Visual Explainability in Image Classification using Grad-CAM in Fine-Tuned InceptionV3, ResNet50 and VGG16 Models

João Valério joao.agostinho@estudiantat.upc.edu

Eirik Grytøyr grytoyr.grytoyr@estudiantat.upc.edu

Moritz Andres moritz.nicolas.andres@estudiantat.upc.edu

John Mocettini john.mocettini@estudiantat.upc.edu

Abstract

This paper delves into the cutting-edge field of Visual Explainability in Image Classification, utilizing the Gradient-weighted Class Activation Mapping technique on state-of-the-art models such as **InceptionV3**, **ResNet50**, and **VGG16**. Through fine-tuning these models with optimized parameters on the **CIFAR-10.2** dataset, we enhanced key metrics such as accuracy, precision, and recall. The true value of this research lies in using **Grad-CAM** to perform a deep dive into the visual explainability of these models, which unlocked a great understanding of the significance of this crucial aspect of deep neural networks. By looking at the best and worst classifications we were able to show that the models focused on irrelevant regions for the detection of some classes.

Keywords: InceptionV3, ResNet50, VGG16, ImageNet, CIFAR-10.2, Transfer Learning, Grad-CAM

1. Introduction

Deep learning models are becoming increasingly popular and are being used in a wide range of applications, such as image and language processing, healthcare, finance, and self-driving vehicles. These models are able to achieve state-of-the-art performance on a variety of tasks, but they are often perceived as being black boxes that are difficult to interpret and understand.

This lack of interpretability can be a barrier to the wider adoption of deep learning models, as users may be hesitant to trust or rely on a model if they are unable to understand the prediction or decision process [1].

In this project, the use of visual explainability in image classification using Grad-CAM and the process of fine-tuning three deep neural network models (InceptionV3, ResNet50, and VGG16) will be investigated.

The goal of the project is to understand the significance of visual explainability in deep neural network models, and to study the relation between hyper parameters, overfitting, and model architecture for qualitative and quantitative performance.

Firstly, the theoretical aspects of the project are briefly explained in order to clarify the base premises this study is focused on. Next, the models are fine-tuned using the CIFAR10.2 preprocessed dataset, in order to conclude the optimal configuration for each. In that process, accuracy, recall and precision will be used to estimate the generalization error.

Finally, Grad-CAM is utilized as a tool to perform a qualitative analysis of the visual explainability of the distinct models. In this part, it is intended to analyze edge case situations for each model, in order to comprehend the differences between them and the importance of explainability, a recently growing field.

[2] is an example of previous related work. This study from 2021, examines how grad-CAM can be used to explain visible characteristics with the VGG16, and Inception V3 model on Tea-leaves. They conclude that grad-CAM explains the factors, in a satisfactory way, with VGG16 giving the best result. In contrast, our study has higher focus on the transfer learning part, includes an extra model, and uses a more complex database, to see how generalizable this conclusion is.

2. Background

In the models' selection process, it was intended to determine algorithms from the same year, with room for improvement in terms of accuracy and some diversity regarding each's configuration, such as overall architecture functioning, depth, and parameters. Thus, each model's configurations, as well as Grad-CAM, are theoretically described in the subsequent units.

2.1. InceptionV3

Top-1 accuracy: 77.9%, **Depth**: 184, **Parameters**: 23.9M

The InceptionV3, first introduced in the paper '*Rethinking the Inception Architecture for Computer Vision*' [12] in 2015, is a model based on the Inception architecture, which was developed to address the problem of finding the optimal balance between the number of filters in a network and the computational cost of training and inference. It uses a series of Inception modules, which are composed of parallel branches with different filter sizes and configurations, to extract features from the input images. The modules are then connected to a global average pooling layer and a fully connected layer, which produce the final prediction.

The strong points of this model are its good performance, efficient architecture, and ability to handle a large number of classes. On the other hand, its weak points include its complex architecture, sensitivity to hyperparameter optimization, and the possibility to underperform on tasks that require a deeper network.

2.2. ResNet50

Top-1 accuracy: 74.9%, **Depth**: 107, **Parameters**: 25.6M

The ResNet50, first introduced in the paper '*Deep Residual Learning for Image Recognition*' [13] in 2015, is a model based on the ResNet architecture, which was developed to address the problem of training very deep neural networks. It uses skip connections, which allow the model to skip over one or more layers, to address the issue of vanishing gradients and improve the flow of information through the network. This allows ResNet50 to be trained very deeply (up to 50 layers) without suffering from the degradation of performance that is commonly observed in very deep networks.

Lastly, the major benefits and drawbacks concluded in prior works are listed. The positive aspects of this model are its good performance, deep architecture, and residual connections, while its negative ones include longer training time, more parameters, and underperformance on tasks where a more efficient network is required.

2.3. VGG16

Top-1 accuracy: 71.3%, **Depth**: 16, **Parameters**: 138.4M

The VGG16, first introduced in the paper '*Very Deep Convolutional Networks for Large-Scale Image Recognition*' [11] in 2015, is a model characterized by its simplicity and the use of small convolutional filters. It consists of a series of convolutional layers, followed by a few fully connected layers. The convolutional layers use 3x3 filters and use a stride of 1, which allows the model to learn spatial hierarchies of features. VGG16 also uses max pooling layers to downsample the feature maps and reduce the spatial dimensionality of the input.

Finally, the principal advantages and weaknesses inferred in previous studies are mentioned. Thus, the optimistic elements of this model are its good performance, simple architecture, and a large number of parameters to tune. The unfavorable ones consist of its long training time, sensitivity to hyperparameter optimization, and potential underperformance on tasks where a more efficient network is needed.

2.4. Grad-CAM

Gradient-weighted Class Activation Mapping (Grad-CAM), visualization technology is widely used in image classification, image captioning, and visualization of deep learning models [3].

First introduced in the 2016 paper '*Grad-CAM: Why did you say that?*' [4], it is a technique for visualizing the regions in an image that are most important for a CNN to produce a prediction.

It works by using the gradients of the output of the CNN with respect to the activations of the final convolutional layer to produce a heatmap, highlighting these important regions. Being able to see these areas where the CNN is focusing its attention can provide insight into how it is processing the input image.

