Artificial Neural Networks Project

Self-Supervised Scene Classification

Assignment Report

Jakub Niedziela / s0224740

1. Introduction.

The aim of this academic project is to develop and evaluate the predictive capabilities of few distinct models, leveraging the principles of fully supervised and self-supervised learning techniques. The first two models are based on fully supervised learning, where the task is to classify different scenes from static images. These models will be trained using images paired with corresponding scene annotations. However, fully supervised learning is often limited by the need for extensive annotated data, which is expensive to acquire. To overcome this limitation, self-supervised training methods are proposed. These methods employ a technique called a 'pretext task'—a secondary process that helps extract generic feature representations from unannotated data. The objective is to adapt this feature representation for other tasks or use it as is. Following this, the aim is to compare these models. This involves determining their prediction accuracy, scrutinizing the visual explanations for the predictions made by each model, and interpreting the features internally encoded by the models visually.

In Section 2, a detailed description of the scene datasets utilized in this study is provided, laying the groundwork for understanding the data upon which the models operate. Following this, Sections 3 and 4 delve into the essentials of the experiment - they elaborate on the specific architectures deployed for the image classification task, along with the transformations and perturbations employed to facilitate this process. The results of models are presented in section 5. Moving forward, Sections 6 casts light on the explanation and interpretation methodologies harnessed in this study. These sections present the rationale behind the selection of specific methods and discuss their applications in generating insightful visualizations of the trained models. Finally, Section 7 wraps up the study by drawing conclusions based on the findings. This involves a comprehensive evaluation of the utilized methods, their efficiency, and the insights gleaned from the visualizations.

2. Dataset

The dataset at the heart of this project is a diverse collection of images, categorized into 15 distinct scene types, including office, kitchen, living room, bedroom, store, industrial, tall building, inside city, street, highway, coast, open country, mountain, forest, and suburb. This rich set of scene classes offers a wide spectrum of data to work with, thus presenting a robust challenge for the models being developed.

To facilitate model development and evaluation, the dataset is divided into two portions - a training set and a validation set. The training set serves as the foundational basis on which our model learns how to classify images. Each epoch of model training is followed by an evaluation phase, which uses the validation set. This ensures that assessment of the model performance on data it has not been trained on, therefore, giving us a more accurate measure of its predictive abilities.

This dataset is specifically designed for our project and contains black and white images. Although the images have three channels like coloured images, each channel holds the same value, resulting in a grayscale image. This adds another layer of complexity to our classification task, as we will be working without colour cues that can sometimes be vital in scene recognition.

The dataset is of a substantial size, containing a total of 4485 images. The training set comprises 1500 images, with an even distribution of 100 images for each of the 15 classes. This will ensure a balanced learning process, without any class being favoured due to a larger number of samples.

For the validation set, we have a larger pool of 2985 images. However, unlike the training set, the number of images per class in the validation set varies. This is likely to more closely mirror the real-world conditions, where the distribution of different scene categories would not be perfectly balanced. As such, it offers a robust means to evaluate the trained models' performance under more realistic conditions.

3. Supervised learning Scheme

3.1. Models Description

In this project, first task was to fine-tune a convolutional neural network architecture, specifically a pre-trained EfficientNet-B0 model, on our 15-scene dataset. The EfficientNet-B0 was initially trained on the ImageNet dataset and its architecture and training scheme were adapted for this task.

In the first attempt, the EfficientNet-B0 model was used, without any data transformations, except for resizing the images to a consistent shape of (224, 224). For the classifier, a linear layer with an input size of 1280 and output size of 15 was employed, using a dropout rate of 0.2 to prevent overfitting. Adam optimizer and CrossEntropy loss function were chosen, considered suitable for multi-class classification problems. The model was trained for a maximum of 100 epochs, with an early stopping mechanism that halted training if there was no improvement in validation accuracy for 15 consecutive epochs. The chosen batch size was 32. This approach led the model to early stopping at 41 epochs with a final accuracy of 98.07% on the training set and 88.48% on the validation set, therefore reporting substantial overfitting.

For the second attempt, in order to tackle the overfitting issue, the same architecture was maintained, but additional data transformations were incorporated. Weights normalization was applied to the images using the ImageNet values for both the training and validation sets, as well as RandomResizedCrop, RandomHorizontalFlip, and RandomVerticalFlip transformations to enhance the model's generalization capabilities. The same maximum number of epochs and classifier were used, but a ReduceLROnPlateau learning rate scheduler with a patience of 10 was introduced additionally. The early stopping criterion was extended to 30 epochs without improvement. This approach resulted in an early stop at 90 epochs, with slightly lower training accuracy at 97.6% but improved validation accuracy at 91.3%. Furthermore, the training and validation loss and accuracy presented much more stable learning curve, as can be seen in the training plot in appendix.

Additionally, for a comparative study, another training sequence was carried out, using the pre-trained EfficientNet-B0 model, but this time weights of the feature extraction part were frozen. The maximum of 100 epochs, an early stopping patience of 30 epochs were maintained. However, for data transformations, AutoAugments consistent with the ImageNet policy along with normalization were used. All other settings mirrored the second model. This experiment ran for the full 100 epochs and resulted in a training accuracy of 95.67% and a validation accuracy of 86.87%.

These experiments offered valuable insights into the influence of data augmentation, normalization, learning rate scheduling, and fine-tuning versus freezing of pre-trained model weights

on the model's performance. The second approach yielded the best validation accuracy, highlighting the importance of these strategies in achieving superior generalization performance.

3.2. Design choices

The choices made in designing the models were guided by a combination of recommended practices, computational efficiency, and the aim to boost the models' generalization abilities.

The selection of a batch size of 32 was based on a compromise between computational efficiency and the quality of the gradient estimate. Smaller batch sizes tend to provide a more accurate estimate of the gradient, which can aid learning. However, they also take longer to process. On the other hand, larger batch sizes can speed up training but might not converge as fast or to the best performance. A batch size of 32 is a common choice, offering a good balance.

Input normalization of images using the mean and standard deviation of the ImageNet dataset was applied, as the EfficientNet-B0 model was pre-trained on this dataset. Normalization helps in maintaining a consistent scale and distribution for the input data, making the learning process smoother and more stable. By using the ImageNet statistics, we ensure that our normalization aligns with the assumptions made when the model was originally trained (as pretrained weights are used).

Data augmentation techniques such as RandomResizedCrop, RandomHorizontalFlip, and RandomVerticalFlip were introduced in the second attempt to increase the diversity of the training set. They work by applying random transformations to the images, effectively increasing the size of the training set and helping the model generalize better to unseen data.

The introduction of a learning rate scheduler, specifically the ReduceLROnPlateau, was aimed at dynamically adjusting the learning rate based on the model's performance. By reducing the learning rate when the validation performance plateaus, the model can more finely adjust the parameters, potentially leading to improved performance.

The early stopping mechanism, with patience set at 15 epochs in the first attempt and 30 in the others, was introduced to prevent overfitting. By stopping training if there's no improvement on the validation set for several epochs, we ensure that the model doesn't just memorize the training data but generalizes well to unseen data.

