Effect of Synthetic Data on an Established Classifier

Niklas Khoss, MSc.   
and Elias Marcon, BA

*Abstract*—Medical imaging data generation and labeling is time and resource intense. Approaches to synthesize data have been shown to generate realistic images, which reduce the need for acquisition and classification of real data. Although generative methods and their synthetic data are evaluated before release, they are rarely tested in combination with existing classifiers. In this study, we set out to create synthetic data of histopathological breast cancer images and test an independent classification network with it, to investigate the effect synthetic data has on the training and classification performance.

Keywords—DenseNet, GAN

# Introduction

Over the past decades, the amount and generation speed of imaging data in healthcare and medical fields has skyrocketed, but the access to and distribution of it is still an ongoing issue. There are plenty of factors, which exacerbate the sharing of data across hospital and research/ecudational institutions, among them are privacy issues of the patients, sharing is not yet common among them, as well as legal or polititcal reasons. A second problem especially for machine learning or deep learning [1] approaches, which use medical imaging data, is the necessity for labels. Setting these labels is both time and resource intense, because they have to be set by experts in their respective field. Alleviating the need for the professionals’ time would directly reduce costs and workloads, which could be invested otherwise.

To overcome this bottleneck, researchers have begun to leverage a specific artificial intelligence tool name “Generative Adversarial Networks” [2] or GAN in short. This network architecture and its variations [3] usually consist of a Generator, which generates data similar to the input data, and a Discriminator, which tries to distinguish original from synthesized data. Within the field of medical imaging, GANs have been used to achieve different goals [4], among them are:

* Reconstruction
* Synthesis
* Segmentation
* Classification

A second approach to overcome the shortage of trained personel time is to automate and/or assisst the classification of pathologies in medical images. The astounding performance of various deep learning tools has been shown on many types of data over the years [5].

In this study, we aimed to investigate the effect of synthetic data on the classification performance of a DenseNet [6], a special type of convolutional network, in an attempt to justify the generation and usage of synthetic data, which would alleviate the need for patient data labeled by professionals and its concomittant problems.  
 To achieve this, we first established the performance of a DenseNet, pre-trained on Imagenet [7], a database with over 14 million images, on a pathohistological breast cancer image dataset, BreakHis [8] [9]. The second step included testing different GAN architectures and generating synthetic images using the BreakHis dataset. Lastly, we compared the training and test performance of the DenseNet on combinations of the original and synthetic datasets.

# Methods and Materials

## BreakHis Dataset

The BreakHis dataset [8] is open source and consists of 9,109 microscopic images malignant of breast cancer tissues collected from 82 patients. It contains 2,480 benign and 5,429 malignant samples in 4 magnifications, 40X, 100X, 200X and 400X. Each image is 700x460 pixels in size and has 3 channels (RGB)). The dataset is split into 8 subtypes, 4 benign tumor types (adenosis, firboadenoma, phyllodes and tubular) and 4 malignant tumor types (ductal, lobular, mucinous and papillary carcinoma).

## DenseNet

“DenseNet [6] is a densely connected convolutional network utilises dense connections between layers, through Dense Blocks, where we connect all layers (with matching feature-map sizes) directly with each other. To preserve the feed-forward nature, each layer obtains additional inputs from all preceding layers and passes on its own feature-maps to all subsequent layers.” [10] A schematic architecture is shown in Figure 1.

## Generative Adversarial Networks

A schematic workflow and architecture can be seen in Figure 2. The chosen architectures consisted of less than 10 convolutional layers and were adapted to fit our data and use-case.

## Data processing, Workflow and Training.

The first preprocessing steps consisted of restructuring the dataset and filtering out incorrectly shaped images.Before usage, images have been rescaled from 700x460 pixels to 64x64, 128x128 or 256x256 pixels, because both the DenseNet and the GAN architectures needed square input images.

For all networks and training steps, we used the TensorFlow/Keras framework and the Adam optimizer with an initial learning rate of 10-4. Training and validation losses were logged and saved as a .CSV file at the end of the training.

## Cluster and computational resources

All the code has been written in Python 3.8 and most of the development has been done locally using Jupyter notebooks. Once we hit resource limitations for training networks, we ported all notebooks to python scripts and utilized the FHTW HPC, which consists of 19 nodes, each with 1 AMD Ryzen 9 3900X, 1 NVIDIA GeForce RTX 2080 Ti GPU, 32 GB RAM and 1 80 GB Hard Disk

# Results

## DenseNet classification performance on BreakHis dataset

First, we set out to train and test the DenseNet on the BreakHis dataset to check how well the network could classify the original data. In the initial trial, we did not alter the input images. We performed a train/validation/test split containing 64%/16%/20% images respectively. Within a handful of epochs, the DenseNet achieved training and validation accuracies well above 90% (see Figure 4).

Following the initial trial, we systematically tested the DenseNet on all given magnifications and 3 different image sizes, 64x64, 128x128 and 256x256 pixels. For simplicity’s sake, we opted to rescale the images, instead of cropping and to decide the fitness of this decision after we see the quality of the results.

