

Decipherment of script : Using synthetic data generator's

Tiramdasu Vyshnavi *

Jatinder Kaur †

Department of Mathematics

Chandigarh University

Mohali

Punjab

India

Abstract

This paper is created to give a brief idea about the generative adversarial networks and the handwritten digits recognition using the synthetic data generators with the MNIST data. In this paper Tensorflow is used along with GAN's to recognise the handwritten digits. The 2014 introduction of Generative Adversarial Networks (GANs) by Ian J. Goodfellow revolutionized deep learning and artificial intelligence. A subclass of generative models known as GANs uses a dual neural network structure—a generator and a discriminator—to transform unsupervised learning. GANs create varied, real data samples, influencing the design, entertainment, and healthcare sectors. They are employed in a variety of fields, notably text-to-image translation and image processing. Deep learning relies heavily on neural networks, which are also used in pattern analysis and picture identification. The generative models that are discussed here produce data that is similar to the source datasets and is categorized using density estimation. One popular kind of GANs is made up of a discriminator that separates created data from real data and a generator that produces synthetic data[5][9]. Their adversarial training improves the discriminator's ability to distinguish between actual and false samples while honing the generator's capacity to generate realistic data. Neural network activation factors introduce non-linearity; their applicability varies throughout models and tasks. An open-source package called TensorFlow makes GANs and other AI tasks easier[3][15]. Using the MNIST dataset, an experiment demonstrating handwritten digit recognition using TensorFlow and the Rectified Linear Unit (ReLU) activation function is presented. After several epochs, the results show a significant improvement, demonstrating the effectiveness of GANs in picture identification.

Subject Classification: G06N 3/04.

Keywords: Neural networks, CNN, Perceptron, Generative model, GAN, Activation function.

* E-mail: vyshnavivyshu2563@gmail.com (Corresponding Author)

† E-mail: jatindere1486@cumail.in

1. Introduction

Generative Adversarial Networks also refers to as GAN. Generative Adversarial Networks was first discovered and introduced by the Ian J. Goodfellow in the year 2014. GAN is a kind of artificial neural network. GAN's are a Deep Learning based generative algorithm and moreover it is used for unsupervised learning that is Generative Adversarial Networks works in an unlabelled data which is the form of photos, videos, voice recordings etc and many more type of data can be used in unsupervised learning for GAN[3][5]. A family of deep learning models called generative adversarial nets (GANs) and it can be used to create data samples that are realistic and varied. GANs operate by training a generator and a discriminator, two conflicting neural nets. As the discriminator gains the ability to distinguish between authentic and fraudulent data samples, the generator picks up the skill of creating new data samples. Images, films, text, and music have all been produced using GANs in addition to many other types of data. GANs are still being developed, yet they have the power to completely change a variety of sectors. GANs can be used, for instance, to come up with fresh ideas for designs and goods, produce lifelike special effects for films and video games, and invent novel medical procedures [2]. Along with GAN's Tensorflow is used in a handwritten digit's recognition. An open-source software library for artificial intelligence and machine learning referred to as Tensorflow.

2. Neural Networks

In Artificial Intelligence (AI) a Neural Network is a machine learning (ML) model and the neural network is most popularly and widely used in deep learning (DL). As one subset of machine learning is called deep learning. And artificial intelligence is superset of machine learning. And it is also a particular kind of deep learning. Neural networks can be composed of artificial neurons. And it works by passing the data from one layer to another layer[3][8][12]. Neural networks can be used efficiently in many fields such as image recognition, pattern recognition, product maintenance etc. it is also called as Artificial Neural Networks (ANN). The neural network has the different kinds of layers. That are shown in Figure 1:

