Machine Learning Algorithm for Classification of Breast Ultrasound Images

A Comparison of CNN and Transfer Networks
InceptionV3, ResNet50V2, VGG19 and Xception

Jennie Karlsson & Jennifer Ramkull

LUND
UNIVERSITY

Department of Mathematics

Abstract

Breast cancer is the most common type of cancer worldwide and the incidence is increasing. Women in low- and middle-income countries have a high mortality to incidence ratio mainly due to a lack of resources and organized health-care. A cheap and reliable breast diagnostic tool could enable an earlier diagnosis for low-resource countries and contribute to a reduction in breast cancer mortality. We suggest that a point-of-care ultrasound device combined with machine learning (ML) could be a viable solution for accessible breast diagnostics.

The aim of the thesis was to develop an ML algorithm by using three different convolutional neural networks (CNN) approaches with the goal of classifying breast ultrasound images as malignant or benign: (a) A simple CNN, (b) transfer learning using the pre-trained convolutional bases InceptionV3, ResNet50V2, VGG19 and Xception and (c) eleven networks based on combinations of the four transfer networks in (b) so-called deep feature networks.

The data consisted of two breast ultrasound image data sets: (1) An Egyptian data set collected by Cairo University at Baheya Hospital consisting of 487 benign images and 210 malignant images. This data set was divided into 80% training, 10% validation and 10% test. (2) A Swedish data set collected at Unilabs Mammography Unit at Skåne University Hospital consisting of 13 benign images and 264 malignant images. This data set was only used to evaluate the models.

All networks were evaluated using AUC, sensitivity, specificity and weighted accuracy. For the networks obtained from transfer learning Gradient-weighted Class Activation Mapping (Grad-CAM) was performed to generate heatmaps indicating which part of the image contributing to the decision of the network.

The best result was achieved by the deep feature combination of InceptionV3, Xception and VGG19 with an AUC of 0.93 and sensitivity of 95.65%. The results show that the deep feature combinations have the best performance. Possible future improvements include expanding the data set by collecting more images.

Acknowledgements

This project was carried out on the initiative of Kristina Lång at Unilabs Mammography Unit, Skåne University Hospital. We want to thank breast radiologist and associate professor Kristina Lång, Division of Diagnostic Radiology, Department of Translational Medicine, Lund University, for giving us the opportunity to work on this thesis, for the data she collected and for supervising the project. We also want to thank Karl Åström and Ida Arvidsson, Centre of Mathematical Sciences, Lund University, for supervising the project and providing valuable input and feedback.

Contents

1

Introduction

Cancer is the second leading cause of death in the world. In Sweden breast cancer is the most common type of cancer in females representing approximately 30% of female cancer cases [1]. In high income countries, such as Sweden, mammography screening and timely and effective treatments, has led to a reduction in breast cancer mortality [2].

In low- and middle-income countries resources for breast-cancer care is lacking, including breast imaging devices and trained personel resulting in 62% of all breast cancer deaths in the world in 2012 [3, 4, 5]. Due to the lack of health-care resources it is not reasonable to attempt to apply screening and diagnostic imaging as it is performed in high-income countries [6]. Having access to a simple, cheap and user friendly breast diagnostic tool could be a solution to enable a timely diagnosis of breast cancer prior to an advanced, untreatable stage.

Artificial intelligence (AI) is improving its usage in image recognition and is getting increasingly popular. One part of AI is the use of deep learning, such as convolutional neural networks (CNN). CNN has the possibility to be used for detection, classification and segmentation of breast ultrasound image (BUSI) data and has proven to generate great results. Using a cheap portable ultrasound system together with CNN as a basis for diagnosis could potentially be a reasonable solution for low- and middle-income countries.

The Unilabs Mammography Unit at Skåne University Hospital has collected a data set consisting of labeled BUSI data. A second set of labeled breast ultrasound (BUS) images can be obtained through the report "data set of breast ultrasound images", collected by Cairo University at Baheya Hospital, Egypt [7].

The aim of this thesis is to develop a machine learning (ML) algorithm for diagnosis of breast cancer using BUS images with the help of different CNN approaches based on the two data sets. First, the aim is stated followed by the medical background. Then, the different data sets are presented followed by a theory section describing the deep learning architectures used in the thesis as well as some common evaluation metrics. Data pre-processing steps and ML models are presented in the method section, followed by obtained results. Lastly, a discussion is provided with further analysis of the results and possible application.

2

Aims

The aim of the thesis is to develop a machine learning algorithm for diagnosis of breast cancer using BUS images with the help of the following CNN approaches:

- A simple CNN.
- Transfer learning using the pre-trained convolutional neural networks InceptionV3, ResNet50V2, VGG19 and Xception.
- Deep features based on combinations of the pre-trained networks mentioned above.

3

Medical Background

3.1 Breast Cancer

The common factor for all cancer diseases is cells starting to grow uncontrollably, which can lead to the formation of a malignant tumor. A tumor does not have to be cancer, in that case it is called benign, but when a tumor is cancerous it is called malignant. The difference between benign and malignant is that malignant tumors can spread through the body [8].

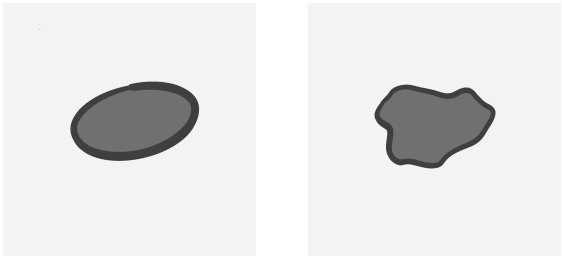
As of today breast cancer is the second most common type of cancer in Sweden. The cause of breast cancer are several, such as hereditary factors, lifestyle, and hormonal influence. The majority of breast cancer cases are, however, sporadic [1].

Different types of imaging can be used to detect breast cancer. Mammography, an x-ray method, is used in both screening and diagnostic imaging. Mammography can have a low sensitivity in dense breasts. Ultrasound is central in breast imaging as it can be used to further characterize masses detected on mammography, it does not expose the patient to radiation and is not detrimentally affected by breast density. In addition, ultrasound is a relatively cheap imaging method compared to other modalities [9, 10].

3.2 Characteristics of Malignant Tumors

To distinguish malignant tumors from benign, there are certain imaging features that in combination can be used to characterize the lesion. When analysing an ultrasound image of a tumor the features to take in consideration are shape, margin, orientation, echo pattern and posterior features [11].

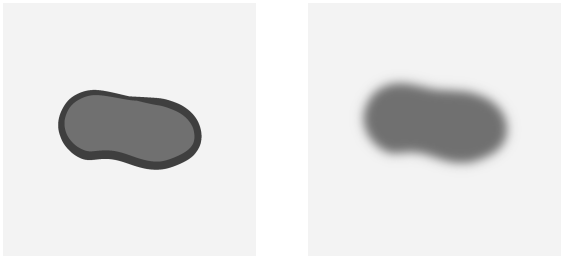
Shape
The shape of a tumor can be divided into round, oval and irregular. Visualizations of oval and irregular shapes are displayed in Figure 3.1.



(a) **(b)**

Figure 3.1: (a) Oval shape, (b) Irregular

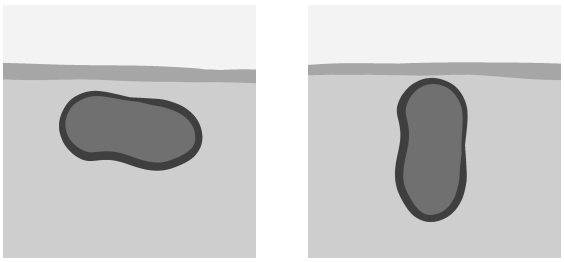
Margin
The margin of a tumor can be defined as circumscribed or not circumscribed. Figure 3.2 are displaying examples of circumscribed and not circumscribed tumors.



(a) (b)

Figure 3.2: (a) Circumscribed, (b) Not circumscribed

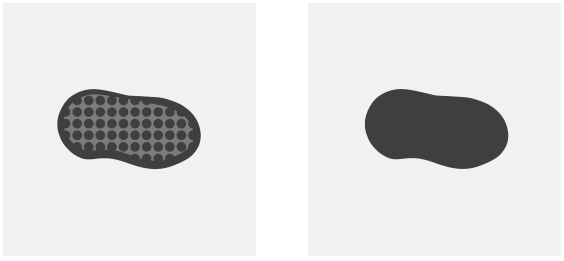
Orientation
The orientation of a tumor is defined as either parallel or not parallel to the skin. In Figure 3.3 examples of a tumor being parallel and not parallel to the skin is visualized.



(a) (b)

Figure 3.3: (a) Parallel to the skin, (b) Not parallel to the skin

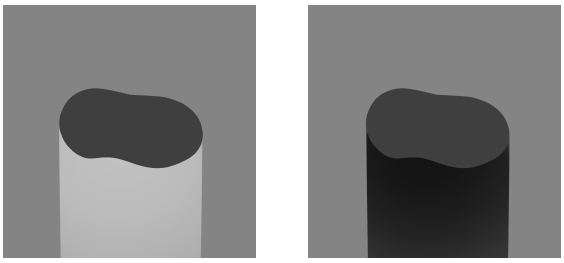
Echo Pattern
The echo pattern can in combination with other characteristics help indicate whether a tumor is cancerous or not. The echo patterns are appearing inside the tumor and depending on the type of tumor they look differently. Hence it is an effective characteristic to use as help to separate solid tumors from fluid filled lesions [12]. The echo patterns can be divided into echoic and anechoic, in other words echo is existing and not. Examples of an echoic tumor and an anechoic tumor are shown in Figure 3.4.



(a) (b)

Figure 3.4: (a) Echoic, (b) Anechoic

Posterior Features
Posterior features are a result of the attenuation characteristics of a tumor. It can be defined as either enhancement or shadowing. Enhancement is when the brightness in an ultrasound image is increased typically posterior to a cyst. Shadowing appears when the ultrasound beam is either completely absorbed or reflected in a structure which leads to a loss of information below the structure hence a shadow will appear [13], which is often the case posterior to dense malignant masses. Visualizations of shadowing and enhancement are displayed in Figure 3.5, which is often the case posterior to dense malignant masses.



(a) (b)

Figure 3.5: (a) Enhancement, (b) Shadowing

4

Data

The data used in the thesis consists of labeled BUSI data of three classes: malignant, benign and normal. The normal class includes images of normal tissue, i.e. no tumor. Two image data sets were used in this thesis, the first one collected by Unilabs Mammography Unit at Skåne University Hospital Sweden and the second one collected by Cairo University at Baheya Hospital, Egypt. The following section describes the data sets further.

4.1 The Swedish Data Set

The data set collected by Skåne University Hospital contains 293 images with class distribution according to Table 4.1. Since the images are unpreprocessed and collected using different machines, the actual ultrasound data are of varying quality and different shapes; for example the ultrasound data could be either rectangular or trapezoidal as depicted in Figure 4.1. Three example images from this data set can be seen in Figure 4.2. The collection of data for the project was approved by the Swedish Ethical Review Authority (2019-04607).

Table 4.1: Distribution of Swedish data set.

Class	Number of Images
Malignant	264
Benign	13
Normal	16
Total	293

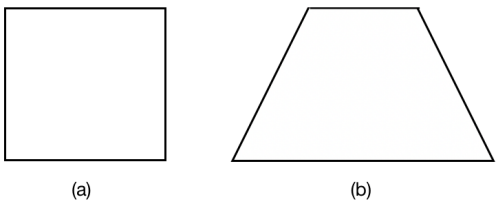


Figure 4.1: Ultrasound shapes used in the Swedish data set: (a) rectangular 2D, (b) trapezoidal 2D.

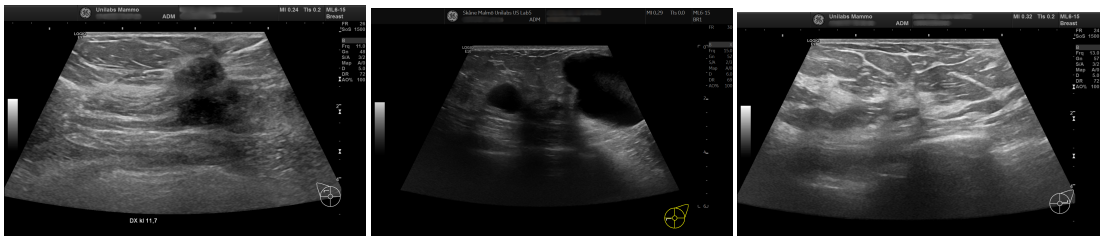


Figure 4.2: Three example images from the data set collected by Skåne University Hospital. From left: malignant, benign and normal.

4.2 The Egyptian Data Set

The second set of BUSI data is an open data set collected by Cairo University [7]. This set contains 780 images distributed according to Table 4.2. As apparent in Figure 4.3, the images have been pre-processed by cropping out unimportant information. The image *benign (82).png* was moved to the malignant class after consultation with Kristina Lång, Unilabs Mammography Unit at Skåne University Hospital.

Table 4.2: Distribution of the Egyptian data set.

Class	Number of Images
Malignant	210
Benign	487
Normal	133
Total	780

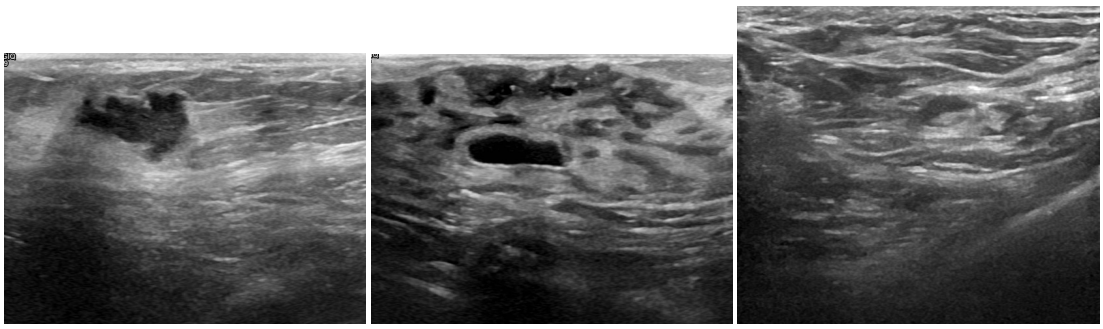


Figure 4.3: Three example images from the Egyptian data set. From left: malignant, benign and normal.

