Project Title: Creation and evaluation of synthetic data using GAN technique in medical applications

Authors

Dhananjay Mukhedkar <dhamuk@kth.se>
Yeongwoo Kim <yeongwoo@kth.se>

Abstract

Convolutional Neural Network (CNN) for image classification has been applied to classify data and decreased the number of weights compared to Artificial Neural Network (ANN) for image classification. The reduction in parameters is accomplished by the architecture of CNN, which is comprised of the convolution of layers for reducing dimensionality. However, as CNN grows larger, the number of weights still increases, and the necessity of more training dataset increases. In case of a small dataset, CNN models prone to have low accuracy, which prevents a model from deployment. Therefore, this paper tried to explore generative modeling to augment a dataset and how the augmented data affects the accuracy of CNN. As for the generative modeling, Generative Adversarial Network (GAN) was selected, since the technique creates synthetic data for wide applications without domain knowledge. Such synthetic or augmented data would be useful in applications where there is a shortage of data, or reducing complexity in handling high-resolution data is needed. This research is focused on using GAN for medical applications in skin cancer lesions detection.

Keyword: Machine Learning, CNN, GAN, Medical data, Data augmentation, Skin cancer

Table of contents

1. Introduction

Image classification and recognition have been improved as CNN is introduced, and one of the most interesting fields was the medical domain. For machine learning models like CNN, it is essential to have sufficient and accurate data. However, such data is not always easy to access or to handle in case of medical applications where there are several challenges as follows: 1) accessing data has privacy issues, but the data have a large number of features 2) images with high resolution are computationally heavy to handle [1].

These challenges degrade the performance of CNN, though the accuracy of prediction is highly important in such a field. Due to this, image augmentation and synthetic data creation are being researched and investigated. Currently, though data augmentation techniques exist, they are labor-intensive and require domain knowledge. Recently, as neural networks have been researched as generative models, Generative Adversarial Network (GAN) has become one of the promising techniques for synthetic data generation and generative modeling. GAN is finding applications across various domains, and in this research, we investigate its applications in the medical domain for generating images used for skin cancer classification.

In this paper, we examined that the image augmentation by GAN would improve accuracy of CNN. To be specific, there are two parts of hypotheses of this paper we clarified: 1) GAN model is simply applied to any input data without domain knowledge and generates synthetic images as data for skin cancer cell detection models using CNN, 2) The output data to be evaluated against an existing CNN model trained on real data and evaluate the impact on its performance and compare results.

2. Related works

Computer-aided diagnosis has been discussed for years, and there are papers regarding the improvement of classification accuracy [2]. However, traditionally the medical domain has its limitation since the privacy of data prevents researchers from collecting datasets. The limitation led researchers to work on a small dataset.

Due to the lack of availability of sufficient data, different techniques were applied to learn the features from the limited data. According to Maciej A. Mazurowski at la, the backpropagation over particle swarm optimization was proven to classify medical data efficiently [1]. However, this was not a fundamental solution to the small size of the dataset. In order to overcome the insufficient size of a dataset, the data augmentation techniques, such as noise addition, cropping, or interpolation, were explored. However, they are domain-specific, and different methods were proven to be efficient. In the worst case, the image augmentation led a classification model to overfitting where a model should avoid [3]. However, augmentation techniques are required in the medical field, and attempts to generate new data were devised. Therefore, linear combinations of training images were experimented and gave positive results [4]. Zeshan Hussain et al. showed a comparison between different augmentation strategies and the extent to which an augmented training set retains properties of the original medical images, which determines model performance [8].

Recently, new techniques in generative modeling such as GAN are utilized to augment images. GAN was firstly deployed by Ian J. Goodfellow et al. [5], which allowed a network to learn the features and characteristics of input data and make synthetic data. The images were genuine, as the synthetic image differs from the original images. Our work focused on leveraged the GAN technique to augment input images. This technique would be readily applied to various cases since the synthetic data could be regarded as a training dataset to train CNN. This would overcome the downside of the traditional image augmentation, which had to be chosen carefully with domain knowledge.