The decision to freeze the feature extraction part in the third model was to test how well the pre-trained ImageNet features would generalize to our dataset without any fine-tuning. It's a common practice when the new dataset is relatively small or very similar to the original dataset the model was trained on. This decision provided an interesting contrast to the fully trainable models in the first and second attempts.

4. Self-supervised learning Scheme

4.1. Rotation Models Description

4.1.1. Rotation Classification

For the task of rotation classification, EfficientNet-B0 model, originally pre-trained on ImageNet was trained to identify the applied geometric transformation, or rotation, of an input image. The EfficientNet-B0 was reconfigured by replacing its original classifier with a new one, designed to assign images into one of four categories, each representing a rotation of 0°, 90°, 180°, or 270°. Similarly, to the fully supervised scheme, both the feature extraction and classifier parts of the model were made trainable in this phase. The model was fed images that underwent one of four specific rotation transformations: 0°, 90°, 180°, or 270°. These transformations expanded our geometric transformation

set, G, which now contains four unique image rotations. In a more formal definition, if Rot (X, φ) symbolizes an operation that rotates the image X by φ degrees, our geometric transformations encompass the four rotations G = g(X|y)4y=1, represented by g(X|y) = Rot (X, (y − 1)90). This correlates to φ = {0, 90, 180, 270}.

For the pretext task, each image was subjected to all four rotations, quadrupling the size of both the training and validation datasets. The transformations applied to the training data were solely RandomResizedCrop and normalization using ImageNet values. Given the rotation-centered nature of the task, vertical or horizontal flips were omitted. The model was trained for a maximum of 30 epochs with an early stopping patience set at 5. The classifier was a linear 1280-15 layer with a dropout rate of 0.2, furthermore Adam optimizer and Cross Entropy loss were employed. The learning rate was managed using the ReduceLROnPlateau scheduler with a patience of 3. After 21 epochs, the training was halted, with the model demonstrating a training accuracy of 95.65% and a validation accuracy of 96.01%.

4.1.2. Rotation Scene Classification

For the final rotation classification task, settings similar to those used in the pretext task were preserved. However, one key alteration was that freezing the feature extraction portion of the model. This strategic decision significantly trimmed down the number of trainable weights from a hefty 4,012,672 to a more manageable 19,215, thus making the model more computationally efficient. Additionally, the final classifier was adjusted to correspond to the 15 different classes found in the scene dataset, effectively tailoring the model to the specific requirements of this task. The model underwent training for the maximum 100 epochs, with no early stopping reported, resulting in a final training accuracy of 66.33% and a validation accuracy of 68.58%.

4.2. Perturbation Models Description

4.2.1. Perturbation Classification

In this phase of the project, the EfficientNet-B0 model, pre-trained on the ImageNet dataset, was trained, to classify specific perturbations made to an input image. Similar to the previous pretext task, it was achieved by removing the original classifier component of the EfficientNet-B0 and replacing it with a new classifier. This new classifier was designed to classify images based on two types of perturbations: black or white.

To implement the perturbations, a random square region on an image was defined, measuring 10x10 pixels, and altered the pixel values within this window to either 0 (black perturbation) or 255 (white perturbation). Consequently, an input image containing a black perturbation would be labelled as “0”, and an image with a white perturbation would receive a “1” label.

For the pretext task, a Random Horizontal Flip and a Random Vertical Flip for data augmentation were applied, in addition to normalizing the data using ImageNet values. For the validation set, only normalization was used. The model was trained for a maximum of 30 epochs with an early stopping patience of 5, using a linear 1280-2 classifier with a 0.2 dropout rate. The Adam optimizer was used alongside the ReduceLROnPlateau learning rate scheduler, which had a patience of 3. The model finished training after 10 epochs due to early stopping, achieving a training accuracy of 99.77% and a validation accuracy of 99.51%.

4.2.2. Perturbation Scene Classification

Moving to the scene classification task, the classifier was adjusted to a linear 1280-15 with a 0.2 dropout rate to suit the 15 classes within our scene dataset. Furthermore, same transforms were maintained as in the pretext task for consistency. Notably, the weights of the feature extraction part of the model were frozen, which substantially reduced the number of trainable weights from 4,010,110 to just 19,215. The maximum number of training epochs was set to 100 with an early stopping patience of 30. The same learning rate scheduler was used, but with a patience of 10 this time. The model completed training over all 100 epochs, resulting in a training accuracy of 54.47% and a validation accuracy of 50.49%.

Settings for all the models described in two previous sections are presented in Table 1.

Table 1. All models' settings.

Model	Supervised V1	Supervised V2 (Final Fully supervised)	Supervised frozen features	Rotation Pretext (Self-supervised)	Rotation Final	Perturbation Pretext (Self-supervised)	Perturbation Final
Image size	(224x224)	(224x224)	(224x224)	(224x224)	(224x224)	(224x224)	(224x224)
Learning rate	0.001	0.001	0.001	0.001	0.001	0.001	0.001
Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam
LR scheduler	No	Yes	Yes	Yes	Yes	Yes	Yes
Batch size	32	32	32	32	32	32	32
Max Epochs	100	100	100	30	100	30	100
Patience	15	30	30	5	30	5	30
Epochs Trained	41	90	100	21	100	10	100
Fully connected	(1280 → 15)	(1280 → 15)	(1280 → 15)	(1280 → 4)	(1280 → 15)	(1280 → 2)	(1280 → 15)

5. Results

The comparison of results obtained by each of the final 4 models as described in previous sections (fully supervised version 2, fully supervised with frozen feature weights, rotation, perturbation) comparison of results are presented in Table 1.

Table 2. Comparison of final models results.

	Fully supervised version 2 (final)	Fully supervised with frozen feature weights	Rotation Scene Classification	Perturbation Scene Classification
Train accuracy	97.60	95.67	66.33	54.47
Validation accuracy	91.29	86.87	68.58	50.49

While both fully supervised models exhibit a degree of overfitting on the training dataset, this is not an unusual occurrence when training intricate neural networks with limited amounts of training data. Overfitting is not necessarily indicative of poor model performance, especially if the model continues to perform well on the validation dataset. For example, despite its overfitting tendencies, the first model delivered a high level of performance on the validation set. This indicates that, despite the overfitting, the model was still successful in identifying relevant patterns from the training data, which enabled it to generalize effectively to unseen data. Therefore, it is important not to judge a model's capacity strictly by its overfitting behaviour, but also by its ability to accurately predict new data.

