Kütahya Dumlupınar University
Institute of Graduate Studies

Journal of Scientific
Reports-A
E-ISSN: 2687-6167
Number 50, September 2022

RESEARCH ARTICLE

WY-NET: A NEW APPROACH to IMAGE SYNTHESIS with GENERATIVE ADVERSARIAL NETWORKS

Emrullah ŞAHİN[1*], Muhammed Fatih TALU[2]

[*1]Kütahya Dumlupınar University, Faculty of Engineering, Department of Software Engineering, Kütahya, essahin950@gmail.com, ORCID: 0000-0002-3390-6285
[2]İnönü University, Faculty of Engineering, Department of Computer Engineering , Malatya, fatihtalu@gmail.com, ORCID: 0000-0003-1166-8404

Receive Date:21.07.2022 *Accepted Date: 15.09.2022*

ABSTRACT

Conditional image synthesis is the translation of images from different domains with the same dimensions into each other. Generative Adversarial Networks (GANs) are commonly used in translation studies in this field. With the classical GAN approach, data are transferred between the encoder and decoder of the generator network in the image translation. While this data transfer increases the quality of the translated image, it also leads to data dependency. This dependence has two negative effects: First, it prevents the understanding of whether the encoder or the decoder causes the error in the translated images, other causes the image synthesis quality to depend on the parameter increment of the network. In this study, two different architectures (dY-Net, uY-Net) are proposed. These architectures are developed on the principle of equalizing high-level feature parameters in cross-domain image translation. The first of these architectures concentrates on the speed of image synthesis, the other on its quality. There is a significant reduction in data dependency and parameter space in the dY-Net architecture, which concentrates on speed performance in image synthesis. The uY-Net architecture, which concentrates on image synthesis quality, attempts to maximize the results of metrics that measure quality like SSIM and PSNR. Three different datasets (Maps, Cityscapes, and Denim2Mustache) were used for performance testing of the proposed architectures and existing image synthesizing approaches. As a result of the tests, it has been seen that the proposed architecture synthesized images with similar accuracy, although it has approximately 66% parameters compared to DiscoGAN, which is one of the existing approaches. The results obtained show that WY-Net architectures, which provide high performance and translation quality, can be used in image synthesis.

Keywords: *Generative adversarial networks, Image synthesis, Deep learning, Image to image translation*

1. INTRODUCTION

Deep learning is a subfield of machine learning where the features that best express data can be learned autonomously. Using this area is increasing gradually, and it is also multiplying in the subareas it branches. Recently, deep learning-based applications have been increasing. For example, it is used in many fields, such as language translation, chatbots, face identification, voice signature, disease diagnosis, data augmentation, autonomous vehicles, and anomaly analysis. Although this field

contains many learning methods, one of the most popular todays is Generative Adversarial Networks (GANs). Goodfellow and his team developed this learning method based on the min-max algorithm in 2014 [1].

GANs are a special type of neural network developed to model relationships between samples in a dataset [1-3]. The network tries to learn the features that best express these samples to model the relationship between the samples [2,3]. The basic structure of these networks is based on the game theory of the famous mathematician John Nash and comprises two convolutional modules that learn by working with each other in an adversarial manner [1]. One of these modules is called the generator network, and the other is called the discriminator network. The generative network tries to simulate a random or special data block (sampler) given to it to the desired data sample. The discriminant network is a classifier that tries to distinguish between the synthesized sample by the generator network and the real data sample. During the training phase, the generator and discriminant network parameters are fed with the similarity cost (loss) of the real and predicted (synthesized) outputs. The working mechanism of these models is shown in Figure 1. In this schema, real data with synthesized data from the generator network are given as input to the discriminator network. Then the cost of similarity between the outputs of the discriminator network is calculated. With this cost, the discriminator and generator network are fed.

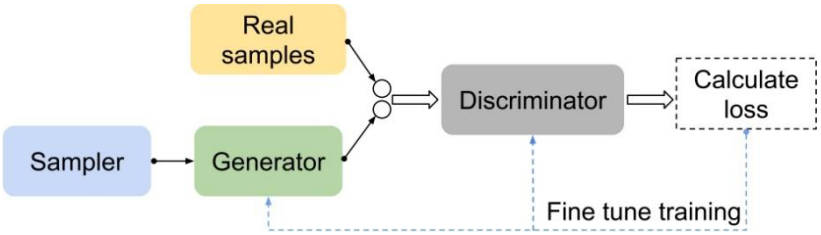

Figure 1. The basic schema of GAN architectures.

Since the development of Generative adversarial networks, multiple types have been developed for use in image generation and manipulation. One of the first examples in this area is the DCGAN [2] architecture developed by Radford et al. in 2015. This architecture can learn the numerical distribution of its features by training with examples in a dataset consisting of images. It is a light and powerful model that can be used in data augmentation. One of the biggest problems in image studies is the low-resolution problem. In solving this problem, the perceptual similarity cost-based SRGAN [3] architecture developed by Ledig et al. has achieved significant success. Convolutional neural networks-based VGG [4] network is used so that the content in the images is not distorted while synthesizing low resolution images in high resolution. The Progressive GAN [5] architecture, which performs progressive learning to reduce distortions in high resolution synthesized images, was developed by Karras et al. in 2017. After learning with 4x4 image scale in the architecture training phase, it continues until it synthesizes 1024x1024 images by increasing the scale step by step. The networks with attention mechanisms have been developed so that GAN architectures focus on the important parts instead of focusing on the whole image. One of these studies, the spatial attention and spectral normalization-based SAGAN [6] architecture, was developed by Zhang et al. in 2018 to reduce the instability of the learning curve due to unbalanced learning in the training phase of the generator and discriminator network in GAN architectures. The StyleGAN [7] architecture fed with content and style image was developed by Karras et al. to transfer any style image to a content image

in 2018. The modulation and reverse modulation-based StyleGAN2 [8] architecture to minimize the distortions on the network sourced image in StyleGAN architecture was developed in 2019. The StyleGAN3 [9] architecture by including Fourier-based features and various changes in this architecture to solve the signal problem from the hidden field caused by translation and rotation in the StyleGAN architectures was developed in 2021. The BigGAN [10] architecture, which can generate class-level images on the ImageNet dataset for high-resolution synthetic image generation, was developed by Brock et al. in 2018. This architecture tries to minimize cosine similarity without restricting the norms between binary filters for quality and diverse image synthesis. This regularization term, added to the parameters of the filters, enables the synthesizing of high-quality images belonging to different classes. The BigGAN-based BigBiGAN [11] architecture to develop a system capable of generating more realistic images was developed by Donahue et al. in 2019. This architecture has an encoder that can convert real images into a random variable, and a discriminator network operating with a binary cost function. For better quality learning, the system tries to bring the values from the encoder network and the values from the hidden field to the same representation.

