Synthetic Retinal Images from Unconditional GANs

Sangeeta Biswas, Johan Rohdin and Martin Drahanský

***Abstract*— Synthesized retinal images are highly demanded in the development of automated eye applications since they can make machine learning algorithms more robust by increasing the size and heterogeneity of the training database. Recently, conditional Generative Adversarial Networks (cGANs) based synthesizers have been shown to be promising for generating retinal images. However, cGANs based synthesizers require segmented blood vessels (BV) along with RGB retinal images during training. The amount of such data (i.e., retinal images and their corresponding BV) available in public databases is very small. Therefore, for training cGANs, an extra system is necessary either for synthesizing BV or for segmenting BV from retinal images. In this paper, we show that by using unconditional GANs (uGANs) we can generate synthesized retinal images without using BV images.**

I. INTRODUCTION

Computer-aided diagnosis systems (CADS) for retinopathy are largely demanded by eye specialists. Advanced machine learning algorithms used for developing CADS are data hungry. However, available retinal image databases are small since the acquisition and labeling processes of retinal images are expert-dependent, tedious, time-consuming and costly. Besides, retinal image databases may suffer from severe class imbalance due to the rare nature of some pathology (e.g., venous omega loops in diabetic retinopathy). By synthesizing retinal images we can increase the size and heterogeneity of the training database which ultimately helps machine learning algorithms be more robust for developing CADS for retinopathy as well as for research. Now these days, Generative Adversarial Networks (GANs) [1] are used in almost all areas of image [2], [3], [4], speech [5], [6], [7], [8] or text processing [9], [10]. Even though GANs have been used to deal with many well known and challenging tasks such as segmentation, reconstruction, detection, denoising or classification, they are mainly used for synthesis. In [11], it has been reported that in medical image processing, the majority of GANs based works (around 40%) are for synthesizing medical images. Originally, GANs were proposed as an entirely unsupervised generative framework [1]. In that work, only noise vectors sampled from a known distribution were used to generate 'fake-but-realistic' data points. Later unsupervised framework was turned into a supervised generative framework by using *extra information* (e.g., class labels) along with noise vectors. For clarity, the former GANs are called *unconditional* or *unsupervised* GANs (uGANs) [12], [11], whereas the later GANs are named *conditional* GANs (cGANs) [13]. As medical image synthesizers, cGANs have been applied more than uGANs [11]. The few works [14], [15], [16], [17] found in the literature for GANs based retinal image synthesizer are all based on cGANs. In all those works, blood vessel (BV) trees were used as the extra information. How to obtain BV trees made those works different. In [14], ground truth of BV trees were used, whereas in [15] synthesized BV trees generated by additional GANs were used. In [16], a U-Net [18] was used to get BV trees from RGB retinal images, whereas in [17] an adversarial autoencoder was used to synthesize BV trees. In spite of having potentiality, uGANs based retinal image synthesizers have not been explored properly in past. In this paper, we show that it is possible to generate realistic retinal images by uGANs as long as the hyperparameters that control the balance between its two competing training objectives are carefully tuned.

This work has received funding from the Ministry of Education, Youth and Sports, Czech Republic (Project No: 0008371 - International mobility of researchers at the Brno University of Technology), and the European Union's Horizon 2020 Research and Innovation Programme under the Marie Sklodowska-Curie and it is co-financed by the South Moravian Region under grant agreement No. 665860.

This work was supported by the Ministry of Education, Youth and Sports, Czech Republic, from the Large Infrastructures for Research, Experimental Development and Innovations project "IT4Innovations National supercomputing Center - LM2015070"

The authors are with the Faculty of Information Technology, Brno University of Technology, Brno, 61200, Czech Republic {biswas, rohdin, drahan}@fit.vutbr.cz

II. GANs BASED SYNTHESIZER

GANs based retinal image synthesizer consists of two neural networks: a generator (**G**) and a discriminator (**D**). The target of **G** is to generate RGB retinal images, $\hat{\mathbf{X}} \in \mathbb{R}^{h \times w \times 3}$, which look as realistic as the images in the training database, $\mathbf{X} \in \mathbb{R}^{h \times w \times 3}$. Here, h and w denote height and width of retinal images.

