DOI: https://doi.org/10.15276/aait.03.2021.4
UDC 004.932: 616-006.04: 004.822

Comparison of generative adversarial networks architectures for biomedical images synthesis

Oleh M. Berezsky[1)]
ORCID: https://orcid.org/ 0000-0001-9931-4154; ob@wunu.edu.ua. Scopus ID: 6505609877
Petro B. Liashchynskyi[1)]
ORCID: https://orcid.org/0000-0002-3920-6239; p.liashchynskyi@st.wunu.edu.ua
[1)] West Ukrainian National University. 11, Lvivska St. Ternopil, 46009, Ukraine

ABSTRACT

The article analyzes and compares the architectures of generative adversarial networks. These networks are based on convolutional neural networks that are widely used for classification problems. Convolutional networks require a lot of training data to achieve the desired accuracy. Generative adversarial networks are used for the synthesis of biomedical images in this work. Biomedical images are widely used in medicine, especially in oncology. For diagnosis in oncology biomedical images are divided into three classes: cytological, histological, and immunohistochemical. Initial samples of biomedical images are very small. Getting training images is a challenging and expensive process. A cytological training dataset was used for the experiments. The article considers the most common architectures of generative adversarial networks such as Deep Convolutional GAN (DCGAN), Wasserstein GAN (WGAN),Wasserstein GAN with gradient penalty (WGAN-GP), Boundary-seeking GAN (BGAN), Boundary equilibrium GAN (BEGAN). A typical GAN network architecture consists of a generator and discriminator. The generator and discriminator are based on the CNN network architecture. The algorithm of deep learning for image synthesis with the help of generative adversarial networks is analyzed in the work. During the experiments, the following problems were solved. To increase the initial number of training data to the dataset applied a set of affine transformations: mapping, parallel transfer, shift, scaling, etc. Each of the architectures was trained for a certain number of iterations. The selected architectures were compared by the training time and image quality based on FID (Freshet Inception Distance) metric. The experiments were implemented in Python language. Pytorch was used as a machine learning framework. Based on the used software a prototype software module for the synthesis of cytological images was developed. Synthesis of cytological images was performed on the basis of DCGAN, WGAN, WGAN-GP, BGAN, BEGAN architectures. Google's online environment called Collaboratory was used for the experiments using Nvidia Tesla K80 graphics processor.

Keywords: Deep learning; generative adversarial networks; biomedical images; images synthesis

***For citation*: Berezsky O. M., Liashchynskyi P. B. Comparison of generative adversarial networks architectures for biomedical images synthesis. *Applied Aspects of Information Technology.* 2021; Vol.4 No.3: 250–260. DOI: https://doi.org/10.15276/aait.03.2021.4**

INTRODUCTION

Biomedical images are a wide class of images that are used for diagnosis. Among those images, there is a subclass of cytological, histological, and immunohistochemical images which are used for diagnosis in oncology. Obtaining those images is time consuming and quantitatively limited. As a result, modern classifiers, which are based on convolutional neural networks [1], face the problem of small datasets. One of the consequences of using a small dataset of training data is the relatively low accuracy of classification. That is why the expansion of datasets artificially for the training of modern classifiers is the actual task. Nowadays, generative adversarial networks are popular methods of image synthesis. Generative Adversarial Networks (GANs) are the type of neural network used to synthesize images, video, and audio. This type of network has appeared very recently. In 2014, researcher Ian Goodfellow demonstrated how new data can be synthesized using neural networks [2, 3]. For the first time, these networks were used to synthesize handwritten numbers (using MNIST dataset) and human faces (using CIFAR-10 dataset).

1. LITERATURE ANALYSIS

For now, possibilities of generative adversarial networks are still studied by many researchers [4]. However, the most common application of GAN is image synthesis. Most of the research in this area is aimed at improving the quality of synthesized images. Researchers create new architects and improve already known algorithms [5, 6]. Generative adversarial networks have also found their application in medicine.

Recent studies have shown that artificial neural networks can achieve better results in segmentation and classification than humans [7]. However, modern classifiers require a lot of training data. Because the process of collecting medical data can be complicated due to the influence of many factors (time, price, availability, etc.), GAN-networks are primarily used to expand training datasets [8, 9]. In this case, it is also possible to increase the number of samples in dataset with the help of affine transformations [10]. But this method is not suitable if there is a need to generate a large amount of data, because the images created in this way are not completely new, but are only distorted versions of the existing ones.

In [12] researchers use the GAN network to

synthesize images of cell nuclei and their masks for further segmentation. They divide training process into two different stages. In the first step, they train a generator to synthesize binary masks from a latent vector. In the second step, they use a conditional generative adversarial network to synthesize images based on the generated masks and the latent vector. In the end, the authors get two generators. One for mask synthesis, the other for mask-based image synthesis.

The authors of [13] use the GAN to synthesize images of affected liver tissues that have a resolution of 64 by 64 pixels. Experimental results of the classification showed that the accuracy increased from 77 % to 85 % when using a training dataset, which was expanded with synthesized images.

A typical GAN network architecture consists of two different neural networks that play against each other. The first network is called a discriminator (denoted by the letter D). Its task is to verify the plausibility of images – distinguish real images from artificially synthesized. Real images are data from the training samples. The discriminator accepts the image as the input, and as the output produces the probability that the image is real. Usually the discriminator is a simple binary classifier. The second network is called a generator (denoted by the letter G). The task of the generative network is to synthesize new data. The generator picks up a random vector of a certain dimension from a certain latent space (usually from a normal distribution) and tries to convert it into a real image [6, 13].

The basic work principle of any GAN can be described as follows. The discriminator must distinguish between real and synthesized data. Based on a certain loss function the discriminative model should maximize the probability of real data and minimize the probability of synthesized. Similarly, the generator must synthesize images that are as close as possible to the real ones, so that the discriminator classifies them as real data. As the conclusion, the generative model tries to maximize the probability of the synthesized images.

Training algorithm of any GAN network performs parameter optimization for both disciminator and generator.

