

# University of Oslo

## IN4310 - Deep Learning for Image Analysis

23 March 2023

Mandatory assignment | 1 | Berezin Ilya

### Contents

Task 1.....	2
Data processing.....	2
Perform a train-validation-test split.....	2
Obtained splits verified on being disjoint. ....	2
Data transformation.....	3
Batch size.....	3
Model (ResNet18) .....	3
Metrics plotted after training.....	4
Testing the best model (SGD optimizer and learning rate of 0.01) .....	4
Confusion matrix and Classification report.....	5
Increased number of epochs to 35 for the best converging model.....	6
SoftMax scores tensor.....	6
Top-10 and bottom-10 ranked images.....	6
Task 2: Internals of network feature map statistics.....	7
Extracted features and the percentage of non-positive values for selected feature maps .....	7
Task 3: Internals of network feature map statistics (MSc) .....	8
Top-k eigenvalues of the empirical covariance matrix .....	8

# Task 1

## Data processing

\*Code was developed in Google Colab IDE due to availability of GPU's computational resources, please see the results in attached IN4310\_Project\_v3\_7.ipynb with generated outputs, including training epochs statistics and metrics plots.

Perform a train-validation-test split of the data and use a manual seed so that you and we can reproduce the same split your experiments. Split the 17034 images into 3k for testing, 2k for validation and the remainder for training. Split every class separately (stratified splitting).

1. `np.random.seed(42)` was used for reproducibility.
2. 2500 images of each class were loaded from corresponding class folders for training to provide 'equal opportunities' for each image class.
3. Data was split as 10% for the test dataset and 18% for the validation dataset (20% of remaining 90%).
4. Image Files were unzipped from Google Drive to an internal Google Colab memory (choice due to the available free GPU for model training).
5. CIFAR-10 dataset downloaded and unzipped the same way to speed-up IO operations.

Obtained splits verified on being disjoint.

```
[7] # Verify the splits are disjoint
def verify_dj(train_data, val_data, test_data):
    all_data = np.concatenate((train_data, val_data, test_data))
    unique_data = set(all_data)

    if len(unique_data) == len(all_data):
        print("The splits are disjoint")
    else:
        print("There are common elements between the splits")

# Call the function to verify the splits
verify_dj(train_data, val_data, test_data)
```

The splits are disjoint

## Data transformation

```
[8] # Data transforms
    data_transforms = {
        'train': transforms.Compose([
            transforms.RandomResizedCrop(224),
            transforms.RandomHorizontalFlip(),
            transforms.ToTensor(),
            transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
        ]),
        'valid': transforms.Compose([
            transforms.Resize(256),
            transforms.CenterCrop(224),
            transforms.ToTensor(),
            transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
        ])
```

## Batch size

is set to 64 to utilize no more than 4Gb RAM.

## Model (ResNet18)

has been trained (finetuned) displaying progress and current accuracies and losses for both training and validation dataset for different hyperparameters (ADAM, SGD optimizers, 0.01, 0.001 learning rates and different number of epochs).

### ResNet18 (Residual Network) model architecture:

This particular ResNet model has been slightly modified from its original architecture, the final fully connected layer has been adapted to output 6 features instead of the standard 1000 for ImageNet classification.

---

**conv1:** A 2D convolutional layer with 3 input channels, 64 output channels, a kernel size of 7x7, a stride of 2, and padding of 3.

**bn1:** A batch normalization layer for the 64 output channels of the previous layer, with an epsilon value of 1e-05 and momentum of 0.1.

**relu:** A ReLU activation function applied in-place.

**maxpool:** A 2D max-pooling layer with a kernel size of 3, a stride of 2, and padding of 1.

**layer1 to layer4:** Four sequential layers, each containing two BasicBlock modules. Each BasicBlock consists of two convolutional layers with BatchNorm2d and ReLU activation functions, and optionally a downsample layer to adjust the input shape when required.

