UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics & Computer Science

Image based Bee Health Classification

Shruti Chawane
M.Sc. Thesis

Industrial Supervisor:
Kiran Jayaraj
Cheif Technology Officer
InsectSense

Supervisor:
prof. Dr. D.V. Le Viet Duc
prof. Dr. Paul Havinga
prof. Dr. E.T. Martinez

Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

Honeybees are very important to the ecosystem and mankind as they play a vital role in the pollination process of the agricultural process and providing a balance to the bio-diversity of ecosystem. The state of the environment can also be deduced by keeping a track of bee health status. This leads to the need of a bee health classification system which can ubiquitously identify bee health status. Recent decades have seen research and development in the area of artificial intelligence to perform classification tasks from input images via a camera. Therefore, an image dataset with over five thousand images is taken into consideration for this project. This work presents the recent advances in the area in the last two to three decades, experiment with the state of the art models namely VGG-16 and DenseNet-121. The development methodology is inspired by the technique of transfer learning with pre-trained VGG-16 and DenseNet-121 on the ImageNet dataset. Initially, the reasons why the task of image based bee health classification differs from the pre-training of the deep models on ImageNet are mentioned. The reasons are also verified by the results of experiment 1. The results of experiment 1 provide a very clear picture as to what kind of features are extracted by both the models from images from ImageNet and the BeeImage dataset. The pre-trained models are observed to successfully extract fine and smooth features from underlying images that includes edges, texture, shape, surroundings of the objects. But the down stream task demands even finer features to detect bee health like orientation of bee, wing structure and quality, bodily deformities etc. In extension, the work establishes evaluation metrics for the task of image based bee health classification and critically analyses the results of model performances with evaluation metric of F1 scores and macro-F1 scores. The project also delivers on the best fine-tuning strategy for the image base bee health classification. In the end, it is found that DenseNet-121 based models outperform VGG-16 based models. Also, it is observed that both models have three 'feature-extracting blocks' along with two 'interpreting blocks'. It is deduced and concluded that the 'interpreting blocks' do indeed need fine-tuning to perform better than the "out of the box" pre-trained deep models. In the end, the research question derived with the project is answered conclusively.

Acknowledgement

First, I would like to thank Dr. Le Viet Duc of University of Twente, Netherlands not only for his supervision during this thesis, but also the preparatory Research Topics course. His feedback and critical analysis of my work has proven to be a vital part for completion of this project. He has helped and supervised me to build research questions that could be aligned with the particular use case. Secondly, I'm grateful to my external supervisor, Kiran Jayaraj of InsectSense who maintained his support during the project and provided with guidance necessary to carry out this project.

Also, I would like to acknowledge the support provided by the committee members Dr. Paul Havinga and Dr. Estefanía Martínez of University of Twente, Netherlands.

Last, I am very thankful for the support and encouragement received by my family throughout my studies, as well as my friends, on whom I could always count for necessary comfort while working on my thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Beekeeping is an ancient tradition and it has been carried out for several millennia. Honeybees are very important to mankind [1]. For starters, while managing the growing population and hence growing food requirements, the contribution of bees for providing high quality food (honey, royal jelly and pollen) and various products (beeswax, propolis, honey bee venom) cannot be ignored. Second, honeybees play a major role in the pollination process of agricultural crops. Farmers have evolved to grow more crops that are dependant on pollination for growth (i.e. fruit, vegetables, seeds, nuts and oil seeds). Finally, honeybees play a major role in providing a balance to the bio-diversity of ecosystem. By providing pollination, which is an enabler of food production, they help in maintaining and protecting animal and plant species. Honeybees are also used to evaluate the state of the environment. Their existence, non-existence and quantity represent a specific state of the environment and indicate towards necessary actions. If the development and health of honeybees is observed, in quite some cases the state of the environment can be deduced and lead to necessary action in due time.

InsectSense's BeeSense is a project that aims to build a system which can inculcate the method of training and conditioning the honeybees to detect volatile organic compounds (VOCs). Honeybees are considered to have extremely sensitive olfactory receptors that can sense the VOCs in the range of parts per billion [2]. Using specially designed hardware and image recognition software, BeeSense can be used to train the honeybees to detect VOCs or diagnose diseases such as Covid-19 [3]. The project BeeSense has an additional functionality to provide the beekeeper with a health status of the honeybees in training. Keeping track of health condition during recurring training of honeybees through BeeSense will help the beekeeper to deduce beehive conditions and take necessary precautions in time. To provide the beekeeper with the health status, images of bees taken by a camera in BeeSense hardware module would be used. The goal of this project is to research and build the best possible artificially intelligent system to categorize honeybee health status from it's images.

To hold experiments and conclude results the Bee Annotated Dataset [4] is used which contains more than 5,000 thousand bee images labelled for their health status. The dataset is publicly available at Kaggle. A detailed analysis of the dataset and it's components is discussed in Chapter 3. The technique of transfer learning [5], [6] will be used to train models during the course of experiments.

1.1 Project Goals

The goal of this project is to research and build the best possible model for the task of classifying bee health status from it's image. Most of the current prevailing work on the task of image classification uses the technique of deep convolutional networks [7], [8] and residual training techniques [6], [9]. However, Zhu et. al. [10], discuss that such complex and deep models need training data in high amounts to train themselves from scratch. Hence, the scope of this project includes deep pre-trained networks (with and without the architecture of residual training) to fine-tune with transfer learning [5] on the bee annotated dataset [4]. Furthermore, the project compares the performance of fine-tuned DenseNet-121 [9], and VGG-16 [8] on the task of image based bee health classification.

1.2 Motivation and Research Questions

The pre-trained deep models are pre-trained on the task of object detection from the ImageNet dataset.The models need to be down-streamed or fine-tuned for the task of image based bee health classification. Hence, a comparison would be needed to identify similarities or differences in the pre-training and down-stream task. This comparison would be critical to decide and guide, as to what level the pre-trained deep models need to be fine-tuned. Also, it should be noted that the complete deep model need not to be fine-tuned for best classification performance. This stems from the fact that the deep pre-trained models are pre-trained over millions of images giving the models the ability to extract fine low-level features from input images. Also, it would be important to observe what features are the deep pre-trained models able to extract from images from the BeeImage dataset. This will give a very close look on what features do the pre-trained models focus on and consider important for the task of image classification. Moreover, it will also visualise what features can be extracted for the task of image based bee health classification from honey bee images.

The second part of the project focuses more on the training and evaluation strategy of the deep pre-trained models. It is necessary to evaluate the fine-tuned models using an evaluation metric that is relevant to the project goal of helping the bee keeper while at the same time is not skewed because of the distribution the evaluation test split. Furthermore, there are different strategies to fine-tune the deep pre-trained models that are discussed in further chapters. Hence, the best training strategy has to be identified to fine-tune the deep pre-trained models. In effect, the best training environment has to be concluded. Something to note is that the goal is not to deliver the best possible model, but the best training environment that is possible for the task of image based bee health classification. A better performing model can be found alternatively to this project with hyper-parameter tuning and possibly a bigger train set. However, the goal is to identify the best training environment and strategy for deep pre-trained models namely VGG-16 and DenseNet-121 on the task of image based bee health classification. The fine-tuned models could be then evaluated using the decided evaluation metric so the best training environment is concluded. Hence, the broad research question is summed by RQ as to what level can transfer learning support the task of image based bee health classification. The RQ is then divided into sub questions in alignment with the motivation presented above. To complete the goals of this project the following research questions arise:

RQ) How does transfer learning with available pre-trained CNN models perform on the task of image-based bee health classification?
Sub RQ1) What are the requirements which make image-based bee health classification different from the classification problems with ImageNet?
Sub RQ2) What image-based features can be inherited from pre-trained models with ImageNet for the bee health classification?
Sub RQ3) Which evaluation metric would be effective to evaluate the performance of artificially intelligent systems for classifying bee health status from their images?
Sub RQ4) What is the most suitable transfer learning scheme for image-based bee health classification, given pre-trained CNN models with ImageNet?

1.3 Technical and Scientific Contributions

This work aims to experiment with prevalent transfer learning techniques to build on the application of image based bee health detection in the domain of pervasive computing. The project goals defined above make it clear that the project will deliver on a classification system fine-tuned using transfer learning while experimenting to find the best possible strategy to fine-tune the deep pre-trained models. Chapter 2 discusses the history and evolution of artificially intelligent systems designed for classifying input images. The latest developments mention the use of transfer learning technique with deep pre-trained models to develop state of the art models. The research defines research questions around its motivation as discussed in the previous section. The results of designed experiments is used to conclude answers to those questions.

The broad research question would quantify and enlist the results of transfer learning techniques on the task of image based bee health classification. This is a novel approach of detecting bee health statuses using deep pre-trained models. Important thing to note is the deep models are pre-trained not on bee images rather on images of all possible objects in environment (and present in the ImageNet dataset). It is interesting to evaluate the performance of a model trained to detect objects but then fine-tuned to detect health statuses of honey bees. The project also conducts a detailed comparison between the based dataset used for pre-training the deep models and the dataset used for fine-tuning for the downstream task. Another comparison between both tasks helps to deduce that both tasks indeed differ on fundamental features needed to classify for target classes however the models successfully learn the downstream task on the chosen dataset as discussed in chapter 6. Moreover, the dataset chosen for the project is visibly imbalanced towards some of the classes. This is also discussed in Chapter 3 in the detailed analysis of the dataset. Hence, the experiments also give a view on the performance of deep pre-trained models on an imbalanced dataset. Using some pre-processing and data augmentation techniques discussed also in Chapter 3, the deep pre-trained models successfully manage class imbalance in the dataset for the downstream task as shown by the results in chapter 5. It is imperative to note that deep models which are trained on object detection successfully manage class imbalance in the dataset of the downstream task which require completely different base features for classification. This is something which needs to be acknowledged that deep pre-trained models are indeed sensitive to a lower representation in the downstream tasks. The experiments and evaluation metrics are designed to conduct and measure performances in the discussed scope to create contributions as discussed in this chapter.

Chapter 2

Background and Related Works

This chapter is divided into two sections.The first sections enlists and discusses background and related works in the domain of honey bee health monitoring and the other section enlists and describes the evolution of machine learning and deep learning techniques on the task of image classification. This particular domain is chosen since this research aims to hold experiments and comparisons of artificially intelligent systems on an image dataset.

2.1 Honey bee health

Beekeeping has been around humanity for about a millenia. The western honey-bees, *Apis mellifera* are an important part of the ecosystem and help in sustaining biodiversity by providing essential pollination for a wider range of crops and plant [1]. Over the past decade or so, many beekeepers have reported the unusual decline of bee colonies and numbers in several countries across the globe [11], [12]. There are various causes leading to bee loss which include attacks by pathogens and invasive species – such as the Varroa mite *(Varroa destructor)* [13], the Asian hornet *(Vespa velutina)* [14], and the small hive beetle *(Aethina tumida)* [15] and environmental changes (e.g. habitat fragmentation and loss) [16] . In order to maintain a healthy bee stock, it is important to monitor and maintain it globally.

For the past century, research has been going on the various diseases infesting upon the honey bees. Varroa is one of the eminent parasitic mites that has been a problem for the beekeepers for a long time. The type species of the Varroa was described early in the twentieth century from a colony eastern honey bees [17]. The common honey bees are very vulnerable to the Varroa mites. Since the mites feed on the brood, they can weaken and kill the colonies [18] [13] [19]. Also, the Varroa mites cause transmission of viruses like deformed wing virus(DWV), sacbrood virus, acute paralysis complex to honey bees which results in deadly diseases [20] [21].

Another invasive pest for bee hives is the small hive beetle *Aethina tumida* (SHB) [22]. The small hive beetle is a destructive insect that can damage honey bee colonies. If it gets too heavy, it can cause the bees to abandon their nests. Just like small hive beetles and Varrao mites, ants are also a problem for the honey bee colonies. The ants feed on the honey bees brood and disseminate viruses to honey bees by invading hives and transmitting the viruses to hive cells [23]. The other factors that possess threat to the honey bees are attacks of the robber bees [24], the queen missing in the hive [25].

The problems mentioned above make bee health monitoring important so that countermeasures can be taken promptly. There are several traditional methods that have been carried out for bee monitoring. One of the common methods to detect the Varroa mites is a roll test with powdered sugar or roasted soybean flour [26]. Another traditional method to detect the hive beetles and ants is visual observation. However, these methods are time consuming and not very efficient. Hence, the use of image processing, computer vision and machine learning techniques can be proved much efficient and faster than the traditional methods [27] [28] [29].

2.2 Machine learning systems

This section enlists and discusses evolution of machine learning and deep learning techniques on the task of image classification. The project BeeSense includes a camera to monitor bees during their training process. The images captured from this camera would be then used to categorize health of each bee from a set of predefined classes. This project aims to build an artificially intelligent solution to enable this functionality. The bee monitoring techniques that prevail as the market standard [30] currently involve the use of IoT technologies for real time bee monitoring. Also, recent products and research involve the use of machine learning and deep learning techniques [31] to analyse bee and beehive characteristics.

This project aims to build a solution that categorizes a bee's health from it's image. The techniques of building artificially intelligent systems for the task of image classification have evolved quite a lot during the recent years [32]. From the late 90s neural networks [33] (feed forward networks) were used for the task of image classification. A point to note here is that a simple multi layer perceptron taking input from each pixel of image would not have been a feasible solution. The reason behind it is the inefficiency to process this large size of input (equal to number of pixels in the image) even though the adjacent pixels are spatially correlated. Hence, there was a need to first extract meaningful and low dimensional features of an image to provide to a neural network to learn on. In 1998, Lecun et. al. [34] proposed the LeNet architecture (shown in figure2.1) of training an artificially intelligent system. To tackle the problem (feasibility) mentioned above, the solution proceeded in multiple steps. First, the input images for training are passed through multiple convolutional and activation layers of fixed size to generate and low level meaningful features from images. A neural network is then trained on these features as input using back-propagation to classify images. Hence, ConvNets (LeNet) were successful in extracting low level features from images and training a neural network on these features.

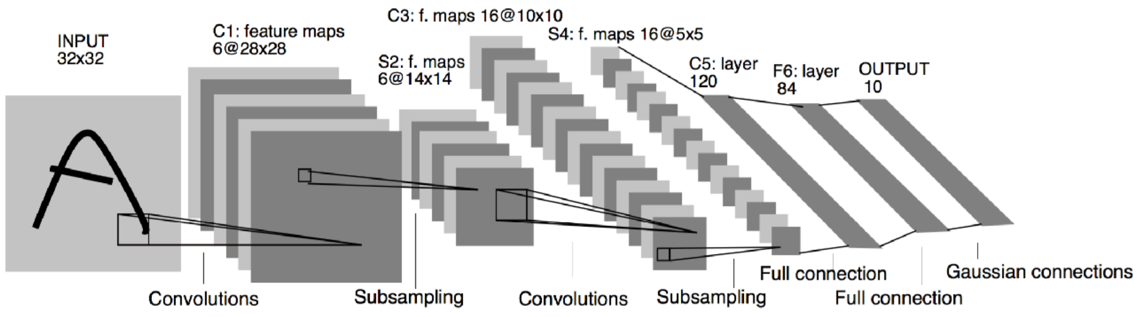


Figure 2.1: LeNet Architecture

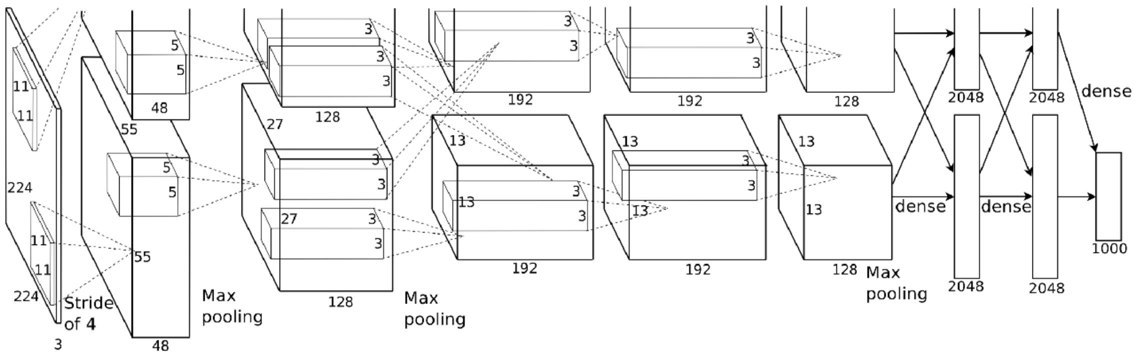
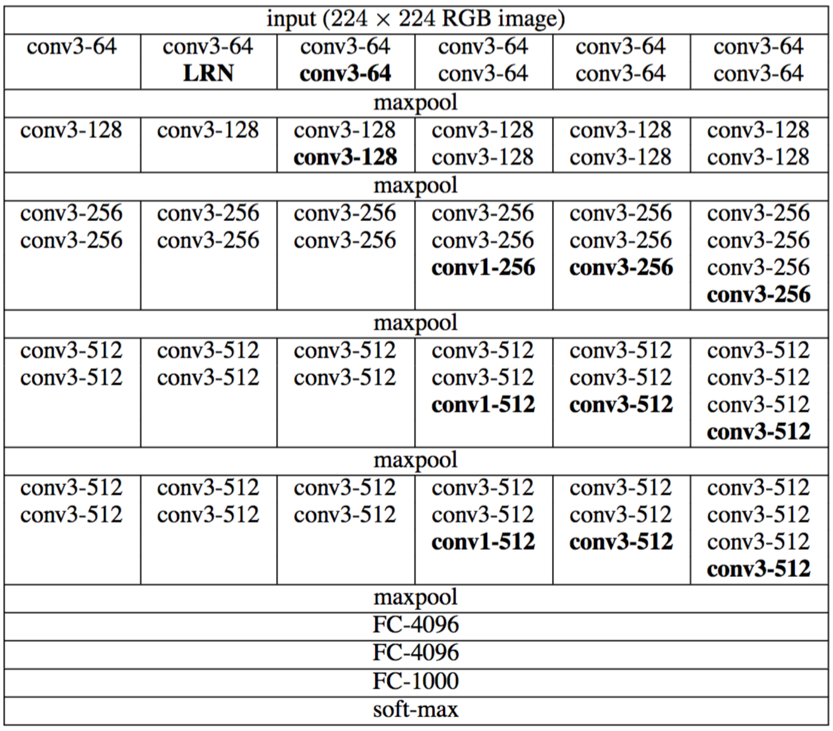


Figure 2.2: AlexNet Architecture

Until 2012, ConvNets or CNNs were not able to make their claim to fame. The reason, as explained by Krizhevsky et. al. [7] in 2012, that the complete potential of CNNs was not being leveraged during training of the model. Krizhevsky proposed several changes to a general LeNet [34] architecture. First of all, better linearity was introduced in the LeNet architure using the ReLu activation. This helped in gradient propagation back through the model. Second, dropout as regularization was introduced. To make representation easier, it can be thought that the model forgets things at random so it can perceive the next input in a better way. Third, the approach introduced data augmentation during training by randomly rotating, translating and cropping of images in the input during training. This helped the system to be able to learn to extract features from images that were not in perfect alignment of the images in the training set. Hence, making the model more robust in terms of performance towards new data. Finally, the most important contribution that AlexNet [7] made, is that it went "deeper". As can be seen in figure2.2, there are more convolutional layers stacked before pooling operations. This equipped the system to extract finer features from the image which in turn led to better classification. AlexNet [7] outperformed the state of the art in 2012 on the ImageNet dataset [35].

