Jurnal Teknik Informatika (JUTIF)
Vol. 3, No. 3, Juni 2022, hlm. 707-715

DOI: https://doi.org/10.20884/1.jutif.2022.3.3.262
p-ISSN: 2723-3863
e-ISSN: 2723-3871

IMPLEMENTATION OF GENERATIVE ADVERSARIAL NETWORKS FOR CREATING DIGITAL ARTWORK IN THE FORM OF ABSTRACT IMAGES

Eric Secada Purba[*1], Hendry[2]

[1,2]Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana,Indonesia
Email: [1]ericprince913@gmail.com, [2]hendry@uksw.edu

(Naskah masuk: 04 April 2022, Revisi : 07 April 2022, diterbitkan: 28 Juni 2022)

Abstract

Abstract painting always has its own place for the fans, the irregular shape in it, and the emotions depicted in the painting, make many people amazed to see it. The success of this abstract image sparked the idea of being able to create an abstract image using Deep Learning Technology. Generative Adversarial Networks (GANs) is one of the Deep Learning technologies that can create it. With the GANs method which has Generator and Discriminator functions in it, it is possible for someone to be able to create it. The generator functions to generate new data through training the data(train), and the Discriminator functions to determine whether the new data is fake or not data through training (train) comparing the generator results with the original data. These two functions are used to create abstract images. Abstract images were obtained through training in 1369 paintings of nature, landscapes, and flowers. The images are trained by comparing the number of epochs used and the results of the abstract images generated from each epoch. The epoch will be divided into three parts, namely the first training using 10 epochs, the second training using 100 epochs, and the third training using 1000 epochs. In this journal, we will compare the results of the three trainings and reach a conclusion which training produces the best abstract image according to the author. From the training, 1000 epoch training was obtained which produces good abstract images.

Keywords: *abstract image, Generative Adversarial Networks, GANs, Generator, Discriminator*.

PENERAPAN GENERATIVE ADVERSARIAL NETWORKS DALAM PEMBUATAN KARYA SENI DIGITAL BERUPA GAMBAR ABSTRAK

Abstrak

Karya seni lukis abstrak selalu memiliki tempat tersendiri bagi penikmatnya, bentuk yang tidak beraturan didalamnya, serta emosi yang digambarkan di dalam lukisan tersebut, membuat banyak orang terpesona melihatnya. Kesuksesan gambar abstrak inilah yang mencetuskan ide untuk dapat membuat suatu gambar abstrak menggunakan Teknologi *Deep Learning*. *Generative Adversarial Networks* (GANs) adalah salah satu dari teknologi *Deep Learning* yang dapat menciptakannya. Dengan metode GANs yang memiliki fungsi Generator dan Diskriminator di dalamnya memungkinkan bagi seseorang untuk dapat menciptakannya. Generator berfungsi untuk menghasilkan suatu data baru melalui pelatihan (*train*) yang dilakukan, serta Diskriminator berfungsi untuk menentukan apakah data baru tersebut merupakan data palsu atau asli melalui pelatihan (*train*) yang membandingkan antara hasil Generator dengan data asli. Kedua fungsi tersebutlah yang digunakan untuk menciptakan gambar abstrak. Gambar abstrak diperoleh melalui pelatihan 1369 lukisan alam, pemandangan, maupun bunga. Gambar-gambar tersebut dilatih dengan membandingkan jumlah epoch yang digunakan dan hasil gambar abstrak yang dihasilkan dari setiap Pelatihan. Epoch akan dibagi menjadi tiga bagian yaitu pelatihan pertama menggunakan 10 epoch, pelatihan kedua menggunakan 100 epoch, dan pelatihan ketiga menggunakan 1000 epoch. Riset ini akan membandingkan hasil dari ketiga pelatihan tersebut dan mencapai kesimpulan pelatihan manakah yang menghasilkan gambar abstrak yang terbaik menurut penulis. Dari ketiga Pelatihan tersebut, diperoleh lah pelatihan dengan 1000 epoch yang menghasilkan gambar abstrak dengan baik.

Kata kunci: *gambar Abstrak, Generative Adversarial Networks, GANs, Generator, Discriminator*.

1. PENDAHULUAN

Lukisan merupakan hasil dari karya seni Lukis yang selalu memiliki nilai lebih dimata banyak

orang. Pembuatan satu buah karya seni dimulai dengan memikirkan sebuah konsep, mencari ide, stimulasi dan kontemplasi, sehingga untuk menghasilkan suatu karya seni Lukis membutuhkan waktu yang tidak sedikit. Karya seni lukis tidak hanya tergantung pada imajinasi atau ekspresi dari si pembuatnya, tetapi pelukis juga harus memiliki berbagai jenis unsur dan prinsip seni rupa, serta gaya atau aliran, kemudian teknik dan media yang digunakan [1]. Dalam riset ini akan membahas salah satu gaya atau aliran karya seni Lukis yaitu Karya Seni Lukis Abstrak.

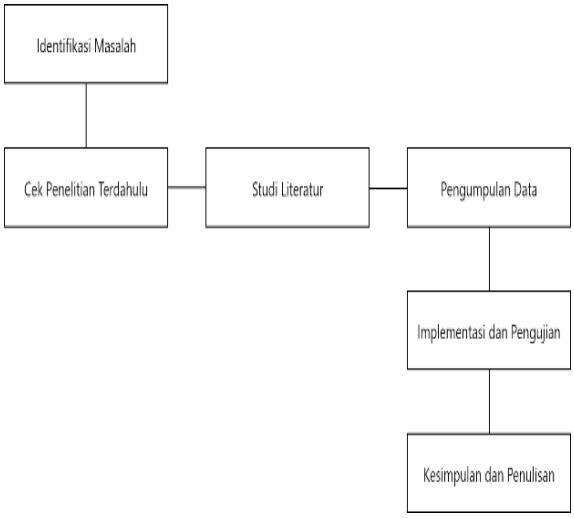
Seni Lukis abstrak biasanya dikenal memiliki bentuk yang tidak biasa karena dalam penciptaannya pelukis tidak langsung merujuk terhadap bentuk aslinya tetapi diubah menjadi bentuk yang lain sesuai dengan imajinasi pelukis [2], sehingga karya seni Lukis abstrak ini tidak dapat dinilai hanya dengan mata telanjang saja. Diperlukan penglihatan atau pendalaman yang khusus untuk mengetahui nilai-nilai yang terkandung di dalamnya dan untuk memahami ekspresi pelukis dalam pembuatannya. Keabstrakan karya seni inilah yang melatar-belakangi penulisan artikel ini, dengan tidak adanya aturan yang berlaku di dalam pembuatannya, membuat suatu pemikiran apakah seseorang dapat membuat karya seni lukis abstrak ini tanpa kemampuan khusus dan biaya yang besar untuk membeli bahan serta peralatan yang dibutuhkan. Pembuatan suatu karya seni lukis abstrak dibutuhkan ide dan nilai tertentu, dimana tidak semua orang memilikinya.