5

Artificial Neural Networks

Artificial Neural Networks (ANN) are designed in a way resembling the biological nervous system, consisting of many computational nodes often referred to as neurons. The nodes are arranged in fully connected layers; an input layer, an output layer and several hidden layers, as can be seen in the simple ANN model in Figure 5.1. The input data is usually the raw pixels of an image, that are passed on to the hidden layers which based on computations make decisions leading up to a final class score computed by the output layer. The computations performed in each node $h_1, ..., h_4$ are depicted in Figure 5.2.

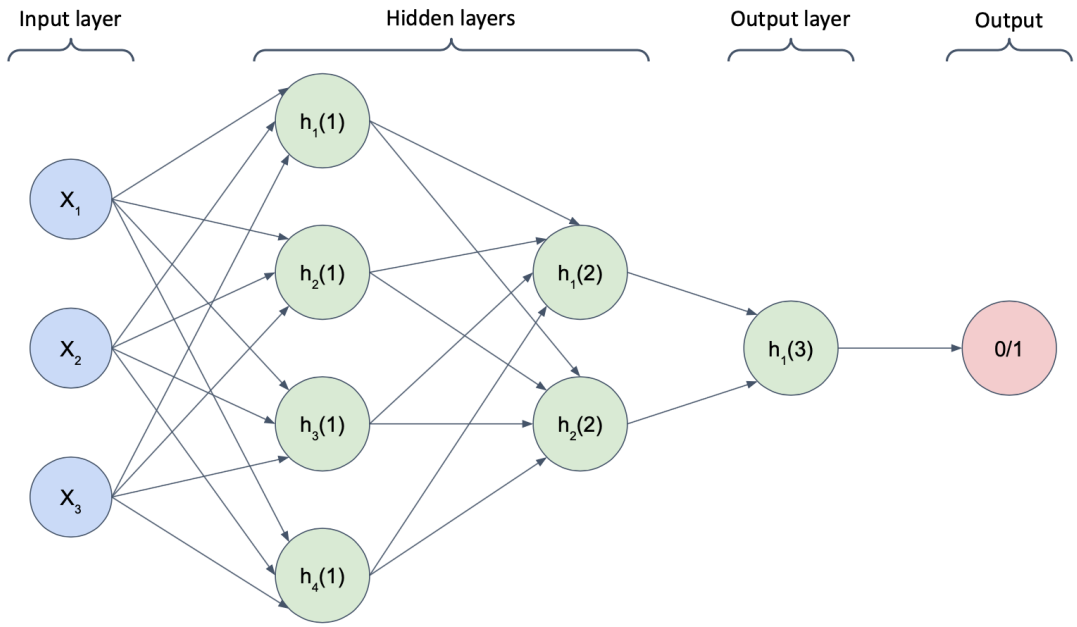


Figure 5.1: Simple binary ANN with two hidden layers where $x_1, ..., x_3$ are the inputs and $h_1, ..., h_4$ are the nodes. The nodes of an ANN take the output from all nodes in the previous layer as input and outputs to all nodes in the following layer, hence $h_1(1), ..., h_4(1)$ in the first hidden layer.

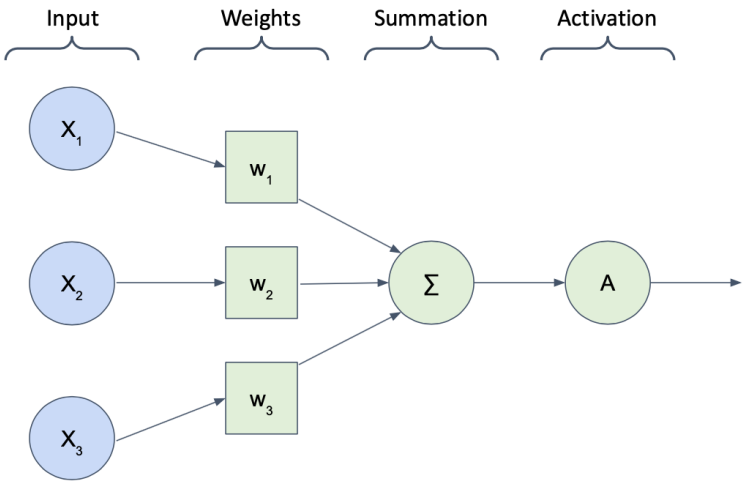


Figure 5.2: The computations performed in each node in Figure 5.1. Here $x_1, ..., x_3$ are the inputs to the node, which are weighted and summed. The last operation in a node is the activation function A, for example the ReLU activation (Equation 5.2) or the sigmoid activation (Equation 5.3).

5.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) are comprised of layered nodes as previously described for ANN. The hidden layers are mainly of the structures convolutional layers, pooling layers and fully connected layers, the latter usually followed by dropout and/or batch normalization layers. Below, each layer is described.

Convolutional Layer
The convolutional layer performs convolution of the image pixels with a moving window called the kernel, producing a 2D activation map for each channel. The mathematical definition of the discrete convolution in two dimensions is

$$(f * g)(x,y) = \sum_{i=-M/2}^{M/2} \sum_{j=-N/2}^{N/2} f(x-i, y-j) g(i,j), \tag{5.1}$$

where f is the image and g is the kernel of dimensions *(M,N)* centered at 0. The variables x and y define the pixel located at coordinates *(x,y)*. As the network is trained the convolutional layer will produce a kernel that gives maximum activation for a certain feature, i.e. peaks in the activation map.

The activation maps are then passed on to the rectified linear activation function (ReLU) which is defined according to

$$ReLU(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0. \end{cases} \tag{5.2}$$

The ReLU function behaves as a linear function but is in fact nonlinear. This property allows the network to learn complex nonlinear relationships in the data.

Unlike the fully connected ANN model, the nodes of the convolutional layer are only connected to a small section of the input called the receptive field. Since image data is usually large, this will speed up computations and make training more efficient.

Pooling Layer
The pooling layer has the 2D activation maps generated from each channel in the convolutional layer as input. It scales the dimensionality of the activation maps in order to reduce the number of parameters in the CNN model and thus reduce the computational complexity. In max pooling layers, this is achieved

by using the MAX-function and a small kernel, usually of size 2x2 moving over the input. In average pooling, the average of the kernel is pooled. The stride parameter determines the step size of the kernel.

Fully Connected Layer
Following several convolutional layers the final classification score is computed by the last fully connected layers. In a fully connected layer, all nodes are connected to the nodes of the two adjacent layers, hence the name. The output from the last convolutional layer is flattened and fed into the first fully connected layer which performs the ReLU activation function (see Equation 5.2) of the dot product between the flattened input and a weight matrix. The weight matrix is learned during training and used when the layer is called during classification. In a binary classification model, the very last fully connected layer has the sigmoid activation function

$$\text{sigmoid}(x) = \frac{1}{1+e^{-x}} \tag{5.3}$$

and an output of size 1. The output is the final classification score, i.e. a value between 0 and 1 where values ≤ 0.5 are more likely to belong to class 0 and values > 0.5 are more likely to belong to class 1.

Dropout Layer
Overfitting is a common problem when training a deep neural network on a limited data set. To avoid overfitting, dropout layers can be added to the model. As the name implies, the layer randomly drops a number of nodes, usually around 50-80% following a fully connected layer. This results in less co-adaption between nodes, which is a common cause of overfitting.

Batch Normalization
Batch normalization is another approach to combat overfitting. During training, the layer normalizes the input using the mean and variance of the current batch of inputs. This results in an output with a mean close to 0 and a standard deviation close to 1. The normalization during training is performed according to

$$\text{batch\_normalization}(batch) = \gamma \frac{batch - \text{mean}(batch)}{\sqrt{\text{var}(batch) + \varepsilon}} + \beta, \tag{5.4}$$

where γ and β are constants which are learned during training and initialized to 1 and 0 respectively. The variable ε is initialized to 0.001 in order to avoid division with zero. This step is performed on every channel of the input batch.

When using the trained model to classify data, the layer instead performs normalization of the input using the learned variables γ and β according to

$$\text{batch\_normalization}(batch) = \gamma \frac{batch - \text{moving\_mean}}{\sqrt{\text{moving\_var} + \varepsilon}} + \beta, \tag{5.5}$$

where moving_mean and moving_var are a moving average of the mean and variance. These variables have been updated each time the layer has been called during training according to

$$\begin{aligned} \text{moving\_mean} &= \text{moving\_mean} * momentum + \text{mean}(batch) * (1 - momentum) \\ \text{moving\_var} &= \text{moving\_var} * momentum + \text{var}(batch) * (1 - momentum) \end{aligned} \tag{5.6}$$

where *momentum* is a configurable constant with default value 0.99 in Python Keras library. Batch normalization is usually placed after a convolutional or fully connected layer before the activation function.

5.2 Training of a CNN

Prior to using a CNN model for classification the CNN has to be trained. This means finding the best weights for optimal classification. Since there are often thousands, if not millions, unknown weights in

a CNN, the optimal set of weights cannot be directly optimized. This calls for an optimization algorithm to navigate through possible set of weights. Below follow some important functions and parameters used when training a CNN.

Optimization Function
The most commonly used optimization algorithm is the Stochastic Gradient Descent (SGD). In this thesis a version of SGD will be used, called the Adam optimization algorithm. Adam is a popular choice since it adapts the learning rate individually for each parameter based on estimates of the first and second order momentum of the gradient. The method is computationally efficient and requires little memory. For further reading about the algorithm see "Adam: A Method For Stochastic Optimization" [14].

Training Set
The data set used for updating of the weights during training is called the training set.

Validation Set
In order to visualize how the model is improving during training, a validation set is used. The point of the validation set is to evaluate the model independently of the training set. The validation set is used during the development of an algorithm and should therefor not be considered a part of the final testing.

Test Set
The test set is the final evaluation set, completely independent of the training process.

Loss Function
Loss is a measure of how close the classified predictions are to the true predictions. The lower loss, the more accurate is the model. In binary classification the loss function binary cross-entropy is used, which is defined as the average

$$\text{Loss} = -\frac{1}{\text{n}}\sum_{i=1}^{\text{n}} y_i \cdot \log \hat{y}_i + (1-y_i)\cdot \log(1-\hat{y}_i), \tag{5.7}$$

where n is the number of samples, $\hat{y}_i$ is the output classification for sample i and y_i is the corresponding ground truth.

Batch Size
The batch size is the number of images that will propagate through the training at the same time. Usually the batch size is chosen as a power of 2 such as 16, 32, ... , 256 in order to fit the GPU. At the end of a batch, the weights of the network are updated. A smaller batch size converges quickly with a trade-off in accuracy due to noise. A larger batch size usually results in slower convergence but better model.

Epoch
When all training data have propagated through the training an epoch is complete. The number of epochs are chosen so that the weights have converged when training is finished.

5.3 Transfer Learning

Transfer learning is a way of reusing pre-trained networks on new tasks. The method is time efficient since there is no need to develop and train a complete model from scratch. To adapt a pre-trained network to a new task the last fully connected layer is cut off and a suiting ANN network is added to the model. The model can be additionally fine-tuned by unfreezing the last convolutional blocks of the pre-trained base during training with the new data set.

The authors of "Comparison of Transferred Deep Neural Networks in Ultrasonic Breast Masses Discrimination" suggest the use of transfer learning for classification of breast ultrasound and achieved

good results using the pre-trained networks ResNet50, InceptionV3 and Xception [15]. In this section the three networks will be presented with the addition of a fourth network, VGG19. All four networks are pre-trained on the database Imagenet, which consists of over 14 million labeled images.

ResNet50V2
ResNet50V2 is an updated version of ResNet50 which is comprised of 50 layers divided into 5 stages, each consisting of a convolutional block and a residual block. Residual blocks are often referred to as identity blocks and their main function is to fast-forward output to a deeper layer. This architecture allows residual networks to have a large amount of layers without an increase in training error which is common in traditional convolutional neural networks due to vanishing gradient [16, 17]. ResNet50 is a smaller version of ResNet152, the winner of ImageNet Challenge in 2015.

InceptionV3
InceptionV3 is 48 layers deep and is built up by three inception modules. The idea behind the inception module is to not have to choose between 1x1, 3x3 and 5x5 convolutions. The result from all three convolutions and a pooling-layer are concatenated and fed into the next module. With the use of factorizing convolutions, a complete inception module has less computational complexity than a single 3x3 or 5x5 convolutional layer [18].

Xception
Xception stands for Extreme Inception and is an extension of the Inception network. The network is 71 layers deep and is based on depthwise separable convolutions which are performed in two steps:

(a) Depthwise convolution, i.e. channel-wise convolution with a kernel of size nxn.

(b) Pointwise convolution, which is equivalent to regular convolution with a kernel of size 1x1.

The addition of depthwise separable convolutions outperforms the previously described Inception algorithm with the same number of parameters [19].

VGG19
VGG19 is a traditional convolutional neural network consisting of 19 layers out of which 16 convolutional layers with kernel size 3x3 and three fully connected layers [20].

5.4 Deep Features

A deep feature algorithm can be constructed using two or more of the pre-trained transfer networks. The algorithm consists of two parts: (a) feature extraction (b) an ANN.

