iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** V **Month of publication:** May 2025

DOI: https://doi.org/10.22214/ijraset.2025.70518

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

Synthetic Devanagari Digits Dataset

Pushkar Joglekar[1], Madhur Vaidya[2], Soham Nimale[3], Hrishikesh Potnis[4], Sarthak Pithe[5]
Department of Computer Engineering, Vishwakarma Institute of Technology

Abstract: ***Devanagari is the most widely used script of Indian languages of Proto-Indo-European origin. However, relatively less research has been conducted on Devanagari, leaving room for significant exploration and advancement. Generative Adversarial Networks (GANs) are unsupervised Machine Learning models which are used for smart augmentation of a dataset. In this paper, we create a collection of high-quality images of devanagari numeric digits from scratch using a novel method of collecting the input data through electronic medium only. This leaves little room for error and unwanted noise compared to conventional methods. The images are of size 128x128 pixels, totalling to 12000 images. Then we train two types of GANs - DCGAN and CGAN - on the images and compare the result of them. In general, the DCGAN models were found to perform much better and produce more realistic images than CGAN. Our work is useful in several applications including Optical Character Recognition and Accessibility Tools.***

Keywords: Character-Recognition, Data-Augmentation, Dataset-Generation, Devanagari, GANs

I. INTRODUCTION

The recent trend of Machine Learning models has suddenly increased the importance of data. Machine Learning models high-quality require data in large quantities for training. Image data is a very important part of this. Text-based image data is required for a variety of tasks like Optical Character Recognition (OCR) technology. For detecting handwritten texts, a model with better performance is needed as compared to applications where only printed material is to be detected. Scripts like Latin already have high-quality datasets of characters and digits available - the most famous of them being the MNIST dataset.

On the other hand, very little research is done on scripts such as Devanagari. Devanagari script is used in several languages including Hindi, Marathi and Sanskrit. Yet there is a scarcity of good quality datasets for this script. The Devanagari script has a lot of variations which change from place-to-place, and thus the script is inherently complex in nature. It is important to study and research about this for the preservation of the script. Manually creating large datasets for Devanagari is very cumbersome due to the slow process and the complexity involved. Thus, techniques such as image augmentation can be used for increasing the size and variety of the datasets. While augmenting using techniques such as rotation and shear can be useful, they are helpful only up to a certain degree as no new information is added. But techniques such as GANs (Generative Adversarial Networks) and VAEs (Variational Autoencoders) can be used as they generate the images from scratch after 'learning' about the datasets. Good quality GANs are always beneficial in increasing the size and variety of datasets by a significant amount. In this paper, we initially want to create a high-resolution dataset for all the Devanagari numeric digits. We also want to assess the extent to which different types of GANs can be used to generate images of Devanagari numeric digits, and compare it with the pre-existing methods too.

The overview of this paper is as follows. The Sections 1 and 2 include the introduction and the related work respectively. They discuss the limitations of the existing systems. Section 3 discusses the methodology for generating Devanagari digits using two different types of GANs. Section 4 talks about the results given by our proposed. Section 5 discusses the possible ways to improve upon and further the research, followed by the conclusion of the paper in Section 6.

II. LITERATURE REVIEW

In their paper, Sinha K and Gupta R [1] created a DCGAN Architecture for all the devanagari characters. They used images of 32x32 size as input. But their results are not great and visually appealing as their generated output is not up to the mark of their original dataset.

G Jha and H Cecotti [2] in their paper created Generative Adversarial Networks for 4 different scripts - Bangla, Devanagari, Latin and Oriya. Their approach was based in such a way to study if the performance classifier models like CNNs can be increased by increasing the size of the input data for the model by including GAN-generated images. They devised seven different types of tests to compare CNN performance on original input data versus GAN-based input data. They found out that adding a few GAN-generated images (up to the size of the original dataset itself) can increase the classifier performance, but increasing that even more decreases the performance and accuracy of these models.

©IJRASET: All Rights are Reserved

Warkhandkar et al. [3] created a GAN for all devanagari characters, on images of size 32x32. Their generated images are not smooth and the lines have gaps in them, and the performance can definitely be improved. S Kaur and K Verma [4] also created a GAN for devanagari characters but the result has scope for betterment.

D Vishwakarma et al. [6] compared the performance of GANs which generate single characters at a time (DCGAN) to CGAN. They used the MNIST dataset for this, and concluded that DCGANs are better at producing artificial images as compared to CGANs.

In M Bisht and R Gupta's paper [7], they mentioned their approach for creating a single Conditional GAN (CGAN) for creating Devanagari characters and digits. The numeric digits generated by the CGAN are of quite remarkable quality as CGAN is usually of inferior quality as compared to GANs which generate a single character (like DCGANs).

