Abstract Pattern Image Generation using Generative Adversarial Networks

Mohamed Mahyoub
Tutor Reach
United Kingdom
m.mahyoub@tutorreach.com

Friska Natalia
Faculty of Engineering and Informatics
Universitas Multimedia Nusantara
Tangerang, Indonesia
friska.natalia@umn.ac.id

Sud Sudirman
School of Computer Science and Mathematics
Liverpool John Moores University
Liverpool, United Kingdom
s.sudirman@ljmu.ac.uk

Sadiq H. Abdulhussain
University of Baghdad
Baghdad, Iraq
sadiqh76@yahoo.com

Basheera M. Mahmmod
University of Baghdad
Baghdad, Iraq
basheera.m@coeng.uobaghdad.edu.iq

***Abstract*—Abstract pattern is very commonly used in the textile and fashion industry. Pattern design is an area where designers need to come up with new and attractive patterns every day. It is very difficult to find employees with a sufficient creative mindset and the necessary skills to come up with new unseen attractive designs. Therefore, it would be ideal to identify a process that would allow for these patterns to be generated on their own with little to no human interaction. This can be achieved using deep learning models and techniques. One of the most recent and promising tools to solve this type of problem is Generative Adversarial Networks (GANs). In this paper, we investigate the suitability of GAN in producing abstract patterns. We achieve this by generating abstract design patterns using the two most popular GANs, namely Deep Convolutional GAN and Wasserstein GAN. By identifying the best-performing model after training using hyperparameter optimization and generating some output patterns we show that Wasserstein GAN is superior to Deep Convolutional GAN.**

Keywords—Abstract Pattern, Image Synthesis, Generative Adversarial Networks, Deep Convolutional GAN, Wasserstein GAN

I. Introduction

The textile industry is growing rapidly worldwide. In 2021, the global textile market size was estimated at USD 993.6 billion and is anticipated to grow at a rate of 4.0% from 2022 to 2030 [1]. To keep the revenue coming, textile designers need to come up with creative ideas every day. Since this task is something that depends on a person's skill and experience, it is becoming very difficult for apparel designers to continuously design unseen, realistic, and eye-catching patterns for clothes. This is not limited to just apparel, it also consists of all other types of fabrics such as curtains, bedsheets, etc. Exploring this area with the help of deep learning neural networks can resolve this problem. Since not much work is done in the industry of fashion, it can generate insights that were not known earlier.

The common approach to resolve this problem is to have an automated design generation mechanism that could generate unique attractive designs which can further be enhanced by human intervention otherwise can be used as is. And this process should inculcate classes so that designs are generated based on a specific class. For instance, there can be classes such as floral patterns, checks, quotes, solids, abstracts, and many more. The research that has happened for now is limited to classifying the different classes of textiles and generating basic designs, though the need to have attractive patterns is not achieved yet.

Generative Adversarial Network (GAN) [2] is a neural network architecture that can be used to generate synthetic images that are similar but not identical to the images used to train the network. This is a relatively new technology that was only been introduced in 2014 but has received a lot of attention in the Artificial Intelligence research communities. We will be investigating this exciting and relatively new technology in generating or synthesizing new abstract images that is suitable to be used as patterns in apparel design. In particular, we would like to find out what pre-processing techniques can be applied to the input images to improve the quality of the generated images. We will also investigate how the GAN models can be evaluated when generated images are abstract in nature. The main aim of this research study is to propose a model that can generate abstract design patterns for apparel in the textile industry using a publicly available dataset of abstract pattern images. We will achieve these by 1) generating abstract design patterns using two state-of-the-art GAN models, 2) identifying the best-performing model after hyperparameter optimization, and 3) generating textile design patterns using the final models.

II. Literature Review

A. *Overview of Generative Adversarial Networks*

GANs architecture (shown in Figure 1) is based on a zero-sum game. There are two opponents, one is the "generator" and the other is the "discriminator". Random noise is added as an input to a generator and the generator creates an output. This output acts as an input for the discriminator. The discriminator assigns a score from 0 to 1, the more realistic the generated output is, the highest score is provided by the discriminator and vice versa [2]. The generator loses the battle when the discriminator correctly classifies the generated images. In this case, the generator needs to continuously improve the quality of images to win over the discriminator. On the other hand, the discriminator loses when it fails to distinguish the original images from the generated ones.