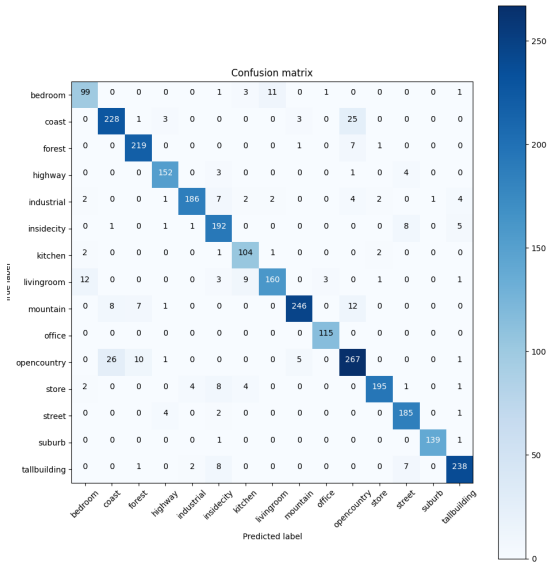
Regarding the scene classification results from the rotation and perturbation tasks, it is evident that they fall significantly below the performance of the fully supervised models. This suggests that during the process of re-tuning the feature extraction weights of the model to optimize it for the pretext task, the model becomes more attuned to identifying different types of features that may not necessarily correspond to object descriptors. Consequently, when classifiers are switched, the model's performance deteriorates, as evidenced by the higher performance of the ImageNet-pretrained model, which only requires adjustment to the classifier.

Thus, the takeaway from these results is that while pretext tasks might serve as a useful warm-up for models with randomly initialized weights, the application of these tasks to pretrained models does not seem to offer significant benefits in terms of enhancing model performance. Instead, these tasks appear to potentially steer the model towards learning features that may not be directly relevant to the main task at hand, which can ultimately lead to a reduction in overall model performance.

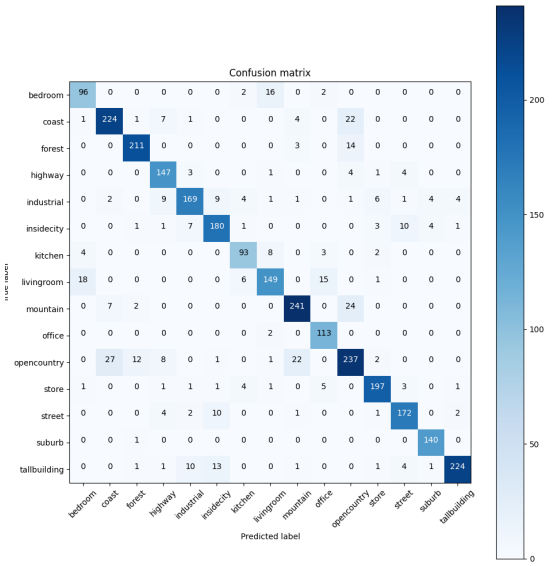
In addition to performance metrics, confusion matrices were generated for each of the models described, offering a more detailed breakdown of their class prediction capabilities. These matrices reveal insights into the predominant points of confusion for each model and are presented in Figure 1.

Fig.1 Confusion matrices for reported models.

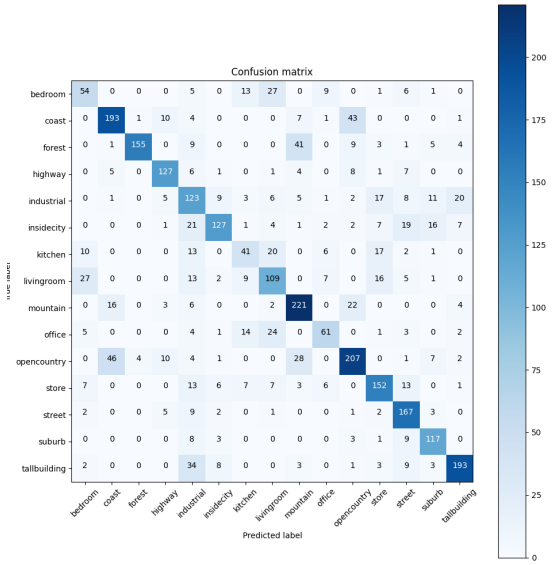
a) Fully supervised version 2



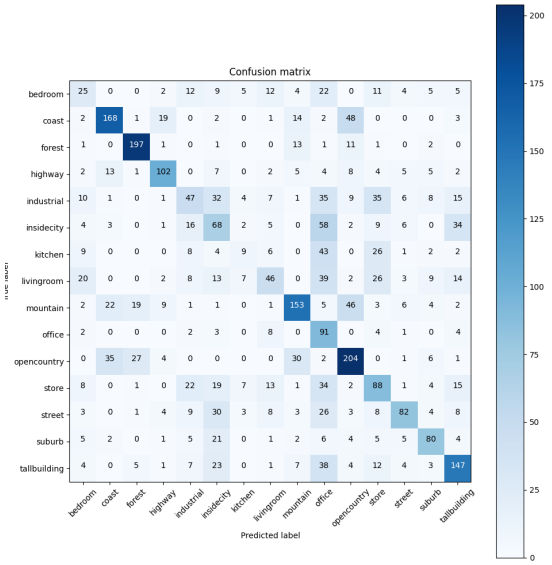
b) Fully supervised with frozen feature weights



c) Rotation Scene Classification



d) Perturbation Scene Classification



For the model with frozen feature extraction weights, the main points of confusion were found between certain pairs of classes: the living room and kitchen, the open country and coast, the open country and mountain, inside city and street, and inside city and tall building. However, when we fine-tune the feature extraction weights in the second fully supervised model, most of these issues are largely addressed, with the notable exception of the open country and coast pair. The improved differentiation between these other classes is reflected in the overall improvement of model accuracy.

The confusion matrices for the rotation and perturbation models paint a slightly different picture. While these models do correctly classify many instances, there is a noticeably higher degree of misclassification compared to the fully supervised models. For the rotation scene classification model, confusion is highest between the open country and coast classes, but there are several class pairs that the model struggles to distinguish correctly. For the perturbation model, the most frequent misclassifications occur between the inside city and office classes. Particularly poor performance was observed for the kitchen class, with the model correctly classifying only nine images as belonging to this class, while failing to identify many other images as being from this class.

6. Model Explanation and Interpretation

In this section, we delve deeper into the inner workings and interpretability of the trained models, commenting on both fully supervised and self-supervised approaches. We harness the power of two distinct methodologies to help demystify the reasoning behind the models' decisions. The first method employed is Score-CAM, a score-weighted visual explanation technique for convolutional neural networks. This approach enables us to generate intuitive, visually oriented explanations for our model predictions, highlighting key areas of interest within the input data. Secondly, we use a model inversion technique to provide further insights into our models. This method permits us to approximate the input data from the output predictions, effectively revealing what features our models focus on when classifying scenes. By applying these two methods, we aim to provide a deeper understanding of our models' decision-making processes, thus enhancing the transparency and interpretability of our results.

6.1 Score-CAM visualisations

For purpose of this visualisation 8 correctly classified images from 8 different classes were chosen and the score-CAM method was applied to output images with heatmap applied. Visualisations depict original image and class activation mappings from three different layers in model – first, second and last convolutional layer. Chosen classes are bedroom, coast, forest, highway, inside city, mountain, office and open country.

Figure 2 showcases a grid of images as interpreted by the fully supervised model. In the initial layers, the model's focus while making its predictions is somewhat difficult to ascertain. The heatmap does reveal some emphasis on delineating object boundaries, which potentially indicates the model's attempts to distinguish groups of similar pixels. However, in the second layer, the variances are even less discernible, with the network appearing to hone in on a multitude of small details, leading to a relatively homogeneous heatmap.

