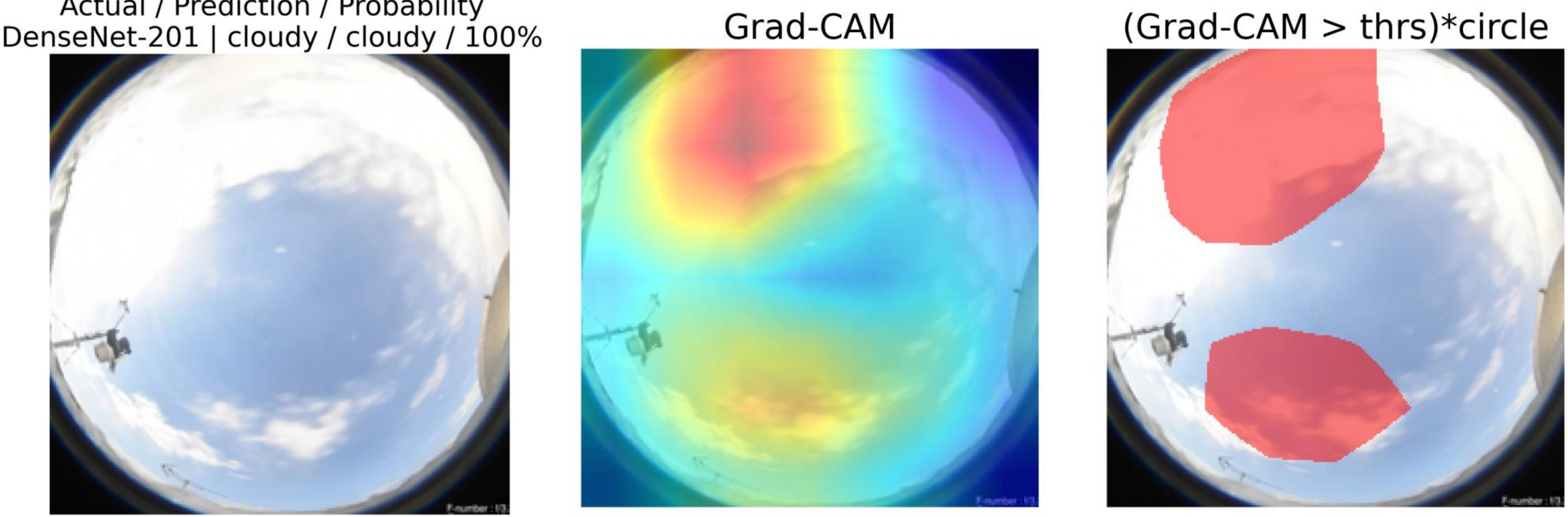
Can an image classifier trained only with image-level labels produce pixel-level labels?



DenseNet-201

(finetune, last epoch)