**layer1:** Two BasicBlocks with 64 input and output channels.

**layer2:** Two BasicBlocks with 128 output channels, and downsample from 64 to 128 channels.

**layer3:** Two BasicBlocks with 256 output channels, and downsample from 128 to 256 channels.



**layer4:** Two BasicBlocks with 512 output channels, and downsample from 256 to 512 channels.

**avgpool:** An adaptive average pooling layer that reduces the spatial dimensions to 1x1.



**fc:** A fully connected (linear) layer that takes 512 input features and outputs 6 features, corresponding to the number of classes in the classification task.

Training with learning rate: 0.001 and optimizer: SGD

Epoch 1/25



train progress: 100%  169/169 [01:04<00:00, 3.02it/s, acc=tensor(0.7945, device='cuda:0', dtype=torch.float64), loss=0.198]  
loss: 0.6007 accuracy: 79.45%  
valid progress: 100%  43/43 [00:11<00:00, 3.79it/s, acc=tensor(0.9107, device='cuda:0', dtype=torch.float64), loss=0.23]  
loss: 0.2467 accuracy: 91.07%

Epoch 2/25

train progress: 100%  169/169 [01:03<00:00, 3.05it/s, acc=tensor(0.8757, device='cuda:0', dtype=torch.float64), loss=0.299]  
loss: 0.3499 accuracy: 87.57%  
valid progress: 100%  43/43 [00:11<00:00, 3.83it/s, acc=tensor(0.9133, device='cuda:0', dtype=torch.float64), loss=0.243]  
loss: 0.2250 accuracy: 91.33%

.....

Epoch 25/25

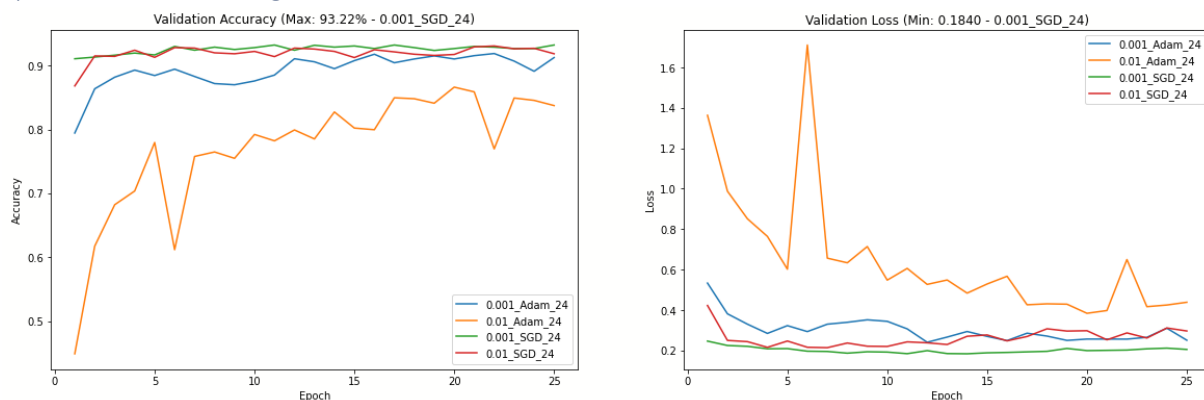
train progress: 100%  169/169 [01:04<00:00, 2.82it/s, acc=tensor(0.9419, device='cuda:0', dtype=torch.float64), loss=0.17]  
loss: 0.1592 accuracy: 94.19%  
valid progress: 100%  43/43 [00:12<00:00, 3.74it/s, acc=tensor(0.9322, device='cuda:0', dtype=torch.float64), loss=0.0354]  
loss: 0.2055 accuracy: 93.22%

-----  
Best Validation Accuracy 0.9322  
Learning Rate 0.001  
Optimizer SGD  
Epoch 11  
-----


Best model saved to /content/drive/My Drive/IN4310/model\_acc0.9322\_lr0.001\_SGD\_epoch11\_images15000.pth