Table 1 shows a truncated example classification report of the DenseNet test results on the 40X/128x128 pixel images. Overall accuracy ranges from 77-89% and the precision & recall values differ across benign and malignant classification, with malignant f1-scores ranging from 83-92% and benign between 62-80%.

For the final training, we trained the network for 100 epochs and compared the resulting accuracies across datasets. Test accuracies ranged from 77-89% with training dataset 40X/256x256px being the highest and 200X/128x128 the lowest (see Table 2). We saw a general trend of an inverse relation of magnification and accuracy, the higher the magnification, the worse the accuracies. The best accuracies in regard to image sizes were achieved on the 40X dataset.

Table Example classification report of DenseNet on test data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | precision | recall | f1-score | support |
|  |  |  |  |  |
| benign | 0.78 | 0.77 | 0.77 | 125 |
| malignant | 0.89 | 0.9 | 0.9 | 274 |
| accuracy |  |  | 0.86 | 399 |

Table 3 shows the precision, recall and f1-scores of the DenseNet trained on the original BreakHis dataset. In general, malignant images have been classified better in terms of precision, recall and f1 across all tested combinations.

## Synthesizing Breast Cancer Images using GANs

Next, we set out to create a pipeline GAN and use it to synthesize data based on the BreakHis dataset. Before we learned about the High-performance cluster (HPC) of the university, we were limited to our personal computers. This limitation was the basis for our decision to use simple GAN architectures instead of complicated ones. We implemented 3 different, although similar, architectures. Their performance had been tested on the MNIST [11] dataset (64x64 and 128x128) before usage.

The first one was based on the TensorFlow DCGAN [12] architecture trained on 256x256 scaled images, but failed to produce meaningful images, had a severe lack of variety even after training for 12,000 epochs or roughly 30 hours. Most importantly though, we faced Memory issues even on the HPC, which rendered this architecture unusable.

The second architecture [13] was used on 3 different image sizes (64px,128px and 256px) and trained for up to 150,000 epochs, but also failed to produce meaningful images. An example generated image (64x64) can be seen in Figure 5.

The last architecture [14] we tested, was trained on every single sub-dataset and for up to 100,000 epochs to ensure convergence. Although we did see said convergence up to a point, we also observed an unexpected divergence of the discriminator (d\_loss) and generator (a\_loss) losses (see Figure 6).

Following the loss plots, we chose appropriate model checkpoints for each subtype.

The models were able to generate images, which closely resembled the input data, although only 4 out of 8 subtypes were able to do so. The remaining models generated noise or otherwise unusable images. Fortunately, out of the 4 usable models, two were trained on benign (adenosis and phyllodes) and two on malignant (lobular and papillary) data. Fig. 7 shows a juxtaposition of an original and a synthesized image of the phyllodes tumor subset.

Although the quality of the synthetic images was better than expected the issue of missing variety within the generated datasets persisted. To analyze the variety, we calculated the FID score, which, simply put, compares the distribution of generated images with the distribution of a set of real images. A score close to 0 implies no difference between the datasets and higher scores meaning reduced similarity. We therefore compared each subset with their original counterpart and also performed one control, where we compared the same subset with itself. The results can be seen in Figure 8. As can be seen, our generated images had catastrophic FID-scores of well over 2500, meaning that our generated images do not resemble the original data well.

## Effect of Synthetic Data on DenseNet Performance

After having evaluated the DenseNet, we created 300 synthetic images for each tumor type: 150 each for benign and malignant, a total of 1200. Then we trained the DenseNet on three different datasets each containing the 4 tumor subtypes mentioned before:

* Original BreakHis dataset
* Mixed dataset (50% original, 50% synthetic)
* Synthetic dataset

After training on these datasets for 50 epochs, the resulting differences in training and validation accuracies and losses across the various training set types are shown exemplary for the synthetic dataset in Figure 9, but were similar for other datasets. Across all types, the validation accuracies converged towards 1 and training accuracies were close to 1.

Test accuracies, shown in Table 4, were not as high as training and validation accuracies and differed greatly between datasets with the network trained purely on synthetic data being the highest at 100%, whereas the network trained on mixed data and tested on original data showed the lowest accuracy of 57%. We could observe, that testing on the same type of data resulted in the best accuracies within a type and lowered gradually with reducing amounts.

Another observation is that the accuracy of the network trained on the mixed dataset is higher for testing with synthetic than original data. Also, when trained on mixed, the accuracy for synthetic data is higher than for mixed data, indicating a tendency to perform better on synthetic than original data. Interestingly, when trained on synthetic data, the accuracy is higher on original data, than when trained on mixed data.

Lastly, the remaining metrics, precision, recall and f1-score, are shown in Table 5. Similar to other examples shown before, some combinations show a tendency to favour either precision or recall for either benign or malignant data, seemingly in an effort to reduce Type I or II errors. As before, networks trained on mixed data, seem to be better at classifying synthetic data than mixed or original.