In 2015, came the next big milestone in the field of image classification. Proposed by Simonyan et. al. [8], the VGGNet outperformed the state of the art systems on the Imagenet dataset [35]. The architecture mainly focused on "going deeper". As can be visualized in figure 2.3 there are significantly more convolutional operations before pooling when compared to AlexNet [7]. Another important thing to note, is the size of the convolutional layers is mostly set to a 3X3 matrix. Implying that the model has more number of convolutional layers but of small sizes. The authors provide three major motivations for building such an architecture. First of all, smaller filters lead to further more non-linearity which in turn makes more degrees of freedom for the model. Second, stacking more of such filters increases the reception of the model. Implying, if two such filters are stacked next to each other the systems then gains a 5X5 pixel reception; if three, a 7X7 reception and so on. Hence, the proposed system [8] also includes findings from AlexNet [7]. Finally, using small filters limits the number of parameters of the system which is good since the model, as the name suggests goes "very deep".



input (224 × 224 RGB image)					
conv3-64	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64
	LRN	**conv3-64**	conv3-64	conv3-64	conv3-64
maxpool					
conv3-128	conv3-128	conv3-128	conv3-128	conv3-128	conv3-128
		conv3-128	conv3-128	conv3-128	conv3-128
maxpool					
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
			conv1-256	**conv3-256**	conv3-256
					conv3-256
maxpool					
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
			conv1-512	**conv3-512**	conv3-512
					conv3-512
maxpool					
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
			conv1-512	**conv3-512**	conv3-512
					conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 2.3: VGGNet Architecture

The VGGNet [8] and AlexNet [7] both had the idea of building deep networks to extract finer features for better image classification. However, researchers realised that stacking more layers on top of each other doesn't always lead to a better classification performance. The reason behind this are vanishing gradients and the layers are not optimized during training. Hence, researchers came up with the technique to counter this effect named residual learning [6]. As shown in figure 2.4 there is an identity mapping after every other layer. This mapping is proposed to exist after every other layer throughout the network. This connection creates an additional path for the network to back-propagate error hence solving the problem of vanishing gradients. Another advantage of having the identity mapping connection is the later layer perceives input as it's previous layer along with it's own perception. This acts as an added advantage during training to form better understanding of the image to the model. ResNet [6] until very recently was the best performing model on image classification task on the ImageNet dataset [35]. Another major milestone architecturally was developed on the same logic. In 2018, Huang et. al. [9], proposed the DenseNet architecture which had an identity mapping like that of ResNet [6] amongst all the layers of the network as can be seen in figure 2.5.

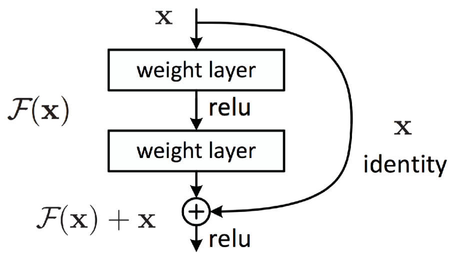

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 2.4: Residual Learning

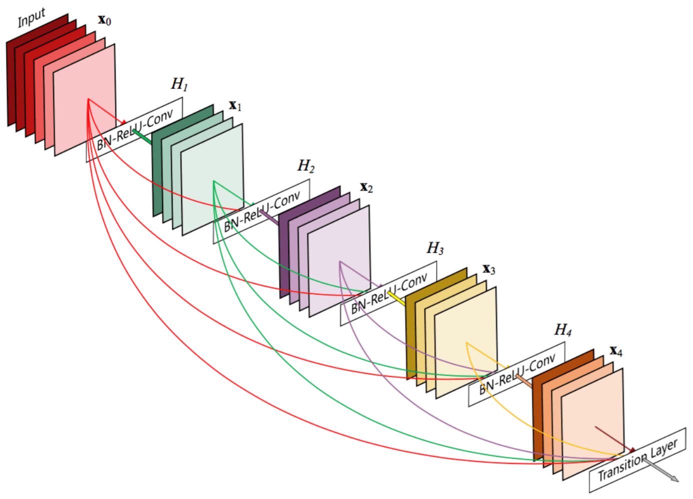


Figure 2.5: DenseNet Architecture

Hence, from these works the two major points to note are, first, going deeper increases the classification performance of a model. However, just stacking up convolutional layers only helps to train the system to some extent. If the number of layers are significantly high, the gradients start to vanish and the model doesn't train to the full extent. Second, to minimize/solve the problem of vanishing gradients residual learning technique is used, creating an additional channel to back-propagate gradients. Hence an ideal approach to an image classification task would be to develop a deep convolutional system trained with residual learning. However, training such deep and complex models on a small dataset is not the right approach. Zhu et. al. [10] discuss that high amount of training data is needed to ideally train more and more complex models. Hence training a complex model like a deep ResNet [6] from scratch on a small dataset would not lead to the best possible model. Weiss et. al. [5] discuss the evolution of an old technique to solve this problem. The solution, transfer learning. Different complex networks that discussed in the prevailing sections have been pre-trained on different big datasets like ImageNet [35]. An instance of these pre-trained models are open source and used along with transfer learning to build artificially intelligent systems. A general architectural representation of this approach can be seen in figure 2.6. However, while using transfer learning with pre-trained systems leads to a question on the training methodology. The question becomes, which pre-trained model is chosen and of what size?

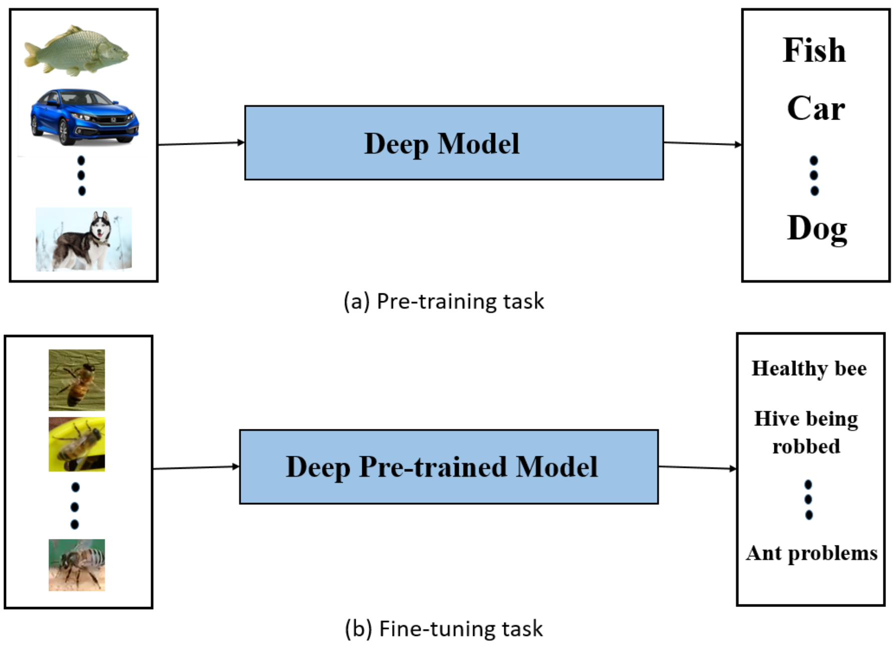


Figure 2.6: Transfer Learning

As the rising amount of data and the technology to compute complex operations increases, machines have known outperform humans on several image classification tasks [36]. In history, when Krizhevsky et al. [37] proposed their deep convolutional approach for the task of image classification on the ImageNet dataset, they set the standard for further experiments [38]. Deep convolutional networks were widely accepted and preferred for the task of image classification [39], [40], [41]. However, training deep convolutional networks from scratch takes a lot of time, resources and data. As a resolve, it is prevalent to initialize a pre-trained deep model and then downstream it for a smaller dataset for a particular task. Some of the state of the art systems for the task of classification with transfer learning include Rahul et. al.'s approach [42] where a ReseNet architecture is fine-tuned to detect audio spoofs. Another benchmark of transfer learning based approiach is the one proposed by Akash [43]. Another example of a state of the art classifier on natural plant images to detect deseases is the one presented by Anjaneya [44].

Chapter 3

Dataset

This section describes the dataset which is being used for the experimental setup. The dataset is taken from Kaggle [4] containing images which were extracted from still time-lapse videos of bees. The frames were averaged to calculate a background image and each frame of the video was subtracted against that background to bring out the bees in the forefront. The bees were then cropped out of the frame so that each image has only one bee.

There are total 5172 images in the dataset with 9 columns namely 'file', 'date', 'time', 'location', 'zipcode', 'subspecies', 'health', 'pollen_carrying' and 'caste'. The target class among these columns will be 'health'. There are 6 unique values of the 'health' class. These 6 values are 'Varroa, Small Hive Beetles', 'ant problems', 'few varrao, hive beetles', 'healthy', 'hive being robbed' and 'missing queen'. The distribution of the target class is represented in table 3.1 and shown in figure 3.1.

Class name	**Count**	**Relative count**
Varroa, Small Hive Beetles	472	0.09
ant problems	457	0.09
few varrao, hive beetles	579	0.11
healthy	3384	0.65
hive being robbed	251	0.04
missing queen	29	0.01

Table 3.1: Distribution of target class

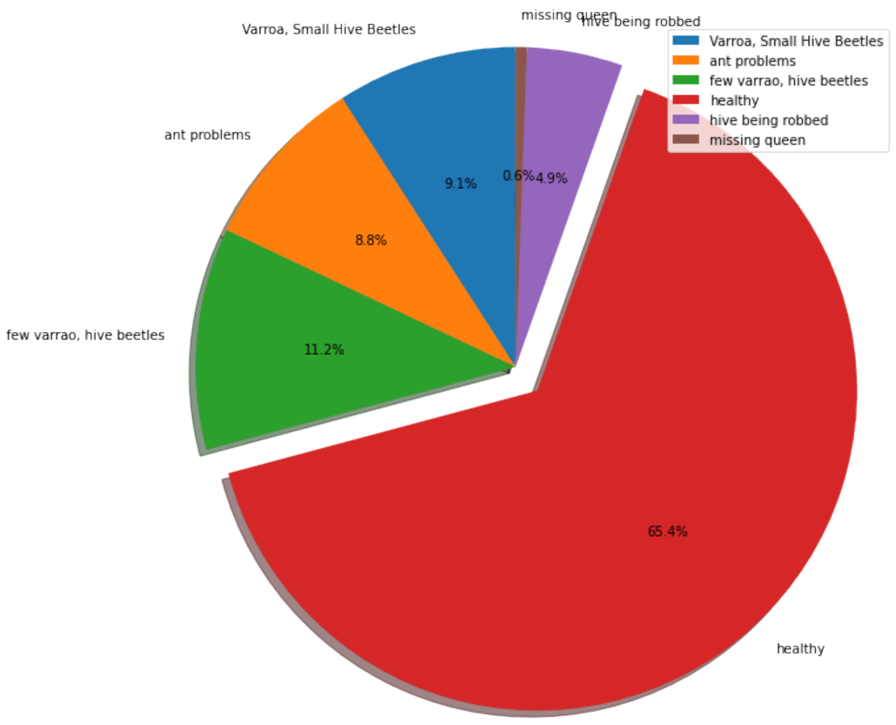


Figure 3.1: Distribution of target class

As the dataset is not evenly distributed across classes and the target class 'missing queen' has only 29 images in the dataset, the data for this class is not taken into consideration. This is done so the model doesn't receive train and evaluation data with little to no representation in the training set, which would lead to deviation from the model's optimal learning capabilities.

3.1 Data Pre-processing

As it can be seen from 3.1, the number of images for each class are not balanced. This would create a problem for the model training since the model will tend to learn more about the class with more samples in the training set. Not giving the model a chance to generalize itself. However, the deep pre-trained models are considered to be sensitive towards smaller sample set as well. Moreover, during training the input images are randomly flipped horizontally and vertically and also rotated by random angles. This is done to tackle the class imbalance and provide the model with more general view of bee images. Another measure taken is that during the experiments all models are fed images in a batch size large enough to include some images from all classes most of the times. Also to tackle class imbalance a stratified train-test split has been done. The dataset is split into train and test sets in a way that preserves the same proportions of examples in each class as observed in the original dataset.

The train set forms 80% of the total dataset and the test set takes 20% of the total dataset.

The train set has 4114 total images. The distribution of classes in the train set is shown in the table 3.2 and figure 3.2. Similarly, the distribution of classes in the test set with 1029 total images is shown in the table3.3 and figure3.3. After splitting the dataset, the images are saved in two different directories namely 'Train' and 'Test'. Both the directories contain images inside sub-directories named same as the target classes.

Class name	**Count**	**Relative count**
Varroa, Small Hive Beetles	377	0.09
ant problems	366	0.09
few varrao, hive beetles	463	0.11
healthy	2707	0.65
hive being robbed	201	0.05

Table 3.2: Distribution of train set

Class name	**Count**	**Relative count**
Varroa, Small Hive Beetles	95	0.09
ant problems	91	0.09
few varrao, hive beetles	116	0.11
healthy	677	0.65
hive being robbed	50	0.05

Table 3.3: Distribution of test set

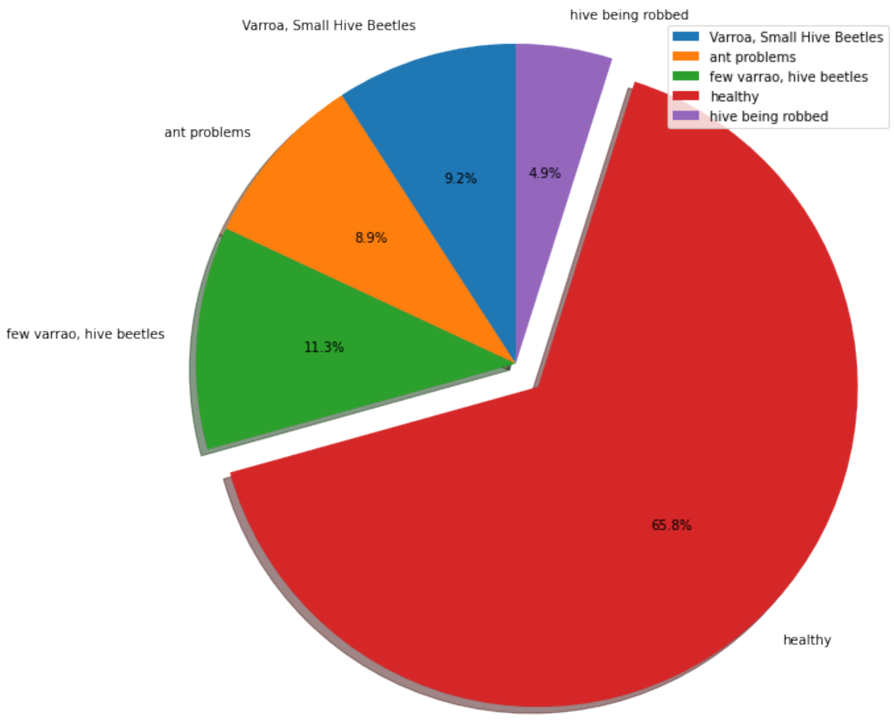


Figure 3.2: Distribution of train set

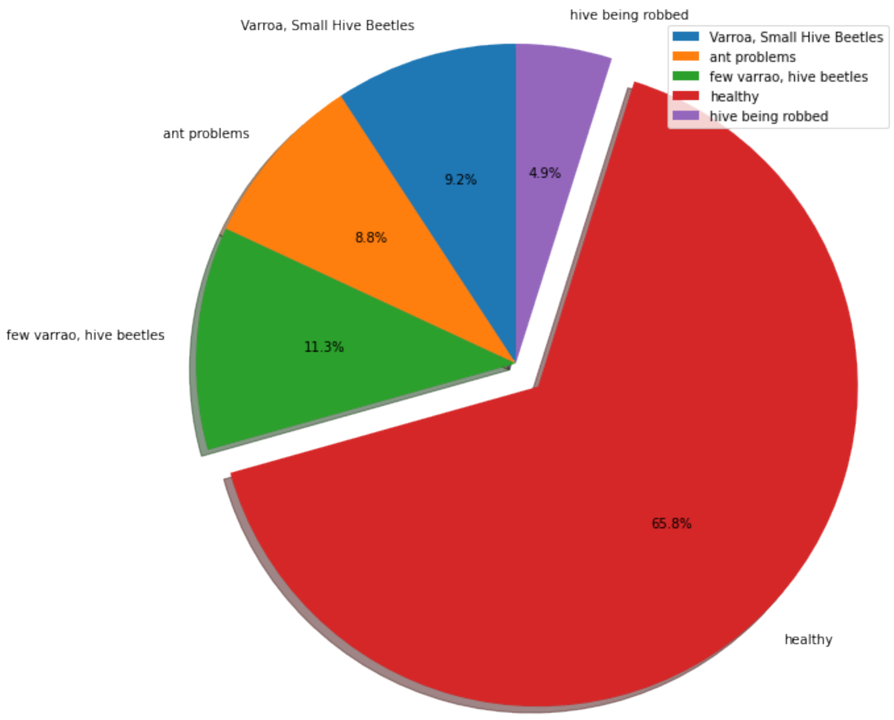


Figure 3.3: Distribution of test set

Chapter 4

Methodology

This chapter outlines the approach that is taken to hold research and conduct experiments to answer the proposed the research questions. The broad question in general is to evaluate how transfer learning with deep CNN models pre-trained on ImageNet dataset perform on the task of image based bee health classification. To conclusively answer this question with reasoning, four sub-questions have branched out.

