GEAC: Generating and Evaluating Handwritten Arabic Characters Using Generative Adversarial Networks

Lamia Aljoudi, Tahani Alkhodidi, Aisha Fallatah, Aya Bashy, Nahed Ali, Norah Alqahtani, Nujood Almajnooni, Ahad Allhabi, Telawa Albarakati, and Tarik Alafif
Computer Science Department, Jamoum University College, Umm Al-Qura University, Jamoum, Saudi Arabia
{s434023394, s435002408, s434005473, s434023127, s434002057, s435009725, s436012782, s435001206, s43308778}@st.uqu.edu.sa
tkafif@uqu.edu.sa

***Abstract*—Generative Adversarial Network (GAN) has made a breakthrough and great success in many research areas in computer vision. Different GANs generate different outputs. In this research work, we apply different GANs to generate handwritten Arabic characters. A basic GAN, Vanilla GAN, Deep Convolutional GAN (DCGAN), Bidirectional GAN (BiGAN), and Wasserstein GAN (WGAN) are used. Then, the results of the generated images are evaluated using native-Arabic human and Fréchet Inception Distance (FID). The qualitative and quantitative results are provided for the images generation and evaluation. In experimental evaluation, WGAN achieves better results in FID with a value of 96.007. On the other hand, DCGAN achieves better results in native-Arabic human evaluation with a value of 35%.**

***Index Terms*—Handwritten Arabic number, image generation, evaluation, GAN, VanillaGAN, DCGAN, BiGAN, WGAN**

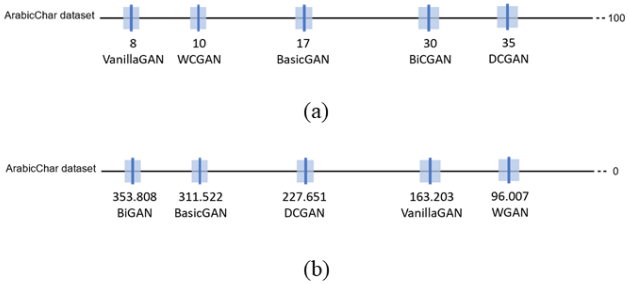

Fig. 1: Our evaluation metrics on handwritten ArabicChar dataset. (a) Native-Arabic human evaluation. (b) FID evaluation.

I. Introduction

Generative models such as Generative Adversarial Networks (GANs) have become very successful in image generation [5] in the past few years. The basic GAN consists of a game of two components, a generator (G) and a discriminator (D). The G generates fake images by learning the samples distribution using random variables to deceive the D. On the other hand, the D learns to classify the real and fake images. The G and D play a min-max game whereas D attempts to minimize the two errors for real and fake images while G attempts to maximize the fake images errors. It can be expressed as follows:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \quad (1)$$

where x is an image example, p is the data distribution, and z is a latent variable (noise).

Different types of GANs have been proposed in recent years such as Vanilla GAN [5], Deep Convolutional GAN (DCGAN) [8], Wasserstein GAN (WGAN) [2], and Bidirectional GAN (BiGAN) [3]. They share common GAN components, the generator and discriminator, but they differ in their architectures. Figure 2 shows our samples of different generated handwritten Arabic characters from different GANs.

In this work, our contributions are as follows:

- We introduce a small-scale handwritten Arabic characters dataset which contains connected and non-connected forms of handwritten Arabic characters.
- We apply different GANs to generate different generated handwritten Arabic characters.
- We evaluate the different generated handwritten Arabic characters using Fréchet Inception Distance (FID) and native-Arabic human evaluations. The evaluations are used to evaluate the quality of the generated images by estimating their realness and fakeness. Figure 1 shows our evaluation metrics on our handwritten Arabic characters dataset.

The rest of this paper is organized as follows. In **Section II**, related work is reviewed. In **Section III**, we describe our handwritten Arabic characters dataset. In **Section IV**, we present our approach for generating different handwritten Arabic characters. In **Section V**, our experimental results and evaluations are provided. A discussion is provided in **Section VI**. Finally, we conclude our work in **Section VII**.

