Jurnal Teknik Elektro dan Komputasi (ELKOM) ISSN : 2685-1814 (*Print*)
Volume 4 Nomor 1 Maret 2022 ‖ Hal. 21-29 ISSN : 2685-7677 (*Online*)

Penciptaan Karakter Anime Otomatis Dengan Menggunakan *Generative Adversarial Networks*

Vika Vitaloka Pramansah[1], Dadang Iskandar Mulyana[2], Titi Silfia[2], Rudi Tri Jaya[2]

[1] Sistem Informasi, STIKOM Cipta Karya Informatika, [2] Teknik Informatika, STIKOM Cipta Karya Informatika
Jl. Raden Inten II No.8, RT.5/RW.14, Duren Sawit, Kec. Duren Sawit, Kota Jakarta Timur
Daerah Khusus Ibukota Jakarta 13440
E-mail: vitalokavika@gmail.com

Naskah Masuk: 29 Januari 2022; Diterima: 16 Februari 2022; Terbit: 25 Maret 2022

ABSTRAK

Abstrak – Karakter anime merupakan karakter fiksi yang memiliki keunikan tersendiri dan karakternya menggambarkan karakter yang dimiliki manusia dengan arsitektur serta desainnya yang unik. Setiap tahun, ada karakter anime baru yang dimunculkan melalui media Televisi, Webtoon, Netflix, Youtube dan sebagainya. Sehingga terdapat penelitian yang meneliti tentang hal ini, bagaimana menciptakan karakter anime secara otomatis dengan Computer Vision. Penelitian ini bertujuan untuk menciptakan karakter Anime otomatis menggunakan *Generative Adversarial Networks* (GAN), GAN merupakan model *generative* yang membuat *instance* data baru menyerupai data pelatihan kita. Penelitian ini menggunakan data training sebanyak 63.565 citra dan data uji sebanyak 10.000 citra serta menggunakan PyTorch untuk melatih GAN. Berdasarkan hasil penelitian ini, GAN dapat dengan baik menghasilkan karakter Anime baru dengan *error* dari waktu ke waktu yang cukup tinggi untuk data trainingnya memiliki rata-rata 8.5445 untuk *error generator*, rata-rata *error discriminator* yaitu 0.1587, rata-rata skor *discriminator* yaitu 0.9362 dan rata-rata skor *generator* adalah 0.0603. Namun data testingnya memiliki *error* yang lebih rendah dan skor yang lebih tinggi untuk *generator* yaitu menghasilkan nilai rata-rata untuk *error generator* adalah 5.4472, *error discriminator* 0.6511, skor gambar asli 0.7913 dan skor gambar palsu 0.2056.

Kata kunci: *Generative Adversarial Networks*, Wajah Anime, *Generator, Discriminator*, Jaringan Saraf Tiruan.

ABSTRACT

Abstract **-** Anime characters are fictional characters that have their own uniqueness and the characters describe the characters possessed by humans with their unique architecture and designs. Every year, there are new anime characters that appear through Television, Webtoon, Netflix, Youtube and so on. So there is research that examines this, how to create anime characters automatically with Computer Vision. This study aims to create automatic Anime characters using Generative Adversarial Networks (GAN), GAN is a generative model that creates new data instances resembling our training data. This study uses training data of 63.565 images and test data of 10.000 images and uses PyTorch to train GAN. Based on the results of this study, GAN can well produce new Anime characters with errors from time to time which are high enough for the training data to have an average of 8.5445 for the generator error, the average error discriminator is 0.1587, the average score discriminator is 0.9362 and the average score generator is 0.0603. However, the testing data has a lower error and a higher score for the generator, which produces an average value for the generator error is 5.4472, the discriminator error is 0.6511, the original image score is 0.7913 and the fake image score is 0.2056.

Keywords: Generative Adversarial Networks, Anime Face, Generator, Discriminator, Neural Network.

