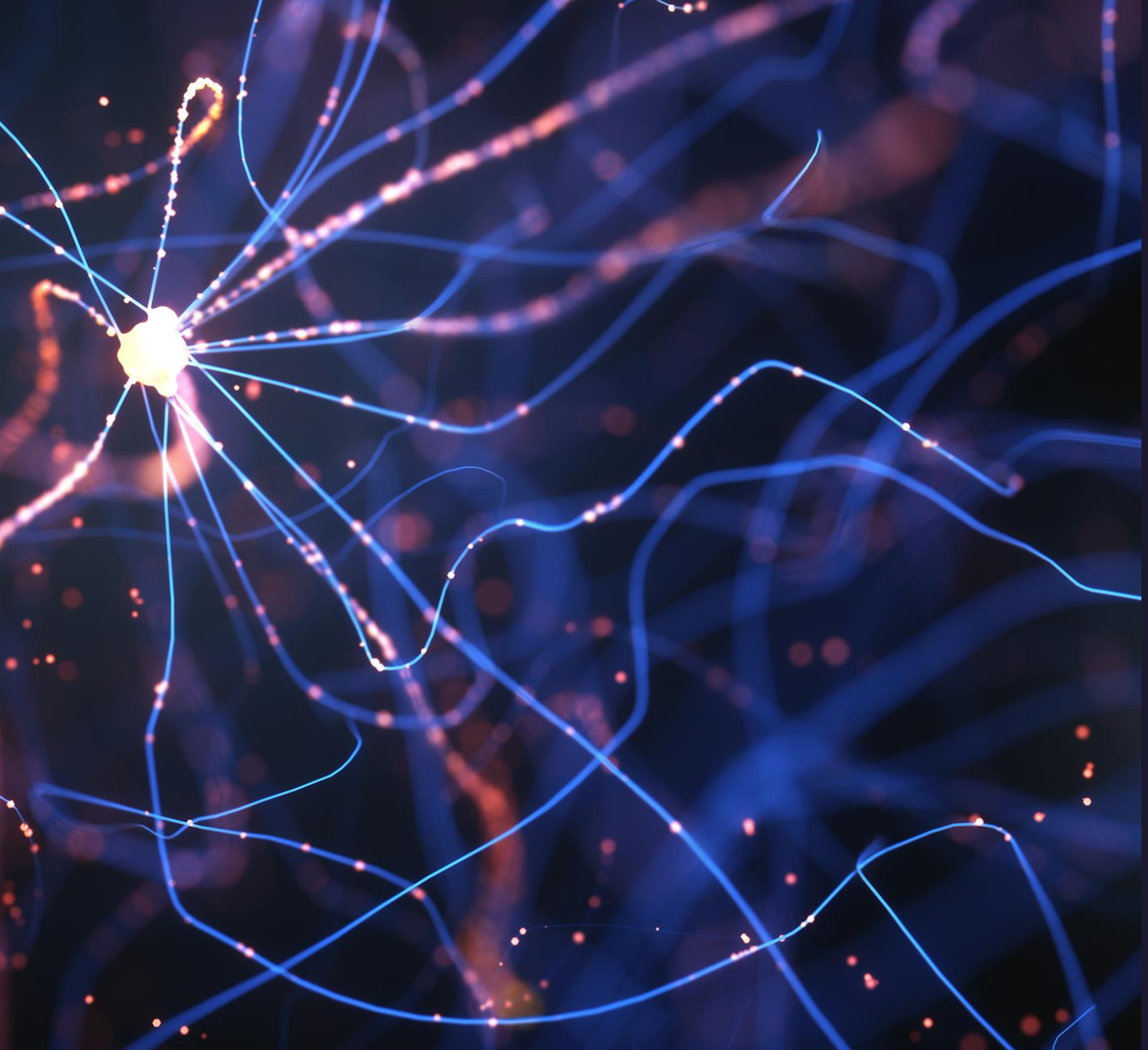


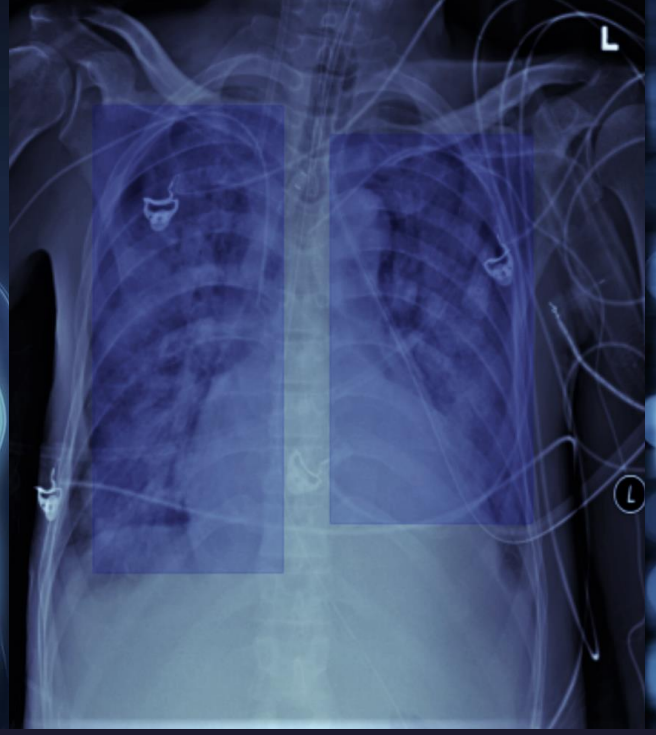
Pneumonia Detection with RetinaNet

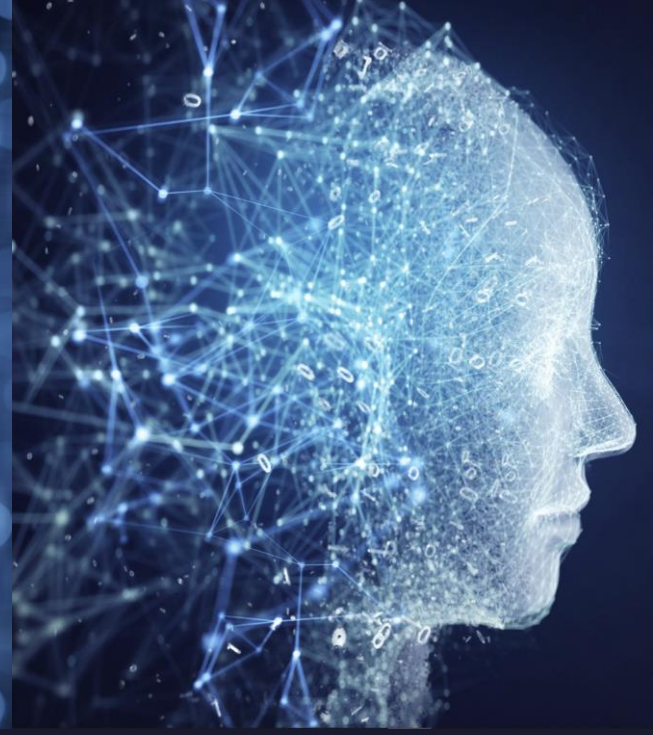
Manuel Ivagnes

Riccardo Bianchini

Valerio Coretti

La Sapienza University of Rome



Background

Pneumonia is an inflammatory condition of the lung primarily affecting the small air sacs known as alveoli. Symptoms typically include some combination of productive or dry cough, chest pain, fever and difficulty breathing. The severity of the condition is variable. Pneumonia is usually caused by infection with viruses or bacteria, and less commonly by other microorganisms. Identifying the responsible pathogen can be difficult. Diagnosis is often based on symptoms and physical examination. Chest X-rays, blood tests, and culture of the sputum may help confirm the diagnosis. - wikipedia

Identifying cases of Pneumonia is tedious and often leads to a disagreement between radiologists. However, computer-aided diagnosis systems showed the potential for improving diagnostic accuracy.

Input data

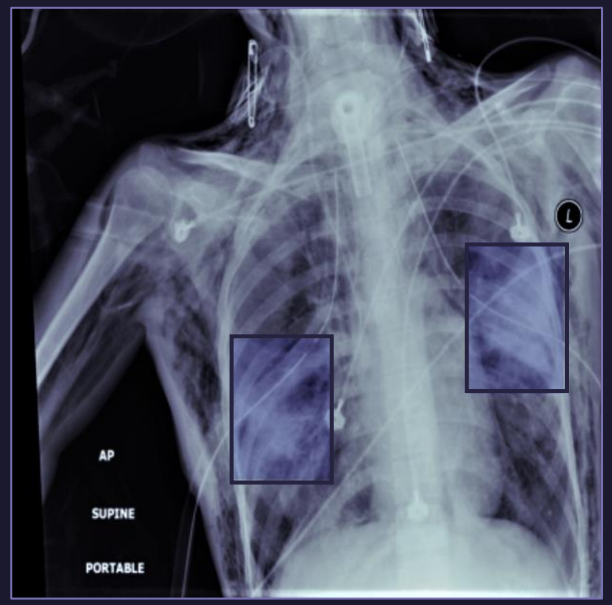
Dataset Overview

Preprocessing & Augmentations

The dataset was publicly provided by the **US National Institutes of Health Clinical Center**. It comprises frontal-view X-ray images from **26684** unique patients. Each image was labelled with one of three different classes from the associated radiological reports:

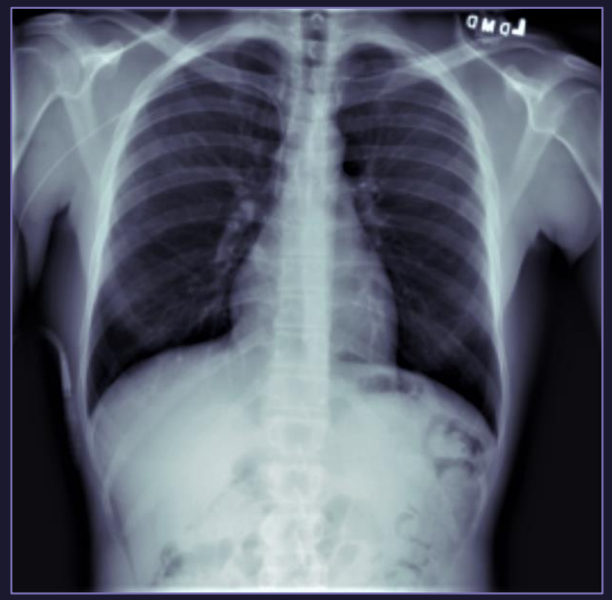
LUNG OPACITY

- fuzzy clouds of white in the lungs, associated with pneumonia



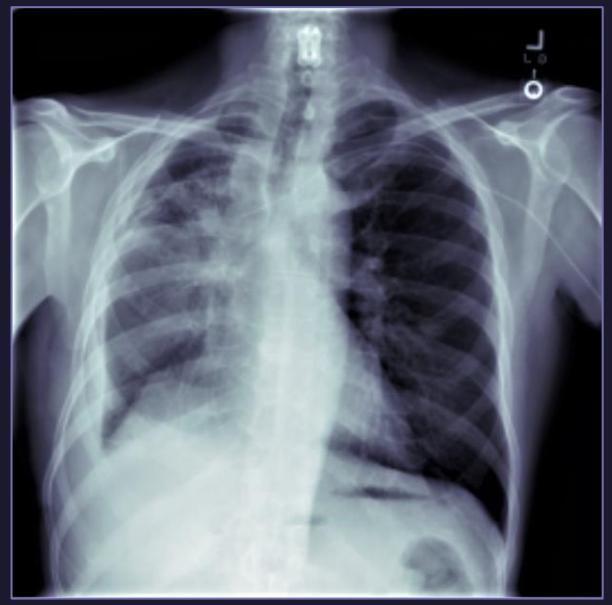
NORMAL

- healthy patients without any pathologies found



NO LUNG OPACITY / NOT NORMAL

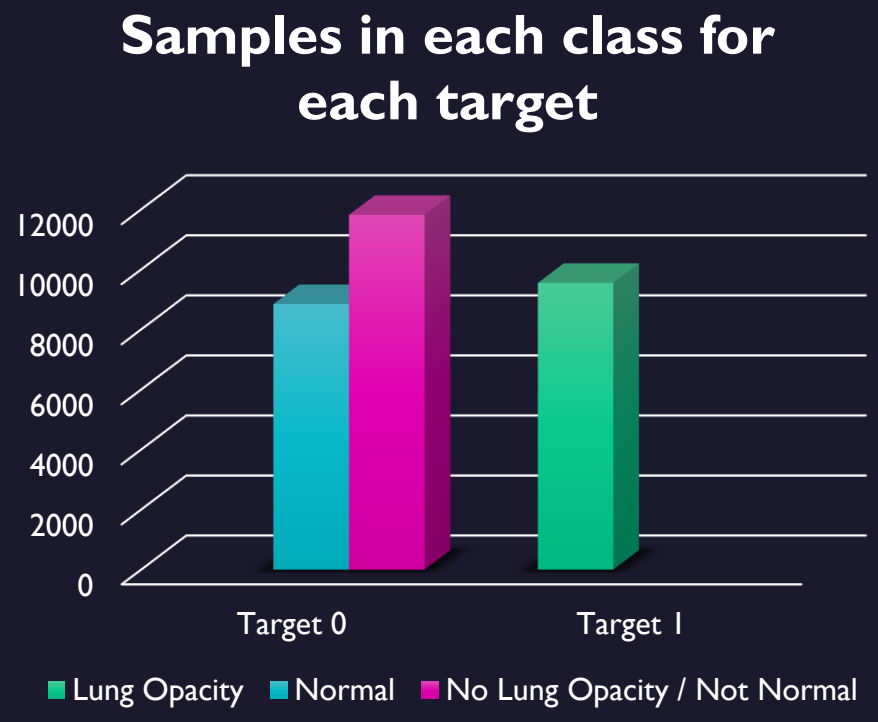
- lung opacity regions, but without diagnosed pneumonia

