

Heuristic Feedback for Generator Support in Generative Adversarial Network

Dawid Połap^a and Antoni Jaszcz^b

Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44-100 Gliwice, Poland

Keywords: Generative Adversarial Network, Neural Network, Heuristic, Image Processing.


Abstract: The possibilities of using generative adversarial networks (GANs) are enormous due to the possibility of generating new data that can deceive the classifier. The zero-sum game between two networks is a solution used on an increasingly large scale in today's world. In this paper, we focus on expanding the model of generative adversarial networks by introducing a block with a selected heuristic algorithm. The additional block allows for creating a set of features extracted from the discriminator. The heuristic algorithm is based on the analysis of feature maps and extracting the position of selected pixels. Then they are clustered into averaged sets of features and used on created images by the generator. If the specified number of points within any set of features is higher than the threshold value, then the generator performs classical training. Otherwise, the loss function is subject to the penalty function. The proposed mechanism affects the operation of the GAN through additional sample analysis concerning containing specific features. To analyze the solution and impact of the proposed heuristic feedback, tests were performed based on known data sets.


1 INTRODUCTION

Recent years have brought new neural classifiers and new methods of training including federated learning. The huge development is caused by the need for fast and accurate analysis in the applications of these methods (Yang et al., 2022; Salankar et al., 2023). An example of what is the Internet of Things, Web 3.0, Industry 4.0 (Zhang et al., 2022), etc. In addition, neural networks are trained on huge data sets, which contributes to high efficiency, as well as networks that enable the generation of new data. Of course, the basic application of such data may be augmentation, however, the achievements of artificial intelligence have allowed the creation of solutions that enable the generation of data that can be mistaken for the works of people. Examples of this are image generators and different types of CNN-based models (Shahriar, 2022; Artemjew and Tadeja, 2022; Xu et al., 2023), etc.

The described possibilities are a consequence of creating a zero-sum game where the players are two neural networks. One trains to classify data as real and fake, and the other generates new samples to deceive the other network. Such contention is called

generative adversarial networks (GANs) (Cai et al., 2022). The very wide application of such a solution has contributed to increasing the capabilities of neural networks in the industry. On the other hand, scientific research on this type of network and their rivalry is one of the main trends in improving their shortcomings as well as increasing the accuracy or speed of achieving high values of evaluation metrics. Augmentation allows to the creation of artificial data to increase the number of samples that train neural networks. In the case of dedicated applications, creating a large amount of data in training databases may be unattainable in a short time. Hence, generators are used to streamline the base building process as well as to enable faster achievement of higher efficiency. Such a solution was presented in (Scarpiniti et al., 2022), where the authors used GAN to generate audio data presented in graphical form. For this purpose, the focus was on spectral representation, which enabled the correct further classification of the data. A similar application was the use of augmentation to obtain equinumerous training sets in the problem of diagnosing bearing faults (Liu et al., 2022a). GAN training involves a pre-prepared database for identifying true and false images. Hence, in (Hung and Gan, 2022) the modification of the generator architecture with a double encoder and decoder was shown.

^a  <https://orcid.org/0000-0003-1972-5979>

^b  <https://orcid.org/0000-0002-8997-0331>

The purpose of the modification was to increase the amount of data in the training set, which in the end allowed to obtain a higher accuracy of the classifier by more than 10%. GAN can contribute to network learning by generating synthetic images indicative of segmentation. Such an example of use is shown in (Ali and Cha, 2022). The authors proposed a proprietary model of the GAN network and performed tests based on images of damage to concrete elements. Based on the presented research results, the authors indicate that GAN-type networks can surpass other neural networks in accuracy. The topic of data generation and their reliability is discussed in (Mozo et al., 2022). The researchers presented the methodology of using a heuristic algorithm to select the best generator during GAN training.