II. Related Work and Datasets

An Arabic Handwritten Characters dataset was created in [4] for the purpose of handwritten Arabic characters recognition using Convolutional Neural Network. The dataset consists of 16,800 of non-connected and segmented handwritten Arabic characters. The written characters were manually

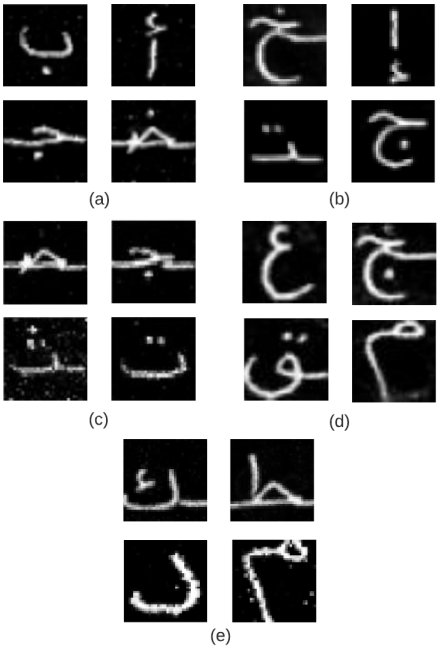

Fig. 2: Examples of different generated handwritten Arabic characters from different GANs. (a) BasicGAN. (b) DCGAN. (c) VanillaGAN. (d) WGAN. (e) BiGAN.

written and collected from sixty individuals. The dataset was divided into training and testing sets. The training set includes 13,440 images of characters while the testing set includes 3,360 images of characters. However, this dataset doesn't contain connected forms of Arabic characters which are essential in the Arabic language. To the best of our knowledge, this is the only handwritten Arabic characters dataset publicly available.

Generating and evaluating face images had been reported by recent Some research works in [6, 9]. Zhou et al. [9] used Amazon Mechanical Turks (MTurks)[1] to evaluate generated faces and other objects from FFHQ, CelebA, ImageNet, and CIFAR-10 datasets. StyleGAN, ProGAN, BEGAN, and WGAN-GP were evaluated using CelebA-64 and CIFAR-10 datasets. SN-GAN and BigGAN were evaluated using ImageNet. However, MTurks are not suited for some domains of generated images. One of these domains is the domain of Arabic characters in our research work. In this case of evaluation, it requires humans with an Arabic language background.

Heusel et al. [6] introduced another evaluation method, called Fréchet Inception Distance (FID). The purpose of FID measurement tool was to evaluate GANs by obtaining the similarity of the generated data to the original data. FID was applied to images with Gaussian noise, Gaussian blurr, black rectangles implanting, swirling, and noise with salt and pepper to compute a disturbance level.

In this research work, we apply different GANs to generate different handwritten Arabic characters. Also, we use connected and non-connected forms of Arabic characters in our experiments. Then, the resulted generated images from these GANs are evaluated using FID and native-Arabic human evaluations. The evaluations are used to evaluate the quality of the generated images by estimating the their realness and fakeness.

III. Our Handwritten Arabic Characters Dataset

In this section, we briefly describe our handwritten Arabic character datatset, namely ArabicChar dataset. The details of this dataset is also provided.

We manually created 134 images of handwritten continuous characters. Nine individuals were involved in this creation. We created three examples for the character "dal" and "thal", four examples for the characters "beh", "dal", "feh", "lam", "reh", "sad", "tah", "the","theh", "zeh", "zah" and "wow", five examples for the characters "Jeem", "hah", "khah", "ain", "ghain", "yeh", "seen", "sheen" and "qaf", six examples for the character "alef", and finally seven examples for the characters "Heh", "kaf", "meem" and "noon". The number of examples per character depends on the number of the characters connected forms.

The size of each image is 32×32 stored in a png extension. The images were cropped using Lasso Tool in Photoshop and converted into binary format. Figure2 shows examples of some of the handwritten Arabic characters and their connected forms in this dataset. The original images in this dataset and the generated images are publicly available [2].

IV. Our Approach

Our approach is mainly divided into two parts, image generation and image evaluation. Figure 3 shows the pipeline methodology of the proposed approach.

In the image generation part, we apply different GANs such as VanillaGAN, DCGAN, WGAN, and BiGAN in addition to the BasicGAN to generate handwritten Arabic characters. We train each GAN seperately. Figure 5 shows the training curves for each GAN. After training, each GAN generates a different output since they differ in their architectures. We generate a total of 40,485 handwritten Arabic characters. Table I shows the number of generated images for each GAN.