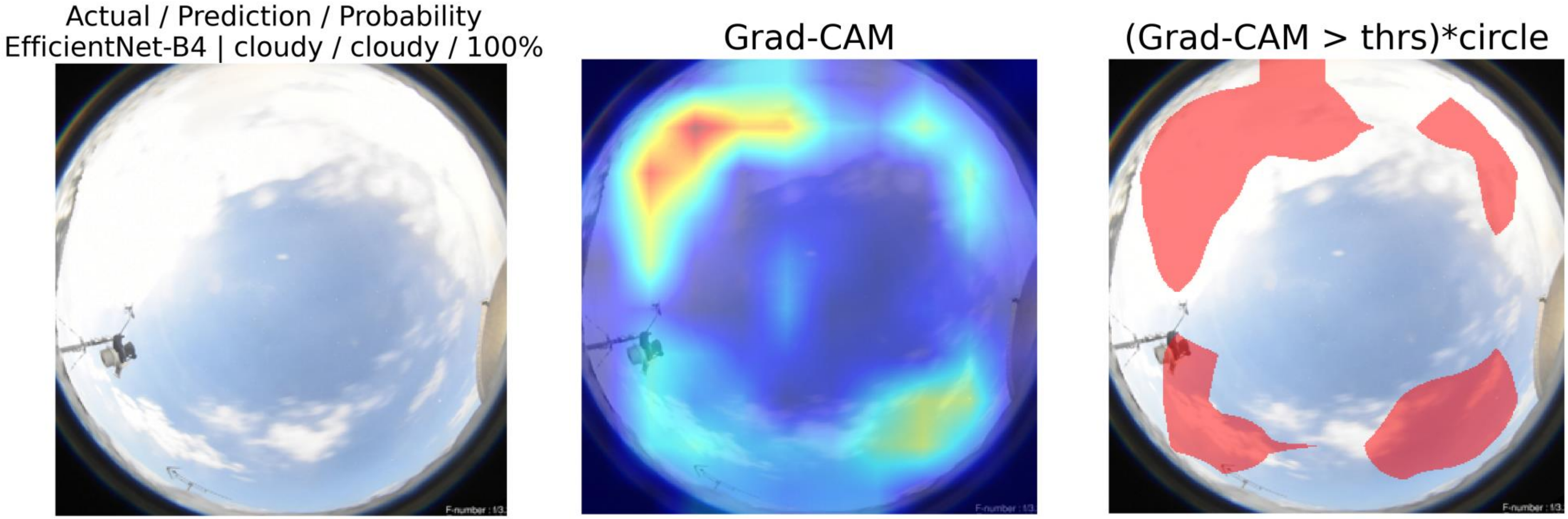




EfficientNet-B4

(finetune, last epoch)

For this image, EfficientNet-B4 detects more cloud regions (via Grad-CAM) than DenseNet-201.



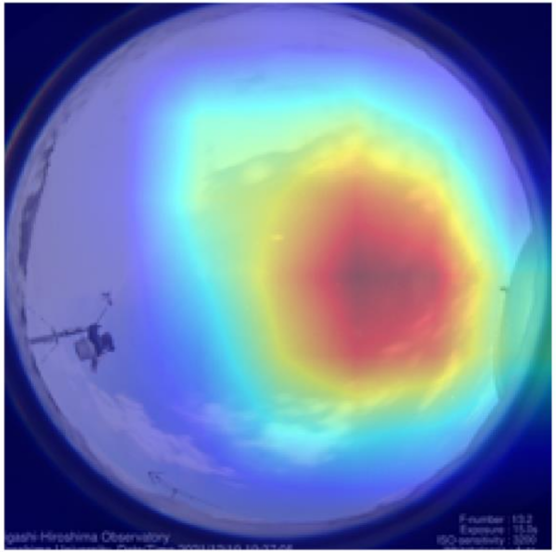
ResNet-34

(finetune, last epoch)

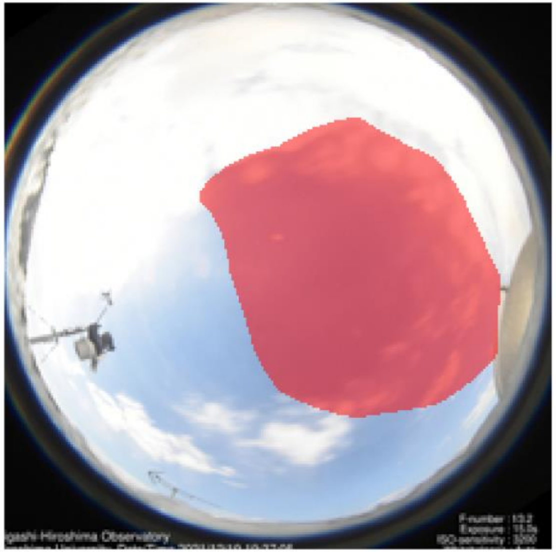
ResNet-34 uses different regions than DenseNet-201 and EfficientNet-B4 to classify the sky as cloudy

Actual / Prediction / Probability
ResNet-34 | cloudy / cloudy / 100%

Grad-CAM



(Grad-CAM > thrs)*circle



Summary and Future works

- Summary
  - CNN Architecture: EfficientNet > DenseNet > ResNet
  - Training Strategy: Fine-Tuning > From-Scratch > Frozen
  - Grad-CAM provides interpretability for CNN predictions
  - Grad-CAM results can be used for weakly-supervised semantic segmentation