Contrary to **G**, the target of **D** is to decide what the probability is that retinal images are from $\mathbf{X}$ or from $\hat{\mathbf{X}}$. During training, the parameters of **D** and **G** are updated in an iterative process so that in each iteration both **G** and **D** reach closer to their targets. After some iterations, they are expected to reach a Nash Equilibrium point where none of them can improve their performance anymore, i.e. Eq. 1 holds.

$$p_{\hat{\mathbf{x}}} = p_{\mathbf{x}}, \tag{1a}$$

$$\mathbf{D}(x \in \mathbf{X}) = \mathbf{D}(\hat{x} \in \hat{\mathbf{X}}) = \frac{1}{2}, \tag{1b}$$

where $p_{\hat{\mathbf{x}}}$ and $p_{\mathbf{x}}$ are the probability distributions of $\hat{\mathbf{X}}$ and $\mathbf{X}$, respectively. For uGANs based synthesizer, the targets of **G** and **D** can be formulated as Eq. 2 and Eq. 3, respectively:

$$\mathbf{G} : z \in \mathbb{R}^d \mapsto \hat{x} \in \hat{\mathbf{X}}, \text{ so that} \tag{2a}$$

$$p_{\hat{\mathbf{x}}} = p_{\mathbf{x}}, \tag{2b}$$

$$\mathbf{D} : x \in \mathbf{X} \mapsto 1, \text{ and,} \tag{3a}$$

$$\hat{x} \in \hat{\mathbf{X}} \mapsto 0, \tag{3b}$$

where z is a noise vector sampled from any known distribution p_z [e.g., $p_z \sim \mathcal{N}(0, I)$, or $p_z \sim \mathcal{U}(-1, 1)$]; d is the dimension of z. For cGANs based synthesizer, the targets of **G** and **D** can be formulated as Eq.4 and Eq.5, respectively:

$$\mathbf{G} : (z, c) \mapsto \hat{x} \in \hat{\mathbf{X}}, \text{ so that} \tag{4a}$$

$$p_{\hat{\mathbf{x}}} = p_{\mathbf{x}}, \tag{4b}$$

$$\mathbf{D} : (x, c) \mapsto 1, \text{ and,} \tag{5a}$$

$$(\hat{x}, c) \mapsto 0, \tag{5b}$$

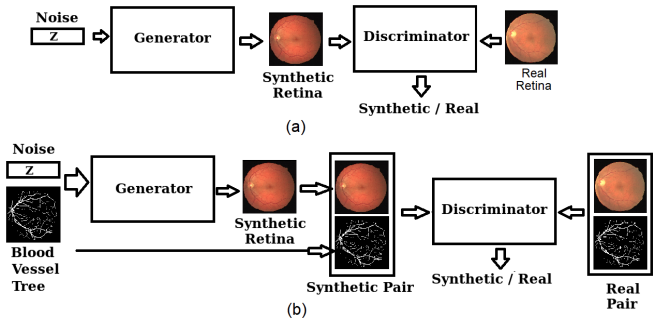

Fig. 1. (a) Unconditional GANs based retinal image synthesizer and (b) conditional GANs based retinal image synthesizer.