(a) An appropriate number of layers are cut off the pre-trained networks including the last softmax layer. The last layer is preferably a flattening layer or the output from a fully connected layer. Features are then extracted from the training-, validation- and test set by simply passing the data through the shortened pre-trained networks, concatenating the results into a large feature vector.

(b) Lastly, a simple ANN for classification is constructed and trained on the feature vectors of the training set. To evaluate the performance, the feature vectors of the test set are fed to the trained ANN.

5.5 Evaluation Metrics

To evaluate the performance of a machine learning algorithm different types of metrics can be useful. In the following section the metrics used in this thesis will be explained.

Sensitivity and Specificity
To measure the sensitivity and specificity the expressions true positive (TP), true negative (TN), false positive (FP) and false negative (FN) are needed. One way of illustrating these terms and their relations to each other is the confusion matrix. Figure 5.3 is showing an illustration of a confusion matrix.

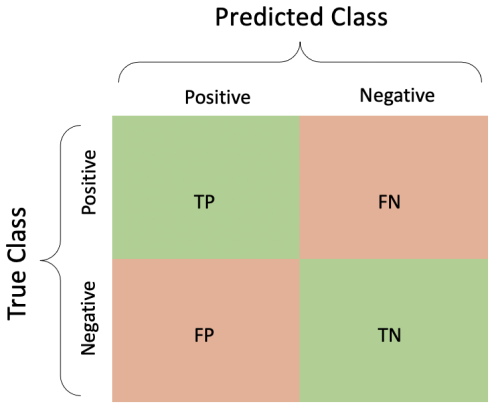


Figure 5.3: Confusion matrix displaying the relation between true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN).

Here, malignant is the positive class and benign is the negative class. Hence a true positive is a malignant tumor classified as malignant, and a false positive is a benign tumor classified as malignant.

Sensitivity (true positive rate) is the probability that an instance with the classification positive actually gets classified as positive. In other words, it is a measure of how many malignant tumors that actually gets classified as malignant. Sensitivity is defined according to

$$Sensitivity = \frac{TP}{TP+FN}. \tag{5.8}$$

It can also be valuable to measure the specificity (true negative rate). The specificity is the probability that an instance with the classification negative truly is negative i.e. how many benign tumors get classified as benign. The specificity is defined as

$$Specificity = \frac{TN}{TN+FP}. \tag{5.9}$$

ROC Curve and AUC
The receiver operating characteristic (ROC) curve illustrates the performance of a classification model at all possible thresholds. Threshold is the value that decides if an instance should belong to the negative or the positive class. The ROC curve consists of $1-$ specificity (false positive rate) at the X-axis and sensitivity (true positive rate) on the Y-axis. The ideal is to have a sensitivity of one and $1-$ specificity equal to zero, hence the ideal position in a ROC plot is the upper left corner. An example of ROC curve can be found in Figure 5.4.

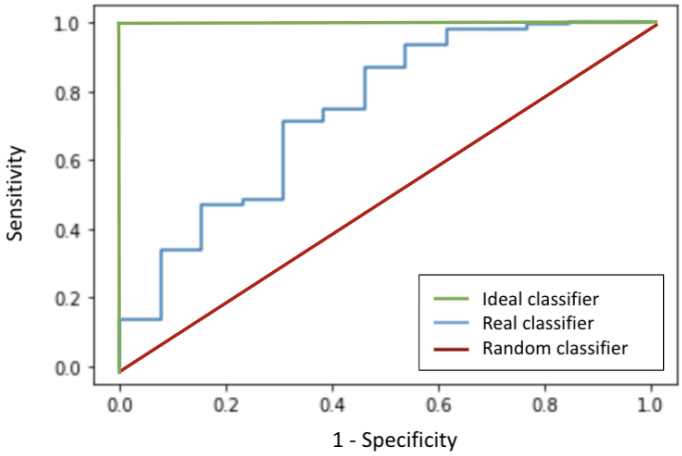


Figure 5.4: ROC plot with three curves: ideal, real and random classifier

Area under the curve (AUC) is a metric that can be determined from the ROC curve where a larger area under the curve implies a better classifier. The AUC is a value in the range 0 to 1, where 1 corresponds to when a 100% of the predictions are correct.

Accuracy

Accuracy is a measurement of how often the predicted class is equal to the real class. The number of times a predicted class is matching the real class is divided by the number of all predictions, i.e.

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN}, \tag{5.10}$$

where TP, FP, TN and FP is explained in Figure 5.3. This equation results in a number between 0 and 1, where 0 is no matches and 1 is when all predictions are correct.
If there is an imbalance between the classes, the accuracy might give a misleading number. In that case, the weighted accuracy can be used. As mentioned earlier, sensitivity is the probability that an instance labeled as positive actually gets predicted as positive. Specificity works in the same way but with respect to the negative class. Hence the weighted accuracy can be formulated as

$$Weighted\ Accuracy = Sensitivity \times weight + Specificity \times (1 - weight), \tag{5.11}$$

where the term weight is used to make sure both classes has equal impact on the accuracy in case of imbalance between the classes. For equally important classes the weight is set to 0.5.

Training and Validation Accuracy and Loss

For each epoch during training, it is a good idea to evaluate the model by letting it predict both the training set and the validation set. The accuracy and loss can then be plotted against number of epochs. By analyzing the plots, it can be determined whether the model is overfitted or not and whether the model is actually learning something during the training. For an overfitted model, the accuracy of the training set is increasing a lot faster than the accuracy of the validation set. Also, the loss plot will show a slower decrease for the validation data if the model is overfitted. For an ideal model, the plotted data from the validation data and training data should be overlapping, meaning the performance is equally good on both data sets.

Grad-CAM

In "Grad-CAM: Visual explanations from Deep Networks via Gradient Based Localization" by Selvaraju et al., the authors describe how there usually exists a trade-off between accuracy and interpretability [21]. Simple networks that are easy to understand, usually perform worse. When a complicated CNN fails, it is difficult to understand where and why. Gradient-weighted Class Activation Mapping (Grad-CAM) can be used to analyze such networks.

Using the Grad-CAM technique, it is possible to produce a heatmap over the classified image where high values indicate parts of the image that contribute to the network's decision. A heatmap is constructed given an input image and the top predicted class using the following formula:

(a) The gradient of the score for the given class with respect to the feature maps from the last convolutional layer is calculated.

(b) Then, a linear combination is performed between the obtained weights and the feature maps.

(c) Lastly, a ReLU (Equation 5.2) is applied to the linear combination in order to get rid of negative values, as negative values are more likely to belong to other classes.

The result is a heatmap highlighting parts in the image that contributed to the model's classification [21].

6

Method

6.1 Class Definition

Both the Swedish data and the Egyptian data contained three classes: benign, normal and malignant. For the algorithm it was chosen to narrow the problem down to only two classes. Class 0 being benign and class 1 being malignant. The normal class was excluded completely.

6.2 Data Split

The Egyptian image data set was for each class randomly split into 80% training, 10% validation and 10% test. Since the Swedish data set is imbalanced, it cannot effectively be used as training data. Therefore, the complete data set was used for evaluation.

6.3 Crop Images

In order to use the Swedish images some pre-processing steps had to be done. The data set contained images with prints such as patient information, dates and markers, also the ultrasound part of the image had different shapes. With such prints in the images, there is a risk of the ML algorithm to look at this information opposed to the actual ultra sound data. A MatLab script to crop the data properly was implemented. The cropped images had a rectangular shape resembling the Egyptian data set.

6.4 Augmentation

Using the image data augmentation tool in Keras, the Egyptian training-set was expanded. Performing augmentation of BUSI data, it is important to not alter diagnostic characteristics of the images such as rotation and too aggressive horizontal- and vertical shifts. The augmentation step was implemented using horizontal-shift, vertical-shift, shear and zoom, all in the range of 10%. The data was rescaled appropriately to fit each model according to Table 6.1. An additional pre-processing function was implemented to add noise in order to make the Egyptian training set similar to the Swedish images. Examples of augmented images can be seen in Figure 6.1.

Table 6.1: Input sizes of the simple CNN and the pre-trained networks

Model	Input size
Simple CNN	299x299
InceptionV3	299x299
ResNet50V2	224x224
VGG16	224x224
Xception	299x299

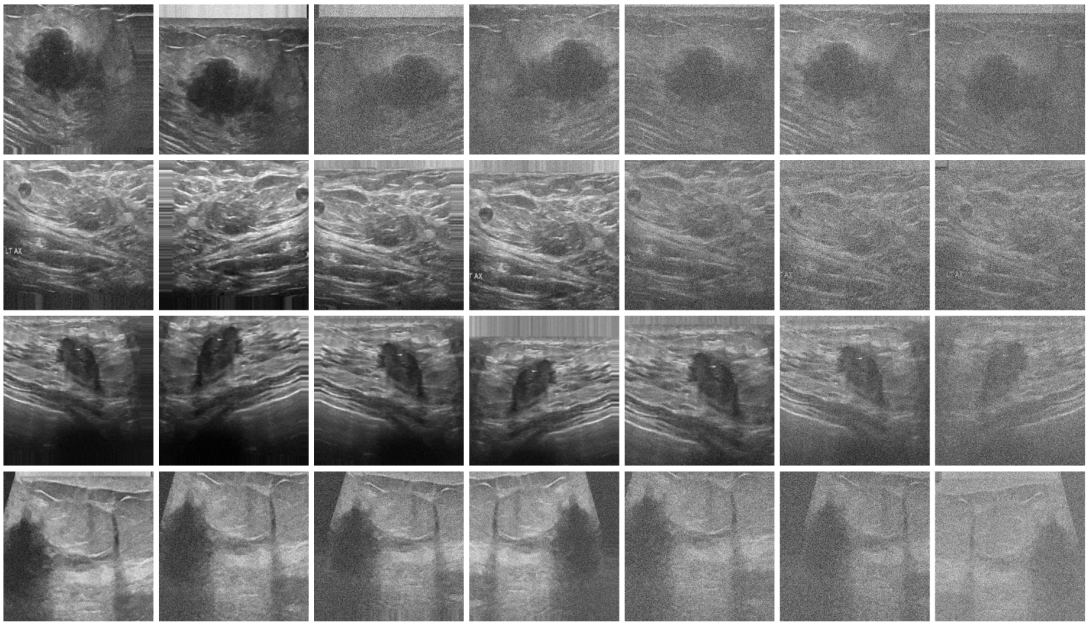


Figure 6.1: Examples of augmented images. Each row displays some random augmentations of an image.

6.5 Machine Learning Algorithms

Some of the training parameters were set to the same values for all of the networks that were used. These parameters can be seen in Table 6.2-6.3.

Table 6.2: Training parameters for Simple CNN and transfer models.

Batch size	32
Epochs	25
Learning rate	$2 \cdot 10^{-5}$

Table 6.3: Training parameters for deep feature models.

Batch size	32
Epochs	35
Learning rate	$2 \cdot 10^{-5}$

6.5.1 Simple CNN

A simple CNN was created in Python using Keras according to Figure 6.2. The network consisted of six convolutional layers, the first three layers with output sizes of 32 and the last three with the sizes of 64, 128 and 256. The kernel size was set to 3x3 for all of the convolutional layers and three channels were used. Before each activation, batch normalization was performed. The activation was then followed by a maxpooling layer with a kernel size of 2x2 and stride 2. Following the convolutional part two fully connected layers of size 1024 were implemented. The last layer had the sigmoid activation function and from this layer the output were either benign (0) or malignant (1).

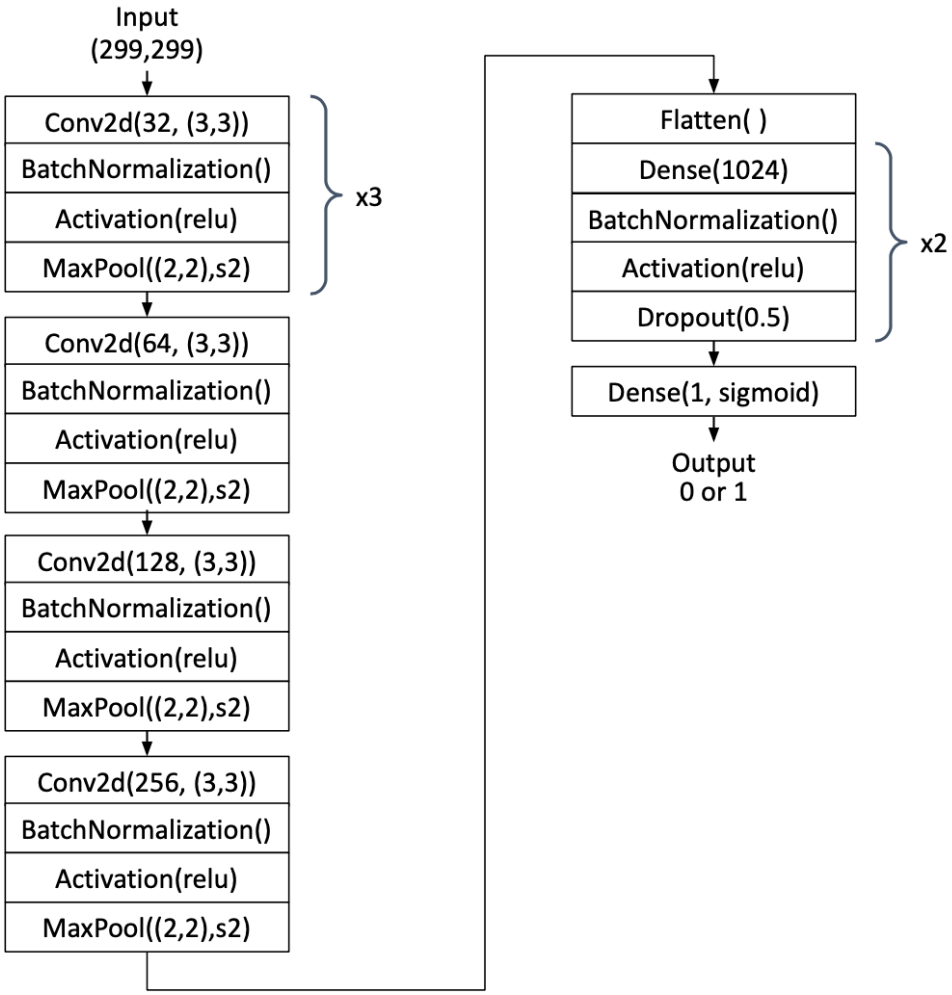


Figure 6.2: Architecture of Simple CNN.