Best model also saved as "best\_model.pth"

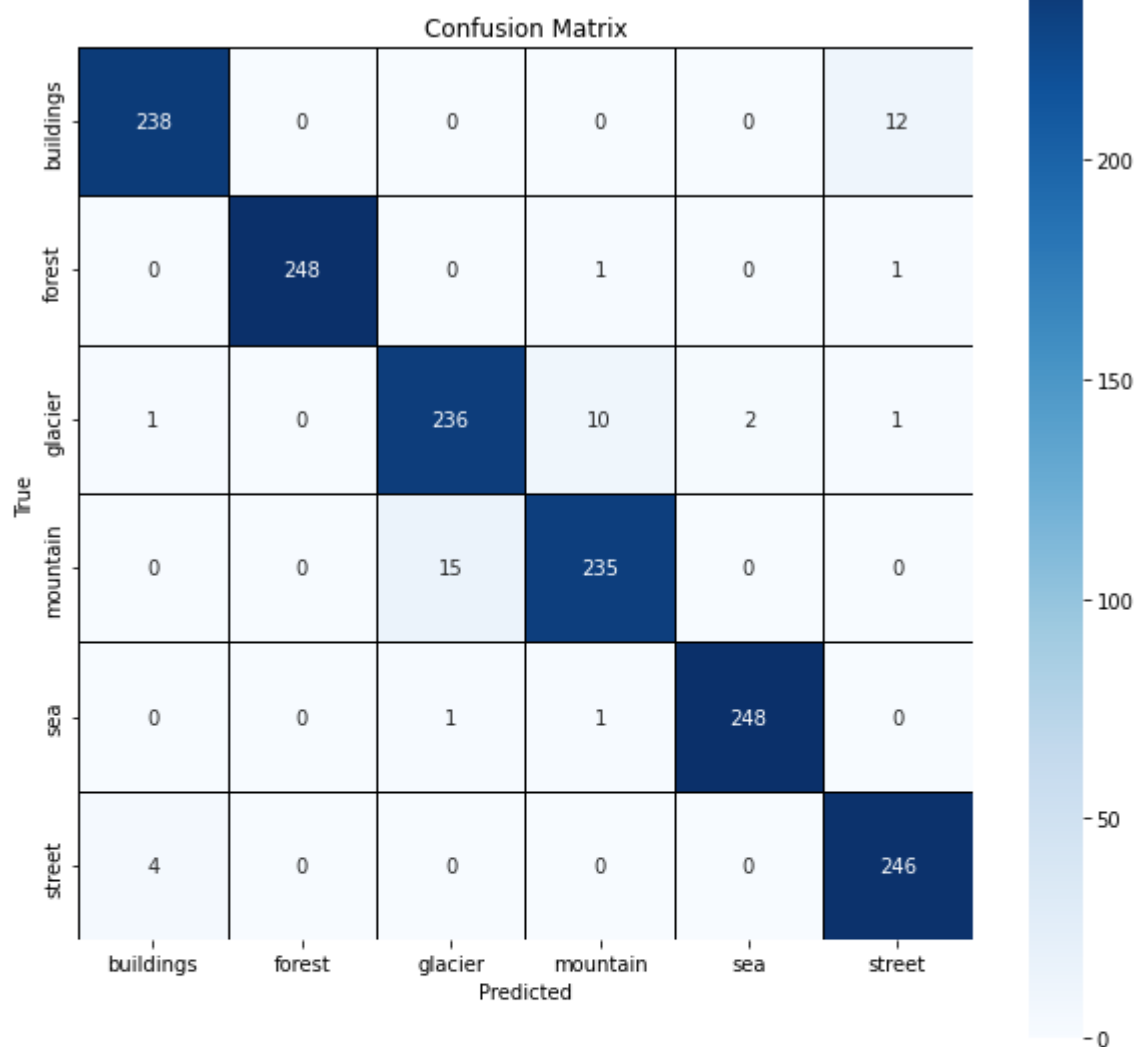
Metrics plotted after training.