where c is some kind of auxiliary information. During training, the parameters of **D** and **G** are updated iteratively targeting to find $\mathbf{D}^*$ and $\mathbf{G}^*$ given by Eq. 6 and Eq. 7 for uGANs and cGANs, respectively:

$$\mathbf{G}^*, \mathbf{D}^* = \arg\min_{\mathbf{G}} \max_{\mathbf{D}} \left(\mathbb{E}_{x \sim p_{\mathbf{x}}} [\log \mathbf{D}(x)] + \mathbb{E}_{z \sim p_z} [\log(1 - \mathbf{D}(\mathbf{G}(z)))] \right). \tag{6}$$

$$\mathbf{G}^*, \mathbf{D}^* = \arg\min_{\mathbf{G}} \max_{\mathbf{D}} \left(\mathbb{E}_{x \sim p_{\mathbf{x}}, c \sim p_c} [\log \mathbf{D}(x, c)] + \mathbb{E}_{z \sim p_z, c \sim p_c} [\log(1 - \mathbf{D}(c, \mathbf{G}(z, c)))] \right). \tag{7}$$

In previous works (e.g., [14], [15], [16], [17]) gray scaled BV trees were used as c (i.e., $c \in \mathbb{R}^{h \times w \times 1}$). That means, in those works **G** used both noise vectors and BV trees for generating retinal images and **D** discriminated between the pairs of synthetic retinal image and BV trees from the pairs of real retinal image and BV trees (see Fig. 1 (b)). Contrary to cGANs based synthesizer, **G** in uGANs based synthesizer generates retinal images only from noise vectors z and **D** discriminates only between synthetic and real retinal images (see Fig. 1 (a)). Although, uGANs based synthesizer requires a simpler data processing step than the cGANs based synthesizer, in the literature we have not found any uGANs based retinal image synthesizer.

III. BALANCE BETWEEN GENERATOR AND DISCRIMINATOR

By following the gradient-based training approach given in [1], theoretically, we can reach the Nash Equilibrium point. In the gradient-based approach [1], (see Algorithm 1 in Table I), first **D** is trained for k times after which **G** is trained once. This training process continues until it convergences. In this approach there is only one hyperparameter, i.e., k to control the number of updates of **D**, and no hyperparameter to control the number of updates of **G**. Variations of this approach are also seen in the literature where one hyperparameter, say r is used to control the number of updates of **G** instead of **D**. In [1] for generating images of digits, faces, animals, or vehicles, $k = 1$ was chosen because it was the *least expensive option*. For some systems such as cGANs based retinal image synthesizer in [14], updating **G** twice while keeping $k = 1$ for each iteration was better. In our experiments, neither of these settings worked well. With $k = 1$, the generated images lack complex structures of BV trees, the macula or the optic disk (see 1st row of Fig. 3). On the other hand, with $r = 2$ while keeping $k = 1$ the generated images were noisy (as shown in the 1st row of Fig. 6). Therefore, we have decided to keep both hyperparameters

TABLE I
ALGORITHMS OF UGANS

Algorithm 1

for n iterations **do**
- For k times
  - Prepare a mini-batch of retinal images $\{(x, \hat{x})_{i=1}^{m}\}^{kn}$ where m is the mini-batch size.
  - Update **D** using mini-batch $\{(x, \hat{x})_{i=1}^{m}\}^{kn}$.
- Update **G** once using a mini-batch of noise vectors, $\{(z)_{i=1}^{m}\}^{n}$.

Algorithm 2

for n iterations **do**
- For k times
  - Prepare $\{(x, \hat{x})_{i=1}^{m}\}^{kn}$.
  - Update **D** using $\{(x, \hat{x})_{i=1}^{m}\}^{kn}$.
- For r times
  - Prepare $\{(z)_{i=1}^{m}\}^{rn}$.
  - Update **G** using $\{(z)_{i=1}^{m}\}^{rn}$.

for training uGANs based retinal image synthesizer as shown in Algorithm 2 in Table I.

Finding the appropriate values of k and r is crucial for the performance of GANs based system since they help to balance **D** and **G** during training. If they are not properly tuned then **G** will end up generating noisy retinal images without complex structures. Both too small and too large values of k can prevent **D** from giving proper feedback to **G**. If k is too small, **D** may not learn distinguishable features between real and fake images and therefore the feedback given to **G** will be quite random. If k is too large comparing to r, **D** may become so strong that (small) changes in the fake images generated by **G** will not affect the predictions of **D** significantly and therefore the feedback given to **G** will be too weak.