An interesting solution is the use of GAN networks to generate super-resolution images (Jia et al., 2022). To enable this, the generator architecture in GAN has been refined through three processing blocks: a convolution block, an upsampling block and an attention fusion block. Another solution is to use the GAN model for the classification or labeling of samples. An example of such use is the development of the architecture with the addition of additional discriminators (Liu et al., 2022b). The described solution operates on a proprietary middle-generation module, which is introduced between the generator and the combined discriminator. The conducted research showed that it is an effective and universal solution in terms of adaptability to data sets. Another use of GAN is to generate data to attack intrusion detection systems (Lin et al., 2022). The proposed solution is based on the use of a generator to modify the original data. The authors of the research, based on the results, indicate the superiority of the proposal over other methods due to the much greater generalization of attacks. Again, in (Mira et al., 2022), GAN was used to increase the reality of video-to-speech models. The model is based on video-to-speech conversion, and then the result is evaluated by a critic. This solution increases the credibility and realism of the obtained measurements. Such solutions are also analyzed through the safety and correctness of the results. An example of research in this area is the implementation of an evaluation mechanism using a clustering algorithm (Venugopal et al., 2022). The authors drew attention to the cosine similarity of the data assuming the preservation of statistical properties.

Based on the analysis of the literature, it can be seen that GAN-type architectures are a very important element used in generating various data. This is a solution to increase the accuracy of the network during training the classifier due to the creation of new train-

ing data. Based on these motivations, in this research paper, we present the architecture of the GAN network extended with a dedicated analysis of results using a heuristic algorithm. The analysis of the heuristic algorithm enables better adaptability of the generated images to increase the efficiency of the generator. In addition, the use of heuristic feedback allows to speed up the generator's operation. The main contributions of this research are:

- heuristic feedback mechanism for image analysis,
- extending GAN architecture by additional block to support generator,
- proposed GAN evaluation by known datasets and comparison with state-of-art.

2 PROPOSED METHODOLOGY

GAN architecture is based on two networks: discriminator \mathcal{D} and generator \mathcal{G} . The discriminator will be trained using a database $X = \{x_0, x_1, \dots, x_{|X|-1}\}$. The result of the classification of a i -th sample will be marked as $\mathcal{D}(x_i)$. The generator will be trained by a random given as latent vector z , so the created sample will be denoted as $\mathcal{G}(z)$. Evaluation of the generated sample by discriminator will be $\mathcal{D}(\mathcal{G}(z))$. Moreover, let us denote Z as a set of generated vectors. $E(\cdot, \cdot)$ as an error function between a given two values. The loss function for the discriminator will be defined as:

$$L_{\mathcal{D}} = E(\mathcal{D}(x), 1) + E(\mathcal{D}(\mathcal{G}(z)), 0), \quad (1)$$

and for the generator, it will be:

$$L_{\mathcal{G}} = E(\mathcal{D}(\mathcal{G}(z)), 1). \quad (2)$$

Applying a binary cross-entropy formula, the above loss functions will be changed as:

$$L_{\mathcal{D}} = - \sum_{x \in X, z \in Z} \log(\mathcal{D}(x)) + \log(1 - \mathcal{D}(\mathcal{G}(z))) \quad (3)$$

and for the generator, it will be:

$$L_{\mathcal{G}} = - \sum_{z \in Z} \log(\mathcal{D}(\mathcal{G}(z))). \quad (4)$$

2.1 Heuristic Feature Extractor

In GAN architecture, the training process is performed separately for the discriminator and the generator. Hence, we can consider training a discriminator. The idea is to train the network to classify the incoming sample as true or false. In the case of real samples, the convolutional neural network will extract the features of images occurring in a specific class. To

analyze them, we propose the use of a heuristic algorithm that will enable the detection of those features that the classifier treats as significant. To access these features, feature maps from the last convolution layer will be extracted after the training process ends in the current iterations. The obtained feature maps for a given image will be presented as a set of n images $S = \{s_1, s_2, \dots, s_n\}$.