6.5.2 Transfer Learning

The transfer networks were implemented in Python using Keras. All pre-trained networks were imported from Keras Applications using the Imagenet weights. The idea behind transfer learning is to use the pre-trained network as a convolutional base and then add some additional fully connected layers including a last classification layer with the sigmoid activation. The majority of the base will be frozen during training and the last 1-2 convolutional blocks will be fine tuned to the new data set. In some cases, classification can improve if layers of the pre-trained base are cut-off. Where to cut and freeze the network differ from case to case and was experimentally determined. Below follows a presentation of each transfer network.

ResNet50V2

The input size of ResNet50V2 is 224x224. All layers of the base were used in the final transfer network and the complete base was frozen during training. Two fully connected layers of size 2048 and 1024 respectively were added, each with batch normalization prior the ReLU activation followed by a dropout layer of 50%. The output layer had a sigmoid activation to fit the binary classification problem. Figure 6.3 displays a schematic of the network.

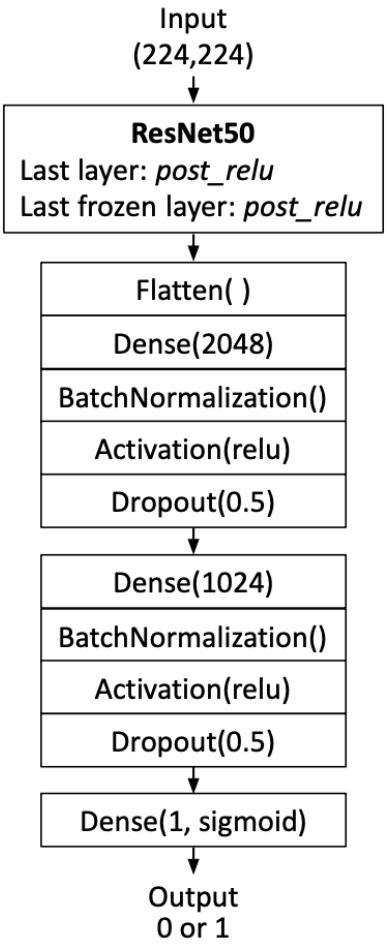


Figure 6.3: Schematic of the layers in the transfer network based on ResNet50V2.

Xception

The input size of Xception is 299x299. All layers of Xception were used with the last convolutional block unfrozen during training. An additional fully connected layer of size 1024 was added to the model along with batch normalization prior the activation function followed by a dropout layer of 50%. The output layer had the sigmoid activation function. Figure 6.4 displays a schematic of the Xception transfer network.

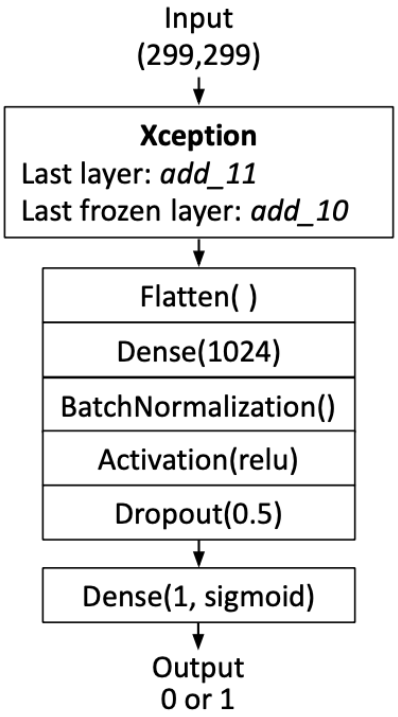


Figure 6.4: Schematic of the layers in the transfer network based on Xception.

InceptionV3

The input size of InceptionV3 is 299x299. The last Inception module was cut off, leaving the model with last layer *mixed10*. The two last inception modules were unfrozen during training. Five additional fully connected layers of size 1024 was added to the model along with batch normalization prior the activation function followed by a dropout layer of 50%. The output layer had the sigmoid activation function. Figure 6.5 displays a schematic of the InceptionV3 transfer network.

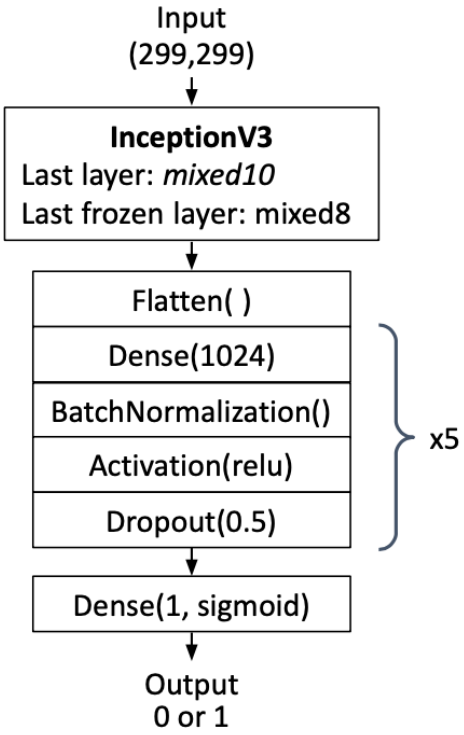


Figure 6.5: Schematic of the layers in the transfer network based on InceptionV3.

VGG19

The input size of VGG19 is 244x244. All layers of VGG19 were used with the last convolutional block unfrozen during training. Two additional fully connected layer of size 1024 was added to the model along with batch normalization prior the activation function followed by a dropout layer of 50%. The output layer had the sigmoid activation function. Figure 6.6 displays a schematic of the VGG19 transfer network.

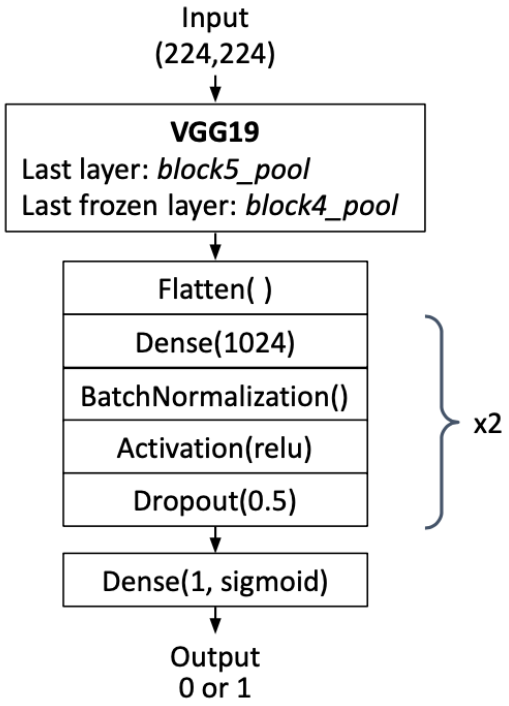


Figure 6.6: Schematic of the layers in the transfer network based on VGG19.

6.5.3 Deep Features

All possible deep feature models were constructed using the four trained transfer models generated in section 6.5.2 according to Figure 6.7. All transfer networks were combined into a total of 11 different deep feature networks. As shown in the schematic, the output from the last dense layer of size 1024 in each of n models were concatenated into a large feature vector of size $n \cdot 1024$. The feature vectors were fed into an ANN consisting of 4 fully connected layers, each of size 1024. The last layer had the sigmoid activation function which outputs the classification.

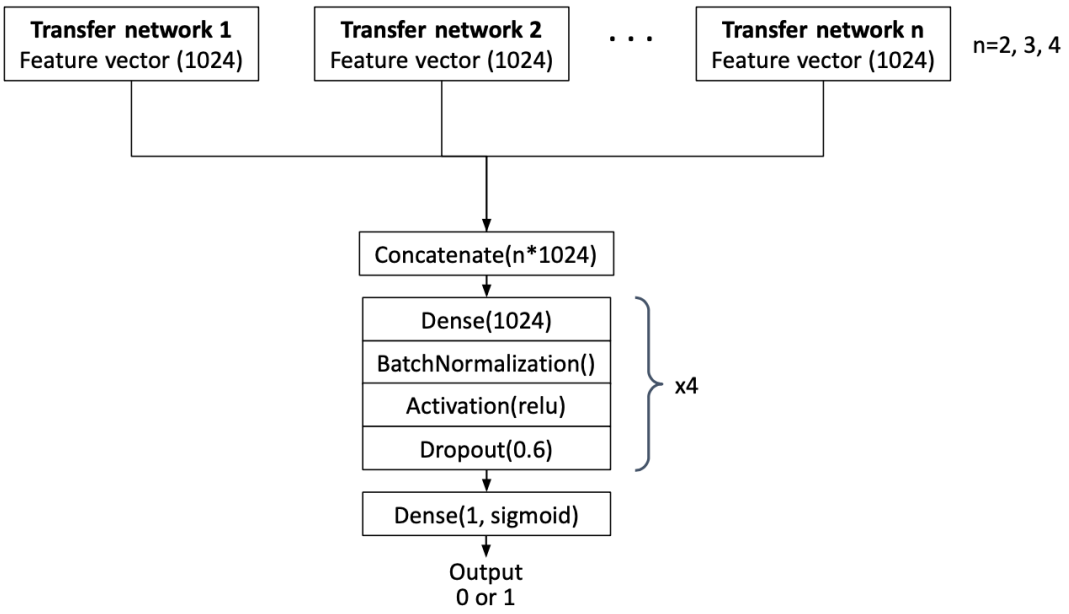


Figure 6.7: Architecture of deep feature networks.

6.6 Evaluation

After training each model an evaluation was performed. The first step was to compare the accuracy and loss of the training data to the accuracy and loss of the validation data for each epoch. This was done by generating two plots; one displaying the accuracy, and one displaying the loss against the number of epochs.

The next step was letting the model predict the Swedish data set and the Egyptian test set. The following procedure was performed on each of the two different test sets:

(a) The predicted labels and true labels obtained from a test set was used to generate a confusion matrix. This was done with help of the confusion matrix function in the Scikit-Learn library. For all models the threshold was initially set to 0.5.

(b) The next step was to generate confusion matrices for some different thresholds. Here, thresholds between 0 and 1 with a step size of 0.1 were used. The goal was to find a threshold where the two classes had as equally good performance as possible with the condition that the sensitivity was better than the specificity rather than the other way around. The confusion matrix for the optimal threshold was then used to obtain the final specificity, sensitivity and weighted accuracy according to Equation 5.9, 5.8 and 5.11.

(c) Finally, a ROC curve and an AUC score (see Section 5.5) was generated for the model using the ROC curve and AUC functions from the Scikit-learn library.

The best model was chosen based on primarily AUC score and secondly sensitivity. The decisions of the final transfer networks were analyzed using Grad-CAM according to Section 5.5. Heatmaps were produced for all images of the Swedish test set.

7

Results

All models were evaluated on the two test sets: the random test set from the Egyptian data set and the complete Swedish data set. The classification threshold was tweaked so that the optimal result was obtained. In the following section, the optimal threshold together with the performance measurements specificity, sensitivity, weighted accuracy and AUC are presented for each model.

7.1 Simple CNN

Below follows the training and validation plots for the simple CNN. Figure 7.1 is displaying the accuracy and Figure 7.2 is displaying the loss. This is followed by the result obtained when evaluating the simple CNN on the Egyptian test set (Table 7.1) and the complete Swedish data set (Table 7.2).

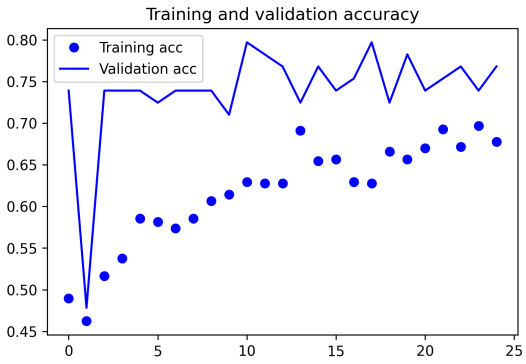


Figure 7.1: Training and validation accuracy for simple CNN.

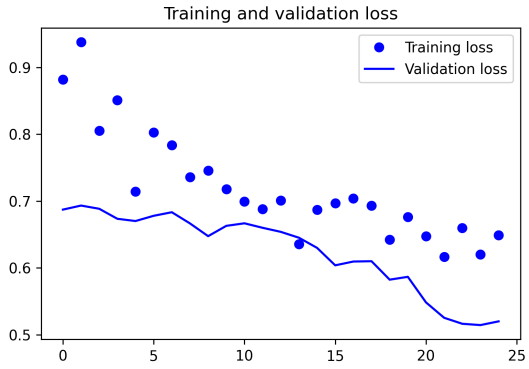


Figure 7.2: Training and validation loss for simple CNN.

Table 7.1: Performance of the Simple CNN on the Egyptian test set.

Model	Threshold	Specificity	Sensitivity	Accuracy	AUC
Simple CNN	0.3	56.10%	82.61%	69.35%	0.76

Table 7.2: Performance of the Simple CNN on the Swedish test set.

Model	Threshold	Specificity	Sensitivity	Accuracy	AUC
Simple CNN	0.3	38.46%	84.15%	61.31%	0.63

7.2 Transfer Learning

In the following section training and validation plots for the transfer models are displayed in Figure 7.3 to Figure 7.10. Further the result obtained for the transfer models based on InceptionV3, ResNet50V2, VGG19 and Xception is presented. The performance on the Egyptian test set and Swedish data set is displayed in Table 7.3 and Table 7.4 respectively.