As we delve into the final layer, we observe that the model has, to some extent, learned to differentiate between objects, with the heatmap accentuating parts or whole objects within the image. For instance, in the "bedroom" class, the model's attention seems to be drawn to what can be identified as a bed. Similarly, for the "highway" class, the model predominantly concentrates on the road. Thus, we can infer that the model progressively refines its ability to distinguish relevant features as we move through the layers.

Figure 2. Score-CAM on fully supervised version 2.

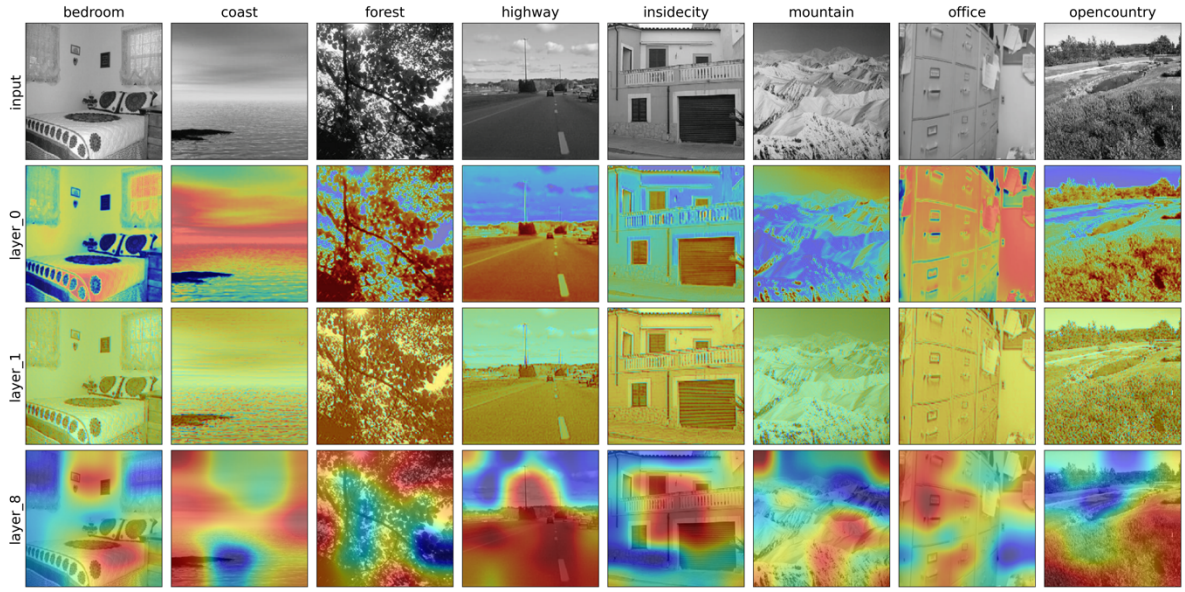


Figure 3. Score-CAM on fully supervised with frozen feature weights.

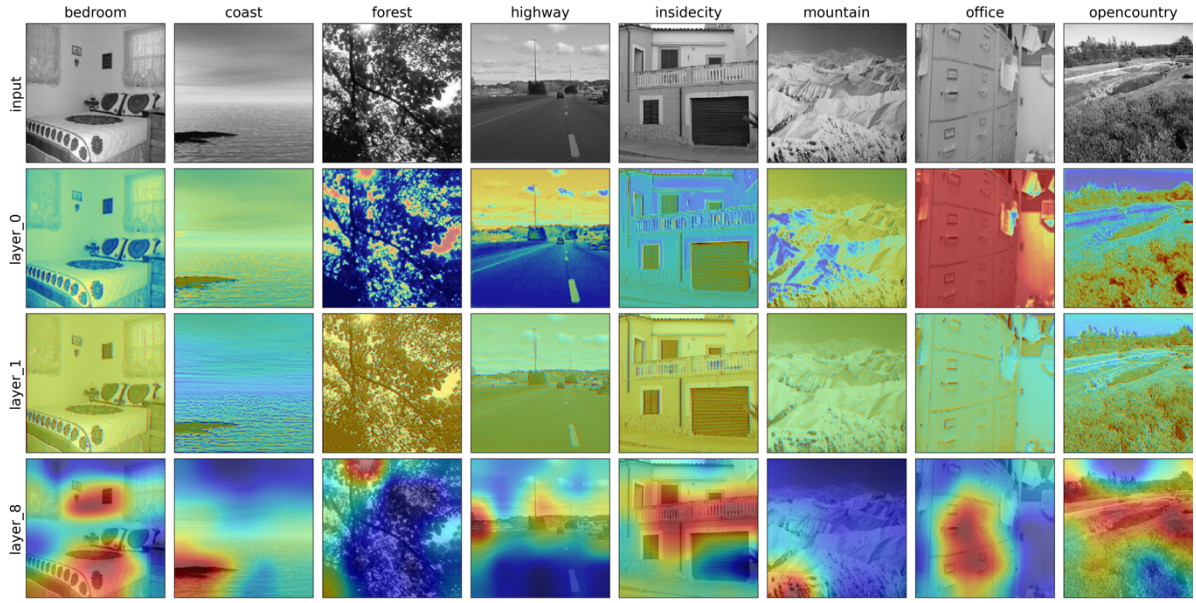


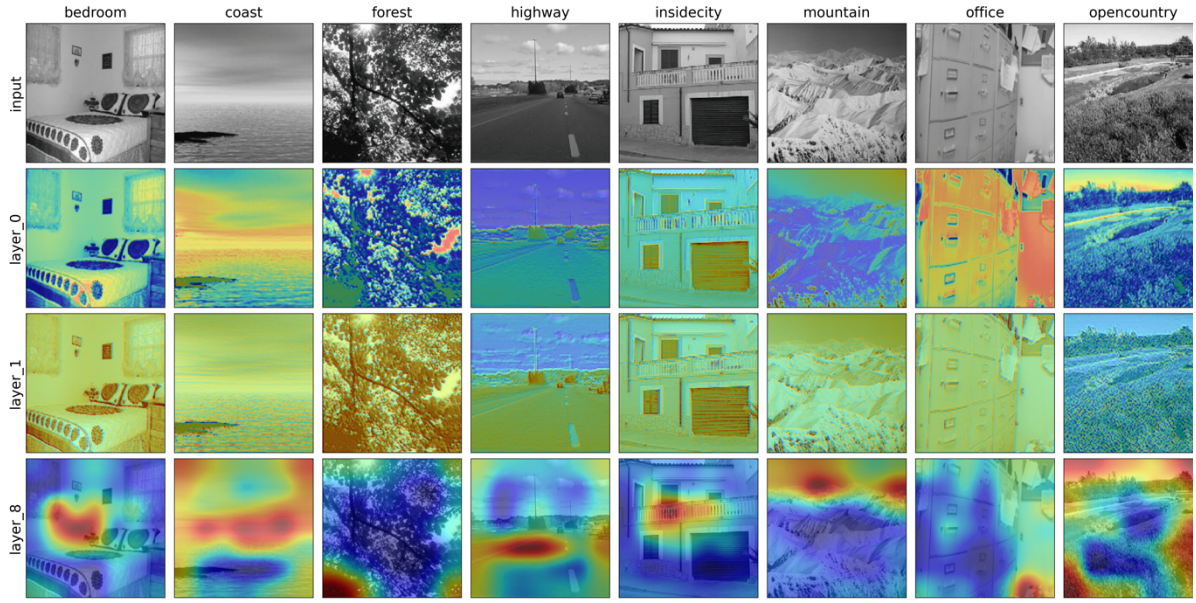
Figure 3 illustrates a grid of images as interpreted by the model with frozen ImageNet weights. The initial two layers bear considerable resemblance to the corresponding layers of the previously described model, although there are notable differences. For instance, we observe a colour switch along the boundaries of groups in classes like "highway" or "forest" in the first layer.