The purpose of the training is to minimize the next loss function $V(G,D)$, where E is the expected value.

$$\min_G \max_D V(G,D) = E_{x\sim p_{data}(x)}[\log D(x)] + \\ + E_{z\sim p_z(z)}[\log(1-D(G(z)))].$$

In the perfect scenario, the discriminator is trained until it achieves its optimal value depends on the concrete generator, the generator is trained after the discriminator. But in practice the discriminator could be trained for the specific number of iterations, then the generator is updated along with the discriminator. Also, as a good alternative for the generator the next equation is used $\max_G \log D(G(z))$ instead of the $\min_G \log(1-D(G(z)))$ [6].

The generalized GAN algorithm could be descibed as follows. In the first step the generator takes a complete random sample from noise prior (Gaussian, for example) and tries to generate an image. This image is passed through the discriminator which task is to classify if the passed image is either the real one or generated.

Algorithm. Minibatch stochastic gradient descent algorithm of GANs. The number of steps to apply to the discriminator, k, is a hyperparameter [2].

for number of training iterations **do**
 for k steps **do**
1) Sample minibatch of m noise samples $\{z^{(1)},...,z^{(m)}\}$ from noise prior $p_g(z)$
2) Sample minibatch of m samples $\{x^{(1)},...,x^{(m)}\}$ from real data distribution $p_{data}(x)$.
3) Update the disciminator by ascending its gradient:

$$\nabla_{\theta_d} \sum_{i=1}^{m} [\log D(x^{(i)}) + \log(1-D(G(z^{(i)})))].$$

end for

1) Sample minibatch of m noise samples $\{z^{(1)},...,z^{(m)}\}$ from noise prior $p_g(z)$.
2) Update the generator by descending its gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} [\log(1-D(G(z^{(i)})))].$$

end for

Training of GAN networks is a quite complicated process because of so-called zero-sum game. When the discriminator starts better distinguish generated images from the real ones, the generator also needs to perform better. It works back and forth. When the generator is better, the discriminator must catch up.

Model training consists of two steps: training of the discriminator and updating parameters of the generator with concerning the discriminator. On the

first step of training only disciminator's parameters are updated. On the second step discriminator's parameters are kept frozen, instead only generator's parameters are updated.

Classical GAN networks have their cons. The first problem is gradient vanishing [14]. Because the generator is in feedback with the discriminator, it can only update its parameters through it. The gradients become so small that they have almost no effect on the parameters of the generator and the learning process stops. This problem occurs when the discriminator error quickly converges to zero. This also could occur when the discriminator model is much more powerful than the generator. When building an architecture, it is important to balance the complexity of the discriminator and generator models [14].

2. THE PURPOSE OF THE ARTICLE

The purpose of this paper is to analyze and compare generative adversarial networks architectures for biomedical images synthesis that will help us choose which network generates the best visual quality images. To achieve this goal, the following main objectives of the study were solved:

1) analyzing known architectures of generative adversarial networks;

2) perfoming experiments on real data using known architectures;

3) analyzing the results of the experiments.

3. GAN ARCHITECTURES OVERVIEW

All GAN architectures could be divided into 3 main types – fully connected, convolutional, and conditional GANs. By combining them and using various optimizations, researchers obtain new architectures that outperform each other very well. For now, there are many different architectures, using which researchers can achieve very good results.

DCGAN. Deep Convolutional Generative Adversarial Network was first used and proposed in 2014 [2]. The authors used convolutional neural network (CNN) for both generator and discriminator.

The main advantage of using CNN over fully-connected architecture is that CNN takes images as input. Also, the architecture itself is built in a more optimal way. This approach has significantly reduced the number of parameters, training time and increase the accuracy of such networks [1].

Modern GAN architecture may consist of convolution, pooling and ReLU layers. The most important part of any neural network is activation function. The most popular one which is often used with CNN and GAN network is called ReLU (Rectifier Linear Unit). The working principle is very straight: if the input value is less than 0 it outputs 0, otherwise the input value becomes new output $f(x) = \max(x, 0)$. The generator and discriminator use different final activation function: tangent and sigmoid respectively. General example of DCGAN is shown in Fig. 1.

WGAN and WGAN-GP. A common problem in the training of GANs is the so-called mode collapse [15]. When it happens, the generator begins to synthesize the same images regardless of the input data. The WGAN architecture allows to get rid of this problem partially [16]. An innovation in this architecture is the application of a new loss function based on the Wasserstein distance. The advantage of this architecture is that the discriminator is replaced by the critic (Fig. 2).

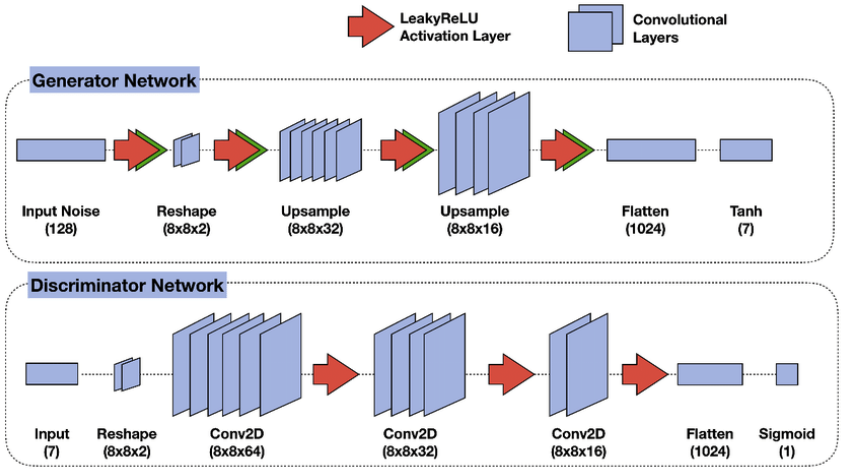


Fig. 1. **Simple DCGAN generator and discriminator**

Source: https://www.researchgate.net/publication/335341342_DijetGAN_a_Generative-Adversarial_Network_approach_for_the_simulation_of_QCD_dijet_events_at_the_LHC