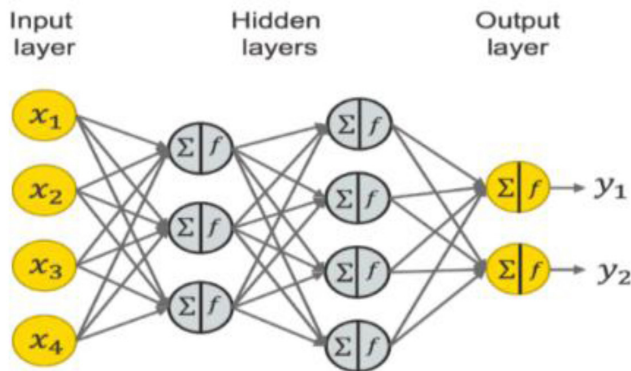


Figure 1

Architecture of Neural Networks

A neural network is a structure made up of three main layers:

- Input Layer
- Hidden Layers
- Output Layer

The data is fed into the input layer then the data sent to the hidden layer. The hidden layer processes the data and generates the output value.

3. Generative Model:

Now-a-days generative models are full in a boom. Generative models are a kind of machine learning model. Generative model is an unsupervised learning technique. The data distribution is part of a generative model, which also provides the likelihood of a certain example. Since generative models may give a probability to a sequence of words, they are frequently considerably simpler than GANs and are used, for instance, to predict the next word in a sequence [3][12]. While there may be some differences or noise in the generated data, it may still be similar to the original data set. A model that can create a new image of a horse that has never existed but nevertheless seems almost genuine might be created, for instance, using a data set of horse photographs.[3][4][6] The generative models are classified into different types. The types of Generative model's are discussed below along with the Figure 2.

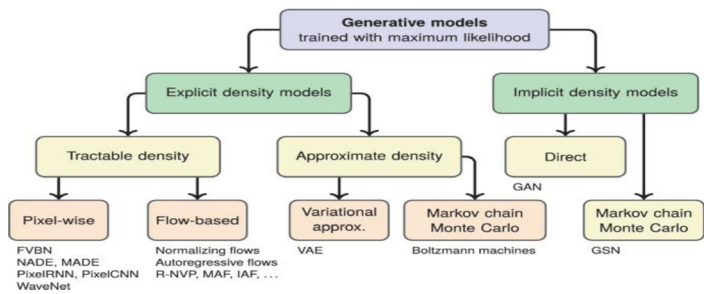


Figure 2
Taxonomy of Generative Model

The figure demonstrates the different generative model types according to the density estimates for those models, and whether or not actual estimation of density is being dealt with directly.

4. Activation Function

By computing the weighted sum and then applying bias, the activation function determines if a neuron needs to be stimulated or not. Adding non-linearity to a neuron’s output is the aim of the activation function. We have a plethora of activation functions. The use of activation function depends on various models designs and workloads may benefit from different activation functions.[8][9][10][13][14]. Experimentation is the best technique to figure out which activation function to utilize. To choose which activation function best meets our demands, try a variety of them and access how well they work for a particular issue.

The equations and ranges of the activation functions given table 1:

Table 1
Equations and ranges of different Activation functions

Activation Functions		
Function	Equation	Range
ReLU	$f(x) = \max(0, x)$	0 to infinity
Sigmoid	$f(x) = 1 / (1 + e^{-(x)})$	0 to 1
Tanh	$f(x) = (e^{x} - e^{-x}) / (e^{x} + e^{-x})$	-1 to 1
Leaky ReLU	$f(x) = \max(0.01x, x)$	Negative infinity to infinity
Softmax	$f(x_{-j}) = e^{x_{-j}} / \text{sum}(e^{x_{-j}})$	0 to 1 (sum of probabilities = 1)

5. Generative Adversarial Networks

Generative Adversarial Networks also refers to be a GAN's. It is a kind of generative model. GAN's are a Deep Learning framework that make to use a pair of neural networks to generate a new and artificial data. It is first initiate by Ian J. Goodfellow in the year 2014. The objective of GAN's is to build a data so that it features a data distribution. GAN's operate unsupervised learning tasks in machine learning. GAN's are commonly used in different kinds of tasks such as anomaly detection, generating images, training a generative model, Translation from text to image and image to image etc.