In the last layer, however, the differences become more pronounced. The regions of higher attention, indicated by the red portions of the heatmaps, are smaller and seem to differentiate between objects in a more coherent fashion as compared to the previous model where these regions sometimes span almost the entire image (as seen in classes like "coast", "highway", "office", and "mountains"). Yet, there are instances where both models show striking similarities. For example, in the "bedroom" class, both models exhibit high focus on and above the bed object, suggesting a shared understanding of certain features across different model.

Figure 4 illustrates the Score-CAM heatmaps applied to images for the rotation model. As with the previous models, it's challenging to discern distinguishable features in the initial two lower-level

layers. However, in the final layer, we can again observe larger regions displaying high attention values (represented by the warmer colours in the heatmap).

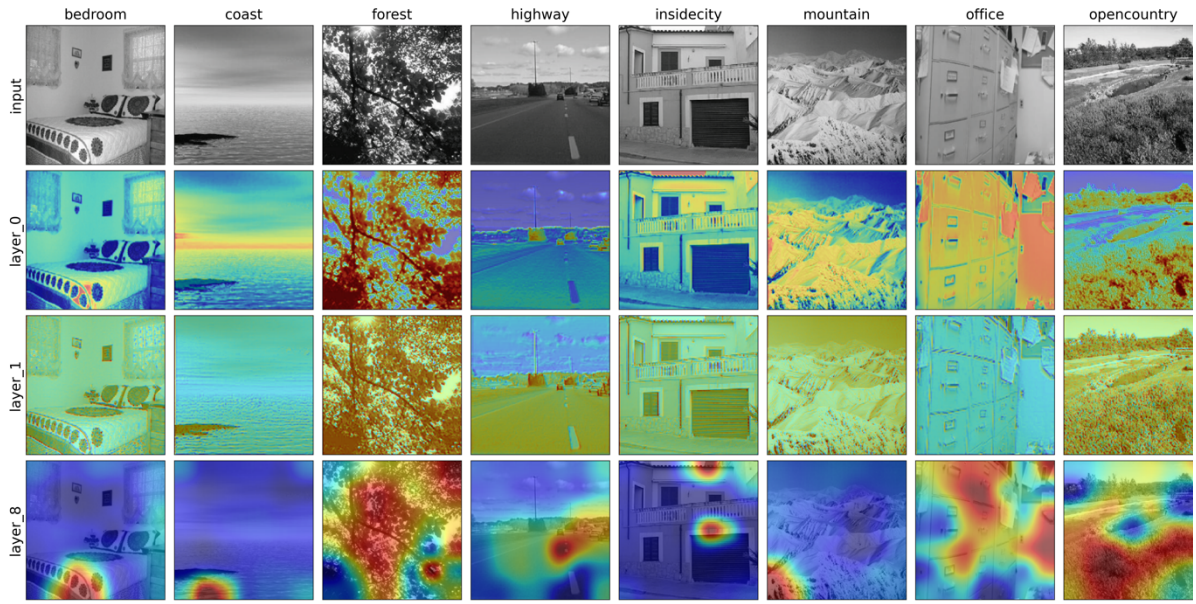
Figure 4. Score-CAM on fully rotation scene classification.



Even though this model correctly classified these examples, for most of the images, it seems to focus on different aspects compared to the previous models. For instance, in the "bedroom" class, high heatmap values are concentrated in the region between high-class activation mappings for the two previous models. This suggests that the rotation model may be interpreting and focusing on different features than the fully supervised models. Nevertheless, there are instances of overlap in interpretation as well; for example, in the "coast" class, the heatmap bears some resemblance to the one in the fully supervised model.

Figure 5 presents the Score-CAM results for the perturbation scene classification model. Intriguingly, compared to the rotation model, this model concentrates on relatively smaller regions of the image for several classes, such as "bedroom", "coast", and "inside city". This may be due to the model's adaptation during the self-supervised phase, where it learned to identify perturbations that were restricted to smaller regions within images.

Figure 5. Score-CAM on fully perturbation scene classification.



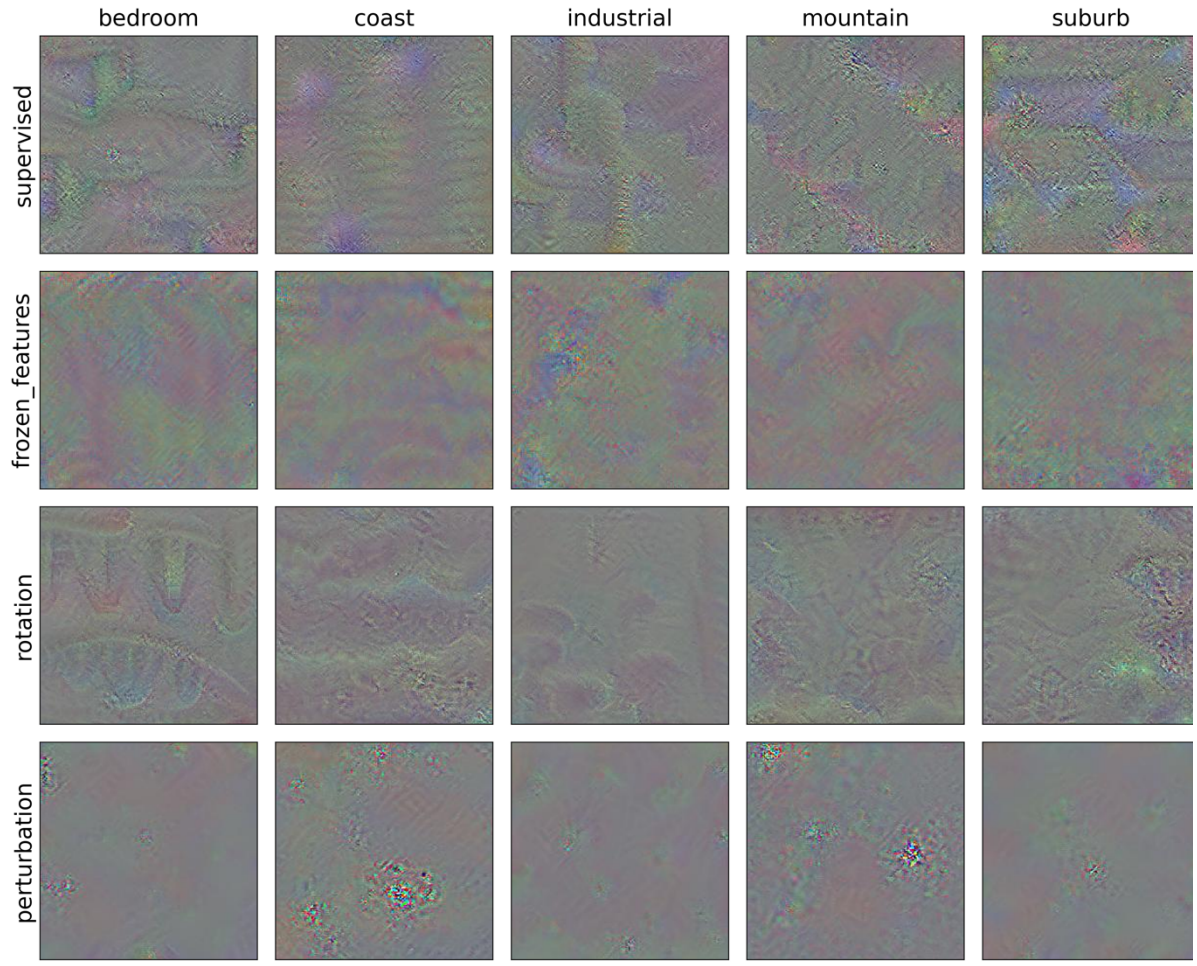
In the initial layers, the heatmaps are generally similar to those from all previous models. The second layer tends to focus on minuscule features, resulting in a homogeneous distribution of activation, while the first layer appears to distinguish between the boundaries of different objects to a certain degree.