Several papers have created GANs for various scripts other than Latin, like Bangla [8], Nepali [9] [10], Kannada [11], Tamil [12], Gurumukhi [13] and Arabic [14].

In [15], Lajish and Sunil study various strokes of characters of Devanagari language, and created the primitive and feature set for devanagari character set too.

A notable dataset of Devanagari characters was introduced by Acharya et al. in [16]. This dataset consists of a total of 92 thousand images of 46 classes (characters). All the images here are of size 32x32 pixels. The whole dataset was created by scanning handwritten characters. This is the standard dataset which is used as the data source by several papers.

From this Literature Review, we concluded that several attempts have been made to generate digits and characters of Devanagari digits, though there is large scope for improvement. In nearly all of the cases, the created GANs do not generate images of a quality comparable to the original dataset. CGANs often lack in quality compared to DCGANs but CGANs are much easier as only one model can be used to generate images of all classes. Along with this, all of the papers either create a dataset or use one which was based on asking volunteers to write the characters or digits on paper sheets, scanning them and then creating a dataset out of it. While this approach has its merits, it is often prone to errors and irregularities, and is a manual labour for converting the handwritten sheets to actual digital images.

III.METHODOLOGY

A. Overview

The methodology in this study has several stages, namely-

1) Data Acquisition
2) Data Preprocessing
3) Synthetic image generation using Generative Adversarial Networks (GANs)
4) Performance Monitoring

B. Data Acquisition

We propose a novel method of collecting handwritten text data, through an Android application. Conventionally, handwritten data collection is a physical process. Contributors handwrite text physically on paper, and then they are scanned to use it for training. These images need to further go through various preprocessing steps like normalization.etc. All this leads to lower resolution images. Many handwritten datasets are created in this way, and are thus prone to errors.

In contrast, in digital methods, the data is acquired through a digital framework itself. This does not lead to loss of data as no analog-to-digital conversion occurs once the data is recorded. In our case, users directly write the numeric digit on a canvas of 128x128 pixels through the application. Due to this, we get high resolution images which are less prone to noise. Therefore less preprocessing is required and quality of data is maintained. The application randomly varies the thickness of the brush in the drawing, which helps create and maintain variety in the images. Lastly, the physical method is confined to a smaller distribution of contributors and hence is possibly biased, but in the digital method anyone with the access of the application can contribute. This novel method helped create a high-quality and high-resolution initial image dataset.

C. Data Preprocessing

Once the images were collected, they were cleaned and prepared for training. Data augmentation methods like rotating and scaling the images to add more variety. We include translation of images because convolutional layers in the neural network handle small movements in the images.

D. Synthetic image generation using GANs

Generative Adversarial Networks (GAN) based models were used for generating synthetic images.

1) Deep Convolutional Generative Adversarial Networks (DCGAN)

A DC GAN consists of two neural networks, Generator and Discriminator. Generator takes random noise as input and outputs a synthetic image. Discriminator inputs both original and synthetic images and tries to discriminate which one is fake. During training, the generator improves its results trying to fool the discriminator, while the discriminator tries to get better at finding fake images.

The DC GAN's loss function is stated in Fig. 1

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

Fig. 1 DCGAN Generator loss function

where -
D = Discriminator output
G = Generator output
x = input image for Discriminator
z = input noise for Generator
E = Expected value

This is a minimax loss function. The Generator tries to minimize the loss, while the Discriminator tries to maximize it. The Generator can affect only the log(1-D(G(z))) term, so, for the Generator, the loss is equal to minimizing the log(1-D(G(z))) term.
This loss function may get stuck at a local minima, so a modified version of loss called the Wasserstein Loss can be used. In this case, the Discriminator tries to maximize the D(x) - D(G(z)) term, while the Generator tries to maximize the D(G(z)) term.

2) CGAN(Conditional Generative Adversarial Network)

CGAN is a type of Generative Adversarial Network, where we can put a condition on the GAN to generate a specific type of output only.
Due to this condition, the new loss function is as stated in Fig. 2.

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

Fig. 2 CGAN Generator loss function

where -
D = Discriminator output
G = Generator output
x = input image for Discriminator
y = condition
z = input noise for Generator
E = Expected value

E. Model Architectures

Both DC GAN and CGAN have their respective architectures for generator and discriminator models. DC GAN is trained to generate a specific digit, while CGAN can be trained for all digits at once. Therefore, in general architectures of generators and discriminators of CGAN tend to be more complex.