In this project, Grad-CAM is used as a tool for understanding the behavior of the distinct CNNs, through visual explanations for the predictions assembled. From that perspective, it is possible to improve the models, as well as understand their premises in the decision-making process, which are of extreme importance in certain domains. Nonetheless, this only provides a qualitative explanation of the model's behavior

3. Methodology

This chapter describes the selected data and experimental steps. For the experiments Python, with Tensorlow is used. The grad-cam implementation is based on this: https://gist.github.com/RaphaelMeudec/e9a805fa82880876f8d89766f0690b54

3.1. Base Model Data Description

The CNN models described above provided by TensorFlow's library are used as a base of the original fine-tuning process. They are provided with pre-trained weights through the ImageNet dataset.

This is a large-scale image dataset that was created for the purpose of training and evaluating CNNs for image classification and object detection. It consists of more than 14 million images organized into over 20,000 categories [5]. These are arranged hierarchically, with higher-level categories containing more specific subcategories.

Using models with pre-trained weights to perform classification has a set of advantages, such as:

1) Since the pre-trained models have already been trained on a large dataset, they have already learned to recognize a wide range of features in images.
2) Pre-trained models save significant computational and human resources.
3) Pre-trained models serve as a good baseline for comparison (benchmark), in which it is possible to comprehend the effects of distinct fine-tuning approaches.

3.2. Fine-Tuned Model Data Description

In the current project, the data in the fine-tuning process is CIFAR-10.2, a variant of the CIFAR-10 image classification dataset developed in 2020, which can be found here: https://github.com/modestyachts/cifar-10.2. This dataset is derived from the same source as CIFAR-10 and assembled via a similar process in the paper '*Harder or Different? A Closer Look at Distribution Shift in Dataset Reproduction*' [6].

The CIFAR-10.2 consists of 12000 32×32 color images in 10 classes, with 1200 images per class. From all of the mentioned, 10000 corresponds to the training set and 2000 to the test set, in which the labels are equally represented. Furthermore, this multiclass dataset contains the following labels: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

According to its paper, the dataset is more difficult than CIFAR-10 with almost a 15% accuracy drop in their test on ANN. This increased difficulty can support the need for visual explainability and fine-tune in this study. Additionally, as the dataset contains low-resolution images and has few classes, the training and analysis are faster and simpler to perform.

3.3. Preprocessing

In order to feed the three models that are going to be developed, it is essential to preprocess the data mentioned above. The subsequent steps are applied in the following order:

1) **Outliers, Missing, Incorrect, Irrelevant and Redundant Values**: Since the dataset is predefined, it is expected that those aspects will not be an issue. Nevertheless, some examples from each category will be manually inspected.
2) **Normalization**: For each model, the normalization technique that was used for the training of the base model is used, performed by the predefined model-specific preprocessing functions in TensorFlow (scaled between -1 and 1 for InceptionV3 and zero-centered without scaling for the remaining).
3) **Encoding**: The categorical labels are converted into numerical ones through One-Hot Encoding.
4) **Split**: The split ratios are 10000 (83%) and 2000 (17%) for the train and test sets, respectively. During the training process, 10% of the data is used for validation and 90% for training.

Via this process, the proper dataset is obtained to perform classification.

3.4. Fine-Tuning

When the CIFAR-10.2 dataset is correctly preprocessed, the following steps will be conducted, in order to perform the transfer learning and fine-tuning of the models.

1) The additional hidden layers and the new output layer replace the original output layer of the model's architecture.
2) The base model, with the pre-trained weights, is frozen, while the added layers are unfrozen.
3) The model is trained on the CIFAR-10.2 dataset.
4) The second half of the layers in the base model will be unfrozen and trained with the added layers at a tenth of the learning rate.
5) Step 3 is repeated, however, the process is more meticulous because the initial weights are close to the convergence point.
6) The model is tested.

This procedure is conducted for different combinations of parameters. In order to perform a more efficient search, which involves multiple hyperparameters compared to the grid or random searches, the Bayesian optimization strategy is used [8]. Some of the parameters will be fixed and others variable throughout the optimization. The variable parameters are selected as they are considered important parameters according to [9]. In detail the search process is as follows:

To reduce the search space, the number of added hidden layers will not be a part of the variable hyperparameters. Instead, the process will add a new layer one-at-a-time, and define the optimal parameters. The process ends when adding a new layer doesn't improve the validation accuracy.
In the first added layer the fixed parameters are ReLU and output Softmax, in which the optimizer is Adam. Furthermore, the variable parameters are learning rate [10^{-3} – 10^{-5}], dropout rate [0 – 0.6], batch size [16, 32, 64] and loss function [Cross entropy, KLDivergence]. The number of iterations for the Bayesian optimization is set to 35.

When the best configuration is obtained for the first added hidden layer, the least layer-dependent parameters are fixed. This includes learning rate, loss function, and batch size, while the dropout rate and the number of units will be optimized for the new layer configuration. This trade-off is done to limit the search space and exploit the information from the previous Bayesian optimization. The number of hidden units in each layer has the relation shown in equation 1.

$$units_i = units_p * 2^{-i} \quad for\ 0 \leq i < n \tag{1}$$

Where "p" represents the variable parameter and "i" represents the layer number. non-integer values are truncated.

The process terminates when the addition of layers is not improving the final accuracy on the validation sets over the last 5 epochs, where a final optimal model configuration can be concluded through quantitative results (accuracy, precision, and recall).

As the main regularization, dropout layers are used. It was also considered to use weight decay, but as the experiments in this approved thesis describe [11], dropouts performed better when the dataset was small and the model is complex.

Lastly, for additional regularization and efficiency, the early stop strategy is defined by 5 consecutive epochs without improvement.

To validate the performance of the fine tuning, a benchmark for each model is obtained based on the grouping of the ImageNet classes to the Cifar10.2 classes. The grouping is performed with the

hyponyms of WordNets synset which is the basis for the classes of ImageNet in the first place. This maps the classes from Cifar to ImageNet. Note that we used the “bovid” instead of “deer”, as it was the closest semantic class in ImageNet

3.5. Visual explainability analysis

To the previous process, where the most suitable combinations are achieved, the complementary analysis tool, Grad-CAM, is used in the visual explainability of the results obtained.

The implementation is based on the implementation of Raphael Meudec [10]. The heat map can be generated from each individual layer block, which looks at different parts of the image. In order to properly detect the detailed regions, only the last layers in the models will be analyzed. This is then overlaid on the original image.