Riset ini bertujuan untuk memanfaatkan teknologi *Deep Learning* untuk menciptakan suatu karya seni lukis abstrak tanpa media dan peralatan yang banyak serta ide yang cemerlang dalam penciptaannya. Memang hasil dari pembuatan karya seni lukis ini akan berupa karya seni digital, tetapi seperti yang diketahui bahwa saat ini terdapat tren jual-beli karya *Non Fungible Token* (NFT) yang sangat mudah untuk penjualannya maupun hanya sekedar dipamerkan. Ketertarikan banyak orang dalam *Non Fungible Token* (NFT) dapat menjadi peluang untuk memamerkan hasil dari pembuatan karya seni lukis abstrak ini, serta dapat memperoleh keuntungan jika berhasil menjualnya, dan memperkenalkan kepada orang banyak bahwa hasil dari karya seni lukis yang mereka lihat dibuat dengan teknologi Deep Learning.

Pembelajaran Struktural Mendalam (*Deep Learning*) saat ini sangatlah populer, salah satu Algoritma yang digunakan dalam pembuatan karya seni lukis abstrak ini adalah *Generative Adversarial Networks* (GAN's). GAN's dapat digunakan untuk menghasilkan foto realistis seperti pemandangan gunung maupun danau [3], membuat suatu salinan gambar tampak seperti aslinya [4], menghasilkan gambar beresolusi tinggi dari gambar yang memiliki resolusi rendah [5], mengklasifikasikan objek untuk keperluan medis [6], bahkan dapat digunakan untuk mendeteksi malware [7]. Pada Riset ini akan membahas mengenai bagaimana cara penggunaan *Generative Adversarial Networks* (GAN's) dalam menghasilkan suatu Karya seni lukis abstrak. Diharapkan dengan penggunaan Algoritma ini dapat menghasilkan suatu karya seni lukis abstrak yang memiliki nilai yang terkandung di dalamnya, dan juga orang lain dapat menyukai hasilnya.

2. METODE PENELITIAN

Metode yang digunakan dalam Penelitian ini adalah penelitian eksperimental, dimana akan melakukan uji coba terhadap Kecerdasan Buatan (artificial intelligence) untuk menciptakan gambar abstrak, kemudian diperoleh suatu algoritma dari Deep Learning yaitu Neural Network. Terdapat banyak sekali jenis dari Neural Network dan dipilihlah Generative Adversarial Networks [8] [9] untuk mengerjakan penelitian ini. Alur pengerjaan penelitian ini dapat dilihat pada Gambar 1.



Gambar 1. Tahapan Penelitian

Penelitian dimulai dengan identifikasi masalah untuk membuat pedoman yang akan digunakan untuk menyelesaikan penelitian ini. Setelah itu akan dilakukan pengecekan terhadap penelitian terdahulu untuk melihat keterkaitan masalah yang sedang dikerjakan dengan penelitian yang pernah dikerjakan sebelumnya. Setelah itu akan dilakukan studi literatur dan pengumpulan data, dan diimplementasikanlah literatur yang telah dipelajari dengan data yang diperoleh supaya dapat diuji. Setelah pengujian dilakukan maka dihasilkan kesimpulan serta penulisan jurnal ini.

2.1. Penelitian Terdahulu

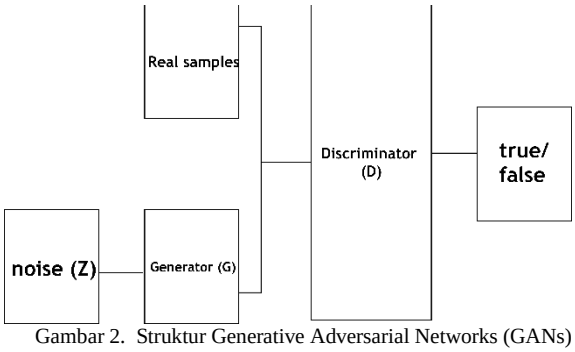
Penelitian terdahulu yang berjudul "Klasifikasi Gambar Batik Dengan CNN" dimana algoritma CNN termasuk salah satu dari arsitektur *Deep Learning*. Metode *Convolutional Neural Network* (CNN) merupakan salah satu algoritma pengklasifikasian gambar dengan operasi

konvolusionnal yang menggabungkan beberapa lapisan pemrosesan, menggunakan beberapa elemen yang beroperasi secara paralel dan terinspirasi oleh sistem saraf biologis [10]. Tujuan penelitian ini mengukur seberapa efektif algoritma CNN terutama dengan Bahasa Pemrograman *Python* untuk melakukan *Image Classification* dan tentunya hal ini bermanfaat karena sekarang ini teknologi kecerdasan buatan untuk mengenali objek sangat dibutuhkan dalam kehidupan sehari - sehari guna mempermudah kehidupan. Algoritma CNN disini bertugas untuk mengklasifikasikan gambar batik melalui 5 jenis batik yang ada di *dataset*, diantaranya batik Pekalongan, batik Semarang, batik Solo, batik Sragen, dan batik Yogyakarta. *Convolutional Neural Network* (CNN) biasanya digunakan untuk mengenali maupun mendeteksi sebuah objek pada data image [11]. CNN terdiri dari neuron yang memiliki *weight*, *bias* dan *activation function*.