Image to image translation: A major challenge in image translation is to get a high-dimensional image from a low-dimensional visual space [13,14,16]. An example can be given for a better understanding. Translations such as coloring a black-and-white image [13], getting a semantic label map from an image with an edge map [13,16], converting a real image into an animation image [12] can be given as examples. These translations are difficult to perform with basic image processing algorithms. It is known that GAN architectures are widely used to solve these high-level problems [2-16]. Today, researchers for image-to-image translation have proposed many GAN architectures [12-16]. Although the architectures have differences among themselves, the primary aim is to generate a quality artificial image at an appropriate temporal cost by making an excellent translation between the two domains. One of the first architectures developed in the field of image-to-image translation is Pix2Pix [13]. This architecture was developed by Isola et al. in 2016 to translate between pairs of images in supervised datasets. The architecture includes the L_1 metric in addition to the traditional GAN loss. A new version of this architecture, the Pix2PixHD [14] architecture, was developed by Wang et al. in 2018 to synthesize realistic images from high-resolution semantic image maps. Researchers at Nvidia enhanced the SPADE [15] architecture by adding spatial adaptive normalization on top of the basic structure of the Pix2PixHD architecture to produce realistic nature photographs from unlabeled semantic image maps. The CycleGAN [16] architecture based on double-sided validation was developed by Zhu et al. to perform image-to-image translation in unsupervised datasets. The MixNMatch [17] architecture for various content generation and in-depth image change was developed by Li et al. in 2020. This architecture has a design that takes content, shape, poses, and background images and combines them. To increase the quality of semantic map-based image synthesis, the study called "Panoptic-based Image Synthesis" for panoptic map-based image production was developed by Dundar et al. in 2020 [18]. The GFP-GAN [19] architecture with spatial feature conversion to synthetically generate old or poor-quality images of human faces in high-quality, noiseless, and colorful was developed by Wang et al. in 2021. The Real-ESRGAN [20] architecture, which can reduce effects such as low resolution, blur, compression, and noise in an image and concentrate on over one problem area at the same time for quality image synthesis, was developed by Wang et al. in 2021. The Fourier convolution-based LaMa [21] architecture to remove unwanted objects or regions on the image without disturbing the basic design of the actual image was developed by Suvorov et al. in 2021.

JSR A
Journal of Scientific Reports

In this study, we proposed a new architecture can translate from image-to-image more efficiently. Performance comparison of this architecture was made with DiscoGAN architecture on three different datasets. In Section 2.1, explanations are given about the similarity metrics used in the study. In Section 2.2, DiscoGAN [22] architecture's basic design is explained. In Section 2.3, the basic design of the proposed (WY-Net) architecture is explained. Information about the datasets used is given in Section 2.4. In Section 3, the results obtained in the study are shared.

2. MATERIAL AND METHODS

2.1. Similarity Metrics

In this section are given the basic similarity methods frequently used in the study.

2.1.1. SSIM

Structural similarity metric (SSIM) is a method that analyzes the perceived change in the image along with important perceptual properties, such as brightness, masking, and contrast. This metric is a statistical measurement built on the mean (μ) and standard deviation (σ) parameters in calculating the similarity between the image pairs, ignoring the positional difference between the pixels in the image [23].

Calculation of structural similarity between real (x) and predicted (y) image statistically is shown in Eq. (1). In this equation, μ_x and μ_y denote the pixel mean of the real and predicted image, the variance of σ_x^2 and σ_y^2, while σxy denotes the covariance between the real and predicted image. In addition, the constant values $c_1 = (k_1 L)^2$ and $c_2 = (k_2 L)^2$ are calculated by taking the L value 255., which specifies the pixel pitch.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{({\mu_x}^2 + {\mu_y}^2 + c_1)({\sigma_x}^2 + {\sigma_y}^2 + c_2)} \tag{1}$$

2.1.2 MSE

It is a comparison metric where the square of the point difference is taken to calculate the similarity cost between two data samples [24]. The main formula of this metric is shown in Eq. (2). In the equation, each pixel difference between the real (x) and predicted (y) image is squared and summed. Divide the total result by the number of pixels (n).

$$L_2 = MSE(x,y) = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \tag{2}$$

2.1.3. MAE

It is a comparison metric in which the absolute value of the point error is taken to calculate the similarity cost between two data samples [25]. The main formula for this metric is shown in Eq. (3). In the equation, the absolute value of each pixel difference between the real (x) and predicted (y) image is taken and summed. Divided the total result by the number of pixels (n). While measuring, the direction of the error is not considered, it focuses on the absolute difference.

$$L_1 = MAE(x,y) = \frac{1}{n}\sum_{i=1}^{n}|x_i - y_i| \tag{3}$$

JSR A
Journal of Scientific Reports

2.1.4. PSNR

It is a logarithmic-based metric used for the ratio between the maximum power of a signal and the power of noise that affects the accuracy of its representation [26]. The signal is a real image or data, while noise is the error caused by compression or distortion in the data. This ratio between two images is calculated in decibels. The main formula for this metric is shown in Eq. (4). For the measurement between the real (x) and predicted (y) image in the equation, the $MSE(x,y)$ value and the largest pixel value of the real image (MAX_x^2) are included in the calculation.

$$PSNR\ (x,y) = 10 * log_{10}\left(\frac{MAX_x^2}{MSE(x,y)}\right) \tag{4}$$

The higher the measurement value, the better the signal quality, that is, the lower the cost between the real and the predicted data.

2.1.5. Hinge embedding loss

It is a metric often used to calculate the similarity between nonlinear or semi-supervised data blocks [27]. This metric measures the distance between a real input vector (x) and a label vector (y) (containing 1 or -1). The main formula of the metric is given in Eq. (5). The L value in the equation contains the input and label list. The number of features in the list is expressed as n.

$$L_{HE} = \begin{cases} mean(L), & if\ \ reduction = 'mean', \\ sum(L), & if\ \ reduction = 'sum', \end{cases} \tag{5}$$

$$L = \sum_{i}^{n} \ell(x_i, y_i) = \begin{cases} L_2(x_i, y_i), & if\ \ y_i = 1, \\ max(0, m - L_2(x_i, y_i)), & if\ \ y_i = -1, \end{cases}$$

2.2. DiscoGAN

The DiscoGAN (Learning to discover cross-domain relations with generative adversarial networks) is a conditional GAN architecture developed to explore the relationship between different datasets and synthesize quality images [22]. This architecture has a structure that can work on supervised or unsupervised datasets.

2.2.1. Generator network

The DiscoGAN architecture has two generator networks with the same structure. This architecture learns two separate translation functions: the first network (*G_X2Y*) learns to translate the image in the X-domain (*Real_X*) into the predicted Y-image (*Fake_Y*), while the second network learns to translate the image in the Y-domain (*Real_Y*) into the predicted X-image (*Fake_X*). After the initial processing, the architecture generates the reconstructed *Recons_X* image by passing the predicted *Fake_Y* image through the *G_Y2X* network for complete learning, while passing the *Fake_X* image through the *G_X2Y* network to generate the reconstructed *Recons_Y* image. The basic diagram of the generator networks in the DiscoGAN architecture is shown in Figure. 2. The first translation takes place from *X* to *Y* and the other from *Y* to *X*. Reconstructed images are used for double-sided verification. This validation is calculated between the real and reconstructed images with the L_2 metric.

JSR A
Journal of Scientific Reports

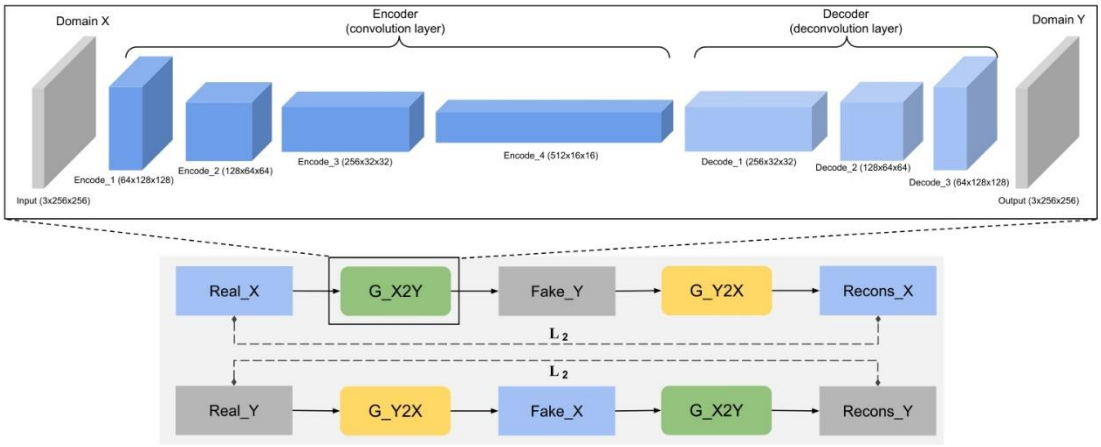

Figure 2. The basic schema of DiscoGAN generator networks. DiscoGAN's generator *G_X2Y* and *G_Y2X* networks learn 2 separate translations.