1) DCGAN

Generator of DCGAN has an input noise of 100 pixels, and outputs a 128x128 image. It consists of convolutional layers followed by dense layers. It is a sequential model.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

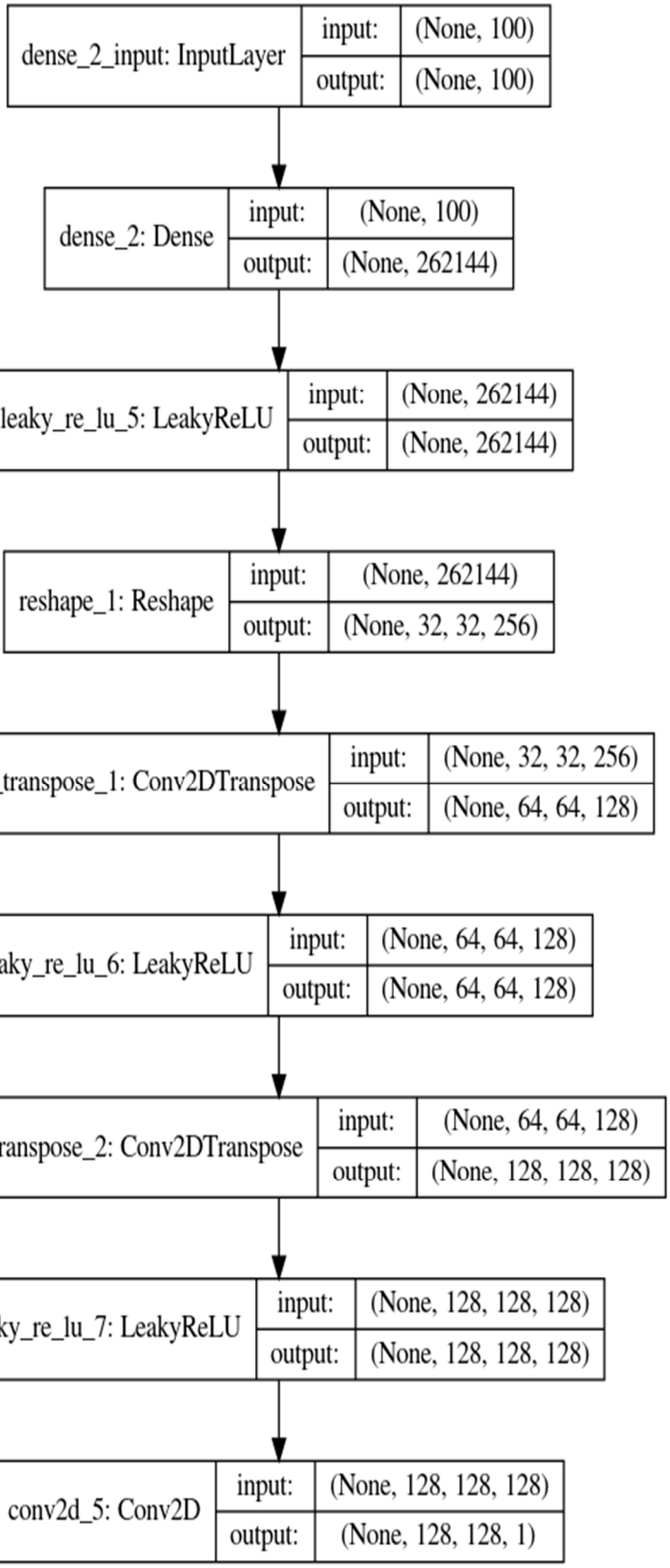


Fig. 3 DCGAN Generator Architecture

The discriminator in GAN takes the generated output of the generator and outputs 0 or 1, that is true or false.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

2) CGAN

Purpose of the Generator model in CGAN is similar to that of DCGAN. It takes a noise vector of size 42 pixels as input, and we get a 128x 128x1 greyscale image. The generator is made up of multiple convolutional layers, followed by batch normalization. It is a non sequential model.