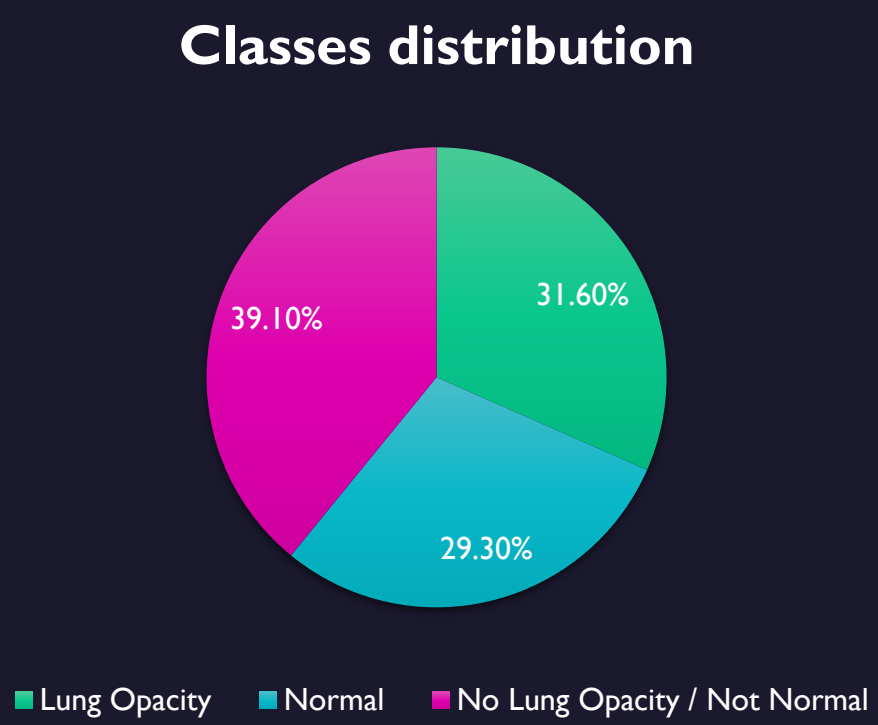


NIH Clinical Center
America's Research Hospital

Labels are stored inside two CSV files, containing **30227** rows each.

Class	Target	Patient
Lung Opacity	1	9555
Normal	0	8851
No Lung Opacity / Not Normal	0	11821





30227 label rows for 26684 images?

3543 patient ID duplicates, corresponding to multiple bounding boxes on the same image.

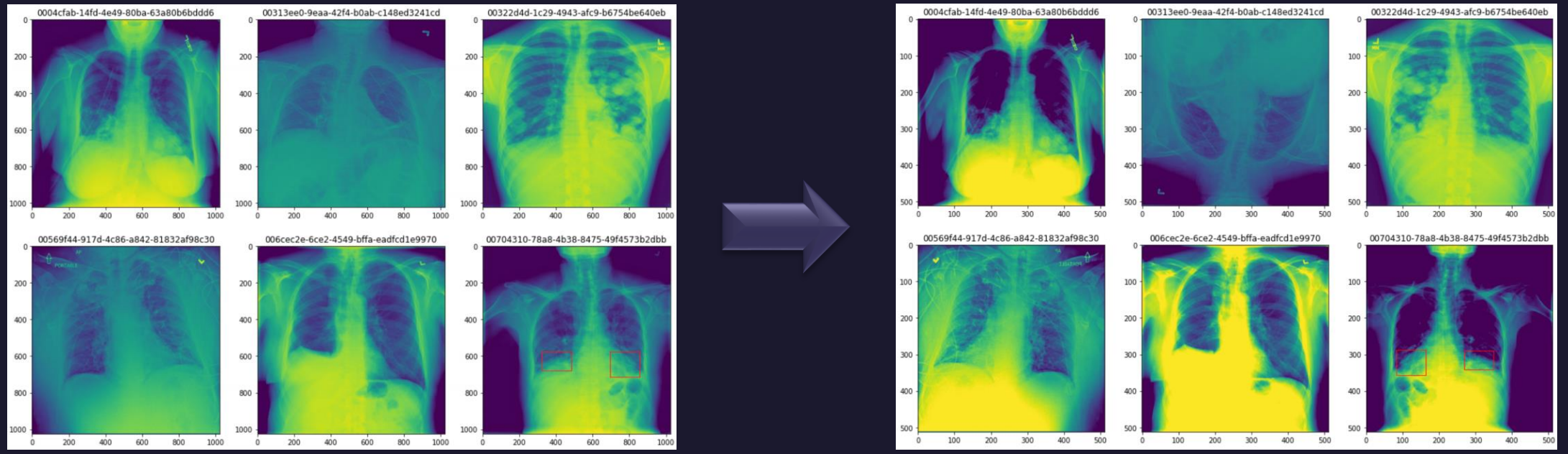
	patientId	x	y	width	height	Target
4	00436515-870c-4b36-a041-de91049b9ab4	264.0	152.0	213.0	379.0	1
5	00436515-870c-4b36-a041-de91049b9ab4	562.0	152.0	256.0	453.0	1
8	00704310-78a8-4b38-8475-49f4573b2dbb	323.0	577.0	160.0	104.0	1
9	00704310-78a8-4b38-8475-49f4573b2dbb	695.0	575.0	162.0	137.0	1

# Exams (tuples)	Class	# Entries
1	No Lung Opacity / Not Normal	11821
1	Normal	8851
1	Lung Opacity	2614
2	Lung Opacity	3266
3	Lung Opacity	119
4	Lung Opacity	13

On the full report there is also an overview about the bounding boxes analysis

Preprocessing & Augmentation steps:

1. Bounding boxes are converted to **Pascal VoC** encoding, i.e. each bounding box is identified by the top left-hand corner and bottom right-hand corner (from [x, y, w, h] to [x_1, y_1, x_max, y_max])
2. **Train / Validation / Test** dataset split => around 80% of the samples for the training set, 10% for the validation set, and 10% for the test set
3. Images are converted from DICOM format to pixel arrays (Sex, Age, Exam date)
4. Augmentation using the **imagaug** pytorch library



Training set augmentations

Resize_only	iaa.Resize(512)
Light	iaa.Resize(512), iaa.Affine(scale=1.1, shear=(2.5,2.5), rotate=(-5, 5))
Heavy	iaa.Resize(512), iaa.Affine(scale=1.15, shear=(4.0,4.0)), iaa.Fliplr(0.2), iaa.Sometimes(0.1, iaa.CoarseSaltAndPepper(p=(0.01, 0.01), size_percent=(0.1, 0.2))), iaa.Sometimes(0.5, iaa.GaussianBlur(sigma=(0.0, 2.0))), iaa.Sometimes(0.5, iaa.AdditiveGaussianNoise(scale=(0, 0.04 * 255))),
Heavy_with_rotations	iaa.Resize(512), iaa.Affine(scale=1.15, shear=(4.0,4.0), rotate=(-6, 6)), iaa.Fliplr(0.2), iaa.Sometimes(0.1, iaa.CoarseSaltAndPepper(p=(0.01, 0.01), size_percent=(0.1, 0.2))), iaa.Sometimes(0.5, iaa.GaussianBlur(sigma=(0.0, 2.0))), iaa.Sometimes(0.5, iaa.AdditiveGaussianNoise(scale=(0, 0.04 * 255))),

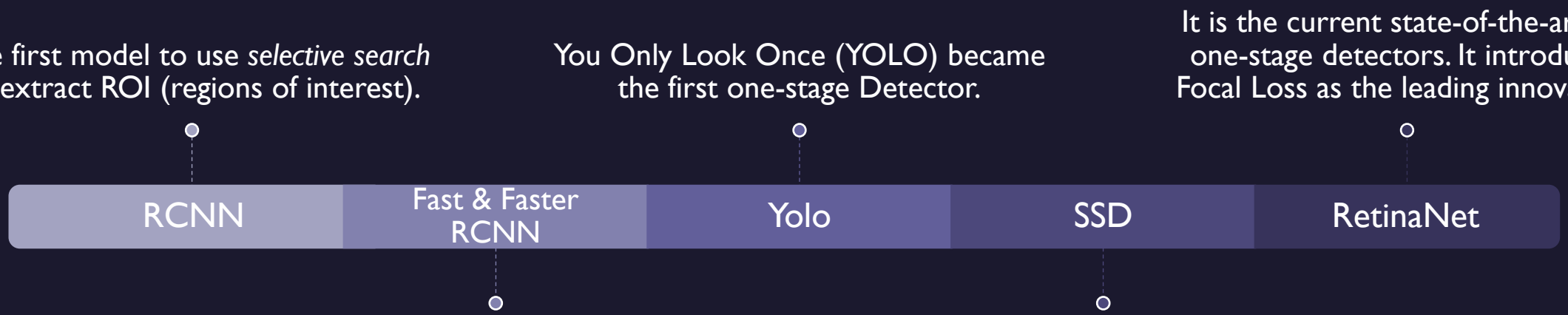
On Validation and Test set we apply resize_only

The model

Object detection

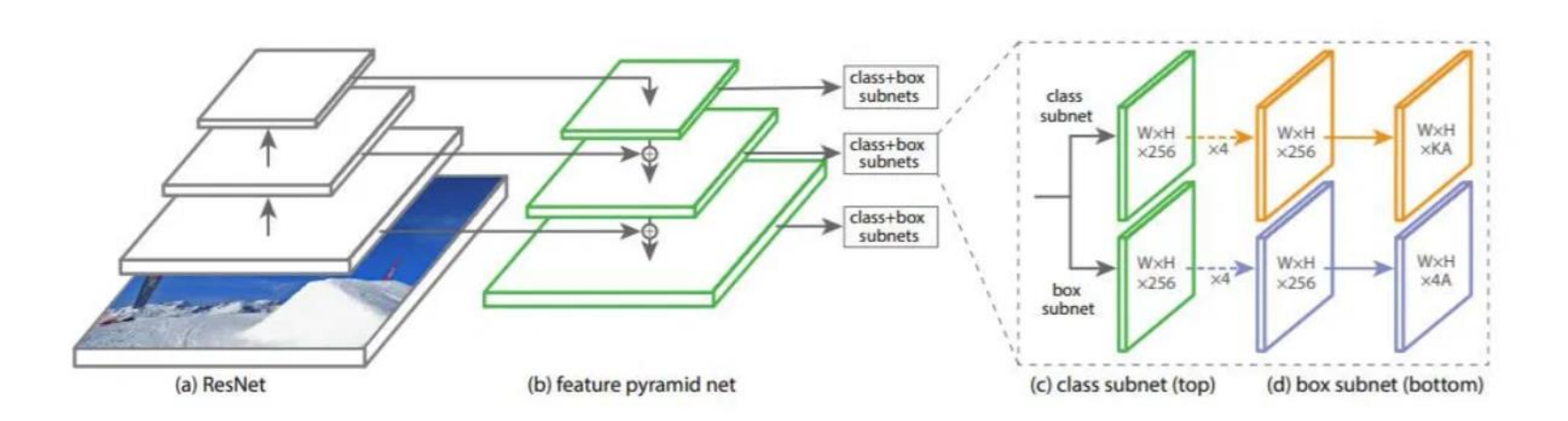
Retinanet Overiew

Object detectors: from 2 stages to 1 stage



RetinaNet: the architecture

- Backbone Network(a+b): composed by a bottom-up pathway and a top-down pathway with lateral connections.
- Classification sub-network(c): fully convolutional network is attached to each FPN level for object classification.
- Regression sub-network(d): is attached to each feature map of the FPN in parallel to the classification network.
- Global sub-network(not shown in the image): an extra classification sub-network attached to the FPN in parallel to the previous ones.