3. Methods

3.1. Data and Tools

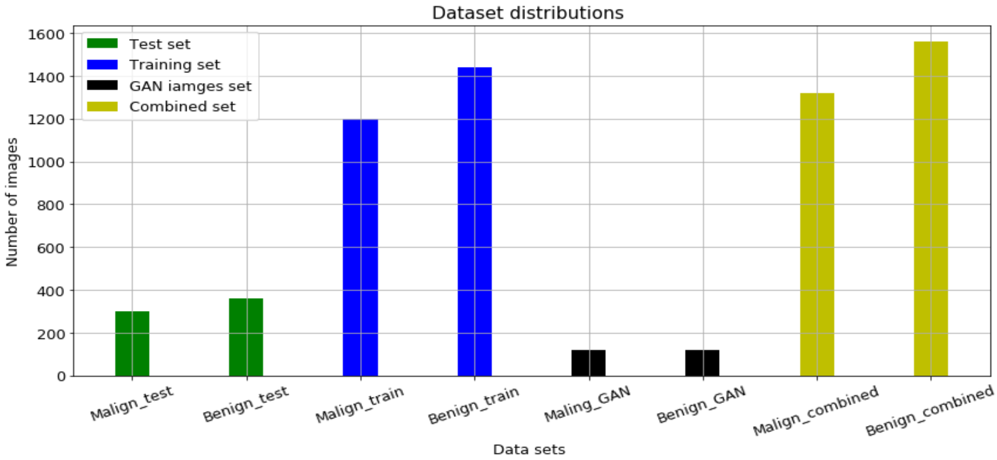

Figure 1. Size of different datasets used

For this research, two-class classifications of skin cancer tumor cells, malign or benign is performed. We used an open-source dataset published by ISIC. The dataset consists of real skin cancer images, and the distribution of the dataset is in Figure 1. When it comes to the frameworks, we used Pytorch for the generation of GAN, Keras for training our model, and the Google Colab platform for execution.

The dataset is divided into a training set (80%) and validation set using cross-validation (20% of the training set) and test a model performance on a separate test set (20%). First, a model is trained by the real images, and its performance is tested on the test set as a benchmark. Subsequently, GAN is trained to create synthetic data which are mixed with the real images to form a combined amalgamated dataset. Here, the synthetic data is not mixed with the test set, since the accurate evaluation of CNN has to be measured by real dataset images.

Subsequently, the same CNN model is trained on this combined dataset and finally evaluate its performance on the common test set used earlier. For evaluation different metrics such as accuracy, F1-score, confusion matrix, sensitivity and specificity are applied. Figure 1 shows the data distribution we use after all the pre-processing. The mixing of fake data was done in different proportions. Firstly, 240 fake images and then 880 fake images are generated from the trained GAN, and these fake images are amalgamated into the training set to form the two combined training set of real and synthetic images.

3.2. GAN Architecture

The applied GAN structure is Deep Convolutional Generative Adversarial Network (DCGAN) since the DCGAN can learn the features of original images and generate synthetic data. This network is built by Pytorch, and the details of the network structure are in Figure 2.

In our work, a pre-built code of DCGAN is applied to our work [6], but we modified the structure of DCGAN from 8 hidden layers to 7 hidden layers. This modification requires reconstruction of output layer of generator and input layer of discriminator. By this modification, we can train GAN and CNN fast. The behavior of DCGAN is as follows: 1) Deconvolutional Network (DN) generates a synthetic image from a random noise, 2) CNN classifies the synthetic image, and gives feedback to the DN and CNN by backpropagation with decaying gradient descent [6].