1. PENDAHULUAN

Karakter anime yang merupakan gambar ciptaan manusia sangat dipengaruhi oleh teori perlindungan kekayaan intelektual yang dicetuskan oleh *Robert M. Sherwood*. Mengkarakterisasi struktur gambar anime secara otomatis telah menjadi upaya penelitian yang kaya. Anime yang memiliki karakteristik disetiap desain dan arsitektur yang unik dapat mematuhi invarian intrinsik dan menunjukkan struktur statistik multi-skala yang secara historis sulit untuk diukur [1]. Peningkatan kualitas model gambar secara substansial merupakan kemajuan terbaru dalam pembelajaran mesin [2]. Proes pembuatan anime cukuplah panjang

dan rumit, terlebih anime sebelumnya merupakan gambar. Dalam pembuatan anime, tahapan yang pertama adalah perencanaan dimana penulis anime membuat ide-ide serta konsep karakter anime sesuai dengan sifat yang akan karakter anime tersebut miliki contohnya karakter anime yang baik akan memiliki senyum lembut dan mata yang besar ceria. Tahap kedua adalah *Storyboard* dimana penulis akan menentukan dialog, potongan cerita, latar serta gerakan karakternya. Tahap ketiga adalah *Layout* dimana penulis akan menambahkan objek pada pada adegan cerita yang telah dibuat. Tahap keempat yaitu *Key Animation* dimana gerakan detail karakter anime dibuat. Tahap kelima yaitu *In-Beetween Animation* dimana penulis memastikan gerakan yang dilakukan karakter anime begitu halus dan tampak realistis. Tahap keenam adalah *Digitalize* yaitu pewarnaan serta pemberian efek visual agar terlihat seperti karakter 3D (tiga dimensi). Tahap terakhir yaitu *Editing* dimana adegan yang telah dibuat siap ditambahkan musik serta ditambahkan suara para tokohnya agar semakin nyata.

Berdasarkan proses yang lumayan panjang tersebut, pembuatan wajah karakter anime memakan waktu yang lumayan lama karena penulis harus memastikan agar karakter yang dibuatnya unik dan berbeda dari karakter lainnya. Dari sinilah penelitian ini dimaksudkan untuk membuat wajah karakter anime yang baru secara otomatis menggunakan GAN. Pemodelan generatif melibatkan pengambilan satu set sampel yang diambil dari distribusi pembangkit data yang tidak diketahui Pnyata dan menemukan perkiraan Pmodel yang sangat mirip dengannya [3]. *Generative Adversarial Networks* (GAN) adalah kerangka kerja yang kuat yang digunakan untuk menyesuaikan model generatif secara implisit [4]. Pengaturan dasar terdiri dari dua jaringan *generator* dan *discriminator*. *Generator* membuat data palsu dengan memasukkan umpan balik dari *discriminator* dan *generator* belajar bagaimana membuat *discriminator* mengklasifikasikan hasil data/outputnya dianggap asli [5]. Berdasarkan hal ini, kita juga akan mengetahui nilai *error* dan skor *generator* dan *discriminator* yang dimana *discriminator* harus membedakan gambar yang *generator* hasilkan apakah dianggap asli ataukah palsu.

2. KAJIAN PUSTAKA

Generative Adversarial Network (GAN) menunjukkan hasil yang mengesankan dalam pembuatan gambar, transfer gambar, resolusi super dan banyak tugas *generative* lainnya. Inti dari GAN dapat diringkas sebagai pelatihan sebagai pembeda generator model dan model asli secara bersamaan, di mana model *discriminator* mencoba untuk membedakan citra yang dihasilkan dari *generator* dan citra asli dimana *generator* mencoba menghasilkan citra realistis yang diharapkan tidak dapat dibedakan oleh *discriminator*. Hal tersebut dapat digambarkan sebagai *error* yang merugikan yang diterapkan pada *generator* dan *discriminator* dalam proses pelatihan yang sebenarnya, yang secara efektif mendorong citra yang dihasilkan *generator* serupa dengan data asli.