The application of the heuristic algorithm is based on finding selected features. For this purpose, the algorithm performs two stages of action: environment recognition, i.e. pixel analysis to identify the background of the image, and then analysis of pixels of the opposite color. If the background of the image is black, the heuristic analyzes white pixels. However, it should be noted that individual pixels do not indicate a specific feature. To enable the heuristic to analyze the image and return specific patterns representing the feature, a two-stage displacement is proposed. The first is the location of the most intensive areas, followed by a search and neighborhood analysis that indicates the area's surroundings. The exact operation of the heuristics will be described on the example of the selected algorithm, which is the fox algorithm (Połap and Woźniak, 2021). It is an algorithm inspired by the behavior of foxes when hunting prey. We assume that each heuristic algorithm has two basic parameters, which are the number of individuals N and the number of iterations T . The number of individuals determines the number of points (here pixels) that will be analyzed in one iteration. Within each iteration, the algorithm moves the points according to two mechanisms, which are global and local displacement. Assume that, the initial population of foxes is generated at random, so there will be a set of pixels' coordinates marked as $P = \{p_1, p_2, \dots, p_N\}$, where $p_i = (x_i, y_i)$ on image I , so the i -th pixel values is $I(p_i)$ and $x_i \in \langle 0, w \rangle$, $y_i \in \langle 0, h \rangle$ (w, h is width and height of the image I). At the beginning of each iteration, the best individuals p_{best} in the entire population are found - the selection depends on the given fitness function $F(\cdot)$. Each pixel in t -th iteration is placed according to a global movement, which allows the evaluation of pixels much further away from the current position. It is made using the following equation:

$$p_i^t = p_i^{t-1} + \lceil \alpha \cdot \text{sign}(p_{best}^t - p_i^{t-1}) \rceil, \quad (5)$$

where $\alpha \in (0, d(p_i^{t-1}, p_{best}^t))$ is a random coefficients in the given range, where $d(\cdot)$ is Euclidean metric. Then, each fox p_i makes a selection regarding further movement according to a random parameter $\mu \in \langle 0, 1 \rangle$:

$$\begin{cases} \text{Move closer} & \text{if } \mu > 0.75 \\ \text{Stay and disguise} & \text{if } \mu \leq 0.75 \end{cases} \quad (6)$$

If the choice is the movement, then the value of the subject's field of view is determined by:

$$r = \begin{cases} \left\lceil a \frac{\sin(\phi_0)}{\phi_0} \right\rceil & \text{if } \phi_0 \neq 0 \\ \text{rand}(0, 1) & \text{if } \phi_0 = 0 \end{cases}, \quad (7)$$

which allows the individual to be moved to a better position in the immediate vicinity as:

$$\begin{cases} x_i^t = \lceil ar \cdot \cos(\phi_1) \rceil + x_i^{t-1} \\ y_i^t = \lceil ar \cdot \sin(\phi_1) \rceil + \lceil ar \cdot \cos(\phi_2) \rceil + y_i^{t-1} \end{cases}, \quad (8)$$

where $\phi_1, \phi_2 \in \langle 0, 2\pi \rangle$ and $a \in \langle 0, 1 \rangle$. In the situation that a choice is made to stay (Eq. (6)), then the individual does not perform a local displacement.

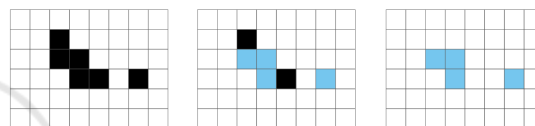


Figure 1: Visualization of the heuristic algorithm (from left to right): the original image, individuals in the heuristic algorithm located important pixels, returned a set of features in the given image.