4.1 Problem Formulation

The task of image based bee health classification is a classification problem with label set as the possible health conditions of a honey bee. The input space of this problem consists of images of individual bees. Hence, in mathematical representation, it is a 5-class classification problem, given an image X predict the class Y from the label set 'Varroa, Small Hive Beetles', 'ant problems','few varrao, hive beetles', 'healthy', 'hive being robbed'.

4.2 ImageNet and bee annotated dataset

As mentioned before, to conclusively answer the research question with reasoning, four sub-questions have branched out. The first sub-question (i.e. Sub RQ1) focuses on the uniqueness of the task of image based bee health classification and how it is different from classification tasks of the ImageNet dataset. The ImageNet dataset is a collection of more than 14 million images. All the images are hand annotated and verified by multiple annotators. The images are annotated by the object presented in the given image. The label set of the ImageNet dataset contains more than 20,000 annotations. The categories or annotations are nouns since they represent the object presented in an image for example 'blueberries', 'horse' etc. These annotations are then mapped to WordNet hierarchy [45] in which each node contains thousands of images labelled to itself. Some of the images from the ImageNet dataset and their corresponding labels are presented in figure 4.1.

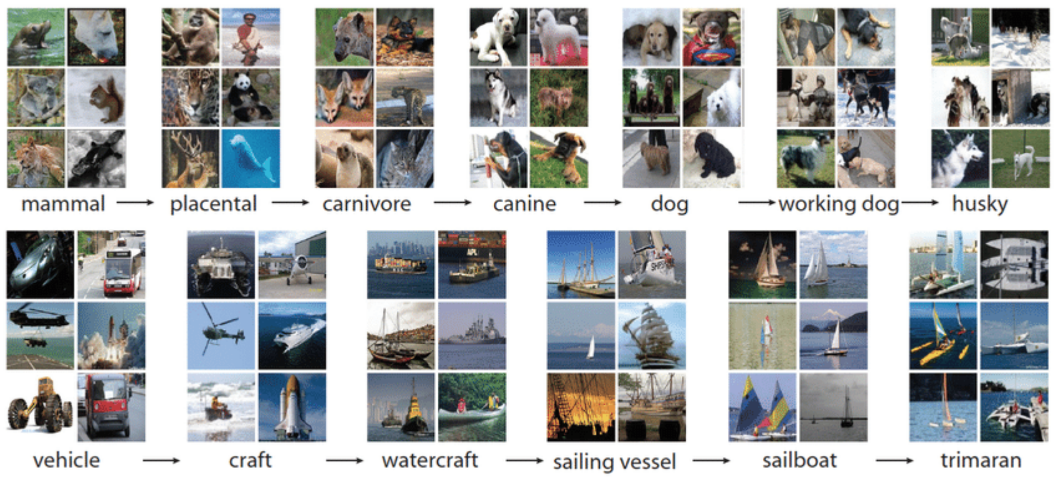


Figure 4.1: ImageNet Example

Since the dataset contains images of objects, classification models trained on this dataset are able to classify an image to a class in the label set i.e. to a noun presented in the image. Intuitively, to classify an image to a noun or detecting objects would need the model to comprehend fine features like edges, shapes and textures. Also, the model might be a bit focused on assessing the surrounding of the object in the image as well as it might support with detecting the object within the surrounding. However, the task of image based bee health classification is even more granular and fine grained than object detection. As can be seen in below figure 4.2 from bee image dataset the features separating healthy bees from others are infinitesimally small and can not easily be read by human eyes. When compared to the task of object detection where a cat has to be distinguished from a dog, this task seems fairly difficult for the system. Moreover, the features that are needed to be extracted for such a task might include orientation of the bee, signs of deformities/injuries on its body, the orientation in which if so a honey bee might be stuck or might be unusual. Intuitively, the features needed to classify bee health conditions might not even be very visible in small cropped images to the machine. Hence, features like the shape of the bee or surroundings of the object (in this case the bee) might not be very relevant to the model. Hence, we can say that the task of image based bee health classification is very different than a classification task on the ImageNet dataset. The features extracted from pre-trained models for images sampled from both data sets are discussed in further sections.

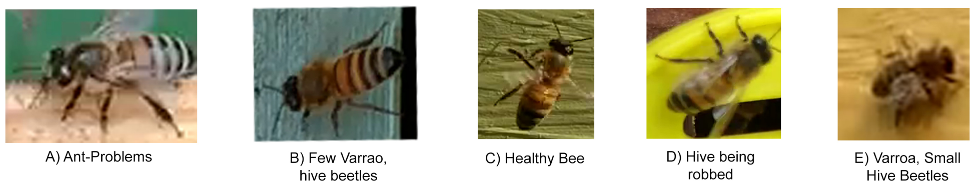


Figure 4.2: Examples from BeeImage Dataset

4.3 Feature extraction

The features extracted for the task of image classification plays an important role for the model to classify the input images. Over the training/fine-tuning process, the model is expected to improve at feature extraction. Implying, the more the model weights are optimised during training/fine-tuning of model, the model is able to extract better features from input image. However, if the size of the training set is not proportional to the size of the model in concern, the training might even result in a model that extracts features that might not be as fine and clear as it was able to extract before training. This is because complex models need higher amount of training data to fine-tune themselves since the gradient of every batch is distributed to a far greater number of weights during training. It is hence necessary to understand what kind of features are extracted by the pre-trained models for the images from the ImageNet Dataset and also the BeeImage dataset before the fine-tuning process.

Moreover, this will also provide an insight as to what kind of features are necessary for the model for images from the ImageNet dataset since the models are pre-trained on the ImageNet dataset. Hence, to visualize the features extracted from the pre-trained models, and experiment, experiment 1 is designed. The experiment is described in detail in the further sections and is designed to visualize and compare the features extracted by the pre-trained models from images of both the ImageNet and BeeImage dataset. Also, the experiment will identify key unique features that different pre-trained systems are able to extract. This will end in a defined set of unique features that different pre-trained models can extract from underlying images.

4.4 Transfer Learning

The next two sub-research questions i.e. sub RQ 3 and sub RQ 4 focus on the fine-tuning and evaluation of the pre-trained models. To conclusively answer sub-RQ4 an experiment, experiment 2 is designed. The pre-trained models used for the experiment are VGG-16 and DenseNet-121. The experiment is designed to find the best strategy to fine-tune the pre-trained models. There can be various strategies to fine-tune complex pre-trained convolution networks as represented in figure 4.3.

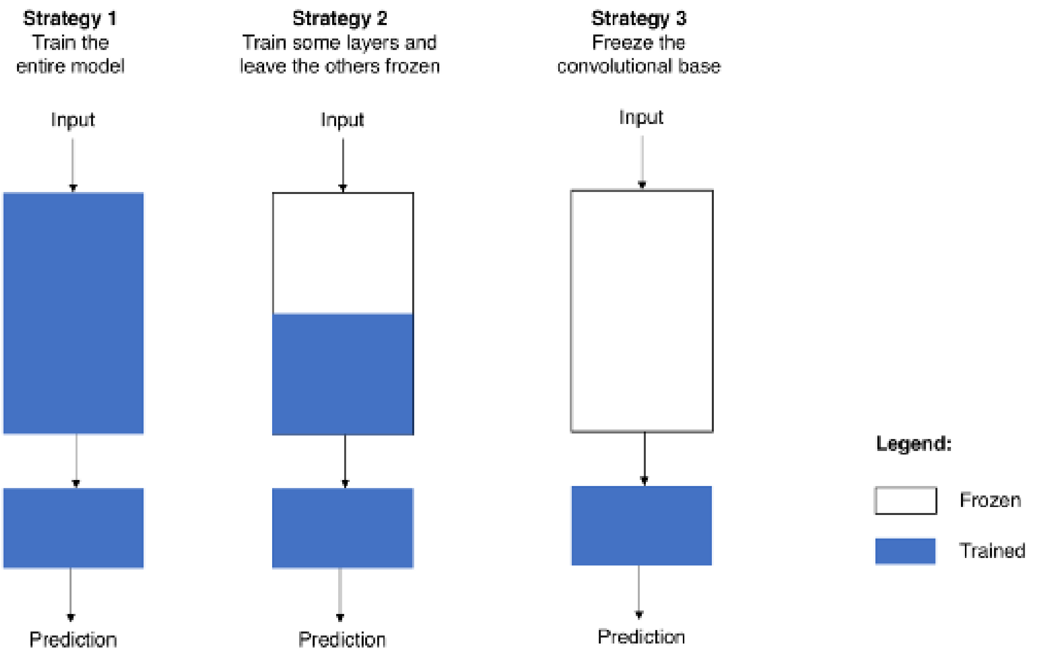


Figure 4.3: Fine-tuning strategies

The first strategy puts forward the whole model for training, implying that all the convolutional blocks along with the classification layers on top of the model are optimized during training. This indeed provides the model the most scope for optimising itself for this specific task. However, the amount of data needed to optimize models of this size might not resemble that of the training set. The second strategy freezes half of the model while training, implying conserving the weights of feature extraction blocks and optimising those of the further blocks. This might be the best strategy to move forward with the given dataset. However, the deep pre-trained models within the scope of this project consist of 5 convolutional blocks leading two possible parts of this strategy. The third strategy suggests to freeze the complete pre-trained model while training and just fine-tune the classification layers on top. Experiment 2 is designed to fine-tune the pre-trained models with the mentioned strategies and compare results. The results from the experiment will conclusively answer sub-RQ4. The research and results of the experiments would conclusively answer the broad research question being to analyse how transfer learning performs on the task of image based bee health classification.

4.5 Experimentation and evaluation setup

As outlined in the previous sections, two experiments are proposed namely experiment 1 and experiment 2 to answer sub-RQ2 and sub-RQ4 respectively. This section describes the proposed experiments in detail and the evaluation metric that would be used to compare systems in experiment 2.

4.5.1 Experiment 1

This experiment is defined to analyse and compare the different key features that are extracted by deep pre-trained models. The deep pre-trained models selected are VGG-16 and DenseNet-121. These models are selected because of the findings presented in section 2.2 i.e. these deep pre-trained models have outperformed other prevalent conventional techniques on the task of multi-class image classification hinting towards the fact that these models are successful in extracting fine low level meaningful features from underlying images. Hence, to answer the sub RQ2, feature maps that are activated by these deep models would be analysed.

Two images would be fed to these deep models consisting the image of a 'husky' from the ImageNet dataset as represented in section 4.2. The other one is an image of a healthy honeybee as shown in section 4.2. The image is chosen randomly from the train and the test set. Also both images are normalized and rescaled to a resolution of 224X224 pixels before input to the deep pre-trained model. The feature maps activated by these deep models would be visualized and analysed. The visualizations of feature images would be images that model is able extract from the underlying image after the convolutional block. The figure 4.4 below represents the architecture that would be used to conduct this experiment. This analysis will lead to a set of unique key features essential for image classification that are being extracted by these deep models. This set would answer sub RQ2 by enlisting the features that these deep pre-trained models are able to extract from the underlying images.

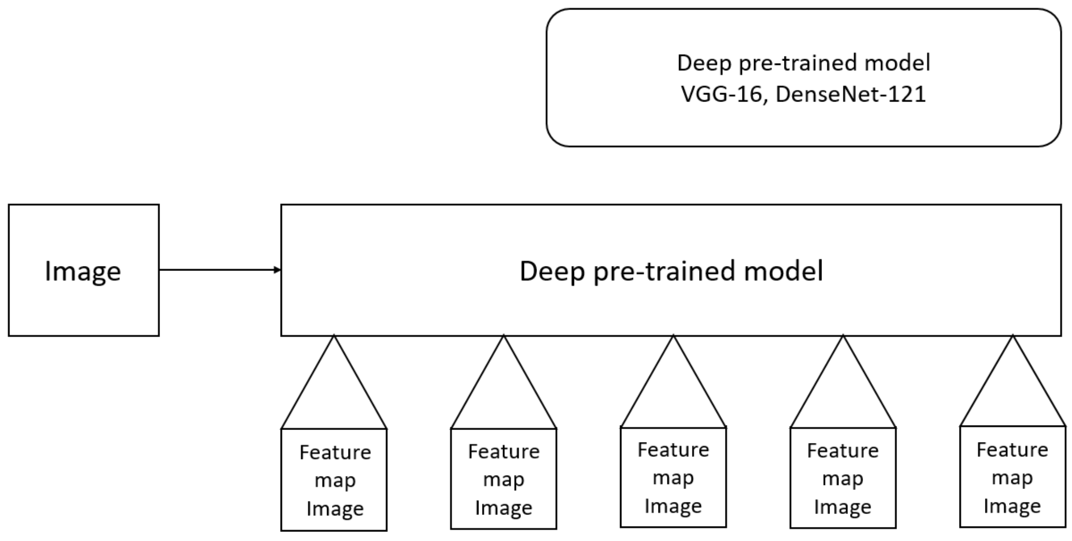


Figure 4.4: Architecture - Experiment 1

4.5.2 Experiment 2

This experiment is designed to identify the most suitable transfer learning technique for the task for image based bee health classification. To conduct this experiment, the two deep pre-trained models mentioned in the previous section would be fine-tuned on the task of image based bee health classification with the strategies outlined in section 4.4. Both deep pre-trained models i.e. VGG-16 and DenseNet-121 consist of 5 convolutional blocks. Hence, the strategies defined in section 4.4 lead to four possible variants of fine-tuning a deep pre-trained model as represented in figure 4.5.

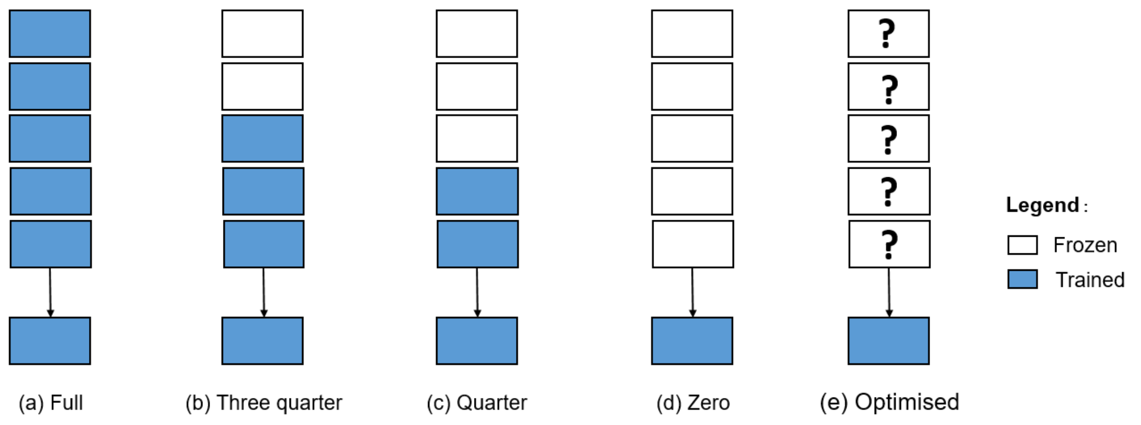


Figure 4.5: Fine-tuning strategies - Experiment 2

The four possible outcomes of the defined strategies in section 4.4 are visualised in figure 4.5. The variant of each deep-model is named for the ease of comparison, namely 'Full', 'Three-quarter', 'Quarter' and 'Zero', each representing a possible variant. The 'Full' model is the outcome of strategy 1 in section 4.4 where all convolutional blocks of the pre-trained deep model are fine-tuned along with the classification layers on top. Both the 'Three-quarter' and 'Quarter' variants branch out from strategy 2 where half of the convolutional base model is frozen while training while half of the base model is optimised during the training process. The 'Zero' variant is a result of strategy 3 where the complete base model is frozen and the classification layer on top is optimised. Important to consider here is the outcome of experiment 1. The results of experiment 1 would identify the convolutional blocks that are responsible for extracting lower level fine features and hence up to which convolutional blocks would need to be optimally frozen for a better training environment. The variants that would result from the outcome of experiment 1 is unknown and hence it is named the 'optimised' variant. Hence the complete set of models in the designed experiment include 'vgg-full', 'vgg-three-quater', 'vgg-quarter', 'vgg-zero', 'vgg-optimised', 'densenet-full', 'densenet-three-quarter', 'densenet-quarter', 'densenet-zero' and 'densenet-optimised'. The naming of the fine-tuned variants is straight-forward, implying that the variant 'vgg-full' represents the model with VGG-16 as the base model fine-tuned with strategy 1 i.e. the whole model along with top layers is optimised during training.

