Conditional Generative Adversarial Network for Devanagari Handwritten Character Generation

Mamta Bisht
Electronics and communication Engineering
Jaypee Institute of Information Technology
Noida, India
17402010@mail.jiit.ac.in

Richa Gupta
Electronics and communication Engineering
Jaypee Institute of Information Technology
Noida, India
richa.gupta@jiit.ac.in

***Abstract*— In the deep learning era, the model requires more variations in the large dataset for better understanding. Generative Adversarial Networks (GANs) are often used to learn to generate various images. The design of GAN consists of a collection of neural networks used in an unsupervised way. In GAN, many kinds of arrangements of neural networks are possible for various applications in many areas of computer vision and machine learning. Conditional Generative Adversarial Network (CGAN) is one of them helpful in the generation of the definite type of images. This work illustrates the use of CGAN for generating specific Devanagari handwritten characters.**

Keywords*— *CGAN, Devanagari Handwritten Character Generation, Convolutional Neural Network, Deep Learning.

I. INTRODUCTION

Generative Adversarial Networks are initially presented by Ian J. Goodfellow et al. [1] in 2014 and abbreviated as GANs. GANs are used for generative modeling which is an unsupervised learning problem using deep neural network technique, although a clever design of GAN makes the generative model a supervised learning problem. Many models are extended for generators as GAN offers new training designs and have practical applications in many areas of computer vision and machine learning such as handwritten font generation [2], text synthesis [3], image-to-image translation [4], [5], face aging [6], [7], image manipulation applications [8], object detection [9]–[11], medical fields [12], style transfer [13] and many more [14]. The Conditional Generative Adversarial Networks (CGANs) is one of them and provides a method to control the generation of image attributes. GANs are architecture for generator model and a discriminator model as follows:

A. Generator

The input to the model is fixed-length random vector (also called latent space) drawn from a Gaussian distribution and it generates new images in the problem area. The nodes in random input vector space will correspond to the nodes in the problem area and after training, providing a reduced appearance of the data distribution. The basic block diagram of generator is shown in Fig. 1. Convolutional neural networks are used in the generator model.

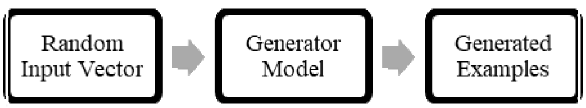

Fig. 1. Basic block diagram of generator.

B. Discriminator.

It is a classification model to classify examples as actual or generated. The training dataset consists of actual images and the generated images come from the output of generator model. The basic block diagram of discriminator is shown in Fig. 2. Convolutional neural networks are used in the discriminator model.

Fig. 2. Basic block diagram of discriminator.

The architecture of GAN is shown in Fig. 3. The training of generator for generating realistic images and discriminator model for distinguishing in actual and generated images are set jointly. The generator creates a batch of images, which are then given to the discriminator, along with actual instances from the area, to be classified as actual or generated images. In the next round, the discriminator is modified to update its understanding to distinguish between actual and false images, while the generator is modified depending on how effectively the generated images deceived the discriminator. In this way, the generator model and the discriminator model are competing against one another. When the discriminator correctly differentiates between actual and false images, it is benefitted and the model parameters do not need any modification, whereas the generator is penalised and the model parameters need modification. When the generator deceives the discriminator, it is benefitted and the model parameters are not changed, while the discriminator is penalised and its model parameters are updated.

In [15] authors proposed a multi-scale multi-class GAN for Chinese handwritten characters generation. To control the outputs with the different conditions they added additional knowledge in both generator and discriminator. In [2], the authors presented a Chinese font generation approach depending on the least-squares CGAN. Generation of Bangla handwritten characters is discussed in [16]. CGAN is also used by authors in [17] for synthetic class-specific Bangla handwritten character generation. In literature, works found for Devanagari handwritten character recognition [18], [19] using deep learning techniques but the huge data accessibility is difficult particularly in Indian languages for better understanding of deep learning models. This paper discusses a special type of GAN called CGAN for

Devanagari handwritten characters generation. The experimental results on Devanagari Handwritten Characters Dataset (DHCD) show that the image generated by CGAN is clearer and the training is stable.