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

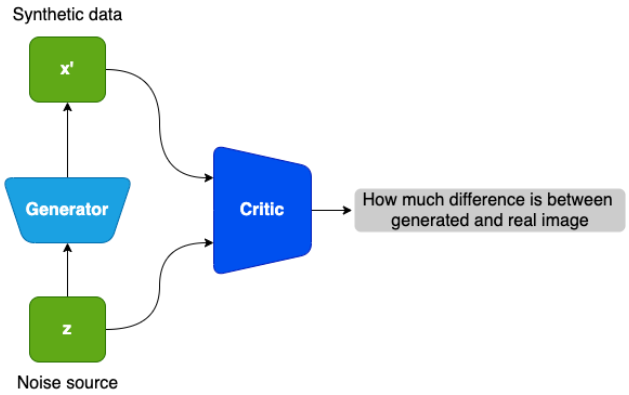


Fig. 2. **WGAN and WGAN-GP structure**
Source: compiled by the authors

Thus, it does not try to classify the image as real or synthesized anymore, but shows a certain error of how different the synthesized image is from the real one. The trainig process is about to minimize this value. But this architecture also has its drawbacks (quite a long learning time, even for low-resolution images). Therefore, the authors proposed a new model WGAN-GP, which is an improved version of the WGAN and uses gradient penalty as the method of preventing overfitting [10]. Despite this advantages, WGAN leads to a slow optimization process in practice.

BGAN. It is known that for each fixed generator there is a unique and optimal discriminator [10]. It follows that the generator is also optimal when the discriminator outputs the probability of 0.5 for all examples created from the latent space. However, in practice, it is almost impossible to get an ideal discriminator [2]. Therefore, the authors propose to make changes to the original loss function of the generator so that the discriminator outputs the probability of 0.5 for all generated data [18]. This method significantly improves the stability of learning. This network is the same as the original DCGAN (Fig. 3) but with a bit rearranged loss function.

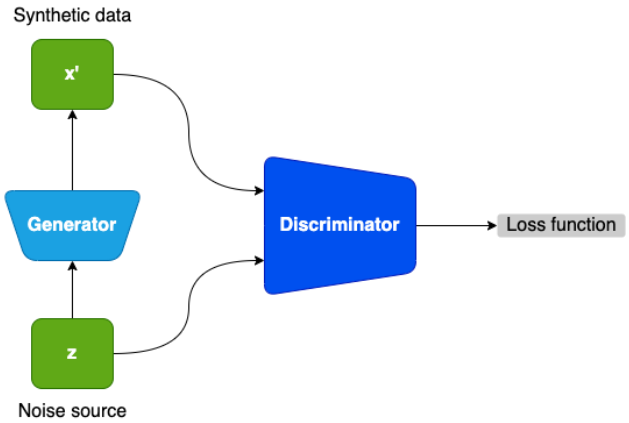


Fig. 3. **BGAN structure (same as DCGAN)**
Source: compiled by the authors

BEGAN. In this network, the discriminator is an autoencoder, which encodes the input images into latent vectors, and then decodes them back into images. Thus, the discriminator returns a reconstruction error between the input and the restored image instead of the probability value [19]. The basic idea is that the same reconstruction error values for real and generated images ultimately lead to the same distribution of real and generated data. As the author says [19] the discriminator of this architecture is deep convolutional neural network as autoencoder. The generator uses the same architecture as the decoder of the disciminator but with different weights. Basic structure of BEGAN is shown in Fig. 4.

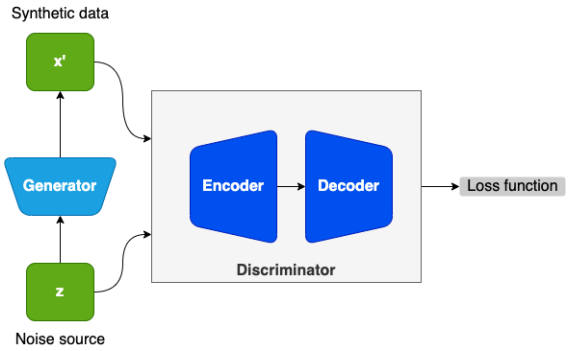


Fig. 4. **BEGAN structure**
Source: compiled by the authors

BigGAN. This architecture is used in class-conditional image generation. It was designed to unite the best approaches from the previous researches [20]. The BigGAN architecture is focused on scaling the GAN model to generate better quality and larger images. This is done by using more model parameters (more convolutional layers with large number of hyper-parameters), larger batch size, architectural changes. As a result, BigGAN is capable of generating higher-quality and larger images such as 256 by 256 and 512 by 512. Despite all of the improvements to the training process which have focused on some changes to the objective, the main disadvantage is still the training time and large computing resources.

StyleGAN. This architecture proposes large changes to the generator model rather than the discriminator. Some changes are the following: mapping network to map points from latent space to an intermediate latent space, use of the intermediate latent space to control image style in the generator [21]. As a result, the model is capable not only of generating high-quality images (resolution of 1024 pixels) but also offers control over the image style on the different levels (from the details to more general features). This model was used for generating human faces so far.

Table 1. **GAN-architectures characteristics**

Architecture	Charachteristic	Resolution
DCGAN	Low image quality, but relatively small training time. No need of large computing resources.	64
WGAN	Better image quality compared to DCGAN. Very slow training process.	64
WGAN-GP	High image quality, no mode-collapse because of gradient penalty. Very slow training time (up to several days).	any
BGAN	Better training stability compared to DCGAN, but low-quality images.	64
BEGAN	Higher image quality compared to DCGAN but very slow training.	any
BigGAN	High image quality, several image resolutions. Slow training time. Large computing resources.	128, 256, 512
StyleGAN	Very high image quality and size. Ability to control over generated image style from details to more general features. Very slow training time (up to several days). Huge computing resources.	1024

Source: **compiled by the authors**

4. EXPERIMENTAL RESULTS

4.1. Dataset

For the experiments the training dataset of 64 by 64 pixel cytological images was used [22, 23]. At the initial stage, the total number of images in the dataset was 78, which is a rather small value.