2.2.2. Discriminator network

The DiscoGAN architecture has two discriminator networks with the same structure. First of these networks (*D_X*) tries to classify the output of the two images as real or fake by trying to learn the properties of *Real_X*, which is the real image in the X-domain, and *Fake_X*, which is the fake image. The other network (*D_Y*) tries to classify the output of the two images as real or fake by trying to learn the properties of *Real_Y*, which is the real image, and *Fake_Y*, which is the fake image, in the Y-domain. The basic schematic of the discriminator networks is given in Figure 3.

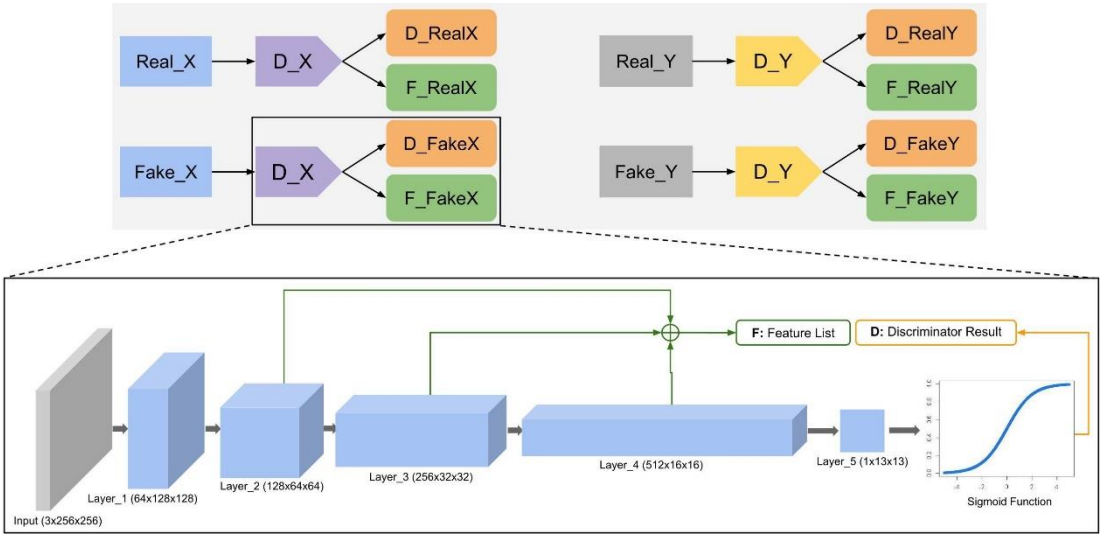

Figure 3. The basic schema schematic of DiscoGAN discriminator networks. DiscoGAN's discriminator *D_X* and *D_Y* networks learn 2 separate classification processes.

JSR A
Journal of Scientific Reports

The first classification takes place between *Real_X* and *Fake_X* and the other between *Real_Y* and *Fake_Y*. The feature list of each image is extracted so that the generator networks can learn better. Outputs starting with the prefix *D* are used for the cost of discriminator, and *F* for generator. According to this scheme, two outputs are obtained from the image given as input to the discriminator network. The first output is the passed output from the activation function that the discriminator network uses for the adversarial loss. The second output is the feature list obtained from layers 2, 3, and 4 of the networks. This feature list is used for the cost function of the generator network. For example, the *D_RealX* result obtained by giving the *Real_X* image to the *D_X* network represents the output of the discriminator network, while the *F_RealX* value represents the feature list of this image.

2.2.3. Loss functions

The cost function of the generator networks in the DiscoGAN architecture is built on reconstructed, adversarial, and feature losses. The cost of discriminator networks in the architecture is based on the adversarial loss value in traditional GAN architectures.

Generator Loss. In this architecture, the L_2 metric is used for reconstructed loss. This is computed between the *Recons_X* and *Recons_Y* images reconstructed with the input images *Real_X* and *Real_Y*. Its formulation is expressed as Eq. (6).

$$\mathcal{L}_{const_X} = L_2(Real\_X, Recons\_X) \tag{6}$$

The hinge embedding loss function is used for feature cost calculation. This process is calculated between the real (*F_RealX*) and fake (*F_FakeX*) feature list obtained from the discriminator network. It is shown in Eq. (7). It is calculated by taking the λ_1 value in the equation to 0.9.

$$\mathcal{L}_{feature_X} = \lambda_1 * L_{HE}(F\_RealX, F\_FakeX) \tag{7}$$

The adversarial loss value is calculated with the fake (*D_FakeX*) value obtained from the activation function of the discriminator network. This process is shown in Eq. (8). It is calculated by taking the λ_2 value in the equation to 0.1.

$$\mathcal{L}_{adv_X} = \lambda_2 * \arg\min_{G\_X2Y}(log(1 - D\_FakeX)) \tag{8}$$

The loss function required for the X-domain of the network is expressed like Eq. (9). The starting ratio in the equation is initially given as $r = 0.01$. This value is changed to $r = 0.5$ after 10000 iterations of training.

$$\mathcal{L}_{G\_X2Y} = r * \mathcal{L}_{const_X} + (1 - r) * \left(\mathcal{L}_{feature_X} + \mathcal{L}_{adv_X}\right) \tag{9}$$

The overall loss of the generator networks is calculated for both areas and then added up. The total loss function is written like Eq. (10).

$$\mathcal{L}_G = \mathcal{L}_{G\_X2Y} + \mathcal{L}_{G\_Y2X} \tag{10}$$

JSR A
Journal of Scientific Reports

Discriminator Loss. The required loss value for the X-domain of the discriminator networks is calculated between the *D_RealX* blocks of the real data and the *D_FakeY* blocks of the fake data we want to translate. This process is illustrated in Eq. 11.

$$\mathcal{L}_{adv_{D\_X}} = \underset{D\_X}{argmax}(log(D\_RealX) - log(1 - D\_FakeX)) \tag{11}$$

The total loss of the two discriminator networks is calculated like Eq. 12.

$$\mathcal{L}_D = \mathcal{L}_{adv_{D\_X}} + \mathcal{L}_{adv_{D\_Y}} \tag{12}$$

2.3. WY-Net: Proposed Method

When we want to translate from one image domain to another with GAN architectures, unlike Auto-encoders, the encoder and decoder of the generator network are evaluated together. The disadvantage of this is that it cannot be understood that the real error is from the encoder or decoder. The WY-Net architecture proposed in this study allows us to understand this problem by trying to reduce the features of the images in the two domains to the common denominator while translating from one image domain to another image domain. We delegate the feature extraction of images from two domains to a single network. The advantage of this is that it makes the architecture lighter and faster, while also showing how the network will translate between image domains.