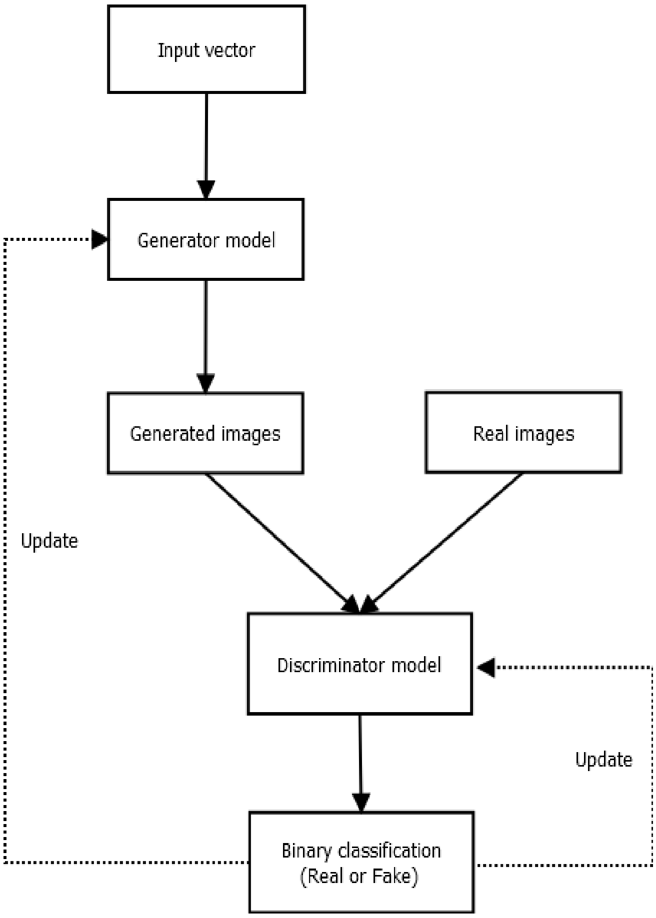

Fig. 3. GAN architecture.

II. Proposed Work

The conditional generative adversarial network (CGAN), is a GAN that uses a generator model to conditionally generate images [20]. If a class label is available, image generation can be conditional on it, allowing for the targeted generation of images of a specific type. The discriminator is also conditioned on two inputs as input sample (actual or generated) and the additional label information. In this way, a conditional GAN can be used to create samples from a domain of a specific form. The architecture of CGAN is shown in Fig. 4.

Network Architecture: CGAN

In this section, the architecture of the proposed CGAN model for the DHCD is discussed. The dataset consists of Devanagari handwritten characters of 78,200 samples for training and 13,800 samples for testing. Here, an embedding layer is added in the discriminator and generator model to provide additional class label information to the model. The description of the discriminator and the generator used in the proposed CGAN is as follows:

Discriminator: A new second input is defined as an integer for the class label of the image. The 46 classes for the DHCD map to a different 50-element vector through the embedding layer. The output is forwarded to a fully connected layer with linear activation and then reshaped into one channel of a 28x28 image and further concatenated with the input image. It looks like a two-channel input image to the next convolutional layer which has 3x3 filters of depth 128 and 2x2 stride value. LeakyReLU is used as an activation function for the discriminator. Next one more convolutional layer which has 3x3 filters of depth 128 and 2x2 stride value is also added with the same activation. Further, the flattening layer and a dropout of 0.4 are also added. Further, a dense layer with sigmoid activation is added for binary classification as real or fake. The model is compiled with Adam optimizer and loss as binary cross-entropy.

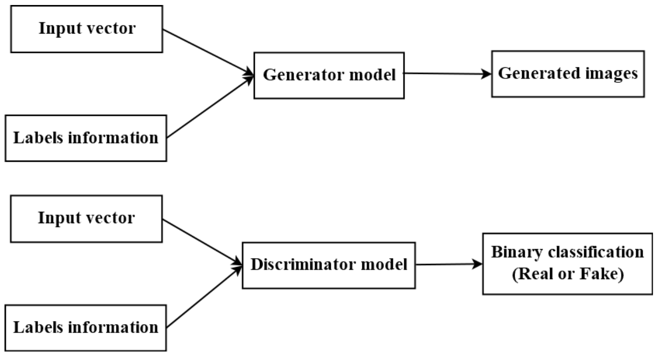

Fig. 4. CGAN architecture.

Generator: In the generator model the latent space points are conditional on the provided class label and the additional class information is added through embedding layer similar to discriminator model. The output is forwarded to a fully connected layer with linear activation and then reshaped into one channel of a 7×7 feature map. The latent space input to the generator is followed by a fully connected layer and activation and then reshaped into a low-resolution image of 7x7 with 128 channels. Next concatenation of 7×7×1 and 7×7×128 is used and then up sampled twice to outputs a single 28×28 grayscale image.