Sebelumnya, penelitian yang terkait dengan penciptaan citra dengan GAN telah dilakukan yaitu "*Towards the Automatic Anime Characters Creation with Generative Adversarial Networks*" yang diteliti oleh Yanghua Jin, Jiakai Zhang, Minjun Li, Yingtao Tian, Huachun Zhu, dan Zhihao Fhang, dalam penelitian ini menggunakan sampel 12.800 gambar dan menghasilkan gambar palsu dengan menggunakan kondisi yang sesuai untuk setiap sampel gambar yang asli [6]. Penelitian kedua yaitu "Generalisasi dan Ekuilibrium dalam *Generative Adversarial Nets* (GANs)" oleh Arora, S., Ge, R., Liang, Y., & Zhang, Y menggunakan dataset MNIST sebanyak 60.000 label (gambar tulisan tangan) dan dataset CelebA sebanyak 200.000 data (gambar wajah manusia), menghasilkan skor 3.82 untuk WasserteinGAN dan skor 4.04 untuk mix + WasserteinGAN (dengan 5 komponen) [7]. Penelitian ketigaa yaitu "Sintesis Gambar Bersyarat dengan GAN Pengklasifikasi Tambahan Abstrak," oleh A. Odena, C. Olah, dan J. Shlens, menggunakan dataset sebanyak 1.000 kelas ImageNet, menghasilkan akurasi *Inception* rata-rata dari model (10.1% ± 2.0%) dengan akurasi data *training* sebesar 81% [8].

Pada penelitian ini, bertujuan untuk menciptakan karakter Anime otomatis menggunakan *Generative Adversarial Networks* (GAN) dengan dataset berjumlah berjumlah 63.565 citra sebagai data *training* dan 10.000 citra sebagai data *testing* dimana karakter anime yang akan dihasilkan merupakan penggabungan dari karakter anime yang ada di dataset yang telah disediakan. Kita akan menganalisa bagaimana *discriminator* dapat membedakan antara gambar yang telah kita hasilkan apakah dianggap asli ataukah palsu. Dataset diambil dari Kaggle oleh Spencer Churchill yang berjudul "Anime Face Dataset" tahun 2020, URL nya yaitu https://www.kaggle.com/splcher/animefacedataset dimana sebanyak 63.565 digunakan sebagai data *training* dan 10.000 data *testing.*

2.1 *Generative Adversarial Networks*

Generative Adversarial Networks (GAN) merupakan model *generative* yang membuat *instance* data baru menyerupai data pelatihan kita. Dalam penelitian ini menggunakan PyTorch serta menggunakan 63.565 citra untuk melatih GAN. Pytorch merupakan sebuah *library AI* (*Artificial Intelligence*) yang sangat handal dimana Facebook yang mengembangkannya. Pytorch digunakan secara khusus untuk

mengembangkan aplikasi berbasis AI di bidang NLP (*Natural Linguistic Processing*) dan CV (*Computer Vision*).

2.2 Anime

Karakter anime yang merupakan gambar ciptaan manusia sangat dipengaruhi oleh teori perlindungan kekayaan intelektual yang dicetuskan oleh *Robert M. Sherwood*. Mengkarakterisasi struktur gambar anime secara otomatis telah menjadi upaya penelitian yang kaya sampai saat ini.

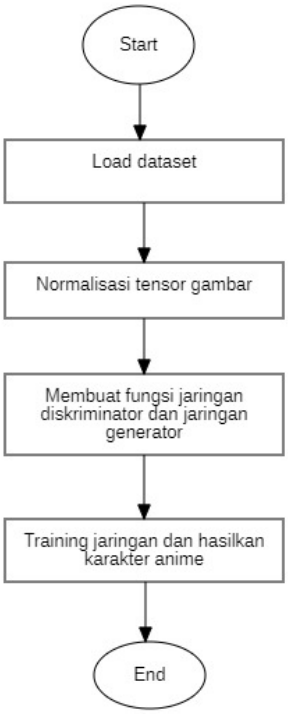
Gambar 1. Contoh Anime

2.3 PyTorch

PyTorch merupakan library untuk bahasa pemprograman Python yang memfasilitasi pembangunan proyek *Deep Learning*. Python adalah bahasa pemprograman yang mudah dibaca dan mudah dipahami. PyTorch menekankan kemampuannya yang fleksibel dan memungkinkan model *Deep Learning* untuk diekspresikan dalam Python. PyTorch mendukung grafik komputasi dinamis yang memungkinkan untuk mengubah perilaku jaringan dengan cepat.