The used images are subset of biomedical images which are the structural and functional images of human and animal organs and is designed to diagnose diseases and determine the anatomical and physiological image of the body [24, 25], [26, 27], [28, 29]. Usually, these images are obtained as a result of the use of technical means of visualization in medicine and biology. Among biomedical images, the following subclasses can be distinguished: cytological, histological and immunohistochemical images.

Therefore, the dataset was expanded to 800 images using Python library called Rudi which uses some of the affine transformations (random flip, rotation, distortion, skewing, zooming). The library parameters were set to default [30].

Cytological images are images of cells of the body, histological – images of tissues, and immunohistochemical – images of cells and their reactions to certain markers. Examples of these images are shown in Fig. 5, Fig. 6 and Fig. 7 respectively.

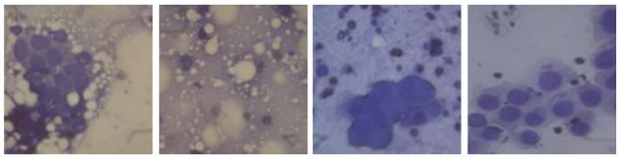


Fig. 5. **Cytological images**
Source: **compiled by the autors**

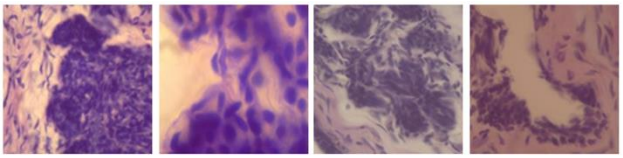


Fig. 6. **Histological images**
Source: **compiled by the autors**

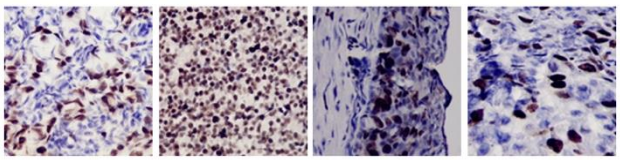


Fig. 7. **Immunohistochemical images**
Source: **compiled by the autors**

4.2. Comparison of GAN architectures

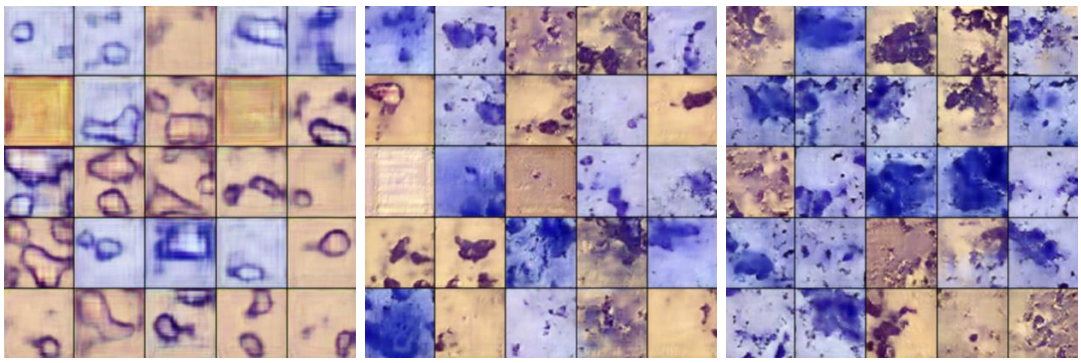
For the experiments the next architectures were chosen: DCGAN, WGAN, WGAN-GP, BGAN, BEGAN. The reason for not including BigGAN and StyleGAN is that these models require huge computing resources and more importantly, large training dataset which we do not have. So the chosen architectures and the training parameters are given in Table. 2. The table shows the name of the architecture, the optimizer used (variation of the gradient descent method), the number of iterations, the size of the batch and the training time. The size of the batch is used in the mini-batch gradient descent algorithm, and determines the number of training examples in one batch, which is selected randomly at each iteration. The network error calculation is based on that batch. The most popular optimization algorithm today is Adaptive Momentum Optimization (Adam). Comparison of synthesized images is shown on the figures below.

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

Table 2. **Training parameters of GAN architectures**

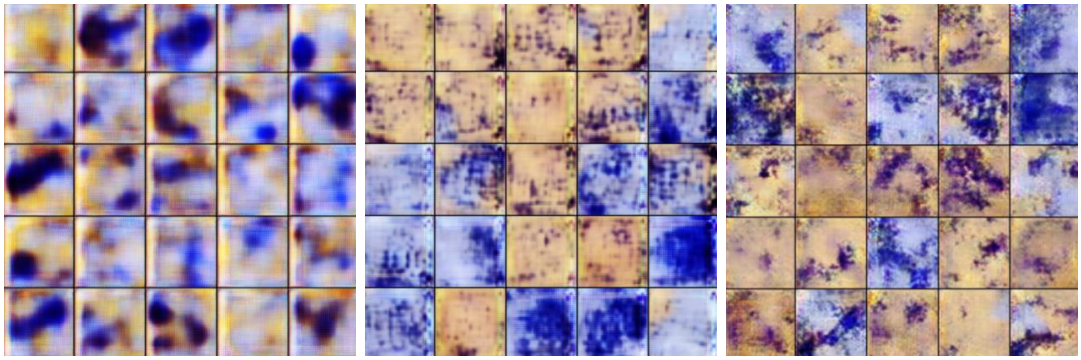
No.	Architecture	Optimizer	N. of iterations	Batch size	Training time	Average time per iteration
1	DCGAN	Adam	15 000	64	41 mins	0,16 sec
2	WGAN	RMSProp	15 000	32	1 h 52 mins	0,44 sec
3	WGAN-GP	Adam	15 000	64	5 h 43 mins	1,37 sec
4	BGAN	Adam	15 000	64	3 h 44 mins	0,89 sec
5	BEGAN	Adam	15 000	64	6 h 25 mins	1,54 sec