Testing the best model (SGD optimizer and learning rate of 0.01)

Testing progress: 100%  24/24 [01:47<00:00, 3.59s/it, acc=0.929]  
Test accuracy: 96.73%

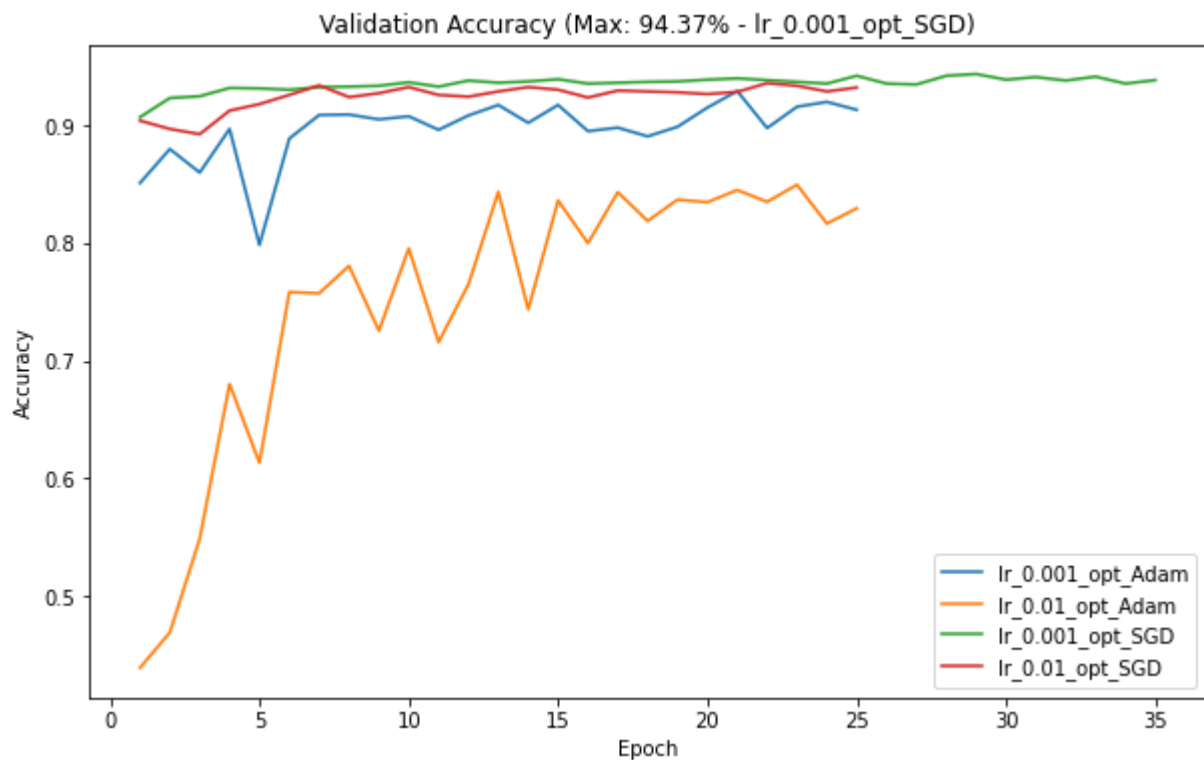
## Confusion matrix and Classification report



### Classification Report:

	precision	recall	f1-score	support
buildings	0.98	0.95	0.97	250
forest	1.00	0.99	1.00	250
glacier	0.94	0.94	0.94	250
mountain	0.95	0.94	0.95	250
sea	0.99	0.99	0.99	250
street	0.95	0.98	0.96	250
accuracy			0.97	1500
macro avg	0.97	0.97	0.97	1500
weighted avg	0.97	0.97	0.97	1500