4.5.3 Evaluation metric

The systems designed to identify the honey bee health should be precise about categorizing a honey bee to a certain health status. This is so that the bee keeper can take needed actions accordingly. If the system detects a wrong disease, the bee keeper might take actions that might not bear fruit towards improving the bee hive's health. Hence, the system should precisely be able to identify the health status of a bee from its image. Hence, precision is a metric that has to be used for the evaluation of the fine-tuned systems. The equations of calculating precision and further macro-precision are represented in equation4.1 and equation4.5. Another metric that is considered for evaluation of the systems is recall i.e. equation4.2 and further macro-recall i.e. equation4.4. These metrics represent to what degree is a model sensitive towards the target class. This metric is important to measure to what ratio are fine-tuned models able to detect a disease from an input image of a honey bee. This metric also represents the effectiveness of the working system by representing how many times is the model able to identify the underlying disease. Intuitively, both precision and recall would be at trade off with each other and hence a metric

F1 represented by equation 4.3, is also used to evaluate the models. The metric macro-F1 (4.6) represents how much a model is sensitive and specific towards the target class. Something to note is that macro averaging technique is considered for calculating evaluation metrics. This is adopted to provide each class in the test set with equal importance while calculating evaluation metrics for comparison. Hence, sub-RQ3 is conclusively answered by the set of evaluation metrics {macro-Precision, macro-Recall, macro-F1}.

$$\textit{Precision} = \frac{\textit{All Correctly Classified Examples for a target class}}{\textit{All Classified Examples for this class}} \tag{4.1}$$

$$\textit{Recall} = \frac{\textit{All Correctly Classified Examples for a target class}}{\textit{All Examples for this class}} \tag{4.2}$$

$$\textit{F1} = \frac{\textit{2 * Precision * Recall}}{\textit{Precision + Recall}} \tag{4.3}$$

$$\textit{macro-Recall} = \textit{avg (Recall for all target classes)} \tag{4.4}$$

$$\textit{macro-Precision} = \textit{avg (Precision for all target classes)} \tag{4.5}$$

$$\textit{macro-F1} = \textit{avg (F1 for all target classes)} \tag{4.6}$$

Chapter 5

Results and Discussions

This chapter enlists and discusses the results of experiments described in section 4.5. The first experiment is designed to compare extracted features of images from different datasets by different models. While the second experiment is designed to identify the best possible fine-tuning technique for the task of image based bee health classification on this dataset.

5.1 Experiment 1

This section enlists and discusses the results of experiment 1 in detail. The section 4.5.1 describes that two images are chosen as sample to visualize the feature maps activated by the two different deep pre-trained models namely VGG-16 and DenseNet-121. This experiment is designed to visualise and observe the lower level fine and smooth features that the deep pre-trained models are able to extract from input images. The input images are selected from both datasets ImageNet and BeeImage dataset. The images of extracted features are reported in this section and discussed in detail. A conclusion on which convolutional blocks of the two mentioned models are responsible for extracting lower level features is also drawn. The figures 5.2, 5.3, 5.4 and 5.5 represent the features extracted by the deep pre-trained models. The (a) component of the figures represent the image which was input to the models for feature extraction. The (b) component represents the features extracted by the models by the 1st convolutional block. The (c) component visualises the output of the 2nd convolutional block and the (d), (e) and (f) components represent the output from the 3rd, 4th and 5th convolutional blocks of the models. Each component contains multiple images of the extracted features of the input image because all convolutional blocks try to focus on different possible features of an input image in differentiating ways and channelises all outputs. For example, in some feature map the focus would be on edges while on some maps it might be focusing on contrast. The number of channels in convolutional blocks of VGG-16 is 64 while for DenseNet-121 is 512. However, the visualisations have been capped to 64 channels.

5.1.1 VGG-16

The figures 5.2 and 5.3 represent the features extracted from VGG-16. The first figure i.e. figure 5.2 visualizes the features extracted from the input image of a husky. This image is chosen from the ImageNet dataset. The VGG-16 model has been initially pre-trained on the ImageNet dataset as well. As can be seen, lower level features extracted by the 1st convolutional block are fine and smooth. The model is able to extract edges, shapes, edges in the background in the first convolutional block. It can also be observed that the extracted features or feature maps are translated and are also visible in the output of the 2nd convolutional block and also in the output of the 3rd convolutional block. However, the output from the 4th and 5th convolutional blocks does not represent much of the image but just shaded blocks.

The figure 5.3 represents the features extracted by the same pre-trained VGG-16 model on an input image selected from the BeeImage dataset. A similar trend, as observed above with the input example from ImageNet, is observed in this case. The lower level features extracted from input image of a bee are smooth and fine however they do not translate upto the output of the 4th and 5th convolutional block. This hints towards the fact that the lower blocks of the VGG-16 model are responsible feature extraction and passing the extracted features over to higher blocks for interpretation and vectorization. Another thing to note would be, since the last two blocks are responsible for the interpreting the extracted features, they would need to fine-tuned for interpreting the features extracted for this task. This is because the interpretation understanding of the 4th and 5th block would be very general (21,000 target classes in ImageNet), and would need to be optimised for a specific task. Also, the lower level blocks might not need fine-tuning since the blocks can already extract and visualise fine and smooth features of the bee.

5.1.2 DenseNet-121

The figures 5.4 and 5.5 represent the features extracted by DenseNet-121. Although, as mentioned before that the convolutional blocks of DenseNet-121 have 512 channels, the visualisations are capped at 64.

A similar trend as seen with the VGG-16 model is observed with the case of DenseNet-121 with feature extraction. The lower level blocks i.e. 1st, 2nd and 3rd convolutional blocks are able to extract very fine and smooth feeatures however the higher level blocks ouput only shaded squares. Something to note would be that the output from the 1st and 5th blocks of both models seems similar, however the ouputs from 2nd, 3rd and 4th block of DenseNet-121 differ from the ones coming of VGG-16. This shows that the DenseNet-121 is deep enough to transform output of the 1st block and generate even finer features in the blocks in the middle. This shows that the DenseNet-121 has much deeper networks to enhance and augment the extracted features when compared with VGG-16. Since a similar trend is observed across both the base models, the first three convolutional blocks are referred to as 'feature-extracting blocks' and the fourth and fifth blocks are referred to as 'interpreting blocks' further in the discussion.

Also, since it can be deduced that the first three convolutional blocks are 'feature-extracting blocks' and the last two blocks are 'interpreting blocks' with both models, it can be concluded that the 'optimised' variant of both models is exactly same as the 'quarter' variant of both the models. Hence, the results of 'optimised' variant are reported as the 'quarter' variant in the next section. This is also represented by figure 5.1

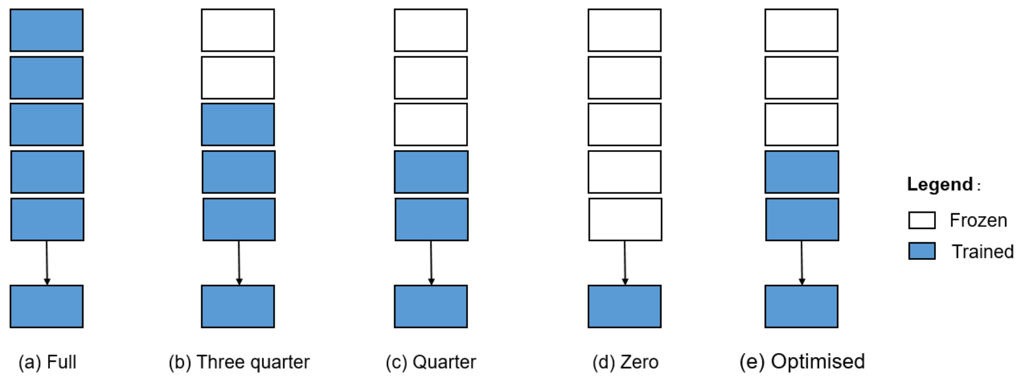
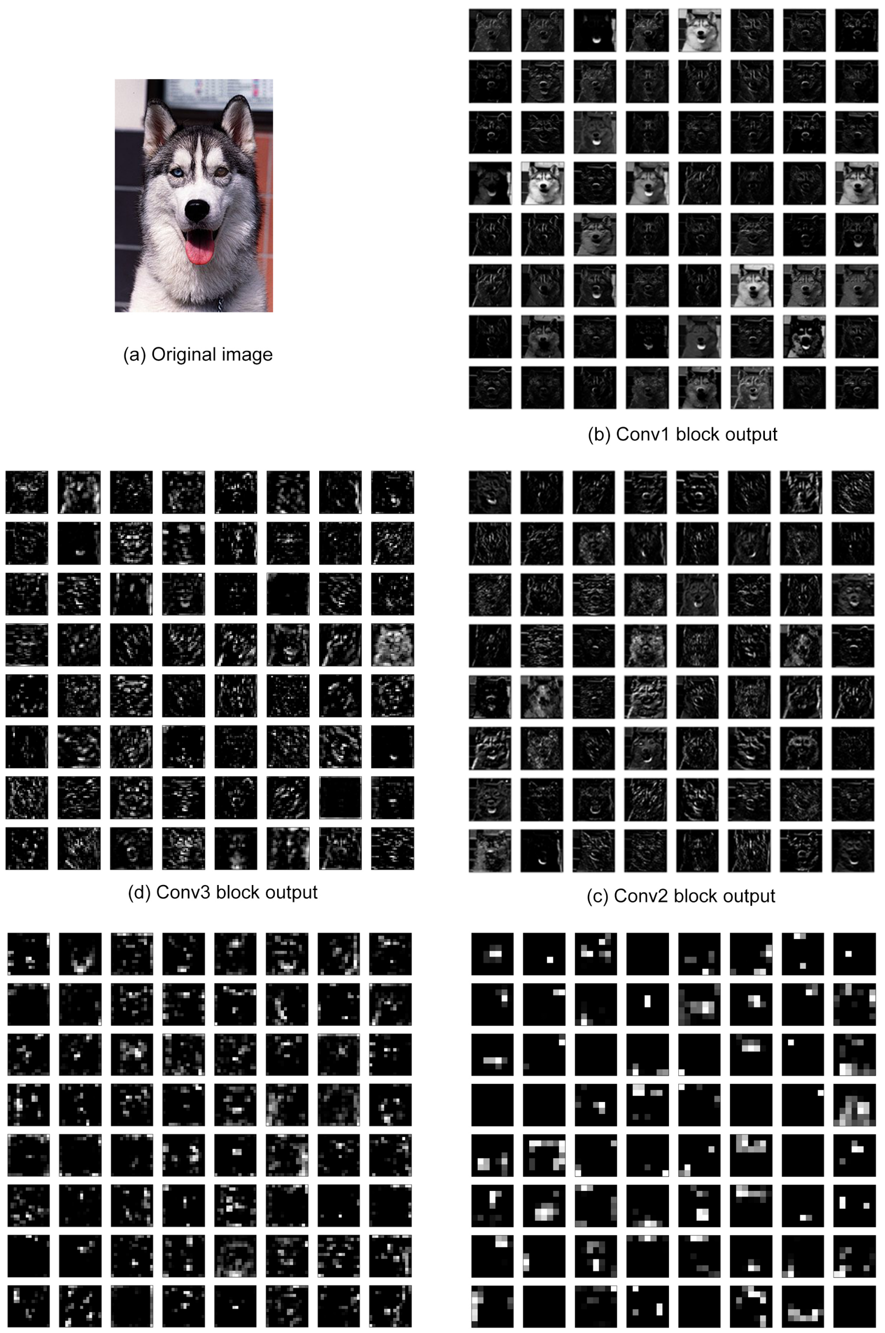


Figure 5.1: Result of Experiment 1



(a) Original image

(b) Conv1 block output

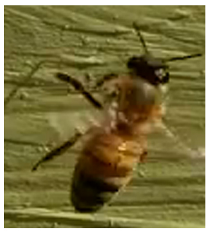
(d) Conv3 block output

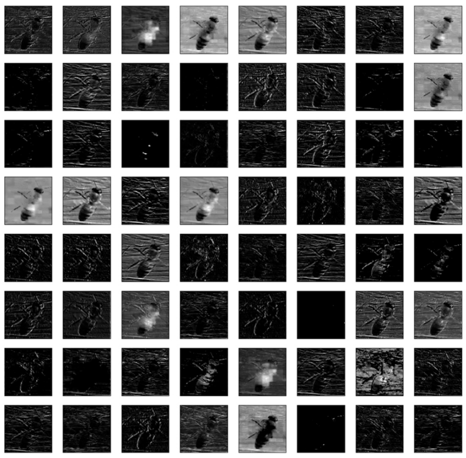
(c) Conv2 block output

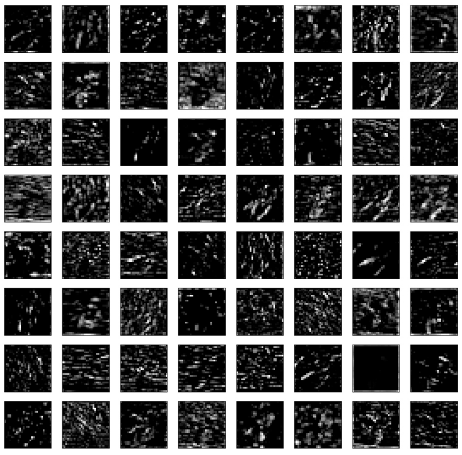
(e) Conv4 block output

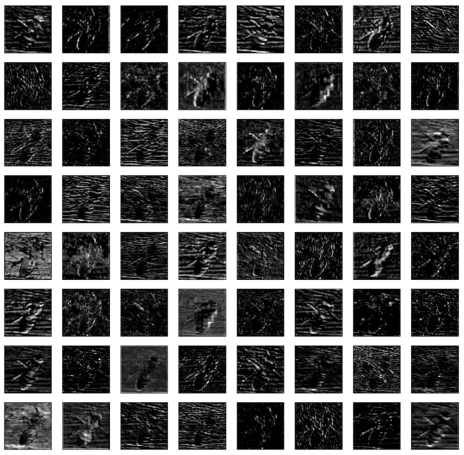
(f) Con5 block output

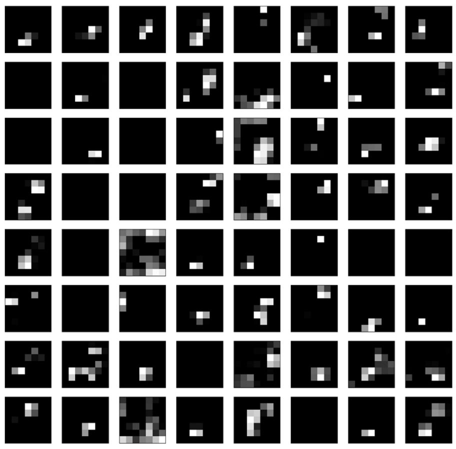
Figure 5.2: Feature extraction of ImageNet example with VGG-16


