JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Crypto AI: Digital nostalgic art generation using GAN and creation of NFT using Blockchain

Karthik S[1], Anupama A S[2], Deekshith S A[3]

[1,2,3] *Students Department of Computer Science and Engineering, Dr. Ambedkar Institute of Technology, Bengaluru, India*

[1]itskarthik3119@gmail.com [2]anupamaas12@gmail.com [3]deekshithsamarnath@gmail.com

Ms. Lavanya Santhosh[4], Ms. Monisha Dhanraj[5], Diwakar Ganesan[6]

[4]*Assistant Professor, Department of Computer Science and Engineering, Dr. Ambedkar Institute of Technology, Bengaluru, India*

[5]*Founder and Director,* [6]*Managing Director at Frondeur Labs Private Limited, Bengaluru, India*

***Abstract* – In recent years AI has been a dominating force in many industries. So, we propose a project which combines the technologies of deep learning and blockchain where we use Generative Adversarial Networks to generate digital art which can be sold as NFT's. In this paper, we navigate through the process data extraction and processing to training the neural networks to generate art and finally NFT generation of the digitalised art which was extracted from the AI. We hope that this paper shines a light on recognising AI creating its own niche in the world of digital art industry.**

Index Terms – NFT, StyleGAN, Rinkeby Testnet, pixelated art, Ethereum Blockchain.

I. INTRODUCTION

In the modern times as, the technology improved with better computational performance it has become more viable to train neural networks due to which an AI can be trained to learn to anything in this world. Artificial Intelligence can be applied on numerous industries from the field of science to the art industries.

Since its inception, blockchain has been a pretty intriguing concept in and of itself. A few years ago, the idea of decentralization was suggested, and it has since spread like wildfire over the world, luring many developers and enthusiasts from all walks of life to experiment with it. Today, it has been discovered that the idea of decentralization may effectively close many loopholes and has a wide range of applications, from audits to food safety. Blockchain is no less popular, with many people mining it all over the world and developers interested in learning more about this extremely potent technology.

Many nations are currently gradually switching from paper money to digital transactions based on the Unified Payment System. It's interesting to explore the idea of digital money in and of itself. At the moment, the cryptocurrency market is very speculative, and cryptocurrencies are at the center of the storm.

Along with terms like Blockchain and cryptocurrency, there is another unusual and puzzling term called Non-Fungible Token, or NFT. There are several markets [1] for these digital scarce artifacts, and people have quickly discovered how to create and sell digital treasures. Artists, entertainers, athletes, and gamers from a variety of genres have now established positions in the market to sell their NFTs, tokenized digital objects whose alleged authenticity is guaranteed by the reverse traceability of blockchain transactions.

Some of the most well-known NFTs were made by creators using a variety of software tools, including Adobe Photoshop, Illustrator, NFT Creator, and a few mobile apps including SketchArt and Fotor.

Here we create art by training a StyleGAN [3] with a processed dataset of nostalgic art based on the images from Gameboy characters to childhood cartoon shows which is trained on a single NVDIA GPU provided by Google Colab Pro. Once the algorithm is trained the images generated by generator will be sold as NFT's and as it has been created by an AI which makes it our unique selling point.

Once the digital artwork is ready, we install the required dependencies such as Node, Truffle, Ganache UI, Open Zepplin [2], etc. Then we write and test our smart contract on local Blockchain, having ERC721 token standard and mint NFT on local Blockchain, and deploy our contract on Rinkeby testnet which then will become available on Opensea testnet.

II. TECHNICAL TERMS

A. Decentralization and Blockchain

Banks are one of the best examples of a centralized organization, which means that public people do not get to have an opinion on the decisions made by the bank, even though it's the people's money in the bank. On the contrary in Decentralization the transaction is like a public record, each and every transaction that happens is transparent to the public.
Consider there are independent computers called nodes connected to the internet, and a public database that is shared across all these nodes, and any data added to this database must be agreed upon by all the nodes, this public database can be referred to as a Blockchain.