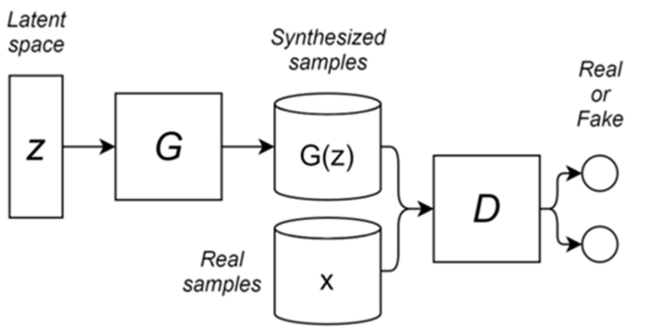

Fig. 1. An overview of Generative Adversarial Networks architecture.

The initially generated images will never be like the original images and so it is important to continuously train the model and optimize it so it can improve the quality. Although this type of training is not like how other neural networks are trained. In this, either the generator or the discriminator is made stationary one at a time and the other respective component is optimized [3], i.e., when the generator is made stationary then the discriminator will be worked upon and when the discriminator is made stationary then the generator will be worked upon.

During the training of the generator, the discriminator is made stationary and once that is done, the generator starts generating images. On these images, the discriminator checks and assigns a score, and this output from the discriminator is transferred to the generator. Since there is some difference between the original images and the generated images at the start, loss from the discriminator is fed to the generator. The outcome of this iteration will be that generator is able to fool the discriminator by producing real-like images and the score of the discriminator is very close to one. Likewise, during the training of the discriminator, the generator is made stationary. The discriminator scores the output of the generator regularly and differentiates between original images and generated images. This allows a discriminator to efficiently score fake images near zero.

GANs have been used in a variety of applications ranging from image synthesis to image enhancement. Silva et al. have used Conditional GANs to generate post-treatment PET images of lymphoma using pre-treatment images [4]. This was done to avoid subjecting human beings to harmful radiation and save costs. To achieve this, PET scan images were converted from 3D to 2D and sliced manually for maximum intensity projection. Once the images were extracted, then they were co-registered for achieving the best outcomes based on metrics such as structural similarity index and peak signal-to-noise ratio.

Eltahan et al. implemented GANs to model the spatial distribution of Earth's surface's radiation fluxes in the upward direction [5]. They used the ERAS dataset in the 2000-2020 period. This model was able to reproduce fluxes within the range of $\pm 10\%$ root mean square error. This approach requires additional work to assess the uncertainty of the model and the response. The model's capability can further be improved by increasing the number of filters in the generator to extract more features or by weakening the discriminator.

Abramian and Eklund implemented a CycleGAN model that is based on an unsupervised image-to-image translations framework [6]. This model reconstructs facial image that has been anonymized for data sharing purposes. The reason behind using CycleGAN was to solve the problem of data being unpaired and the problem of identifying a mapping within two domains. Here CycleGAN resolved the problem by employing a cyclic constraint in which the data is converted to a new or different domain and reverted to the original. Results suggested that face blurring does not provide enough protection against attempts to identify the test subject however face removal can be more robust.

Akimoto et al. proposed something new which was never been done earlier which is the generation of 360-degree high-resolution images using GANs [7]. To achieve this, a two-stage generation with series and parallel convolution layers (dilated) was used. In this way, 360-degree input images can be fed and the remaining part of the image would be completed by the GANs. The results of this research indicated that the distortion observed in the final generated images of structures such as buildings was much clear than the other baseline model such as Pix2pixHD [8].

Ali et al. implemented a model which could remove ink marks added by pathologists on detected tumor regions [9]. Ink marks are generally added as they form a part of a medical record and once the record is digitalized, there is no way of removing ink from images. With this model, ink-free images are generated without losing information or image resolution. In this model, a convolution neural net is used to first classify whether it contains ink or not, then a consistent cycle generative adversarial network model for pixel restoration. Results show that this model generates ink-free images. This model was evaluated using Peak Signal to Noise Ratio (PSNR), Structure Similarity, and VIF with final values of 28.73 dB, 0.71, and 0.78 respectively.