(a) Original image


(b) Conv1 block output


(d) Conv3 block output


(c) Conv2 block output


(e) Conv4 block output

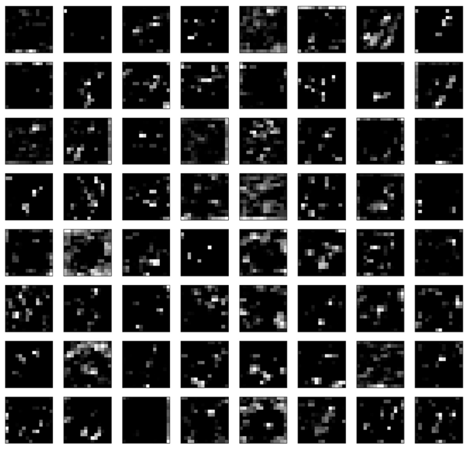

(f) Con5 block output

Figure 5.3: Feature extraction of honeybee image with VGG-16

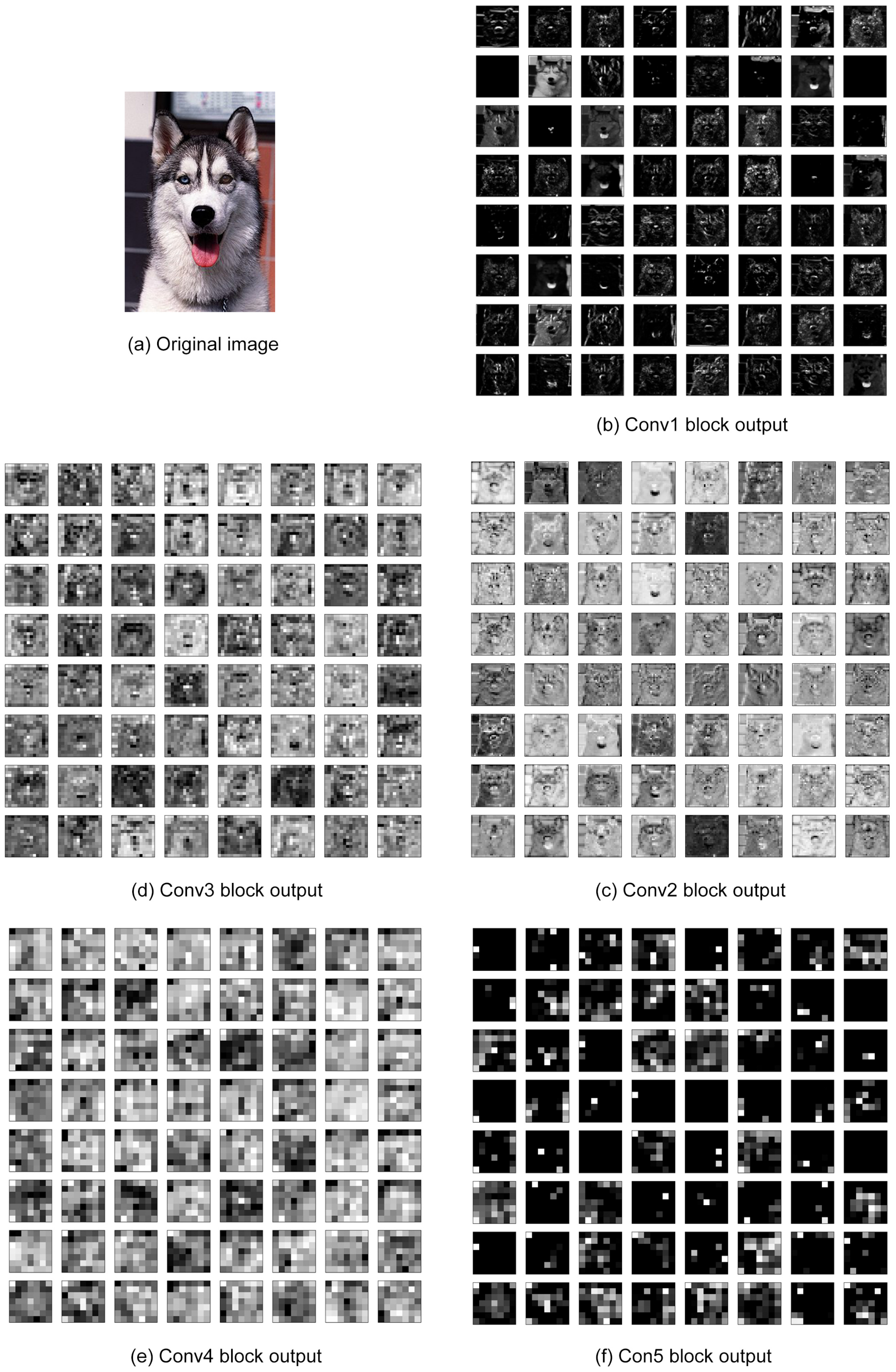
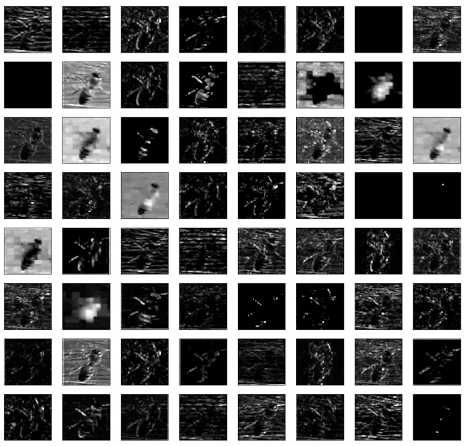
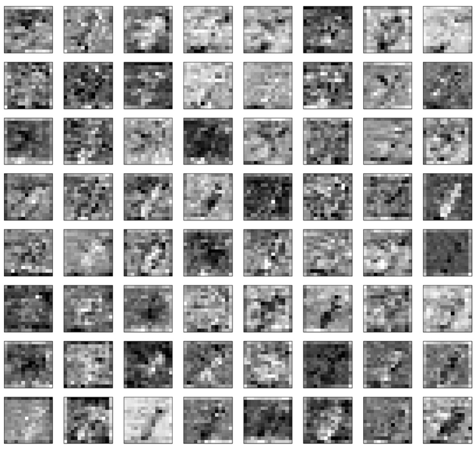
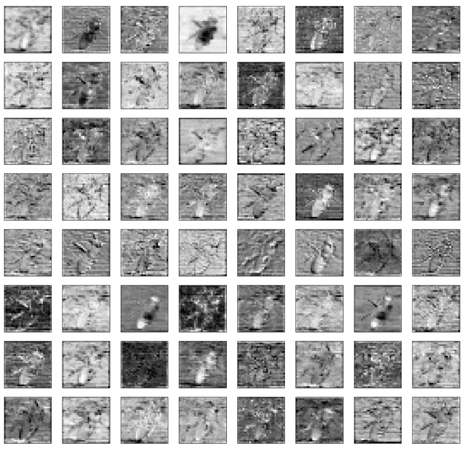


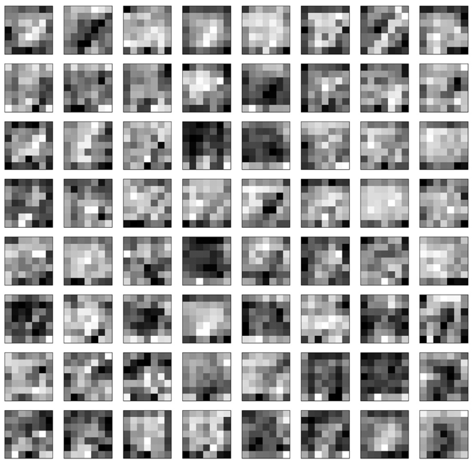
Figure 5.4: Feature extraction of ImageNet example with DenseNet-121

(a) Original image


(b) Conv1 block output


(d) Conv3 block output


(c) Conv2 block output


(e) Conv4 block output

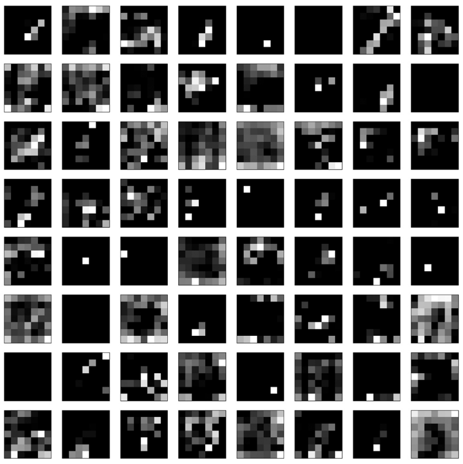

(f) Con5 block output

Figure 5.5: Feature extraction of honeybee image with DenseNet-121

5.2 Experiment 2

This section enlists and discusses the results of experiment 2 in detail. Detailed results of this experiment are also presented in Appendix B. The section 4.5.2 describes the four fine-tuning strategies for fine-tuning the deep pre-trained models. As discussed, two models namely, VGG-16 and DenseNet-121 are fine-tuned and their classification performance on the test set is discussed. The test set is described in detail in chapter 3. The four different strategies of fine-tuning along with the variant to be considered after analysing results of experiment - 1 leads to a total ten fine-tuned models, namely 'vgg-full', 'vgg-three-quater', 'vgg-quarter', 'vgg-zero', 'vgg-optimised', 'densenet-full', 'densenet-three-quarter', 'densenet-quarter', 'densenet-zero' and 'densenet-optimised. As discussed in the above section, the 'feature-extracting blocks' of the deep models are the first three and the 'interpreting blocks' are the top two convolutional blocks. Preserving the feature extraction abilities of the models, and fine-tuning the rest of the model, would effectively result in 'quarter' variants of the models. The nomenclature of these models is also discussed in detail in section 4.5.2 along with experiment 2. The evaluation metrics that are highlighted while definition of the experiment are also reported in this section.

5.2.1 VGG

Model name	**macro-Precision**	**macro-Recall**	**macro-F1**
vgg-full	0.47	0.50	0.45
vgg-three-quarter	0.57	0.59	0.57
vgg-quarter	0.78	0.70	**0.73**
vgg-zero	0.81	0.34	0.38

Table 5.1: Results of VGG models

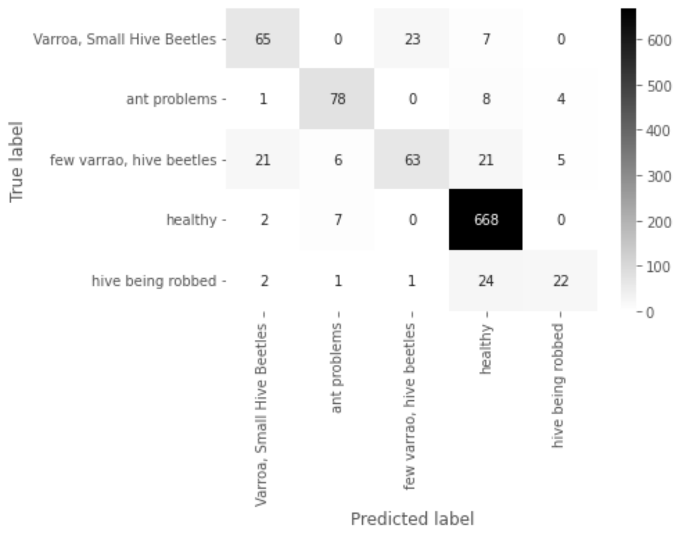


Figure 5.6: Confusion matrix vgg-quarter

The table 5.1 above represents the results of performance of the different VGG models. As can be seen, vgg-quarter out-performs other VGG-16 models in consideration with a macro-F1 of 0.73. Following the vgg-quarter model are the vgg-three-quarter, vgg-full and vgg-zero models with a macro-F1 of 0.57, 0.45 and 0.38 respectively. As discussed earlier, macro-F1 scores represent a wholesome classification performance of the model taking into consideration both the precision scores and recall scores which are at trade off. The figure 5.6 represents the confusion matrix determined by the predictions of the vgg-quarter model on the test set. It can be seen that the model indeed learns to distinguish between multiple health classes from input bee images.

The model vgg-quarter hence also results in a macro-precision of 0.78 and a macro-recall of 0.70. However, the precision scores of this model lag behind the precision scores of the vgg-zero. But this does not imply that vgg-zero is more precise at detecting bee health statuses from it's image. Upon close analysis and comparison of the confusion matrices determined by the predictions from both the fine-tuned models it can be observed that that vgg-zero doesn't learn much from the training process and ends up classifying almost all test samples as 'healthy'. The confusion matrix of the vgg-zero model is visualised by figure 5.7. The number of 'healthy' samples are high in number in the test set and hence boost the precision scores of the model. This is one of the major reasons why F1 scores are taken into consideration to provide a more 'realistic' insight on the classification performance of the model. Something else to note would be if the 'weighted' technique of averaging for classification metrics was considered instead of the 'macro' technique, the results might have been skewed and might not represent a holistic view. Simply because the distribution of the test set is highly imbalanced, leading to 'boosted' or 'inflated' metrics as seen above in case of precision scores of vgg-zero. The precision scores, recall scores and F1 scores for vgg-quarter all exceed the other models with VGG-16 as the base model.

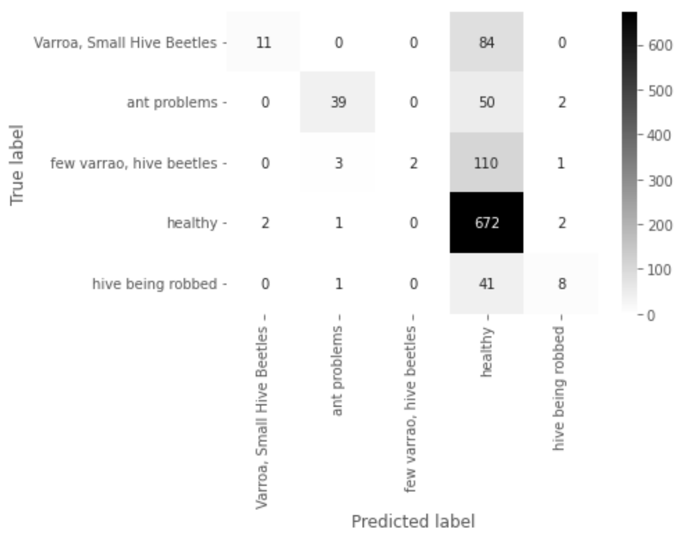


Figure 5.7: Confusion matrix vgg-zero

5.2.2 DenseNet

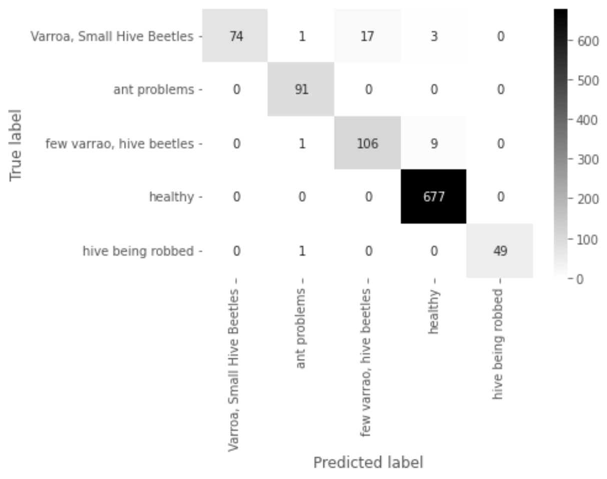


Figure 5.8: Confusion matrix for densenet-quarter

Model name	macro-precision	macro-recall	macro-F1
densenet-full	0.90	0.90	0.90
densenet-three-quarter	0.90	0.92	0.91
densenet-quarter	0.96	0.93	**0.95**
densenet-zero	0.87	0.71	0.77

Table 5.2: Results of DenseNet models

The table 5.2 represents the classification results of the models that are fine-tuned with DenseNet-121 as the base model. As can be seen in this case too, the densenet-quarter model outperforms all its other counterparts with a macro-F1 of 0.95. The performances of densenet-three-quarter, densenet-full and densenet-zero follow with a macro-F1 of 0.91, 0.90 and 0.77. A similar trend is observed with models with VGG-16 as its base as discussed above. These trends are discussed in detail further in this section.

The densenet-quarter brings a macro-precision and a macro-recall of 0.96 and 0.93. These values also exceed the other macro-precision and macro-recall values reported in the table. It can also be observed that densenet-three-quarter and densenet-full perform close to the densenet-quater model with their individual macro-recall scores. However, the densenet-quarter model outperforms other models 'significantly' by a margin of 0.06 at least. The confusion matrix determined by the predictions of densenet-quarter on the test set is visualised in figure 5.8.

5.2.3 VGG vs DenseNet

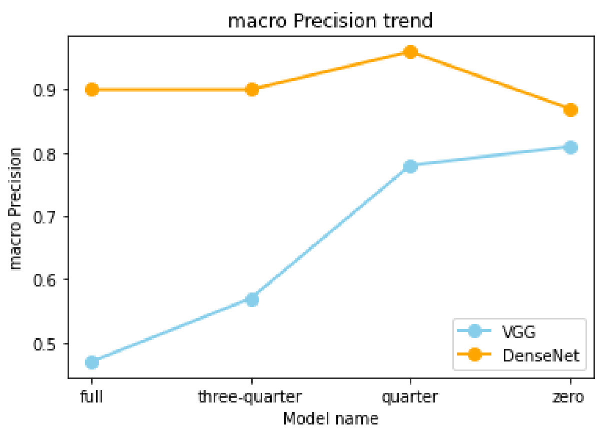


Figure 5.9: macro Precision Trend

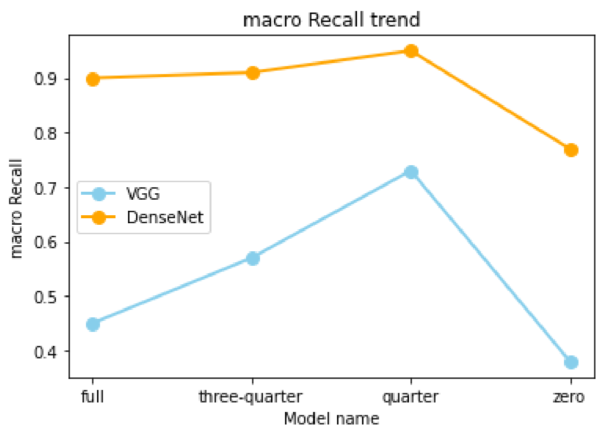


Figure 5.10: macro Recall Trend

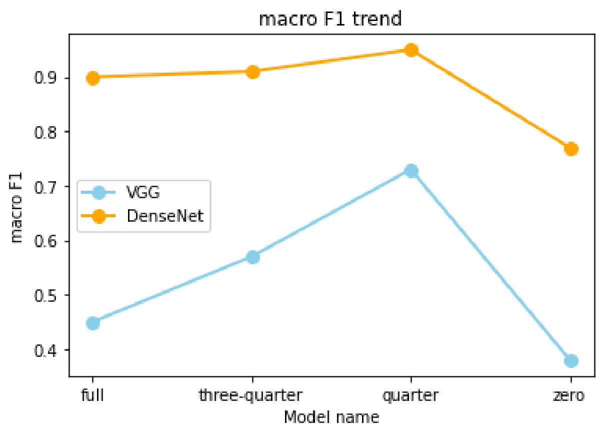


Figure 5.11: macro F1 Trend

The evaluation metrics in consideration for the experiment includes macro-Precision, macro-Recall and macro-F1 scores. The visualisations 5.9, 5.9 and 5.11 represent the trend of these metrics across different model variants for both the VGG-16 based and DenseNet-121 based models. As per expectations, the DenseNet-121 based models outperform VGG-16 models as can be observed in all three plots. This can be credited to the bigger dense base of the DenseNet-121 based models providing more residual input for the layers to interpret features as discussed in chapter 2 section 2.2. Also, as discussed in section 5.1, the features extracted by DenseNet-121 were more clear and finer when compared to that of VGG-16. Hence, DenseNet-121 based models resulted in better performance when compared to those models with VGG-16 as their base models.