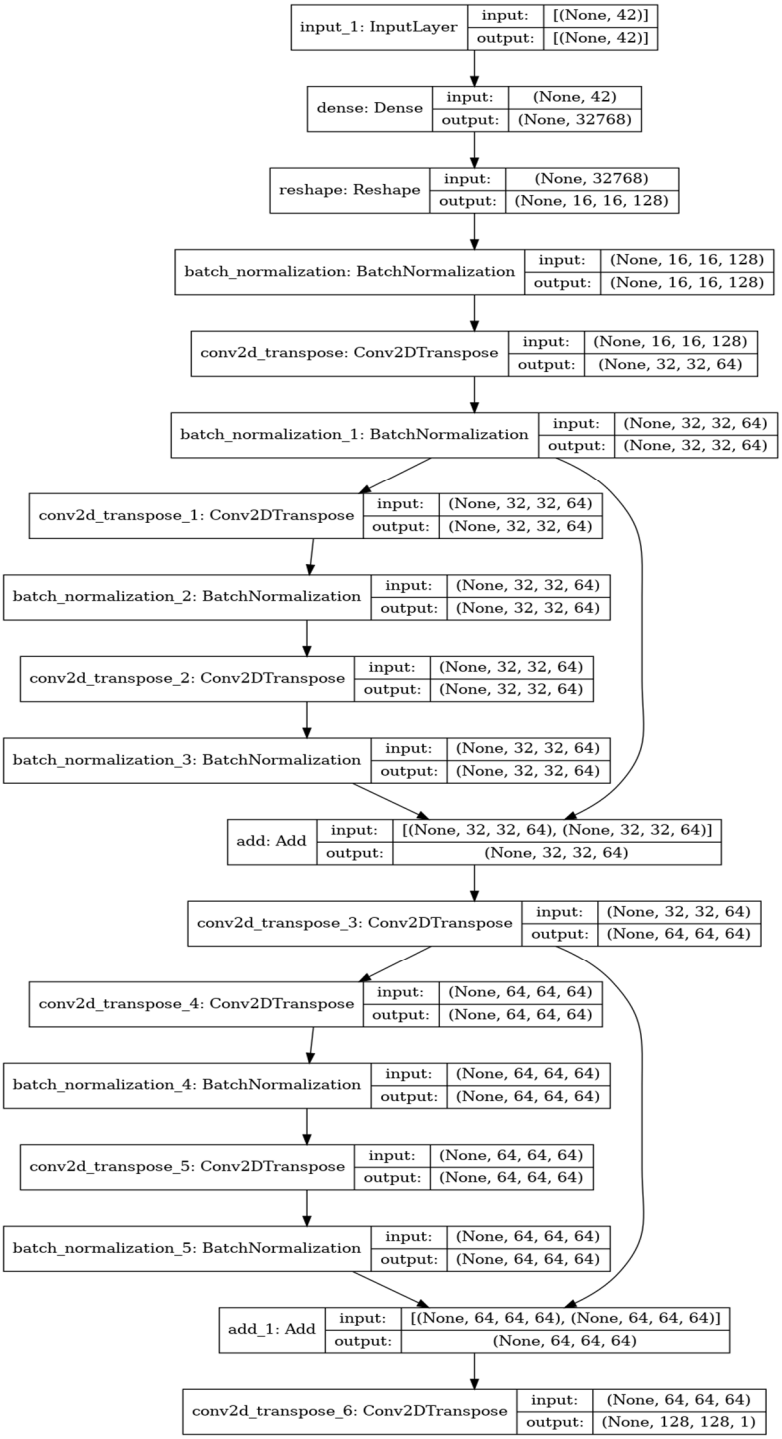


Fig. 5 CGAN Generator Architecture

The discriminator model takes the output of the generator model and its label for which it was generated, and outputs 0 or 1 that is false or true. It is a sequential model having convolutional layers, and some dropout layers for regularization.