Terdapat penelitian terdahulu yang berjudul "Pembuatan karya seni digital dengan metode *Style Transfer* menggunakan Algoritma CNN" Penelitian ini bertujuan untuk menciptakan suatu karya seni gambar dengan mengubah gambar pribadi (foto maupun lukisan) menjadi gambar yang memiliki gaya lukis seniman terkenal. Sebagai contoh seorang fotografer memiliki potret gambar gunung lalu fotografer tersebut ingin hasil gambarnya memiliki gaya/*style* lukisan vincent van gogh maka ia menggunakan Algoritma CNN untuk melakukan "*transfer Style*" ke dalam gambar yang dia miliki [12]. Bisa dikatakan bahwa fungsi *Convolutional Neural Network* (CNN) disini adalah menggabungkan dua buah gambar untuk menghasilkan gambar baru. CNN dapat menghasilkan suatu karya seni digital yang baru tanpa membandingkan bahwa karya seni tersebut Asli atau Palsu, sehingga penggunaan algoritma CNN dalam pembuatan karya seni digital ini kurang baik hasilnya. Jika karya seni ini ditimbang menggunakan algoritma *Generative Adversarial Network* (GANs) maka diskriminator akan sangat mudah mengatakan bahwa karya seni tersebut bernilai Palsu.

2.2. Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GAN's) diperkenalkan pertama kali oleh Ian J. Goodfellow bersama tujuh orang lainnya pada tahun 2014 melalui Jurnal mereka yang berjudul *Generative Adversarial Nets* [8]. Penggunaan *Generative Adversarial Networks* (GAN's) sangat berbeda dengan Algoritma *Deep Learning* lainnya, dimana Algoritma lainnya memanfaatkan penuh terhadap *dataset* yang ada dimana *dataset* tersebut dilatih untuk memaksimalkan hasil akurasi dari setiap pengujian yang dilakukan, semakin banyak *dataset* yang dimiliki maka semakin baik hasil yang didapatkan . *Generative Adversarial Networks (GAN's)* memiliki pendekatan yang berbeda dengan algoritma lainnya, algoritma ini menggunakan dua jaringan syaraf buatan (*neural network*) untuk menyelesaikan permasalahan yang ada. Kedua jaringan syaraf buatan tersebut adalah Generator yang berfungsi untuk mengambil sampel data dari *dataset*, dan Diskriminator yang berfungsi untuk mengklasifikasikan bahwa sampel data itu bernilai asli atau palsu [9], [13], [14]. Dengan adanya kedua *neural network* tersebut, maka akan dihasilkan suatu data yang baru yang sangat menyerupai data inputan-nya.



Gambar 2. Struktur Generative Adversarial Networks (GANs)

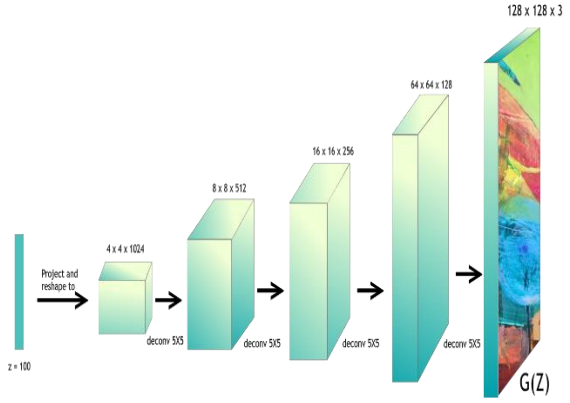
Struktur *Generative Adversarial Networks* (GANs) yang terdapat pada Gambar 2 menunjukkan bahwa fungsi dipisahkan menjadi Generator dan Diskriminator . Untuk mempermudah penjelasan disini Generator diinisialisasikan dengan fungsi G dan Diskriminator dengan fungsi D, dengan X sebagai data asli yang ada di dalam *Dataset*, terakhir terdapat Z sebagai variabel acak. Setiap hasil dari G akan diwakili G(z) sebagai sampel baru hasil dari Generator. Data X akan dimasukkan ke dalam Diskriminator D untuk memberikan label terhadap G(z). jika Diskriminator D menghasilkan label 1(*True*) berarti G(z) dianggap mirip dengan X. Jika Discriminator D menghasilkan label 0(*False*) berarti G(z) dianggap tidak mirip dengan X. untuk hasil dari Generator G sangat baik, Diskriminator D harus mengklasifikasikan G(z) dengan label 0. tujuan G adalah untuk membuat kinerja data yang dihasilkan G(z) pada D (D(G(z)) konsisten dengan kinerja *Real Samples* X pada D.

2.3. Deep Convolutional Generative Adversarial Networks (DCGANs)

Untuk meningkatkan Performa dari model yang akan dibangun, disini ditambahkan Deep Convolutional Generative Adversarial Networks (DCGANs) untuk menggunakan *Convolutinal Layers* pada Diskriminator Nya dan *transpose convolution layers* pada Generator Nya. Sebelum membahas lebih lanjut, Diperlukan pengertian mengenai apa itu Deep Convolutional Generative Adversarial Networks (DCGANs). DCGAN pertama kali diperkenalkan pada tahun 2016 melalui jurnal yang berjudul "Unsupervised Representation Learning with Deep Convolutional Networks" karya Alec Radford dan Luke Metz [15]. Dalam jurnal

tersebut menjelaskan bahwa DCGAN tercipta melalui proses perluasan Generative Adversarial Networks (GANs) dengan menambahkan *Convolution Neural Network* (CNN) kedalamnya . Untuk mengetahui perbedaan dari GANs dan DCGANs dapat dilihat dari arsitektur terbentuknya DCGAN[15] yaitu :