2.3.1. Improving generator network

WY-Net architecture has a generator network with one encoder and two decoders. The encoder tries to learn the features of the data of the X and Y-domain. One of the decoders tries to reconstruct the data belonging to this domain from the features of the X-domain, the other does the same for the Y-domain. This architecture learns two separate translation operations, the first translation is from the X-domain to the Y-domain, the second translation is from the Y-domain to the X-domain. The first translation yields three outputs, *Fake_Y*, *Feature_X*, *Recons_X* data respectively. The first output is the image in the X-domain translated to the Y-domain. The second output is the feature map of the data in the X-domain obtained after passing through the encoder. The third output is the reconstructed image of the data in the X-domain, with this image double-sided validation is performed. The same translation process is done for the Y-domain. The WY-Net architecture proposed in this study has two different uses: 1) dY-Net based on DiscoGAN generator network; 2) uY-Net based on U-Net [28] generator network. The dY-Net architecture concentrates on temporal and hyperparameter optimization, while uY-Net architecture concentrates on image quality. Two different design prototypes of the WY-Net architecture are shown in Figure 4. The first translation takes place from *X* to *Y* and the other from *Y* to *X*. Reconstructed images are used for double-sided verification. This validation is calculated between the real and reconstructed images with the L_2 metric. For a better comparison, a similarity calculation is made between the real and fake data with the L_1 metric. Then, the high-level features of the real *X* and *Y* images are compared with the L_{HE} metric and reduced to the common denominator.

JSR A
Journal of Scientific Reports

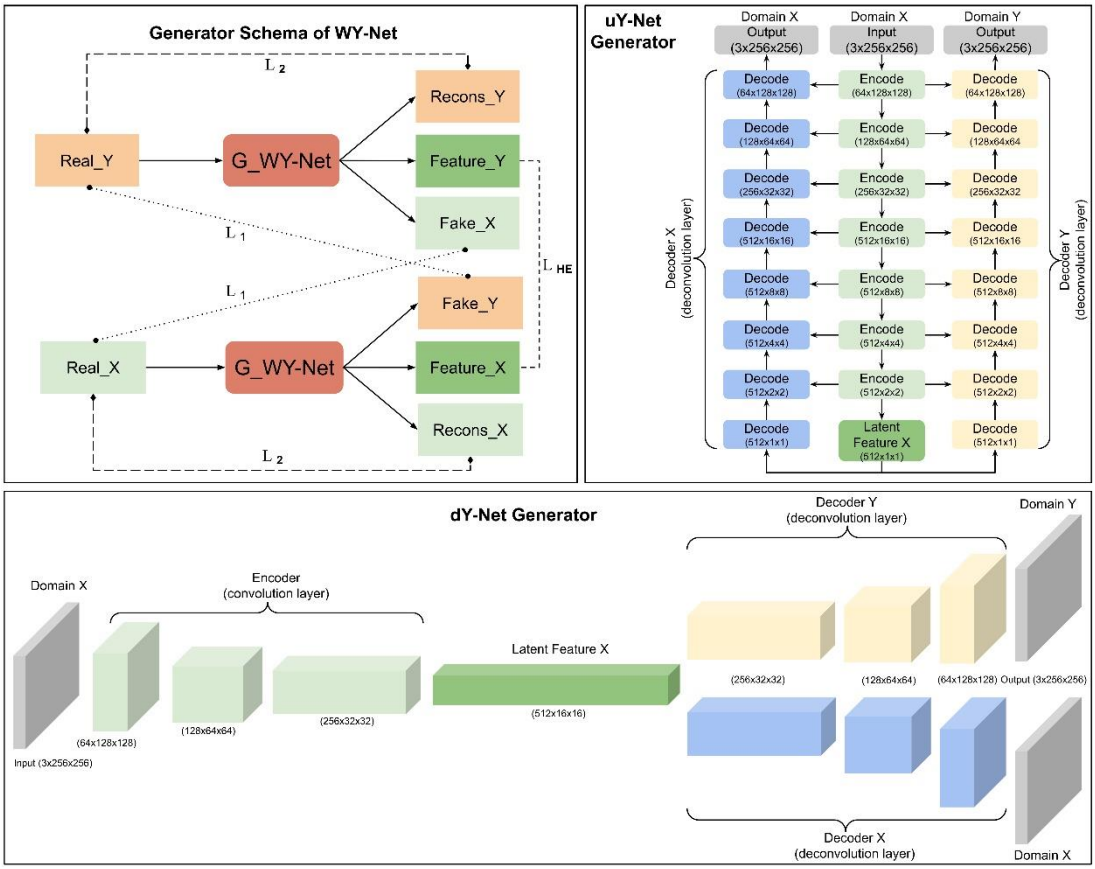

Figure 4. The basic schema of WY-Net generator network. Generator network (G_WY-Net) of WY-Net architectures (dY-Net, uY-Net) learn 2 separate translations.

2.3.2. Improving discriminator network

The WY-Net architecture has a discriminator network that takes real images (*Real_X*, *Real_Y*) and fake images (*Fake_X*, *Fake_Y*) together as input data, unlike traditional GAN architectures. This process is to be compatible with the generator network because, while the generator network tries to reduce the features of the two domains to the same denominator, the discriminator network must also reduce the images of these domains to the same denominator. Thus, the translation process between the two domains is ensured to be balanced. The basic diagram of the discriminator network is given in Figure 5. Discriminator network (D_WY-Net) of WY-Net architectures (dY-Net, uY-Net) tries to reduce real (*Real_X*, *Real_Y*) and fake (*Fake_X*, *Fake_Y*) data to a common denominator for better understand cross-domain image similarity. Outputs starting with the prefix *D* are used for the cost of discriminator, and *F* for generator. In the first stage of this diagram, the output of the discriminator network (*D_Real*) and the feature list (*F_Real*) of the real images are shown, while in the second stage, the *D_Fake* and *F_Fake* data of the synthesized fake images are shown. WY-Net discriminator network architecture was developed based on DiscoGAN.

JSR A
Journal of Scientific Reports

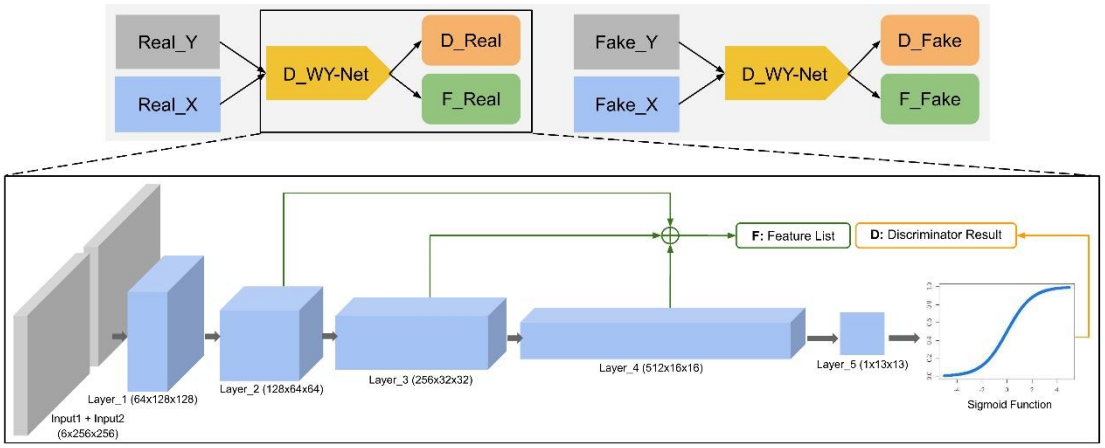

Figure 5. The basic schema of WY-Net discriminator network.

2.3.3. Improving loss functions

The generator network loss function of WY-Net architecture is built on reconstructed, similarity, adversarial, and feature losses. The loss of the discriminator network is based on the traditional adversarial loss value. As the adversarial loss value, the hinge loss [29] technique was used instead of the min-max algorithm.