Further, this method will be used to examine strengths and weaknesses of the models, and to propose potential improvements of the training data. This will be in relation to the quantitative results and the individual classifications of the photos.

As a last note, the analysis is focused, predominantly, on a set of edge cases, weaknesses, and strong points of the models.

4. Results and analysis

Taking into account the previous details, quantitative and qualitative information regarding the models accomplished is obtained. The first metric is divided into three parts. The Bayesian optimization process, the training data, and the final test of the models. Furthermore, since the models were run on Google collab with varying resources (pro-version & non-pro, different sessions, and therefore resource allocation at Google’s whim), the training time is excluded from the analysis. Finally, the qualitative analysis regards the Grad-Cam image outcomes.

As a final note, all the results achieved are illustrated in appendix A for quantitative and B for qualitative.

4.1. Quantitative

Firstly, through Bayesian Optimization the best combination of parameters for the new layers was concluded, as shown in appendix A, table 1.3, 2.3, and, 3.3. Accordingly, only 1 extra hidden layer for the models presented the best validation accuracy. [ResNet50: figures A.1.1/A.1.2, InceptionV3: figures A.2.1/A.2.2, VGG16: figures A.3.1/A.3.2]. As the base models are already considerably complex and trained on a large set of data, the addition of layers might result in overfitting. In fact, those should only contribute to the models’ specialization.

Besides that, the accuracies of the parameter combinations tested for InceptionV3 compared to the other two algorithms demonstrated considerable stability, with a maximum variation of 8% [ResNet50: figures A.1.1, InceptionV3: figures A.2.1, VGG16: figures A.3.1]. However, when the second layer was added, ResNet50 and VGG16 appear to demonstrate more stability than InceptionV3 [ResNet50: figures A.1.2, InceptionV3: figures A.2.2, VGG16: figures A.3.2]. Except for some outliers, it seems like the models are robust to the selection of parameters. There was no clear convergence over the optimization, indicating that the variation in data and model randomness had a larger impact, as long as the parameter ranges are relatively good.

According to the previous analysis, the results of the most suitable combinations for ResNet50, InceptionV3, and VGG16 are largely diverse since the only common parameter is the Loss Function

(Cross-Entropy) [ResNet50: table A.1.3, InceptionV3: table A.2.3, VGG16: table A.3.3]. This underpins the theoretical differences between the models, as described in chapter 3. In fact, the loss function is similar, because this is mainly related to the classification type problem and the data.

Moreover, considering the best configuration, the fine-tuning process was performed by first training only the additional layer (stage 1) and then the whole model (stage 2) as described in 4.4 [ResNet50: figures A.1.4/A.1.5, InceptionV3: figures A.2.4/A.2.5, VGG16: figures A.3.4/A.3.5]. According to the results, it is detected as a general pattern that the specialization of the additional layer allows for the highest improvement in the training accuracy.

However, it is essential to indicate that, even though an early stopping strategy was implemented, the models, in both stages, seemed to overfit the data, since the progress achieved on the training set compared to the validation is not proportional. Especially in the second stage, it is an interesting fact that despite the large overfit, Resnet50 and VGG16 did not utilize the regularization parameter dropout rate in the best configuration. This might indicate that another regularization strategy can give better results. Nonetheless, this was accepted, because it was within the valid conditions for the training process and led to desirable results.

Despite the previous paragraph, it is inferred that the fine-tuned execution is correctly implemented as it was possible to considerably improve from the benchmark results described in Appendix A table 1.1.

Then, the models were tested with the test set, in order to obtain accuracy, precision, recall and the confusion matrices [ResNet50: table A.1.6, InceptionV3: table A.2.6, VGG16: table A.3.6]. Note that for the first 3 metrics mentioned, the values were obtained regarding stages 1 and 2 of the fine-tuning.

Thus, from the accuracy results, it is concluded that the best model, ResNet50, achieved the highest accuracy of 86%, while VGG16 was the worst one with 81%. Intermediately, InceptionV3 accomplished 85%. The main improvements in the models occurred in the adapting phase, where ResNet50, InceptionV3, and VGG16 had an absolute improvement of, respectively, 32%, 30%, and 30%, regarding the benchmark values. Furthermore, since the accuracies achieved in the adapting phase were extremely similar, the main difference between the final models' accuracies occurs in stages 2. While ResNet50 and InceptionV3 were able to improve by 9% and 10%, respectively, VGG16 increased only by 6% the accuracy [ResNet50: table A.1.6, InceptionV3: table A.2.6, VGG16: table A.3.6].

Relative to the average precision and recall values, the ranking relation between the models is similar to the accuracies described. Further, the precision is never lower than the recall, indicating that the true positive cases detected among the positive predictions are equal to or higher than the true positive among the real true positives. This might suggest that the models are not identifying an excess of true positives or, in other words, are not biased into a certain label [ResNet50: table A.1.6, InceptionV3: table A.2.6, VGG16: table A.3.6].

Finally, according to the confusion matrices, fine-tuning improved the results of the classification task. In fact, between stages 1 and 2 there was always an increase in the true positives in the ResNet50 and InceptionV3 models. In the VGG16 the scenario was not as optimistic, however, only in 4 labels (automobile: 0%, cat: -1%, frog: 0%, and ship: -2%) there was no improvement. Additionally, the major improvements between stages were 34% (horse), 15% (automobile), and 16% (airplane and deer) in the ResNet50, InceptionV3, and VGG16 algorithms, respectively [ResNet50: figures A.1.7/A.1.8, InceptionV3: figures A.2.7/A.2.8, VGG16: A.3.7/A.3.8].

By considering only the second stage, through the top 3 classifications it is understandable that the results achieved by ResNet50 and InceptionV3 are more similar to each other than to VGG16. While these models share horse, dog, and automobile in their top 3 classifications, the only similarity for VGG16 is with ResNet50 for the bird label. Furthermore, the worst 3 classifications (cat, truck, and airplane) for ResNet50 and InceptionV3 are entirely coincidental. However, considering VGG16, only the cat label is similar. As a final point, it is interesting to denote that while automobile and frog are the most difficult labels for VGG16, they are in the top 3 of ResNet50 and InceptionV3. On the other hand, while ResNet50 and InceptionV3 present difficulties identifying trucks, for VGG16 this label is the best one [ResNet50: figures A.1.7/A.1.8, InceptionV3: figures A.2.7/A.2.8, VGG16: A.3.7/A.3.8].