III. Experimental Analysis & Result

In [18] authors discuss 36 basic Devanagari characters and 10 basic numeral characters. We worked on these 46 basic classes of Devanagari script. We have run the discussed model for 100 epochs. We found that most of the generated character or numeral from the generator seemed realistic as shown in Fig. 5, Fig. 6, Fig. 7 and Fig. 8. These figures show 10 handwritten generated characters and numerals in sequential order of Devanagari characters and numerals.

IV. Conclusion

The dataset is the most important component in machine learning era. However, acquiring data is one of the most difficult tasks in many cases. Conditional generative adversarial networks, on the other hand, can create any type of data of a certain class for machine learning. In this work, we explore CGAN network and proposed a convolutional neural network based generator and discriminator for it. We found that the proposed CGAN model performs well and the generated images look very realistic. In the future, the work may be extended for handwritten Devanagari words generation. The results obtained from CGAN can be further analysed using Frechet Inception Distance (FID).

Fig. 5. CGAN generated samples for क ख ग घ ङ च छ ज झ ञ ट ठ.

Fig. 6. CGAN generated samples for ड ढ ण त थ द ध न प फ ब भ.

Fig. 7. CGAN generated samples for म य र ल व श ष स ह क्ष त्र ज्ञ.

Fig. 8. CGAN generated samples for ० १ २ ३ ४ ५ ६ ७ ८ ९.

REFERENCES

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," Advances in neural information processing systems, vol. 27, pp. 2672–2680, 2014.

[2] X. Lin, J. Li, H. Zeng, and R. Ji, "Font generation based on least squares conditional generative adversarial nets," Multimedia Tools and Applications, vol. 78, no. 1, pp. 783–797, 2019.

[3] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 1316–1324.

[4] C. Wang, C. Xu, C. Wang, and D. Tao, "Perceptual adversarial networks for image-to-image transformation," IEEE Transactions on Image Processing, vol. 27, no. 8, pp. 4066–4079, 2018.

[5] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8789–8797.

[6] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 5810–5818.

[7] Z. Wang, X. Tang, W. Luo, and S. Gao, "Face aging with identity-preserved conditional generative adversarial networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7939–7947.

[8] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen, "Attgan: Facial attribute editing by only changing what you want," IEEE transactions on image processing, vol. 28, no. 11, pp. 5464–5478, 2019.

[9] K. Ehsani, R. Mottaghi, and A. Farhadi, "Segan: Segmenting and generating the invisible," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 6144–6153.

[10] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1222–1230.

[11] Y. Bai, Y. Zhang, M. Ding, and B. Ghanem, "Sod-mtgan: Small object detection via multi-task generative adversarial network," in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 206–221.

[12] Y. Xue, T. Xu, H. Zhang, L. R. Long, and X. Huang, "Segan: Adversarial network with multi-scale l 1 loss for medical image segmentation," Neuroinformatics, vol. 16, no. 3, pp. 383–392, 2018.

[13] H. Chang, J. Lu, F. Yu, and A. Finkelstein, "Pairedcyclegan: Asymmetric style transfer for applying and removing makeup," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 40–48.

[14] A. Jabbar, X. Li, and B. Omar, “A survey on generative adversarial networks: Variants, applications, and training,” arXiv preprint arXiv:2006.05132, 2020.

[15] J. Liu, C. Gu, J. Wang, G. Youn, and J.-U. Kim, “Multi-scale multi-class conditional generative adversarial network for handwritten character generation,” The Journal of Supercomputing, vol. 75, no. 4, pp. 1922–1940, 2019.

[16] M. F. Sikder, “Bangla handwritten digit recognition and generation,” in Proceedings of International Joint Conference on Computational Intelligence, 2020, pp. 547–556.

[17] Z. K. Nishat and M. Shopon, “Synthetic Class Specific Bangla Handwritten Character Generation Using Conditional Generative Adversarial Networks,” in 2019 International Conference on Bangla Speech and Language Processing (ICBSLP), 2019, pp. 1–5.

[18] M. Bisht and R. Gupta, “Multiclass recognition of offline handwritten Devanagari characters using CNN,” Int. J. Math. Eng. Manage. Sci., vol. 5, pp. 1429–1439, 2020.

[19] M. Bisht and R. Gupta, “Offline handwritten Devanagari modified character recognition using convolutional neural network,” Sādhanā, vol. 46, no. 1, pp. 1–4, 2021.

[20] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” arXiv preprint arXiv:1411.1784, 2014.