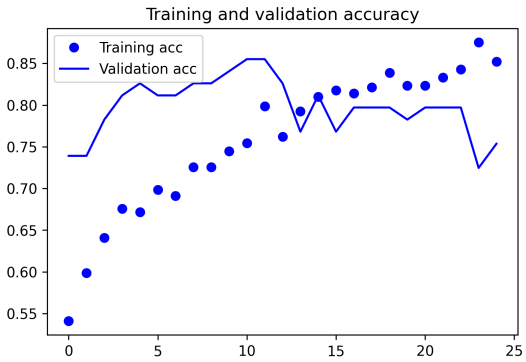


Figure 7.3: Training and validation accuracy for transfer model based on InceptionV3.

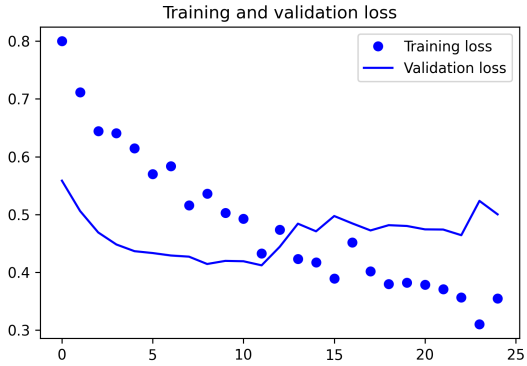


Figure 7.4: Training and validation loss for transfer model based on InceptionV3.

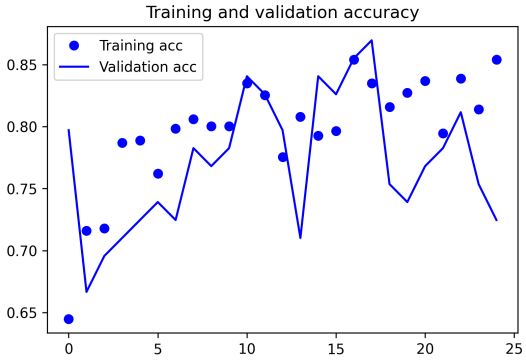


Figure 7.5: Training and validation accuracy for transfer model based on ResNet50V2.

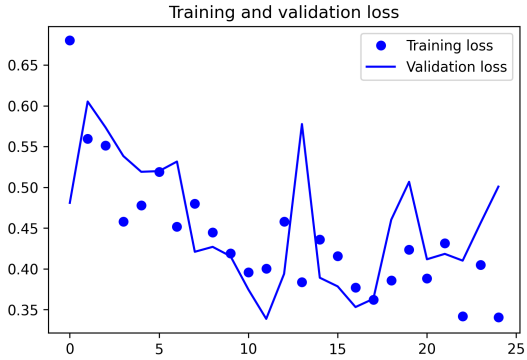


Figure 7.6: Training and validation loss for transfer model based on ResNet50V2.

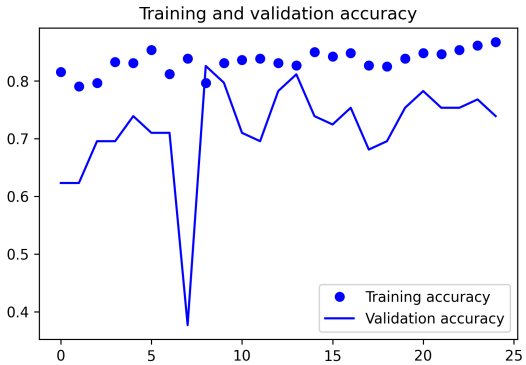


Figure 7.7: Training and validation accuracy for transfer model based on VGG19.

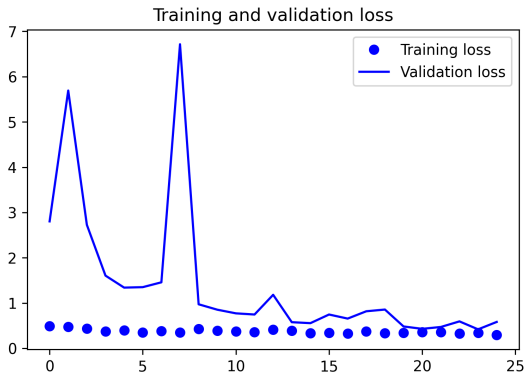


Figure 7.8: Training and validation loss for transfer model based on VGG19.

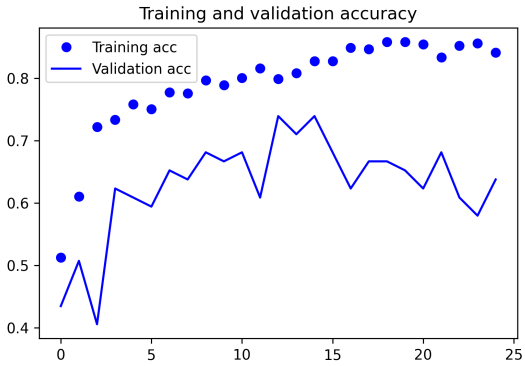


Figure 7.9: Training and validation accuracy for transfer model based on Xception.

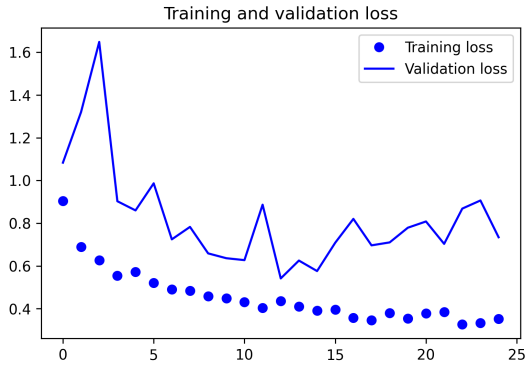


Figure 7.10: Training and validation loss for transfer model based on Xception.

Table 7.3: Performance of transfer models on the Egyptian test set.

Transfer model	Threshold	Specificity	Sensitivity	Accuracy	AUC
InceptionV3	0.4	80.49%	82.61%	81.55%	0.90
ResNet50V2	0.2	78.05%	82.61%	80.33%	0.91
VGG19	0.1	82.93%	82.61%	82.77%	0.89
Xception	0.5	80.49%	86.96%	83.72%	0.92

Table 7.4: Performance of transfer models on the Swedish data set.

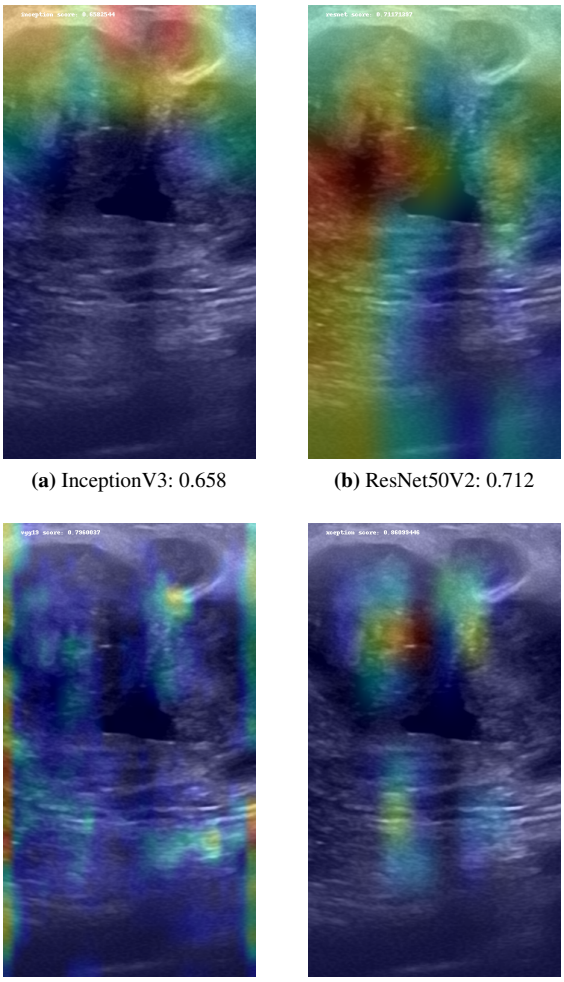
Transfer model	Threshold	Specificity	Sensitivity	Accuracy	AUC
InceptionV3	0.5	61.54%	75.09%	68.32%	0.73
ResNet50V2	0.4	69.23%	75.85%	72.54%	0.81
VGG19	0.4	69.23%	74.34%	71.79%	0.77
Xception	0.4	61.54%	80.00%	70.77%	0.75

7.2.1 Grad-CAM

In the following section a selection of images from the Swedish data set are displayed along with their respective heatmaps from all four models. The heatmaps show which parts of the image that are more likely to be malignant. Red areas equals high heatmap value, i.e. areas in the image that contributes to high classification score. Blue areas are closer to zero. For visualization purpose, all heatmaps are individually normalized with respect to the maximum value, i.e. the scale of the heatmap is not the same in every image.

Images Misclassified by All Models

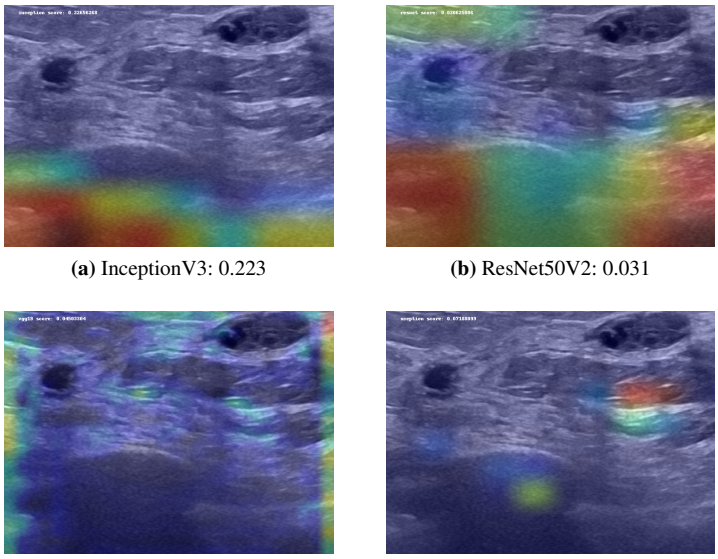
In total 16 images (out of which 1 benign and 15 malignant) from the Swedish data set were misclassified by all four transfer models. Below, two of them are displayed with their respective heatmaps and corresponding classification scores.



(a) InceptionV3: 0.658 **(b)** ResNet50V2: 0.712

(c) VGG19: 0.796 **(d)** Xception: 0.861

Figure 7.11: Benign image classified as malignant by all transfer models. Red areas contribute to higher classification score.



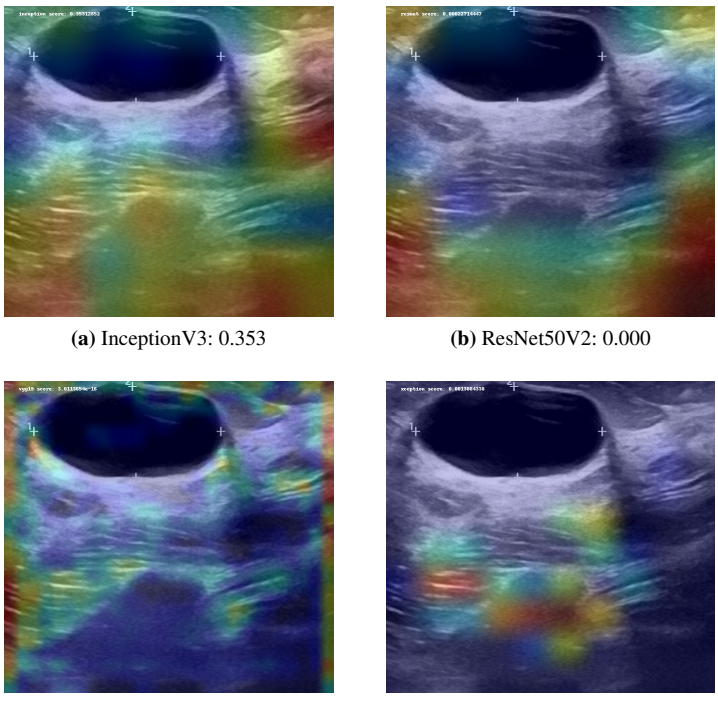
(a) InceptionV3: 0.223 **(b)** ResNet50V2: 0.031

(c) VGG19: 0.045 **(d)** Xception: 0.072

Figure 7.12: Malignant image classified as benign by all transfer models. Red areas contribute to higher classification score.

Images Correctly Classified by All Models

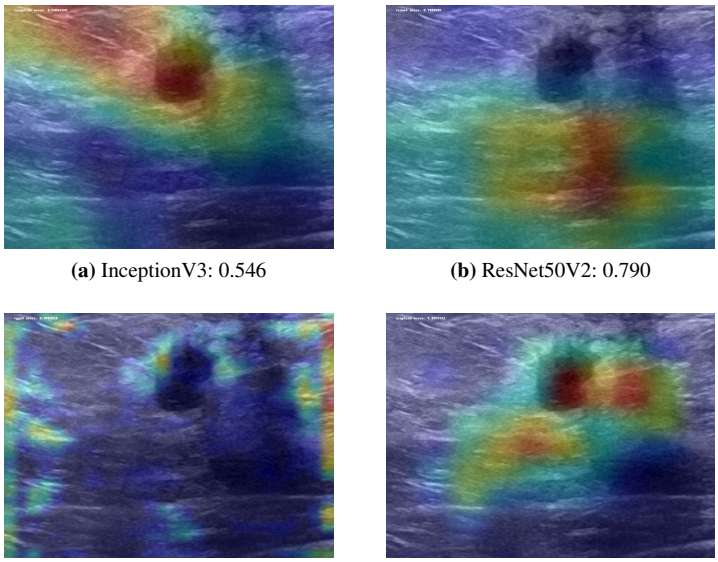
In total 142 images (out of which 6 benign and 136 malignant) from the Swedish data set were correctly classified by all four transfer models. Below, three of them are displayed with their respective heatmaps and classification scores.