From this, it is possible to conclude that ResNet50 and InceptionV3 are the closer models due to their underlying configuration. Besides that, they are also the most suitable ones for the task in the study.

4.2. Qualitative

From the previous methodological process the most suitable models were obtained and tested in the test set, in order to use Grad-CAM and perceive the main regions of focus. The results obtained are displayed in Appendix B and they are complemented with the accuracy values from the confusion matrices, as well as the human accuracy according to [15]. This analysis is presented by category.

Frog (ResNet50 88%, InceptionV3 86%, VGG16 80%, Human 95%): All models focus on the same frog's characteristics, the face, attributing intermediate importance to the body. However, when the scale of the frog is reduced images are labeled incorrectly. The models had reasonable activations for images of multiple objects although they were not trained to identify multiple objects. Finally, the color between the background and the animal is also a misleading clue for the models.

Cat (ResNet50 76%, InceptionV3 77%, VGG16 66%, Human 85%): In the cat labels, the performance of the models is not sufficiently good. In fact, in all the models the regions of focus are not deeply intuitive, since these select particular regions of the cat's face, even though the disposition is symmetric in the image. Precisely, VGG16 and InceptionV3 focus on the ear as a main component for the identification. Furthermore, ResNet50 and VGG16 consider that the nose is also an important element. Thus, these distinct aspects seem to be a human-like approach.

Dog (ResNet50 88%, InceptionV3 90%, VGG16 82%, Human 93%): The performance of the three models is suitable, especially for InceptionV3. According to the results, ResNet50 and VGG16 focus mainly on the eye area of the dog, the InceptionV3 takes into account the whole face, which might be the distinctive factor of the accuracy achieved. Notably, the ears are not an area of interest for the models. This might be due to the high intra-class and geometrical variance of these, which can be perceived as an irrelevant component to focus on.

Horse (ResNet50 90%, InceptionV3 90%, VGG16 82%, Human 96%): In this label, it is seen that ResNet50 and InceptionV3 focus on the body of the horse, while VGG16 emphasizes the head, which, according to the results, the first strategy seems to be the most useful one. However, from a human perspective, it would probably be more useful to have access to the head of the horse than to the torso to perform an identification.

Deer (ResNet50 86%, InceptionV3 85%, VGG16 88%, Human 90%): According to the results all the models performed approximately equally, and their strategies are also similar since the three tend to focus on the horns of the deer. This is understandable considering the domain since it is a quite unique characteristic. However, focusing on the torso too would be a relevant additional region for identification, as the predictions without large horns are misclassified.

Bird (ResNet50 88%, InceptionV3 85%, VGG16 85%, Human 95%): In the provided examples, all three misclassifications have merit, as the image for VGG16 does resemble a plane, even to the human eye, while the others share the same close-up image of a large bird's face, quite different from the typical bird.

Airplane (ResNet50 84%, InceptionV3 76%, VGG16 84%, Human 94%): The preferable models in plane classification, ResNet50, and VGG16, have apparently the same strategy as InceptionV3 since all of them try to identify the body of the airplane, however, the latter has more difficulty in doing so. Interestingly, none of them considers the wings as an important component. In fact, the models misclassify some birds as airplanes, due to the body structure when in flight.

Automobile (ResNet50 89%, InceptionV3 90%, VGG16 74%, Human 97%): The significant differences in performance rely on the areas of the image considered, since the latter focus on the wheels of the automobile, while the remaining pay special attention to its shape, which seems to be the most intuitive approach. Focusing only on the wheels is an undesirable feature, as it is a bad representation of a car.

Truck (ResNet50 83%, InceptionV3 82%, VGG16 89%, Human 97%): Without a doubt, all three models suffer from inter-class variation between autos and trucks, when the trucks don't have the "angular shape" however, they still perform well. As mentioned above, VGG16 shows a unique focus on the wheels of vehicles, which in this case, seems to have a positive effect on the Truck class.

Ship (ResNet50 87%, InceptionV3 92%, VGG16 80%, Human 96%): In this label, the three models appear to select the front of the ship as the main feature, which leads to suitable results. Particularly, it is notable that, even though InceptionV3 achieved the soundest results, it can focus on areas not related to the ship, the water. However, according to the domain and the data, in fact, this can be a good indicator. Further, VGG16 also focuses on unrelated areas to the object like the boat carrier which is undesirable and can be seen as overfitting to the dataset with respect to the real world.

In summary, some general patterns can be pinpointed. Usually, the main areas of focus can be non-intuitive for us, humans. Firstly, while we tend to focus on the overall shape of the object/animal and on some distinctive characteristics, the models rely directly on the latter, which partially can explain the overfitting, from the quantitative analysis. For example, focusing on the wheels of an automobile on an image with low resolution can lead to identification mistakes. However, the models operate on a strict domain with only 10 classes, where they try to exploit all the possible unique characteristics to be successful. Thus, it is inferable that these depend heavily on the domain. For instance, in a dataset of vehicles, the wheels wouldn't be a decisive factor. As a last point, the consideration of the background can play a deceitful or orientational role in the labeling.

Looking at the images of interest it was interesting to find images that did not belong to any class (e.g. the ball-shaped object misclassified as an airplane). These out-of-distribution (OOD) samples were interesting to analyze with GradCam, as it gives insights into a more high-level generalization capability of the models. Further, it was interesting to examine the focus of the models on images with multiple objects (e.g. multiple frogs), as it requires the classifier to choose the most relevant object implicitly.

Finally, a reasonable way to improve the result would be to add more variety to the database. It seems that in general, the classifiers are struggling when the images are close up, or the background is complex. Thus, adding more of those examples would be helpful. An alternative is to duplicate the

images with augmentation techniques, such as cropping, translation, or random erasing, as described in [7].

As a last note, it is relevant to note that the overall human performance on CIFAR-10 was estimated by [15] at 93.91% on average and is therefore also not perfect. This can, however, also be due to bad/ misclassified samples. Comparing our models to this performance creates confidence regarding the models' performance.

5. Discussion

As the project has limited time and computational resources, there are some aspects that should be taken into account. One of the main limitations is that the optimization process was performed using a single train-validation split, and k-fold cross-validation was not used. This could have provided more robust results and reduced the variance. Furthermore, statistical methods such as the Friedman test are not used to determine the best model.

The optimization process could also have been improved by including more hyperparameters, or longer optimization processes.