For feature extraction, we propose to analyze feature maps from the discriminator. The first iteration of heuristic algorithms analyzes the environment which is the image. This is to determine the color of the features you are looking for. To this end, the first iteration of the heuristic is evaluated against a simple fitness function that returns the average value of the components of the RGB model, that is:

$$F_1(I, p_i) = \frac{1}{3} \sum_{k \in \{R, G, B\}} I_k(p_i). \quad (9)$$

It should be also noted, that in the case of a grayscale image, the values of all three components are the same, hence we can assume that the above average is equal to the value of any of the components. For this function, there is no preferred best value, so the best individual in the population will be selected at random. After the first iteration, each individual is evaluated and the average values of all individuals are determined. If this value is less than 128 (half of the color range in the RGB model), the background color will be black. However, when the average value is higher than 128, the background will be white and the searched features will be black.

Subsequent iterations can analyze the image by looking for features that may be significant for the classifier. For this purpose, individuals are evaluated

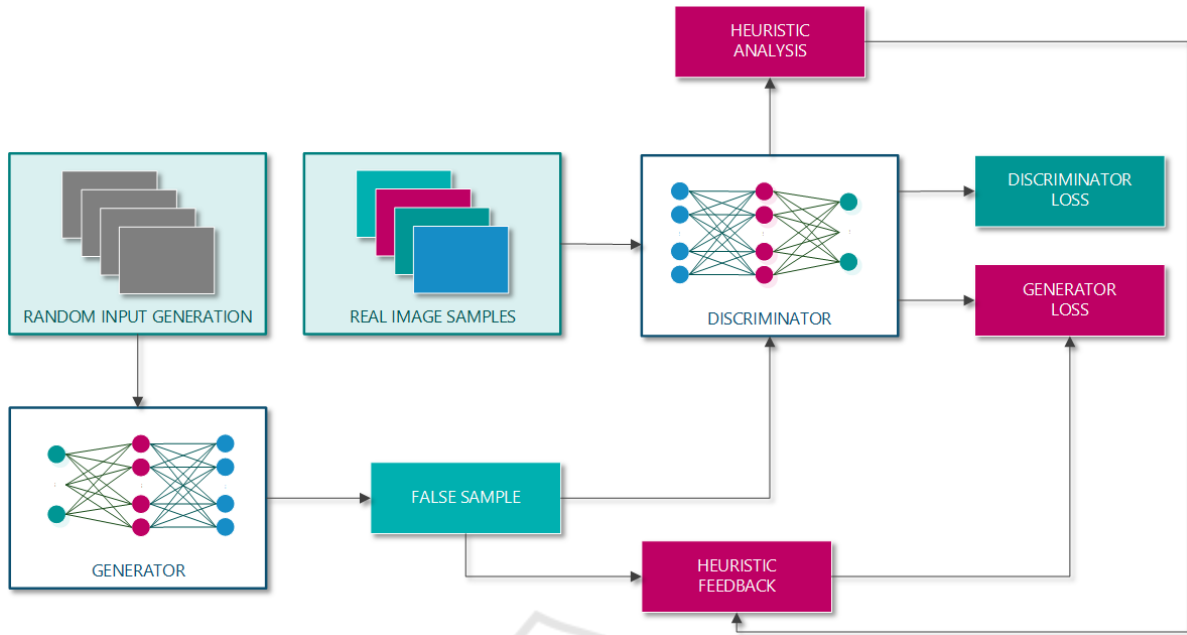


Figure 2: Visualization of the operation of the GAN architecture with the proposed heuristic algorithm.

using the second fitness function (the higher the value, the more important the pixel):

$$F_2(I, p_i, \xi) = \sum_{j=-\xi}^{\xi} \sum_{l=-\xi}^{\xi} F_3(I, (x_i, y_i), (x_i + j, y_i + l)), \quad (10)$$

where $F_3(\cdot)$ is an auxiliary function that evaluates a given pixel by determining whether it is a pixel indicating a difference in neighboring values. This is important because the boundary points will be interesting, i.e.:

$$F_3(I, p_1, p_2) = \begin{cases} 1 & \text{if } I(p_1) > 128 \text{ and } I(p_2) < 128, \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

The above equations search on a white background, but in the opposite case, majority and minority signs will be reversed.