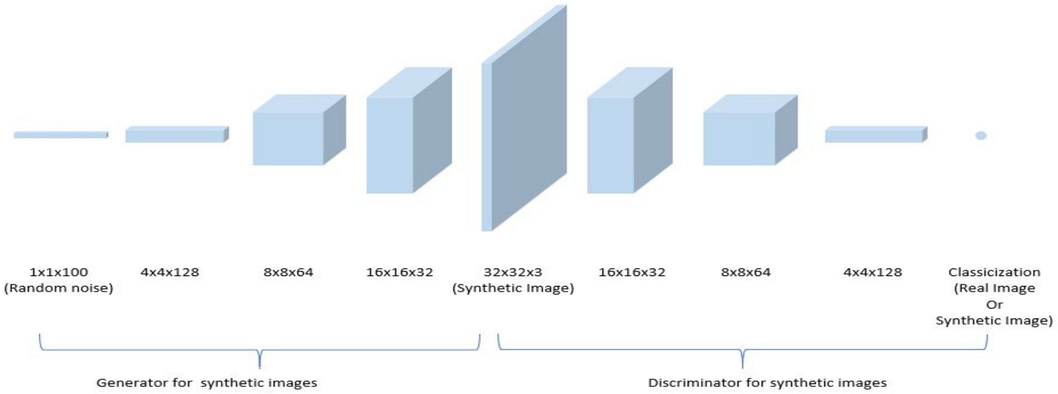

Figure 2. Structure of GAN

In our case, the data is divided into two classes: malignant and benign. According to the classes, DCGAN is made for each malignant and benign lesion, as two networks having an identical structure of Figure 2 are built. One network generates synthetic images of malignant lesions, and the other network generates synthetic images of benign lesions.

DN generates a bigger tensor by applying weights and nonlinear function to each entry of noise. This means that the random noise can be regarded as a unique identifier of a synthetic image. Thus, after the GAN model is fully trained by backpropagation, the unique identifier is changed randomly in order to generate different images.

3.3. CNN Architecture

We built a simple CNN model with four convolution layer followed by one fully connected layer as the output layer. Dropout was used for regularization. Though the discriminator at GAN is CNN, the purpose of CNN at GAN is the differentiation between real and synthetic images. On the other hand, this CNN is not to differentiate real or synthetic images, but to diagnose diseases such as skin cancer. Synthetic images generated in the middle of Figure 2 were extracted and supplied to this CNN at the input layer of Figure 3 with real images. The CNN architecture in Figure 3 was trained first on real images and next on real and synthetic images. The results of both models were compared by evaluating on a common test set. The architecture of both CNN models was unchanged for fair comparison.