Backbone network

Bottom-up pathway:

- This part of the network is the encoder.
- In our development, we used three different pre-trained encoders: ResNet 50, Se-ResNext50 and PnasNet.
- It is used for feature extraction. It calculates the feature maps at different scales, irrespective of the input image size.

Top-down pathway:

- This part of the network is the decoder.
- This pathway is composed by a Feature Pyramid Network(FPN).
- It is used to up samples the spatially coarser feature maps from higher pyramid levels. The lateral connections merge the top-down layers and the bottom-up layers with the same spatial size.

Subnetworks: classification and regression

Classification subnetwork

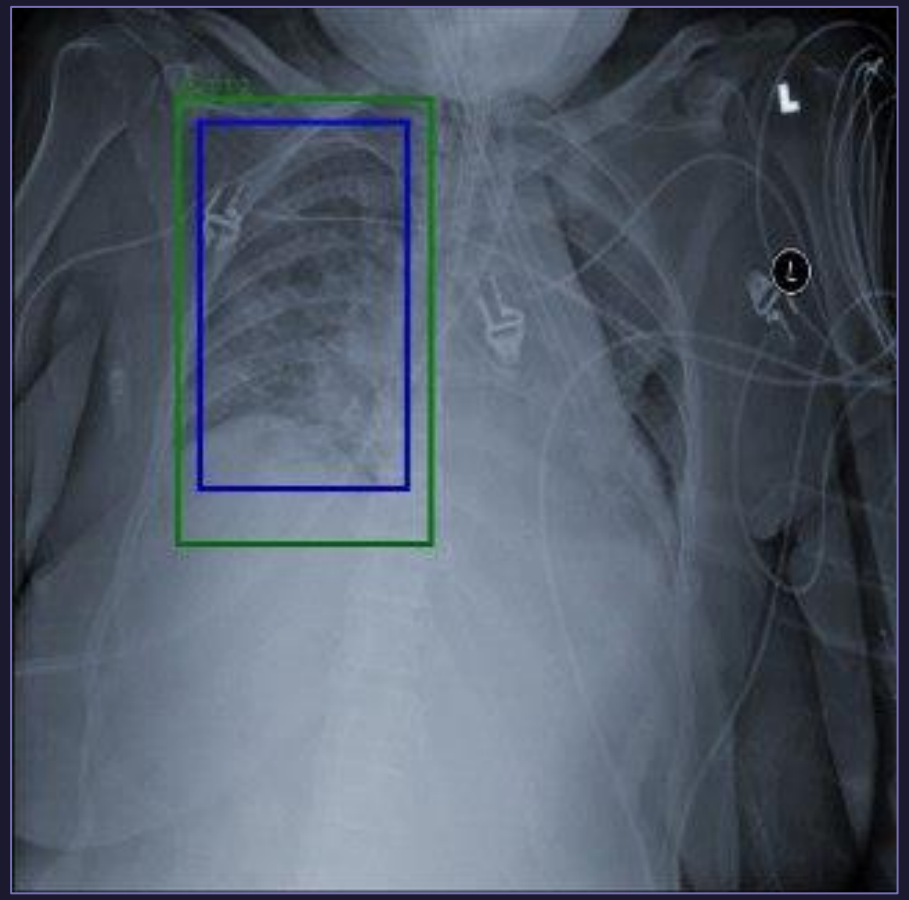
- The classification subnet is a fully convolutional network (FCN) attached to each FPN level.
- Consists of four 3x3 convolutional layers with 256 filters, followed by RELU activations.
- Then, another 3x3 convolutional layer with KxA filters are applied, followed by sigmoid activation.
- The shape of the output feature map would be (W, H, KA).

Regression subnetwork

- The regression subnet is attached to each feature map of the FPN in parallel to the classification subnet.
- The design is identical to that of the classification subnet, except that the last convolutional layer is 3x3 with 4A filters.
- Therefore, the shape of the output feature map would be (W, H, 4A).

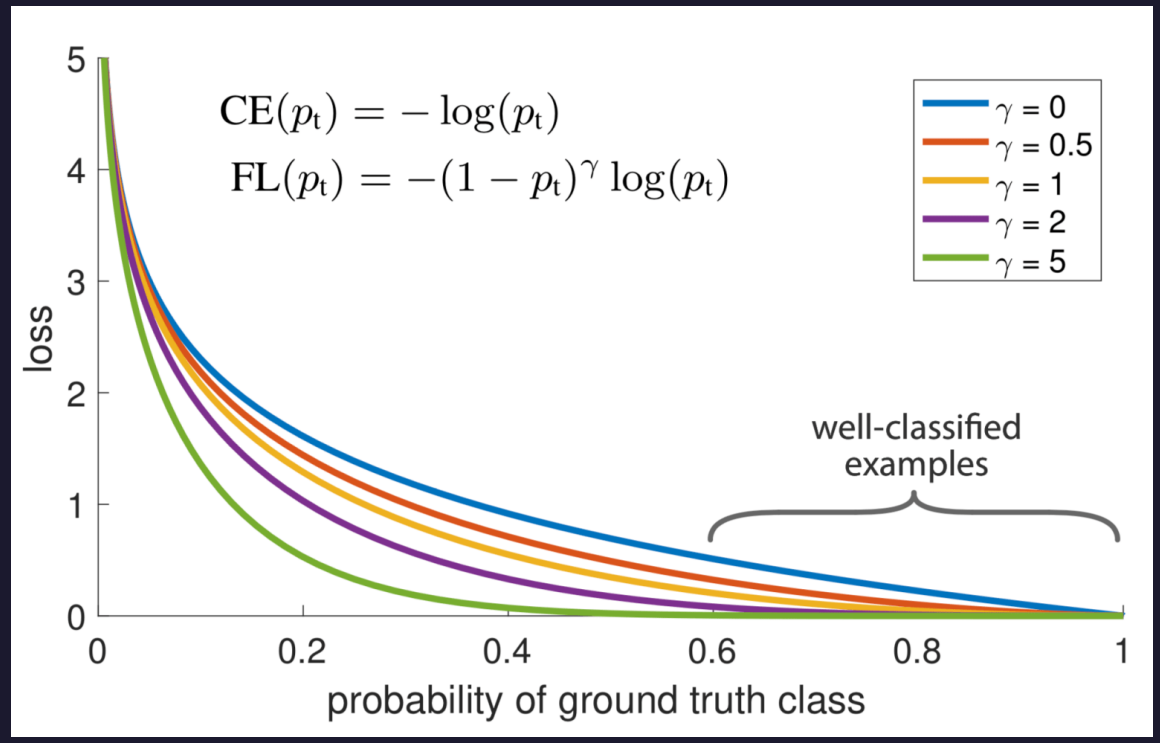
Global classification subnet

- The global classification subnetwork is not directly used for the classification, but it helps improve the results.
- The classes are: ‘No Lung Opacity/Not Normal’, ‘Normal’ and ‘Lung Opacity’.
- It consists of one max pooling 2D layer, followed by a 3x3 convolutional layer and then by a RELU activation.
- These layers are followed by an average and a max-pooling 2D, before a Linear layer and the log softmax activation.



Focal Loss

- Focal Loss is an improved version of Cross-Entropy Loss trying to handle the class imbalance problem by assigning more weights to complex or easily misclassified examples and down-weight easy examples.
- Researchers have proposed $(1-p_t)^{\gamma}$ to be added to Cross-Entropy, with $\gamma \geq 0$ as a tunable focusing parameter, to achieve the goal.
- We used an α -balanced variant of Focal-Loss:
  $$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t)$$
  with $\alpha = 0.25$ and $\gamma = 2$.



Experiments & Results

The encoders

Results

We try different combinations of encoders, augmentations and number of samples

Encoders:

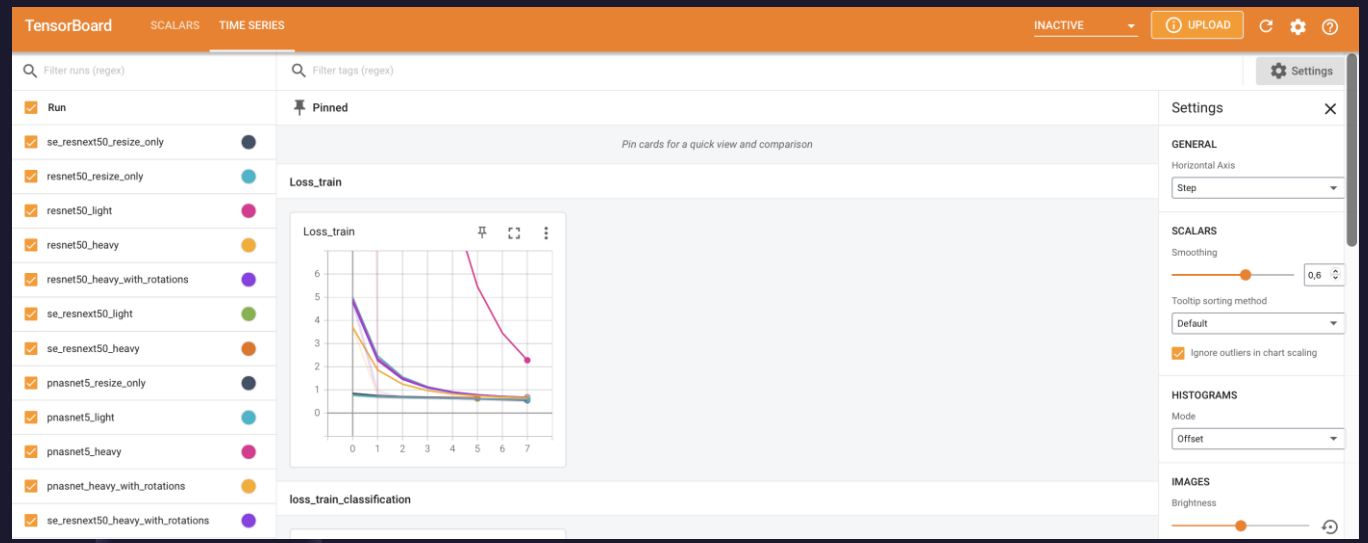
- Resnet50
- SeResNeXt50
- PnasNet5

Losses:

- Total Loss
- Classificaion Loss
- Global classification Loss
- Regression Loss

Augmentations:

- Resize only
- Light
- Heavy
- Heavy with rotations

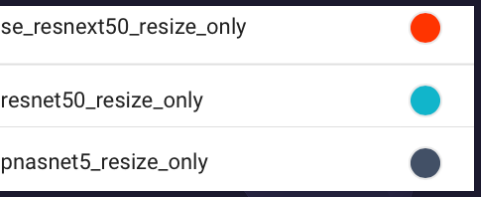


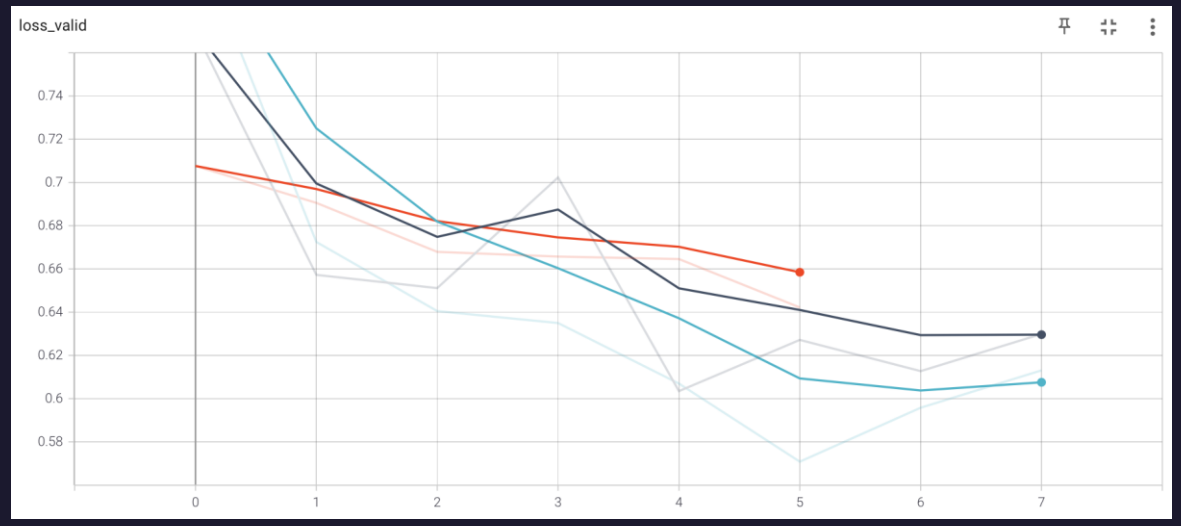
We draw the resulting losses in different plots.