Generator Loss. In this architecture, the reconstruction loss is calculated using the L_2 metric as in Eq. 8. This is computed between the *Recons_X* and *Recons_Y* images reconstructed with the input images *Real_X* and *Real_Y*.

The similarity loss is calculated using the L_1 metric. This process is calculated between the input images *Real_X* and *Real_Y* and the translated *Fake_Y* and *Fake_X* images to the other domain. It is shown in Eq. 13.

$$\mathcal{L}_{similarity_X} = L_1(Real\_X, Fake\_X) \tag{13}$$

The hinge embedding loss function is used for feature cost calculation. This process is the sum of the result of the real (*F_Real*) and fake (*F_Fake*) feature list obtained from the discriminator network and the result of *Feature_X* of the X-domain and *Feature_Y* of the Y-domain from the generator network. This process is shown in Eq. 14. It is calculated by taking the λ_1 value in the equation to 0.9.

$$\mathcal{L}_{feature} = \lambda_1 * \left(L_{HE}(F\_Real, F\_Fake) + L_{HE}(Feature\_X, Feature\_Y)\right) \tag{14}$$

Adversarial loss: It is calculated on the fake (*D_Fake*) result passed through the activation function of the discriminator network. This process is expressed as Eq. 15. It is calculated by taking the λ_2 value in the equation to 0.1.

$$\mathcal{L}_{adv} = \lambda_2 * mean(-D\_Fake) \tag{15}$$

JSR A
Journal of Scientific Reports

The overall loss of the generator network is expressed as Eq. 16. The starting ratio in the equation is initially given as $r = 0.01$. This value is changed to $r = 0.5$ after 10000 iterations of training.

$$\mathcal{L}_G = r * \left(\mathcal{L}_{const_X} + \mathcal{L}_{const_Y} + \mathcal{L}_{similarity_X} + \mathcal{L}_{similarity_Y} \right) + (1 - r) * \left(\mathcal{L}_{feature} + \mathcal{L}_{adv} \right) \quad (16)$$

Discriminator Loss. The loss of the discriminator network is calculated between the *D_Real* blocks of the real data and the *D_Fake* blocks of the fake data we want to translate. This process is illustrated in Eq. 17. The result is obtained by passing the averaged data block through the Relu activation function.

$$\mathcal{L}_D = relu\big(mean(1 - D\_Real)\big) + relu\big(mean(1 + D\_Fake)\big) \quad (17)$$

2.4. Datasets

2.4.1. Denim2Mustache

The Denim2Mustache dataset contains 950 image pairs (denim-mustache), 900 of which are training and 50 are tests [30]. The dataset includes front and back photos of different denim products, such as trousers, skirts, and shorts, and the mustache patterns on these denim images. Each image has a size of 256 × 256 × 3. Sample images of this dataset are shown in Figure 6(a).

2.4.2. Cityscapes

The Cityscapes dataset contains various stereo video images from the streets of 50 different cities [31]. It contains semantic segmentation maps belonging to 30 different classes, such as a motor-vehicle, road, building, and human. In this study, 2975 image pairs were used in the training phase and 500 in the testing phase. Sample images of this dataset are shown in Figure 6(b).

2.4.3. Maps

The Maps dataset comprises satellite images around New York and corresponding Google map images [13]. In these satellite images, there are objects and regions such as buildings, parks, roads. In this study, 2095 image pairs were used in the training phase and 98 in the testing phase. Sample images of this dataset are shown in Figure 6(c).

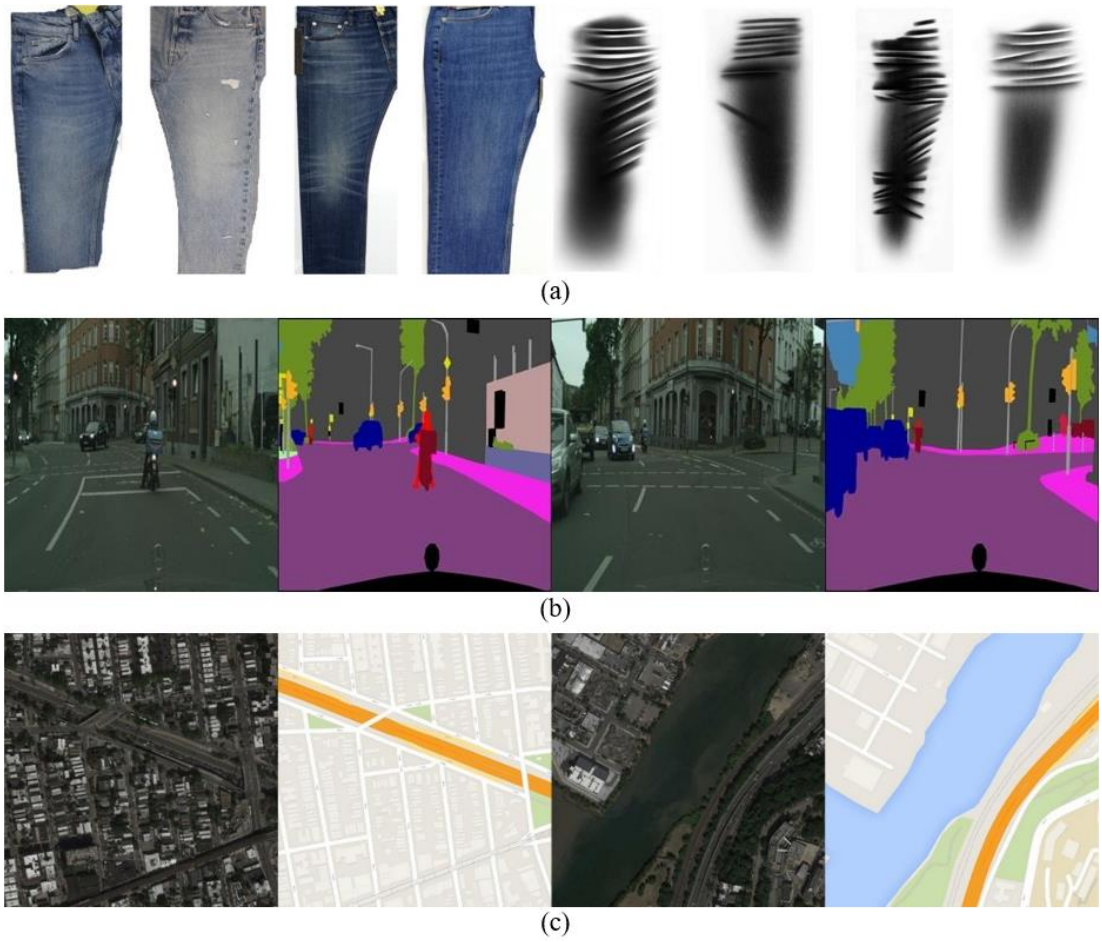

Figure 6. Example images from the Denim2Mustache (a), Cityscapes (b), and Maps (c) datasets.

JSR A
Journal of Scientific Reports

3. RESULTS

One of the most up-to-date architectures in image synthesis is DiscoGAN. In this section, the performances of the proposed WY-Net architecture and the DiscoGAN architecture in image synthesis are presented comparatively. Table 1. shows the structures and features of the architectures comprehensively.

Table 1. Quantitative property comparison of proposed methods and DiscoGAN.