Han et al. implemented a model which generates synthetic magnetic resonance images of the most crucial part of the body which is the human brain using GANs [10]. Some difficulties faced during the implementation were fewer contrast images as well as high consistency in different brain images. This model generated 128 x 128 resolution images which even the expert physician couldn't distinguish between synthetic and original during Turing Test. The dataset used for this model is Brats 2016 which contains brain tumor image data. Different types of GAN architectures used in this model are DCGAN and WGAN whereas WGAN showed the most promising results.

More related to our work, researchers have been able to classify and generate specific patterns such as checks and floral designs for apparel, hence generating abstract designs would be an extension of the previous research. Fayyaz et al. designed a model to generate textile designs using GANs [11]. In their approach, they first tried to improve the overall accuracy of state-of-the-art results in the classification of textile design patterns by 2% with the help of data cleaning and labeling. Then on a newly obtained dataset, they applied various generative models such as Wasserstein Generative Adversarial Networks Gradient Penalty (WGANs GP), Deep Convolutional GANs (DCGANs), and Convolutional Variational Autoencoders (CVAEs) for all classes separately and compared the performance of these using the inception score. As per the findings, WGANs seemed more promising, style transfer model was used along with it to generate more appealing textile design patterns.

Our study will be equipped with all modern tools and technologies such as powerful deep-learning libraries and high-specification machines with powerful GPUs which can be useful in reducing model training times and improving model accuracy. With this, the trained models will be able to generate good-quality synthetic images specifically different from the ones present in the learning dataset. Since these models have never been trained on these types of abstract images, it will generate new insights and metrics which will be an addition to previous research.

III. Material and Method

Every machine learning or neural network model undergoes a set of steps in the model-building process. These steps can vary based on the tasks' requirements and objectives. In our research, these steps are shown in Figure 2. The figure identifies the first phase of the model building which consists of Data Gathering, Data Pre-Processing, and Exploratory Data Analysis steps.

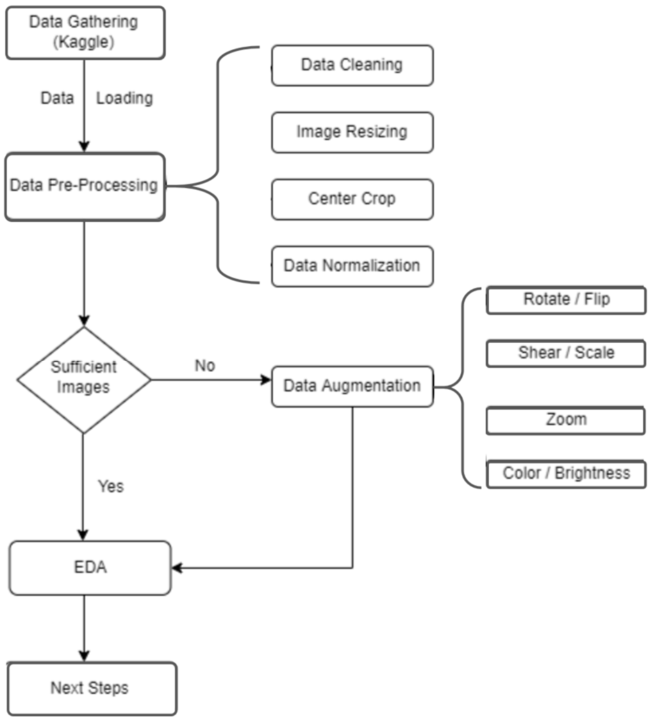

Fig. 2. The first phase of the workflow.

A. Data Gathering

Data gathering or data acquisition is a crucial step in any machine learning or deep learning model creation and any person from the data science community knows that it is better to have too much data than to have less data than is needed. Since this research is based on abstract image synthesis and generation, the dataset required for this research must contain images of this specific kind. There are many datasets available that are specifically created for image syntheses such as the Celeb-500k dataset [12], Facial Features dataset [13], or MNIST dataset [14] but none of these are compatible with the generation of artistic design patterns that can be used in apparel designing. So, a custom dataset that only consists of abstract design images and artistic design images can fulfill our needs. For this research, the data has been acquired from the well-known website Kaggle where the required data is publicly available [15]. Images presented here were originally scraped from the WikiArt website [16]. With the help of scraping, only images that contained artistic patterns were fetched and other images were discarded. The dataset contains 2782 abstract images which will be used for model creation and training.