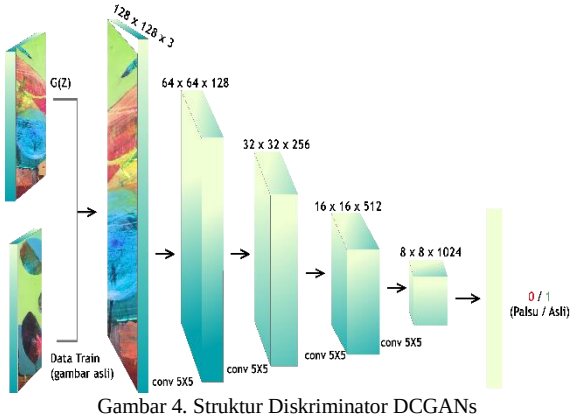
1. Mengganti *Polling Layer* pada Diskriminator dengan *strided convolutions* dan di *fractional-strided convolutions* pada Generator nya
2. Pada Diskriminator dan Generator menggunakan *Batch Normalization* yang berfungsi untuk meningkatkan kecepatan jaringan syaraf tiruan dan supaya lebih stabil
3. *fully connected hidden layers* dihapus untuk arsitektur yang lebih mendalam
4. Generator menggunakan aktivasi RelU untuk semua lapisan, tetapi terdapat pengecualian pada *output* yaitu menggunakan TanH
5. Diskriminator menggunakan aktivasi LeakyReLU untuk semua lapisannya.



Gambar 3. Struktur Generator DCGANs

Pada Gambar 3 terdapat struktur dari Generator DCGANs[16]. Misalnya kita punya 100 *random noise* vektor sebagai inputan nya, kemudian kita proyeksikan dan bentuk kembali random *noise vektor* tersebut ke dalam bentuk 4x4x1024. Pada Gambar tersebut, diinginkan untuk mengubah gambar 4x4x1024 menjadi bentuk 128x128x3. Lakukanlah proses sebelumnya sampai memperoleh hasil yang diinginkan, kemudian setiap prosesnya harus mengikuti arsitektur daripada DCGANs, yaitu untuk setiap *Layer* menggunakan *Batch Normalization* dan menggunakan aktivasi RelU kecuali pada G(Z) menggunakan TanH. Seperti itulah cara untuk *Upsample* gambar yang dimiliki.

Diskriminator berfungsi untuk mengetahui sebuah gambar bernilai asli atau palsu [15], [16]. Dengan gambar yang dihasilkan dari Generator, diskriminator akan bekerja untuk mengidentifikasikan gambar tersebut dengan cara mencoba mengembalikan gambar tersebut ke bentuk awal. Seperti terlihat pada Gambar 4 bahwa Diskriminator merupakan kebalikan dari Generator, ditambah dengan gambar asli yang dimiliki di dalam Dataset, maka akan meningkatkan kemampuan Diskriminator untuk mengidentifikasikan suatu gambar bernilai asli atau palsu. Untuk Diskriminator juga menggunakan kaidah arsitektur dari DCGANs yaitu menggunakan Batch Normalization dan menggunakan aktivasi LeakyReLU untuk semua layer.



Gambar 4. Struktur Diskriminator DCGANs

2.4. Fungsi Objektif GANs

Setiap model *Deep Learning* sebuah fungsi harus didefinisikan untuk memperoleh suatu hasil. Pada *Generative Adversarial Networks* memiliki suatu Fungsi Objektif yang digunakan bersamaan dengan model yang dibangun. Disini akan dibangun fungsi objektif dari struktur GANs[14], [17].

$$\underset{G}{Min}\ \underset{D}{Max}\ Ex{\sim}px[logD(x)] + Ez{\sim}px\left[\log\left(1 - D\left(G(z)\right)\right)\right] \quad (1)$$

Fungsi objektif (1) terdiri dari x sebagai *real* data (*Sampel* data) dan z adalah *noise* vektor, G adalah Generator dan D adalah Diskriminator, px adalah distribusi dari x (*real* data) dan pz adalah masing-masing dari *noise* vektor. D(x) merupakan hasil dari *training* Diskriminator,G(z) adalah hasil dari Generator dan D(G(Z)) adalah Hasil dari Discriminator yang memberikan nilai/hasil dari G(z). Diskriminator dilatih untuk memaksimalkan pemberian nilai terhadap hasil dari generator dengan memaksimalkan fungsi log D(x) + log(1 − D(G(z))). Kemudian Generator dilatih untuk dapat menipu Diskriminator dengan meminimalkan fungsi log(1 − D(G(z))). Melalui pelatihan inilah Generator akan berusaha semaksimal mungkin untuk menghasilkan gambar tidak semirip mungkin dengan dataset(*real* data), dan Diskriminator juga akan berusaha untuk mengidentifikasi hasil dari Generator untuk mengetahui gambar tersebut asli atau palsu.

2.5. Loss Function GANs

Pada dasarnya *loss function* digunakan untuk mengetahui performa suatu model yang sedang dikembangkan. Pada riset ini *loss function* digunakan untuk mengembangkan serta mengukur

Generator dan Diskriminator DCGANs untuk mengetahui sejauh apa performa dari kedua model ini. Seperti yang sudah dibahas sebelumnya bahwa Diskriminator bertujuan untuk memaksimalkan fungsi log (1 − D(G(z))) sehingga Diskriminator akan meminimalkan fungsi D(G(z)). Sehingga *loss function* dari Diskriminator adalah:

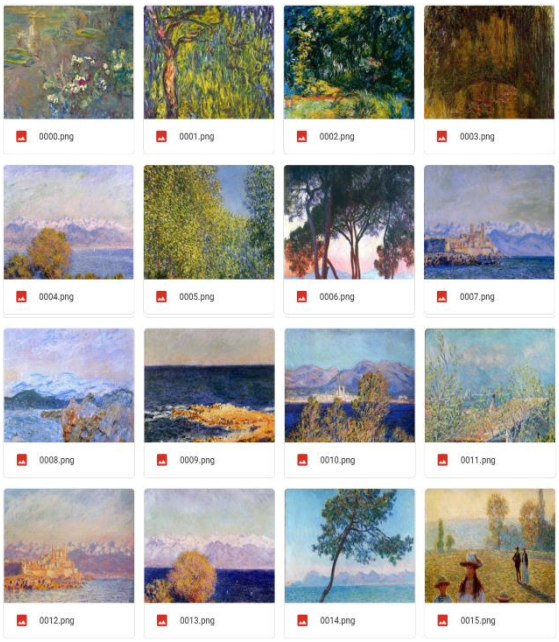
$$L^D = Max\,[\,log\,D(x) + Log(1 - D(G(z)))\,]\quad (2)$$

Dengan fungsi tersebutlah (2) Diskriminator akan memaksimalkan fungsi D(x) selagi meminimalkan fungsi D(G(z)). Untuk itu Generator harus berusaha semaksimal mungkin untuk menipu Diskriminator untuk hasil gambar yang dihasilkan Generator, dengan cara meminimalkan fungsi log (1 − D(G(z))). Sehingga *loss function* dari Generator (3) adalah:

$$L^G = Min\,[\,Log(D(G(z)))\,]\quad (3)$$

Generator tidak memiliki hak atas *dataset* (*real* data) sehingga generator tidak memiliki wewenang/kuasa atas D(x) sehingga Generator tidak dapat mengubahnya. Tetapi Generator dapat memaksimalkan fungsi G(z) untuk menghasilkan gambar yang dapat menipu Diskriminator.

2.6. Dataset



Gambar 5. Dataset Lukisan

Karya seni lukis abstrak adalah cara lain bagi pelukis/seniman untuk mengekspresikan diri mereka dengan cara yang lain dalam berkarya. Jenis seni ini tidak terikat dengan ide atau gambaran yang jelas dimana orang tidak dapat menilainya hanya dengan mata telanjang. Bentuk yang dihasilkan tidak dapat didefinisikan hanya dengan melihat sekilas saja karena bentuk dari karya seni lukis abstrak ini sangat tidak biasa. Penciptaan seni Abstrak dihasilkan melalui "abstraksi" dari alam [18], dimana alam memiliki bentuk dan nilai yang luas untuk dapat dijadikan referensi atau gambaran dalam pembuatan karya seni Abstrak. Pemilihan *dataset* ini juga dilatarbelakangi oleh kemudahan dalam menemukan datasetnya, karena banyak ditemukan di internet dan juga sudah banyak orang yang membuat *dataset* mengenai gambar atau lukisan dari alam. Sehingga setelah dataset diperoleh, *dataset* tersebut di upload ke dalam *google drive* untuk mempermudah pelatihan model yang akan dilakukan, karena pelatihan model dikerjakan melalui *google collaboration*.

Untuk melatih model yang akan dibangun, digunakanlah dataset berupa 1369 lukisan alam seperti pada contoh di gambar 5. *Dataset* ini diperoleh dari situs yang menyediakan *dataset* yaitu Kaggle:https://www.kaggle.com/datasets/varnez/claude-monet-pictorial-works-dataset-wikiart.
Resolusi gambar yang dimiliki adalah 256 x 256.

2.7. Pelatihan Model

Tahapan yang dilakukan untuk menghasilkan sebuah gambar abstrak meliputi beberapa hal yang harus dilakukan. Pertama menerapkan Arsitektur dari *Generative Adversarial Networks* (GANs) dengan menggabungkannya dengan *Neural Network* dari DCGANs untuk memperoleh hasil yang lebih maksimal. Untuk itu pertama harus Menetapkan data train yang akan digunakan, menggunakan Generator dan Diskriminator sesuai dengan kaidah yang telah dijelaskan sebelumnya. Model akan dilatih dengan membandingkan jumlah *Epoch* yang digunakan dan hasil dari setiap pelatihan seperti apa. Setiap *Epoch* akan menggunakan 10 iteration untuk memaksimalkan gambar yang akan dihasilkan. Pelatihan akan membandingkan hasil dari pelatihan model dengan epoch yang berbeda-beda. Model akan dilatih dengan memasukkan jumlah epoch sesuai yang diinginkan, pada riset ini akan menggunakan tiga *epoch* yang berbeda yaitu pelatihan dengan 10,100, dan 1000 *epoch*. Dan kemudian dicari kesimpulan Pelatihan dengan *epoch* manakah yang menghasilkan gambar yang paling paling sesuai dengan hasil yang diharapkan penulis.

3. HASIL DAN PEMBAHASAN

Tabel 1. Pemberian nilai setiap Fungsi

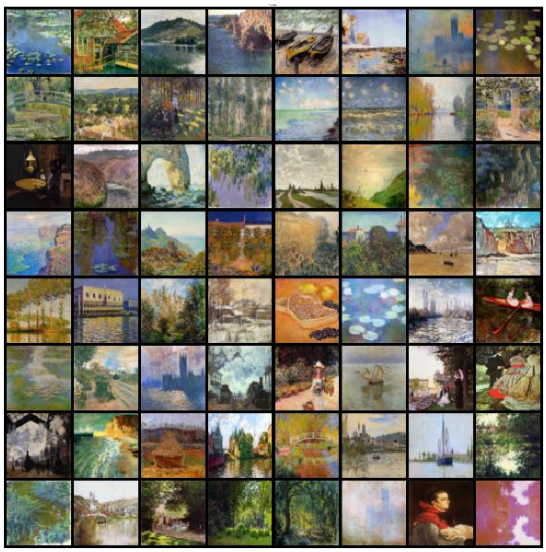
Function/Parameter	Nilai
Dataset (train data)	1369
Ukuran gambar dataset	256 x 256
Ukuran hasil gambar (output)	64 x 64
Noise (Z)	100
Epoch	I = 10
	II = 100
	III = 1000
LeakyRelU	0.2
RelU	0.2
Optimezers	Adam 1.50E-04

Pelatihan dilakukan menggunakan Google Colaboration dengan NVIDIA Tesla K80 GPU

dengan *batch size* gambar sebesar 128. Untuk lebih lengkapnya akan dijabarkan pada tabel 1.