Network Type	**Property Name**	**Model Name**		
		DiscoGAN	**dY-Net**	**uY-Net**
Generator	Number of networks	2	1	1
	Trainable Params	11,022,336	8,266,752	89,283,398
	Forward/backward pass size (MB)	164.00	127.00	135.96
	Params size (MB)	42.04	31.54	340.59
	Estimated total size (MB)	207.54	159.29	477.30
Discriminator	Number of networks	2	1	1
	Trainable Params	5,527,552	2,766,848	2,766,848
	Forward/backward pass size (MB)	74.00	37.00	37.00
	Params size (MB)	21.08	10.55	10.55
	Estimated total size (MB)	96.60	47.56	47.56

In the second stage, training and testing processes were carried out on the Cityscapes, Maps, and Denim2Mustache datasets to compare the architectures. The training and testing processes of the architectures were carried out with the PyTorch deep learning library on a server with an RTX 2080 graphics card, by taking the batch size of 1 for 150 epochs. Adjustments and arrangements such as hyperparameter, epoch, image dimensions have been made on the DiscoGAN architecture. Thus, the necessary environment has been prepared for the performance comparison of these models under appropriate conditions.

Image synthesis quality and the temporal cost of the training process were used as performance criteria. Three different similarity measures (SSIM, PSNR, and MSE) were used to control the synthesis quality of the image pairs in the test set. The performances of the models on the image pairs in the test datasets are shown in Table 2. According to this table, it is seen that dY-Net architecture produces very similar visuals in terms of image quality, although it is approximately 25% more efficient in the generator network structure and 50% more efficient in the discriminator network compared to DiscoGAN. However, it is seen that the uY-Net architecture, which focuses on image quality, produces more successful results in image synthesis than DiscoGAN. The sample outputs of these architectures in the test datasets are shown in Figures. 7 to 9.

Table 2. Performance comparison of proposed methods and DiscoGAN results. This performance comparison was made on the Denim2Mustache, Cityscapes and Maps datasets.

Dataset Name	**Model Name**	**Metric Name**	**Cross-domain image translation direction**	**150 epochs training**

			X → X	**X → Y**	**Y → Y**	**Y → X**	**time (minute)**
Cityscapes	DiscoGAN	SSIM ↑	0.451	0.384	0.676	0.676	715
		PSNR ↑	18.85	15.73	19.92	18.22	
		MSE ↓	943.96	2047.1	716.25	1108.1	
	dY-Net	SSIM ↑	0.835	0.397	0.871	0.688	**475**
		PSNR ↑	27.04	16.32	28.01	18.88	
		MSE ↓	143.61	1781.1	109.40	955.9	
	uY-Net	SSIM ↑	0.945	0.437	0.944	0.696	815
		PSNR ↑	32.45	16.43	35.13	18.50	
		MSE ↓	43.32	1744.7	22.21	1051.9	
Maps	DiscoGAN	SSIM ↑	0.268	0.245	0.589	0.516	495
		PSNR ↑	18.02	15.80	28.15	26.48	
		MSE ↓	1093.3	1809.2	129.76	205.5	
	dY-Net	SSIM ↑	0.742	0.255	0.739	0.515	**335**
		PSNR ↑	23.44	16.14	33.25	26.93	
		MSE ↓	320.57	1691.4	34.89	191.1	
	uY-Net	SSIM ↑	0.968	0.252	0.763	0.541	635
		PSNR ↑	28.73	16.13	33.64	26.87	
		MSE ↓	89.60	1676.7	31.63	204.4	
Denim2Mustache	DiscoGAN	SSIM ↑	0.712	0.673	0.922	0.849	210
		PSNR ↑	19.05	19.06	25.26	19.44	
		MSE ↓	956.1	1923.9	235.09	1049.9	
	dY-Net	SSIM ↑	0.814	0.688	0.970	0.819	**145**
		PSNR ↑	28.30	16.53	32.42	19.18	
		MSE ↓	143.8	1735.6	41.73	1166.8	
	uY-Net	SSIM ↑	0.977	0.739	0.983	0.869	265
		PSNR ↑	36.85	16.69	38.08	20.96	
		MSE ↓	27.46	1725.2	12.92	777.9	

JSR A
Journal of Scientific Reports

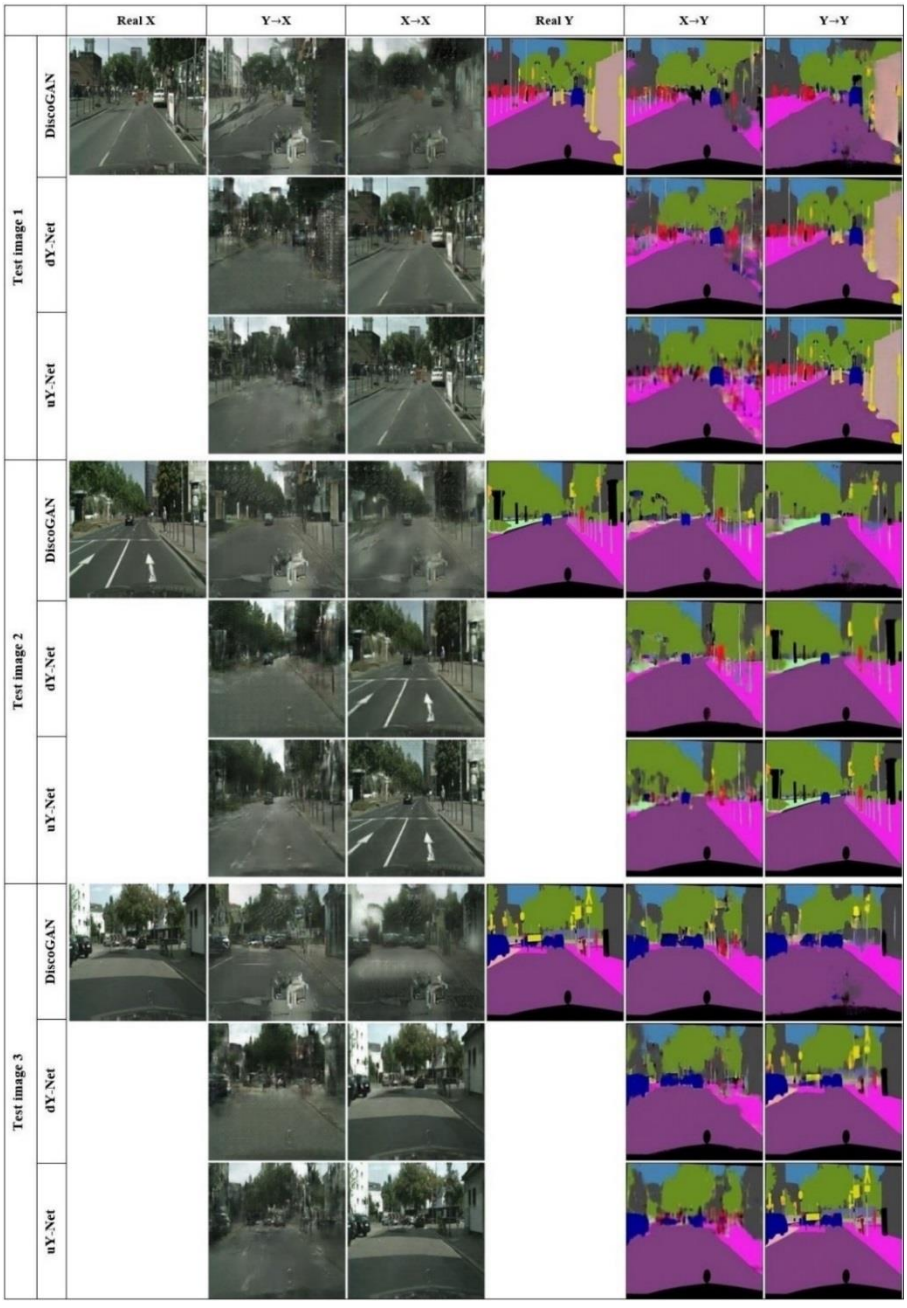

Figure 7. Example results of proposed methods and DiscoGAN on image synthesis cityscapes↔segmentation maps, compared to ground truth. Left to right: real *X*, predicted *X*, reconstructed *X*, real *Y*, predicted *Y*, reconstructed *Y* images. Top to bottom for three image pairs: The DiscoGAN, dY-Net, uY-Net architectures.