After generating the images, we feed the generated images to native-Arabic human and FID for evaluations. The evaluations are aimed to estimate the realness and fakeness of the generated handwritten Arabic characters. The implementation, training, and evaluations details are provided in Section V.

TABLE I: Number of generated handwritten Arabic characters for each GAN.

GAN	Generated Numbers
BasicGAN	12,350
DCGAN	1,935
VanillaGAN	14,800
WGAN	9,900
BiGAN	1,500

[1]https://www.mturk.com/.

[2]The link will placed upon acceptance.

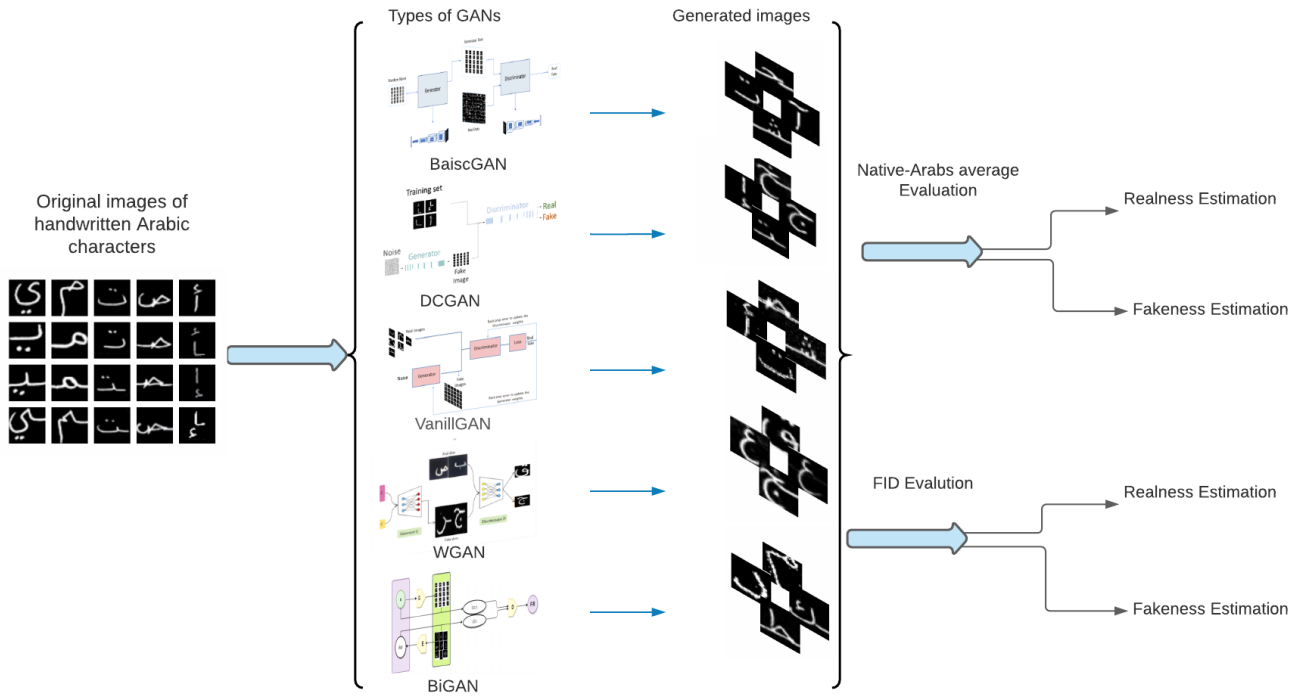

Fig. 3: The pipeline methodology of the proposed approach.

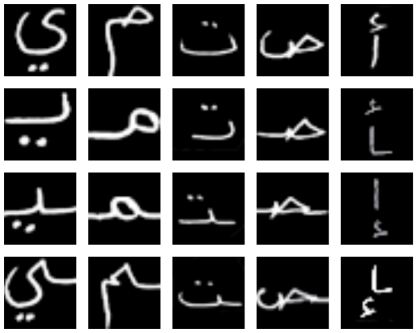

Fig. 4: Samples of some of handwritten Arabic characters and some samples of their connected forms in ArabicChar dataset.

V. Experiments

In this section, we provide implementation and experimental evaluation details for the proposed approach.