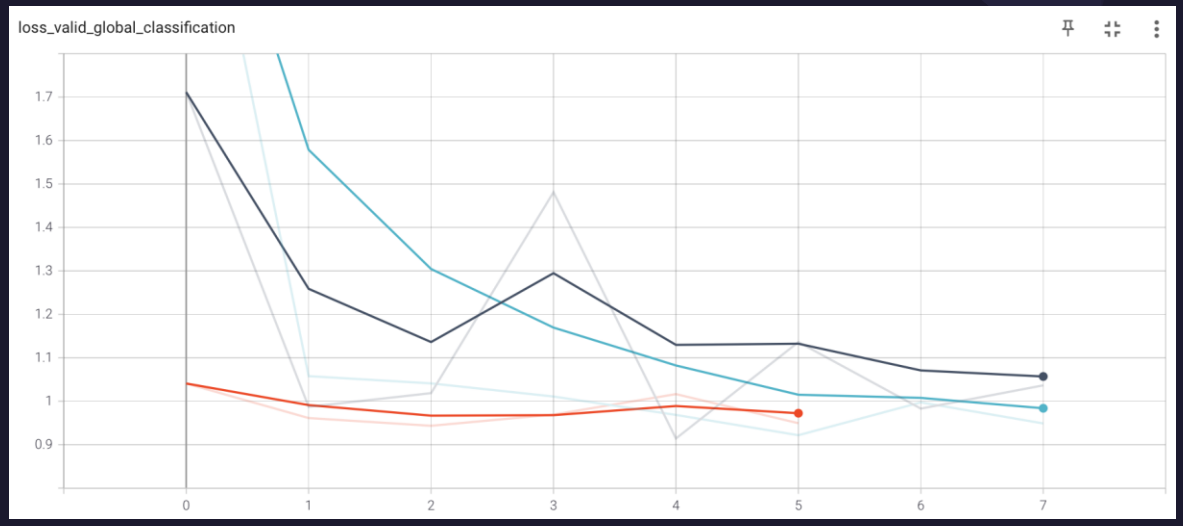
SeResNeXt is the only encoder taken for the 6k samples experiments.

