Breast Cancer Image Classification using Convolutional Neural Networks

Julia Graf[1*†], Jana Hoffmann[1*†], Jessie Midgley[1*†] and Maike Nägele[1*†]

[1*] Algorithms in Bioinformatics, University of Tübingen, Sand 14, Tübingen, 72076, Germany.

*Corresponding author(s). E-mail(s): julia.graf@student.uni-tuebingen.de; jana.hoffmann2@student.uni-tuebingen.de; jessie.midgley@student.uni-tuebingen.de; maike.naegele@student.uni-tuebingen.de;
†These authors contributed equally to this work.

Abstract

Breast cancer, a frequently diagnosed and often life-threatening disease, poses a significant threat, particularly among females. The danger associated with breast cancer underscores the need for advanced techniques to improve the diagnosis of patients suffering from breast cancer. Mammography is a widely used technique for screening breast cancer. In this report, mammographic screens from the CBIS-DDSM Breast Cancer Image Datataset are used to perform binary breast cancer classification into malignant and benign cases. Classification is performed based on a machine learning approach using two different Convolutional Neural Networks (CNNs): VGG16 and ResNet50.Best performance is achieved with the ...

Keywords: keyword1, Keyword2, Keyword3, Keyword4

1 Introduction

Breast cancer is a significant global health concern, impacting the lives of millions each year [1]. Early detection plays an important role in improving health outcomes and reducing mortality rates in cancer patients [2]. Currently, mammograms serve as a primary diagnostic tool in the identification of abnormalities in breast tissue

[2]. However, the challenge of accurately interpreting these images and distinguishing between cancerous and non-cancerous abnormalities still remains [3]. The use of deep learning methods, particularly Convolutional Neural Networks (CNNs), have had a huge successes in medical image analysis [4]. Our project centers around the use of CNNs to classify mammograms into benign and malignant cases which, in theory, could aid healthcare professionals in making informed decisions about patient treatment plans. To train our CNNs we made use of the Curated Breast Imaging Subset of the Digital Database for Screening Mammography (CBIS-DDSM) dataset [5]. Since training a deep CNN requires a sufficiently large dataset in order to avoid overfitting, the issue of limited training samples is often addressed by transfer learning [6]. This technique uses a larger (often unrelated) database to train an initial model, which can then be fine-tuned on the dataset of interest [6]. Since large, publicly available medical imaging datasets are limited, we make use of the ImageNet dataset [7]. To develop our breast cancer classifier, we investigate the performance of two existing CNN architectures, namely VGG-16 [8] and ResNet-50 [9], that have already shown great success for ImageNet classifications, and transfer these pre-learned weights to classify mammogram images. By systematically comparing the performance of these models, we seek to identify the most effective architecture for distinguishing between benign and malignant cases. In addition, we investigate the effectiveness of various image pre-processing and augmentation methods in improving model performance.

2 Materials and Methods

2.1 Dataset

The CBIS-DDSM dataset [5] is an updated version of the Digital Database for Screening Mammography (DDSM) [10], containing digitized mammograms in standard DICOM format instead of the, now obsolete, lossless-JPEG format. The subset of 1644 mammography images from 1566 women that are included in the CBIS-DDSM have been selected and curated by a trained mammographer. The dataset is made up of 753 calcification cases and 891 mass cases, with a pathologic diagnosis for each image. We combine these cases and create a random train:test:validation split of 70:20:10.

2.2 Models

2.2.1 ResNet-50

ResNet-50 is a variant of Residual Network (ResNet) architecture, which makes use of skip connections designed to address the vanishing gradient problem. The vanishing gradient problem typically arises during the training of deep neural networks, when the gradients of the loss function become continuously smaller as they are backpropagated through the network. Skip connections allow the gradients to flow directly though the network, thereby mitigating the vanishing gradient problem. The ResNet-50 architecture consists of 50 layers of weights and several residual learning blocks with varying number of convolutional layers within each block. Each convolutional layer uses Batch normalization as a regularization method. In total, ResNet-50 is made up of a total of 48 convolutional layers, along with 2 pooling layers, producing a grand total of 23 million trainable parameters. The network takes images as input with a dimension of 224 x 224 pixels, and uses softmax in the final layer for the classification of input images [9].