From figure 5.9 it can be seen that the DenseNet-121 based models outperform VGG-16 models with macro-precision scores. However, there is a visible spike with macro-precision score of the vgg-zero model. The unexpected and 'inflated' spike of this model is discussed above in section 5.2.1. Another interesting thing to note is that both the VGG-16 based models and DenseNet-121 models show similar trends across the training strategy used to fine-tune the models as can be seen in figure 5.11 with macro-F1 scores and also in figure 5.10 with macro-recall scores.

5.3 Best training strategy

The trends observed by plotting performance metrics of both the VGG-16 based models and DenseNet-121 based models are similar as can be seen in figures 5.9, 5.9 and 5.11. These trends since can be observed with both VGG-16 models and DenseNet-121 models would conclusively identify the best training strategy for deep pre-trained models. It is clear that the 'quarter' variant emerging from strategy 2 as described in section 4.4 outperforms its counterparts with both VGG-16 based models and DenseNet-121 based models. This also highlighted in figure 5.12.

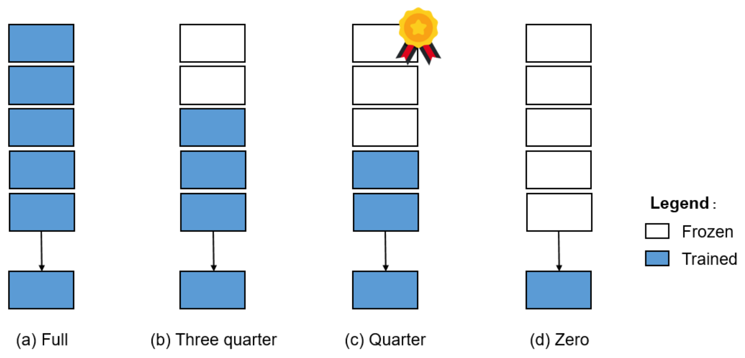


Figure 5.12: Best fine-tuning strategy

From the plots 5.9, 5.9 and 5.11 presented in section 5.2.3 it can be seen that the second best performing model in both cases of VGG-16 based models and DenseNet-121 based models, is the 'three-quarter-variant' which also emerges from strategy 2 as described in section 5.2.3. This hints towards the different fine-tuning results of the 'feature-extracting blocks' and 'interpreting blocks' of the base models. As can be seen in section 5.1 both models are able to extract the features of the input images of the bees upto and until the third convolutional block and the outputs from the fourth block and fifth block hardly represent the input image and its features. Hence, the results indicate that finetuning selected blocks of the deep pre-trained models without disturbing other blocks' weights results in a better performing model in this case. Also, on this basis it can also be defined that the first three convolutional blocks are the 'feature-extracting blocks' of both the deep pre-trained models namely VGG-16 and DenseNet-121 since the 'quarter' variant performs best followed by the 'three-quarter' variant.

This deduction is also supported by the observation that the other two variants i.e. ’zero’ and ’full’ lag behind in performance to that of ’quarter’ and ’three-quarter’ variants. Implying, that the model performance decreased if either the ’feature-extracting blocks’ were optimised during the training process and the ’interpreting blocks’ were not optimised during the training process with a training set of this size. Also from section 5.1, it is observed that the top two blocks are responsible for interpretation of underlying features. Since, ’quarter’ and ’three-quarter’ models outperform the other two variants, it can be said that the interpreting blocks do indeed need optimisation to perform better. In extension, the model tries to optimise its interpreting technique which indicates that it focuses more on areas and aspects of image which the pre-trained model wasn’t initially trained to do. This can also be seen with the case of ’full’ variants in both cases (of VGG-16 and DenseNet-121). The ’full’ variant indeed does try to optimise the lower feature extracting blocks as well, but doesn’t find enough and general data to optimise completely. Hence, decreasing the complete model performance after fine-tuning. This also in turn implies, with enough dataset size the model will also try to extract features from different aspects of the image, which it wasn’t trained to by its pre-training task. Hence, it can be verified that the tasks of pre-training (on ImageNet) and the task of image based bee health classification differ in all aspects from feature extraction to interpretation.

If and only if, the size of the training set is very high for a specific task or the nature of the classification task is similar to that of the pre-training task, the strategy will have to experimented again. This can be said since provided more training data there is a huge probability that the ’feature-extracting blocks’ of the model would be optimised in a better and a more general way across all classes hence making strategy 1 of section 4.4 best for fine-tuning. Also, if the nature of the classification task at hand might be similar to that of the pre-training task, the ’interpreting blocks’ might not need fine-tuning and would be able to interpret lower level features as extracted from the lower blocks making strategy 3 from section 4.4 best for fine-tuning. However, the task of image based bee health classification is discussed to be different of the pre-training task on ImageNet. Furthermore, as explained above and why that strategy 2 from section 4.4 is the best fine-tuning strategy for the task of image based bee health classification with a dataset size similar to that of the task of image based bee health classification with the provided resources.

Chapter 6

Conclusions and Future Works

6.1 Conclusions

This section answers the research question and the sub-questions that were formulated in Chapter 1. To answer the research question conclusively the sub-questions are answered first. The answers to sub research questions and the broad RQ are as follows:

Sub RQ 1. What are the requirements which make image-based bee health classification different from the classification problems with ImageNet? From section 4.2, it can be observed that the classification on ImageNet dataset is different than the task of image based bee health classification. As is also discussed in section 5.3, the deep pre-trained networks after fine-tuning focus on different aspects of the image, which were not of interest to the model after completing the pre-training task. The models in both cases of VGG-16 and DenseNet-121 do indeed try to optimise their 'interpreting blocks', and when experimented try to optimize their 'feature-extracting blocks' as well. This clearly verifies that both tasks are different. Also, as discussed in chapters 2 and 4, the model would need to focus more on bee orientation, body structure, wings shape and other similar small and precise features which the pre-trained models being trained for 21,000 classes on ImageNet couldn't do.

Sub RQ 2. What image-based features can be inherited from pre-trained models with ImageNet for the bee health classification? It is observed from the results of experiment 1, the pre-trained deep models on ImageNet do indeed extract very fine and smooth features from underlying images. The results presented and discussed clearly visualise the extracted features by both VGG-16 and DenseNet-121. It can be inferred that the models are accurate in extracting edges, textures, colour ratios, surrounding aspects of the objects and other features mentioned in the discussion of the results. However, the task image based bee health classification differ from the classification task on ImageNet from feature extraction to interpretation.

Sub RQ 3. Which evaluation metric would be effective to evaluate the performance of artificially intelligent systems for classifying bee health status from their images? From section 4.5.3, it was observed that precision is useful so that the model is able to precisely identify the health status of honeybee from its image so appropriate actions can be considered. Also, recall is an important metric as it would represent the degree of sensitiveness of the model towards the target class. Since both these metrics are at a trade-off. Hence, the harmonic mean of both precision and recall i.e. F1 score, would be the most efficient evaluation metric. Also, since there are multiple target classes, an averaging technique, the macro averaging technique is used to calculate a final evaluation metric. So that equal importance is given to all target classes while the evaluation metric is calculated. So, it can be concluded that the macro-F1 is the best evaluation metric for the models as it represents to what extent the model is sensitive and specific in categorizing the health of the honeybee.

Sub RQ 4. What is the most suitable transfer learning scheme for image-based bee health classification, given pre-trained CNN models with ImageNet? As it can be seen in section 4.4, it can seen that there are three different transfer learning techniques to train the model for image-based bee health classification. From the result of experiment 2 in section 5.3, it can be concluded that the strategy 2, where half parameters were frozen and other half were trained, turned out to be the best transfer learning scheme. Also, the 'quarter' variant (in our case), where the bottom three layers were frozen and top two were trained gave the best macro F1 scores in case of both VGG-16 and DenseNet-121. This trend over both deep models shows that the feature extraction blocks of the deep models shouldn't be disturbed provided there is enough data to optimise weights to extract even finer and general features. This trend also shows that the top two blocks might also not need any fine-tuning if the down-stream task is similar to that of the pre-trained task. Hence, in conclusion the best scheme involves the fine-tuning of the top two blocks responsible for interpretation since the task of image based bee health classification is very different from the task of the pre-training for the deep models. However, the fine-tuning of the lower blocks is dependent on the size of dataset provided for the downstream task in this case image based bee health classification. Given the scope and resources of the project the 'quarter' and 'third-quarter' models perform best (in order), implying that strategy 2 from section 4.4 is the best strategy to fine-tune models for the task of image based bee health classification.

RQ.How does transfer learning with available pre-trained CNN models perform on the task of image-based bee health classification? Transfer learning with available pre-trained CNN models performs well with a macro-F1 of 0.95 and 0.73 delivered by DenseNet-121 and VGG-16 based models. The DenseNet-121 based models perform significantly better than VGG-16 based models since they are deeper and much more complex. Also, the best fine-tuning strategy for the deep pre-trained models is discussed in detail in answers to sub RQs. It is learnt that the both VGG-16 and DenseNet-121 have a set of 'feature-extracting blocks' and 'interpretation blocks' within their architecture. Experiments' results and discussions conclude that that if the interpreting blocks' of the deep models are optimised the models deliver the best results. Also, a conclusion is drawn for fine-tuning tasks in future for deep pre-trained models based on the size of dataset of the downstream task and similarity between the downstream task and pre-training task.

6.2 Future Works

This research elaborates on ways to use and build state-of-the-art artificially intelligent systems for the task of image based bee health classification. In its work, the project identifies best pre-trained deep models that can be used for this task. Furthermore, clear strategies are outlined, experimented and evaluated to identify the best scheme to develop the model based on pre-trained deep models. In its results, the work concludes on specific strategies of fine-tuning dependent on the size of data available for the down-stream task and similarity between the down-stream task and the pre-train task. Future work of this work might include definition of ways to quantify similarity between the pre-train tasks and the down-stream task. This strategy would equip researchers to clearly evaluate similarity between tasks and choose a fine-tuning strategy. In extension, the work might also include a strategy to quantify and define a threshold for down-stream task dataset size to pre-train task dataset size. This would also clearly define what size of dataset might be needed to fine-tune feature extracting blocks and make the performance of resulting models even better.

From an industry perspective, with regards to data collection, it would be beneficial to use images collected in real time from machine BeeSense. The results derived in experiments are from a dataset collection of images of honey bees collected in a free environment. However, the images collected by cameras in the machine might not be able to provide complete bee structure and orientations. Hence, it might beneficial to observe models' performance on real time data from BeeSense. However, it can be said from this work that the models are adequate to determine bee health status from bee images and also deliver good performances. Hence, if good classification performance is not achieved with the discussed models and strategies, it would be a hint to improve data collection or pre-processing techniques. Another aspect of the research would be to involve an entomologist to help and identify what features is the model able to identify which separates targets classes to get a better understanding of the inference mechanism within the models. With this new outlook models can optimized using various techniques to focus and learn more about differentiating features.

Bibliography

[1] A. Klein, B. Vaissière, J. Cane, I. Steffan-Dewenter, S. Cunningham, C. Kremen, and T. Tscharntke, “Importance of pollinators in changing landscapes for world crops,” *Proceedings. Biological sciences / The Royal Society*, vol. 274, pp. 303–13, 03 2007.

[2] J.-C. Sandoz, “Behavioral and neurophysiological study of olfactory perception and learning in honeybees,” *Frontiers in Systems Neuroscience*, vol. 5, 2011. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnsys.2011.00098

[3] E. Kontos, A. Samimi, R. W. Hakze-van der Honing, J. Priem, A. Avarguès-Weber, A. Haverkamp, M. Dicke, J. L. Gonzales, and W. H. van der Poel, “Bees can be trained to identify sars-cov-2 infected samples,” *bioRxiv*, 2021. [Online]. Available: https://www.biorxiv.org/content/early/2021/10/18/2021.10.18.464814

[4] “Beeimage dataset,” https://www.kaggle.com/jenny18/honey-bee-annotated-images.

[5] K. Weiss, T. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big Data*, vol. 3, 05 2016.

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015.

[7] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” *Neural Information Processing Systems*, vol. 25, 01 2012.

[8] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2015.

[9] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” 2018.

[10] X. Zhu, C. Vondrick, C. C. Fowlkes, and D. Ramanan, “Do we need more training data?” *International Journal of Computer Vision*, vol. 119, no. 1, p. 76–92, Mar 2015. [Online]. Available: http://dx.doi.org/10.1007/s11263-015-0812-2

[11] P. Neumann and N. L. Carreck, “Honey bee colony losses,” *Journal of Apicultural Research*, vol. 49, no. 1, pp. 1–6, 2010. [Online]. Available: https://doi.org/10.3896/IBRA.1.49.1.01

[12] D. vanEngelsdorp and M. D. Meixner, “A historical review of managed honey bee populations in europe and the united states and the factors that may affect them,” *Journal of Invertebrate Pathology*, vol. 103, pp. S80–S95, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022201109001827

[13] A. A. Ghamdi, “Modeling of honey bee and varroa mite population dynamics,” 2004.

[14] M. Leza, M. A. Miranda, and V. Colomar, “First detection of vespa velutina nigrithorax (hymenoptera: Vespidae) in the balearic islands (western mediterranean): a challenging study case,” *Biological Invasions*, 12 2017.

[15] A. Cuthbertson, M. Wakefield, M. Powell, G. Marris, H. Anderson, G. Budge, J. Mathers, L. Blackburn, and M. Brown, “The small hive beetle aethina tumida: A review of its biology and control measures,” *Current Zoology*, vol. 59, pp. 644–653, 10 2013.

[16] M. Nouvian, J. Reinhard, and M. Giurfa, “The defensive response of the honeybee Apis mellifera,” *Journal of Experimental Biology*, vol. 219, no. 22, pp. 3505–3517, 11 2016. [Online]. Available: https://doi.org/10.1242/jeb.143016

[17] A. C. Oudemans, “On a new genus and species of parasitic acari.”

[18] P. Akratanakul and M. Burgett, “Varroa jacobsoni: A prospective pest of honeybees in many parts of the world,” *Bee World*, vol. 56, no. 3, pp. 119–121, 1975. [Online]. Available: https://doi.org/10.1080/0005772X.1975.11097554

[19] F. D. Rinkevich, R. G. Danka, and K. B. Healy, “Influence of varroa mite (varroa destructor) management practices on insecticide sensitivity in the honey bee (apis mellifera),” *Insects*, vol. 8, no. 1, 2017. [Online]. Available: https://www.mdpi.com/2075-4450/8/1/9

[20] K. Loope, J. Baty, P. Lester, and E. Rankin, “Pathogen shifts in a honeybee predator following the arrival of the varroa mite,” *Proceedings of the Royal Society B: Biological Sciences*, vol. 286, 01 2019.

[21] A. Beaurepaire, N. Piot, V. Doublet, K. Antunez, E. Campbell, P. Chantawannakul, N. Chejanovsky, A. Gajda, M. Heerman, D. Panziera, G. Smagghe, O. Yañez, J. R. de Miranda, and A. Dalmon, “Diversity and global distribution of viruses of the western honey bee, apis mellifera,” *Insects*, vol. 11, no. 4, 2020. [Online]. Available: https://www.mdpi.com/2075-4450/11/4/239

[22] C. Wenning, “Spread and threat of the small hive beetle,” *American Bee Journal*, vol. 141, 09 2001.

[23] A. N. Payne, T. F. Shepherd, and J. Rangel, “The detection of honey bee (apis mellifera)-associated viruses in ants,” *Scientific Reports*, vol. 10, 2020.

[24] L. von Zuben, D. Schorkopf, L. Elias, A. Vaz, A. Prado Favaris, G. Clososki, J. M. Bento, and T. Nunes, “Interspecific chemical communication in raids of the robber bee lestrimelitta limao,” *Insectes Sociaux*, vol. 63, pp. 339–347, 03 2016.

[25] D. Howard, O. Duran, G. Hunter, and K. Stebel, “Signal processing the acoustics of honeybees (apis mellifera) to identify the "queenless" state in hives,” *Proceedings of the Institute of Acoustics*, vol. 35, pp. 290–297, 01 2013.

[26] M. Ogihara, M. Stoic, N. Morimoto, M. Yoshiyama, and K. Kimura, “A convenient method for detection of varroa destructor (acari: Varroidae) using roasted soybean flour,” *Applied Entomology and Zoology*, vol. 55, pp. 1–5, 09 2020.