The name blockchain is because, every transaction that happens in the network, will be stored in one of the blocks to be

validated and each transaction will be recorded on these blocks which reference to its parent block.

One of the important features of these blocks is they are immutable, which means the data on the block cannot be changed.

B. *Ethereum Blockchain and ETH*

Ethereum is a decentralized Blockchain network that creates a peer-to-peer setup so that smart contracts can be executed in a very safe setting. One of the most well-known native cryptocurrencies is ether, or ETH. It is a cryptocurrency token created for the Blockchain network that can be used by participating nodes to pay for the computing resources they have consumed. There are accounts in Ethereum with specified ETH balances that can be used for transactions.

There are two types of Ethereum accounts, they are Externally Owned Accounts (EOA) and Contract Accounts. EOA is controlled by users who are provided by a typical private key, from which a public key is derived, and using that a public address is derived. The public address serves as the identifier of a user on the network. The Contract Address on the other hand is controlled by the contract. When a Smart Contract is deployed on the network, it returns the public address of that particular contract account.

It must be noted here that a transaction can only be initiated by EOA and a contract account can only respond to it also creating a contract account costs certain fees known as Gas Fees.

C. *Smart Contract*

A smart contract is utilized as a contract, much like real-world contracts are. A Smart Contract is a piece of software that is stored on Blockchain and activates when certain requirements are satisfied. They are employed to automate agreement execution so that all parties are certain of the result. This saves time by doing away with the necessity of a broker or intermediary.

D. *Non-Fungible Token (NFT)*

If Person A had a $10 bill and traded it for another $10 from Person B, nothing would have changed because the $10 bill is fungible. The Pokemon cards with both A and B, however, are unique and cannot be traded for a unit value, hence each of these cards is non-fungible. Being "non-fungible" merely means that they are distinct.

Non-fungible tokens, or NFTs, are individual cryptographic tokens that exist on a blockchain and cannot be replicated. They have a unique identification number and metadata.

E. *ERC721*

ERC-721, sometimes known as ERC721, is an Ethereum token standard. "Ethereum Request for Comments 721" is the abbreviation for ERC 721. These assets are frequently referred to as "non-fungible tokens," or NFTs.

The ERC-721 will enable an application to make use of this standard Ethereum API for NFTs within a smart contract. The standards guarantee that NFTs may be traced and exchanged, assuring accuracy in the ownership information. The majority of ERC-721 tokens and NFTs represent ownership of both physical and digital assets, making them versatile and dynamic.

F. *Truffle and Ganache*

Truffle is a top-notch programming environment, testing framework, and asset pipeline for blockchains operating on the Ethereum Virtual Machine (EVM) with the aim of streamlining the work of developers. Throughout the entire lifecycle of their projects, Truffle empowers developers regardless of whether they decide to build on Ethereum, Hyperledger, Quorum, or one of the continuously growing lists of additional supported platforms.

An advanced development tool called Ganache is used to manage a local blockchain for Ethereum development. Ganache is useful during the entire development process. One can create, implement, and test your projects and smart contracts on the local chain in a deterministic and secure setting. Ganache is a part of the truffle ecosystem.

G. *Rinkeby*

Before being implemented on the Ethereum mainnet, blockchain testing is primarily done on the Rinkeby testnet. It is an Ethereum mainnet fork.

H. *IPFS*

The InterPlanetary File System (IPFS), a protocol and peer-to-peer network, is used in distributed file systems to store and share data. IPFS uses content-addressing to identify each file in a global namespace shared by all computing devices.

I. *GAN*

Generative Adversarial network is a neural network architecture which combines two neural networks called generator and discriminator which work hand in hand to generate data which can be texts, images and sounds.

J. *StyleGAN*

It is an extension to the GAN architecture that proposes large changes to the generator model, including the use of a mapping network to map points in latent space to an intermediate latent space, the use of the intermediate latent space to control style at each point in the generator model, and the introduction to noise as a source of variation at each point in the generator model [4].