5.1 Modules of Generative Adversarial Network's

Generative Adversarial Networks are categorized into two parts:
They are: The **GENERATOR** and The **DISCRIMINATOR**

i. Generator:

Generator is one of the two parts of a GAN. Generator is a convolutional Neural Network (CNN). The generator which learns from the data to generate the new data. That is the generator generates fake data by taking the feedback from another part of the GAN discriminator. The generator used backpropagation technique to train the data. In generator we use loss function measures how the generator is misleading a discriminator. Later then the generator used gradient descent method to keep the loss function. [10][11][14]

- The generator chooses the random noisy data and that random data is a sample data that it is taken from a simple distribution, such as Gaussian distribution.
- The output of the generator generates a synthetic sample data.
- At the time of training the motto of the generator is to reduce the inconsistency between the authentic data and the unauthentic data. [13][15]
- Generator is used to trained to reduce the adversarial loss function.
- Thereafter the training the generator then generate the new sample data that corresponded to the training data.

ii. Discriminator-

Discriminator is a deep convolutional neural network. The work of the discriminator is to learn from the data to distinguish between the real data and the fake data. That is the discriminator proves that the data is provided by generator is a fake data and it shows the difference among the real data and the data which is created by a generator.

- During the training period, the main motto of the discriminator is to get better at finding the breach between the real and the fake data.[14][15][7] The discriminator focuses to specify a high probability to an authentic data and perhaps some phony data.
- In the training process the discriminator and the generator are trained in an adversarial manner. First and foremost, the discriminator is being trained by the samples of real data. After that it is trained with the generated fake data. It helps the discriminator to enhance its proficiency in finding the difference between the fake and the real data[3][7][2][11].

We calculate the objective function also known as loss function to check whether the generator and the discriminator are able to compete with each other or not to their goals.

6. Objective Function (Loss Function)

When training GANs, the generator and discriminator compete in the minimax formula scenario to maximize different goals.[1][2] The definition of the loss functions is as follows:

6.1 Discriminator Loss (Binary Cross-Entropy Loss)

$$LD = -N \log \sum_{i=1}^N [y_{\text{real}} \cdot \log(D(x_i)) + (1 - y_{\text{fake}}) \cdot \log(1 - D(G(z_i)))]$$

Here, N is the batch size, y_{real} is the target label for real data (usually 1), and y_{fake} is the target label for fake data (usually 0).

6.2 Generator Loss (Binary Cross-Entropy Loss)

$$LG = -N \log \sum_{i=1}^N \log(D(G(z_i)))$$

While the discriminator strives to minimize L_D , the generator attempts to minimize L_G . These competing goals result in a fair competition where

the discriminator becomes better at telling actual data apart from phony data while the generator gets better at producing realistic data.

7. Tensorflow

An open-source software package called TensorFlow is used to compute numbers using data flow graphs. Artificial intelligence and machine learning are two uses for it. While TensorFlow is applicable to many other tasks, its primary focus is on deep neural network training and inference [1][2][9]. Researchers and engineers utilize it to create and implement machine learning solutions. The developer and user community for TensorFlow is sizable and vibrant. Google also offers training materials, tutorials, and documentation in support of it.

8. Experiment Outcomes And Conclusion

In this paper the experiment is conducted on a **Handwritten Recognition**. We used the TensorFlow which is a open source python library and Rectified Linear Unit (ReLU) activation function is control the layers in the Generative adversarial Networks (GAN's).

The reason behind the usage of ReLU activation function in this experiment is that the ReLU function is that it introduces the non-linearity. As the GAN model is complex it requires back propagation which is possible by non-linearity. When the handwritten image is processes the image it activates Positive pixel values (ink intensity) activate ReLU neurons. Negative pixel values (background) result in zero activation.