- Future Works
  - Image augmentation (rotation, brightness, contrast, etc.)
  - Dataset expansion (different seasons, times of day)
  - Multi-class classification (e.g., partly cloudy, rainy)
  - Real-time processing (end-to-end pipeline)
  - Robustness testing (e.g., moonlight conditions)
  - Time-series prediction

Appendix

Binary Classification

Task: Data x → Two Discrete Labels y ∈ {0, 1}

1: cloudy
0: fine

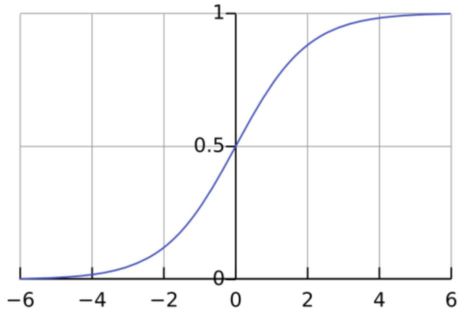
Assume the labels follow a Bernoulli distribution

$p(y|\lambda) = (1 - \lambda)^{1-y} \lambda^{y}$; λ = (probability that y=1)

Train a model $f_\theta(x)$ to predict the distribution parameter λ

$\lambda = \sigma(f_\theta(x))$; σ = sigmoid function from $\mathbb{R}$ to [0, 1]

sigmoid function



The loss function (**binary cross entropy**) is the negative log-likelihood of the training set

$$L(\theta) = -\sum_{i=1}^{N} \log p(y_i|\sigma(f_\theta(x_i)))$$
$$= -\sum_{i=1}^{N} (1 - y_i) \log[(1 - \sigma(f_\theta(x_i)))] + y_i \log[\sigma(f_\theta(x_i))]$$

Prince, 2023, “Understanding Deep Learning”

Detail

- Dataset
  - Total: 8,806 images (4,984 Cloudy + 3,822 Fine) 600x600 size
  - Split (8:1:1):
    - Cloudy: 3,987 / 498 / 499
    - Fine: 3,057 / 382 / 383
    - Total: 7,044 / 880 / 882 (Train / Val / Test)
  - Preprocessing: Resized and normalized using the pre-trained model's pipeline

- Training Setup
  - Epochs: 20
  - Batch size: 64
  - Learning rate: 2e-4

Pre-trained Models: ResNet-34

```
Compose(
    Resize(size=235, interpolation=bicubic, max_size=None, antialias=True)
    CenterCrop(size=(224, 224))
    MaybeToTensor()
    Normalize(mean=tensor([0.4850, 0.4560, 0.4060]), std=tensor([0.2290, 0.2240, 0.2250]))
)
```

timm/**resnet34.a1_in1k** | like 0 | Follow PyTorch Image Models 537

Image Classification | timm | PyTorch | Safetensors | Transformers | arxiv:21

Model card | Files and versions | Community

Model card for resnet34.a1_in1k

A ResNet-B image classification model.

This model features:

- ReLU activations
- single layer 7x7 convolution with pooling
- 1x1 convolution shortcut downsample

Trained on ImageNet-1k in `timm` using recipe template described below.

Pre-trained Models: DenseNet-201

```
Compose(
    Resize(size=256, interpolation=bicubic, max_size=None, antialias=True)
    CenterCrop(size=(224, 224))
    MaybeToTensor()
    Normalize(mean=tensor([0.4850, 0.4560, 0.4060]), std=tensor([0.2290, 0.2240, 0.2250]))
)
```

timm / **densenet201.tv_in1k** like 0 Follow PyTorch Image Models 537

Image Classification | timm | PyTorch | Safetensors | Transformers | imagenet-1k

Model card Files and versions Community

Edit

Model card for densenet201.tv_in1k

A DenseNet image classification model. Trained on ImageNet-1k (original torchvision weights).