A. Implementation

Our experiments for the different GANs and FID evaluations were run on Google Colaboratory with Tensorflow 1.1.0 [1] and PyTorch [7]. A Python programming language version 3.4.3 and Keras were used in the implementation.

BasicGAN. The batch size, learning rate, and momentum were set to 1,000, 0.0002, and 0.5 respectively. Adam optimizer was applied. We used 50,000 epochs. The discriminator consists of five layers while the generator consists of three layers. Sigmoid and Leaky ReLU activation functions were used in the discriminator while Tangent Hyperbolic activation function was only used in the generator. A binary cross-entropy was used as a loss function.

DCGAN. The batch size, learning rate, and momentum were set to 32, 0.0002, and 0.5 respectively. Adam optimizer was applied. We used 3,000 epochs. The discriminator consists of four layers while the generator consists of five layers. Tangent Hyperbolic activation function was used for both discriminator and generator architectures. The last layer in the discriminator used a sigmoid activation function. A binary cross-entropy was used as a loss function.

VanillaGAN. The batch size, learning rate, and momentum were set to 128, 0.0002, and 0.5 respectively. Adam optimizer was applied. We used 2,000 epochs. The discriminator consists of three layers while the generator consists of two layers. Leaky ReLU activation function was used in both the discriminator and the generator. A dropout is also used. A binary cross-entropy was used as a loss function.

WGAN. The batch size, learning rate, and momentum were set to 128, 0.0002, and 0.5 respectively. Adam optimizer was applied. We used 10,000 epochs. The discriminator and the generator consist of eight layers. The sigmoid activation function was used in the discriminator while Tangent Hyperbolic activation function was used in the generator. A Wasserstein was used as a loss function.

BiGAN. The batch size, learning rate, and momentum were set to 32, 0.0002, and 0.8 respectively. Adam optimizer was applied. We used 400 epochs. The discriminator and the generator consist of eight layers. A sigmoid activation function was used in the discriminator while Tangent Hyperbolic activation function was used in the generator. A binary cross-entropy was used as a loss function.