Source: **compiled by the autors**



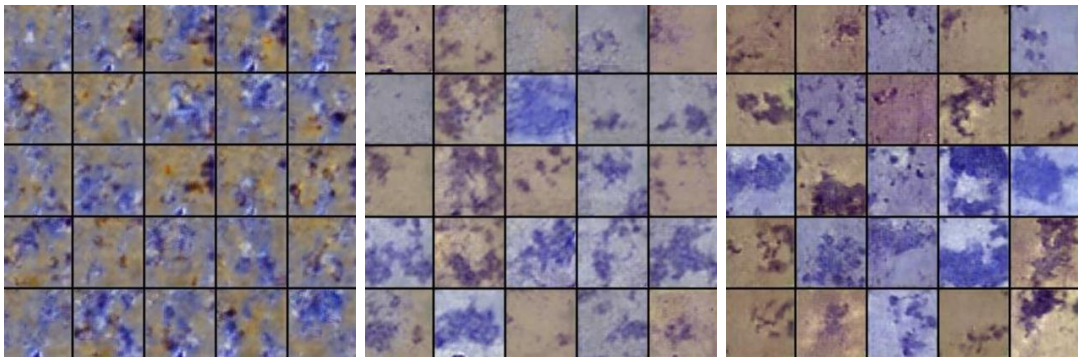
a b c

Fig. 8. **DCGAN results after 1000 (a), 7000 (b), and 15000 (c) iterations**
Source: **compiled by the authors**



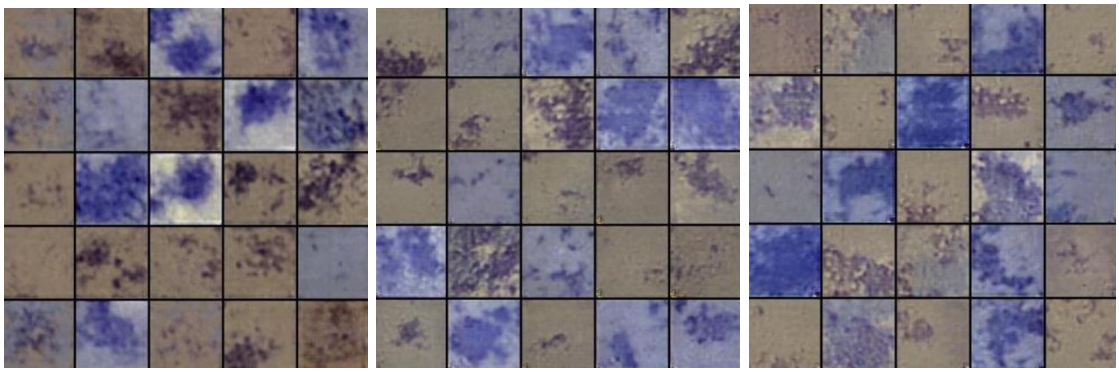
a b c

Fig. 9. **WGAN results after 1000 (a), 7000 (b), and 15000 (c) iterations**
Source: **compiled by the authors**



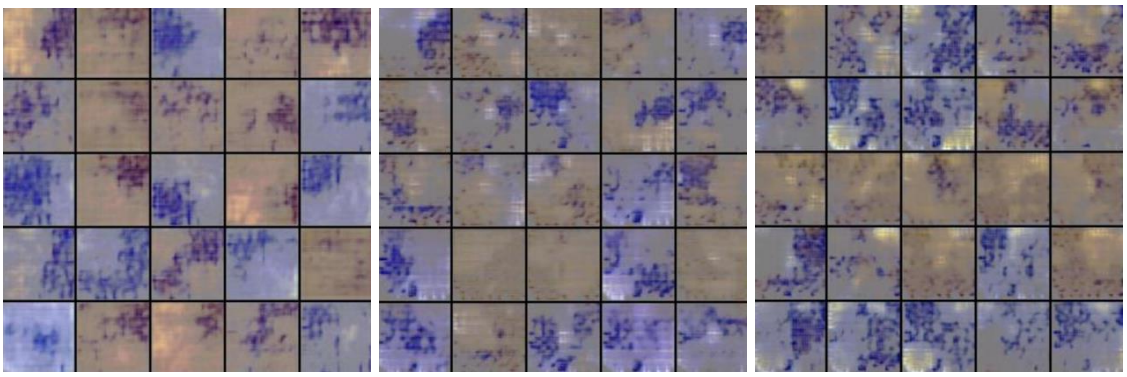
a b c

Fig. 10. **WGAN-GP results after 1000 (a), 7000 (b), and 15000 (c) iterations**
Source: **compiled by the authors**



a b c

Fig. 11. **BGAN results after 1000 (a), 7000 (b), and 15000 (c) iterations**
Source: **compiled by the authors**



a b c

Fig. 12. **BEGAN results after 1000 (a), 7000 (b), and 15000 (c) iterations**
Source: **compiled by the authors**

Each of the GANs were built on the next structure. In the discriminator we used input convolution layer forwarded by LeakyReLU activation function, three conv blocks (convolution layer, batch normalization and LeakyReLU activation), and convolution layer as the output. In the generator we used four transposed convolutional blocks (transposed convolution layer, batch normalization and ReLU activation) and transposed convolution as the output. Kernel size, stride and padding were set to 4, 2 and 1 respectively in all convolutional blocks. Hyperparameters of convolution such as input and output features in the discriminator were doubled in each next block. Initial value was 64. For the generator initial value was 1024. This value was reduced by half in each next block.

Each of the selected networks were trained for 15,000 iterations. Network parameters such as learning rate, optimizer and batch size (also final activation function) were set according to the values given by the authors in the articles. The experiments were performed using the *Nvidia Tesla K80* graphics processor using Google Colaboratory. The Python programming language and the PyTorch library were used to write the code.

To analyze the quality of the synthesized images, the FID (Frechet Inception Distance) metric was calculated. It's given in Table 3. The comparative chart is shown in Fig. 13.