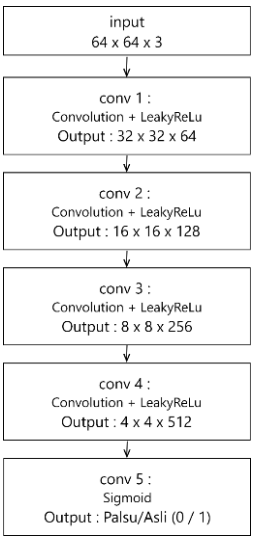
3.1. Training Dataset

Melakukan train terhadap dataset untuk meningkatkan kualitas hasil dari model yang akan dibangun, dataset akan dilatih melalui *transform* data berupa data *train* akan diubah ukurannya (*resize*) menjadi ukuran yang lebih kecil yaitu 128 x 128. Setelah *dataset* di *resize* selanjutnya adalah menggunakan salah satu *package* dari Pytorch [19] yaitu ToTensor yang berfungsi untuk mengubah PIL *image* dengan rentang *pixel* [0, 255] menjadi PyTorch *FloatTensor of shape*(C, H, W) dengan rentang [0.0, 1.0] .Setelah itu akan dilakukan *Normalize* untuk menghasilkan *train data* yang baik dan diperoleh hasil *training dataset* seperti pada gambar 6.



Gambar 6. Training Dataset

3.2. Diskriminator

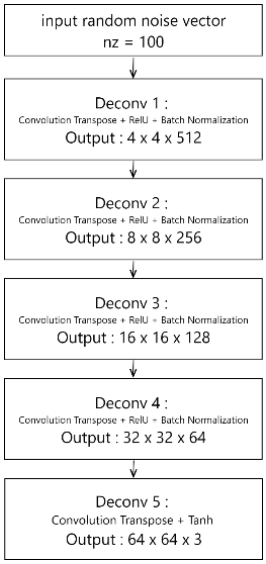


Gambar 7. Tahapan Diskriminator

Menggunakan Diskriminator untuk mengetahui kebenaran atau nilai dari gambar yang dihasilkan dari Generator, apakah gambar tersebut asli atau palsu. Disini Discriminator mengembalikan nilai dari Generator dengan Input hasil dari Generator, yaitu output dimension 64 x 64 x 3. Pada Discriminator tetap harus menggunakan kaidah Arsitektur DCGANs dimana untuk setiap *convolution layer*-nya menggunakan LeakyReLU dan *Batch Normalization* pada setiap layernya kecuali pada outputnya menggunakan *sigmoid* untuk mengetahui apakah gambar tersebut asli atau palsu. Implementasi Diskriminator pada riset ini dapat dilihat pada gambar 7.

3.3. Generator

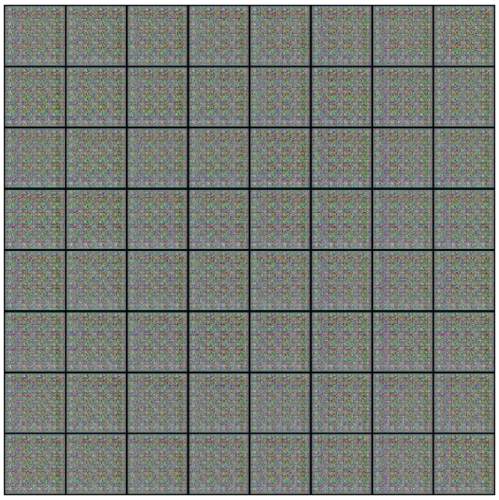
Pada saat Generator dibangun harus sesuai dengan struktur generator yang terdapat pada gambar 3, dan harus mengikuti kaidah arsitektur DCGANs, karena Generatornya akan dibangun dengan Neural Network DCGANs. Pertama akan dilakukan pemberian nilai terhadap Z (*noise*) yaitu 100 kemudian Dilanjutkan ke Deconv 1 dengan output dimension 4 x 4 x 512 dan dilanjutkan sampai Deconv akhir sehingga menghasilkan *output* dimension 3 x 64 x 64, karena *output* yang ingin dihasilkan berupa gambar berukuran 64 x 64. Implementasi Diskriminator pada riset ini dapat dilihat pada gambar 8.



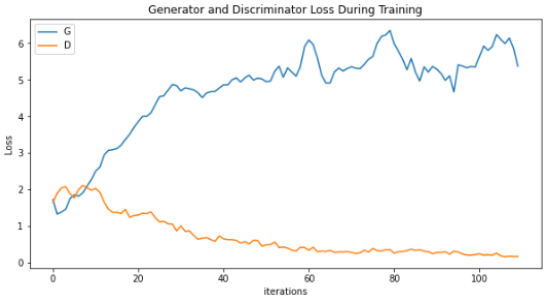
Gambar 8. Tahapan Generator

3.4. Hasil Pelatihan Model

Pelatihan dilakukan dengan menggabungkan *Training dataset* dan mengimplementasikan Generator dan Diskriminator yang akan digunakan pada pelatihan model ini. Setelah itu dilakukan implementasi Fungsi objektif dari *Generative Adversarial Networks* serta menghitung *loss function*-nya. Maka dihasilkan gambar abstrak seperti pada gambar 9, gambar 11, dan gambar 13. Dan grafik dari *loss function* terdapat pada gambar 10, gambar 12, dan gambar 14.