[27] R. Tashakkori and A. Ghadiri, “Image processing for honey bee hive health monitoring,” in *SoutheastCon 2015*, 2015, pp. 1–7.

[28] K. Bjerge, C. E. Frigaard, P. H. Mikkelsen, T. H. Nielsen, M. Misbih, and P. Kryger, “A computer vision system to monitor the infestation level of varroa destructor in a honeybee colony,” *Computers and Electronics in Agriculture*, vol. 164, p. 104898, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168169918310329

[29] D. J. Kale, R. Tashakkori, and R. M. Parry, “Automated beehive surveillance using computer vision,” in *SoutheastCon 2015*, 2015, pp. 1–3.

[30] A. Zacepins and T. Karasha, “Web based system for the bee colony remote monitoring,” in *2012 6th International Conference on Application of Information and Communication Technologies (AICT)*, 2012, pp. 1–4.

[31] V. Kulyukin, S. Mukherjee, and P. Amlathe, “Toward audio beehive monitoring: Deep learning vs. standard machine learning in classifying beehive audio samples,” *Applied Sciences*, vol. 8, p. 1573, 09 2018.

[32] Y. Sun, B. Xue, and M. Zhang, “Evolving deep convolutional neural networks for image classification,” *IEEE Transactions on Evolutionary Computation*, vol. PP, 10 2017.

[33] A. Sarlashkar, M. Bodruzzaman, and M. Malkani, “Feature extraction using wavelet transform for neural network based image classification,” in *Proceedings of Thirtieth Southeastern Symposium on System Theory*, 1998, pp. 412–416.

[34] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, pp. 2278 – 2324, 12 1998.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[36] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015.

[37] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, p. 84–90, may 2017. [Online]. Available: https://doi.org/10.1145/3065386

[38] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “Imagenet large scale visual recognition challenge,” 2014, cite arxiv:1409.0575Comment: 43 pages, 16 figures. v3 includes additional comparisons with PASCAL VOC (per-category comparisons in Table 3, distribution of localization difficulty in Fig 16), a list of queries used for obtaining object detection images (Appendix C), and some additional references. [Online]. Available: http://arxiv.org/abs/1409.0575

[39] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2015.

[40] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015.

[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” 2014.

[42] R. T. P, P. R. Aravind, R. C, U. Nechiyil, and N. Paramparambath, “Audio spoofing verification using deep convolutional neural networks by transfer learning,” 2020.

[43] A. Roy, “Akhcrnet: Bengali handwritten character recognition using deep learning,” 2020.

[44] A. T. S. Kalvakolanu, “Plant disease detection from images,” 2020.

[45] G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, “Introduction to wordnet: An on-line lexical database*,” vol. 3, 01 1991.

Appendix A - Technical specifications of experiments

This chapter mentions the technical specifications used to carry out the project. The table A.1 enlists all specifications

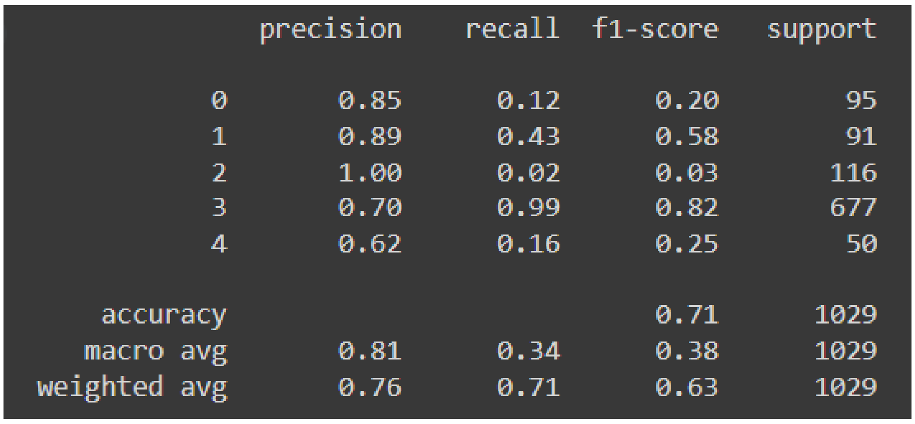
Specification	Purpose
Python	Programming language
Numpy, Pandas, CV2	Data management libraries
Matplotlib, Seaborn	Data visualisation libraries
TensorFlow	Model designing and training
Sklearn	Model evaluation
Seed value = 22	Reproducing results
Learning rate = 2e-3	Optimizing model parameters
Optimizer = Adam	Optimizing model parameters
Epochs = 5	Training of models
Batch size = 32	Training of models
Machine = NVIDIA Tesla K80	Processing
nGPU = 1	Processing

Table A.1: Technical Specifications and Descriptions

Appendix B - Detailed results of Experiment 2

B.1 VGG-16 variants

B.1.1 VGG-16 Zero


```
              precision    recall  f1-score   support

           0       0.85      0.12      0.20        95
           1       0.89      0.43      0.58        91
           2       1.00      0.02      0.03       116
           3       0.70      0.99      0.82       677
           4       0.62      0.16      0.25        50

    accuracy                           0.71      1029
   macro avg       0.81      0.34      0.38      1029
weighted avg       0.76      0.71      0.63      1029
```

Figure B.1: Classification report of zero variant

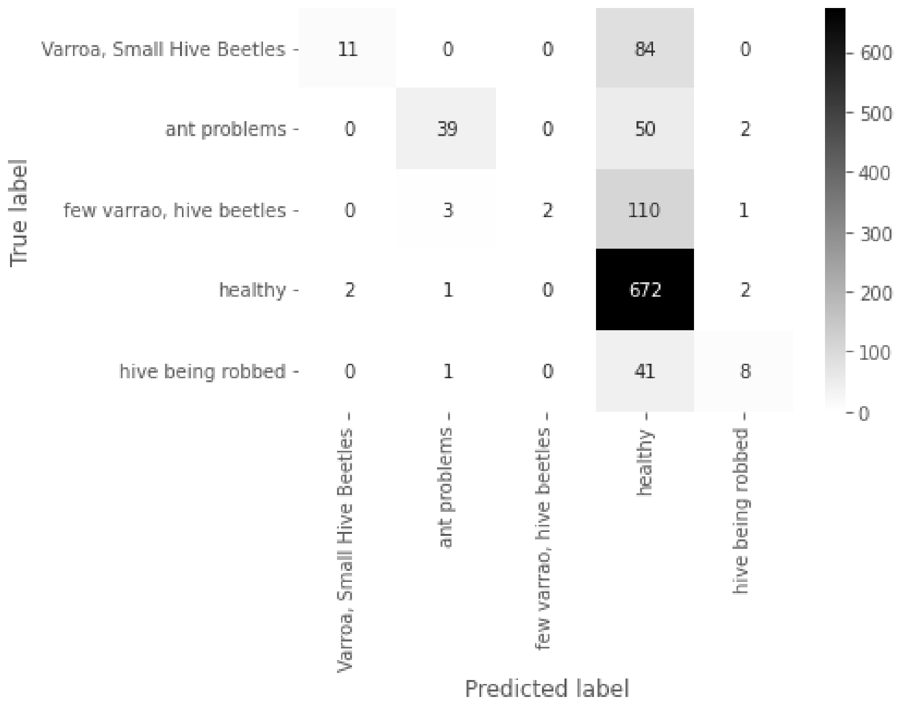
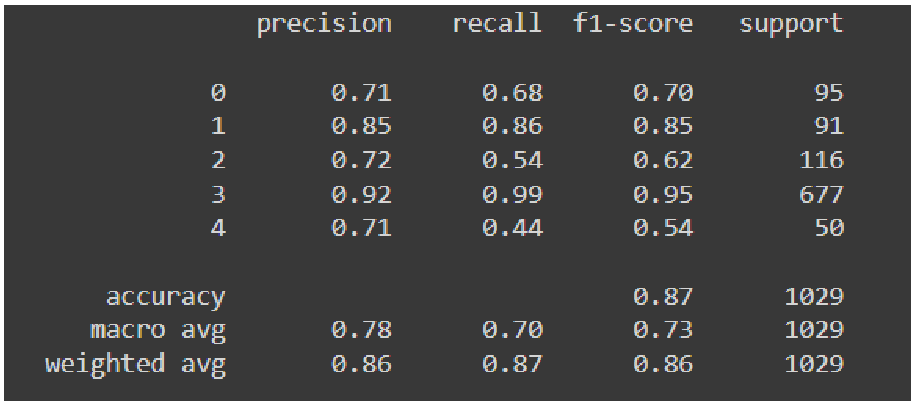


Figure B.2: Confusion matrix of zero variant

B.1.2 VGG-16 Quarter


```
              precision    recall  f1-score   support

           0       0.71      0.68      0.70        95
           1       0.85      0.86      0.85        91
           2       0.72      0.54      0.62       116
           3       0.92      0.99      0.95       677
           4       0.71      0.44      0.54        50

    accuracy                           0.87      1029
   macro avg       0.78      0.70      0.73      1029
weighted avg       0.86      0.87      0.86      1029
```

Figure B.3: Classification report of quarter variant

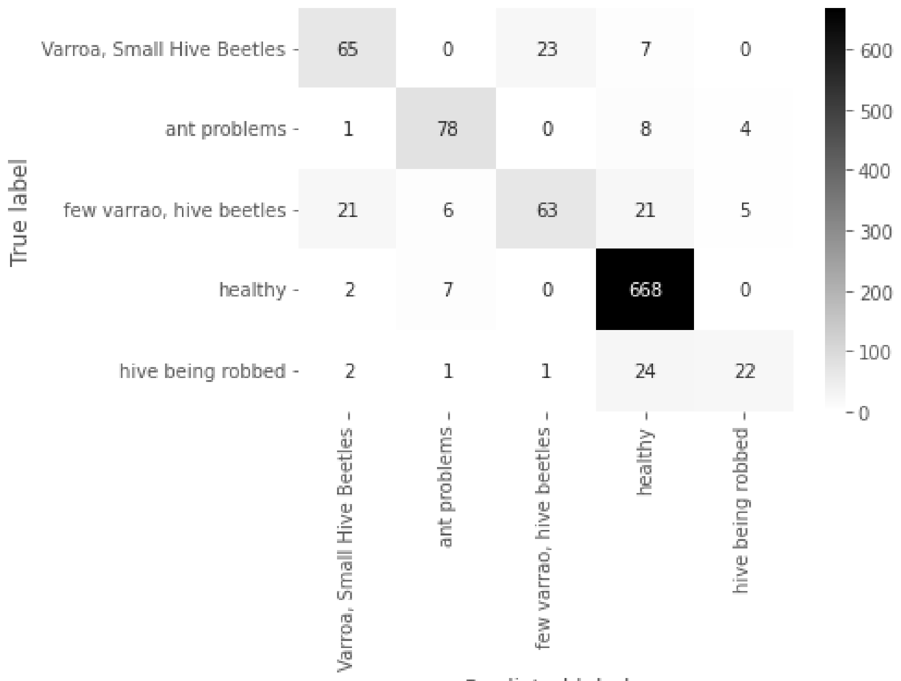
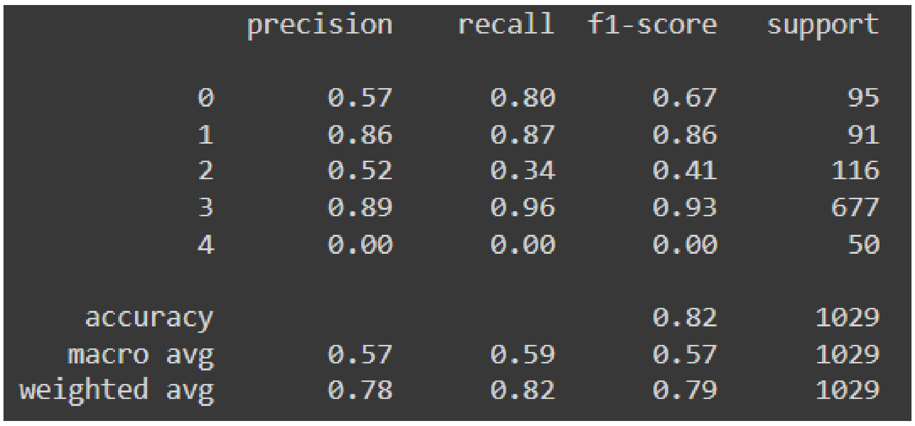


Figure B.4: Confusion matrix of quarter variant

B.1.3 VGG-16 ThreeQuarter

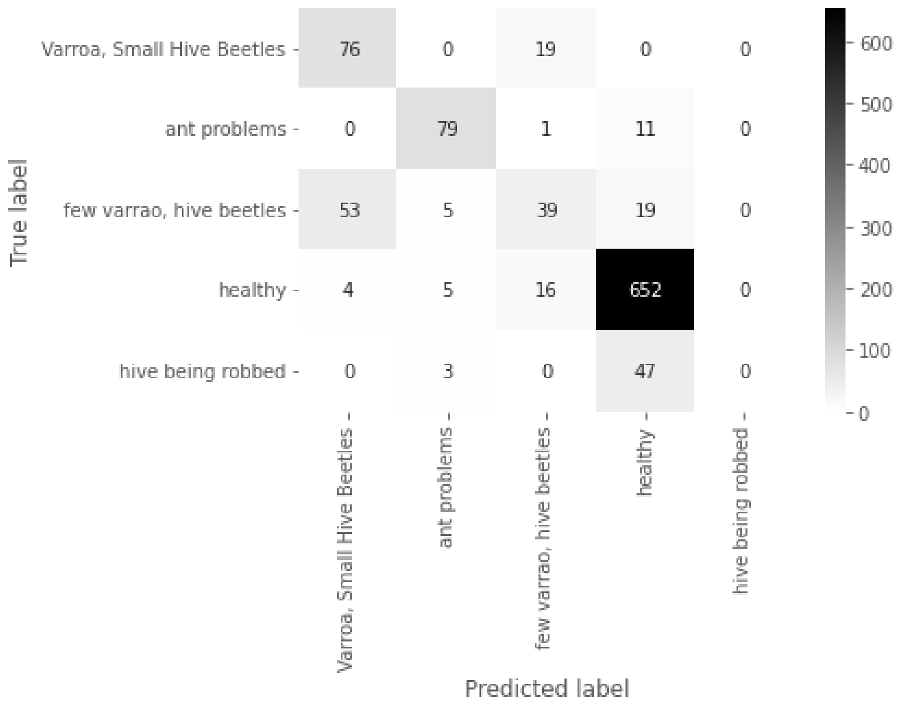


```
              precision    recall  f1-score   support

           0       0.57      0.80      0.67        95
           1       0.86      0.87      0.86        91
           2       0.52      0.34      0.41       116
           3       0.89      0.96      0.93       677
           4       0.00      0.00      0.00        50

    accuracy                           0.82      1029
   macro avg       0.57      0.59      0.57      1029
weighted avg       0.78      0.82      0.79      1029
```

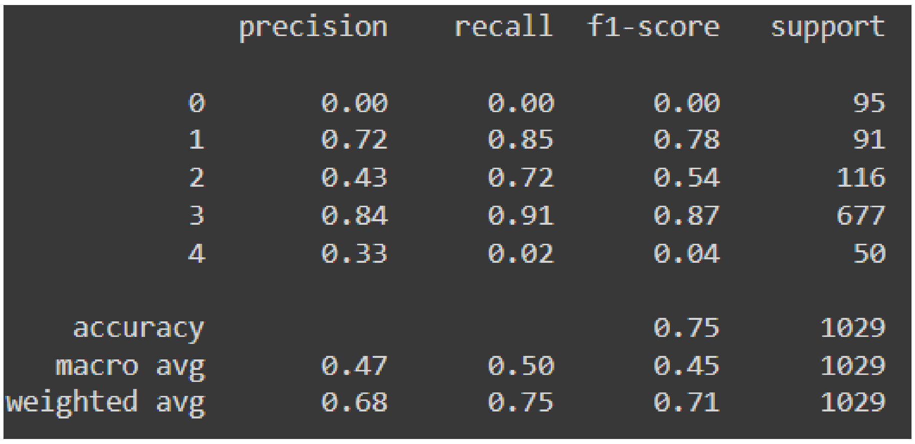
Figure B.5: Classification report of three quarter variant



True label \ Predicted label	Varroa, Small Hive Beetles	ant problems	few varrao, hive beetles	healthy	hive being robbed
Varroa, Small Hive Beetles	76	0	19	0	0
ant problems	0	79	1	11	0
few varrao, hive beetles	53	5	39	19	0
healthy	4	5	16	652	0
hive being robbed	0	3	0	47	0

Figure B.6: Confusion matrix of three quarter variant

B.1.4 VGG-16 Full



```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        95
           1       0.72      0.85      0.78        91
           2       0.43      0.72      0.54       116
           3       0.84      0.91      0.87       677
           4       0.33      0.02      0.04        50

    accuracy                           0.75      1029
   macro avg       0.47      0.50      0.45      1029
weighted avg       0.68      0.75      0.71      1029
```