When the algorithm reaches the iteration limit, the obtained points are clustered to group them according to a certain number of features. The algorithm performs the same operation on all other feature maps and then selects those features that are duplicated in more than half of the maps. In discriminant training, such features are extracted from samples. When all analyses are completed, the set of features for each sample is compared to combine features for the same classes. It is done by clustering into κ classes (in the case when this number is large, then more feature sets reaming to analyze by generator). Within each class, the sets are combined and clustered to the initial num-

Algorithm 1: GAN with heuristic support.

Input: Generator, Discriminator, T_{GAN} training iterations, T heuristic iterations, N heuristic population size, γ matching value

```

1   $i := 0$ ;
2  while  $i < T_{GAN}$  do
3      Train discriminator;
4      Get feature maps;
5      Use heuristics to finding locate features
        by  $T$  iterations with  $N$  points;
6      Combine all features into one set;
7      Reduce the number of feature sets using
        clustering algorithm;
8      Generate random sample;
9      Evaluate sample by discriminator and
        heuristic matching algorithm;
10     if sample is not covered by one of the
        heuristic class more than  $\gamma\%$  points then
11         | Apply penalty function;
12     end
13     Modify weights and filters in generator;
14      $i += 1$ ;
15 end
    
```

ber of points - this makes it possible to obtain averaged values of features within a given class. A simplified visualization of the operation of the heuristic algorithm is shown in Fig. 1.

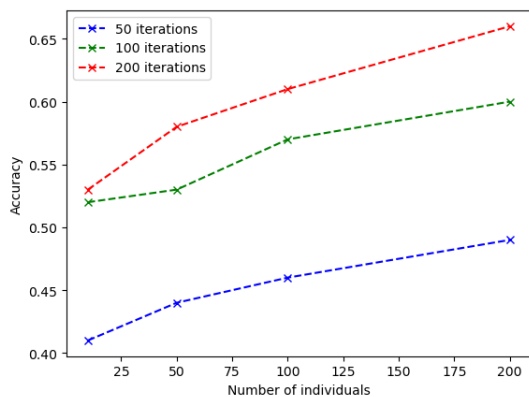


Figure 3: Chart of accurate measurements of the used GAN model depending on the used parameters of the heuristic algorithm.

2.2 Heuristic Feedback Mechanism for Generator

The sets of features prepared by heuristics during the training of the discriminator are finally used during the training of the generator. After the sample is created, it is first checked by feature sets of heuristics. The check consists of superimposing points from a single set on the generated image. The pixel values under these points are then checked. The check consists of verifying the white color, i.e. if the pixel has a value greater than 128, the pixel is passed. this operation is performed for all matrices. If there is a situation in which more than $\gamma\%$ points from any set are covered, that is, the sample contains features of one of the classes.

According to the adopted training model of the generator, the sample is evaluated by the discriminator and the value of the loss function is determined. In the absence of feature assignment by comparing the sets obtained by heuristics, the loss function is increased by 10%, which is understood as the penalty function. A simplified scheme of operation is presented in Alg. 1

3 EXPERIMENTS

To evaluate the proposed method and analyze the coefficients, two classical databases, MNIST (LeCun, 1998) and Fashion-MNIST (Xiao et al., 2017), were selected. Databases contain images of size 28×28 . The DCGAN (Dewi et al., 2022) model implemented in Keras was used as the GAN architecture. All tests were performed on the Intel Core i7-8750H with 24GB RAM and NVIDIA GeForce GTX 1050 Ti.