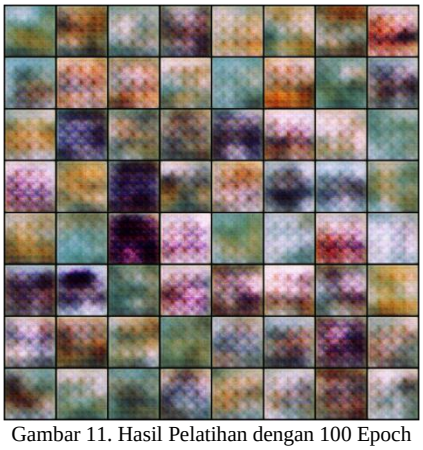


Gambar 9. Hasil Pelatihan dengan 10 Epoch

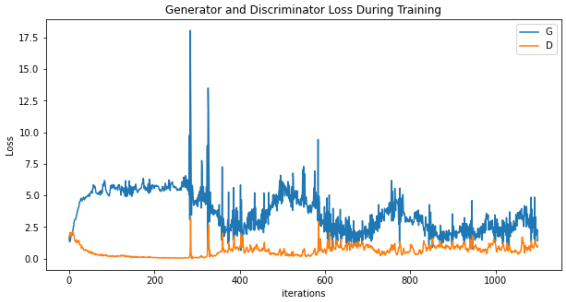


Gambar 10. Grafik Training Generator Loss dan Discriminator Loss

Untuk Pelatihan pertama dengan 10 Epoch diperoleh hasil yang terdapat pada gambar 9 sangat kurang memuaskan jika dilihat, hasil dari setiap gambar tergolong sama. Diskriminator dan generator *Loss* seperti dapat dilihat pada gambar 10 sebenarnya sudah baik karena Generator berhasil menghasilkan gambar yang sangat berbeda dengan *dataset* sehingga Discriminatornya sulit untuk mengenali gambar yang dihasilkan Generator, tetapi untuk gambar yang dihasilkan Generator terlihat kurang baik. Waktu yang dibutuhkan untuk melatih model dengan 10 epoch adalah sekitar 3 menit.

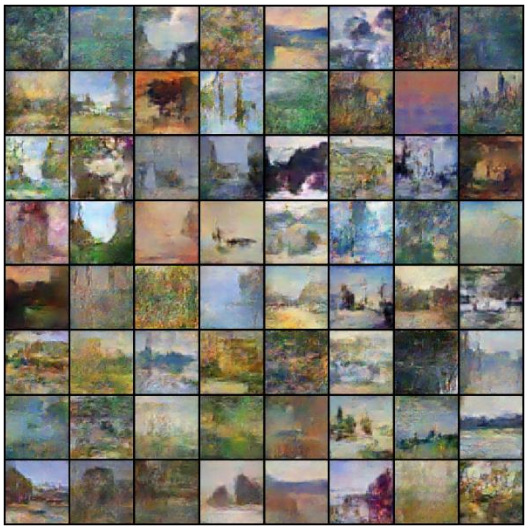


Gambar 11. Hasil Pelatihan dengan 100 Epoch

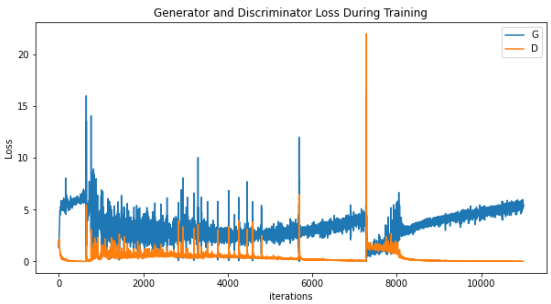


Gambar 12. Grafik Training Generator Loss dan Discriminator Loss

Gambar abstrak yang dihasilkan dari pelatihan kedua dengan 100 *epoch* menunjukkan bahwa hasil gambar memiliki bentuk atau warna yang berbeda tetapi setiap gambar masih terlihat pola yang sama, seperti membentuk belah ketupat atau berlian yang dapat dilihat pada gambar 11. Sehingga pada pelatihan kedua ini masih tergolong belum sepenuhnya selesai sehingga perlu melatih lagi model yang dibangun sampai gambar yang dihasilkan benar-benar berbeda. Untuk hasil dari loss function Diskriminator dan Generator dapat dilihat pada gambar 12 dapat disimpulkan bahwa Generator bekerja dengan sangat baik menghasilkan gambar yang berbeda sehingga Diskriminator kesulitan dalam menilainya. Waktu yang dibutuhkan untuk pelatihan model kedua dengan 100 epoch adalah sekitar 15 menit.



Gambar 13. Hasil Pelatihan dengan 1000 Epoch



Gambar 14. Grafik Training Generator Loss dan Discriminator Loss

Setelah melakukan pelatihan model ketiga dengan 1000 *epoch*, dihasilkan lah gambar abstrak yang dapat dilihat pada gambar 13 dimana memiliki bentuk atau pola yang berbeda di setiap gambarnya. Gambar yang dihasilkan juga lebih jelas dan yang terpenting tetap bersifat Abstrak. Tidak ada gambar yang menyerupai *dataset* (*real image*), semua terlihat abstrak dan sulit dikenali bentuknya. Hasil seperti itulah yang diinginkan melalui Riset ini. Untuk Generator *Loss* berhasil menghasilkan gambar dengan sangat baik yang dapat dilihat pada Gambar 14. Pada Diskriminator *loss* tergolong rendah karena Diskriminator selalu menganggap hasil dari Generator palsu, yang tentunya ini sangat sejalan dengan tujuan dari dibentuknya model pelatihan ini. Dikarenakan tujuan pembuatan Riset ini adalah menghasilkan gambar yang baru melalui GANs dimana jika hasil dari Diskriminator *loss* tinggi, maka generator sangat mirip dengan *dataset* (*real image*) sehingga tidak dapat terbentuk suatu gambar abstrak. Waktu yang dibutuhkan untuk untuk melatih model ini dengan 1000 epoch sekitar 3jam.