B. Data Pre-processing

While manually going through the image, it was observed that some images do not contain any artistic design pattern such as those shown in Figure 3. These images were removed from the dataset. Once we have a cleaned dataset, then we will reload the updated dataset to our code and apply further pre-processing techniques to the image dataset.

Fig. 3. Examples of images that do not contain artistic or abstract patterns.

Images in our dataset are of varied sizes as they all originated from a different source and were clubbed together in a single dataset. But these are still raw images and need to be processed so that all images are the same in size and color channels having RGB encoding. The default setting for a GAN model is 64x64 dimensions so all images will be resized to this size.

Data normalization is an important step to maintain numerical stability during training. Normalizing the data increases the probability of quicker training and more stable gradient descent. As a result, the pixel values of the input images will be standardized to a range of 0–1. There can be several ways of normalizing the image, the first one is common for RGB images and other approaches depending on the project requirements.

C. Data Augmentation

This technique is used when the quantity of data or image samples is not enough for a classification problem or when there is very little data available. Augmentation is mostly used in the case of image data to increase the sample count and variance. Augmentation technique is required to be applied on images where images will be rotated at certain degrees, scaled to a certain limit, and flipped horizontally. All these methods will be used in multiple runs to achieve good performance.

D. Building the GAN model

We developed two GAN models based on different GAN architectures, namely 1) Deep Convolution GAN (DCGAN) [17] and b) Wasserstein GAN (WGAN) [18] with weight clipping. Once the models are trained and generate results, then both models will be evaluated as per the next steps as shown in Figure 4. The diagram shown in this figure is a continuation of that shown in Figure 2 and portrays a common flow chart for both GAN architectures.

The DCGAN model uses convolution layers instead of fully connected layers. Convolution layers are very powerful and have been used in the field of computer vision for many years. They are capable to extract features from an image by applying different filters without affecting the correlation within neighbor pixels. This substitution of dense layers with convolution layers has a great impact on training efficiency. Other changes that further stabilize the GAN training are the replacement of pooling layers with strided convolution layers. This approach boosts the performance, and it is recommended to use strided layers in both Generator and Discriminator. The use of Batch Normalization in the DCGAN model helps in convergence by reducing noise and improving the diversity of generated images. We use Rectified Linear Unit (ReLU) activation function for the hidden layer of the Generator and the Hyperbolic Tangent (tanh) activation function for the outer layer of the Generator. Likewise, we use Leaky ReLU and Sigmoid activation functions for the hidden layer and outer layer of the Discriminator.

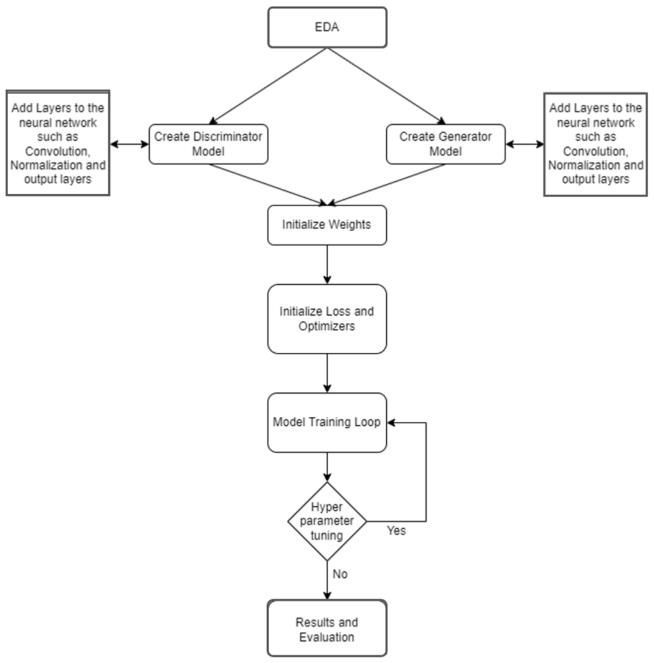
EDA
Add Layers to the neural network such as Convolution, Normalization and output layers
Create Discriminator Model
Create Generator Model
Add Layers to the neural network such as Convolution, Normalization and output layers
Initialize Weights
Initialize Loss and Optimizers
Model Training Loop
Hyper parameter tuning
Yes
No
Results and Evaluation

Fig. 4. The second phase of the workflow.