Finally, the use of Grad-CAM for visual explainability analysis also has some limitations associated. A limited number of samples have been analyzed and Grad-CAM assumes that the features that are highlighted by the heatmap are globally important for the decision-making process of a model, which might not be the case. That being said, the analysis demonstrated an interesting approach to detecting outliers with human revision by looking at bad classifications and could be an interesting topic for further research. In any case, the methods used provide useful and indicative insight into the field.

6. Conclusion

The introduced goal of the project was to understand the significance of visual explainability in deep neural network models, and to study the relation between hyperparameters, overfitting, and model architecture for qualitative and quantitative performance. A range of these hyperparameters was specified, and though not perfect, Bayesian Optimization was useful in efficiently finding effective combinations. Through the application of transfer learning combined with fine-tuning, the experiment generated effective classifiers on a new dataset, all achieving accuracy measures of over 80%. It was noticeable that the models through a two-step tuning process like this converged to relatively good results in most of the defined parameter combinations.

Even Though the performance was good, it was shown that all the models were significantly overfitting, limiting a higher test performance. Through the testing of regularization parameters and visualization, the study indicates that adjusting the dataset based on focus areas might give a better generalization ability, but further studies should confirm this.

Comparing the details of the three models' classification successes and errors through visualization, clear differences can be observed between the individual models, and offer insight and explainability into their performance. This speaks to the utility of visualization in Neural Networks, an emerging technology that this study set out to explore. Evidently, it should be leveraged in future works in the field of AI, contributing to more socially-acceptable and explainable techniques.

References

[1] R. LaLonde, D. Torigian, og U. Bagci, «Encoding Visual Attributes in Capsules for Explainable Medical Diagnoses», i Medical Image Computing and Computer Assisted Intervention – MICCAI 2020, Cham, 2020, s. 294–304. doi: 10.1007/978-3-030-59710-8_29.

[2] P. Banerjee og R. Barnwal, Visual explanation using Grad-CAM for deep-learning based classification of Tea-leaves. 2021.

[3] Y. Li, H. Yang, J. Li, D. Chen, og M. Du, «EEG-based intention recognition with deep recurrent-convolution neural network: Performance and channel selection by Grad-CAM», Neurocomputing, bd. 415, s. 225–233, nov. 2020, doi: 10.1016/j.neucom.2020.07.072.

[4] Selvaraju, Ramprasaath R., et al. "Grad-CAM: Why did you say that?." *arXiv preprint arXiv:1611.07450* (2016).

[5] «Papers with Code - ImageNet Dataset». https://paperswithcode.com/dataset/imagenet (Opened 2. january 2023).

[6] S. Lu *mfl.*, «Harder or different? a closer look at distribution shift in dataset reproduction», i *ICML Workshop on Uncertainty and Robustness in Deep Learning*, 2020.

[7] N. E. Khalifa, M. Loey, og S. Mirjalili, «A comprehensive survey of recent trends in deep learning for digital images augmentation», *Artif Intell Rev*, bd. 55, nr. 3, s. 2351–2377, mar. 2022, doi: 10.1007/s10462-021-10066-4.

[8] J. Wu, X.-Y. Chen, H. Zhang, L.-D. Xiong, H. Lei, og S.-H. Deng, «Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization», Journal of Electronic Science and Technology, bd. 17, nr. 1, s. 26–40, mar. 2019, doi: 10.11989/JEST.1674-862X.80904120.

[9] T. Yu og H. Zhu, «Hyper-Parameter Optimization: A Review of Algorithms and Applications». arXiv, 12. Mars 2020. doi: 10.48550/arXiv.2003.05689.

[10] Meudec, Raphael. "Grad-CAM in TensorFlow 2." https://gist.github.com/RaphaelMeudec/e9a805fa82880876f8d89766f0690b54.

[11] Slatton, T. G. (2014). A comparison of dropout and weight decay for regularizing deep neural networks. Computer Science and Computer Engineering Undergraduate Honors

[12] IOFFE, Sergey; et al. (2015). *Rethinking the Inception Architecture for Computer Vision*. UK: University College London.

[13] HE, Kaiming; *et al.* (2015). *Deep Residual Learning for Image Recognition*. USA: Microsoft Research.

[14] SIMONYAN, Karen; ZISSERMAN, Andrew. (2015). *VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION*. UK: Visual Geometry Group, Department of Engineering Science, University of Oxford.

[15] Ho-Phuoc, Tien (2018). *CIFAR10 to Compare Visual Recognition Performance between Deep Neural Networks and Humans*. Vietnam: The University of Danang – University of Science and Technology.

Appendix A. Quantitative Results

1. Baseline Model Results

Test Performance (class conversion):

Test performance	Precision	Recall	Accuracy
ResNet50	0.60	0.54	0.54
InceptionV3	0.55	0.51	0.51
VGG16	0.60	0.55	0.55

Table 1.1 Test Performance (class conversion)

2. ResNet50

Bayesian Optimization:

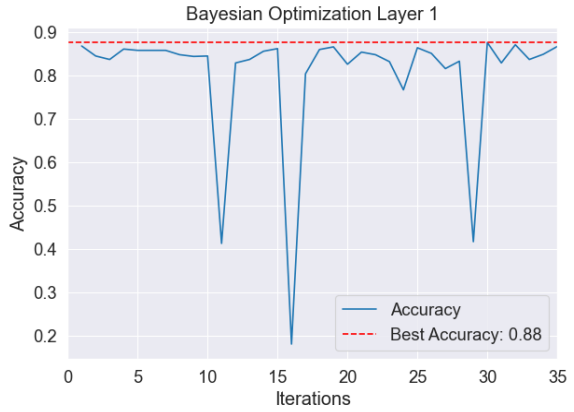


Figure 1.1 - ResNet50 Bayesian Optimization with 1 Layers.

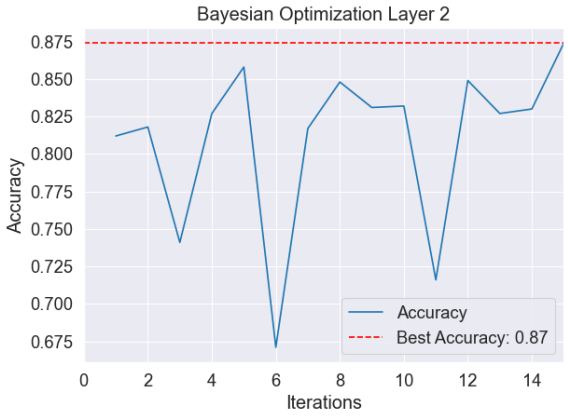


Figure 1.2 - ResNet50 Bayesian Optimization with 2 layers.