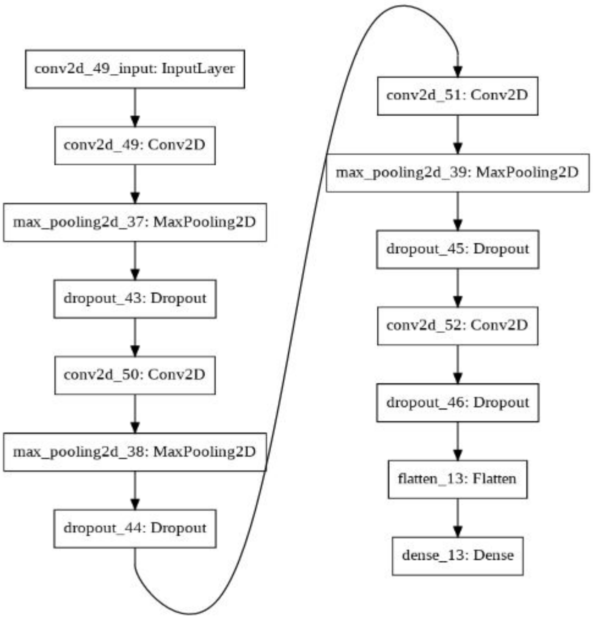

Figure 3. Structure of skin cancer classification CNN

4. Results

4.1. Performance of GAN

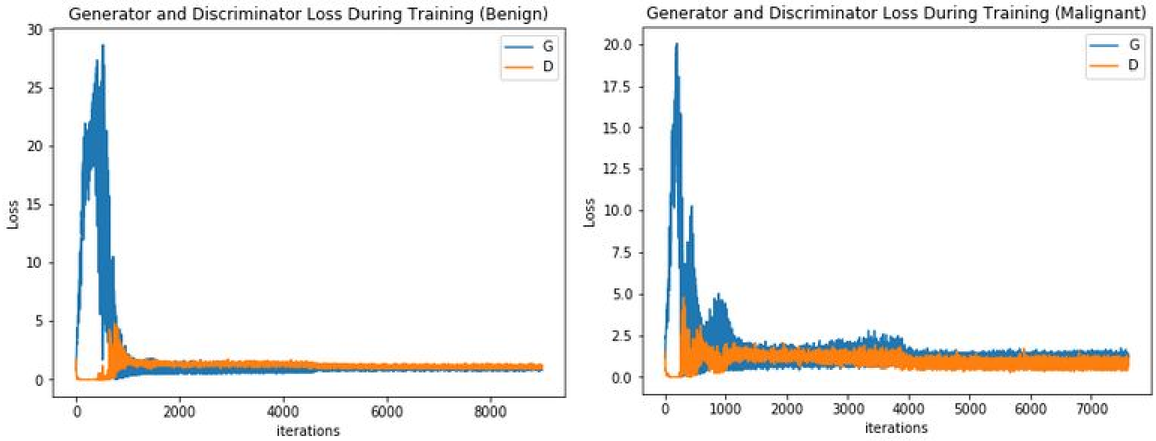

Figure 4. Training of GAN

Training of GAN is adversarial between generator and discriminator, as described in Figure 2. At the beginning of training as in Figure 4, the generators of benign and malignant images made erroneous images, and the discriminator could easily discriminate synthetic images. Therefore, the average of cross-entropy loss of generator (G) is 19.98, while cross-entropy of discriminator (D) is zero. However, as the networks updated themselves iteratively, the generators formed realistic synthetic images, and deceived corresponding discriminator. This, in turn, decreased average of cross entropy loss of generator to 0.89, and introduced the loss to discriminator to 1.11 in average, since the discriminators classified the synthetic images as real images. These errors triggered backpropagation in GAN.

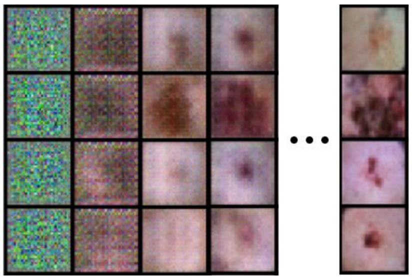

Figure 5. Improvement GAN, random noise (far left), Ends of updates (far right)

As the backpropagation algorithm updated weights of GAN structures, the images made by GAN became realistic as Figure 5. The image on the far left is an image without any backpropagation, so it shows a random noise. As it went through backpropagation, the image slowly formed images. After the updates of the weights, it showed the images in Figure 6.