For the experiment MNIST dataset is used. The MNIST dataset is widely adopted for training and testing in machine learning because of its straightforwardness, availability, and standardized structure. It consists of 60,000 training images and 10,000 testing images of handwritten digits, making it a valuable benchmark for assessing and contrasting different algorithms and models, particularly in tasks related to computer vision.

To Evaluate the Performance of the experiment the Accuracy and Recall performance metrics are used.

Precision assesses the accuracy of positive predictions by determining the proportion of true positives among all instances classified as positive. It reflects the model's reliability in correctly identifying positive outcomes. Recall evaluates the comprehensiveness of positive predictions, indicating the proportion of true positives identified among all actual positive instances.

The before and after results of the experiment are shown below in following figure 3 and figure 4:

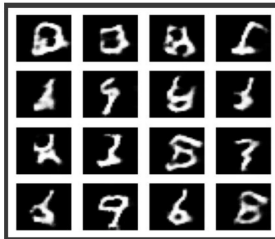


Figure 3

**Handwritten digits before starting
the experiment**

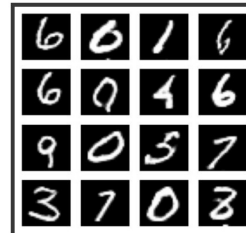


Figure 4

**Handwritten digits in the
experiment after 30 epochs**

9. Future Potential

In future studies, it is possible to conduct experiments using a variety of datasets, such as alphabetical or word datasets, and to explore different activation functions and packages like Keras. This diversified approach allows for comprehensive testing across various data types, providing insights into neural network performance under different conditions. By experimenting with diverse datasets and software tools, researchers can gain a deeper understanding of neural network behavior and effectiveness. Additionally, exploring different activation functions and packages offers opportunities to optimize models for specific tasks or domains, enhancing their adaptability and robustness.

References

- [1] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 53–65 (2018).
- [2] X. Guo and L. Zhao, "A systematic survey on deep generative models for graph generation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 5370–5390 (2022).
- [3] G. T. V. Eswar and P. K. Singh, "Understanding taxonomy of generative models," *Lampyrid: J. Bioluminescent Beetle Res.*, vol. 13, pp. 598–604 (2023).

- [4] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 4401–4410 (2019).
- [5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, et al., "Generative adversarial nets," *Adv. Neural Inf. Process. Syst.*, vol. 27 (2014).
- [6] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Int. Conf. Mach. Learn.*, pp. 7354–7363, PMLR, May(2019).
- [7] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, et al., "ESRGAN: Enhanced super-resolution generative adversarial networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops*, pp. 0–0 (2018).
- [8] A. Jabbar, X. Li, and B. Omar, "A survey on generative adversarial networks: Variants, applications, and training," *ACM Comput. Surv.*, vol. 54, no. 8, pp. 1–49 (2021).
- [9] A. Sajeeda and B. M. Hossain, "Exploring generative adversarial networks and adversarial training," *Int. J. Cogn. Comput. Eng.*, vol. 3, pp. 78–89 (2022).
- [10] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434* (2015).
- [11] T. Chavdarova and F. Fleuret, "SGAN: An alternative training of generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 9407–9415 (2018).
- [12] D. Saxena and J. Cao, "Generative adversarial networks (GANs): Challenges, solutions, and future directions," *ACM Comput. Surv.*, vol. 54, no. 3, pp. 1–42(2021).
- [13] T. Chavdarova and F. Fleuret, "SGAN: An alternative training of generative adversarial networks," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 9407–9415 (2018).
- [14] D. Saxena and J. Cao, "Generative adversarial networks (GANs): challenges, solutions, and future directions," *ACM Comput. Surveys (CSUR)*, vol. 54, no. 3, pp. 1–42 (2021).
- [15] Z. Deng, H. Zhang, X. Liang, L. Yang, S. Xu, J. Zhu, and E. P. Xing, "Structured generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 30 (2017).

Received September, 2024