The FID metric calculates difference between feature vectors of real and synthesized images. This metric is based on Google Inception 3 model and is the improved version of Inception Score (IS) [13].

The metric is calculated on the basis of the average values of m, m_w feature vectors (obtained from the penultimate layer of the Inception model) of real and synthesized images respectively, as well as their covariance matrices. If compared vectors are identical FID values is equal to zero. The metric is described by the next formula:

$$d^2((m,C),(m_w,C_w)) = \|m - m_w\|_2^2 + \\ +Tr(C + C_w - 2(CC_w)^{1/2}),$$

where: m – average value; C – covariance; Tr – trace of the matrix.

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

Table 3. **Comparison of the FID metrics of the given architectures**

Architecture	Number of trainig iterations in thousands										
	1	2	3	4	5	10	11	12	13	14	15
	FID										
DCGAN	24,58	24,16	22,33	21,12	20,66	16,34	15,97	15,56	13,47	12,86	12,67
WGAN	26,89	25,95	23,14	22,36	20,18	16,02	15,11	14,33	13,01	12,96	12,72
WGAN-GP	34,17	33,86	32,12	31,02	29,35	25,01	24,39	23,11	21,58	20,03	19,09
BGAN	16,89	16,56	16,02	15,76	15,21	13,41	13,05	12,78	12,34	11,24	10,03
BEGAN	21,90	21,45	21,12	20,46	20,04	18,23	17,88	17,03	16,64	15,77	15,32

Source: **compiled by the authors**

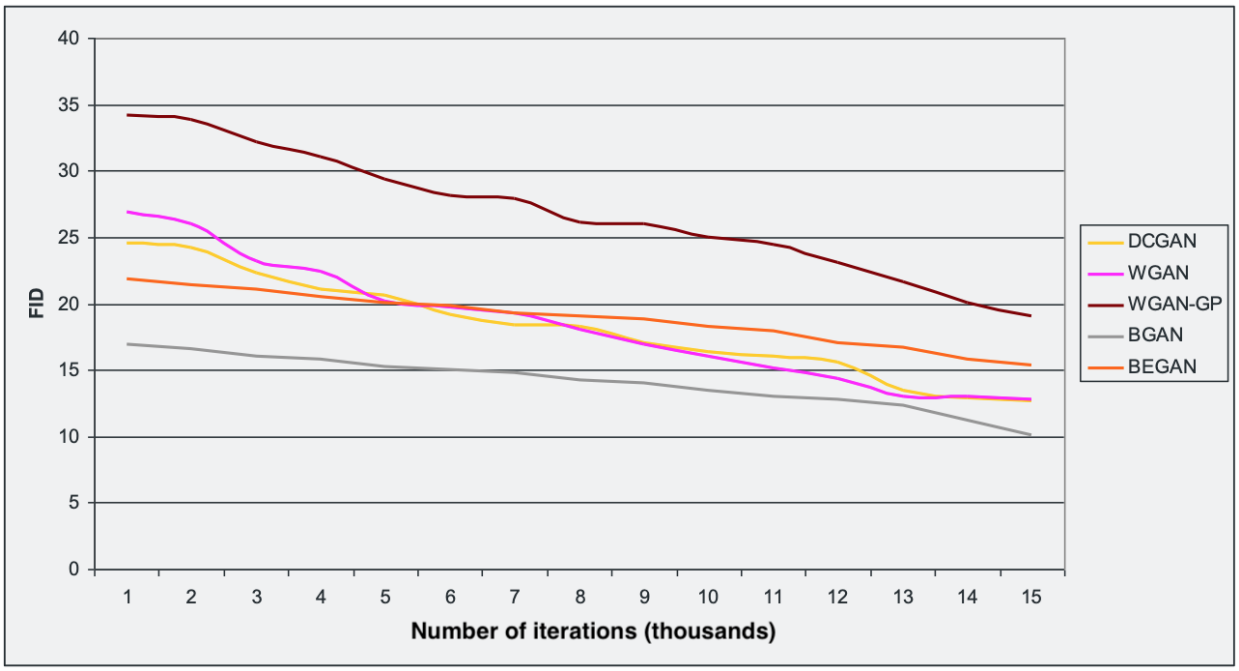


Fig. 13. **Comparison of the FID metrics during training process**

Source: **compiled by the authors**

CONCLUSIONS

The results of this research are following:

1. The article analyzes and compares on the basis of the criteria (training time, FID metric) GAN architecture: DCGAN, WGAN, WGAN-GP, BGAN, BEGAN.

2. Software prototype of the computer module in experiments for the synthesis of cytological images using the analyzed GAN architectures has been developed.

3. After the first thousand training iterations, it can be seen that the WGAN-GP architecture has the greatest error for the FID metric – about 37. This architecture will not be used in further researches.

4. The architectures DCGAN, WGAN, BEGAN have insignificant initial spread (about 4). The interval of error for these architectures remains virtually unchanged over the training time.

5. The BGAN architecture has the best result. The learning error curve varied smoothly from 17 to 10.

6. For the synthesis of cytological images, it is advisable to use the architecture DCGAN, WGAN, BEGAN BGAN and with a smaller number of iterations (about 7000) with training time is approximately 1.5 hours.

7. Direction for further research - improvement of GAN architectures: DCGAN, WGAN, BEGAN BGAN.

REFERENCES

1. Albawi, S., Mohammed, T. A. & Al-Zawi, S. “Understanding of a Convolutional Neural Network”. *International Conference on Engineering and Technology (ICET).* 2017. p. 1–6. DOI: https://doi.org/10.1109/ICEngTechnol.2017.8308186.

2. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. “Generative Adversarial Networks”. *Advances in Neural Information Processing Systems*. 2014. DOI: https://doi.org/10.1145/3422622.