IV. EXPERIMENTS

A. Setup

All implementations were done using TensorFlow's Keras API and Python. A cluster machine with 23 nodes having two Intel Sandy Bridge E5-2470, 8-core, 2.3GHz processors, 96 GB of physical memory, and NVIDIA Tesla Kepler K20m having 4.63 GB Memory per node, was used. In total 1200 images, from the public database MESSIDOR [19] were used. These images were acquired by 3 ophthalmologic departments using a color video 3CCD camera on a Topcon TRC NW6 non-mydriatic retinograph with a 45 degree field of view. The images were captured using 8 bits per color plane. Among those 1200 images 588 images were 960×1440, 400 images were 1488×2240 and 212 images were 1536×2304. In our experiments, all images were re-sized to the same size (i.e., 256×256) by bicubic interpolation. After re-sizing, rectangle images become square and retina become oval shaped. Therefore, synthesized retina were also oval-shaped. In order to get round shaped retina, either rectangle images need to be transferred to square images by cropping dark background before re-sizing at the pre-processing stage, or square shaped synthesized images need to be transferred to rectangle images by re-sizing as a post-processing task. Later approach was chosen here.

The RGB values of the re-sized images were re-scaled to the range of the *tanh* activation function $[-1, 1]$. Except that, no other pre-processing was applied to the training images. As shown in Fig. 2, there were images contained healthy retina as well as retina with pathology (e.g., the bottom-right retina). The mini-batch size was set to 32 (i.e., $m = 32$). Noise vectors were drawn from the uniform distribution. A deep convolutional neural

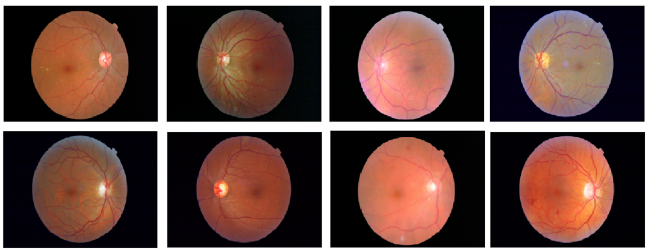

Fig. 2. Some sample images from the training set [Res. 256×256].

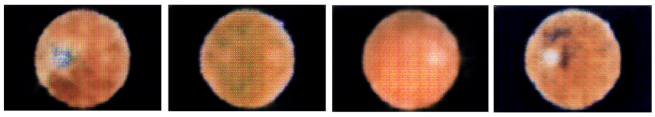

Fig. 3. Synthesized images when k and r had the equal values less than or equal to 4. 1st column: $k = 1, r = 1$, 2nd column: $k = 2, r = 2$, 3rd column: $k = 3, r = 3$, 4th column: $k = 4, r = 4$. [Res. 256×256, $n = 10000$, $d = 512$.]

network based architecture suggested in [20] was followed with minor modifications. Table II shows the model architecture for 256×256 sized images. As loss function, binary cross-entropy was used as shown in Eq 7. As optimizer, RMSProp with a learning rate of 0.0001 and a decay of $3e{-}8$ was used. The dropout value was set to 0.5. For batch normalization, momentum was set to 0.5 instead of default value 0.99. For LeakyReLU, α was set to 0.2 instead of default value 0.3. For all convolutional and transposed convolutional layers, $stride = 2$, $kernel\_size = 5$, and $padding = same$ was used. $l2$ regularization was applied only for weights and biases of the transposed convolutional layers. For all other settings, the default values of TensorFlow's Keras API were used.

In order to obtain clear BV trees in reasonable training time, images with 256×256 resolution, and $n = 10000$ were chosen for all experiments. Although the majority of previous works used $d = 100$, in our initial experiments, better retinal images were achieved when $d = 512$. Therefore, for all later experiments $d = 512$ was used.