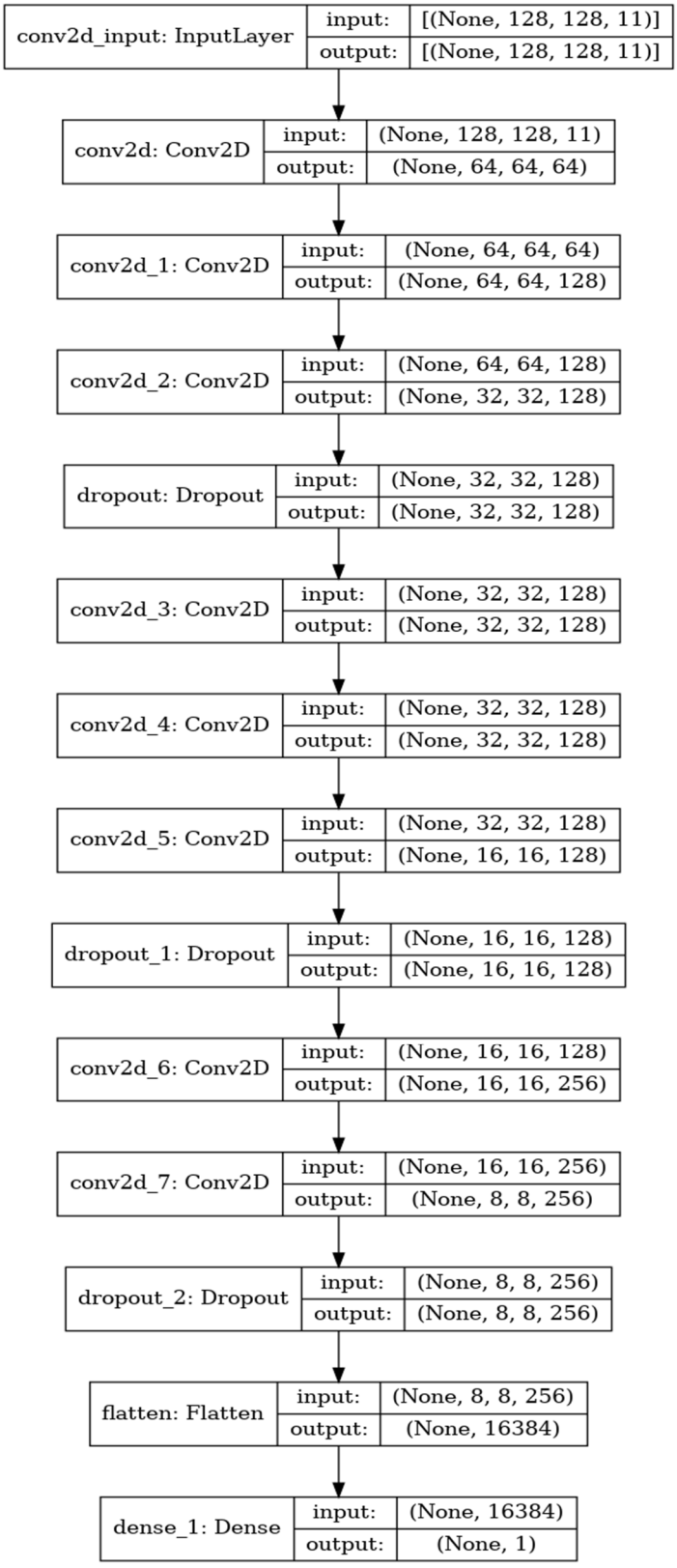


Fig. 6 CGAN Discriminator Architecture

IV. RESULTS

The data was collected as 128x128 sized images, of total 10 classes (one class pertaining to one numeric digit). A total of 96 volunteers helped us with collecting a total of 12000 images (1200 images of each class). The data was augmented to a size of 6000 images per class, which totals to 60000 images.

The data was collected using a simple Android Application on which the volunteers were asked to draw a series of digits. Figure 7 shows the screenshot of the app.