III. DATA PROCESSING

We obtained a total of 87,000 photographs from the internet, which we then hashed to remove duplicates and perform data cleaning, before performing a second round of manual data cleaning to eliminate a few images that we deemed inappropriate as it would act as noise. Our necessary data was nostalgic graphics from video games and cartoons. Now that our dataset had been reduced to 35,000 photos, it was resized to 256x256 pixels to shorten the GAN's training period, and it was then run via a pre-processing script that NVIDIA made available in their GitHub repository. Fig 1 shows the outcomes of this.

Fig. 1. Dataset after pre-processing

IV. Proposed Approach

We use StyleGAN2 ADA to as our AI to generate nostalgic art. This algorithm's neural network architecture is a beautiful piece of technology which combines supervised, unsupervised and reinforcement learning. Unsupervised learning is performed by the generator where it converts random latent variables which is noise into meaningful artwork. Supervised learning is performed by the discriminator where it performs classification on a set of images which is a combination of our input dataset classified as real images and images generated by the generator which will be classifies as fake images, If the image of the generator is classified as real or fake appropriate feedback is backpropagated by the loss function and the generator improves its artwork and this process can be seen as reinforcement learning, A flowchart depicting the working of GAN can be seen below.

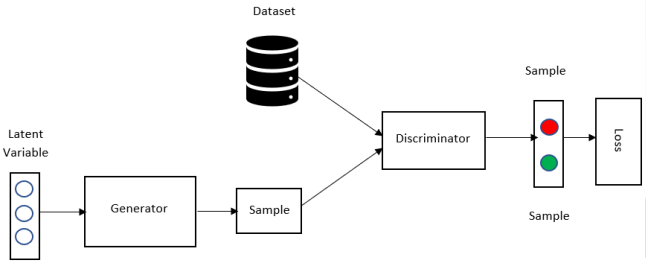

Fig. 2. Conceptual diagram of GAN

Now the pre-processed dataset is fed into the GAN and training is started on the standard hyperparameters and once we saw the generator getting stuck at a local maximum, we decided to tweak the hyperparameters which improved the results of the generator after a few weeks we came to a conclusion that the GAN has achieved convergence we halted the training and extracted the images and our resultant digital art created by an AI was achieved.

Digital art must be translated into NFT once it has been created. We need to construct a smart contract to accomplish that. We call the smart contract, and it responds by creating a new token ID. Each user has a unique Token ID. We now know that the person whose token ID was generated owns the NFT. Before getting started, you'll need to install a few dependencies, including Node.js and a code editor similar to Visual Studio code. Node Package Manager (Npm) is used moving forward. Make a package.json file. Install Truffle next. An initial migration file and migration contract are created as a result. Installing Ganache Desktop UI comes next.

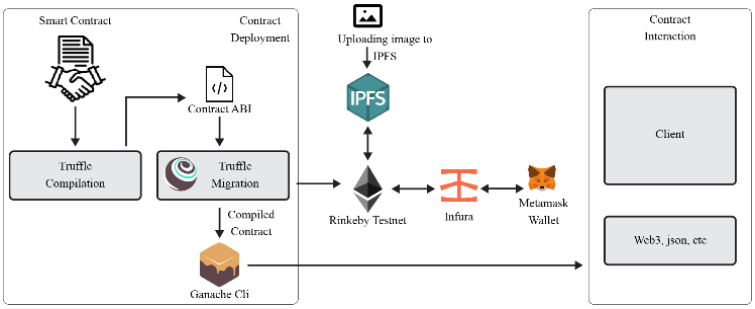

Fig. 3. Smart contract flow diagram

This creates a Local Blockchain with 10 Test Links and 100 ETH each, and also it provides Auto mining features. Once Ganache is up and running, create a workspace and link the project directory. At this stage, it is important to observe that Ganache's host address, port number, and network ID match those of the truffle configuration file.

Next, OpenZeppeline is installed. The openzeppeline functions and procedures can be used to create collectible-specific smart contracts. Using openzeppeline instead of developing smart contracts directly has the benefit of assisting in the implementation of proactive security measures for any web3 Apps.