3. METODE PENELITIAN

Penerapan metode *Generative Adversarial Networks* (GAN) yang dapat secara otomatis menciptakan karakter anime baru dapat dilihat pada diagram alir berikut ini :

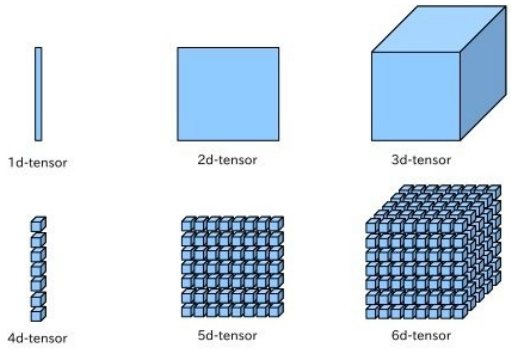


Gambar 2. Diagram alir penelitian

Data yang digunakan melalui studi pustaka dimana mengumpulkan data berasal dari informasi yang tertulis atau dari data sekunder yang didapat melalui referensi buku juga internet [9]. Dataset yang digunakan dalam penelitian ini merupakan dataset publik yang diekstrak dari kaggle berjumlah 63.565 citra sebagai data *training* dan 10.000 citra sebagai data *testing*.

3.1 Normalisasi Tensor Gambar

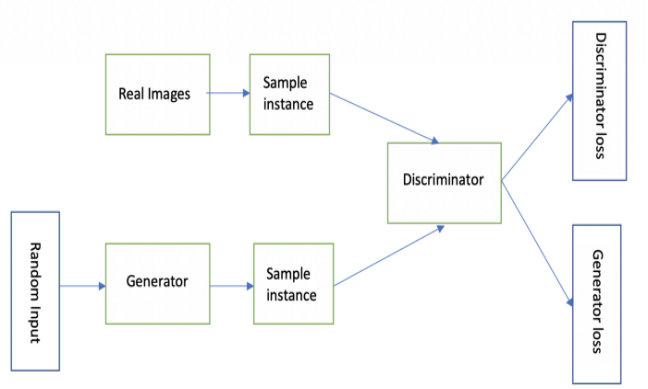
Tensor adalah tempat untuk data dimana lebih sering kita jumpai bertipe numerik. Jadi, ini adalah wadah untuk angka. Tensor merupakan generalisasi matriks yang dapat berubah-ubah dalam hal ini dimensi disebut sumbu. Dalam penelitian ini citra akan dinormalisasikan dengan dikecilkan menjadi ukuran 64x64 piksel. Juga menormalkan nilai piksel dengan mean & standar deviasi 0.5 untuk setiap saluran. Hal ini akan memastikan bahwa nilai piksel berada dalam kisaran (-1, 1) yang lebih nyaman untuk melatih *discriminator*.



Gambar 3. Visualisasi tensor

3.2 Jaringan *Generator* dan *Discriminator*

Jaringan *generator* adalah jaringan yang membuat data palsu dengan memasukkan umpan balik dari *discriminator* dan *generator* belajar bagaimana membuat *discriminator* mengklasifikasikan hasil data/outputnya agar dianggap data yang asli [10]. Jaringan *discriminator* akan memeriksa apakah gambar yang berhasil dihasilkan oleh *generator* itu dianggap asli atau palsu [11].



Gambar 4. Visualisasi jaringan *generator* dan *discriminator*

3.3 *Training* Dataset

Training dataset digunakan untuk mempelajari bagaimana menghasilkan data baru dengan statistik yang sama dengan data *training*. Dalam pelatihan ini, GAN terlatih dalam penciptaan karakter anime dimana karakter tersebut dihasilkan secara otomatis dan unik menurut penglihatan manusia. Dataset yang berjumlah 63.565 citra ini akan dilatih yang kemudian *generator* menghasilkan karakter anime baru.