Pre-trained Models: EfficientNet-B4

```
Compose(
    Resize(size=365, interpolation=bicubic, max_size=None, antialias=True)
    CenterCrop(size=(320, 320))
    MaybeToTensor()
    Normalize(mean=tensor([0.4850, 0.4560, 0.4060]), std=tensor([0.2290, 0.2240, 0.2250]))
)
```

timm timm/**efficientnet_b4.ra2_in1k** like 0 Follow timm PyTorch Image Models 5

Image Classification | timm | PyTorch | Safetensors | Transformers | imagenet-1k

Model card Files and versions Community

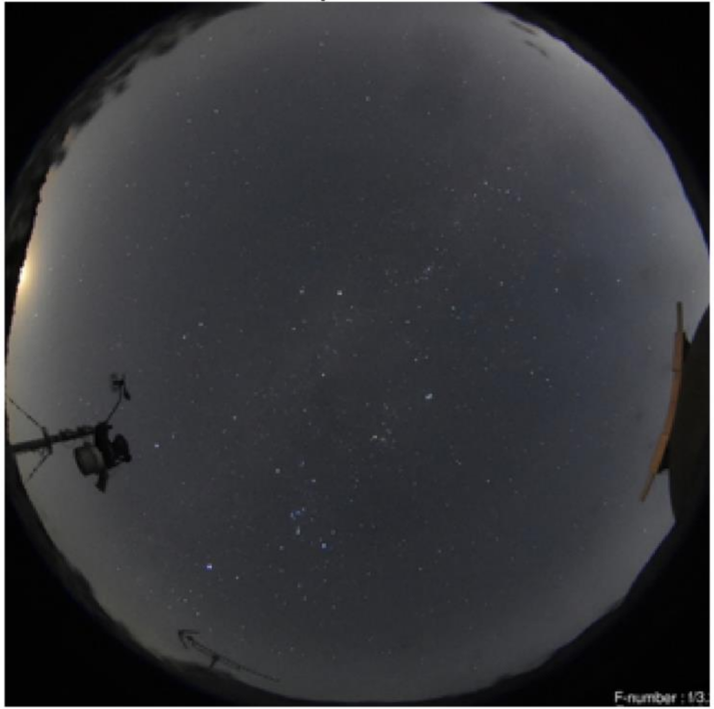
Edit mode

Model card for efficientnet_b4.ra2_in1k

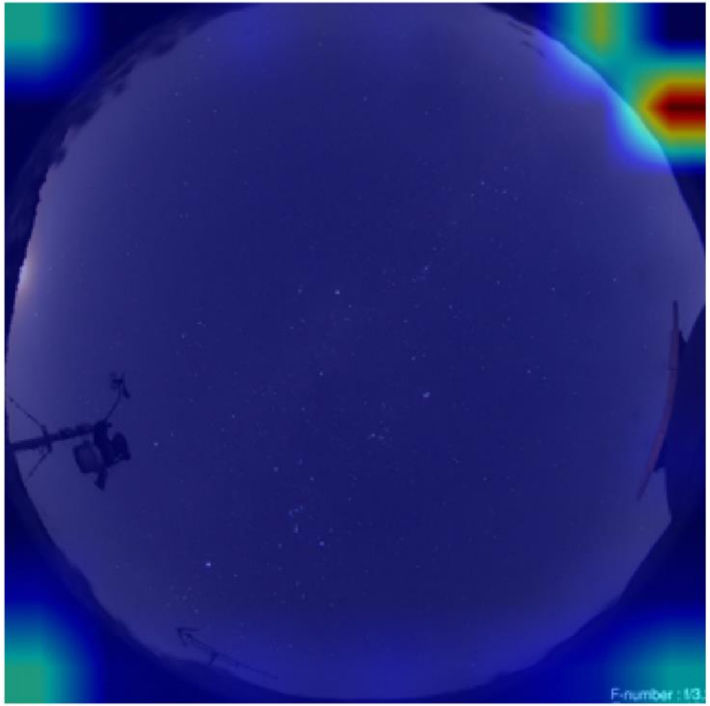
A EfficientNet image classification model. Trained on ImageNet-1k in `timm` using recipe template described below.

Red pixels = higher contribution to the model’s prediction
Blue pixels = lower contribution to the model’s prediction

Actual / Prediction / Probability
EfficientNet-B4 | fine / fine / 100%



fine
Grad-CAM



Wrong prediction (TODO)