In a typical GAN model, let x represent an image, $D(x)$ will represent the discriminator network that will output a probability number which states if x is a real image (labeled 0) or a fake image (labeled 1) i.e. whether x is coming from training data or is it generated as a fake by generator network. The input x to the discriminator should be a three-channel image of dimension 64x64 and based on model functionality $D(x)$ should be high for training image data and low for generated fake images. Similarly, for the generator network, let z represents a vector from latent space of a similar dimension. $G(z)$ function when the generator network is applied on vector z would result in a new vector from training data space. The generator function tries to learn the data distribution of real images and with help of that training, it tries to generate fake images like the identified distribution.

The discriminator is trained to maximize the $\log(D(x)) + \log(1 - D(G(z)))$ function. We achieve this by following the following steps:

1. Fetch real samples from training data and create a small batch.
2. Propagate forward this batch through model D
3. Calculate the loss $\log(D(x))$
4. Calculate the gradients using stochastic gradient descent in the backward propagation
5. Now create fake samples by passing a latent vector z from generator model G
6. Propagate forward this batch of fake images from D and calculate the Loss as $\log(1 - D(G(z)))$
7. Calculate the gradients with backward propagation
8. Pass these gradients obtained for real and fake batches from D's Adam optimizer

The generator is trained to maximize the $\log(1 - D(G(z)))$ function. We achieve this by following the following steps:

1. Classify G's output from the previous D's training step and calculate G's loss
2. Calculate gradients for G in backpropagation
3. Update the params for G using the G's Adam optimizer

The progress of model training can be observed by generating images after each complete iteration. The statistics generated such as Discriminator loss, Generator Loss, the output of Discriminator on real batch, the output of Discriminator on fake batch, etc can also be observed after each iteration and plotted on a time-series graph.

When developing the WGAN architecture, the Discriminator model is replaced by a critic which rates the generated image for its real-ness and fake-ness value. The model development is similar to that of DCGAN with the following modifications:

1. While creating a WGAN model, we will replace the sigmoid or ReLU activation function with a linear activation which will predict the score for the real or fake image instead of just classifying it as real or fake.
2. The class labels of 0 and 1 in DCGAN are replaced by -1 (real) and +1 (fake) as there is nothing fixed in the case of scores.
3. The Wasserstein loss function [19] will be used as a critic instead of a classifier.
4. The model weights for the critic will be constrained to a limited range after every mini-batch update
5. The critic model will be updated more than the generator model.

6. The Adam optimizer [20] in the DCGAN model will be replaced by RMSProp [21] with a smaller learning rate.
7. The gradients of the critic in the WGAN model require clipping unlike the gradients in the DCGAN model and so this step will also be applied while creating the WGAN model.

The training of a WGAN model is more stable than that of a DCGAN as it is not affected by model architecture or by the hyperparameters.

IV. Experiment and Result Analysis

The model implementation is one of the phases of the deep learning lifecycle in which models are trained on initialized weights and retrained again on tuned hyperparameters to achieve the required results. For both DCGAN and WGAN models, it is crucial to initialize weights randomly from a normal distribution. The value of mean and standard deviation must be 0 and 0.02 for convolution and convolution transpose layers, and 1 and 0.02 for batch normalization layers, respectively. This is the first step which is applied after both Generator and Discriminator models are initialized. The Loss function for the DCGAN model is Binary Cross Entropy loss (BCE) [22] and the optimizers used are Adam [20]. The loss function for the WGAN model is Earth Movers Distance or Wasserstein distance [23] and the optimizers used are RMSProp [21]. Some examples of the input images used for the models' training are shown in Figure 5.

Fig. 5. Some examples of the input images used for the models' training before conversion to tensors.

Hyperparameter optimization is carried out during the training steps on eight hyperparameters. The name and the final values of the optimized hyperparameters are summarized in Table I.

TABLE I. Optimized Hyperparameter Values

Parameters	DCGAN Final	WGAN Final
Batch Size	24	32
Image Size	64 x 64	64 x 64
Latent Vector	100	100
Feature Maps (fm_g)	64	64
Feature Maps (fm_d)	64	64
Number of Epochs	400	400
Learning Rate	0.0002	0.0002
Optimizer Parameter	0.5	0.01

The Generator and Discriminator losses during training of the best DCGAN and WGAN models are shown in Figure 6.