In the final layer, the heatmaps bear some resemblance to those from the model with frozen ImageNet weights. This could potentially be attributed to the perturbation model's early termination after only 10 epochs of training, resulting in feature maps that are somewhat similar to those pretrained on the ImageNet dataset.

6.2. Model Interpretation

The Inverted Image Representations technique is designed to provide a visual depiction of specific internal units, elucidating the features these units encode. In this instance, we focus on five selected classes: bedroom, coast, industrial, mountain, and suburb. For each class, across the four models we've analysed, synthetic images were generated to elucidate these internal representations. The outcomes of this process are showcased in Figure 6.

Figure 6. Model Inversion results for 5 classes and 4 models.



For the 'bedroom' class, it's discernible that the most coherent images were generated by the fully supervised and rotation models. In the fully supervised model's synthetic image, one can arguably discern the vague shape of a bed and other smaller objects typically found in a bedroom. However, the synthetic image from the rotation model presents a challenge in interpretation, with features loosely resembling a cow's udder. The images generated by the frozen features model and perturbation model lack any distinguishable features.

The synthetic images for the 'coast' class reflect a similar pattern to those of the bedroom class. The fully supervised and rotation models depict what could be interpreted as waves on a coast. In contrast, the frozen features and perturbation models fail to produce discernible objects.

Regarding the “industrial” class, the supervised model crafts two identifiable shapes – one could be interpreted as the contours of a building (resembling a staircase), and the other as a chimney. In the remaining models, there are no discernible features that would resemble any familiar object. However, in the perturbation model, some small dots are evident, potentially due to the pretext task training that involves encoding black or white squares.

For the 'mountain' class, it's challenging to discern clear features in any of the synthetic images. One might argue that the fully supervised model presents something akin to the outline of a mountain range, but this is a rather tenuous assumption.

Lastly, the 'suburb' class synthetic image from the fully supervised model exhibits well-defined shapes. These shapes could be interpreted as the outline of a house or potentially a house's roof, aligning with the suburban imagery, which typically includes smaller houses with triangular roofs.

The similarities and differences observed in the synthetic images produced by the model inversion technique for each class across various models can be attributed to the distinctive learning methodologies and internal representations that these models have developed. The similarities seen in these synthetic images across the different models are a consequence of all models being trained on the same dataset and task. Despite the differences in training methodologies, all models are exposed to the same inherent distribution of features associated with each class. As a result, they all learn some common representative patterns of each class, which leads to the generation of similar synthetic images across models for a given class.

On the other hand, the differences can be ascribed to the unique learning strategies and the internal feature representations that each model has learned. For instance, the fully supervised models, which are directly trained to map images to their respective classes, might have developed more refined representations specific to each class. Therefore, they tend to generate more recognizable synthetic images compared to self-supervised models. In the case of the self-supervised models, these models are trained with pretext tasks, like rotation or perturbation, that force them to learn some form of invariant representations that might be slightly misaligned with the end task of scene classification. This discrepancy could cause the synthetic images generated by these models to appear less class specific. Furthermore, the models that have frozen weights from ImageNet pretraining carry forward the feature representations learned on ImageNet, which might not necessarily align perfectly with the classes in the current task. This discrepancy might lead to the generation of synthetic images that are harder to interpret or do not clearly represent a specific class.

In summary, these images do not always provide a clear or easily interpretable view of the features learned by the models. However, they can still offer some supplementary information about the internal mechanisms of these models and how different training approaches may affect what the model learns. This method, while not a definitive tool for model interpretation, serves as one of many techniques in our toolkit to attempt to understand the inner workings of complex neural networks. Even though clear-cut interpretations can be elusive, the technique provides a glimpse into the encoded patterns, particularly when used in conjunction with other methods.

7. Conclusions

This study provided a thorough comparative analysis of different learning paradigms for image classification, namely, fully supervised learning and self-supervised learning, using EfficientNet-B0. The experiment involved applying these paradigms to the 15 Scene Dataset and drawing insights from

the model performances, confusion matrices, visual explanations (via Score-CAM), and internal unit visualization (via Model Inversion).

The fully supervised models exhibited a level of overfitting, which, while not ideal, was not entirely unexpected given the complexity of these models and the limited amount of training data. This overfitting, however, did not hamper the models' ability to generalize to unseen data effectively, as evidenced by the high validation accuracies. The self-supervised models, trained on pretext tasks (rotation and perturbation), showed lower performance in scene classification compared to the fully supervised models. This suggests that while the pretext tasks serve as a good warm start for models with randomly initialized weights, they may not be the most effective means of fine-tuning pre-trained models.

Visual explanations using Score-CAM provided an understanding of what the models focused on when making predictions. For the fully supervised models, the heatmaps showed the models' ability to focus on salient regions related to the object of interest in the image. The self-supervised models, particularly the one trained on the perturbation task, focused on smaller regions, possibly due to the training requiring the identification of small, perturbed regions.

Model inversion shed light on the internal representations of the models. While the fully supervised models generated somewhat recognizable features, the self-supervised models struggled to produce clear visualizations. This suggests a disparity in the internal representations learned by these models, likely due to their different learning methodologies.