Due to the high gas costs associated with developing NFT collections directly on Blockchain, our art collectibles must be kept off-chain and are referred to via Toker URI. The Token URI points to a metadata.json file that contains the artwork's description. To compile and migrate the smart contract, we are currently using the Truffle console. A transaction hash, blocks, block number, balance, and other aspects of the transaction are provided once the smart contract is constructed.

Next, we use claim item to mint Tokens. We may examine the transaction to see the hash, nuance, from and to addresses, block number, gas, the gas price we are prepared to pay, and other information. Now, the digital artwork needs to be posted to IPFS. We have used Pinata, which pins the files to IPFS, for this a unique fingerprint known as the content identifier is assigned to this file (CID). We use CID to view the file whenever we need to. The Pinata service API provides us with an API key, and API Secret that must be updated in the .env file along with the Pinata endpoint. Connect to the Rinkeby test network and migrate the smart contracts as the last step.

V. RESULTS

We have successfully trained the GAN until it reached its convergence and have generated images with a low inception distance and can be seen in Fig. 4. Using Ganache and Truffle suit, we were able to successfully compile and deploy the smart contract locally. We obtained the transaction hash and pin size by utilizing the Pinata Service Provider to pin the digital artworks in IPFS. Finally, we have successfully connected to the Rinkeby testnet and generated NFT.

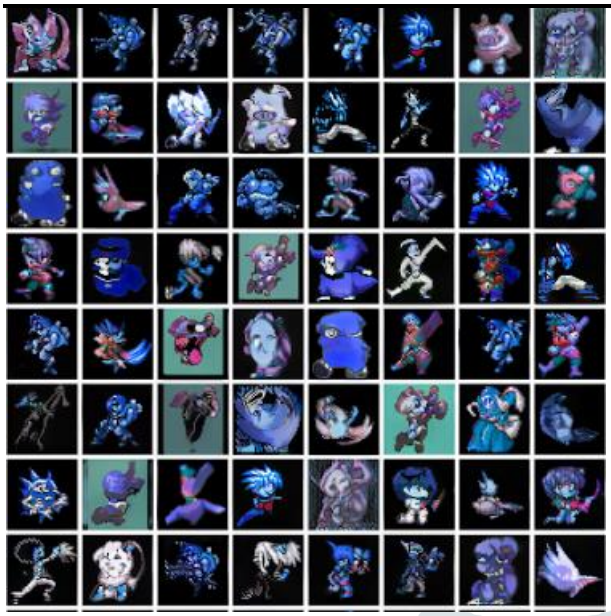

Fig.4. Images generated by StyleGAN

VI. CONCLUSION

We can state that the StyleGAN2's generator network's architecture best suited for our task and demonstrated its ability to produce meaningful art work with a short inception distance. We have also shown how to deploy smart contracts locally and use the Rinkeby Testnet to develop NFT. Additionally, rather than storing the art collection directly on a blockchain, we have instead kept it on IPFS, which is a less expensive option.

However, there is still scope for future improvement of the work flow. Due of limited resources and computing power, our work is now confined to 8-bit pixelated art. If we obtain powerful GPUs, we can alter the GAN and train it to produce high-resolution images or create character animation. This can also be applied to GameFi, a platform for decentralised gaming finance. Additionally, given the availability of ETH currency, the NFTs can also be deployed on the Ethereum Main network and marketplace such as Opensea.

REFERENCES

[1] Saffan Khan, Nishant Agnihotri, "Digital Stack: A NFT Marketplace" *IJCRT*, vol. 10, Issue. 4, April 2022.

[2] Dan Chirtoaca, Joshua Ellul, George Azzopardi, "A framework for creating deployable smart contracts for non-fungible tokens on the Ethereum blockchain", IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS), Aug 2020.

[3] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen and Timo Aila, "Training Generative Adversarial Networks with Limited Data", NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing SystemsDecember 2020 Article No.: 1015Pages 12104–12114

[4] Tero Karras, Samuli Laine and Timo Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks", IEEE Transactions on Pattern Analysis and Machine Intelligence Dec. 2021, pp. 4217-4228, vol. 43. DOI Bookmark: 10.1109/TPAMI.2020.2970919