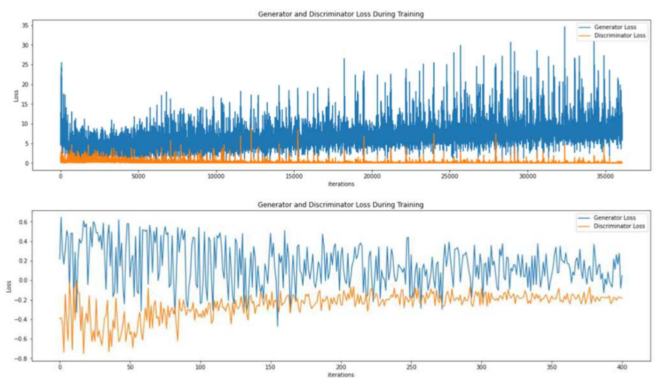

Fig. 6. Generator and Discriminator Losses during training of the DCGAN model (top) and the WGAN model (bottom).

Figure 7 shows some examples of the output images generated by the best DCGAN model. The images all have identical dimension, which is 64x64 pixels. The patterns generated in this image are distinguishable from the input image set and the quality is very good. It can be observed that some images in this model are very similar and seem repeated. This problem is known as *Mode Collapse* in which similar images are generated even though the inputs are different.

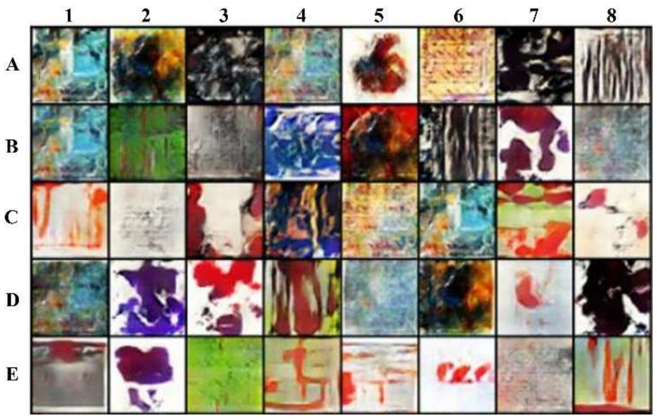

Fig. 7. Some examples of output images of the best DCGAN model. The occurrence of similar-looking images in A1-B1-D1-C6, A2-D6, and A8-B6 cells suggests the presence of a Mode Collapse problem in the model.

Figure 8 shows some examples of the output images generated by the best WGAN model. As before, the images all have identical dimension, which is 64x64 pixels. The patterns generated in this image are also distinguishable from the input image set.

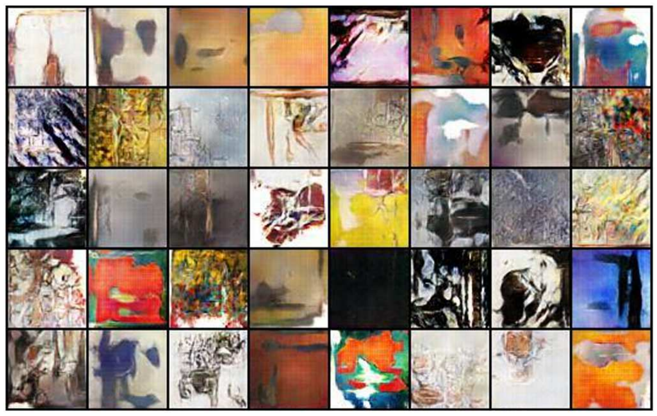
Fig. 8. Some examples of output images of the best WGAN model. No similar-looking images were observed.

The quality and diversity of images generated by the WGAN model are far better than the DCGAN model's generated images. All patterns generated are unique and no mode collapse issue occurred in this model. Patterns generated are completely abstract in design are represents an artistic form.

V. Conclusion

In this paper, we have presented the results of our investigation into using Generative Adversarial Networks on abstract images to produce abstract motif designs. We have investigated the different pre-processing techniques that can be used on an abstract image dataset and how the GAN models can be evaluated when the generated images are abstract in nature. We achieved these by generating abstract design patterns using two state-of-the-art models DCGAN and WGAN, by identifying the best-performing model, and by generating textile design patterns with the finalized model. Our findings showed that both DCGAN and WGAN models have generated very good abstract design patterns however the WGAN model has generated better quality and more diverse outcomes than the DCGAN model where image patterns are not as diverse and where some of the images are repeated due to Mode Collapse.