Layers added	Units	Drop-out rate	Learning rate	Batch size	Loss func.
1	178	0.0	0.001	64	cross-entropy
2	55 + 27	0.0	0.001	64	cross-entropy

Table 1.3 - ResNet50 Best parameters from Bayesian optimization

Train Performance (Fine-Tuning):

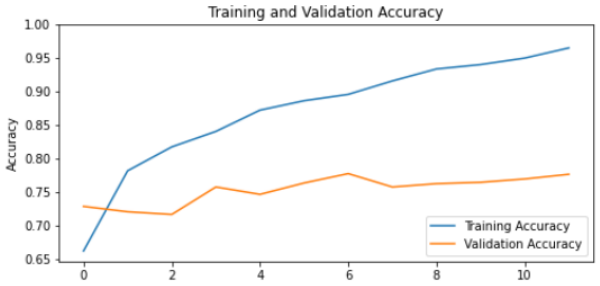


Figure 1.4 - ResNet50 training process of the extra layer (Stage 1).

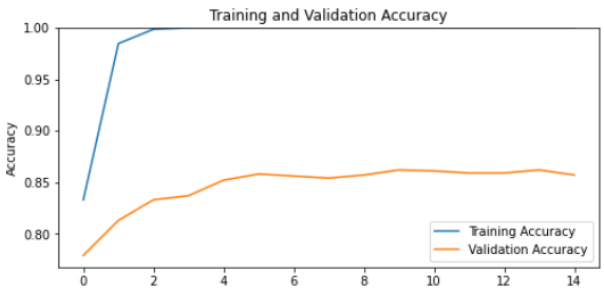


Figure 1.5 - ResNet50 training process of the whole mode (Stage 2).

Test Performance:

Test performance	Precision	Recall	Accuracy
Adaptation phase	0.79	0.77	0.77
Fine-tuning phase	0.86	0.86	0.86

Table 1.6 - ResNet50 test performance.

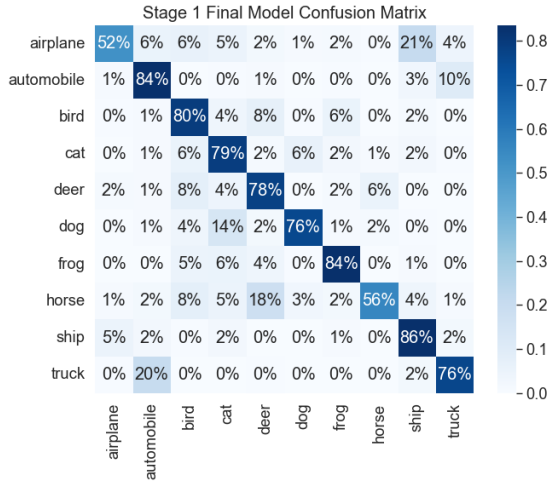


Figure 1.7 - ResNet50 confusion matrix for stage 1.

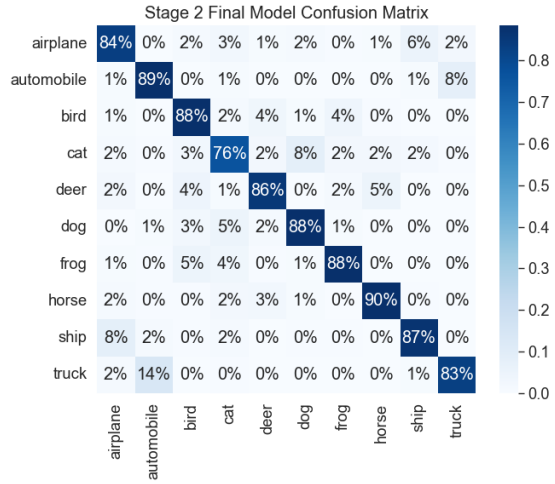


Figure 1.8 - ResNet50 confusion matrix for stage 2.

3. InceptionV3

Bayesian Optimization:

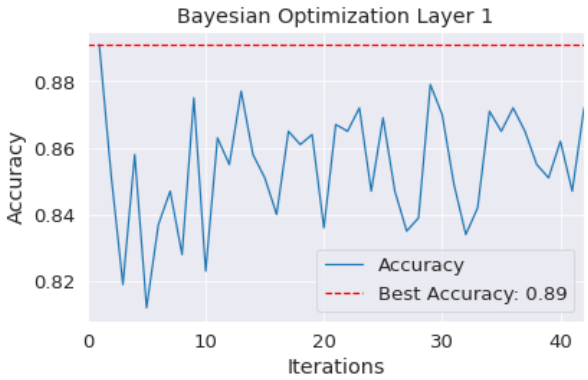


Figure 2.1 - InceptionV3 Bayesian Optimization with 1 layer.

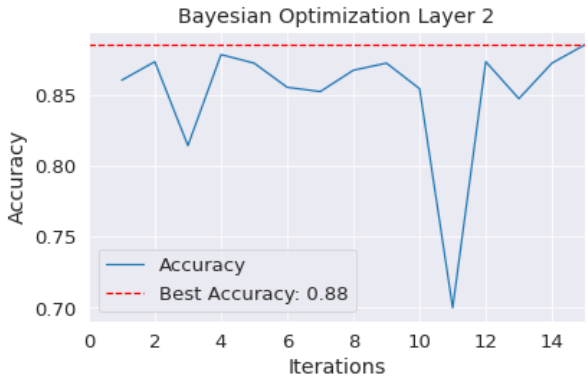


Figure 2.2 - InceptionV3 Bayesian Optimization with 2 layers.

Layers added	Units	Drop-out rate	Learning rate	Batch size	Loss func.
1	93	0.10	5.6*10-4	16	Cross-entropy
2	512 + 256	0.6	5.6*10-4	16	Cross-entropy

Table 2.3 - InceptionV3 Best parameters from Bayesian optimization

Train Performance (Fine-Tuning):

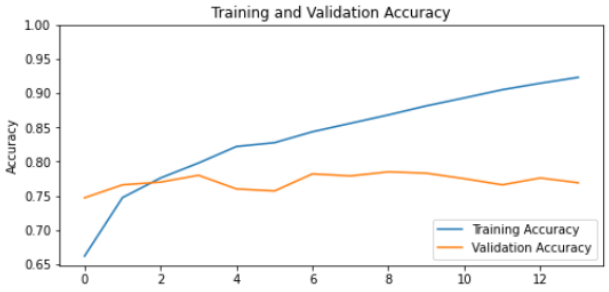


Figure 2.4 - InceptionV3 training process of the extra layer (Stage 1).