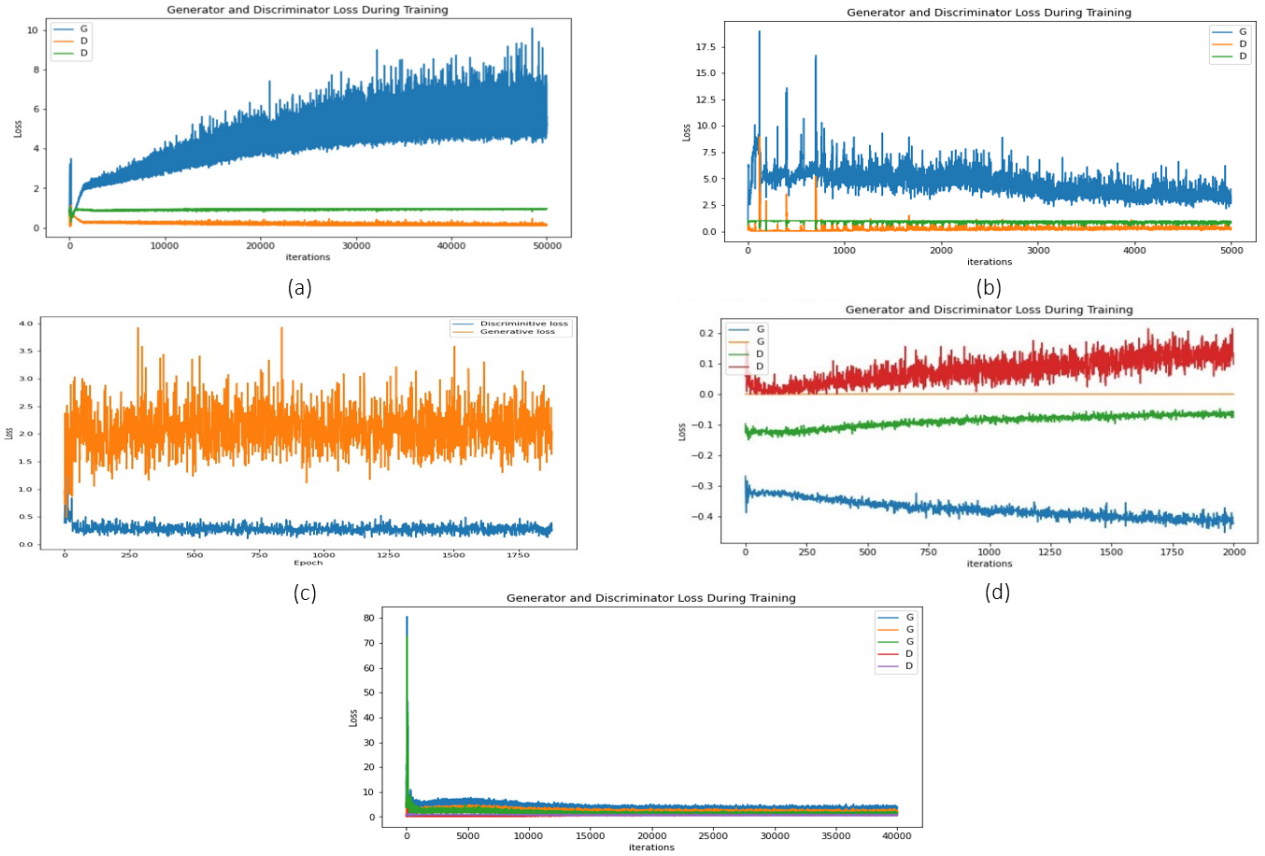

Fig. 5: Training curves for discriminators and generators. (a) BasicGAN. (b) DCGAN. (c) VanillaGAN. (d) WGAN. (e) BiGAN.

TABLE II: Native-Arabic human and FID Evaluations on handwritten ArabicChar dataset. Best values are in bold.

GAN	Reals Error	Fakes Error	Accuracy	FID	Human Evaluation	Training Time in Seconds
BasicGAN	92%	99%	0.9150	311.522	17%	28,800
DCGAN	**81%**	100%	0.8125	227.651	**35%**	21,600
VanillaGAN	**81%**	97%	0.8115	163.203	8%	21,600
WGAN	87%	95%	0.9211	**96.007**	10%	**7,200**
BiGAN	100%	**41%**	**1.000**	353.808	30%	32,400

B. Experimental Results and Evaluations

After training these different GANs, we find out that WGAN achieves 100% training accuracy which is better than other GAN models. DCGAN and VanillaGAN have lower real and fake images errors with 81% compared to other models. For efficiency, WGAN achieves a faster training compared to other models which is achieved in 7,200 seconds. Table II shows the training time for each GAN.

The purpose of evaluations is to estimate the realness and fakeness of the generated handwritten Arabic characters resulted from different GANs. In this research work, we apply human and non-human different evaluations. In our evaluation method, we first use native-Arabic humans to evaluate the generated images. Then, FID is applied for the evaluation.

Native-Arabic Human Evaluation. We used two hundreds and four volunteered native-Arabic humans for the evaluation through an electronic survey using Google Forms. The survey consists of ten questions. Each question asks each evaluator to estimate the realness and fakeness of the generated images. To attain high quality evaluators, only native-Arabic evaluators are selected. Each question is associated with a score which has five options of different GANs generated images. The answers of native-Arabic individuals are different. To obtain the best result from these answers, the average of the corrected results is computed to obtain the final score. The larger score is the better result. Figure7 illustrates the final native-Arabic human evaluations results from the the electronic survey. The evaluation results reveal that DCGAN achieves a 35% of average score which is higher than other GAN models as shown in Table II and Figure 1. BiGAN model is placed second with a 30% of average score. Figure 6 shows qualitative results on handwritten ArabicChar dataset. The larger score of human evaluation points out a better GAN model.

FID Evaluation. FID [6] is a measure that computes a similarity distance between the real images and the generated images. The similarity distance is computed by

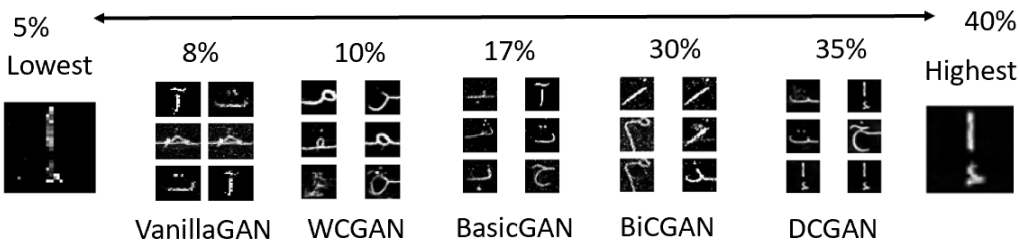

Fig. 6: Samples of generated images from BasicGAN, VanillaGAN, DCGAN, WGAN, and BiGAN trained on our dataset. The images from left to right shows lower to higher native-Arabic human scores.