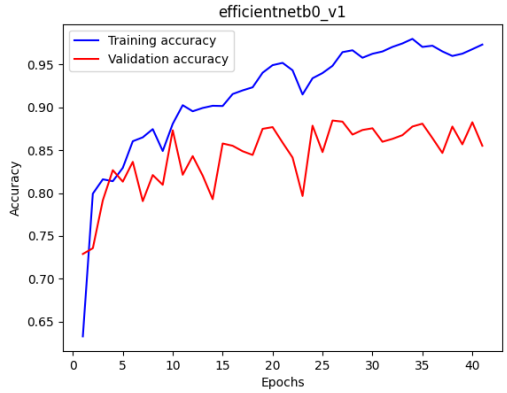
In conclusion, fully supervised learning and self-supervised learning exhibit different strengths and weaknesses when applied to image classification tasks. While the fully supervised models demonstrated strong performance and interpretability, the self-supervised models offered an alternative learning paradigm with its unique advantages. These findings highlight the importance of selecting the right learning strategy, considering the specific task and dataset at hand, and potentially combining the strengths of both paradigms to achieve the best performance. Future research could explore further strategies to enhance self-supervised learning, such as fine-tuning the entire model post-pretext task or exploring different pretext tasks that align more closely with the end-task.

APPENDIX

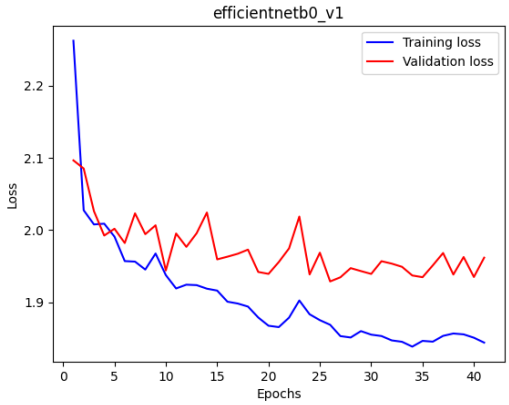
Training plots for all presented models

Figure 7. Training accuracy and loss for all models.