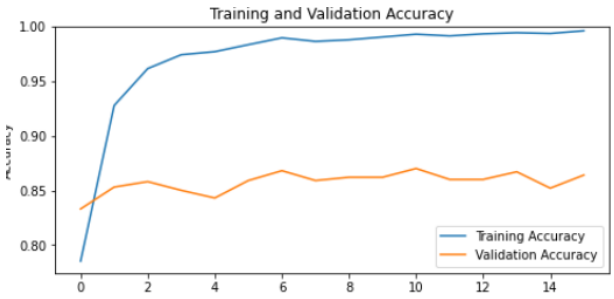


Figure 2.5 - InceptionV3 training process of the whole model (Stage 2).

Test Performance:

Test performance	Precision	Recall	Accuracy
Adaptation phase	0.76	0.75	0.75
Fine-tuning phase	0.85	0.85	0.85

Table 2.6 - InceptionV3 test performance

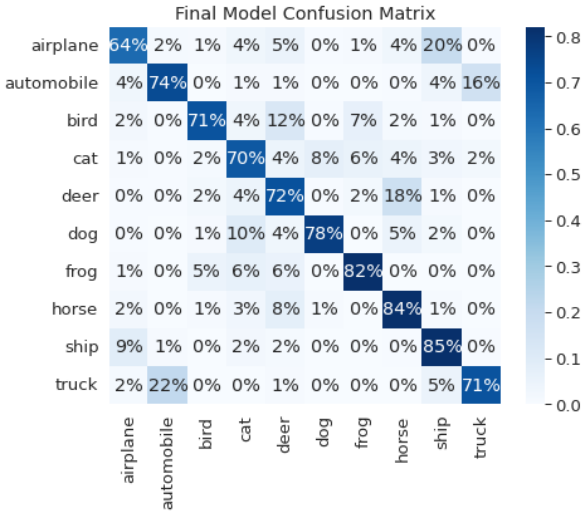


Figure 2.7 - InceptionV3 confusion matrix for stage 1.

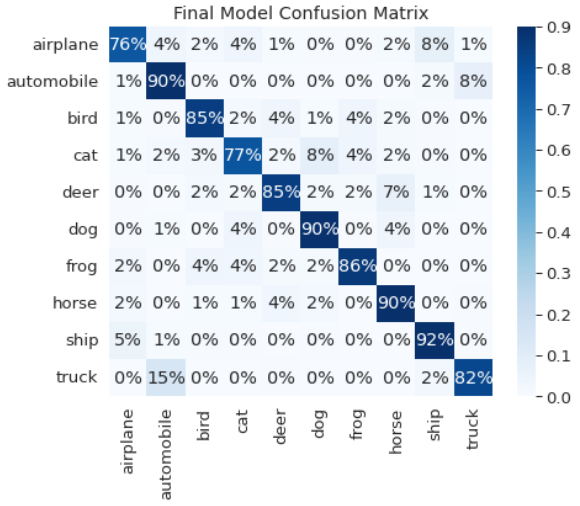
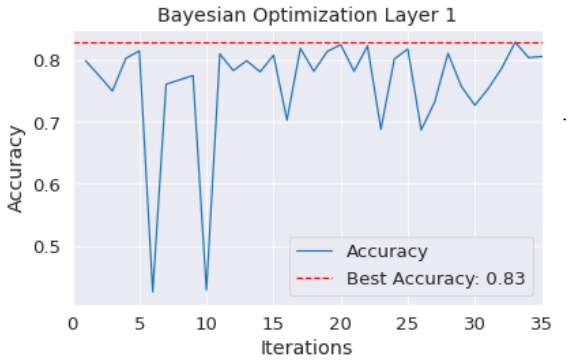


Figure 2.8 - InceptionV3 confusion matrix for stage 2.

4. VGG16

Bayesian Optimization:



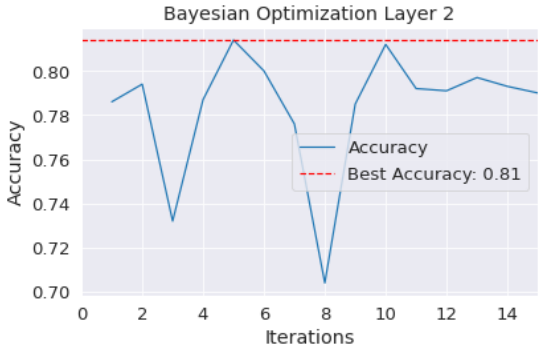


Figure 3.1 - VGG16 Bayesian Optimization with 1 layer.

Figure 3.2 - VGG16 Bayesian Optimization with 2 layers.

Layers added	Units	Drop-out rate	Learning rate	Batch size	Loss func.
1	312	0.6	$7.25*10^{-4}$	64	Cross-entropy
2	512 + 256	0.0	$7.25*10^{-4}$	64	Cross-entropy

Table 3.3 - VGG16 best combination of parameters.

Train Performance (Fine-Tuning):

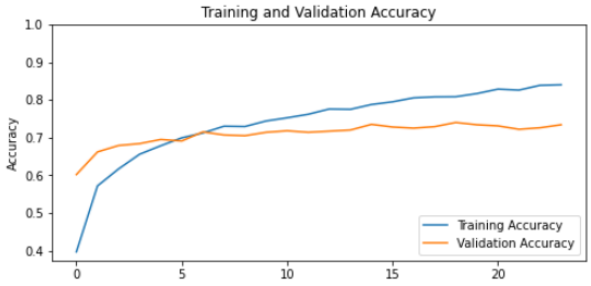


Figure 3.4 - VGG16 training process of the extra layer (Stage 1).