Increased number of epochs to 35 for the best converging model



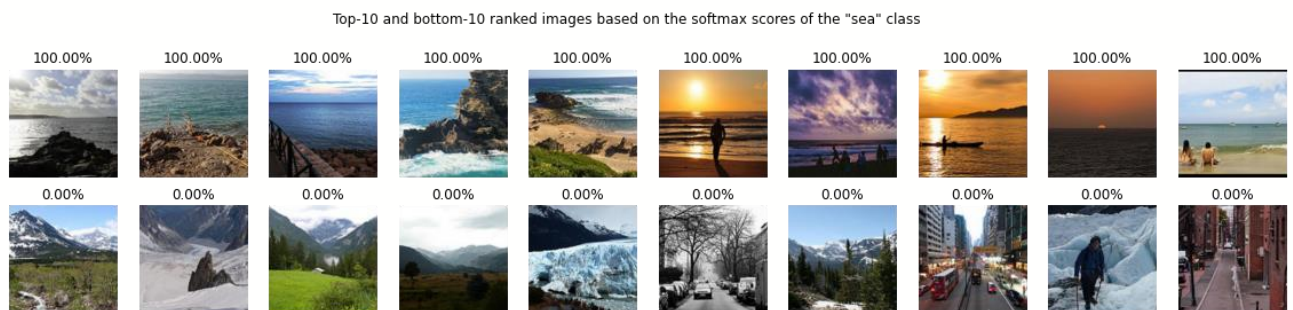
SoftMax scores tensor

```
[16] softmax_scores
```

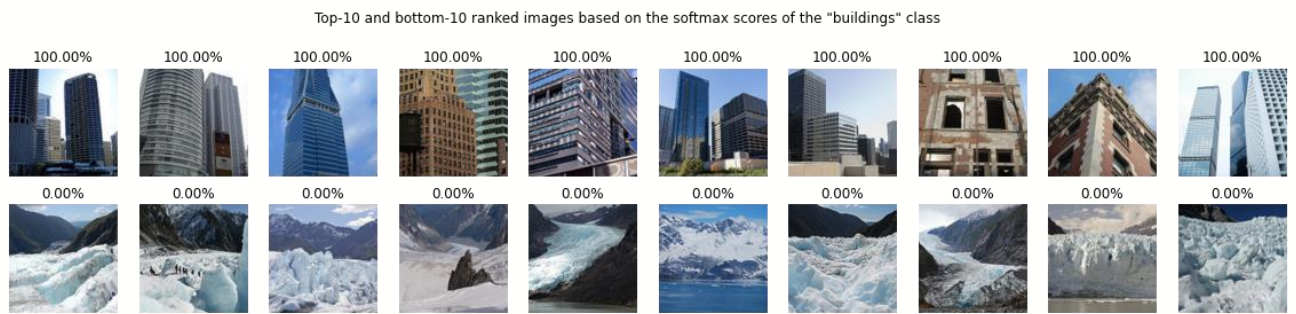
```
tensor([[1.2677e-04, 6.1474e-03, 1.3926e-03, 9.9227e-01, 4.5625e-05, 1.4502e-05],
        [3.3747e-04, 9.9440e-01, 5.1782e-03, 8.0230e-07, 2.5734e-05, 6.1893e-05],
        [1.4682e-06, 9.9974e-01, 2.5627e-04, 1.3607e-07, 5.2376e-07, 3.0957e-06],
        ...,
        [4.6122e-05, 1.3851e-05, 9.6874e-04, 9.3550e-05, 9.9885e-01, 3.1685e-05],
        [3.6631e-04, 1.0747e-03, 9.6500e-01, 3.0067e-03, 3.0245e-02, 3.1223e-04],
        [2.0704e-03, 5.2982e-06, 3.9158e-06, 2.9958e-07, 1.1883e-06, 9.9792e-01]])
```

Top-10 and bottom-10 ranked images.

based on the SoftMax scores of the «sea» class.



based on the SoftMax scores of the «buildings» class.