(a) InceptionV3: 0.353 **(b)** ResNet50V2: 0.000

(c) VGG19: 0.000 **(d)** Xception: 0.001

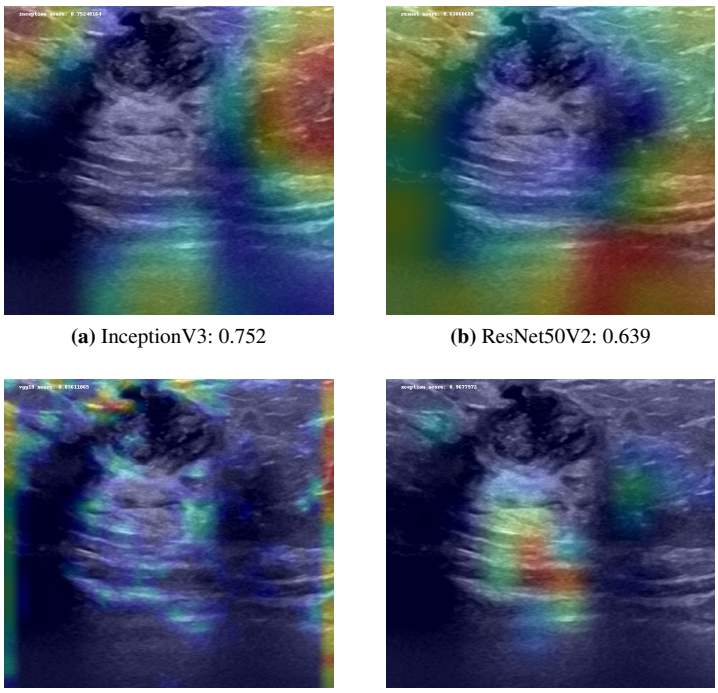
Figure 7.13: Benign image classified as benign by all transfer models. Red areas contribute to higher classification score.



(a) InceptionV3: 0.546 **(b)** ResNet50V2: 0.790

(c) VGG19: 0.988 **(d)** Xception: 0.998

Figure 7.14: Malignant image classified as malignant by all transfer models. Red areas contribute to higher classification score.



(a) InceptionV3: 0.752 **(b)** ResNet50V2: 0.639

(c) VGG19: 0.836 **(d)** Xception: 0.968

Figure 7.15: Malignant image classified as malignant by all transfer models. Red areas contribute to higher classification score.

7.3 Deep Features

Below follows the result obtained when evaluating the different combinations of transfer models on the random Egyptian test set (Table 7.5) and the complete Swedish data set (Table 7.6). All plots for training and validation accuracy and loss can be found in Appendix.

Table 7.5: Performance of deep feature models on the Egyptian data set. The highlighted row contains the scores for the best model.

Deep feature model	Threshold	Specificity	Sensitivity	Accuracy	AUC
InceptionV3 & ResNet50V2	0.4	82.93%	86.96%	84.94%	0.91
InceptionV3 & Xception	0.1	53.66%	95.65%	74.66%	0.91
InceptionV3 & VGG19	0.4	85.37%	86.96%	86.16%	0.92
ResNet50V2 & Xception	0.2	60.98%	91.30%	76.14%	0.90
ResNet50V2 & VGG19	0.3	78.05%	100%	89.02%	0.92
Xception & VGG19	0.3	80.49%	82.61%	81.55%	0.92
InceptionV3, ResNet50V2 & Xception	0.3	65.85%	91.30%	78.58%	0.93
InceptionV3, ResNet50V2 & VGG19	0.3	78.05%	95.65%	86.85%	0.93
ResNet50V2, Xception & VGG19	0.4	80.49%	86.96%	83.72%	0.92
InceptionV3, Xception & VGG19	**0.3**	**82.93%**	**95.65%**	**89.29%**	**0.93**
InceptionV3, ResNet50V2, Xception & VGG19	0.3	80.49%	95.65%	88.07%	0.91

Table 7.6: Performance of deep feature models on the Swedish data set.

Deep feature model	Threshold	Specificity	Sensitivity	Accuracy	AUC
InceptionV3 & ResNet50V2	0.4	53.85%	82.26%	68.06%	0.79
InceptionV3 & Xception	0.2	61.54%	79.25%	70.39%	0.77
InceptionV3 & VGG19	0.4	61.54%	81.13%	71.34%	0.78
ResNet50V2 & Xception	0.3	61.54%	79.62%	70.58%	0.79
ResNet50V2 & VGG19	0.4	53.85%	83.02%	68.43%	0.79
Xception & VGG19	0.4	53.85%	77.74%	65.79%	0.79
InceptionV3, ResNet50V2 & Xception	0.4	53.85%	80.75%	67.30%	0.78
InceptionV3, ResNet50V2 & VGG19	0.5	61.54%	76.98%	69.26%	0.81
ResNet50V2, Xception & VGG19	0.5	69.23%	77.74%	73.48%	0.79
InceptionV3, Xception & VGG19	0.4	61.54%	80.75%	71.15%	0.79
InceptionV3, ResNet50V2, Xception & VGG19	0.5	53.85%	81.89%	67.87%	0.82

8

Discussion

8.1 Class Definition

Initially, three classes were used: normal tissue, benign tumor and malignant tumor. For simplicity, the problem was reformulated into a binary classification problem where class 0 was non-cancer and class 1 cancer, i.e. the images belonging to the classes normal and benign tumor where combined into one class. Using the new class definition it was noted that a large amount of benign images were classified as cancerous. We suspected that the models looked at whether the image contained a tumor or not. To avoid this problem and force the models to actually look at the characteristics of the tumor, all images belonging to the normal tissue class were removed. The final class definition gave a much clearer result of whether the models could identify cancer or not.

8.2 Simple CNN

8.2.1 Architecture

The idea behind the implementation of the CNN was to keep it simple by not adding too many convolutional layers. Initially only three convolutional layers were used. This idea somewhat worked for the Egyptian test set (approximately 60% accuracy) but did not work at all for the Swedish data set (<20% accuracy). To give the simple CNN a chance on the Swedish data set, more convolutional layers were added. Adding more convolutional layers adds weights to the model, making the model more complex. This results in a more general model with the risk of overfitting. The final CNN model is displayed in Figure 6.2 and the result presented in Table 7.1 and Table 7.2. We experimented with adding more convolutional layers, noticing a decrease in performance instead.

8.2.2 Performance

Looking at the training and validation accuracy and loss (Figure 7.1 and Figure 7.2), the validation accuracy seems to be higher than the training accuracy throughout the epochs. The same goes for the loss, where the loss is constantly lower for the validation set than for the training set. This can be explained by the pre-processing function that adds noise to the training set, making the training set harder to classify than the validation set. Figure 7.2 is showing that the validation loss is decreasing and from the plot it seems like it would keep decreasing even if the network were trained over more epochs. This is indicating that the CNN is of suitable size and that the network is at no risk of being overtrained.

The simple CNN generated the worst results for both the Swedish and Egyptian data set. Comparing the scores for the Egyptian test set in Table 7.1 with the results of a similar CNN mentioned in "Comparison of Transferred Deep Neural Networks in Ultrasonic Breast Masses Discrimination" by Xiao et al. [15], the results seem very fair. In the article they achieved an AUC score of 0.78, specificity of 79.22% and a sensitivity of 63.19%. Comparing these numbers with an AUC of 0.76, specificity of 56.10% and sensitivity of 82.61%, the results are in the same range. It is also important to keep in mind that the data set used in the article is approximately twice the size of the one used in this thesis, which could potentially explain the slightly better result.

8.3 Transfer Learning

8.3.1 Architecture

The amount of training in each transfer base was experimentally determined with the starting point of not training at all to increasing the amount of training by one convolutional block at a time. As mentioned in "Comparison of Transferred Deep Neural Networks in Ultrasonic Breast Masses Discrimination" by Xiao et al. [15] performance usually decreased if more than two block were fine-tuned during training . The authors of the article do not explicitly mention which blocks were unfrozen in each model. In contrast to the report mentioned, we also tried cutting blocks off of the models to see if the performance could be improved. The only case of this was for Inception, where the last Inception module was cut off.

The article "Practical Recommendations for Gradient-Based Training of Deep Architectures" by Bengio describes how an overcomplete first fully connected layer is preferred over an undercomplete one and does generally not hurt the performance [22]. The first layer size of 2048 did indeed boost the performance of ResNet50V2 but layer of size 1024 was enough for rest of the models. Bengio also found that *"using the same size for all layers worked generally better or the same as using a decreasing size (pyramid-like) or increasing size (upside down pyramid)"*. We decided to follow Bengio's recommendation and found it to be true. Using a pyramid-like layer configuration in the ResNet50V2 transfer network was solely for the purpose of all models providing the same amount of data in each feature vector in the latter described deep feature models. For the same reason, all transfer models have a fully connected layer of size 1024 as their last layer before the classification layer with the sigmoid activation.

The number of fully connected layers were experimentally determined by using the method proposed by Bengio: simply keep adding layers until the test error does not improve anymore [23]. Combining this with the previously mentioned article by Bengio [22], the layers added were of the same size as the first fully connected layer. Lastly, dropout layers were added of 50%.

8.3.2 Performance

As seen in Figures 7.3-7.10, the training accuracy is in most cases higher than the validation accuracy and the training loss is lower than the validation loss. This implies that most of the models are overfitted to the training data. Since there are not many samples in the training data, this is likely to occur. As mentioned before in the section about the performance of the simple CNN, noise is applied to the training set, hence the training set is more difficult to classify than the validation set. The transfer model based on ResNet50V2 does not seem as overfitted as the rest, since the training and validation accuracy and loss are overlapping.

All transfer models generated good results on both test sets with numbers within the same range. No model was significantly worse or better than the others. The scores obtained for the Egyptian test set (Table 7.3) are similar to the results obtained by Xiao et al. presented in the report "Comparison of Transferred Deep Neural Networks in Ultrasonic Breast Masses Discrimination" [15] and hence seem reasonable. The authors got AUC score 0.91, 0.91, 0.90 for Inception, ResNet50V2 and Xception respectively compared to the AUC score 0.90, 0.91, 0.92 in this thesis. For VGG19, we got an AUC score of 0.89, comparable to the other AUC scores it seems reasonable.

The article "Computer-aided diagnosis of breast ultrasound images using ensemble learning from convolutional neural networks" by Moon et al. [24] presents results of testing a CNN structure on the same Egyptian data set as the one being used in this thesis. Their best network is based on DenseNet121 and resulted in an AUC score of 0.9052, sensitivity of 83.78% and specificity of 90.32%. In contrast to our experiments, they trained the complete network from scratch. In this thesis, the best transfer network was based on Xception, with an AUC of 0.92, sensitivity of 86.96% and specificity of 80.49%. Comparing these with the scores mentioned in the article written by Moon et al. they seem reasonable.

8.3.3 Grad-CAM

The heatmaps produced using the Grad-CAM technique show interesting results (Figure 7.11-7.15). In most cases the models look at the actual tumor, which should be the main factor of whether a tumor is malignant or not since characteristics such as shape, margin, orientation and echo patterns are present in

the tumor area (see Section 3.2). In some cases, models look at regions below the tumor. As mentioned in Section 3.2, the last characteristic is posterior features which is enhancement or shadowing below a structure, for example a tumor. Potentially could this be what the model is seeing and identifying as malignant.

The Grad-CAM heatmaps generated from VGG19 shows some deviant results by highlighting the edges of the image. Since all Grad-CAM images are produced in the same manner, this could be an effect due to the VGG19-architecture. However, the highlighted edges should not be focused on when analyzing the VGG19 heatmaps.

8.4 Deep Features

8.4.1 Architecture

Initially, the complete feature vector directly from each transfer network (before passing through the additional fully connected layers) was used in the large concatenated feature vector. This was not possible when concatenating feature vectors from 3 or more models due to memory limitation. To solve this problem, it was decided that all models should include a fully connected layer with output size 1024 as last layer before the sigmoid activation layer, so that this vector could be used when constructing the large combined feature vector. This would also allow the features to adapt additionally to the training data in case no fine tuning was performed in the transfer base. Comparing the performance when using large feature vectors directly from the base to using feature vectors from the last fully connected layer with output of size 1024, we barely noted any difference. In some cases, the result was slightly better. Similarly, increased performance was observed when choosing the last fully connected layer with output size 1024 compared to choosing the first when several was available (applies to InceptionV3 and VGG19). This makes sense since it allows more complex features to be used.

As mentioned earlier, layers of the same size should be added until there is no increased performance, according to Bengio [23]. This theory was applied to the ANN implemented to classify the feature vectors from different combination of transfer networks. For a fair result, the same ANN was used for all combinations. Tweaking the dropout layers, it was clear that a dropout of 60% was more effective in preventing overfitting.

8.4.2 Performance

When analysing the plots for the training and validation loss of the deep feature models (Appendix) it can be seen that overfitting appears for some of the models. The overfitting occurs around 10-30 epochs into the training and made us suspect that the results would be even better if we had stopped the training a bit earlier. This hypothesis was tested on some of the models and it became clear that in reality the performance of training with 35 epochs were better even though the network got a bit overtrained. What we noticed was that decreasing the amount of epochs resulted in some underfitting instead. In theory the optimal loss plot would contain equal loss for both training and validation. This is difficult to achieve, hence some overfitting is not as unexpected as one would have thought.