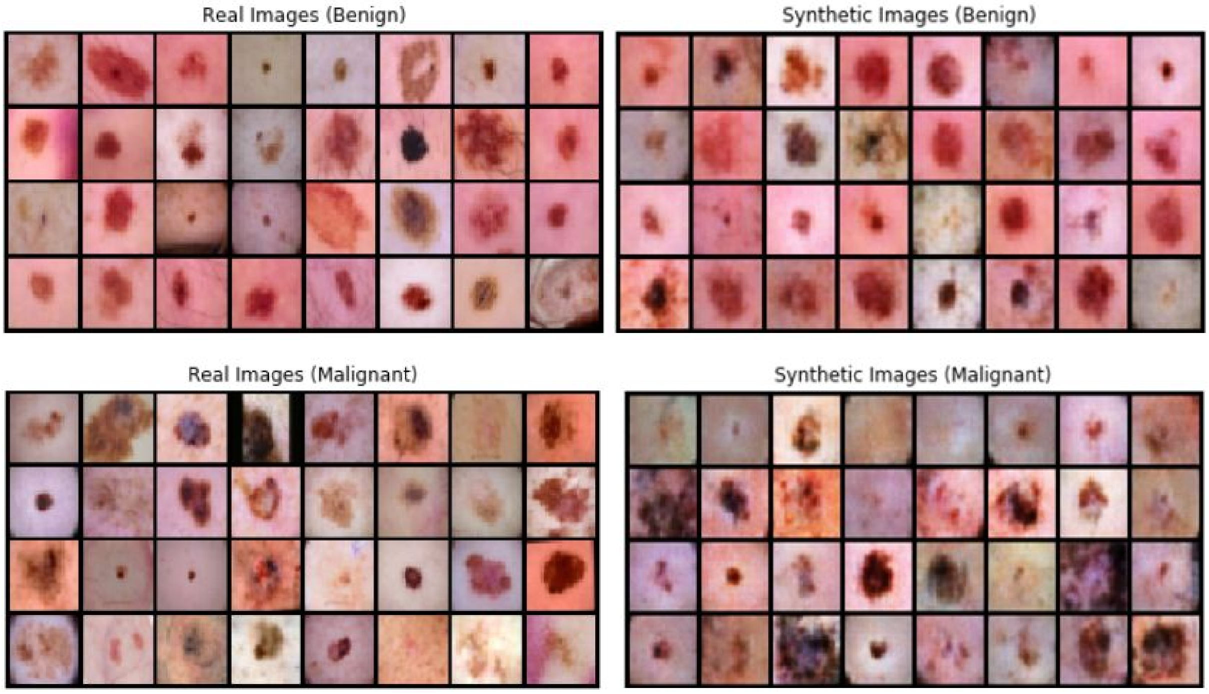

Figure 6. Real images (left) Synthetic images (right)

In this phase, it is not clear to distinguish synthetic images from the original images by human perception. Since each neural network learns the features and traits of either benign or malignant images, it generates the corresponding images. For example, the synthetic data shows broader black spots on the skin.

4.2. Performance evaluation of CNN

Below are the classification reports of model performance on the common test set and the corresponding confucsion matrix figures.

4.2.1. Original data

Table 1. Full classification report: CCN original data

	Precision	**Recall**	**F1-score**	**Support**
Malignant	0.70	0.96	0.81	300
Benign	0.95	0.66	0.78	360
Accuracy			0.80	660
Macro average	0.83	0.81	0.80	660
Weighted average	0.84	0.8	0.79	660
Specificity	0.96			
Sensitivity	0.66			

Table 2. Confusion matrix: CCN original data (accuracy: 80%)

	Predicted: Malignant	**Predicted: Benign**
Actual : Malignant	288	12
Actual : Benign	122	238

🟩 : Correct prediction 🟧 Misclassification

In this case, we trained CNN with real data. The result of the training only with original data is used as a base criterion to compare with other CNNs which would be trained by the combination with synthetic data. Table 1 shows the various performance metrics of the test set from the original data for both cases: malignant and benign. The accuracy of this case is 80%, and the actual number of predictions is available in the confusion matrix in Table 2.

4.2.2. Original + GAN data combined (120 GAN images per class)

Table 3. Full classification report: CCN original and synthetic data

	Precision	**Recall**	**F1-score**	**Support**
Malignant	0.74	0.94	0.83	300
Benign	0.94	0.73	0.82	360
Accuracy			0.83	660
Macro average	0.84	0.84	0.83	660
Weighted average	0.85	0.83	0.83	660
Specificity	0.94			
Sensitivity	0.72			

Table 4. Confusion matrix: CCN original and synthetic data (accuracy: 83%)

	Predicted: Malignant	**Predicted: Benign**
Actual : Malignant	283	17
Actual : Benign	98	262

🟩 : Correct prediction 🟧 Misclassification

In table 3 and 4, 120 synthetic data generated by GAN per class were mixed into training data. The additional data by GAN increased the accuracy by 3%, since the accuracy of this case is 83%. We could observe that the recall rate for benign class increased from 66% to 73%, whereas the recall for malign class dropped from 96% to 94% in Table 3.