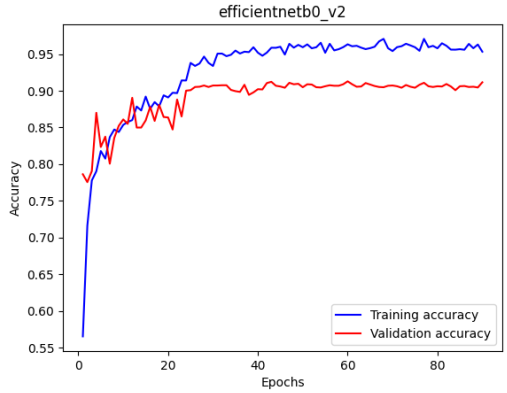
a) Supervised v1 accuracy



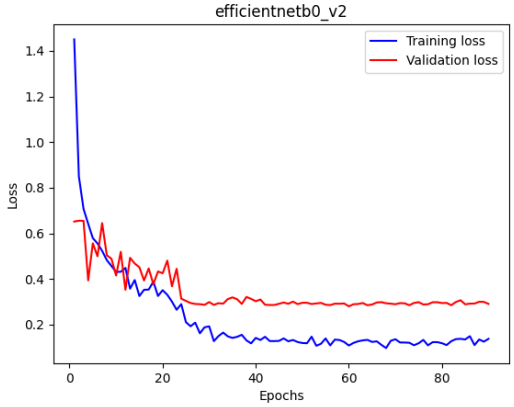
b) Supervised v1 loss



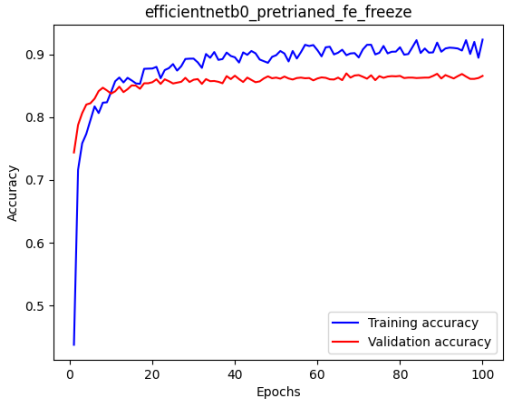
c) Supervised v2 (final) accuracy



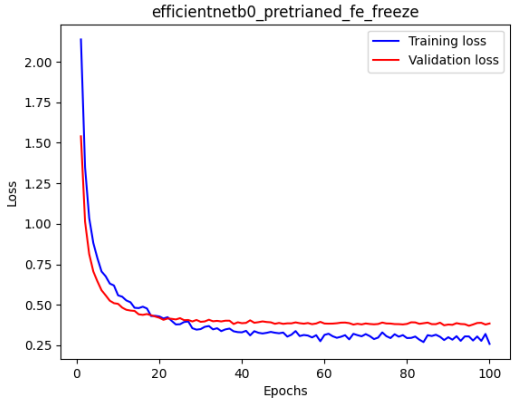
d) Supervised v2 (final) loss



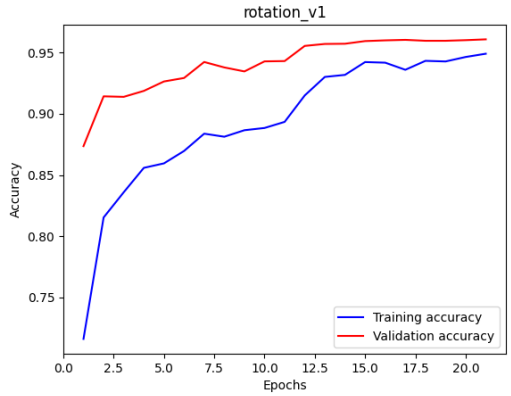
e) Pretrained features frozen accuracy



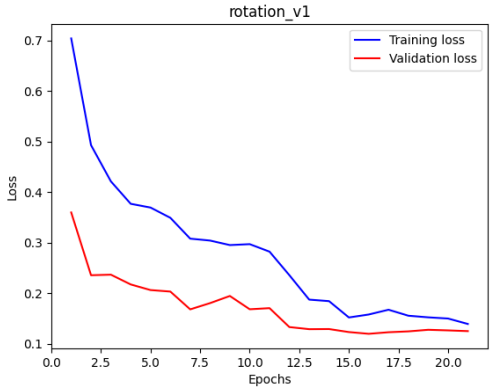
f) Pretrained features frozen loss



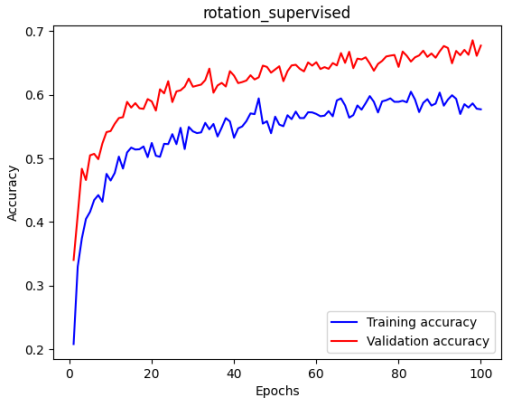
g) Rotation self-supervised accuracy



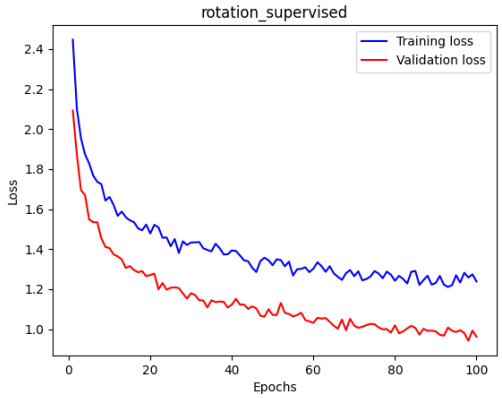
h) Rotation self-supervised loss



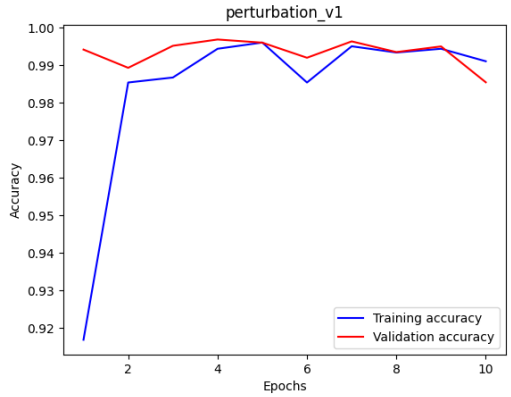
i) Rotation fully supervised accuracy



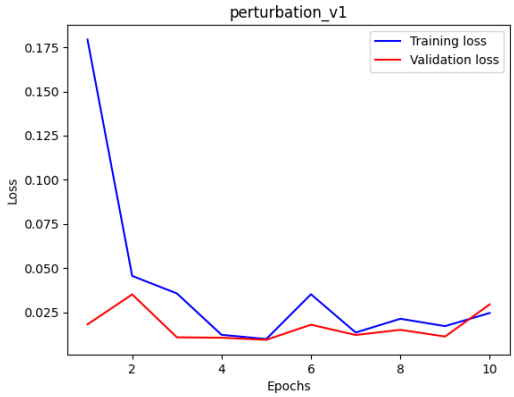
j) Rotation fully supervised loss



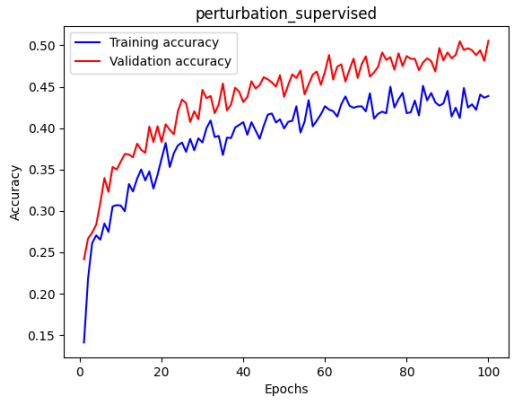
k) Perturbation self-supervised accuracy



l) Perturbation self-supervised loss



m) Perturbation fully supervised accuracy



n) Perturbation fully supervised loss

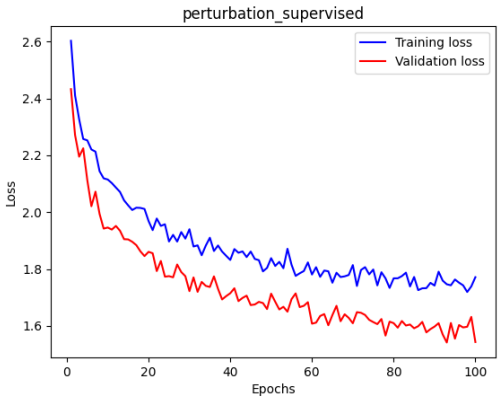


Figure 8. Rotation self-supervised confusion matrix

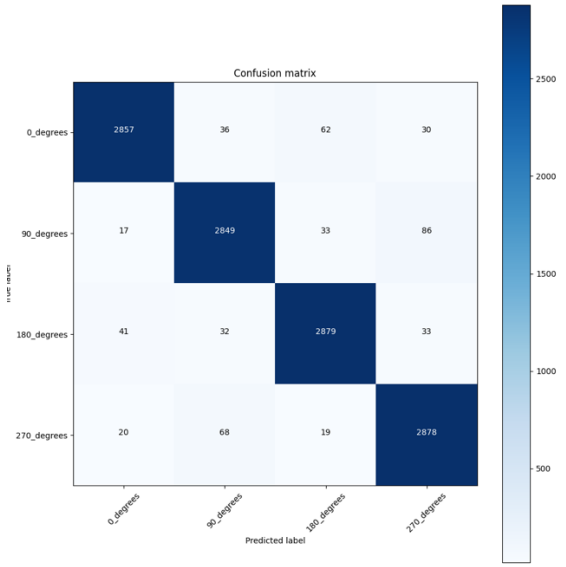


Figure 9. Perturbation self-supervised confusion matrix.

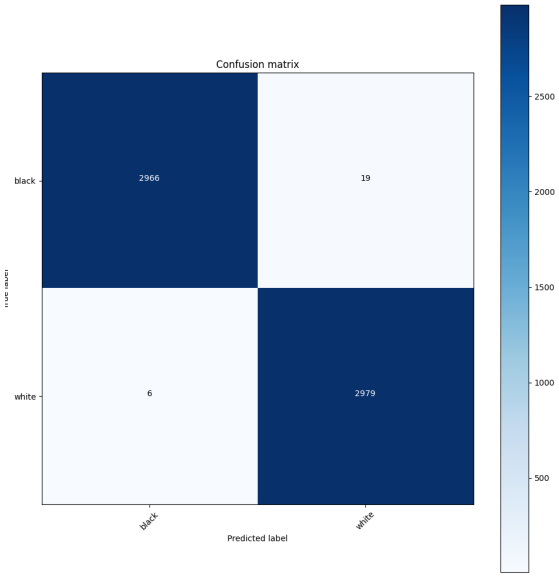


Figure 10. Examples of images with rotation perturbation applied.

Figure 11. Examples of images with black-white perturbation applied.

Project Directory Structure

.

├── 15SceneData	- Folder with the dataset
├── ANN_Project.pdf	- Assignment description
├── explore_dataset.ipynb	- Data Exploration
├── helpers.py	- Various helpful functions and classes
├── images_perturbation.ipynb	- Visualisation of rotations and perturbations
├── inversion.ipynb	- Model Inversion – Interpretation
├── misc_functions.py	- Additional functions for model inversion
├── models	- Folder with saved trained models
├── perturbation_classification_v1.ipynb	- Perturbation pretext task notebook
├── plots	- Folder with all saved plots
├── rotation_classifiction_v1.ipynb	- Rotation pretext task notebook
├── score_cam.ipynb	- Model Explanation – scoreCAM
├── semi_supervised_perturbation.ipynb	- Pretrained perturbation model for 15 classes
├── semi_supervised_rotation.ipynb	- Pretrained rotation model for 15 classes
├── supervised_model_v1.ipynb	- First version of fully supervised model
├── supervised_model_v2.ipynb	- Second (final) version of fully supervised model
└── supervised_pretrained.ipynb	- ImageNet pretrained model with frozen weights