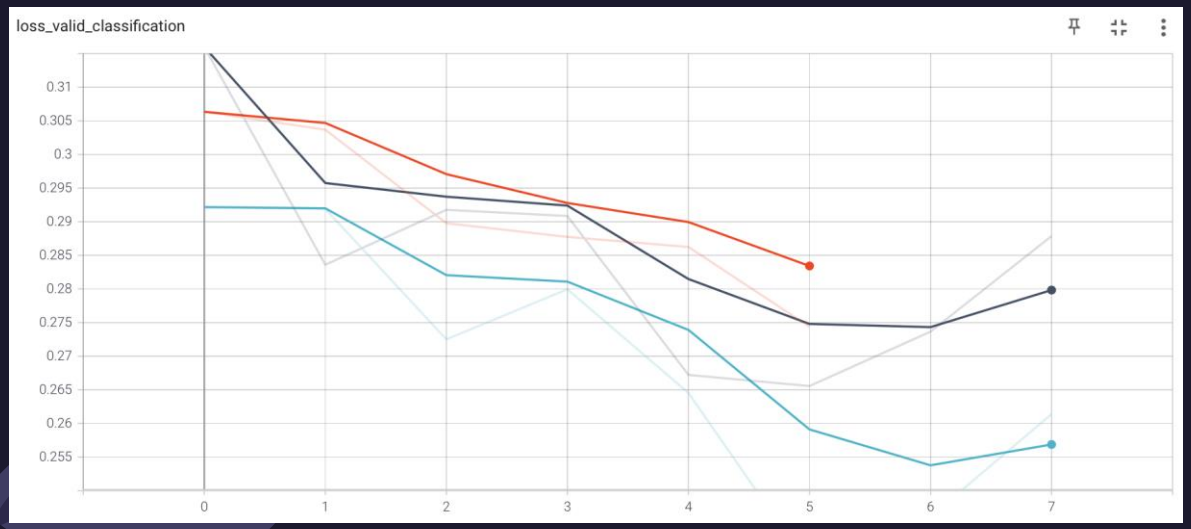
Augmentation: Resize Only

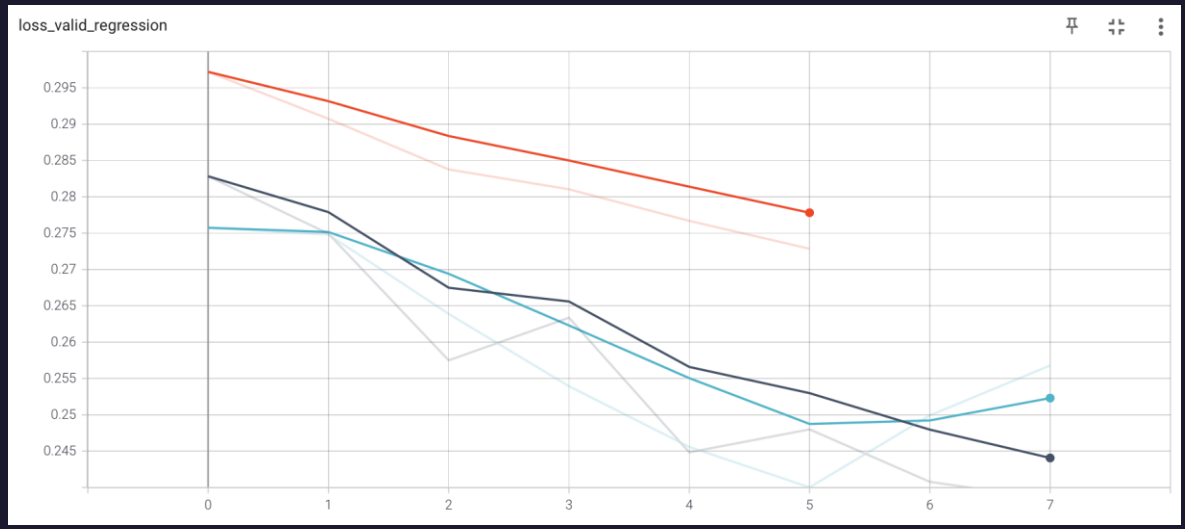
Samples 3000











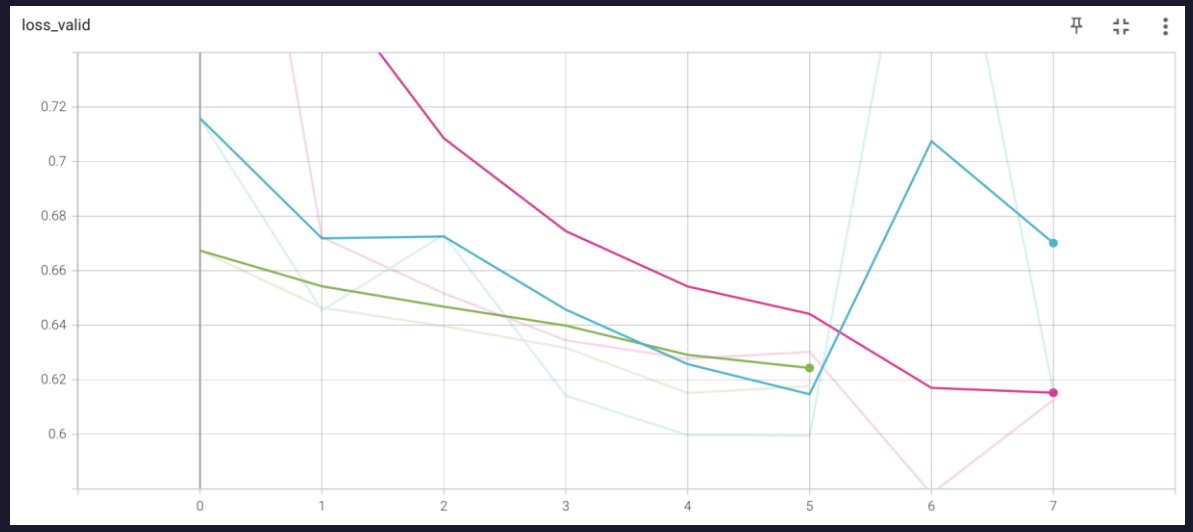
Augmentation: Light

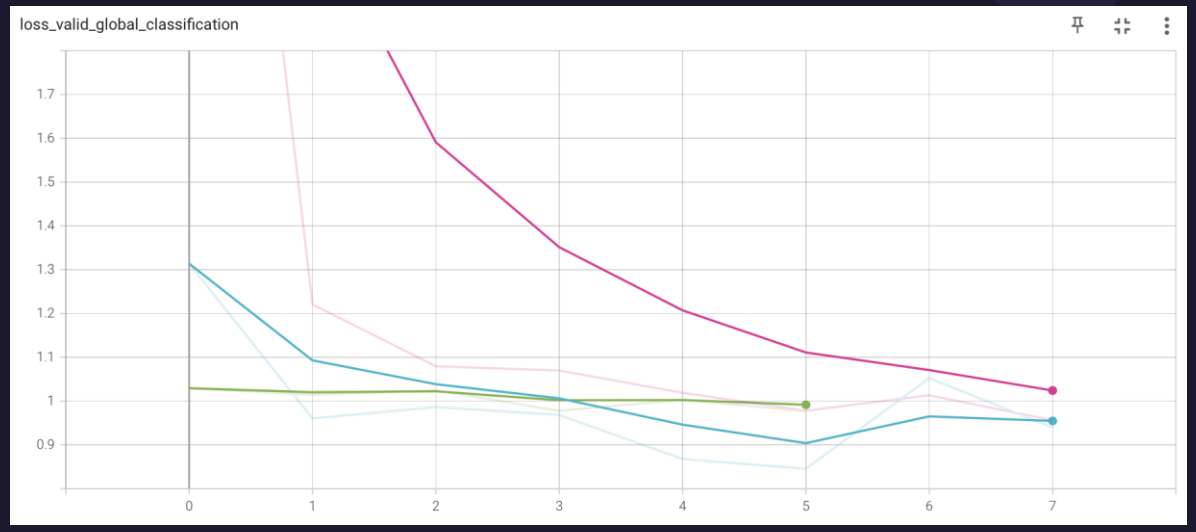
Samples 3000

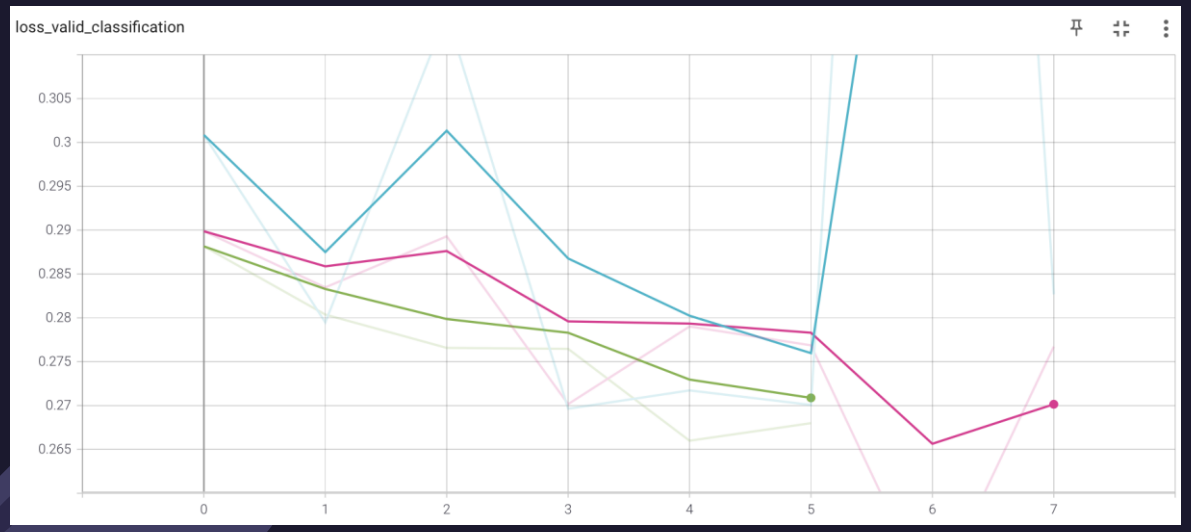
pnasnet5_light

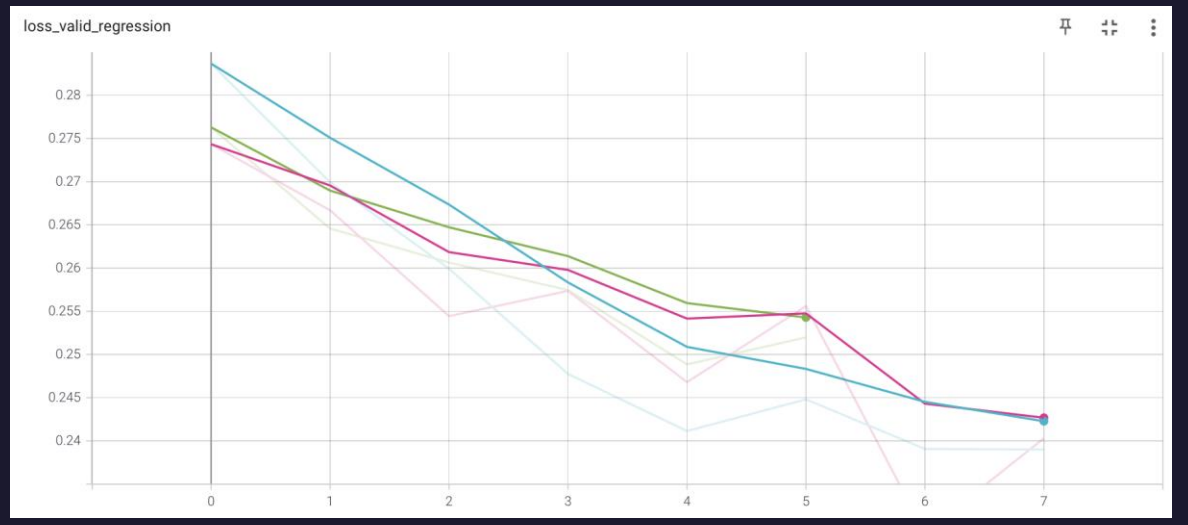
resnet50_light

se_resnext50_light









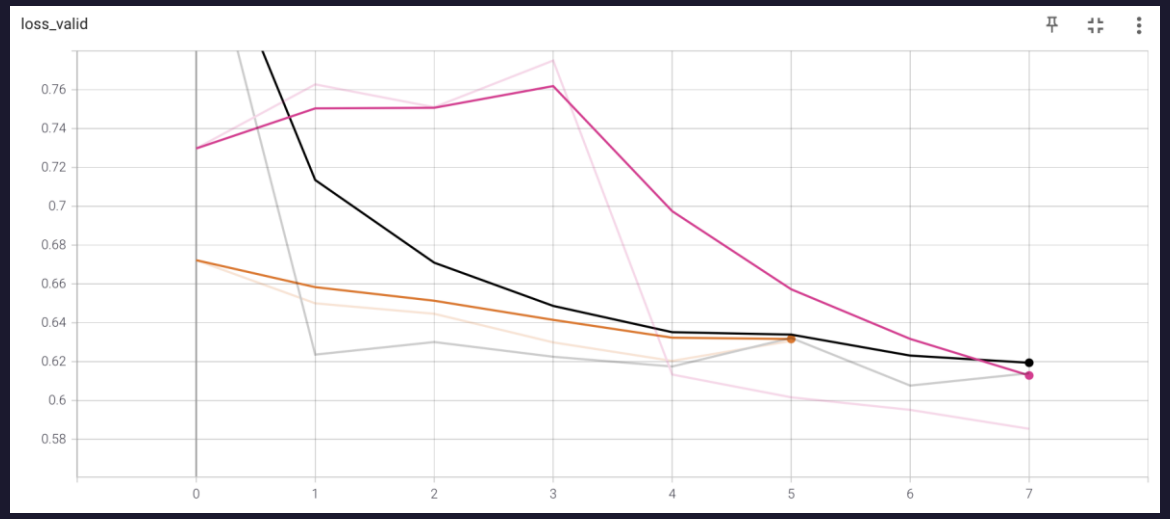
Augmentation: Heavy

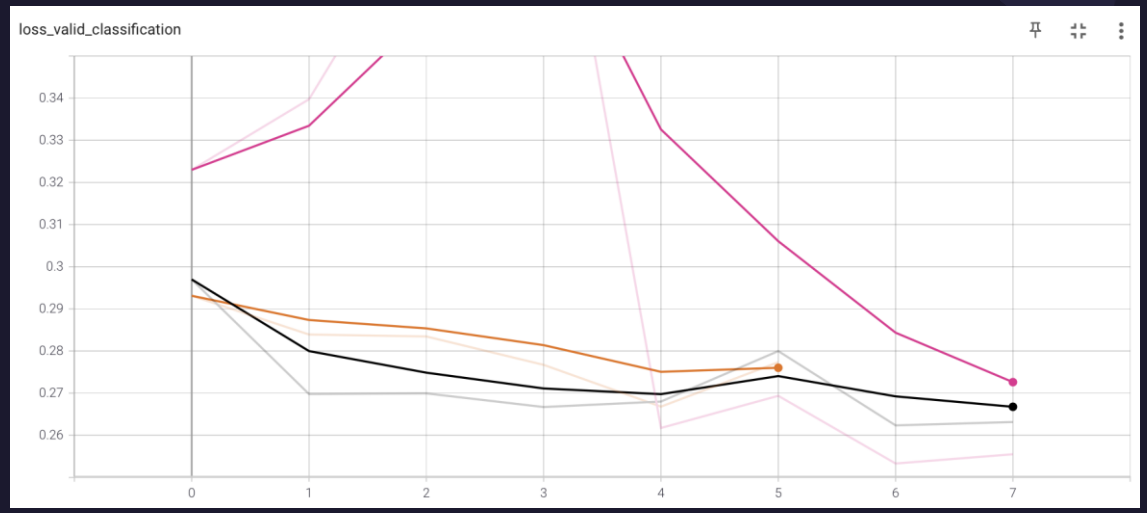
Samples 3000

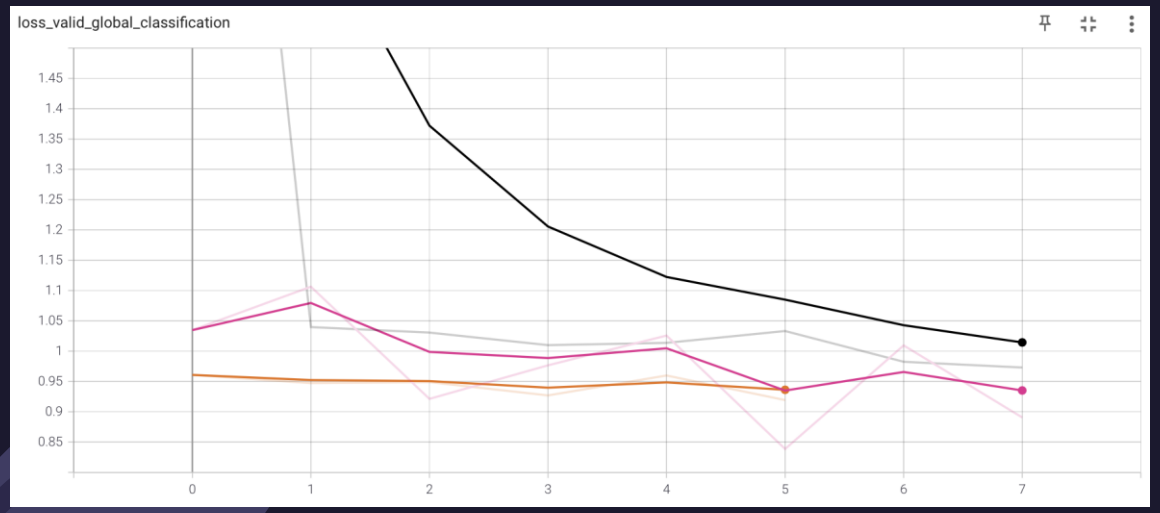