Looking at Table 7.5, the performance did not increase significantly when using features from a larger or smaller number of transfer models. The best model out of all combinations was the deep feature model based on InceptionV3, Xception and VGG19. This model got an AUC of 0.93, sensitivity of 95.65% and specificity of 82.93% when evaluated on the Egyptian test set. In the article "Comparison of Transferred Deep Neural Networks in Ultrasonic Breast Masses Discrimination" by Xiao et al. [15], the authors report their best result of AUC 0.93, sensitivity of 88.73% and specificity of 89.95% for a combination of InceptionV3, ResNet50V2 and Xception. This report did not include VGG19.

The deep feature combinations outperformed the transfer networks. A reason for this might be because they include several different features instead of the features from one single transfer network. The more and better features, the more accurate the classification will be. The result might indicate that the features extracted using ResNet50V2, VGG19 and Xception are better at separating the two classes apart even though they were not necessarily the three best networks by themselves. The best combinations of transfer networks in a deep feature model is very much data dependent. Since the training set

used is small and hence lacking samples covering all possible cases, the result can be varying. As Luo et al. conclude in "How Does the Data set Affect CNN-based Image Classification Performance?": *"when the data set size is small [. . .] we can do more experiments and retain the best results"* [25]. Due to this effect, the exact results presented in this thesis are not expected each time the models are trained. The results were selected so that they represent the majority of the experiments for each network.

To present a clearer result of the varying training, it would have been a good idea to use some kind of uncertainty measure, for example standard deviation. One way of doing this would be by using cross-validation where the data set is divided multiple times into different training and test sets, resulting in multiple models trained under the same conditions. The different models would then be evaluated and an uncertainty could be estimated.

8.5 Evaluation on Data Set Collected by Skåne University Hospital

A common pattern for all models developed in this thesis, is the worse performance on the Swedish data set. In average, the accuracy was 14 percentage points worse for the Swedish data set compared to the performance on the Egyptian test set. This is supported by the average AUC difference of 0.1306. The reason for this is the lack of training data as well as differences between the data sets. The authors of the study "How Does the Data set Affect CNN-based Image Classification Performance?" by Luo et al. conclude that the larger training set, the better the classification performance [25]. It is important for the model to see as many cases as possible during training, so that it can recognize them when classifying unseen data. Not only does this apply to different tumor shapes and sizes, but also different image qualities, such as noise, contrast and other types of visual distortion. As seen for the Swedish data set, the images are not of as high quality as the Egyptian data set. This is probably the main reason why the performance is worse.

8.6 Data Availability

As mentioned above, the lack of data was the main reason for the worse performance on the Swedish data set. An optimal training data set would be images from all types of ultrasound machines and cancer cases. In practice, this is impossible due to laws prohibiting distribution of personal data. For example in Sweden Patientdatalagen combined with the EU law General Data Protection Regulation (GDPR) protect patient data from free distribution [26]. This makes collection of patient data like ultrasound images difficult, limiting the availability of open data sets.

8.7 Conclusion and Further Development

The results obtained in this thesis seem reasonable compared to similar studies. As expected, the transfer networks performed better than the simple CNN. The deep feature combinations of transfer networks performed even better than the transfer networks by themselves.

For further work with this thesis the usage of thresholds should be reconsidered. Determining the thresholds for the classification should be a part of the model, hence it should be chosen with consideration to the validation set not the test set. Another remark would be not using different thresholds for the Egyptian and Swedish data sets. A more reasonable solution is to use the threshold obtained from the Egyptian validation set for testing on both the Egyptian and Swedish sets. Modifying the thresholds for the Swedish data set makes the results less reliable and possible unfair.

An important improvement for future work would be to expand the current data set. Generally more data covers more possible cases giving the data set more variability. Expanding the data set by collecting more images can, as discussed earlier, be difficult due to laws and restrictions when handling patient data. If collecting more images is not possible another option would be to use augmentation. In this thesis some traditional augmentations were used to increase the variety of the images. Another technique is using generative adversarial networks (GAN) to generate new images based on the real images of a data set. The difference between traditional data augmentation and GAN is the way of generating images.

GAN synthesize a whole new image using CNN instead of changing an existing image. In this thesis using traditional augmentation showed improvements. An interesting aspect would be to use GAN as well.

In this thesis there are no markers for the algorithm to know where in an images the tumor is positioned. As the Grad-CAM images illustrates the algorithm is not always making a decision based on the tumor. Regions of interest (ROI) is when a marking around the interesting area in the image is used as a complement to the algorithm, hence to focus more on the tumor. As mentioned before the present algorithm are at times using the area under a tumor as support for its decision. One suggestion would to be using ROI on the tumor and in the cases were superior features are existing also below the tumor.

Bibliography

[1] Folkhälsomyndigheten. *Dödlighet i bröstcancer*. 2021. URL: https://www.folkhalsomyndigheten.se/folkhalsorapportering-statistik/tolkad-rapportering/folkhalsans-utveckling/resultat/halsa/brostcancer-dodlighet/ (visited on 2021-08-06).

[2] H. Moller, F. Sandin, F. Bray, Å. Klint, K. Linklater, A. Purushotham, D. Robinson, and L. Holmberg. "Breast cancer survival in england, norway and sweden: a population-based comparison". *International journal of cancer. Journal international du cancer* **127** (2010), pp. 2630–8. DOI: 10.1002/ijc.25264.

[3] J. Harford, I. Otero, B. Anderson, E. Cazap, W. Gradishar, J. Gralow, G. Kane, L. Niëns, P. Porter, A. Reeler, P. Rieger, L. Shockney, L. Shulman, T. Soldak, D. Thomas, B. Thompson, D. Winchester, S. Zelle, and R. Badwe. "Problem solving for breast health care delivery in low and middle resource countries (lmcs): consensus statement from the breast health global initiative". *Breast (Edinburgh, Scotland)* **20 Suppl 2** (2011), S20–9. DOI: 10.1016/j.breast.2011.02.007.

[4] F. Kamangar, G. Dores, and W. Anderson. "Patterns of cancer incidence, mortality, and prevalence across five continents: defining priorities to reduce cancer disparities in different geographic regions of the world". *Journal of clinical oncology : official journal of the American Society of Clinical Oncology* **24** (2006), pp. 2137–50. DOI: 10.1200/JCO.2005.05.2308.

[5] J. Ferlay, I. Soerjomataram, R. Dikshit, and S. Eser. "Cancer incidence and mortality worldwide: sources, methods and major patterns in globocan 2012". *Int J Cancer* **5** (2015), E359–E386.

[6] H. Gelband, R. Sankaranarayanan, C. Gauvreau, S. Horton, B. Anderson, F. Bray, J. Cleary, A. Dare, L. Denny, M. Gospodarowicz, S. Gupta, S. Howard, D. Jaff, F. Knaul, C. Levin, L. Rabeneck, P. Rajaraman, T. Sullivan Terrence, E. Trimble, and M. Harif. "Costs, affordability, and feasibility of an essential package of cancer control interventions in low-income and middle-income countries: key messages from disease control priorities, 3rd edition". *The Lancet* **387** (2015). DOI: 10.1016/S0140-6736(15)00755-2.

[7] W. Al-Dhabyani, M. Gomaa, H. Khaled, and A. Fahmy. "Dataset of breast ultrasound images". *Data in Brief* **28** (2019), p. 104863. DOI: 10.1016/j.dib.2019.104863.

[8] Cancerfonden. *Godartad eller elakartad tumör?* URL: https://www.cancerfonden.se/om-cancer/symtom-och-orsaker/fragor-och-svar (visited on 2021-08-06).

[9] Vårdguiden. *Bröstcancer*. 2020. URL: https://www.1177.se/sjukdomar--besvar/cancer/cancerformer/brostcancer/ (visited on 2021-08-06).

[10] The American Cancer Society medical and editorial content team. *Breast ultrasound*. 2019. URL: https://www.cancer.org/cancer/breast-cancer/screening-tests-and-early-detection/breast-ultrasound.html (visited on 2021-08-06).

[11] H. Zonderland and R. Smithuis. *Bi-RADS for Mammography and Ultrasound 2013 Updated version*. 2014. URL: https://radiologyassistant.nl/breast/bi-rads/bi-rads-for-mammography-and-ultrasound-2013#mammography-breast-imaging-lexicon-mass (visited on 2021-08-06).

[12] The American Cancer Society medical and editorial content team. *Ultrasound for Cancer*. 2015. URL: https://www.cancer.org/treatment/understanding-your-diagnosis/tests/ultrasound-for-cancer.html (visited on 2021-08-06).

[13] P. R. Hoskins, K. Martin, and A. Thrush. *Diagnostic ultrasound: Physics and equipment*. CRC Press/Taylor Francis Group, 2019.

[14] D. P. Kingma and J. Ba. *Adam: a method for stochastic optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[15] T. Xiao, L. Liu, K. Li, W. Qin, N. Yu, and Z. Li. "Comparison of transferred deep neural networks in ultrasonic breast masses discrimination". *BioMed Research International* **2018** (2018), pp. 1–9. DOI: 10.1155/2018/4605191.

[16] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition" (2015). arXiv: 1512.03385 [cs.CV].

[17] K. He, X. Zhang, S. Ren, and J. Sun. "Identity mappings in deep residual networks" (2016). arXiv: 1603.05027 [cs.CV].

[18] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. "Rethinking the inception architecture for computer vision" (2015). arXiv: 1512.00567 [cs.CV].

[19] F. Chollet. "Xception: deep learning with depthwise separable convolutions" (2017). arXiv: 1610.02357 [cs.CV].

[20] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition" (2015). arXiv: 1409.1556 [cs.CV].

[21] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. "Grad-cam: visual explanations from deep networks via gradient-based localization". *International Journal of Computer Vision* **128**:2 (2019), pp. 336–359. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01228-7. URL: http://dx.doi.org/10.1007/s11263-019-01228-7.

[22] Y. Bengio. "Practical recommendations for gradient-based training of deep architectures" (2012). arXiv: 1206.5533 [cs.LG].

[23] Y. Bengio. *Artificial neural networks: how can i estimate the number of neurons and layers?* 2013. URL: https://www.quora.com/Artificial-Neural-Networks/Artificial-Neural-Networks-How-can-I-estimate-the-number-of-neurons-and-layers/answer/Yoshua-Bengio?share=7b58dc3b&srid=hqJd (visited on 2021-08-06).

[24] W. K. Moon, L. yan-wei, H.-H. Ke, S. Lee, C. F. Huang, and R.-F. Chang. "Computer-aided diagnosis of breast ultrasound images using ensemble learning from convolutional neural networks". *Computer Methods and Programs in Biomedicine* **190** (2020), p. 105361. DOI: 10.1016/j.cmpb.2020.105361.

[25] C. Luo, X. Li, L. Wang, J. He, D. Li, and J. Zhou. "How does the data set affect cnn-based image classification performance?" (2018), pp. 361–366. DOI: 10.1109/ICSAI.2018.8599448.

[26] Sveriges Riksdag. *Patientdatalag (2008:355)*. 2021. URL: https://www.riksdagen.se/sv/dokument-lagar/dokument/svensk-forfattningssamling/patientdatalag-2008355_sfs-2008-355 (visited on 2021-08-06).

A

Appendix

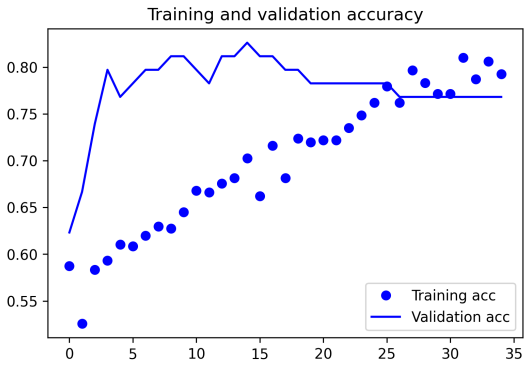


Figure A.1: Training and validation accuracy for deep feature model based on InceptionV3 and ResNet50V2.

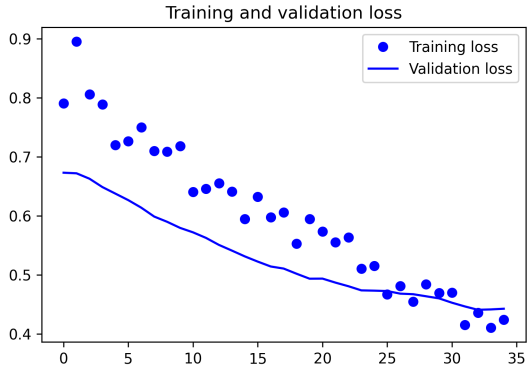


Figure A.2: Training and validation loss for deep feature model based on InceptionV3 and ResNet50V2.

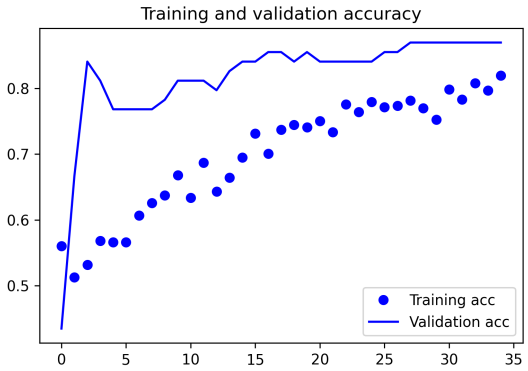


Figure A.3: Training and validation accuracy for deep feature model based on InceptionV3 and Xception.

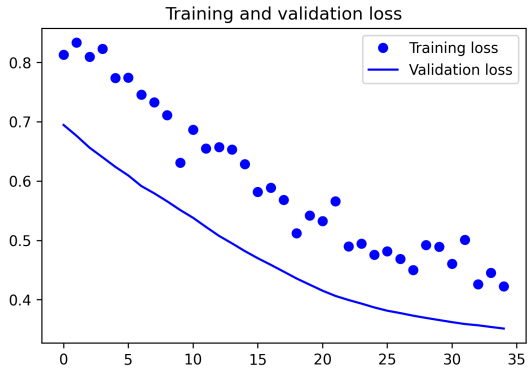


Figure A.4: Training and validation loss for deep feature model based on InceptionV3 and Xception.