3.1 Coefficients Analysis

In the beginning, the heuristic coefficients were analyzed with the assumption of obtaining the best possible results. For this purpose, we used only the MNIST dataset with the following parameters: three different iteration values $T = \{50, 100, 200\}$ and the number of individuals in the population as $N = \{10, 50, 100, 200\}$. The first analysis was to find out the best value for the mentioned heuristic parameters. Therefore, a constant value of $\gamma = 0.5$ was chosen. This γ value means that at least 50% of the localized features (in one of the ten classes - it was set up to create only ten clustered classes) should be covered on the generated samples in order not to apply the penalty function. The training iteration was set as 1000. All tests were performed ten times and all results were averaged. The obtained charts are shown in Fig. 3. The obtained results indicate that the greater the number of individuals and iterations, the higher the accuracy. With only 50 iterations, the accuracy of the GAN with the methodology used increases. However, even with 200 individuals, the accuracy reached below 50%. Doubling the iteration allowed for achieving nearly 15% better results when using 10 individuals. By increasing them to 200, the efficiency reached 60%. However, the best results were seen using 200 iterations, where the accuracy increased rapidly and was able to exceed 65%. However, it should be noted that the use of minimum iteration values or the number of individuals means quite random values due to too small several analyzed image areas. Especially through the use of clustering of these features, which with high randomness may result in generating sets of random features. Hence, the use of a large number of individuals and iterations is an important element.

In the next part of the conducted experiments, we used 100 individuals with 200 iterations to analyze the effect of the *gamma* coefficient. For this purpose, the generator was trained with different parameter values of these parameters: $\gamma = \{0.1, 0.2, \dots, 0.9, 1\}$. The value of the coefficient equal to 0 means that there is no proposed mechanism for using the heuristic algorithm. In Fig. 4 the relationship between the average accuracy of the GAN and the applied value of the coverage coefficient of the set of heuristic features and the generated images is presented. The application of the proposed mechanism achieves results similar to its absence in the case of a coefficient value below 0.4. This is because this is a very small point coverage value and its rapid exceeding will not cause changes in the value of the generator loss function. Hence, at values equal to or greater than 0.4, a dif-

Table 1: Comparison of the accuracy on GAN models with and without the proposed mechanism.

		Accuracy	Discriminator loss	Generator loss
MNIST	DCGAN	0,6934	0,59	0,96
	DCGAN with RFOA	0,7445	0,58	0,82
Fashion-MNIST	DCGAN	0,5234	0,63	1,03
	DCGAN with RFOA	0,6639	0,52	0,83

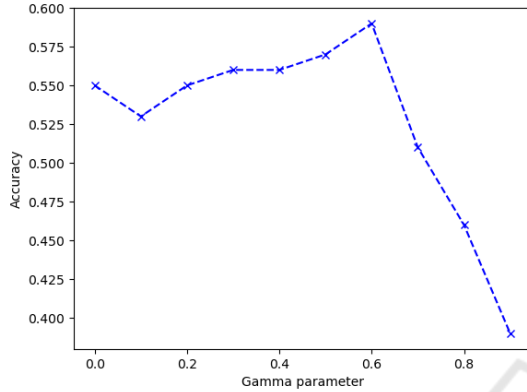
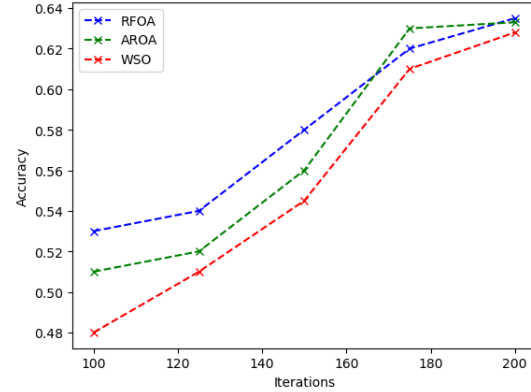


Figure 4: Graph of the average accuracy values to the used gamma parameter.

ference can be seen. The best results were obtained using values $\gamma = 0.6$, where the accuracy with such a small number of training (1000 iterations). Accuracy results were obtained higher by 5% compared to the lack of the proposed method. Higher values indicate frequent use of the penalty function, which in turn manifests itself in worse and worse accuracy.