B. Results

As shown in Fig. 3, when k and r had the same values until 4, the synthetic images did not have any complex structures of BV trees, the macula or the optic disk. However, the models trained by using the same values for k and r above 4 generated the complex structures of retina (see Fig. 4). With larger values of k while keeping $r = 1$, the quality of the synthetic images became better as shown in Fig. 5. On the other hand, with larger values of r while keeping $k = 1$, the synthetic images were very noisy as shown in Fig. 6.

C. Evaluation

To evaluate the GANs based synthesizer, 1000 images were generated and Structural Similarity (SSIM) [21] was estimated for all synthetic image pairs, all real image pairs as well as all combinations of synthetic and real images. The range of SSIM is $[-1, 1]$. The higher the SSIM value, the more similar is the image pair. Table III summarizes the results. In the first row, the mean SSIM of all image pairs is shown. The synthetic images are slightly more similar (SSIM mean 0.66) to each other than the real images (SSIM mean 0.63). The similarity between synthetic and real images is slightly smaller (SSIM mean 0.61). These numbers

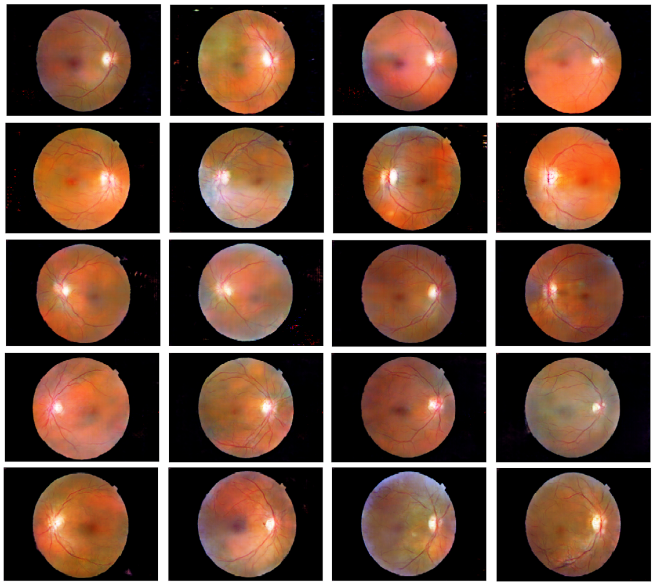

Fig. 4. Synthesized images when k and r had the equal values above 4. 1st row: $k = 5, r = 5$, 2nd row: $k = 6, r = 6$, 3rd row: $k = 7, r = 7$, 4th row: $k = 8, r = 8$, 5th row: $k = 12, r = 12$. [Res. 256×256, $n = 10000$, $d = 512$.]

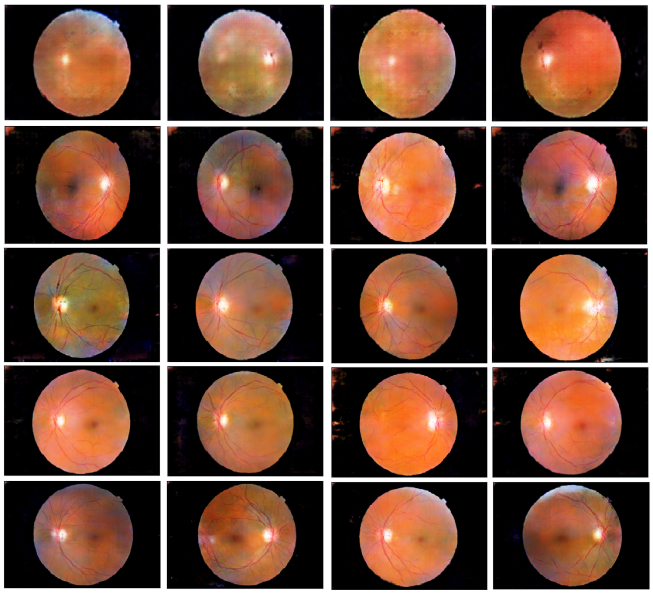

Fig. 5. Synthesized images when k gets higher while $r = 1$. 1st row: $k = 2, r = 1$, 2nd row: $k = 3, r = 1$, 3rd row: $k = 4, r = 1$, 4th row: $k = 5, r = 1$, 5th row: $k = 6, r = 1$. [Res. 256×256, $n = 10000$, $d = 512$.]