3. Ian Goodfellow, Yoshua Bengio & Aaron Courville. “Deep Learning”. *The MIT Press*. 2016. 800 p. ISBN: 0262035618.

4. Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B. & Bharath, A. “Generative Adversarial Networks: An Overview”. *IEEE Signal Processing Magazine*. 2017; Vol. 35. DOI: https://doi.org/10.1109/MSP.2017.2765202.

5. Che, T. “Mode Regularized Generative Adversarial Networks”. 2016. – Available from: https://arxiv.org/abs/1612.02136. – [Accessed: Oct, 2020].

6. Goodfellow, I. “NIPS 2016 Tutorial: Generative Adversarial Networks”. 2016. – Available from: https://arxiv.org/abs/1701.00160. – [Accessed: Oct, 2020].

7. Zhang, Y., Govindaraj, V., Tang, C., Zhu, W. & Sun, J. “High Performance Multiple Sclerosis Classification by Data Augmentation and AlexNet Transfer Learning Model”. *Journal of Medical Imaging and Health Informatics*. 2019; Vol. 9 No 9: 2012–2021.

8. Lan, L., You, L., Zhang, Z., Fan, Z., Zhao, W., Zeng, N., Chen, Y. & Zhou, X. “Generative Adversarial Networks and Its Applications in Biomedical Informatics”. *Frontiers in public health*. 2020; Vol. 8: 164. DOI: https://doi.org/10.3389/fpubh.2020.00164.

9. Berezsky, O. M., Liashchynskyi, P. B., Liashchynskyi, P. B., Suchovych, A. R. & Dolynuk, T. M. “Biomedical Images Synthesis Using Generative Adversarial Networks”. *Ukrainian Journal of Information Technology*. 2019; Vol. 1 No 1: 35–40. DOI: https://doi.org/10.23939/ujit2019.01.035.

10. Zhang, G., You, L., Lan, L., Zeng, N., Chen, W., Poehling, G. G. & Zhou, X. “Risk Prediction Model for Knee Arthroplasty”. *IEEE Access*. 2019; Vol. 7: 34645–34654. DOI: https://doi.org/10.1109/ACCESS.2019.2900619.

11. Pandey, S., Singh, P. & Tian, J. “An Image Augmentation Approach Using Two-Stage Generative Adversarial Network for Nuclei Image Segmentation”. *Biomed. Signal Process. Control.* 2020; Vol. 57. DOI: https://doi.org/10.1016/j.bspc.2019.101782.

12. Frid-Adar, M., Diamant, I., Klang, E., Amitai, M., Goldberger, J. & Greenspan, H. “GAN-based Synthetic Medical Image Augmentation for Increased CNN Performance in Liver Lesion Classification”. *Neurocomputing.* 2018; Vol. 321: 321–331. DOI: https://doi.org/10.1016/j.neucom.2018.09.013.

13. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B. & Hochreiter, S. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. 2017. DOI: https://dl.acm.org/doi/10.5555/3295222.3295408.

14.Weng, L. “From GAN to WGAN”. 2019. – Available from: https://arxiv.org/abs/1904.08994. – [Accessed: Dec, 2020].

15. Zhang, Z., Li, M. & Yu, J. “On the Convergence and Mode Collapse of Gan”. *In SIGGRAPH Asia 2018 Technical Briefs*. 2018. p. 1–4.

16. Arjovsky, M., Chintala, S. & Bottou, L. “Wasserstein Generative Adversarial Networks”. *Proceedings of the 34th International Conference on Machine Learning, in Proceedings of Machine Learning Research*. 2017; Vol. 70: 214–223 – Available from: http://proceedings.mlr.press/v70/arjovsky17a.html. – [Accessed: Dec, 2020].

17. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. “Improved Techniques for Training GANs”. 2016. – DOI: https://dl.acm.org/doi/10.5555/3157096.3157346.

18. Hjelm, R. D., Jacob, A. P., Che, T., Trischler, A., Cho, K. & Bengio, Y. “Boundary-Seeking Generative Adversarial Networks”. 2017. – Available from: https://arxiv.org/abs/1702.08431. – [Accessed: Dec, 2020].

19. Berthelot, D., Schumm, T. & Metz, L. “Began: Boundary Equilibrium Generative Adversarial Networks”. 2017. – Available from: https://arxiv.org/abs/1703.10717. – [Accessed: Oct, 2020].

20. Brock, A., Donahue, J. & Simonyan K. “Large Scale GAN Training for High Fidelity Natural Image Sythesis”. 2018. – Available from: https://arxiv.org/abs/1809.11096. – [Accessed: Oct, 2020].

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

21. Brownlee, J. “A Gentle Introduction to StyleGAN the Style Generative Adversarial Network”. 2020. – Available from: https://machinelearningmastery.com/introduction-to-style-generative-adversarial-network-stylegan. – [Accessed: September, 2020].

22. Certificate for copyright to a work №75359 “Database of Digital Histological and Cytological Images of precancerous and Cancerous States of the Breast “BPCI2100”. Berezsky, O., Melnyk, G., Verbovyi S., Pitsun, O., Nykolyuk, V. & Datsko, T. Registration date 14.12.2017 State Enterprise “Ukrainian Intellectual Property Institute’ (Ukrpatent) https://ukrpatent.org.

23. Berezsky, O. M., Pitsun, O. Yo., Verbovyi, S. O. & Datsko. T. V. “Relational Database of Intelligent Automated Microscopy System”. *Scientific Bulletin of UNFU* (in Ukrainian). 2017; Vol. 27 No 5: 125–129. DOI: https://doi.org/10.15421/40270525.

24. Berezsky, O. M., Batko, Y. M., Berezka, K. M. & Verbovyi, S. O. “Methods, Algorithms and Software for Processing Biomedical Images”. *Economic Thought* (in Ukrainian). 2017. 350 p.

25. Berezsky, O., Melnyk, G., Datsko, T. & Verbovyy, S. “An Intelligent System for Cytological and Histological Image Analysis”. *Proceedings of the 13 th International Conference “The Experience of Designing and Application of CAD Systems in Microelectronics” CADSM 2015*. 24-27 February 2015. Polyana-Svalyava: Ukraine. 2015. p. 28–31. DOI: https://doi.org/10.1109/CADSM.2015.7230787.