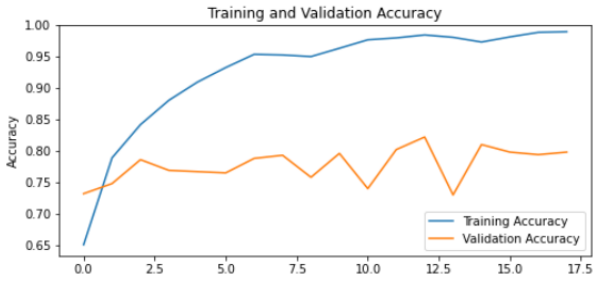


Figure 3.5 - VGG16 training process of the whole model (Stage 2).

Test Performance:

Test performance	Precision	Recall	Accuracy
Adaptation phase	0.75	0.75	0.75
Fine-tuning phase	0.82	0.81	0.81

Table 3.6 - VGG16 test performance

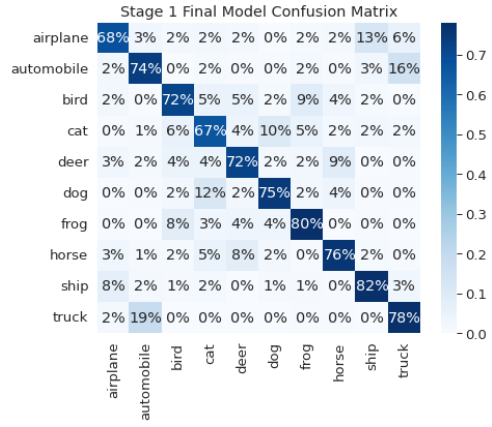


Figure 3.7 - VGG16 confusion matrix for stage 1.

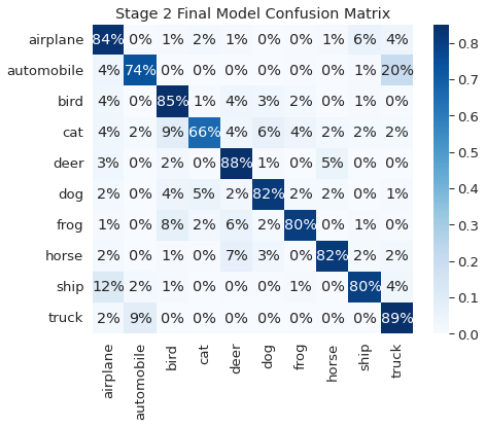
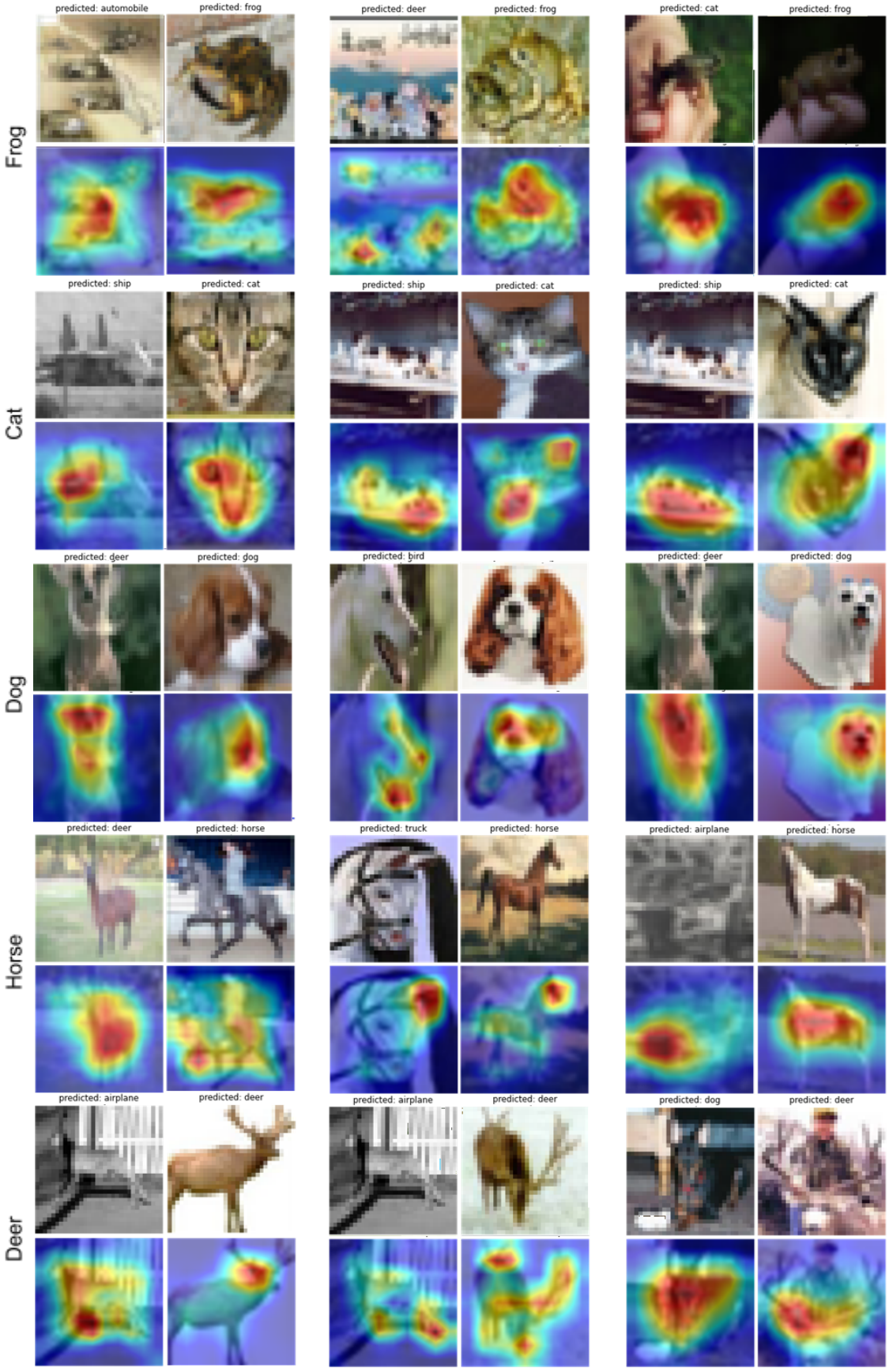


Figure 3.8 - VGG16 confusion matrix for stage 2.

Appendix B. Qualitative Results

Resnet 50, VGG16, and InceptionV3 with the predictions of the lowest and highest score, respectively.



Resnet 50, VGG16, and InceptionV3 with the predictions of the lowest and highest score, respectively.