4.2.3. Original + GAN data combined (480 GAN images per class)

Table 5. Full classification report: CCN original and synthetic data

	Precision	Recall	F1-score	Support
Malignant	0.80	0.86	0.83	300
Benign	0.88	0.82	0.84	360
Accuracy			0.84	660
Macro average	0.84	0.84	0.84	660
Weighted average	0.84	0.84	0.84	660
Specificity	0.86			
Sensitivity	0.82			

Table 6. Confusion matrix: CCN original and synthetic data (accuracy: 84%)

	Predicted: Malignant	Predicted: Benign
Actual : Malignant	258	42
Actual : Benign	66	294

🟩 : Correct prediction 🟧 Misclassification

480 synthetic images generated by GAN per class were mixed into training data, The overall accuracy increased by 4%, as the accuracy of this case was 84%. The recall rate for benign class increased from 66% to 82%, whereas the recall for malignant class dropped from 96% to 86%. One reason for this could the data imbalance as we have more samples in benign class than in malignant class.

Overall, synthetic data from GAN improved the accuracy of diagnosis. However, the increment of accuracy was not proportional to the number of synthetic data, since 120 GAN images improved 3% and 480 GAN images improved 4%. Moreover, this technique showed changes in recalls. Since false negative is vital in the medical domain, decrement of recall in malignant class is critical. When we look into the case with 120 synthetic data, the recall of malignant class is 94% and the accuracy is 83%, while in the case of 480 synthetic data, the recall is 86% and accuracy is 84%. Therefore, we could conclude the 120 synthetic data is more preferable, as considering recall and the medical domain.

5. Discussion and conclusion

5.1. Discussion about hypothesis

Regarding our two initial hypotheses, GAN is proved to work without domain knowledge and generates synthetic images. A qualitative measure of GAN with the full medical examination is hard to describe since dermatologists are required. However, as an engineering method, the losses of generator and discriminator in Figure 4 are converged to values, showing that the generator constantly deceives the discriminator. Therefore, the convergence of the network can be regarded as the effectiveness of GAN.

For the second part of the hypothesis, comparing the results of the model on the real and combined data shows different results with a different number of synthetic images mixed. The overall accuracy of the model increased by 3% with 120 fake images and 4% with 480 fake images in the case of the combined dataset. The recall rate for benign increased with the synthetic images, but the recall for malign decreased. We attributed this to traits of lesions and GAN and imbalance in training set with more benign class images. The most noticeable difference between benign and malignant lesions is the size of black spots. This means when synthetic data have broad black spots labeled as benign, CNN trained by the synthetic data will regard broad black spots as benign. GAN has a chance to generate a broad black region by a non-linear combination of small moles because GAN is trained by benign moles or black spots.

However, overall this shows promising results. It can be inferred that the same model has better generalization performance compared to the model only trained on real data. Also, it was observed that the performance was varying as the model architecture was changed. As the model became more complex, we did not observe better results with synthetic data; in fact, it degraded the performance. One reason could be the overfitting of the model to the available data as the model becomes complex and relatively small size of the dataset. Further investigation can be made to understand the relation between model complexity and the performance on augmented data.

To sum up, image classification by CNN has solved many problems that cannot be done by traditional image detection algorithms. However, the problem of CNN and machine learning techniques is the large size of the input dataset. When the dataset is small, the accuracy of prediction by a machine learning algorithm was limited. By leveraging GAN, we believe data augmentation will be possible and contribute to expanding the area of machine learning techniques.

5.2. Future works

The next step of this research would be execution and experiments of the strategy on a bigger dataset and multiclass classification problems, for instance, skin cancer classification with eight classes. Another approach would be to compare traditional data augmentation technique results with GAN generated data.

With more tumor classes and data, we can estimate the percentage of synthetic data that should be mixed to real datasets to improve results. Such work can be particularly helpful for skin cancer data journals and archives like the International Skin Collaboration.