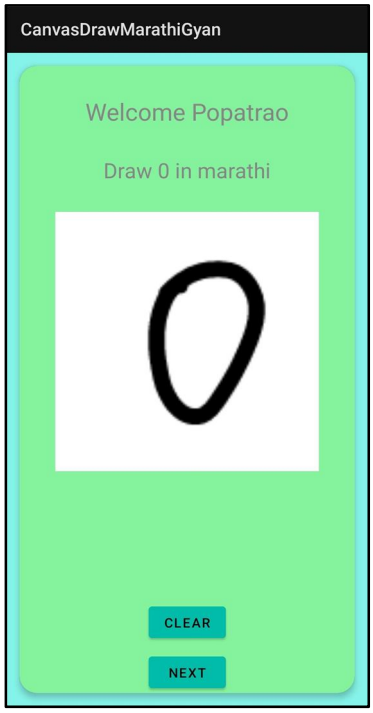


Fig. 7 Screenshot of the application used for taking input data

A. *GAN Output*

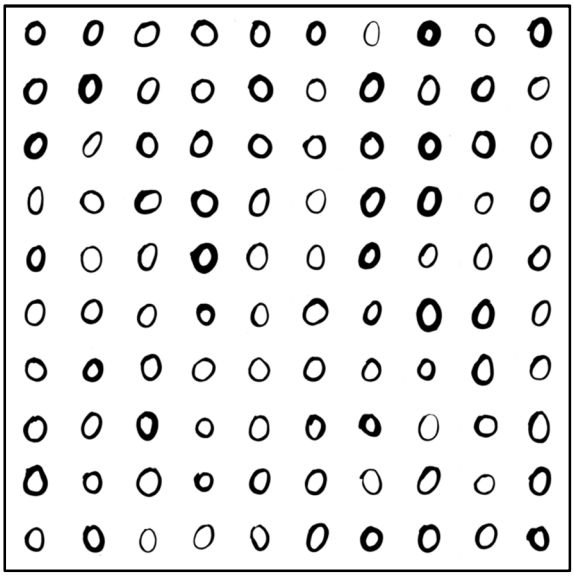


Fig. 8 Generated image of the digit zero in Marathi by GAN generator

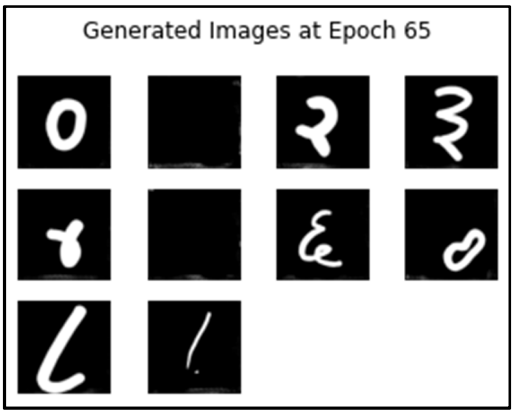


Fig. 9 Generated Images of all Marathi digits by CGAN generator after 65 epochs.

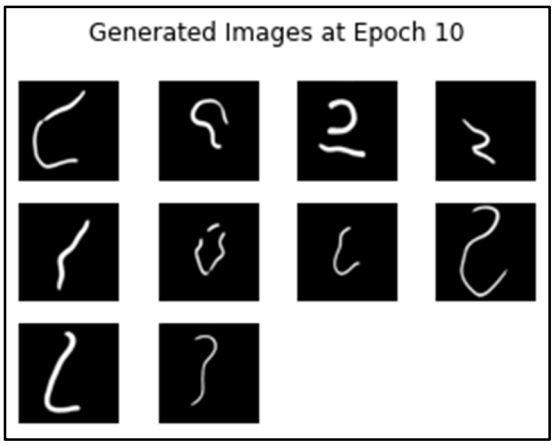


Fig. 10 Generated Images of all Marathi digits by CGAN generator after 10 epochs.

For all the digits, we trained a separate GAN model and a combined GAN model (CGAN) for all the digits combined. The output of Generator of 0 digits of the GAN separate models after 100 epochs/training steps is shown in Figure 8. Figure 9 shows the output of CGAN for each of the numeric digits after 65 epochs/training steps. Clearly, the CGAN model generated the majority of the digits correctly with excellent boldness and shape. On the other hand, the other digits were generated with very low quality; some of them being indistinguishable from the background. A comparative analysis of the images of 10 different Devanagari numeric digits generated by CGAN in the 10th and the 65th epoch (refer to Figure 9 and Figure 10) reveals the quick improvement of quality that CGAN models present.

We observed that the GANs trained for generating a single numeric digit almost always generated better images than the images generated by CGAN - except for the case of the digit '६' (6) - where the CGAN consistently produced better images than the GAN which was trained specifically to generate images of ६. The quality of the generated digit ६ can be observed in the Figure 9. The complicated nature of the digit is the cause for obtaining relatively low quality results using GAN. Meanwhile, since CGAN processes every digit at the same time, it is able to extract comparative information with respect to other Devanagari numeric digits, causing it to produce relatively better results.

The training time was also much faster for DCGANs as compared to the CGAN. All the individual DCGANs for 10 different digits took an average training time of 2 hours, while the CGAN took almost 12 hours. This is due to the large size of the input data, more training required as well as the non-sequential nature of the generator of CGAN. But overall, we observe that the training time of CGAN is much smaller as compared to the total training time of all the DCGANs on all the digits combined, which is about 20 hours.

V. FUTURE WORK

The future scope for this research includes various directions. One of the primary scope is to improve the CGAN model so that it performs as well as or even better than the DCGAN models. That would improve a lot of things - firstly, a single model which would be reliably generating all the digits, and secondly, the training time as only one model needs to be trained instead of several small ones.

Another possible direction is the creation of a large dataset of devanagari characters generated by the GAN models of size 128x128. This would be highly beneficial as with the help of GAN models, datasets of extremely large size can be easily created, while still maintaining the image quality.