4. KESIMPULAN

Berdasarkan hasil dari pelatihan yang dilakukan, gambar yang dihasilkan dapat dikatakan berhasil karena terlihat seperti lukisan abstrak yang dapat dilihat pada pameran atau sekedar di internet. Kecilnya resolusi atau pixel dari gambar yang dihasilkan diakibatkan oleh images size yang didefenisikan diawal hanya sebesar 64 pixel. Hal tersebut disebebkan oleh kurangnya spesifikasi laptop dan coding environtment yang digunakan yaitu Google Colab, sehingga untuk menaikkan pixel gambar menjadi 128 px atau 256 px selalu gagal dilakukan. Penggunaan 1000 epoch dengan 10 iterations setiap 1 epoch sudah sangat baik untuk melatih model yang saat ini digunakan. Karena hasil yang didapatkan memuaskan dan sesuai dengan yang diinginkan.

Untuk meningkatkan pelatihan model ini kedepannya, Peningkatan ukuran conv dan deconv dari 512x512 menjadi 1024x1024 akan sangat berguna untuk menghasilkan gambar yang lebih baik lagi. Untuk meningkatkan nilai dari gambar yang dihasilkan, dimasa yang akan datang kemungkinan akan dilakukan survey kepada orang lain terhadap gambar abstrak yang dibuat oleh algoritma GANs, sehingga hasil yang didapatkan lebih valid dan dapat diterima banyak orang.

DAFTAR PUSTAKA

[1] D. Noventin Maghdalena, Y. Puspita, D. Pendidikan Seni Rupa, and F. Pendidikan Seni dan Desain, "Analisis Estetik Karya Seni Lukis Moel Soenarko yang Bertema Heritage," *ejournal.upi.edu*, vol. 1, no. 2, pp. 1–14, 2019.

[2] S. DAN WACANA Amir Gozali Jurusan Seni Rupa Murni and F. Seni Rupa dan Desain, "DIMENSI SPIRITUAL DALAM SENI LUKIS ABSTRAK KONTEMPORER INDONESIA," *Acintya Jurnal Penelitian Seni Budaya*, vol. 11, no. 1, pp. 1–11, 2019, doi: 10.33153/acy.v11i1.2609.

[3] C. Ledig *et al.*, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," *arxiv*, vol. 5, no. 1, pp. 1–5, 2017, doi: 10.48550/arXiv.1609.04802.

[4] X. Liang, H. Zhang, and E. P. Xing, "Generative Semantic Manipulation with Contrasting GAN," *arxiv*, vol. 1, pp. 1–12, 2017, doi: 10.48550/arXiv.1708.00315.

[5] J. D. Curtó, I. C. Zarza, F. Torre, I. King, and M. R. Lyu, "High-resolution Deep Convolutional Generative Adversarial Networks," *arxiv*, vol. 18, pp. 1–8, 2020, doi: 10.48550/arXiv.1711.06491.

[6] X. Yi, E. Walia, and P. Babyn, "Generative Adversarial Network in Medical Imaging: A Review," *Medical Image Analysis*, vol. 58, pp. 1–22, 2019, doi: 10.1016/j.media.2019.101552.

[7] W. Hu and Y. Tan, "Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN," *arxiv*, vol. 1, pp. 1–6, 2017, doi: 10.48550/arXiv.1702.05983.

[8] I. J. Goodfellow *et al.*, "Generative Adversarial Nets," *NIPS*, vol. v1, pp. 1–8, 2014, doi: 10.48550/arXiv.1406.2661.

[9] D. Volkhonskiy, B. Borisenko, and E. Burnaev, "GENERATIVE ADVERSARIAL NETWORKS FOR IMAGE STEGANOGRAPHY," *ICLR 2017 conference submission*, pp. 1–7, 2016.

[10] H. Fonda, Y. Irawan, A. Febriani, S. Informatika, and H. T. Pekanbaru, "KLASIFIKASI BATIK RIAU DENGAN MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORKS (CNN)," *JIK*, vol. 9, no. 1, pp. 1–10, 2020, doi: 10.33060/JIK/2020/Vol9.Iss1.144.

[11] A. T. Putra, K. Usman, and S. Saidah, "WEBINAR STUDENT PRESENCE SYSTEM BASED ON REGIONAL CONVOLUTIONAL NEURAL NETWORK USING FACE RECOGNITION," *Jurnal Teknik Informatika (Jutif)*, vol. 2, no. 2, pp. 109–118, Mar. 2021, doi: 10.20884/1.jutif.2021.2.2.82.

[12] T. Henighan and S. Physics, "Spatial Control in Neural Style Transfer," *CS231N*, pp. 1–6, 2017.

[13] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, "Unrolled Generative Adversarial Networks," *ICLR*, vol. 4, pp. 1–25, Nov. 2016, doi: 10.48550/arXiv.1611.02163.

[14] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative Adversarial Networks: An Overview," *Institute of Electrical and Electronics Engineers (IEEE)*, vol. 1, pp. 1–7, Oct. 2017, doi: 10.1109/MSP.2017.2765202.

[15] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," *arXiv*, vol. 2, pp. 1–4, Nov. 2015, doi: 10.48550/ARXIV.1511.06434.

[16] S. K. Venu and S. Ravula, "Evaluation of deep convolutional generative adversarial networks for data augmentation of chest x-ray images," *Future Internet*, vol. 13, no. 1, pp. 1–13, Jan. 2020, doi: 10.3390/fi13010008.

[17] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved Techniques for Training GANs," *arXiv*, vol. 1, pp. 1–8, Jun. 2016, doi: 10.48550/ARXIV.1606.03498.

[18] S. Murni and P. S. Rupa, "DISTORSI DISLEKSIA MELALUI LUKISAN ABSTRAK DENGAN REALITAS BERIMBUH (AR) Anjani Imania Citra Afsiser," *Ikonik jurnal seni dan Desain*, vol. 3, no. 2, pp. 14–18, 2021, doi: 10.51804/ijsd.v3i2.993.

[19] S. Sen and K. Sawant, "Face mask detection for covid_19 pandemic using pytorch in deep learning," *IOP Conference Series: Materials Science and Engineering*, vol. 1070, no. 1, pp. 1–5, 2021, doi: 10.1088/1757-899X/1070/1/012061.