Furthermore, it has to be researched more in-depth to see the misdiagnosis caused by synthetic data. In detail, as we can see in the related works, the research regarding the synthetic data is ongoing and has shown the improvement of accuracy [4]. From the papers, we can deduce the validity and necessity of data augmentation in the medical domain.

References

[1] Mazurowski, M.A., Habas, P.A., Zurada, J.M., Lo, J.Y., Baker, J.A. and Tourassi, G.D., 2008. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural networks*, *21*(2-3), pp.427-436.

[2] Kawahara, J., BenTaieb, A. and Hamarneh, G., 2016, April. Deep features to classify skin lesions. In *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)* (pp. 1397-1400). IEEE.

[3] DeVries, T. and Taylor, G.W., 2017. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538*.

[4] Eaton-Rosen, Z., Bragman, F., Ourselin, S. and Cardoso, M.J., 2018. Improving data augmentation for medical image segmentation.

[5] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672-2680).

[6] Nathan Inkawhich*, "DCGAN tutorial",*
Available at: https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html

[7] Ruder, S., 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747.*

[8] Hussain, Z., Gimenez, F., Yi, D. and Rubin, D., 2017. Differential data augmentation techniques for medical imaging classification tasks. In *AMIA Annual Symposium Proceedings* (Vol. 2017, p. 979). American Medical Informatics Association.

Appendix

The link for the dataset used in the paper:
https://www.kaggle.com/fanconic/skin-cancer-malignant-vs-benign

Full structure of GAN:

```
Generator(
  (main): Sequential(
    (0): ConvTranspose2d(100, 128, kernel_size=(4, 4), stride=(1, 1),
bias=False)
    (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): ConvTranspose2d(128, 64, kernel_size=(4, 4), stride=(2, 2),
padding=(1, 1), bias=False)
    (4): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): ConvTranspose2d(64, 32, kernel_size=(4, 4), stride=(2, 2),
padding=(1, 1), bias=False)
    (7): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (8): ReLU(inplace=True)
    (9): ConvTranspose2d(32, 3, kernel_size=(4, 4), stride=(2, 2),
padding=(1, 1), bias=False)
    (10): Tanh()
  )
)
Discriminator(
  (main): Sequential(
    (0): Conv2d(3, 32, kernel_size=(4, 4), stride=(2, 2), padding=(1,
1), bias=False)
    (1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.2, inplace=True)
    (3): Conv2d(32, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1,
1), bias=False)
    (4): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (5): LeakyReLU(negative_slope=0.2, inplace=True)
    (6): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1,
1), bias=False)
    (7): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (8): LeakyReLU(negative_slope=0.2, inplace=True)
    (9): Conv2d(128, 1, kernel_size=(4, 4), stride=(1, 1), bias=False)
    (10): Sigmoid()
  )
```

)

Full structure of CNN:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_49 (Conv2D)           (None, 32, 32, 32)        896
_________________________________________________________________
max_pooling2d_37 (MaxPooling (None, 16, 16, 32)        0
_________________________________________________________________
dropout_43 (Dropout)         (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_50 (Conv2D)           (None, 15, 15, 64)        8256
_________________________________________________________________
max_pooling2d_38 (MaxPooling (None, 8, 8, 64)          0
_________________________________________________________________
dropout_44 (Dropout)         (None, 8, 8, 64)          0
_________________________________________________________________
conv2d_51 (Conv2D)           (None, 7, 7, 128)         32896
_________________________________________________________________
max_pooling2d_39 (MaxPooling (None, 3, 3, 128)         0
_________________________________________________________________
dropout_45 (Dropout)         (None, 3, 3, 128)         0
_________________________________________________________________
conv2d_52 (Conv2D)           (None, 2, 2, 128)         65664
_________________________________________________________________
dropout_46 (Dropout)         (None, 2, 2, 128)         0
_________________________________________________________________
flatten_13 (Flatten)         (None, 512)               0
_________________________________________________________________
dense_13 (Dense)             (None, 1)                 513
=================================================================
Total params: 108,225
Trainable params: 108,225
Non-trainable params: 0
_________________________________________________________________
```