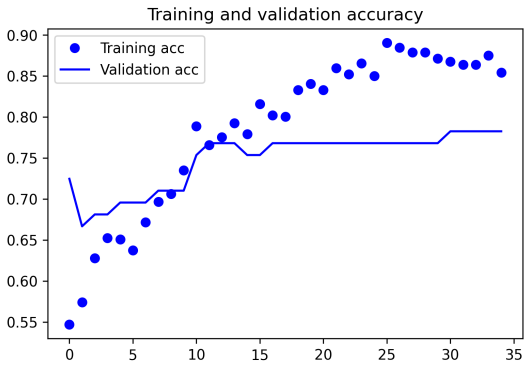


Figure A.5: Training and validation accuracy for deep feature model based on InceptionV3 and VGG19.

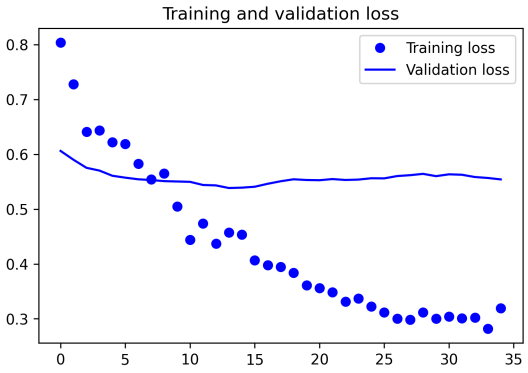


Figure A.6: Training and validation loss for deep feature model based on InceptionV3 and VGG19.

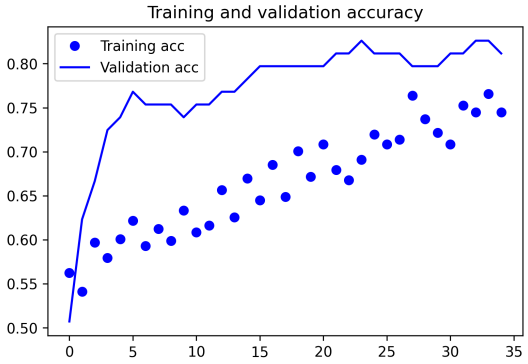


Figure A.7: Training and validation accuracy for deep feature model based on ResNet50V2 and Xception.

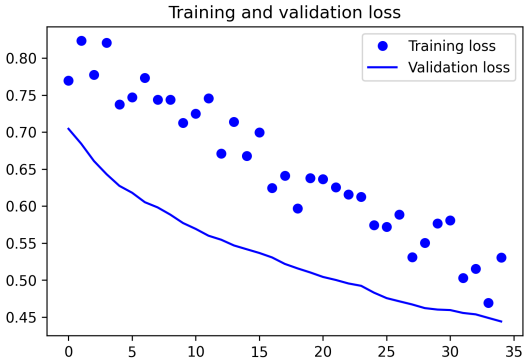


Figure A.8: Training and validation loss for deep feature model based on ResNet50V2 and Xception.

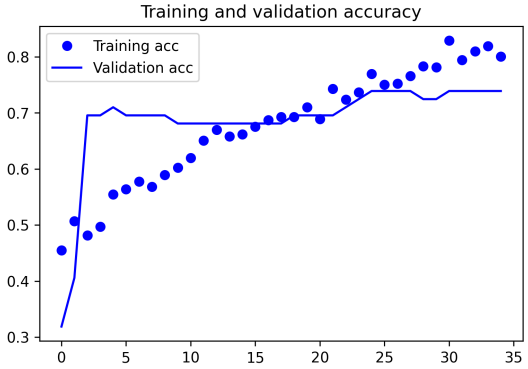


Figure A.9: Training and validation accuracy for deep feature model based on ResNet50V2 and VGG19.

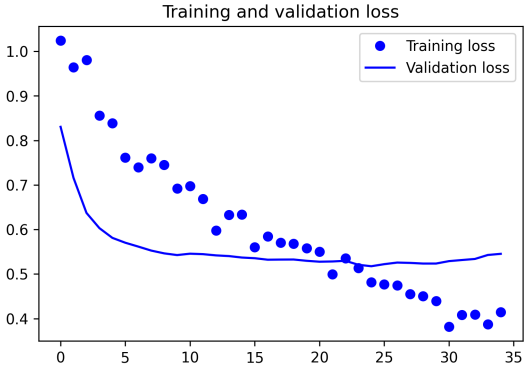


Figure A.10: Training and validation loss for deep feature model based on ResNet50V2 and VGG19.

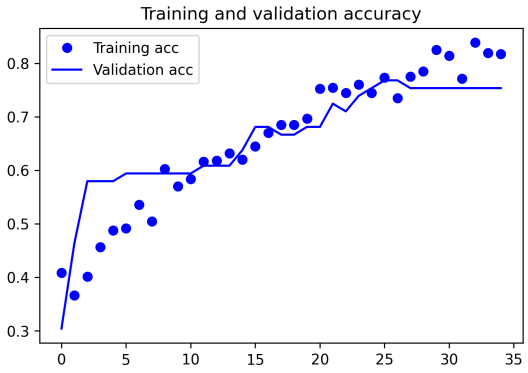


Figure A.11: Training and validation accuracy for deep feature model based on Xception and VGG19.

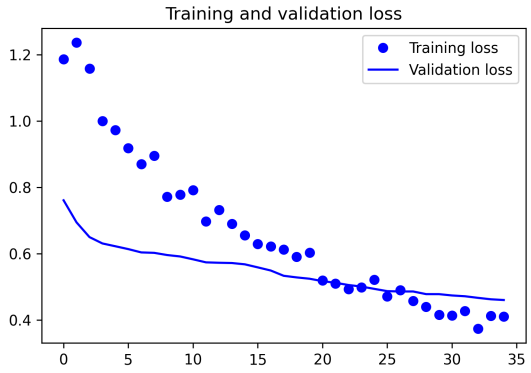


Figure A.12: Training and validation loss for deep feature model based on Xception and VGG19.

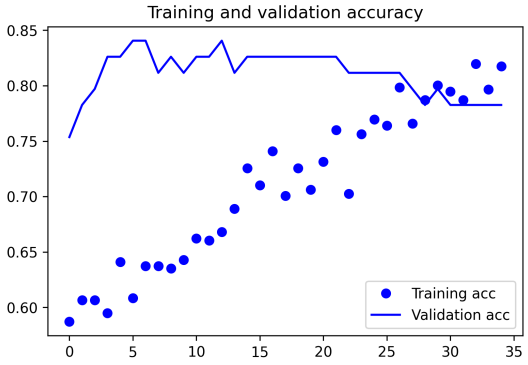


Figure A.13: Training and validation accuracy for deep feature model based on InceptionV3, ResNet50V2 and Xception.

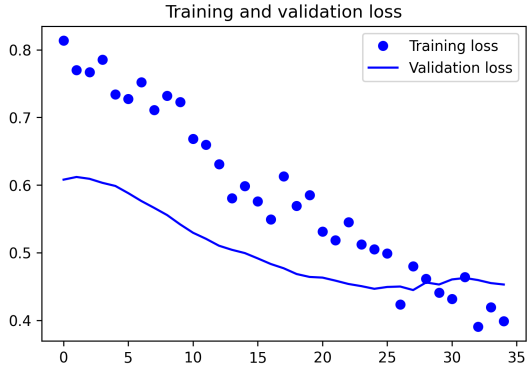


Figure A.14: Training and validation loss for deep feature model based on InceptionV3, ResNet50V2 and Xception.

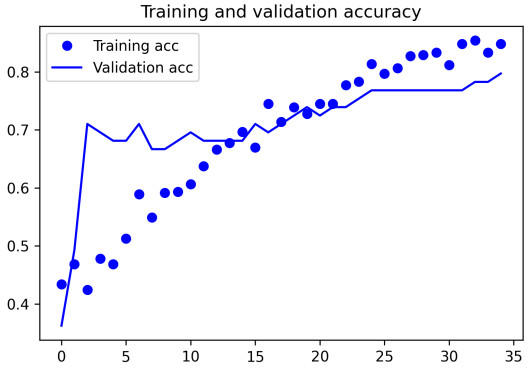


Figure A.15: Training and validation accuracy for deep feature model based on InceptionV3, ResNet50V2 and VGG19.

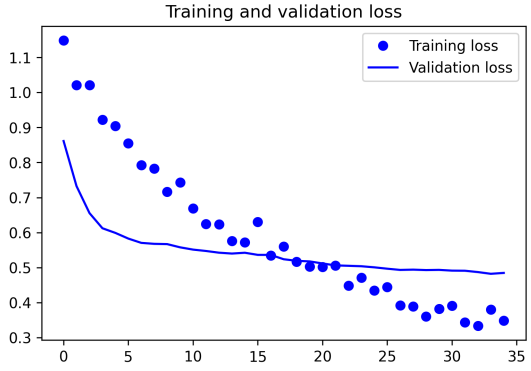


Figure A.16: Training and validation loss for deep feature model based on InceptionV3, ResNet50V2 and VGG19.

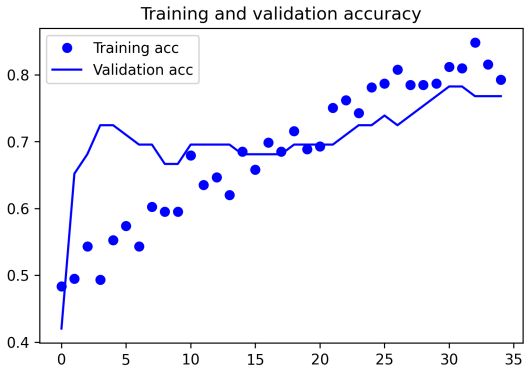


Figure A.17: Training and validation accuracy for deep feature model based on ResNet50V2, Xception and VGG19.

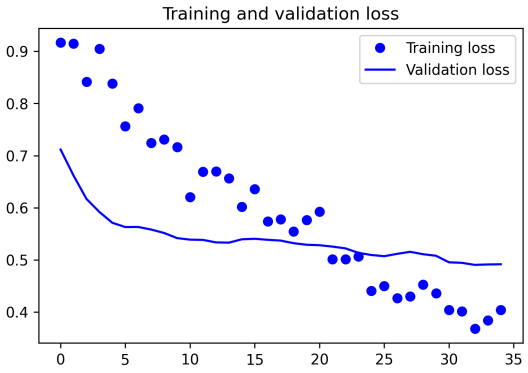


Figure A.18: Training and validation loss for deep feature model based on ResNet50V2, Xception and VGG19.

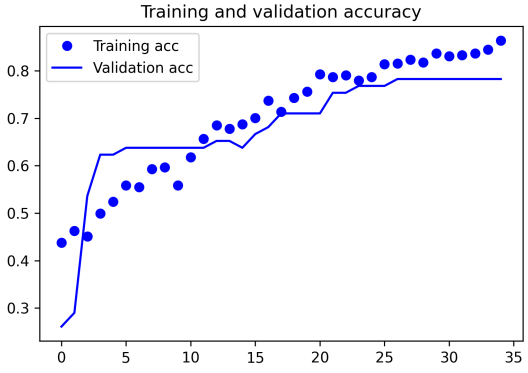


Figure A.19: Training and validation accuracy for deep feature model based on InceptionV3, Xception and VGG19.

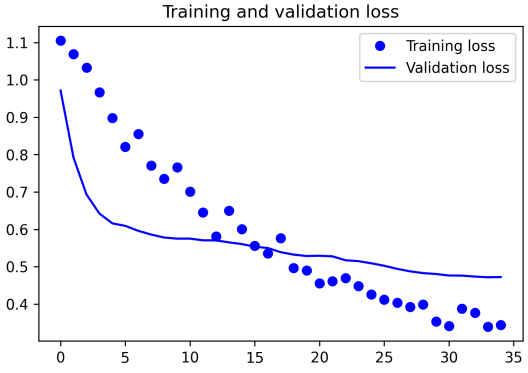


Figure A.20: Training and validation loss for deep feature model based on InceptionV3, Xception and VGG19.

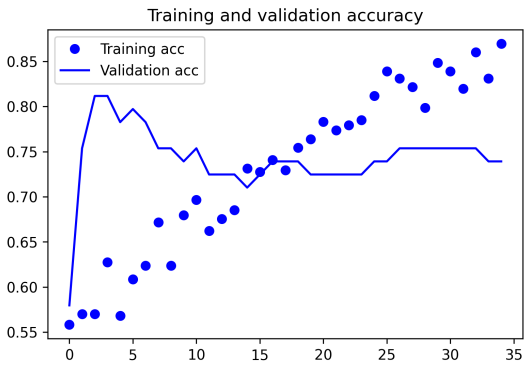


Figure A.21: Training and validation accuracy for deep feature model based on InceptionV3, ResNet50V2, Xception and VGG19.

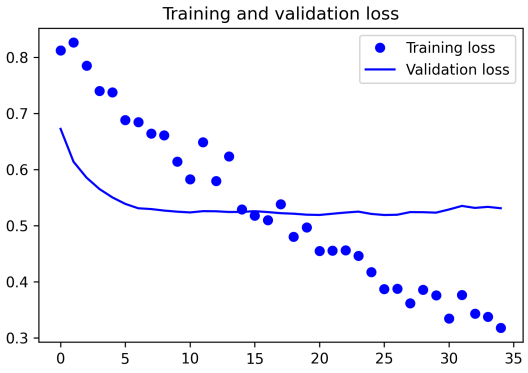


Figure A.22: Training and validation loss for deep feature model based on InceptionV3, ResNet50V2, Xception and VGG19.