JSR A
Journal of Scientific Reports

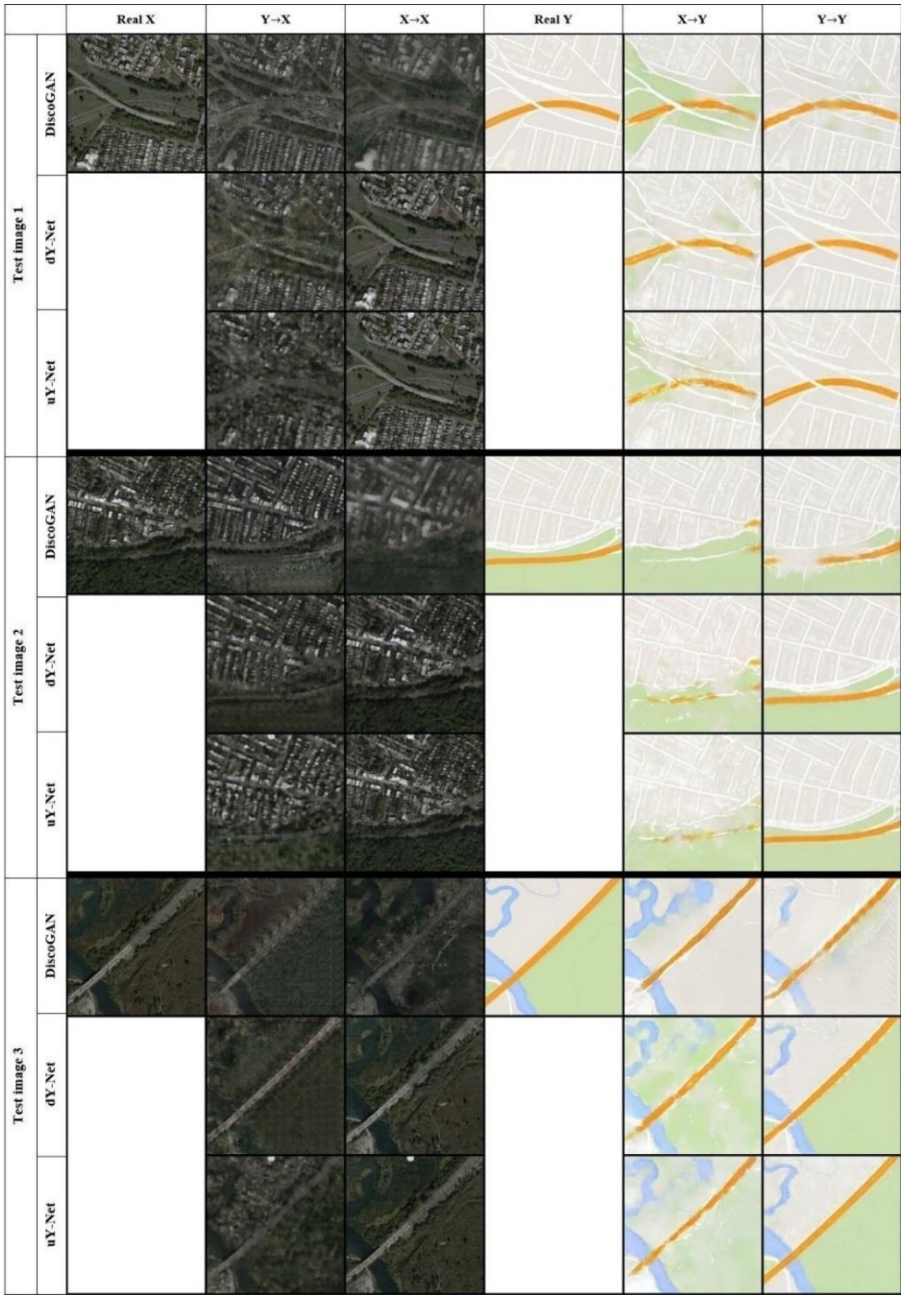

Figure 8. Example results of proposed methods and DiscoGAN on image synthesis satellite images↔Google maps, compared to ground truth. Left to right: real *X*, predicted *X*, reconstructed *X*, real *Y*, predicted *Y*, reconstructed *Y* images. Top to bottom for three image pairs: The DiscoGAN, dY-Net, uY-Net architectures.

JSR A
Journal of Scientific Reports

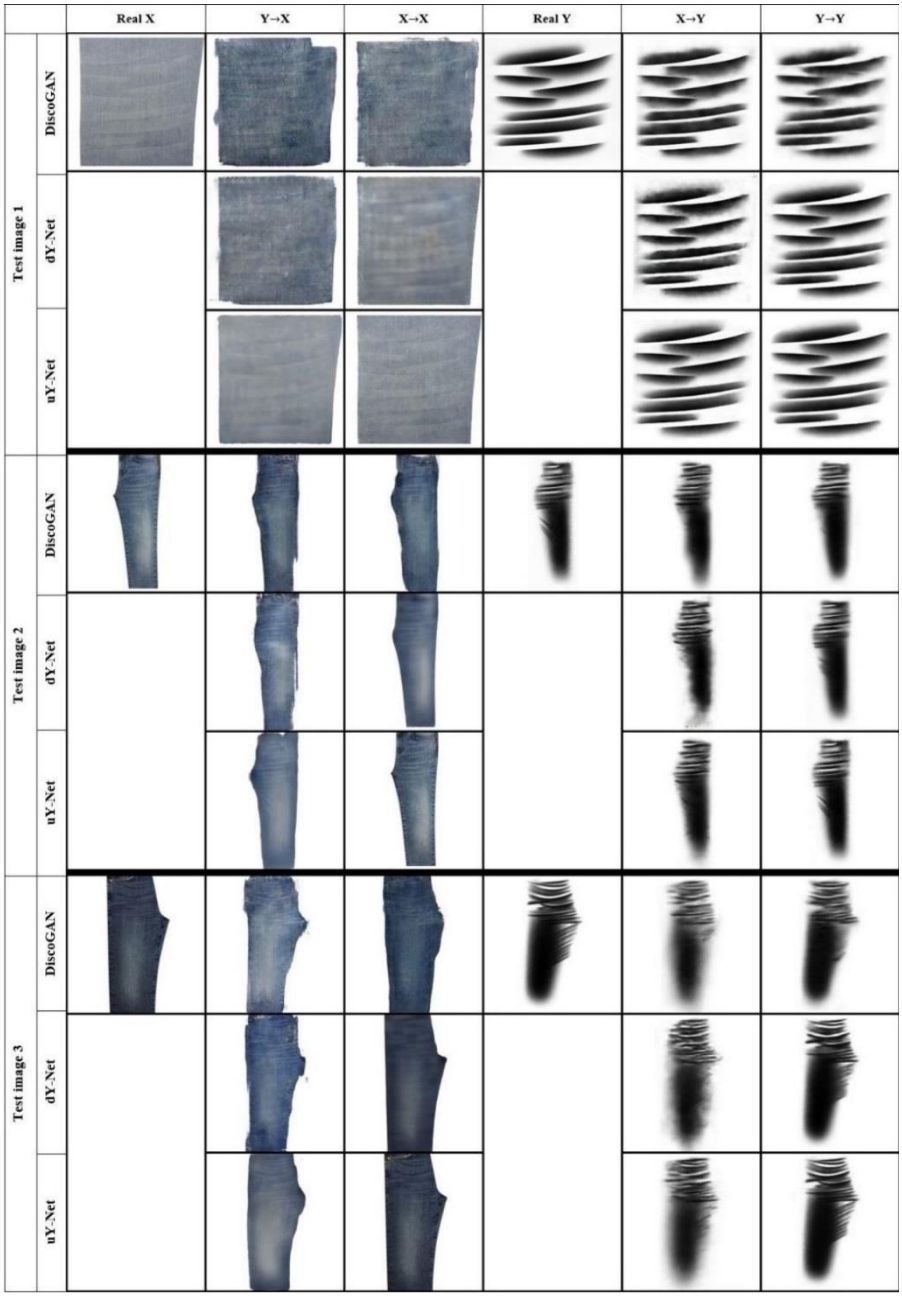

Figure 9. Example results of proposed methods and DiscoGAN on image synthesis denim images↔mustache patterns, compared to ground truth. Left to right: real *X*, predicted *X*, reconstructed *X*, real *Y*, predicted *Y*, reconstructed *Y* images. Top to bottom for three image pairs: The DiscoGAN, dY-Net, uY-Net architectures.