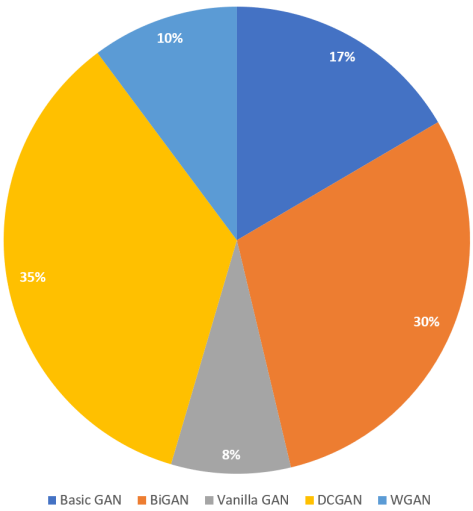

Fig. 7: The final native-Arabic human evaluations results.

computing the fake and real images Gaussians as follows:

$$d^2\left((\boldsymbol{m},\boldsymbol{C}),(\boldsymbol{m}_w,\boldsymbol{C}_w)\right) = \|\boldsymbol{m}-\boldsymbol{m}_w\|_2^2 + \operatorname{Tr}\left(\boldsymbol{C}+\boldsymbol{C}_w-2\left(\boldsymbol{C}\boldsymbol{C}_w\right)^{1/2}\right) \quad (2)$$

where d is the distance, C is the Gaussian, and m is the mean of the feature representation. w refers to the real samples.

The FID evaluation results reveal that WGAN achieves a lower score with 96.007 compared to the other GAN models as shown in Table II. The lower score of FID evaluation points out a better GAN model. VanillaGAN is placed second with a 163.203 score. Figure 1 shows our FID evaluation metric on our handwritten ArabicChar dataset.

VI. Discussion

In this section, we discuss the appearance and clarity of handwritten Arabic characters during the training in each GAN. In BasicGAN, the generated characters started to appear slowly after 1,800 epochs. Their shapes became more clear in 50,000 epochs. In DCGAN, the characters were noisy in early epochs around 50. After that, the quality of the characters began to improve. However, some characters have overlapped in around 650 epochs. The characters started to become more clear and separated after 1,000 epochs. Then, the characters started to appear clearly in 3,000 epochs. In VanillaGAN, the characters started to appear in 140 epochs. The characters became more sharp and clear in 2,000 epochs. In WGAN, we had a difficulty for generating the characters due to the use of the RMSprop optimizer. The images were noisy. So, we replaced it with the Adam optimizer. Then, the characters were clear at 10,000 epochs. In BiGAN, the characters started to appear in 2,000 epochs. The characters were clearly seen in 40,000 epochs.

VII. Conclusion

In this research work, we apply different GANs to generate handwritten Arabic characters. BasicGAN, VanillaGAN, DCGAN, BiGAN, and WGAN are used. Then, the results of the generated images are evaluated using native-Arabic human and FID. The qualitative and quantitative results are provided for the images generation and evaluation. In experimental evaluation, WGAN achieves better results in FID with a value of 96.007. On the other hand, DCGAN achieves better results in native-Arabic human evaluation with a value of 35%.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[3] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[4] Ahmed El-Sawy, Mohamed Loey, and Hazem El-Bakry. Arabic handwritten characters recognition using convolutional neural network. *WSEAS Transactions on Computer Research*, 5:11–19, 2017.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[6] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local

nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

[8] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[9] Sharon Zhou, Mitchell Gordon, Ranjay Krishna, Austin Narcomey, Li F Fei-Fei, and Michael Bernstein. Hype: A benchmark for human eye perceptual evaluation of generative models. In *Advances in Neural Information Processing Systems*, pages 3449–3461, 2019.