VI. CONCLUSION

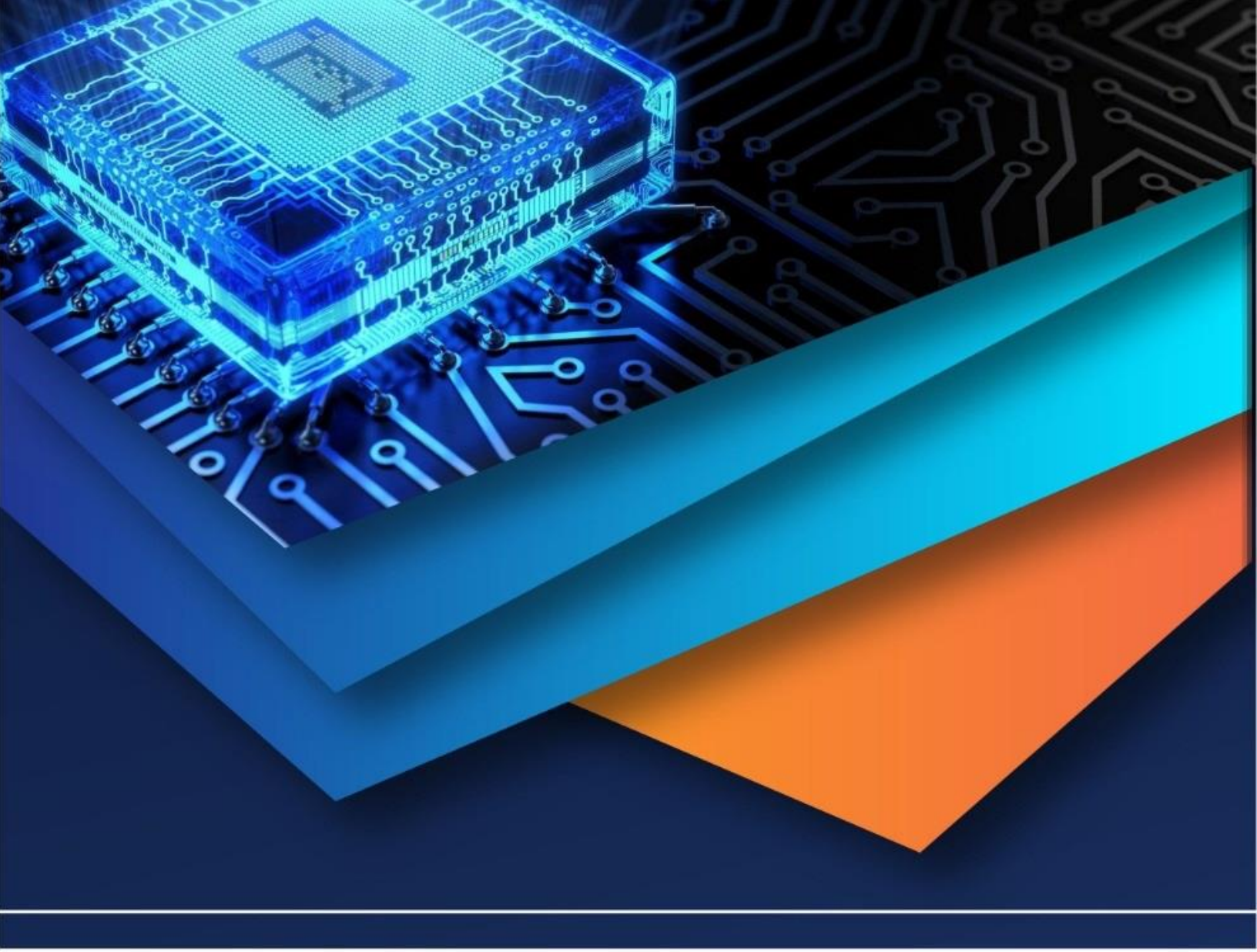
This paper focuses on the synthetic generation of Devanagari numeric digits using Generative Adversarial Networks (GANs). We tested two types of GANs - DCGAN and CGAN. We found out that the DCGAN models individually perform better than the single CGAN model. Collecting and using a dataset of large images (128x128) proved very beneficial. In general the output generated by the DCGANs was very identical to the original dataset.