26. Berezsky, O., Verbovyy, S. & Pitsun, O. “Hybrid Intelligent Information Techology for Biomedical Image Processing”. *Proceedings of the IEEE International Conference “Computer Science and Information Technology” CSIT’2018*. Lviv: Ukraine. 11-14 September, 2018. p. 420–423. DOI: https://doi.org/10.1109/STC-CSIT.2018.8526711.

27. Berezsky, O., Datsko, T. & Verbovyy, S. “The Intelligent System for Diagnosing Breast Cancers Based on Image Analysis”. *Proceedings of Information Technology in Innovation Business (ITIB).* Kharkiv: Ukraine. 7-9 October, 2015. p. 27–30. DOI: https://doi.org/10.1109/ITIB.2015.7355067.

28. Berezsky, O., Verbovyy, S., Dubchak, L. & Datsko, T. “Fuzzy System Diagnosing of Precancerous and Cancerous Conditions of the Breas”. *Proceedings of the XIth International Scientific and Technical Conference Computer Sciences and Information Technology (CSIT’2016).* Lviv: Ukraine. 6-10 September, 2016. p. 200–203. DOI: https://doi.org/10.1109/STC-CSIT.2016.7589906.

29. Karliuk, A., Nastenko I., Nosovets O. & Babenko, V. “Classification of Brain Images by Using the Automatic Segmentation and Texture Analysis”. *Applied Aspects of Information Technology*. 2020; Vol.3 No.4: 263–275. DOI: https://doi.org/10.15276/aait.04.2020.4.

30. Liashchynskyi, P. “Rudi: Lightweight Image Converter and Dataset Augmentor”. *GitHub*. 2019. DOI: https://doi.org/10.5281/zenodo.3374946.

Conflicts of Interest: the authors declare no conflict of interest

Received 23.12.2020
Received after revision 26.02.2021
Accepted 15.03.2021

DOI: https://doi.org/10.15276/aait.03.2021.4
УДК 004.932: 616-006.04: 004.822

Порівняння архітектур генеративних змагальних мереж для синтез у біомедичних зображень

Олег Миколайович Березький[1)]

ORCID: https://orcid.org/ 0000-0001-9931-4154; ob@ wunu.edu.ua. Scopus ID: 6505609877

Петро Борисович Лящинський[1)]

ORCID: https://orcid.org/0000-0002-3920-6239; p.liashchynskyi@st.wunu.edu.ua

[1)] Західноукраїнський національний університет, вул. Львівська, 11. Тернопіль, 46009, Україна

АНОТАЦІЯ

У статті проаналізовано та здійснено порівняння архітектур генеративно-змагальних мереж. Ці мережі будуються на основі згорткових нейронних мереж, що широко застосовуються для задач класифікації. Згорткові мережі вимагають великої кількості навчальних даних, щоб досягнути потрібної точності. У роботі генеративно-змагальні мережі використано для синтезу біомедичних зображень. Біомедичні зображення широко застосовуються в медицині, особливо в онкології. Для постановки діагнозу в онкології біомедичні зображення поділяються на три класи: цитологічні, гістологічні та імуногістохімічні. Начальні вибірки біомедичних зображень є дуже малими. Отримання навчальних зображень є складним і дорогим процесом. Для експериментів використано навчальну вибірку цитологічних зображень. В статті розглянуто найбільш розповсюджені архітектури генеративно-змагальних мереж, такі як DCGAN, WGAN, WGAN-GP, BGAN, BEGAN. Типова архітектура GAN мережі складається із генератора та дискримінатора. В основі генератора та дискримінатора лежить архітектура CNN мережі. . У роботі проаналізовано алгоритм глибокого навчання для синтезу зображень за допомогою генеративно-змагальних мереж. Під час експериментів розв'язано такі задачі. Для збільшення початкової кількості навчальних даних у вибірці застосовано множину афінних перетворень: відображення, паралельний перенос, зсув, масштабування тощо. Кожна з архітектур навчалася протягом визначеної кількості ітерації. Обрані архітектури були порівняні за часом навчання та якістю зображень на основі FID (Frechet Inception Distance) метрики. Для експериментів використано мову програмування Python і фреймворк для машинного навчання Pytorch. На основі використаних технологій розроблено прототип програмного модуля для синтезу цитологічних зображень. Синтез цитологічних зображень проведено на основі DCGAN, WGAN, WGAN-GP, BGAN, BEGAN архітектур. Для проведення експериментів було використано онлайн середовище Google Colaboratory із використанням графічного процесора Nvidia Tesla K80.

Ключові слова: глибоке навчання; генеративна змагальна мережа; біомедичні зображення; синтез зображень

ABOUT THE AUTHORS

Oleh M. Berezsky, Dr. Sci. (Eng), Professor, Head of Computer Engineering Department.
West Ukrainian National University. 11, Lvivska St. Ternopil, 46009, Ukraine
ORCID: https://orcid.org/ 0000-0001-9931-4154; ob@ wunu.edu.ua. Scopus ID: 6505609877
***Research field*:** Artificial intelligence; computer vision; metrics; atrificial neural networks

Олег Миколайович Березький, доктор технічних наук, професор, завідувач кафедри Комп'ютерної інженерії. Західноукраїнський національний університет, вул. Львівська, 11. Тернопіль, 46009, Україна

Petro B. Liashchynskyi, PhD Student of Computer Engineering Department.
West Ukrainian National University. 11, Lvivska St. Ternopil, 46009, Ukraine
ORCID: https://orcid.org/0000-0002-3920-6239; p.liashchynskyi@st.wunu.edu.ua
***Research field*:** Machine learning; artificial intelligence; image recognition; generative adversarial networks

Петро Борисович Ляцинський, аспірант кафедри Комп'ютерної інженерії. Західноукраїнський національний університет, вул. Львівська, 11. Тернопіль, 46009, Україна

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)