## Task 2: Internals of network feature map statistics

Extracted features and the percentage of non-positive values for selected feature maps

📁 Feature map layers:

```
maxpool : torch.Size([1, 64, 56, 56])
layer1  : torch.Size([1, 64, 56, 56])
layer2  : torch.Size([1, 128, 28, 28])
layer3  : torch.Size([1, 256, 14, 14])
layer4  : torch.Size([1, 512, 7, 7])
avgpool : torch.Size([1, 512, 1, 1])
```

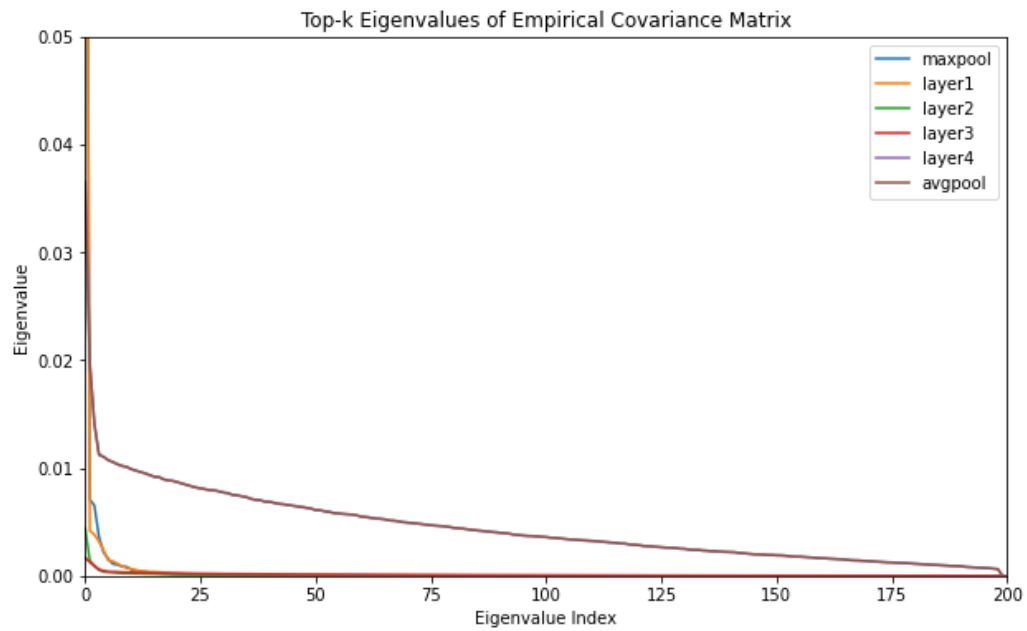
The percentage of non-positive values for selected feature maps:

```
maxpool : 23.30% non-positive values
layer1  : 23.58% non-positive values
layer2  : 49.18% non-positive values
layer3  : 55.48% non-positive values
layer4  : 43.03% non-positive values
avgpool : 1.34% non-positive values
```