REFERENCES

[1] Sinha, K., & Gupta, R. (2024, August). Enhancing Handwritten Devanagari Character Recognition via GAN-Generated Synthetic Data. In Proceedings of the 2024 Sixteenth International Conference on Contemporary Computing (pp. 484-489).
[2] Jha, G., & Cecotti, H. (2020). Data augmentation for handwritten digit recognition using generative adversarial networks. Multimedia Tools and Applications, 79(47), 35055-35068.
[3] Warkhandkar, A. G., Sharief, B., & Bhambure, O. B. (2020). Measuring performance of generative adversarial networks on devanagari script. arXiv preprint arXiv:2007.06710.
[4] Kaur, S., & Verma, K. (2020). Handwritten Devanagari character generation using deep convolutional generative adversarial network. In Soft Computing: Theories and Applications: Proceedings of SoCTA 2018 (pp. 1243-1253). Springer Singapore.
[5] Nannapaneni, R., Chakravarti, A., Sangappa, S., Bora, P., & Kulkarni, R. V. (2022). Augmentation of Handwritten Devanagari Character Dataset Using DCGAN. In Machine Intelligence and Smart Systems: Proceedings of MISS 2021 (pp. 31-44). Singapore: Springer Nature Singapore.
[6] Vishwakarma, D. K. (2020, May). Comparative analysis of deep convolutional generative adversarial network and conditional generative adversarial network using hand written digits. In 2020 4th international conference on intelligent computing and control systems (ICICCS) (pp. 1072-1075). IEEE.
[7] Bisht, M., & Gupta, R. (2021, November). Conditional Generative Adversarial Network for Devanagari Handwritten Character Generation. In 2021 7th International Conference on Signal Processing and Communication (ICSC) (pp. 142-145). IEEE.
[8] Haque, S., Shahinoor, S. A., Rabby, A. S. A., Abujar, S., & Hossain, S. A. (2019). Onkogan: Bangla handwritten digit generation with deep convolutional generative adversarial networks. In Recent Trends in Image Processing and Pattern Recognition: Second International Conference, RTIP2R 2018, Solapur, India, December 21–22, 2018, Revised Selected Papers, Part III 2 (pp. 108-117). Springer Singapore.
[9] Chhatkuli, R. K., Baral, H. P., & Surendra, K. C. (2021). Generating Nepali Handwritten Letters and Words Using Generative Adversarial Networks.
[10] Bhandari, B. B., Dhakal, A. R., Maharjan, L., & Karki, A. (2021). Nepali Handwritten Letter Generation using GAN. Journal of Science and Engineering, 9, 49-55.
[11] Shrisha, H. S., Anupama, V., Suresha, D., & Jagadisha, N. (2022). KGAN: A Generative Adversarial Network Augmented Convolution Neural Network Model for Recognizing Kannada Language Digits. In Communication and Intelligent Systems: Proceedings of ICCIS 2021 (pp. 523-531). Singapore: Springer Nature Singapore.
[12] Murugesh, V., Parthasarathy, A., Gopinath, G. P., & Khade, A. (2022). Tamil language handwritten document digitization and analysis of the impact of data augmentation using generative adversarial networks (GANs) on the accuracy of CNN model. In Machine Learning and Autonomous Systems: Proceedings of ICMLAS 2021 (pp. 159-177). Singapore: Springer Nature Singapore.
[13] Kaur, S., Bawa, S., & Kumar, R. (2023, May). Evaluating Generative Adversarial Networks for Gurumukhi Handwritten Character Recognition (CR). In Machine Intelligence Techniques for Data Analysis and Signal Processing: Proceedings of the 4th International Conference MISP 2022, Volume 1 (pp. 503-513). Singapore: Springer Nature Singapore.
[14] Eltay, M., Zidouri, A., Ahmad, I., & Elarian, Y. (2022). Generative adversarial network based adaptive data augmentation for handwritten Arabic text recognition. PeerJ Computer Science, 8, e861.
[15] Lajish, V. L., & Kopparapu, S. K. (2014, December). Online handwritten Devanagari stroke recognition using extended directional features. In 2014 8th International Conference on Signal Processing and Communication Systems (ICSPCS) (pp. 1-5). IEEE.
[16] Acharya, S., Pant, A. K., & Gyawali, P. K. (2015, December). Deep learning based large scale handwritten Devanagari character recognition. In 2015 9th International conference on software, knowledge, information management and applications (SKIMA) (pp. 1-6). IEEE.
[17] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2020). Generative adversarial networks. Communications of the ACM, 63(11), 139-144.
[18] Fang, W., Zhang, F., Sheng, V. S., & Ding, Y. (2018). A Method for Improving CNN-Based Image Recognition Using DCGAN. Computers, Materials & Continua, 57(1).
[19] Rožanec, J. M., Zajec, P., Theodoropoulos, S., Koehorst, E., Fortuna, B., & Mladenić, D. (2023). Synthetic data augmentation using GAN for improved automated visual inspection. Ifac-Papersonline, 56(2), 11094-11099.

[20] Liu, J., Gu, C., Wang, J., Youn, G., & Kim, J. U. (2019). Multi-scale multi-class conditional generative adversarial network for handwritten character generation. The Journal of Supercomputing, 75, 1922-1940.
[21] Alafif, T., Alharbi, R., Almajnooni, N., Albishry, M., Alotaibi, A., Alsaadi, F., ... & Sabban, S. (2022). GEAD: generating and evaluating handwritten Eastern Arabic digits using generative adversarial networks. International Journal of Information Technology, 1-9.
[22] Gurumurthy, S., Kiran Sarvadevabhatla, R., & Venkatesh Babu, R. (2017). Deligan: Generative adversarial networks for diverse and limited data. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 166-174).
[23] Franc, D., Hamplová, A., & Svojše, O. (2023). Augmenting Historical Alphabet Datasets Using. Data Science and Algorithms in Systems: Proceedings of 6th Computational Methods in Systems and Software 2022, Vol. 2, 597, 132.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)