pnasnet5_heavy

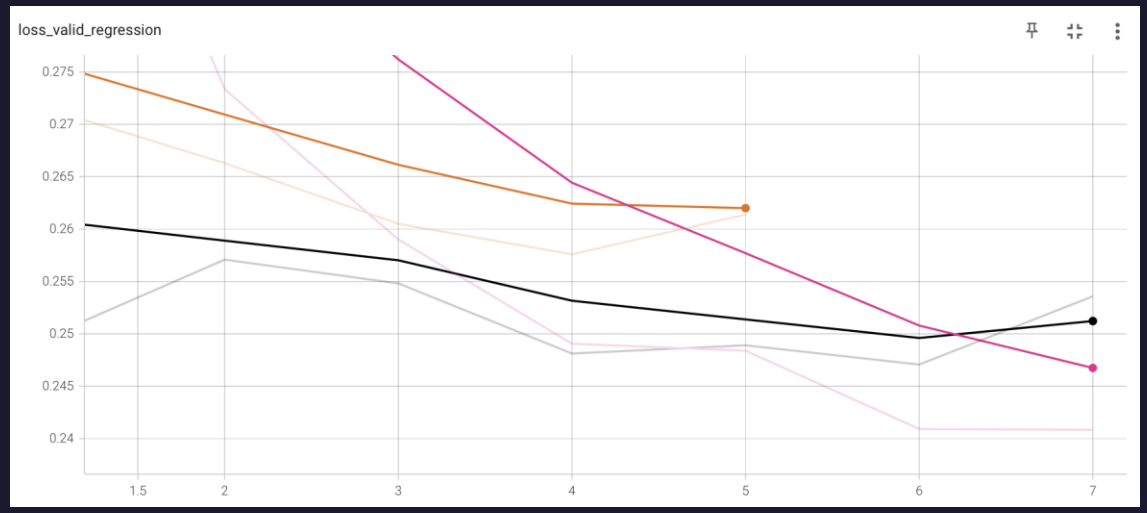
resnet50_heavy

se_resnext50_heavy









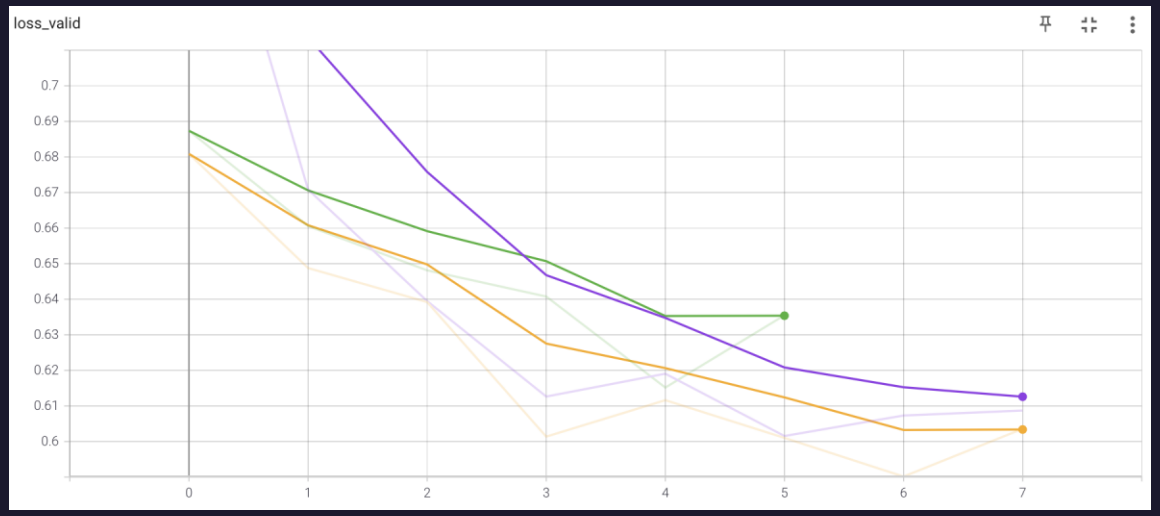
Augmentation: Heavy with rotations

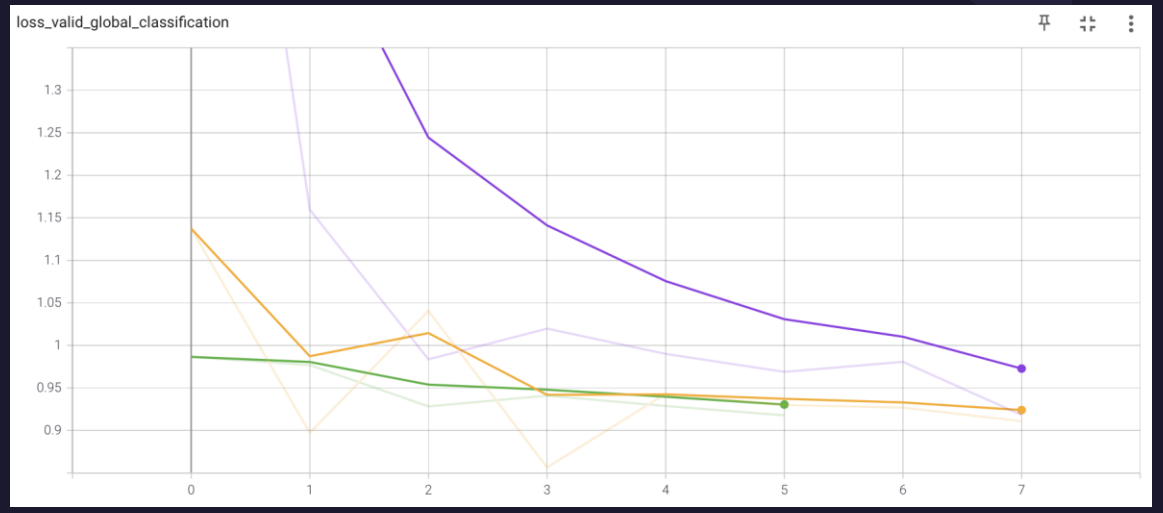
Samples 3000

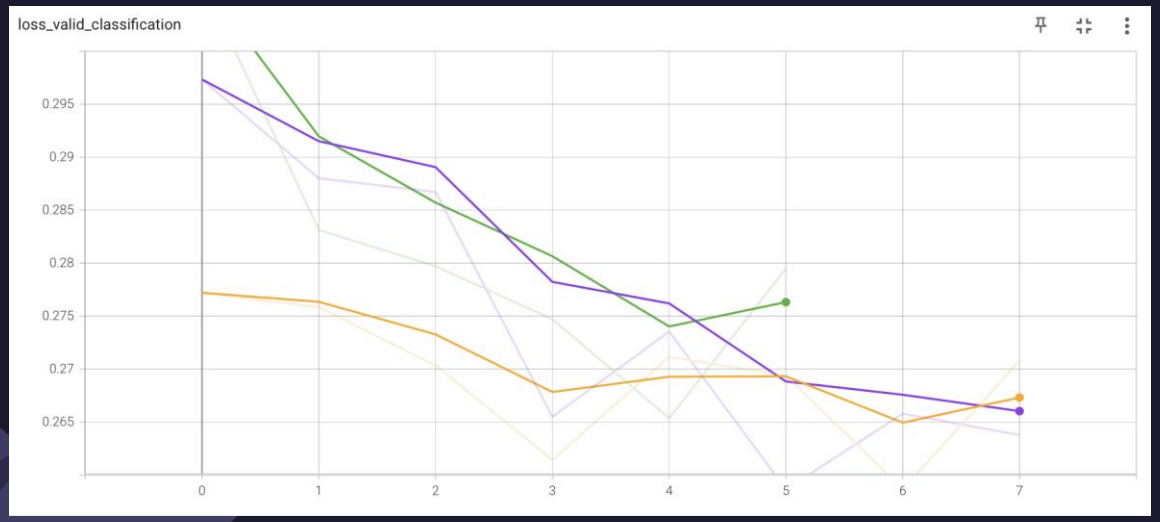
se_resnext50_heavy_with_rotations

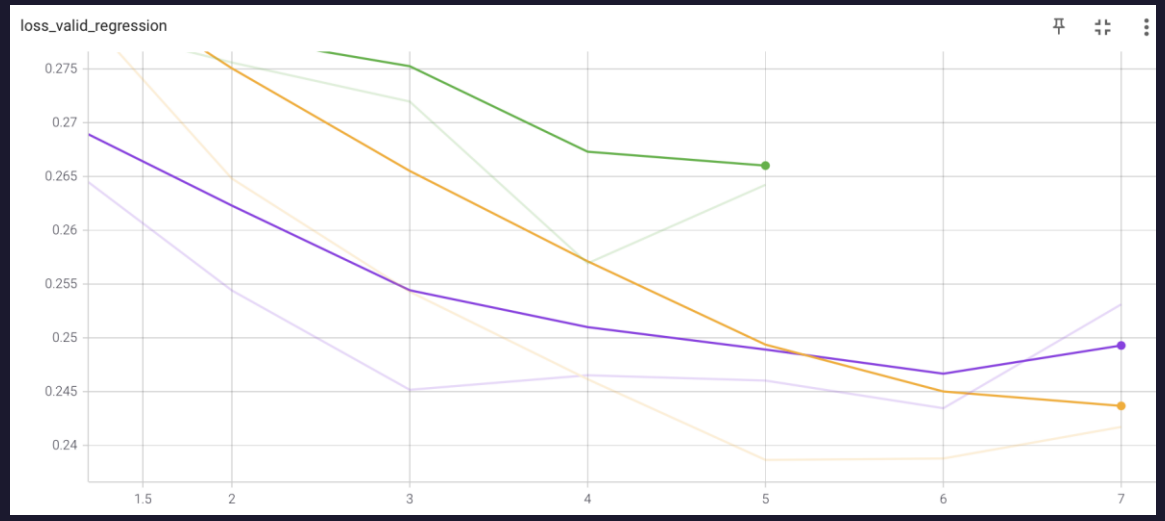
pnasnet_heavy_with_rotations

resnet50_heavy_with_rotations









Encoder: ResNet

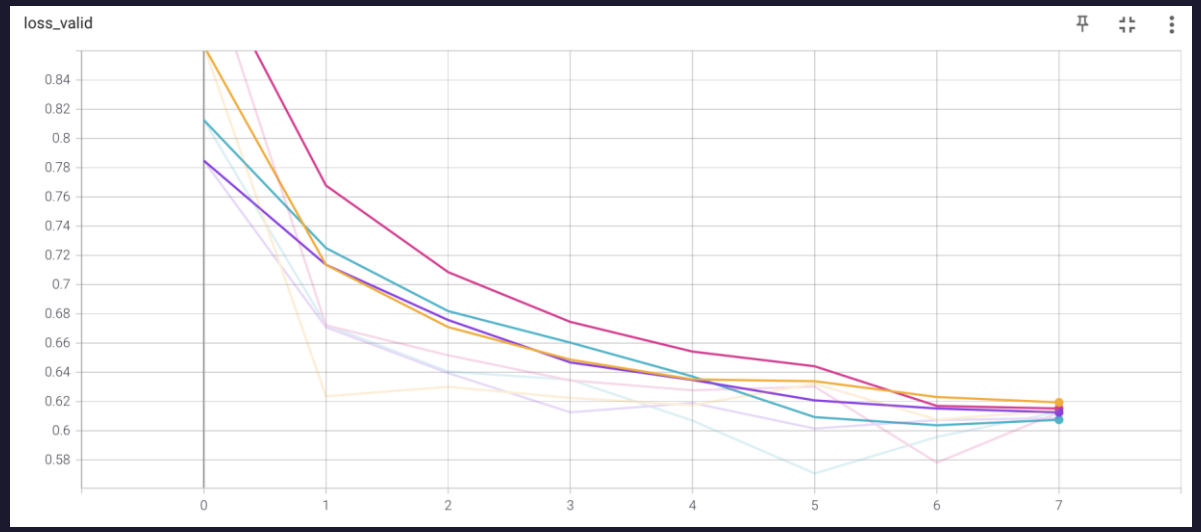
Samples: 3000

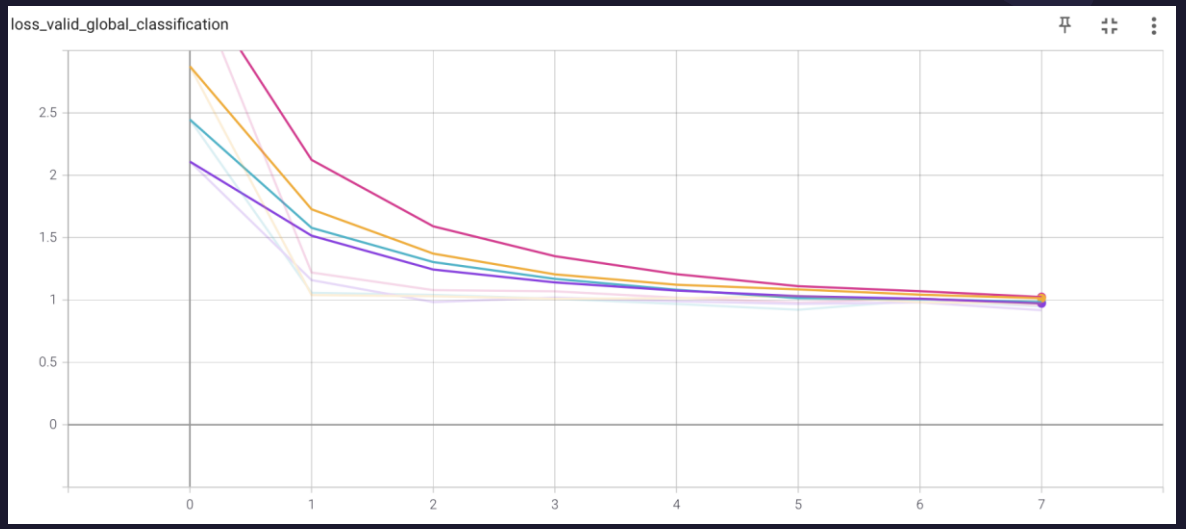