2.2.2 VGG-16

VGG-16 is a Convolutional Neural Network architecture proposed by Simonyan and Zisserman in 2014, containing 16 layers [?]. The VGG-16 architecture takes RGB images of a size of 224x224 pixels.
The model consists of different types of layers. The early layers in the network comprise 13 convolution layers containing 5 max-pooling layers. This is followed by 3 dense layers flattening the output of the previous layers and applying a non-linear sigmoid to perform the actual classification. In total, VGG-16 has approximately 15 million trainable parameters [?].

2.3 Hyperparameter

-What hyperparameter settings did you use?

2.3.1 Preprocessing Images

2.3.2 Fine Tuning

We experiment with re-training the last layer of the pre-trained model with the new images. This means that the original weights from the first to the penultimate layer are preserved or "frozen", and the last layer is replaced with new weights in order to learn task-specific features. In addition, a final fully-connected layer with 128 units was added to the ResNet-50 model in order to increase the capacity of the model.

2.3.3 Optimizer and Learning Rate

We compare the performance of two different optimizers, namely Adam and Nadam. Adam computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients [11]. Nadam is an extension of the Adam optimizer that incorporates Nesterov momentum and can lead to improved convergence [12]. The learning rate is a hyperparameter of the optimizer, determining the weight with which parameters are updated at each iteration during optimization [13]. Yuanyuan et al. even claim that the learning rate is the hyperparameter with the largest influence on the classification accuracy of a model [14]. We compare the two optimization algorithms at three different learning rates. With a larger learning rate, the model tends to converge faster, however with an increased risk of overshooting the optimal value of the loss function [13]. If the learning rate is too small, the training process might be time-consuming.

2.3.4 Flatten vs. Global Average Pooling

In order to pass the output of the last convolutional layer to the fully-connected layer and then further to the softmax layer, the vector needs to be converted into a one-dimensional vector. This can be done by applying a flatten layer where the resulting

one-dimensional vector contains the same values as before the flattening.
Another layer that we can add to reduce the dimensions is a Global Average Pooling layer. This layer creates one feature map for each class of the output and the average of each is passed to the softmax layer [**?**].

2.3.5 Image Data Generator

For the training we only have 2300 images to train on after splitting the images of the dataset in train, validation and test sets. We compare the performance of our model on these original images in the train set to the images of the train set randomly modified by an ImageDataGenerator from tensorflow before each epoch [15]. We use the following parameters:

- rotation_range=40
- width_shift_range=0.2
- height_shift_range=0.2
- shear_range=0.2
- zoom_range=0.2
- horizontal_flip=True
- fill_mode='nearest'

So the input images get randomly flipped by a range between 0° and 40°, shifted on their horizontal axis up to 20% of their width, shifted on their vertical axis up to 20% of their height, zoomed by a factor within a range of 0.8 and 1.2, flipped from left to right and the shear transformation is applied to them with an angel of 0.2°. The fill mode fills pixels outside as the nearest pixel. The model than sees more different images. The amount of images that gets seen in each epoch is the same as before.

2.3.6 Data Augmentation

We also test the performance on an augmentation method we wrote ourselves to increase the amount of images the model sees in each epoch. Therefor we implemented the augmentation techniques Montaha et. al. used in their work of the BrestNet18 model [16]. Through the augmentation we generate from each input image of the train set 7 new ones, by flipping vertically, horizontally, vertically and horizontally and rotating the original and the horizontal flipped image by -30° and 30°. After the augmentation the train set includes 18.400 images.

2.3.7 Keeping Height/Width Ratio

Since the images in the dataset don't have a height and width of 224 pixels and the models we use have these input dimensions we need to resize the images before we can use them. The build-in function resize from scikit-image doesn't keep the aspect ratio of the pictures it processes [17]. Since this can distort important parts of the image we wrote a function that keeps the height/width ratio unchanged while resizing the images. For this we resize the image to the the maximum size in which it still fits in the dimensions (224px,224px) and in which the ratio is kept and fill the parts missing to get a image with the dimensions (224px,224px) black. We compare the performance of the models on images of these two resizing methods.

3 Results

3.1 Validation on Best Model

To evaluate the best VGG16 and ResNet50 model we trained them on 5 different train validation splits and use our pre-defined test set to evaluate the trained model. We generate the different splits by applying the train_test_split function from scikit-learn with five times and set the random_state parameter to None [18]. This way for each run a different values is being used. In the end we plot the ROC-curve for each of these runs and compute the mean test accuracy and the mean test auc.