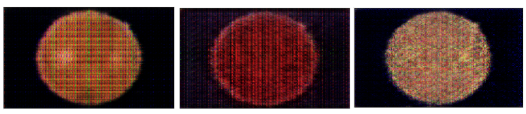

Fig. 6. Synthesized images when r gets higher while $k = 1$. 1st column: $k = 1, r = 2$, 2nd column: $k = 1, r = 3$, 3rd column: $k = 1, r = 4$. [Res. 256×256.]

suggests that the method works well, however it is possible to obtain these results (or even better) with a trivial synthesizer that simply

TABLE II
MODEL ARCHITECTURE FOR 256×256.

D		**G**	
Layer	Output Shape	Layer	Output Shape
Input	(256, 256, 3)	Input	(d)
Conv2D + LeakyReLU + Dropout	(128, 128, 32)	Dense + Reshape	(4, 4, 1024)
Conv2D + LeakyReLU + Dropout	(64, 64, 64)	Conv2DTranspose + BatchNorm + ReLU	(8, 8, 512)
Conv2D + LeakyReLU + Dropout	(32, 32, 128)	Conv2DTranspose + BatchNorm + ReLU	(16, 16, 256)
Conv2D + LeakyReLU + Dropout	(16, 16, 256)	Conv2DTranspose + BatchNorm + ReLU	(32, 32, 128)
Conv2D + LeakyReLU + Dropout	(8, 8, 512)	Conv2DTranspose + BatchNorm + ReLU	(64, 64, 64)
Conv2D + LeakyReLU + Dropout	(4, 4, 1024)	Conv2DTranspose + BatchNorm + ReLU	(128, 128, 32)
Dense + Sigmoid	(1)	Conv2DTranspose + BatchNorm + Tanh	(256, 256, 3)

TABLE III
STATISTICS FOR SSIM (S: SYNTHETIC, R: REAL)

SSIM statistic	S-S	R-R	S-R
Mean SSIM:	0.66	0.63	0.61
Mean of Maximum SSIM:	0.86	0.82	0.73
Maximum SSIM:	0.94	0.88	0.81

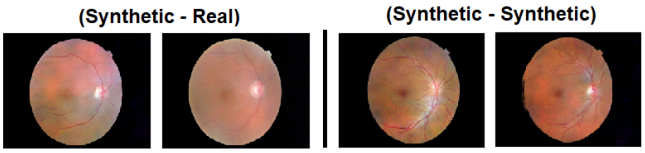

Fig. 7. Image pair having the maximum SSIM when $k = 8$ and $r = 8$. [Res. 256×256, $n = 10000$, $d = 512$]

picks images randomly from the training set. If this is the case, the maximum SSIM of each synthetic image when compared to the training images should be close to one. The second row shows the mean for the maximum SSIM. Clearly, the above mentioned problem did not occur. Finally, the third row shows the maximum SSIM value obtained for any image pair. Fig. 7 shows the pair of images having the maximum SSIM when synthetic images were compared to either the other synthetic or the real images. In conclusion, **G** did not simply memorize training images, rather it showed its creativity in the generated images by using information learned from the training data.

V. CONCLUSIONS

In previous GANs based synthesizers, blood vessel trees were used along with noise vectors to generate retinal images. The results in this paper show that without any extra information (such as blood vessel trees) it is also possible to generate retinal images by GANs. It is also shown how important it is to keep a balance between the two competitors (i.e., generator and discriminator) during training. If this balance is not kept, the generator may end up generating only blurry retina images without high-level structures such as blood vessel tree, optic disc, macula, etc. Generating high resolution retinal images from only noise vectors without having to tune the balancing hyperparameters is our future challenge.

REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *NIPS*, 2014, pp. 2672–2680.

[2] E. L. Denton, S. Chintala, A. Szlam, and R. Fergus, "Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks," in *NIPS*, 2015, pp. 1486–1494.

[3] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," in *IEEE CVPR*, 2017, pp. 5967–5976.

[4] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," in *IEEE CVPR*, 2017, pp. 105–114.

[5] B. Bollepalli, L. Juvela, and P. Alkuo, "Generative adversarial network-based glottal waveform model for statistical parametric speech synthesis," in *INTERSPEECH*, 2017, pp. 3394–3398.

[6] A. Sriram, H. Jun, Y. Gaur, and S. Satheesh, "Robust Speech Recognition Using Generative Adversarial Networks," in *IEEE ICASSP*, 2018, pp. 5639–5643.

[7] J.-T. Chien and K.-T. Peng, "Adversarial Learning and Augmentation for Speaker Recognition," in *Odyssey*, 2018, pp. 342–348.

[8] K. Wang, J. Zhang, S. Sun, Y. Wang, F. Xiang, and L. Xie, "Investigating Generative Adversarial Networks based Speech Dereverberation for Robust Speech Recognition," in *INTERSPEECH*, 2018, pp. 1581–1585.

[9] S. E. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative Adversarial Text to Image Synthesis," in *ICML*, vol. 48, 2016, pp. 1060–1069.

[10] J. Li, W. Monroe, T. Shi, A. Ritter, and D. Jurafsky, "Adversarial Learning for Neural Dialogue Generation," in *EMNLP*, 2017, pp. 2157–2169.

[11] S. Kazeminia, C. Baur, A. Kuijper, B. van Ginneken, N. Navab, S. Albarqouni, and A. Mukhopadhyay, "GANs for Medical Image Analysis," 2018. [Online]. Available: http://arxiv.org/abs/1809.06222

[12] J. Gauthier, "Conditional generative adversarial nets for convolutional face generation," *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual recognition, Winter Semester*, 2015.

[13] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," 2014. [Online]. Available: http://arxiv.org/abs/1411.1784

[14] H. Zhao, H. Li, and L. Cheng, "Synthesizing Filamentary Structured Images with GANs," 2017. [Online]. Available: http://arxiv.org/abs/1706.02185

[15] J. T. Guibas, T. S. Virdi, and P. S. Li, "Synthetic Medical Images from Dual Generative Adversarial Networks," 2017. [Online]. Available: http://arxiv.org/abs/1709.01872

[16] P. Costa, A. Galdran, M. I. Meyer, M. D. Abràmoff, M. Niemeijer, A. M. Mendonça, and A. Campilho, "Towards Adversarial Retinal Image Synthesis," 2017. [Online]. Available: https://arxiv.org/abs/1701.08974

[17] P. Costa, A. Galdran, M. I. Meyer, M. D. Abrámoff, M. Niemeijer, A. M. Mendonça, and A. Campilho, "End-to-End Adversarial Retinal Image Synthesis," *IEEE Transactions on Medical Imaging*, vol. 37, no. 3, pp. 781 – 791, 2018.

[18] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional Networks for Biomedical Image Segmentation," in *MICCAI*, vol. 9351, 2015, pp. 234–241.

[19] E. Decencire, X. Zhang, G. Cazuguel, B. Lay, B. Cochener, C. Trone, P. Gain, R. Ordonez, P. Massin, A. Erginay, B. Charton, and J.-C. Klein, "Feedback on a publicly distributed database: the messidor database," *Image Analysis & Stereology*, vol. 33, no. 3, pp. 231–234, Aug. 2014.

[20] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," in *ICLR*, 2016.

[21] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, 2004.