# Discussion

First, it was apparent, that our choice of GAN architecture was flawed and not well suited for our task at hand, although the complexity of our data seemed comparable to tested datasets. The biggest drawback, after overcoming resource and architectural limitations, was the fact that we could not generate diverse enough data. Literature research showed that other groups [15] were able to access transform the latent dimension necessary to produce varying output.

Although we were not able to produce high quality images using a simple GAN architecture, we nonetheless could show some effect of synthetic training data on the performance of the DenseNet. Generally speaking, testing on data similar to training data yields better results with one big exception being the DenseNet, which was trained on mixed data showing a strong preference for synthetic data. Concomitantly, networks trained on synthetic data fared best when testing on mixed data across types, as well as that it performed better on original data, than the synthetic counterpart.

Another interesting finding was that the network trained on synthetic data performed better when tested with original data than the other way around. To us, this seemed to suggest, that our synthetic data was somewhat useful, although it is likely due to the few images, which were close to a copy of the original data.

In general, the DenseNet, when working only with the original dataset (see Tables 2 and 3) classified malignant images better than benign. This could be a wanted feature, since it is better to falsely diagnose a healthy person with cancer, than to miss it. Worst case scenarios for false positive errors are additional examinations and follow-up tests, whereas false negative errors could potentially lead to delayed or missed therapy, resulting in deterioration of life quality and possibly death of the patient.

Regarding the training and validation accuracies, they seemed to show an overfit, which was strikingly apparent for networks trained and tested on synthetic data, with 100% accuracy and an f1 of 1. We could have stopped the training early to counteract that.

Research groups have shown that GANs may successfully be used to synthesize histopathological images. For example PathologyGAN, which builds upon BigGAN [16] and reaches a FID-score between 9-32 across 3 datasets, compared to our scores of 2500 and more. It is apparent, that the size of their dataset, over 400,000 images, as well as their personnel, computational and temporal resources, far exceeded our means. Therefore, a direct comparison should be taken with a grain of salt. Our initial approach of using a simple GAN was mostly due to inexperience with the technology and limitations on time and computational resources. Given more time and development, we are confident, that we could have additionally implemented existing GANs to further investigate the effect on evaluation performance of the DenseNet.

Lastly, we would like to mention, that we learned a lot throughout this whole endeavour, which was full of up and downs, but in the end quite rewarding.

##### Acknowledgment

We would like to thank Dr. Isabel Dregely and   
Dr. Matthias Blaickner for their constructive help during development. Further, we wanted to express our thankfulness to our partners respectively. Without their support this work would not have been finished. Finally, the FHTW for letting us use the HPC extensively and free of charge.

# References

|  |  |
| --- | --- |
| [1] | Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature,* vol. 521, p. 436–444, May 2015. |
| [2] | I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative Adversarial Networks," *Commun. ACM,* vol. 63, p. 139–144, October 2020. |
| [3] | A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta and A. A. Bharath, "Generative Adversarial Networks: An Overview," *IEEE Signal Processing Magazine,* vol. 35, pp. 53-65, 2018. |
| [4] | X. Yi, E. Walia and P. Babyn, "Generative adversarial network in medical imaging: A review," *Medical Image Analysis,* vol. 58, p. 101552, December 2019. |
| [5] | L. Cai, J. Gao and D. Zhao, "A review of the application of deep learning in medical image classification and segmentation," *Annals of Translational Medicine,* vol. 8, June 2020. |
| [6] | G. Huang, Z. Liu, L. van der Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," *arXiv,* August 2016. |
| [7] | J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009. |
| [8] | F. A. Spanhol, L. S. Oliveira, C. Petitjean and L. Heutte, "A Dataset for Breast Cancer Histopathological Image Classification," *IEEE Trans. Biomed. Eng.,* vol. 63, p. 1455–1462, October 2015. |
| [9] | F. A. Spanhol, L. S. Oliveira, C. Petitjean and L. Heutte, "Breast cancer histopathological image classification using Convolutional Neural Networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2016, p. 2560–2567. |
| [10] | "Papers with Code," [Online]. Available: https://paperswithcode.com/method/densenet. |
| [11] | L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine,* vol. 29, p. 141–142, 2012. |
| [12] | "Tensorflow DCGAN," [Online]. Available: https://www.tensorflow.org/tutorials/generative/dcgan. |
| [13] | "kaggle," [Online]. Available: https://www.kaggle.com/code/sayakdasgupta/fake-faces-with-dcgans. |
| [14] | "Kaggle," [Online]. Available: https://www.kaggle.com/code/nageshsingh/generate-realistic-human-face-using-gan. |
| [15] | A. C. Quiros, R. Murray-Smith and K. Yuan, "PathologyGAN: Learning deep representations of cancer tissue," *Machine Learning for Biomedical Imaging (MELBA),* 2021. |
| [16] | A. Brock, J. Donahue and K. Simonyan, "Large Scale GAN Training for High Fidelity Natural Image Synthesis," *arXiv,* September 2018. |
| [17] | G. Huang, Z. Liu, L. van der Maaten and K. Q. Weinberger, *Densely Connected Convolutional Networks,* arXiv, 2016. |