resnet50_resize_only

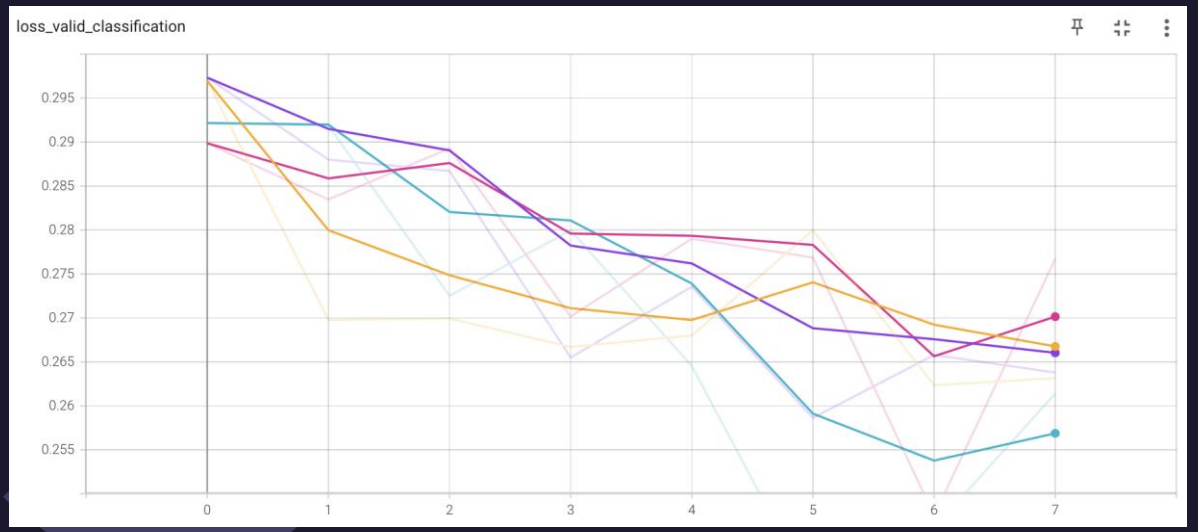
resnet50_light

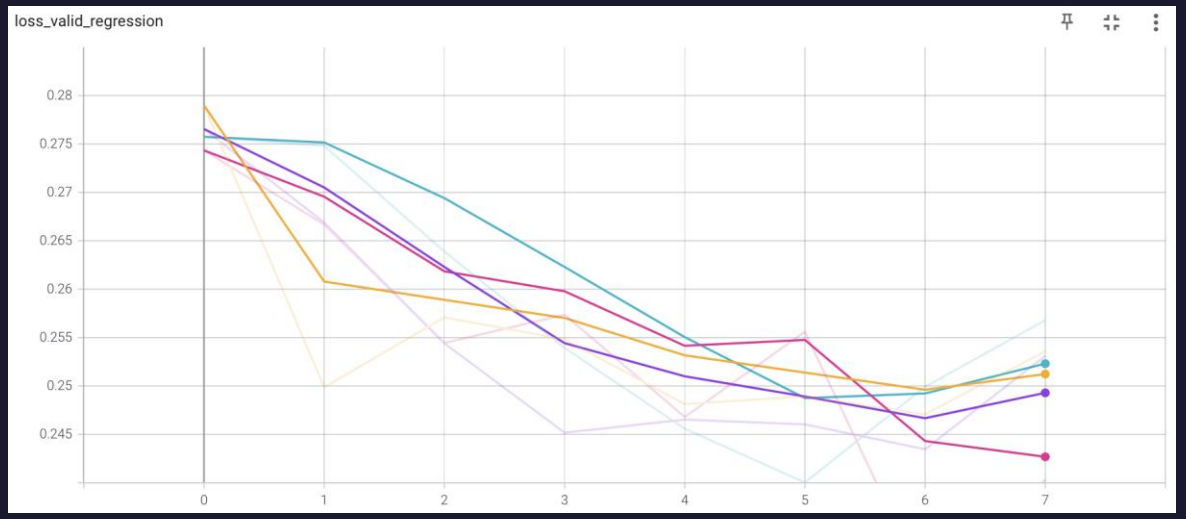
resnet50_heavy

resnet50_heavy_with_rotations



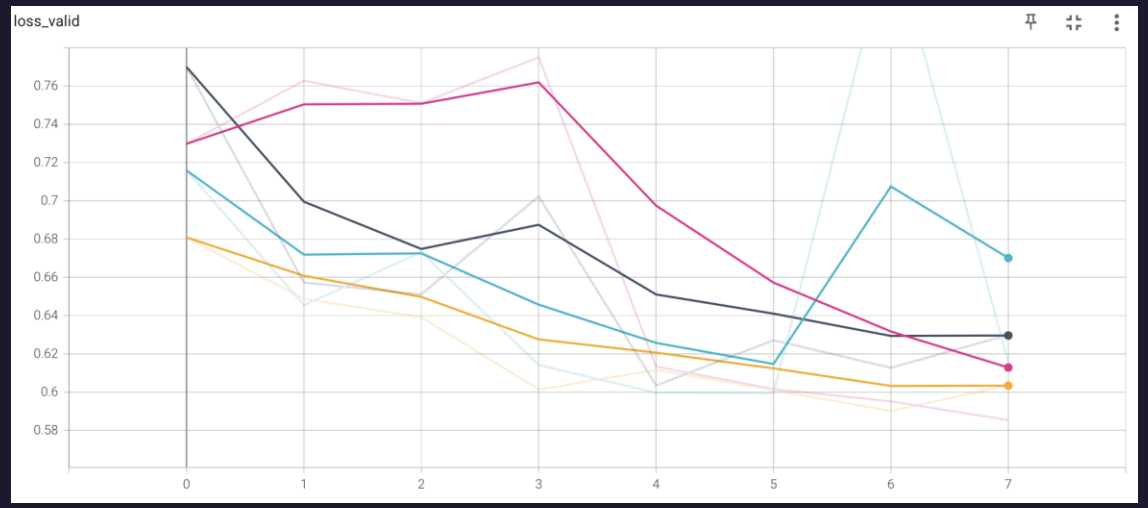


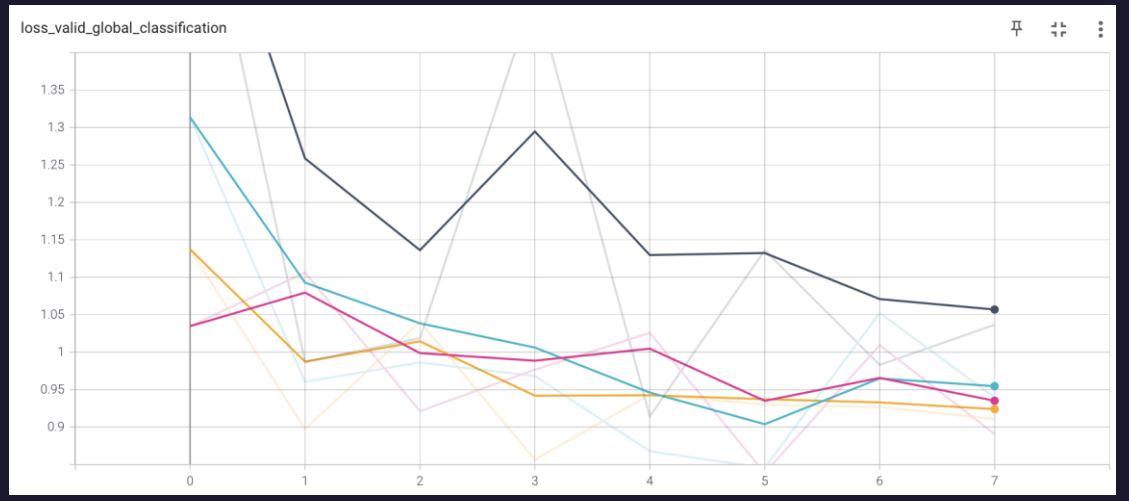


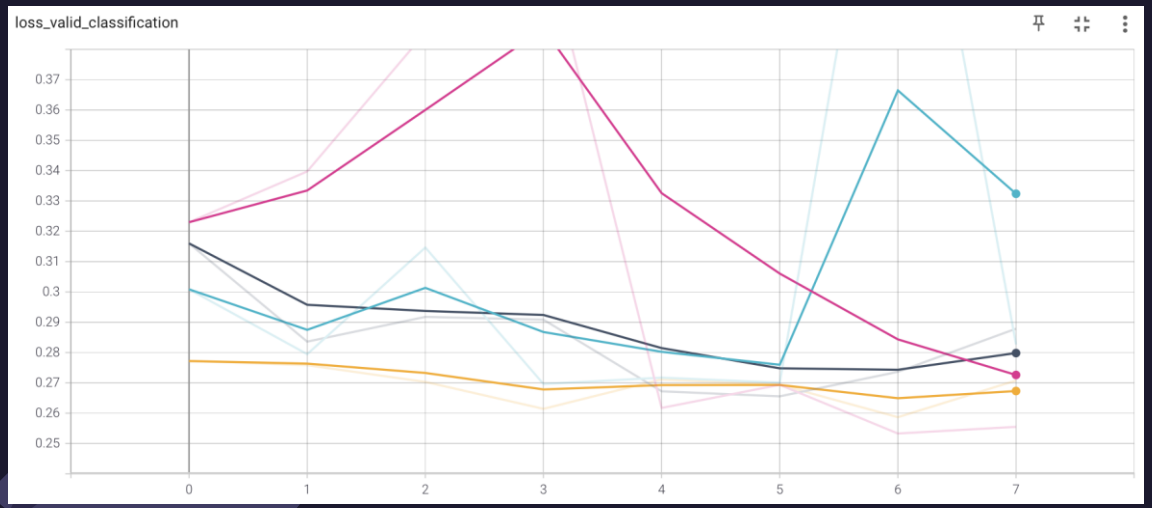


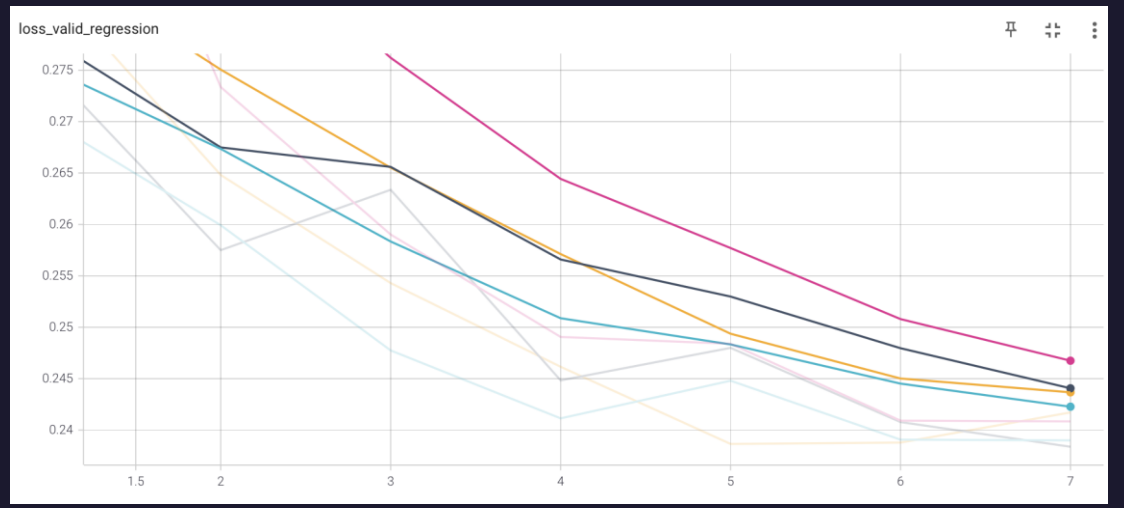
Encoder: PnasNet

Samples: 3000



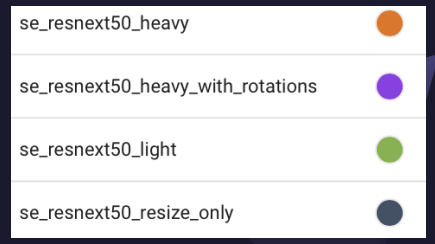


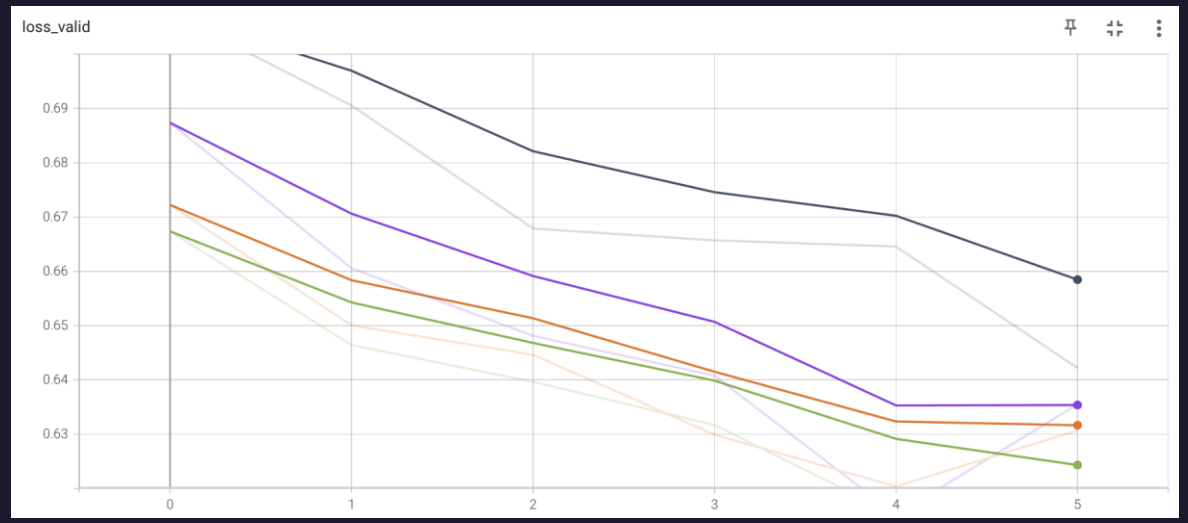


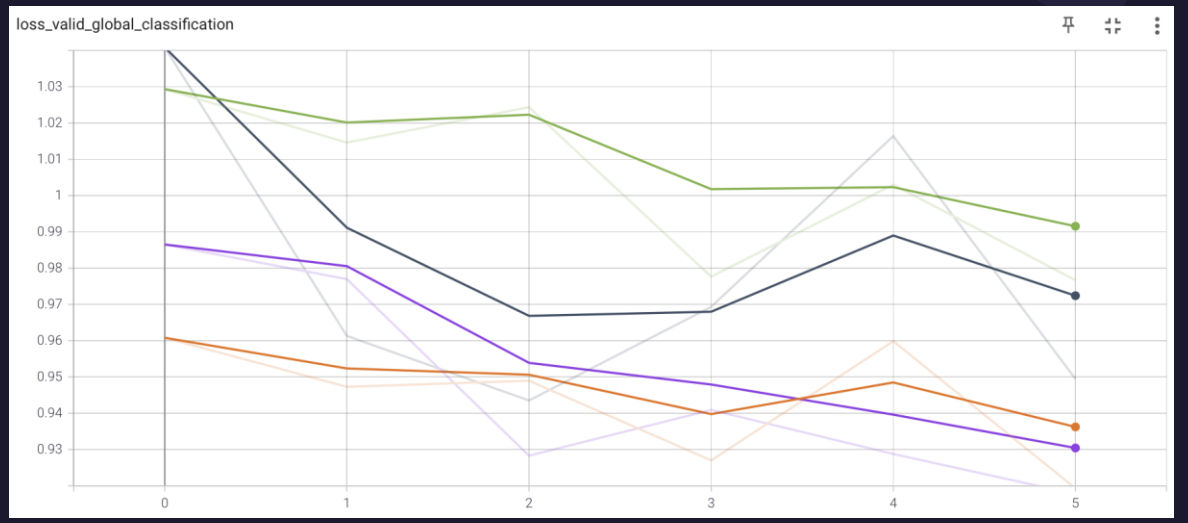


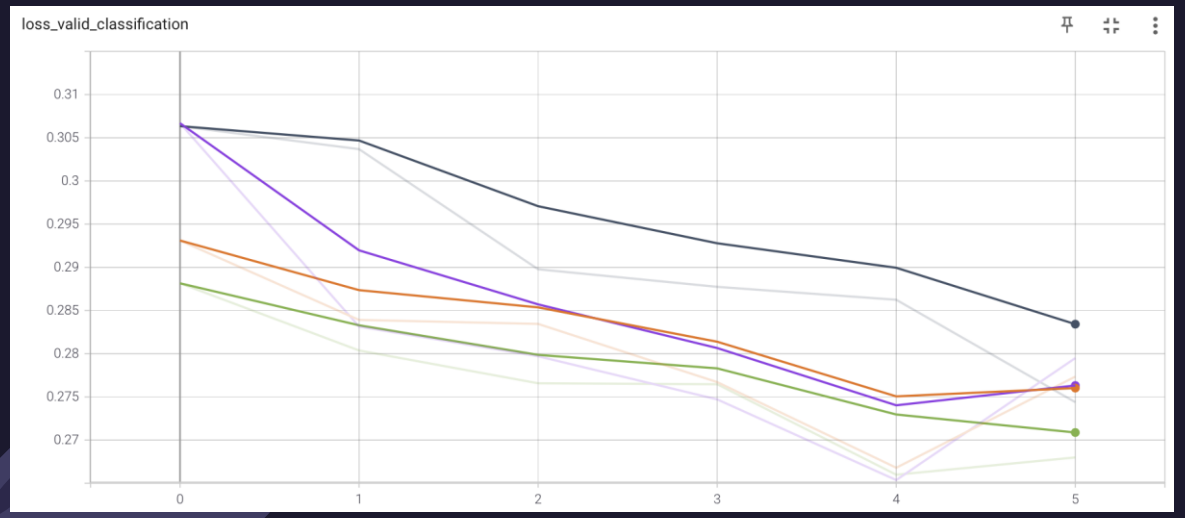
Encoder: SE_ResNeXt