3.2 Heuristic Analysis

The next stage of evaluating the proposals was the use of other heuristic algorithms and the indication of differences between them. Three selected heuristics were implemented, such as RFOA (presented in this paper), artificial rabbits optimization algorithm (AROA) (Wang et al., 2022) and white shark optimizer (WSO) (Braik et al., 2022). All algorithms were identically modified to analyze the images by adding the ceiling function to the equations and using the presented fitness functions. A variant of one hundred individuals in each population and different numbers of iterations were used. Each of the tests for the selected algorithm was performed ten times, and the results were averaged, as shown in Fig.5. All selected algorithms obtained similar results, differing from each other at the level of 0.2%. It should be noted that AROA is the only one to have more jumpy accuracies with the increasing number of iterations. For the other two algorithms, the accuracy increases almost linearly. It is worth noting that each heuristic


 Figure 5: Dependence of the number of iterations for 100 individuals in selected heuristic algorithms with $\gamma = 0.6$ to accuracy.

algorithm works on almost identical operations with the difference related to the modeled equations. Considering that the same coefficient values were used in the tested algorithms, the difference can be stated that with smaller numbers of iterations, RFOA can locate image features faster based on the declared fitness functions.

3.3 GAN Analysis

The best parameters in previous tests turned out to be RFOA with 200 iterations, 200 individuals and the parameter $\gamma = 0.6$. To reduce the number of operations, the number of individuals was reduced to 100, but the number of GAN training iterations was increased threefold. Using these values, we train the GAN model with and without the proposed heuristic mechanism for two datasets: MNIST and Fashion-MNIST. The obtained results are shown in Tab. 1. In the case of the MNIST dataset, the accuracy for the original GAN model was 69,34% with a generator loss of 0,96. In the case of using the proposed mechanism, the accuracy obtained was higher by more than 5%, as it reached 74,45%. A more important factor is the generator loss, which is significantly lower - 0.82. Compared to the original classifier, this is a difference of 0.14. Again, for the second database, the results turned out to be much weaker due to the analysis of more complex objects in terms of features. The

original GAN model achieved an accuracy of 52.34% and the proposed algorithm 66.39%. This is a much higher result, which is visible in the values of the loss function. For the discriminator, there is a difference of 0.11 (while for the previous database, it was 0.01), and for the generator, it was 0.2. Based on the obtained results, it can be concluded that the proposed mechanism of image analysis in terms of having certain features is an effective solution that can significantly affect the operation of GAN models.

4 CONCLUSIONS

In this paper, we presented a modification of the GAN learning model. We extended the operations by building a set of essential features based on the images returned by the discriminator by the heuristic algorithm. The set is used by the generator to check whether the generated image has essential features for the discriminator. If not, the loss function is subject to the value of the penalty mechanism. Based on the test results, it was noticed that the proposed methodology allows for a significant increase in the accuracy of the generator. Especially when the database was much more diverse. It was noticed that the modification of the value of the loss function affects the training of the network due to the learning algorithm, which is ADAM. In addition, the implemented heuristic algorithm can achieve good results with lower parameter values. The presented method is important for enabling more efficient training of the generator in GAN models.