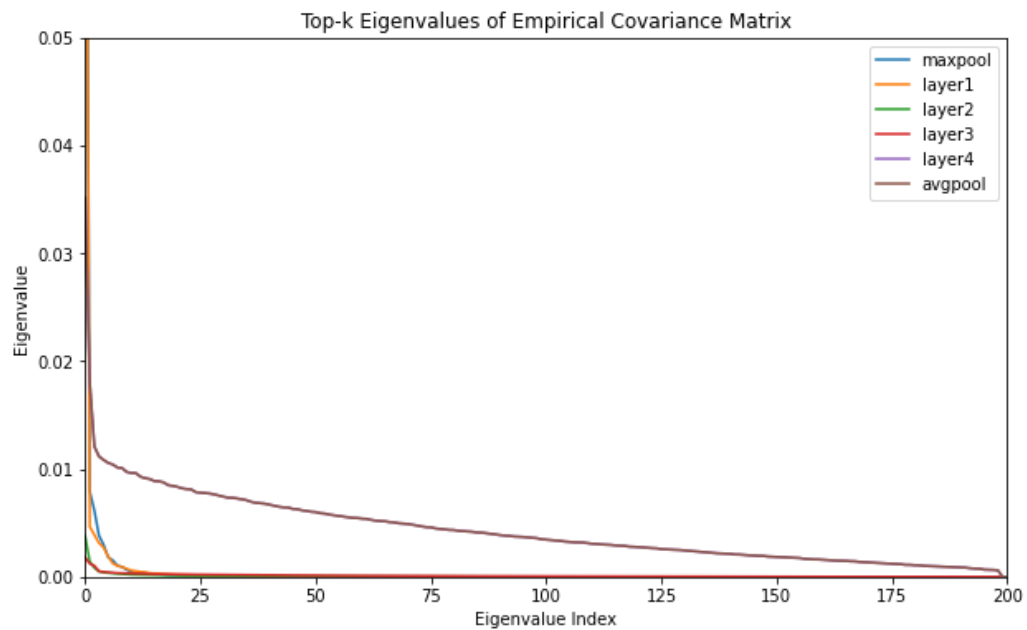
### Task 3: Internals of network feature map statistics (MSc)

Top-k eigenvalues of the empirical covariance matrix

ImageNet initialized model (= without any finetuning) and 200 ImageNet test images,  
upscaled to (224, 224)

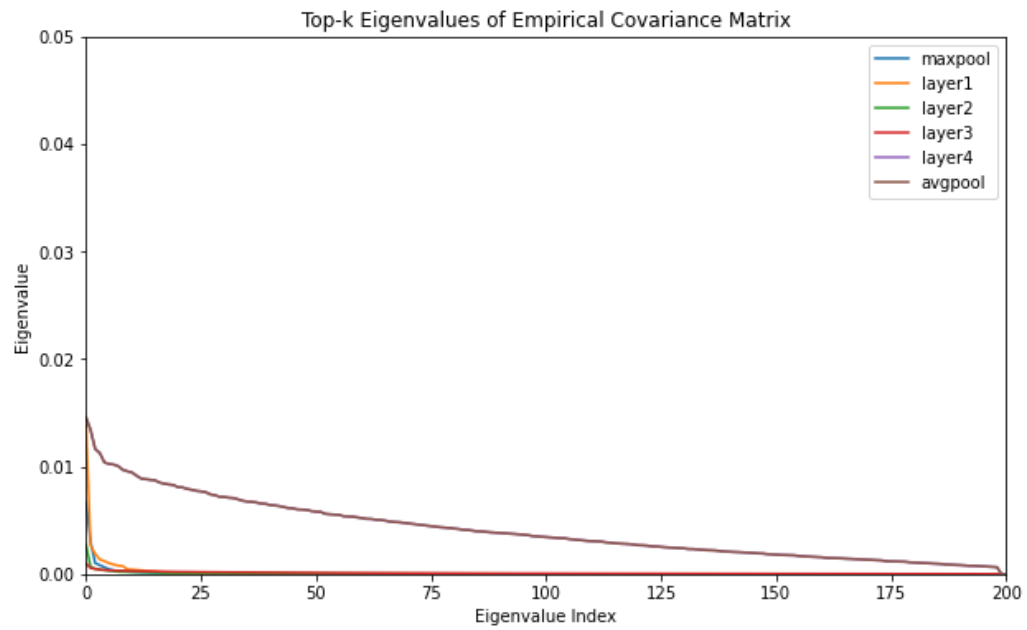


ImageNet initialized model (= without any finetuning) and 200 ImageNet validation images,  
scaled to (224, 224)





ImageNet model (= without finetuning) and 200 CIFAR-10 set images,  
scaled to (224, 224)



ImageNet model (**with finetuning**) and 200 ImageNet test images,  
scaled to (224, 224)