4. DISCUSSION and CONCLUSIONS

In this study, two different architectures, dY-Net, and uY-Net are proposed to be used in the field of image synthesis. The dY-Net architecture has low parameter space and prioritizes performance at image synthesis speed. The uY-Net architecture includes embedding the U-Net structure in WY-Net and prioritizes image synthesis quality. Comparisons of the proposed architectures with existing counterparts are in terms of image synthesis time and accuracy. The basis of WY-Net architectures is the principle of synchronizing the feature spaces of the input and output images. Thus, there is a common use of parameters, and a serious reduction in the number of hyperparameters is realized. Comparisons were made with the DiscoGAN architecture to find out the position of the image synthesizing and speed performances of the proposed architectures in the literature. DiscoGAN architecture, with its bidirectional training structure, is widely used in the field of image synthesis. According to the obtained comparison results, although the number of parameters of dY-Net architecture is 34% lower than DiscoGAN, it can synthesize images of similar quality. It has been observed that uY-Net architecture synthesizes higher quality images than other architectures. Three different datasets (Maps, Cityscapes, and Denim2Mustache) were used in the validation process. These results reveal that two different architectures can be used for applications that prioritize performance or translation quality in image synthesis studies. The next study will be about transferring the feature space synchronization principle used in WY-Net architecture to SSL techniques.

ACKNOWLEDGMENT

This study was funded by the Baykan Denim A.Ş. and the Scientific Research Projects Department of Inonu University with the project number “FKP-2021-2144”. We would like to thank Baykan Denim A.Ş. and Inonu University.

REFERENCES

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S. and Bengio, Y., (2014), Generative adversarial nets. Advances in neural information processing systems, 27.

[2] Radford, A., Metz, L. and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

[3] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A. and Shi, W. (2016), Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4681-4690).

[4] Özdemir, D., and Arslan, N. N., (2022) Analysis of Deep Transfer Learning Methods for Early Diagnosis of the Covid-19 Disease with Chest X-ray Images. Düzce Üniversitesi Bilim ve Teknoloji Dergisi, 10(2), 628-640.

[5] Karras, T., Aila, T., Laine, S. and Lehtinen, J. (2017), Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196.

[6] Zhang, H., Goodfellow, I., Metaxas, D. and Odena, A. (2018), Self-attention generative adversarial networks. In International conference on machine learning (pp. 7354-7363). PMLR.

[7] Karras, T., Laine, S. and Aila, T. (2018), A style-based generator architecture for generative adversarial networks. arXiv preprint arXiv:1812.04948.

[8] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J. and Aila, T. (2019), Analyzing and improving the image quality of stylegan. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 8110-8119).

[9] Karras, T., Aittala, M., Laine, S., Härkönen, E., Hellsten, J., Lehtinen, J. and Aila, T. (2021), Alias-free generative adversarial networks. arXiv preprint arXiv:2106.12423.

[10] Brock, A., Donahue, J. and Simonyan, K. (2018), Large scale GAN training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096.

[11] Donahue, J. and Simonyan, K. (2019), Large scale adversarial representation learning. arXiv preprint arXiv:1907.02544.

[12] B. Li, Y. Zhu, Y. Wang, C. -W. Lin, B. Ghanem and L. Shen. (2021), AniGAN: Style-Guided Generative Adversarial Networks for Unsupervised Anime Face Generation," in IEEE Transactions on Multimedia, doi: 10.1109/TMM.2021.3113786.

[13] Isola, P., Zhu, J. Y., Zhou, T. and Efros, A. A. (2017), Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1125-1134).

[14] Wang, T. C., Liu, M. Y., Zhu, J. Y., Tao, A., Kautz, J. and Catanzaro, B. (2018), High-resolution image synthesis and semantic manipulation with conditional gans. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 8798-8807).

[15] Park, T., Liu, M. Y., Wang, T. C. and Zhu, J. Y. (2019), Semantic image synthesis with spatially-adaptive normalization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2337-2346).

[16] Zhu, J. Y., Park, T., Isola, P. and Efros, A. A. (2017), Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).

[17] Li, Y., Singh, K. K., Ojha, U. and Lee, Y. J. (2020), Mixnmatch: Multifactor disentanglement and encoding for conditional image generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 8039-8048).

[18] Dundar, A., Sapra, K., Liu, G., Tao, A. and Catanzaro, B. (2020), Panoptic-based image synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 8070-8079).

[19] Wang, X., Li, Y., Zhang, H. and Shan, Y. (2021), Towards Real-World Blind Face Restoration with Generative Facial Prior. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 9168-9178).

[20] Wang, X., Xie, L., Dong, C. and Shan, Y. (2021), Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 1905-1914).

[21] Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Silvestrov, A. and Lempitsky, V. (2021), Resolution-robust Large Mask Inpainting with Fourier Convolutions. arXiv preprint arXiv:2109.07161.

[22] Kim, T., Cha, M., Kim, H., Lee, J. K., and Kim, J. (2017, July), Learning to discover cross-domain relations with generative adversarial networks. In International Conference on Machine Learning (pp. 1857-1865). PMLR.

[23] Nilsson, J. and Akenine-Möller, T. (2020), Understanding ssim. arXiv preprint arXiv:2006.13846.

[24] Mihelich, M., Dognin, C., Shu, Y., and Blot, M. (2020), A Characterization of Mean Squared Error for Estimator with Bagging. In International Conference on Artificial Intelligence and Statistics (pp. 288-297). PMLR.

[25] Willmott, C. J. and Matsuura, K. (2005), Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate research, 30(1), 79-82.

[26] Fardo, F. A., Conforto, V. H., de Oliveira, F. C. and Rodrigues, P. S. (2016). A formal evaluation of PSNR as quality measurement parameter for image segmentation algorithms. arXiv preprint arXiv:1605.07116.

[27] Bailer, C., Varanasi, K., and Stricker, D. (2017), CNN-based patch matching for optical flow with thresholded hinge embedding loss. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3250-3259).

[28] Ronneberger, O., Fischer, P., & Brox, T. (2015, October), U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

[29] Lim, J. H., and Ye, J. C. (2017), Geometric gan. arXiv preprint arXiv:1705.02894.

[30] Şahin, E., TALU, M. F., (2021), Bıyık Deseni Üretiminde Çekişmeli Üretici Ağların Performans Karşılaştırması. Bitlis Eren Üniversitesi Fen Bilimleri Dergisi, 10(4), 1575-1589. Doi: 10.17798/bitlisfen.98586.

[31] Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R. and Schiele, B. (2016), The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3213-3223).