References

[1] Grand View Research, "Textile Market Size, Share & Trends Analysis Report By Raw Material (Cotton, Wool, Silk, Chemical), By Product (Natural Fibers, Nylon), By Application (Technical, Fashion), By Region, And Segment Forecasts, 2022 - 2030," 2021. [Online]. Available: https://www.grandviewresearch.com/industry-analysis/textile-market. [Accessed: 22-Oct-2022].

[2] J. Goodfellow Ian, P.-A. Jean, M. Mehdi, X. Bing, W.-F. David, O. Sherjil, and C. Courville Aaron, "Generative adversarial nets," in *Proceedings of the 27th international conference on neural information processing systems*, 2014, vol. 2, pp. 2672–2680.

[3] S. Wang, T.-Z. Huang, J. Liu, and X.-G. Lv, "An alternating iterative algorithm for image deblurring and denoising problems," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 19, no. 3, pp. 617–626, 2014.

[4] G. Silva, I. Domingues, H. Duarte, and J. A. M. Santos, "Automatic Generation of Lymphoma Post-Treatment PETs using Conditional-GANs," in *2019 Digital Image Computing: Techniques and Applications (DICTA)*, 2019, pp. 1–6.

[5] M. Eltahan, N. Daoud, and K. Moharm, "Generative Adversarial Networks (GANs) for spatial upward fluxes radiation estimation," in *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, 2021, pp. 1–5.

[6] D. Abramian and A. Eklund, "Refacing: reconstructing anonymized facial features using GANs," in *2019 IEEE 16th international symposium on biomedical imaging (ISBI 2019)*, 2019, pp. 1104–1108.

[7] N. Akimoto, S. Kasai, M. Hayashi, and Y. Aoki, "360-degree image completion by two-stage conditional GANs," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 4704–4708.

[8] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional GANs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8798–8807.

[9] S. Ali, N. K. Alham, C. Verrill, and J. Rittscher, "Ink removal from histopathology whole slide images by combining classification, detection and image generation models," in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, 2019, pp. 928–932.

[10] C. Han, H. Hayashi, L. Rundo, R. Araki, W. Shimoda, S. Muramatsu, Y. Furukawa, G. Mauri, and H. Nakayama, "GAN-based synthetic brain MR image generation," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, 2018, pp. 734–738.

[11] R. A. Fayyaz, M. Maqbool, and M. Hanif, "Textile Design Generation Using GANs," in *2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2020, pp. 1–5.

[12] J. Cao, Y. Li, and Z. Zhang, "Celeb-500k: A large training dataset for face recognition," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 2406–2410.

[13] Y. Tian, C. Suzuki, T. Clanuwat, M. Bober-Irizar, A. Lamb, and A. Kitamoto, "Kaokore: A pre-modern Japanese art facial expression dataset," *arXiv Prepr. arXiv2002.08595*, 2020.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[15] B. Boulé, "Abstract Art Gallery," *Kaggle*, 2022. [Online]. Available: https://www.kaggle.com/datasets/bryanb/abstract-art-gallery. [Accessed: 22-Oct-2022].

[16] WikiArt, "Visual Art Encyclopedia," *WikiArt*. [Online]. Available: https://www.wikiart.org/. [Accessed: 22-Oct-2022].

[17] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *4th Int. Conf. Learn. Represent. ICLR 2016 - Conf. Track Proc.*, pp. 1–16, 2016.

[18] M. Arjovsky, S. Chintala, and L. Bottou, "{W}asserstein Generative Adversarial Networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, vol. 70, pp. 214–223.

[19] J. Brownlee, "How to develop a wasserstein generative adversarial network (wgan) from scratch." 2019.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv Prepr. arXiv1412.6980*, 2014.

[21] F. Zou, L. Shen, Z. Jie, W. Zhang, and W. Liu, "A sufficient condition for convergences of adam and rmsprop," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11127–11135.

[22] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Ann. Oper. Res.*, vol. 134, no. 1, pp. 19–67, 2005.

[23] C. Frogner, C. Zhang, H. Mobahi, M. Araya, and T. A. Poggio, "Learning with a Wasserstein loss," *Adv. Neural Inf. Process. Syst.*, vol. 28, 2015.