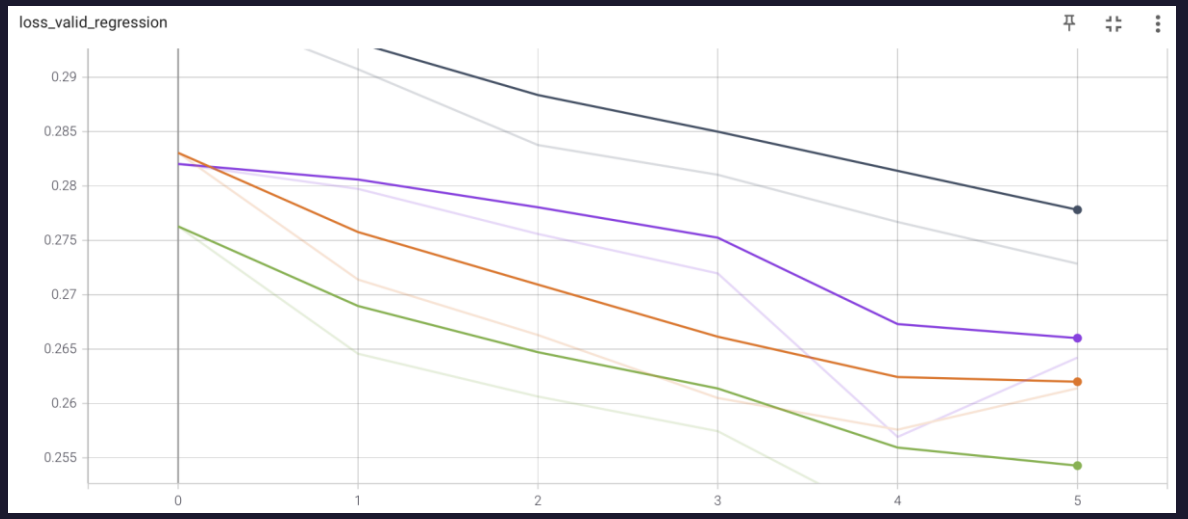
Samples: 3000



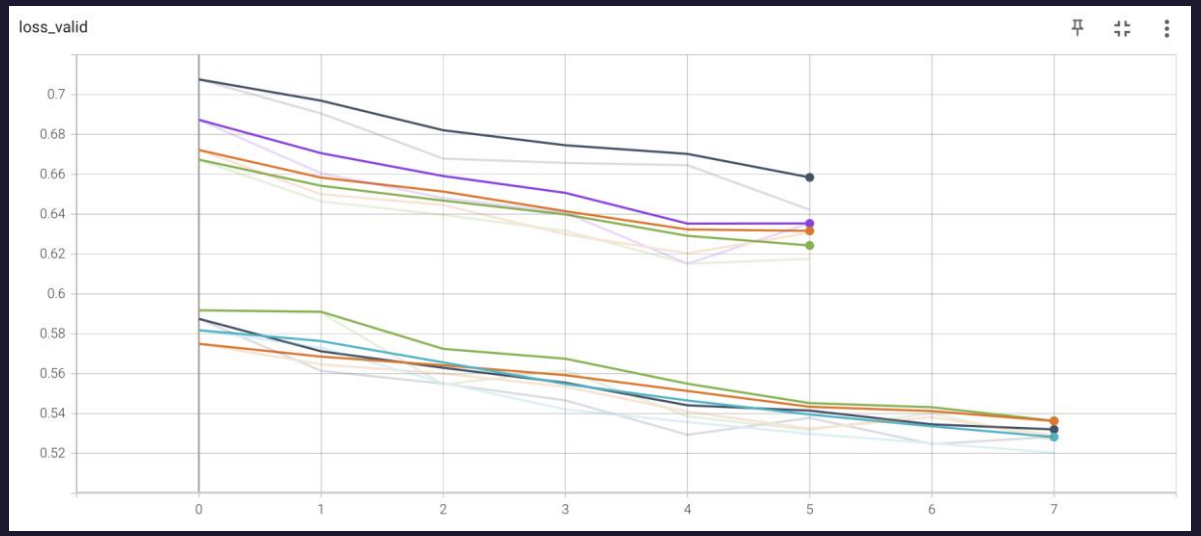


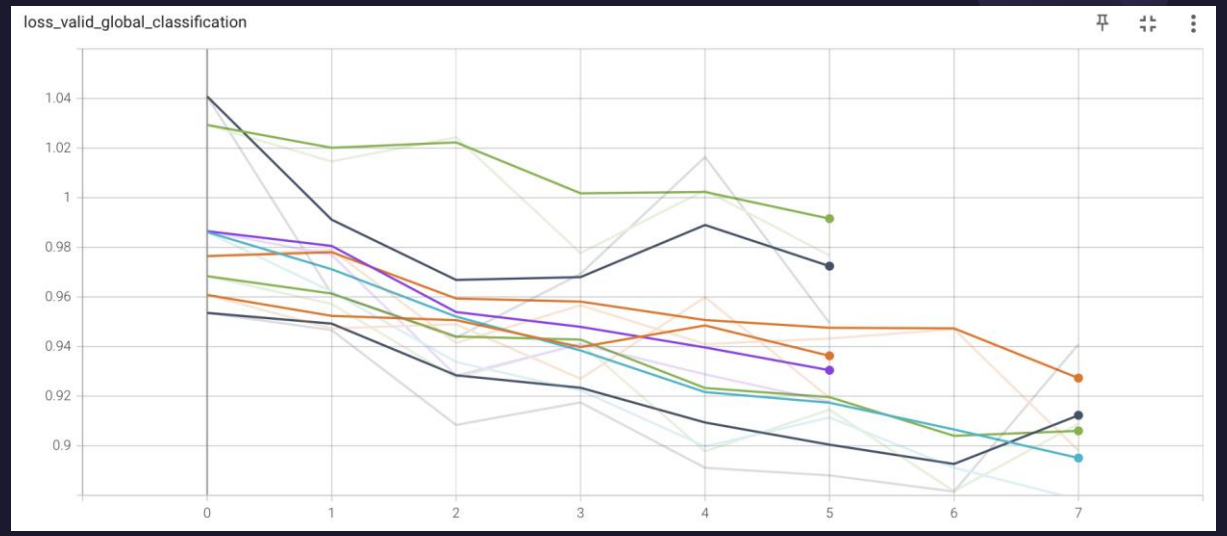


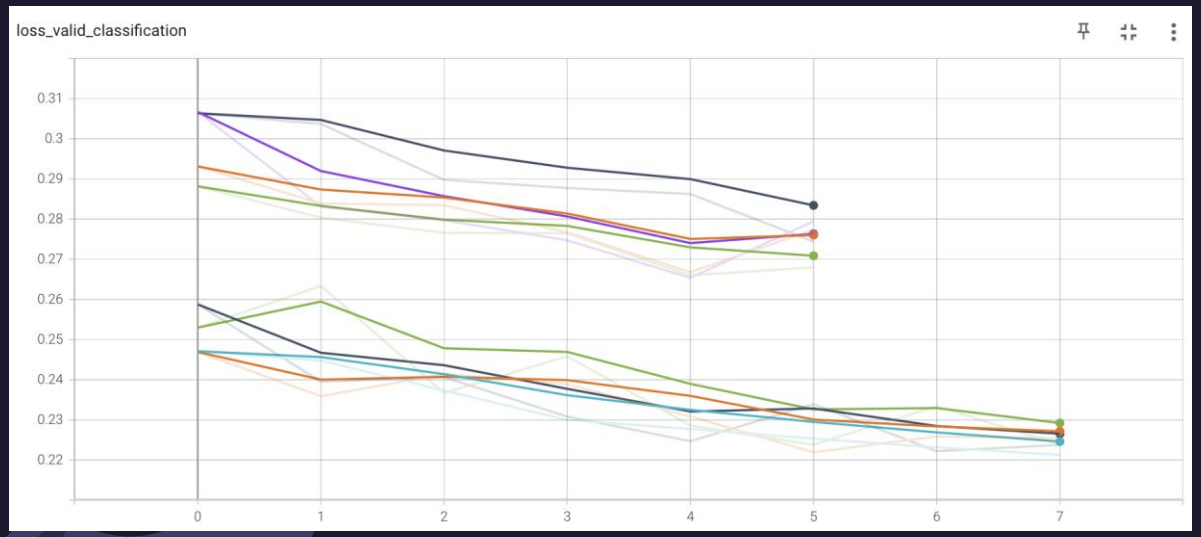


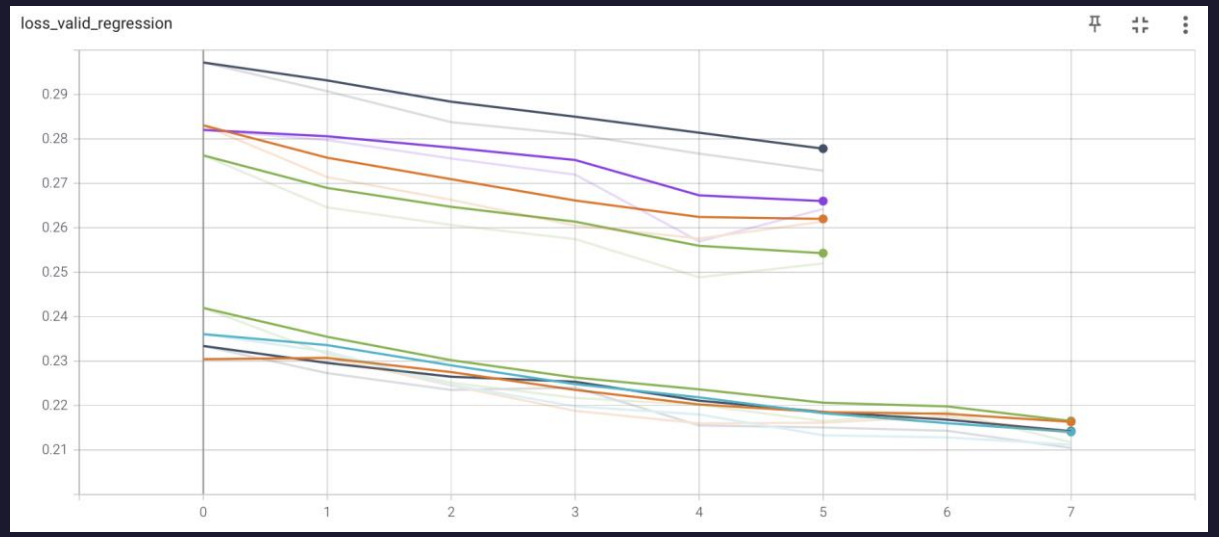


SE_ResNeXt with 3000 samples VS SE_ResNeXt with 6000 samples









Conclusions

- With small samples, the augmentations do not really provide relevant differences between the tests.
- A different encoder may provide improvements.
- The real difference is in the number of samples we provide to the network.