REFERENCES

- Ali, R. and Cha, Y.-J. (2022). Attention-based generative adversarial network with internal damage segmentation using thermography. *Automation in Construction*, 141:104412.
- Artiemjew, P. and Tadeja, S. K. (2022). Using convnet for classification task in parallel coordinates visualization of topologically arranged attribute values. In *ICAART (3)*, pages 167–171.
- Braik, M., Hammouri, A., Atwan, J., Al-Betar, M. A., and Awadallah, M. A. (2022). White shark optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. *Knowledge-Based Systems*, 243:108457.
- Cai, J., Li, C., Tao, X., and Tai, Y.-W. (2022). Image multi-painting via progressive generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 978–987.
- Dewi, C., Chen, R.-C., Liu, Y.-T., and Tai, S.-K. (2022). Synthetic data generation using dcgan for improved traffic sign recognition. *Neural Computing and Applications*, 34(24):21465–21480.
- Hung, S.-K. and Gan, J. Q. (2022). Boosting facial emotion recognition by using gans to augment small facial expression dataset. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.
- Jia, S., Wang, Z., Li, Q., Jia, X., and Xu, M. (2022). Multi-attention generative adversarial network for remote sensing image super resolution. *IEEE Transactions on Geoscience and Remote Sensing*.
- LeCun, Y. (1998). The mnist database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>.
- Lin, Z., Shi, Y., and Xue, Z. (2022). Idsgan: Generative adversarial networks for attack generation against intrusion detection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 79–91. Springer.
- Liu, Y., Jiang, H., Liu, C., Yang, W., and Sun, W. (2022a). Data-augmented wavelet capsule generative adversarial network for rolling bearing fault diagnosis. *Knowledge-Based Systems*, 252:109439.
- Liu, Y., Wei, X., Lu, Y., Zhao, C., and Qiao, X. (2022b). Source free domain adaptation via combined discriminative gan model for image classification. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.
- Mira, R., Vougioukas, K., Ma, P., Petridis, S., Schuller, B. W., and Pantic, M. (2022). End-to-end video-to-speech synthesis using generative adversarial networks. *IEEE Transactions on Cybernetics*.
- Mozo, A., González-Prieto, Á., Pastor, A., Gómez-Canaval, S., and Talavera, E. (2022). Synthetic flow-based cryptomining attack generation through generative adversarial networks. *Scientific reports*, 12(1):1–27.
- Pořap, D. and Woźniak, M. (2021). Red fox optimization algorithm. *Expert Systems with Applications*, 166:114107.
- Salankar, N., Koundal, D., Chakraborty, C., and Garg, L. (2023). Automated attention deficit classification system from multimodal physiological signals. *Multimedia Tools and Applications*, 82(4):4897–4912.
- Scarpiniti, M., Mauri, C., Comminiello, D., Uncini, A., and Lee, Y.-C. (2022). Coval-gan: A complex-valued spectral gan architecture for the effective audio data augmentation in construction sites. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.
- Shahriar, S. (2022). Gan computers generate arts? a survey on visual arts, music, and literary text generation using generative adversarial network. *Displays*, page 102237.
- Venugopal, R., Shafqat, N., Venugopal, I., Tillbury, B. M. J., Stafford, H. D., and Bourazeri, A. (2022). Privacy preserving generative adversarial networks to model electronic health records. *Neural Networks*, 153:339–348.
- Wang, L., Cao, Q., Zhang, Z., Mirjalili, S., and Zhao, W. (2022). Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving en-

- gineering optimization problems. *Engineering Applications of Artificial Intelligence*, 114:105082.
- Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.
- Xu, L., Song, Z., Wang, D., Su, J., Fang, Z., Ding, C., Gan, W., Yan, Y., Jin, X., Yang, X., et al. (2023). Actformer: A gan-based transformer towards general action-conditioned 3d human motion generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2228–2238.
- Yang, W., Xiang, W., Yang, Y., and Cheng, P. (2022). Optimizing federated learning with deep reinforcement learning for digital twin empowered industrial iot. *IEEE Transactions on Industrial Informatics*, 19(2):1884–1893.
- Zhang, T., Gao, L., He, C., Zhang, M., Krishnamachari, B., and Avestimehr, A. S. (2022). Federated learning for the internet of things: Applications, challenges, and opportunities. *IEEE Internet of Things Magazine*, 5(1):24–29.