3.2 VGG-16

Table 2 displays the performance of the model with different hyperparameters. The performance is measured based on two values: the accuracy and the area under the curve (AUC). The first one gives an inside on the correctly predicted classes among all the predictions that have been performed whereas the AUC is based on ...

The training of the respective models is performed on a training dataset. Validation is performed on a separate set of images, not seen by the model before.

Applying a Global Average Pooling layer instead of a Fatten layer gives rise to worse accuracy and AUC, so does the model, including the self-written preprocessing.

The best model is written in bold in the table. The model uses a learning rate of 0.00001, the Adam optimizer, and a flatten layer. The model has a validation accuracy of 72.3% and an AUC of 80.1%.

Figure 1 shows the training and test accuracy and loss per epoch of the model. Training, as well as validation accuracy, increased whereas the loss decreased. Figure 2 displays the ROC curves for the cross-validation applied to a test set using different train/validation splits. Models 1 and 2 have the highest area under the curve and therefore give the best performance to classify new images. The model achieves a mean accuracy on all splits of 68.6% and a mean AUC of 73.7%.

3.3 ResNet-50

– write ResNet results

4 Discussion

what you did in your project how you did it what the results mean? interpret your results and draw conclusion

Table 1 Results of VGG-16 hyperparameter tuning

Prepro	Last Layer Trainable	Adam/ Nadam	Flatten/ GAP[1]	Learning rate	Data Augm.[2]	H/W Ratio[3]	Acc[4]	AUC[5]
built-in	yes	Adam	Flatten	0.0001	no	no		
built-in	no	Adam	Flatten	0.0001	no	no	0.6933	0.7353
self-written	no	Adam	Flatten	0.0001	no	no	0.5920	0.6708
built-in	no	Nadam	Flatten	0.0001	no	no	0.6656	0.7337
built-in	**no**	**Adam**	**Flatten**	**0.00001**	**no**	**no**	**0.7270**	**0.8016**
built-in	no	Adam	Flatten	0.001	no	no	0.6748	0.7034
built-in	no	Adam	GAP	0.00001	no	no	0.5982	0.6355
built-in	no	Adam	Flatten	0.00001	yes	no	0.7086	0.7487
built-in	no	Adam	Flatten	0.00001	no	yes	0.7117	0.7286

[1] Global Average Pooling

[2] using built-in Image Data Generator

[3] Keeping height/width ratio

[4] Accuracy on validation set

[5] AUC on validation set

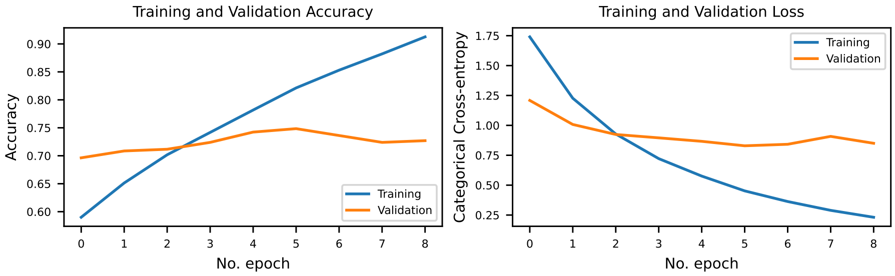

Fig. 1 Training and validation accuracy and loss of model with best performance

5 Conclusion

In the following report, two existing CNN architectures, namely VGG-16 and ResNet-50 are used to classify mammogram images into benign and malignant.

Additionally, data augmentation, data preprocessing, and hyperparamter tuning is performed to achieve better predictions of classes. Performance analysis of the models indicates the possibilities and benefits of artificial neural networks in health-related issues.

For both models an accuracy and AUC of over 65% is achieved. Although, the values indicate a moderate performance it is important to consider the limitations for further improvements.

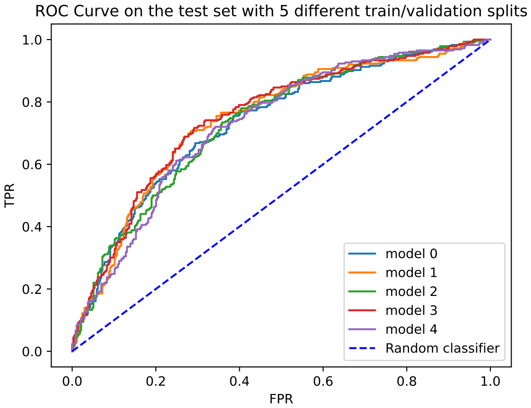

Fig. 2 ROC curve after cross-validation of model with best performance using different training/-validation splits.