Figure B.7: Classification report of full variant

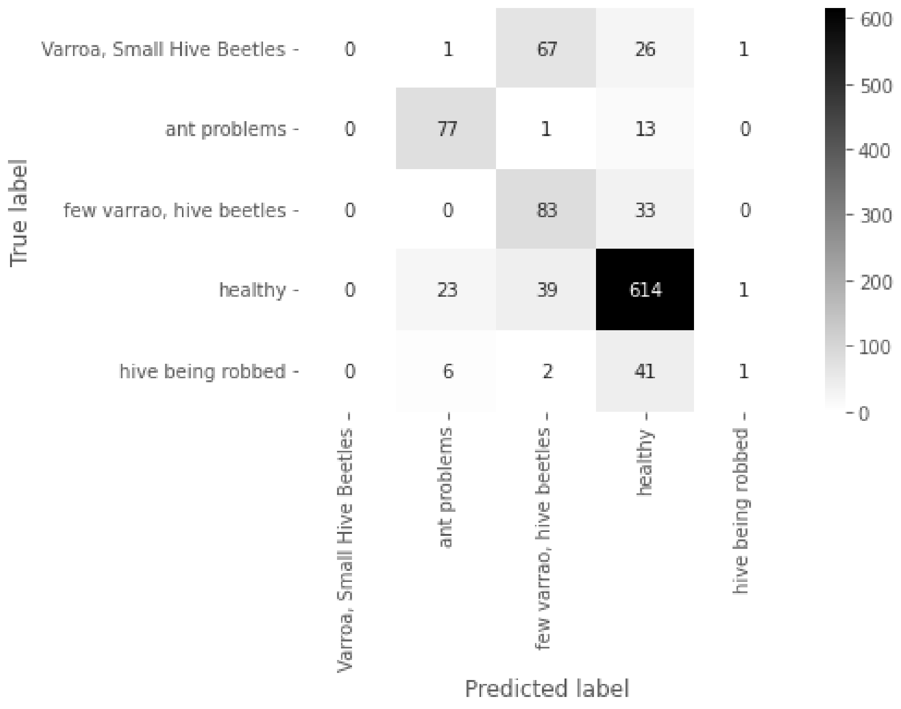
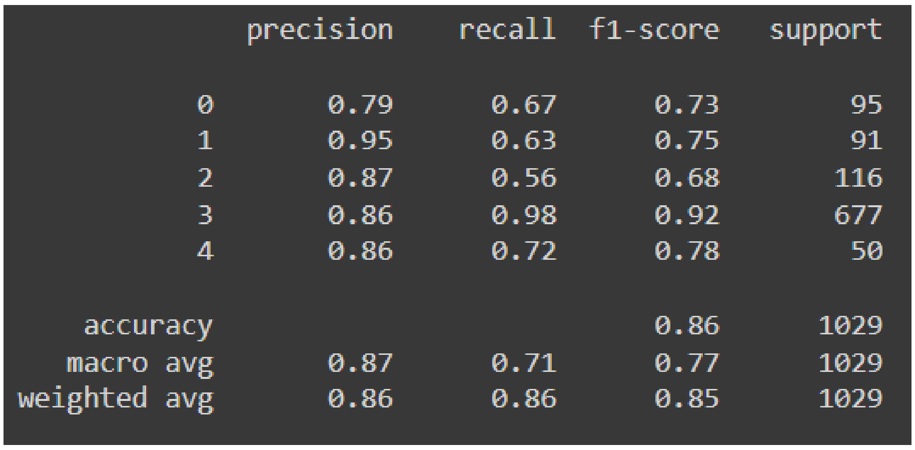


Figure B.8: Confusion matrix of full variant

B.2 DenseNet-121 variants

B.2.1 DenseNet-121 Zero


```
              precision    recall  f1-score   support

           0       0.79      0.67      0.73        95
           1       0.95      0.63      0.75        91
           2       0.87      0.56      0.68       116
           3       0.86      0.98      0.92       677
           4       0.86      0.72      0.78        50

    accuracy                           0.86      1029
   macro avg       0.87      0.71      0.77      1029
weighted avg       0.86      0.86      0.85      1029
```

Figure B.9: Classification report of zero variant

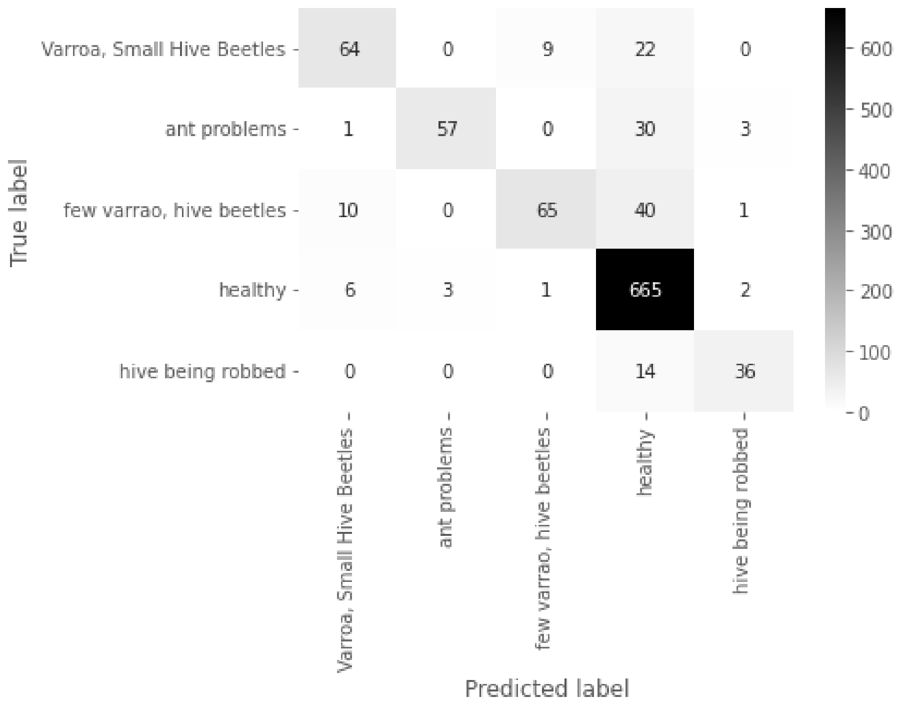
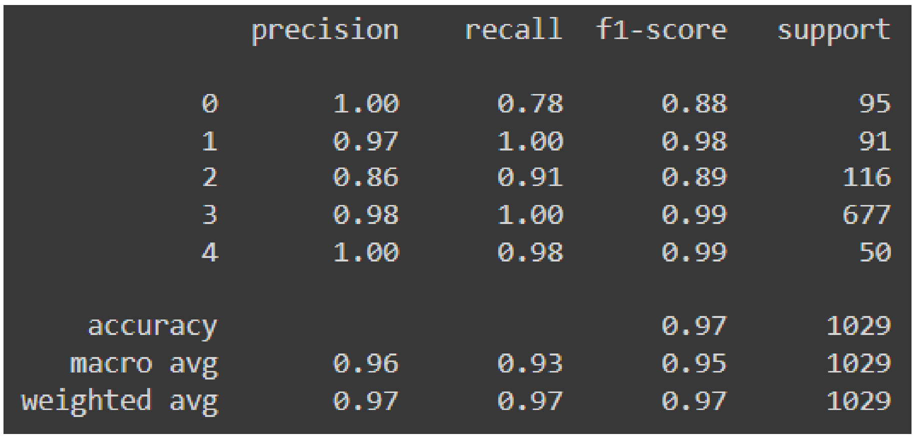


Figure B.10: Confusion matrix of zero variant

B.2.2 DenseNet-121 Quarter


```
              precision    recall  f1-score   support

           0       1.00      0.78      0.88        95
           1       0.97      1.00      0.98        91
           2       0.86      0.91      0.89       116
           3       0.98      1.00      0.99       677
           4       1.00      0.98      0.99        50

    accuracy                           0.97      1029
   macro avg       0.96      0.93      0.95      1029
weighted avg       0.97      0.97      0.97      1029
```

Figure B.11: Classification report of quarter variant

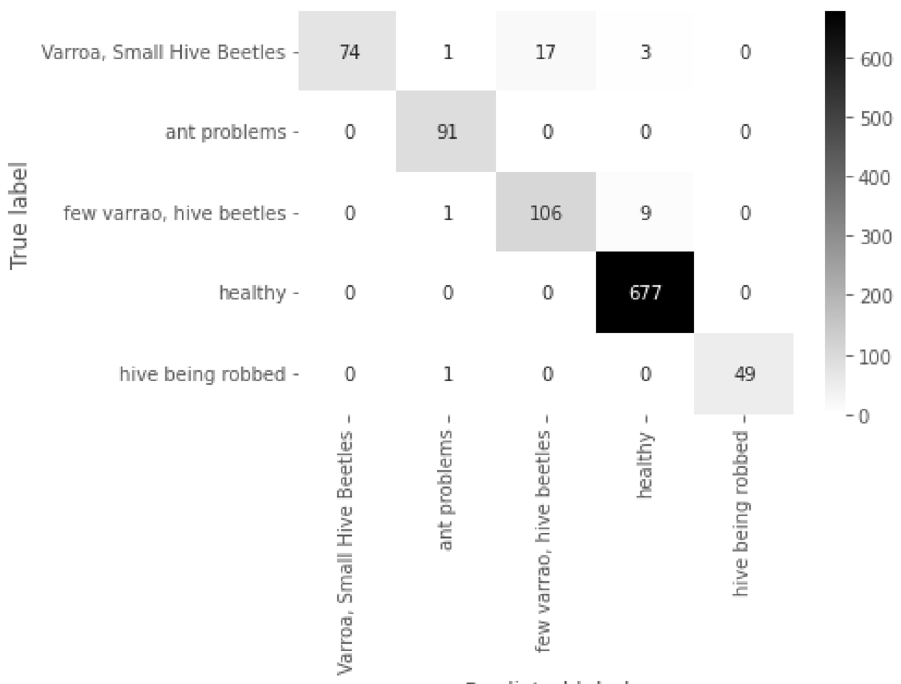
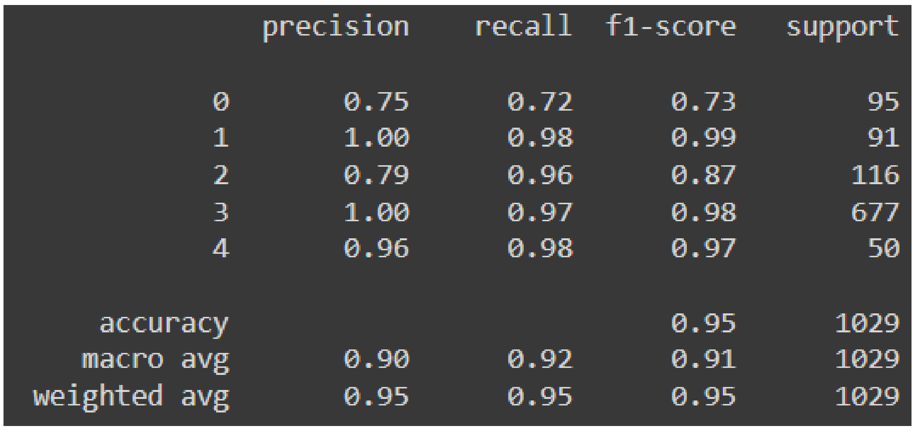


Figure B.12: Confusion matrix of quarter variant

B.2.3 DenseNet-121 Three Quarter



```
              precision    recall  f1-score   support

           0       0.75      0.72      0.73        95
           1       1.00      0.98      0.99        91
           2       0.79      0.96      0.87       116
           3       1.00      0.97      0.98       677
           4       0.96      0.98      0.97        50

    accuracy                           0.95      1029
   macro avg       0.90      0.92      0.91      1029
weighted avg       0.95      0.95      0.95      1029
```

Figure B.13: Classification report of three quarter variant

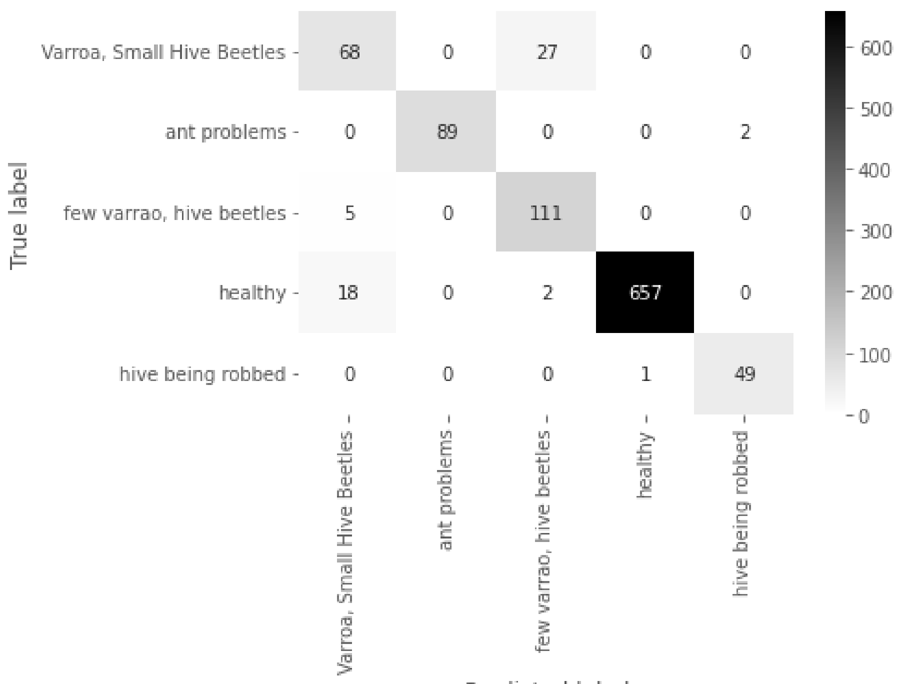
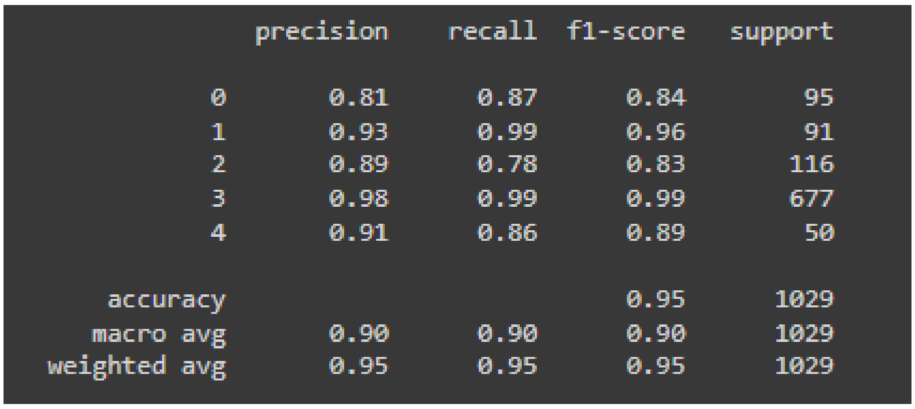


Figure B.14: Confusion matrix of three quarter variant

B.2.4 DenseNet-121 Full


```
              precision    recall  f1-score   support

           0       0.81      0.87      0.84        95
           1       0.93      0.99      0.96        91
           2       0.89      0.78      0.83       116
           3       0.98      0.99      0.99       677
           4       0.91      0.86      0.89        50

    accuracy                           0.95      1029
   macro avg       0.90      0.90      0.90      1029
weighted avg       0.95      0.95      0.95      1029
```

Figure B.15: Classification report of full variant

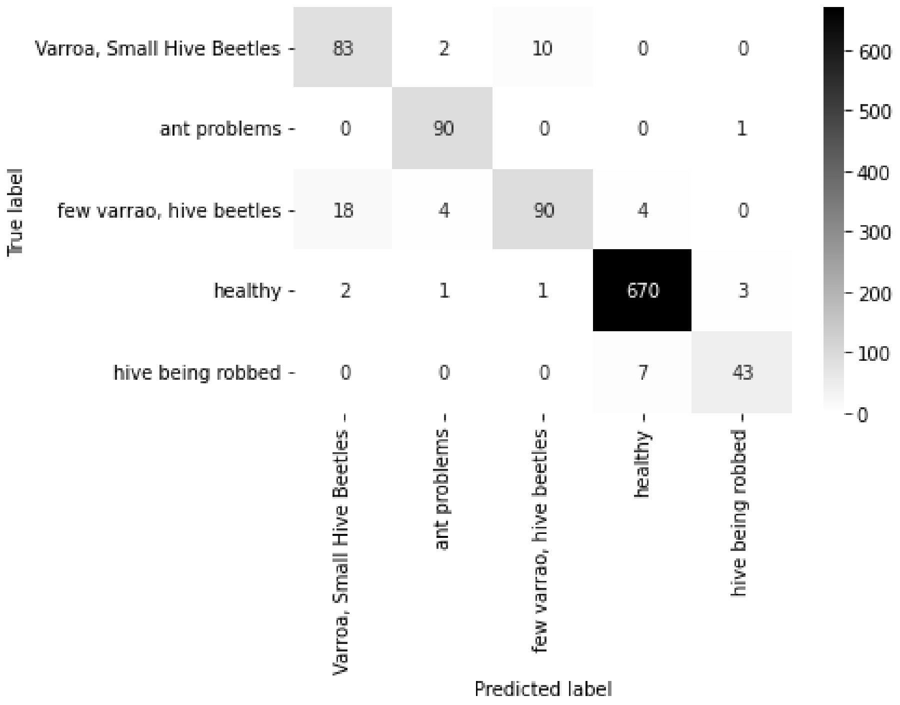


Figure B.16: Confusion matrix of full variant