Table 2 Results of ResNet-50 hyperparameter tuning

Prepro	Last Layer Trainable	Adam/ Nadam	Flatten/ GAP[1]	Learning rate	Data Augm.[2]	H/W Ratio[3]	Acc[4]	AUC[5]
built-in	yes	Nadam	Flatten	0.0001	no	no	0.7209	0.8036
self-written	yes	Nadam	Flatten	0.0001	no	no	0.6564	0.7217
built-in	no	Nadam	Flatten	0.0001	no	no	0.7270	0.8234
built-in	**no**	**Nadam**	**GAP**	**0.0001**	**no**	**no**	**0.7515**	**0.8269**
built-in	no	Nadam	GAP	0.001	no	no	0.7086	0.7959
built-in	no	Nadam	GAP	0.00001	no	no	0.6871	0.7617

[1] Global Average Pooling

[2] using built-in Image Data Generator

[3] Keeping height/width ratio

[4] Accuracy on validation set

[5] AUC on validation set

5.1 Limitations

The CBIS-DDSM dataset, used for classification, consists of 1644 images. For a model to perform good classification the size of the available training data plays an important role. As the data is taken from a publicly available source, there is a limited amount of data available.

Additionally, the performed classification treats “benign-without-callback” and “benign” identically. However, to ensure correct classification one might have to consider further information for the classification of these images.

Finally, hyperparameter tuning was restricted due to time and memory issues. As the servers were not accessible during the time of model training, all the hyperparameter tuning was run locally leading to long runtimes and memory-related issues.

References

[1] Siegel, R.L., Miller, K.D., Wagle, N.S., Jemal, A.: Cancer statistics, 2023. CA: A Cancer Journal for Clinicians **73**(1), 17–48 https://doi.org/10.3322/caac.21763

[2] Mathew, J., Sibbering, M.: In: Wyld, L., Markopoulos, C., Leidenius, M., Senkus-Konefka, E. (eds.) Breast Cancer Screening, pp. 147–156. Springer, Cham (2018)

[3] Elter, M., Horsch, A.: Cadx of mammographic masses and clustered microcalcifications: A review. Medical Physics **36**(6Part1), 2052–2068 (2009) https://doi.org/10.1118/1.3121511

[4] Cai, L., Gao, J., Zhao, D.: A review of the application of deep learning in medical image classification and segmentation. Ann. Transl. Med. **8**(11), 713 (2020)

[5] Lee, R.S., Gimenez, F., Hoogi, A., Miyake, K.K., Gorovoy, M., Rubin, D.L.: A curated mammography data set for use in computer-aided detection and diagnosis research. Sci. Data **4**(1), 170177 (2017)

[6] Kim, H.E., Cosa-Linan, A., Santhanam, N., Jannesari, M., Maros, M.E., Ganslandt, T.: Transfer learning for medical image classification: a literature review. BMC Med. Imaging **22**(1), 69 (2022)

[7] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, ??? (2009)

[8] Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition (2015)

[9] He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition (2015)

[10] Heath, M.D., Bowyer, K., Kopans, D.B., Moore, R.H.: The digital database for screening mammography. (2007). https://api.semanticscholar.org/CorpusID:68362967

[11] Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization (2017)

[12] Dozat, T.: Incorporating Nesterov Momentum into Adam. In: Proceedings of the 4th International Conference on Learning Representations, pp. 1–4

[13] Brownlee, J.: Understand the impact of learning rate on neural network performance (2020). https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/

[14] Li, Y., Zhang, Q., Won, D., Lu, F., Yoon, S.W.: Learning rate optimization in convolutional neural networks for medical images classification. (2020)

[15] tf.keras.preprocessing.image.ImageDataGenerator. https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator. Accessed: 2024-01-23

[16] Montaha, S., Azam, S., Rafid, A.K.M.R.H., Ghosh, P., Hasan, M.Z., Jonkman, M., De Boer, F.: Breastnet18: A high accuracy fine-tuned vgg16 model evaluated using ablation study for diagnosing breast cancer from enhanced mammography images. Biology **10**(12) (2021) https://doi.org/10.3390/biology10121347.PMID:34943262

[17] skimage.transform. https://scikit-image.org/docs/stable/api/skimage.transform.html#skimage.transform.resize. Accessed: 2024-01-23

[18] sklearn.model_selection.train_test_split. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html. Accessed: 2024-01-25