4. HASIL DAN PEMBAHASAN

Langkah pertama yang harus dilakukan yaitu memuat dataset kemudian ukuran citra dari dataset yang kita miliki diubah menjadi lebih kecil, dalam penelitian ini citra akan dikecilkan menjadi ukuran 64x64 piksel. Juga menormalkan nilai piksel dengan mean & standar deviasi 0.5 untuk setiap saluran. Hal ini akan memastikan bahwa nilai piksel berada dalam kisaran (-1,1) yang lebih nyaman untuk melatih *discriminator*. Setelah mengubah ukuran citra maka kita dapat menormalisasi tensor citra dan berikut merupakan citra dari pelatihan:

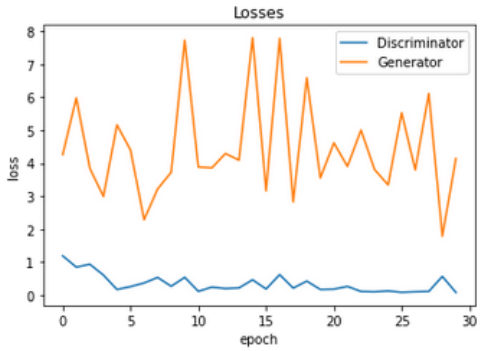
Gambar 5. Citra yang dilatih

Citra yang dilatih merupakan citra *training* kita yang akan kita hasilkan karakter anime baru dengan GAN. Waktu dalam penciptaan karakter dapat sangat lama tergantung jumlah serta komputer kita, jika memakai CPU (*Central Processing Unit*) yaitu prosesor yang bertanggung jawab dalam menjalankan instruksi pada komputer, kemungkinan akan sangat lama dalam menjalankan script codenya maka dalam penelitian ini menggunakan GPU (*Graphics Processing Unit*) yaitu *chip* yang diprogram dan dispesialisasikan untuk berbagai kebutuhan serta membantu kinerja CPU agar lebih baik dalam mengeksekusi perintah kita. Kita dapat mengecek apakah komputer kita tersedia GPU untuk menjalankannya atau tidak, jika tidak memiliki GPU maka akan menggunakan CPU.

Selanjutnya yaitu membuat jaringan *discriminator* dan jaringan *generator*. *Generator* membuat data palsu dengan memasukkan umpan balik dari *discriminator* dan *generator* belajar bagaimana membuat *discriminator* mengklasifikasikan hasil data/*output* nya dianggap asli. Langkah berikutnya dilakukannya *training* untuk menghasilkan karakter anime baru dengan *epoch* 30 dimana ketika seluruh dataset telah melalui proses *training* pada Jaringan Saraf Tiruan sampai dikembalikan ke awal untuk sekali iterasi atau putaran. *Epoch* dapat didefinisikan sebagai jumlah neuron yang dapat melihat semua data yang telah dikumpulkan, sedangkan ukuran batch adalah jumlah contoh pelatihan dalam satu lintasan maju/mundur [12]. Berikut adalah karakter anime yang dihasilkan oleh *generator*:

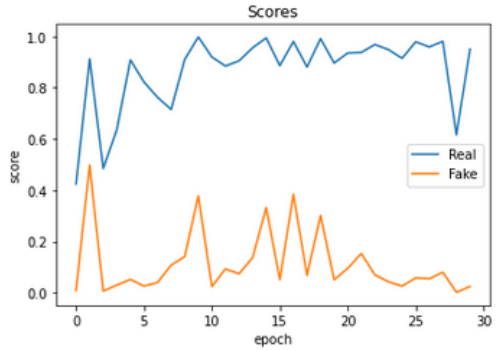
Gambar 6. Karakter Anime yang berhasil dibuat menjadi karakter baru

Setelah itu, dilakukannya visualisasi untuk mengetahui seberapa tinggi *error* dan skor *generator* maupun *discriminator* dari waktu ke waktu dan berikut gambarnya:



Gambar 7. Visualisasi *error* data *training*

Error discriminator adalah salah mengklasifikasikan data asli sebagai palsu, atau data palsu (dibuat oleh *generator*) sebagai asli. *Error generator* kemudian dihitung dari klasifikasi *discriminator* dimana akan dihitung jika berhasil menipu *discriminator*. Berikut visualisasi skor yang didapatkan:



Gambar 8. Visualisasi skor data *training*

Skor *generator* adalah rata-rata dari probabilitas yang sesuai dengan *output discriminator* untuk citra yang dihasilkan. Sedangkan skor *discriminator* adalah rata-rata probabilitas yang sesuai dengan keluaran

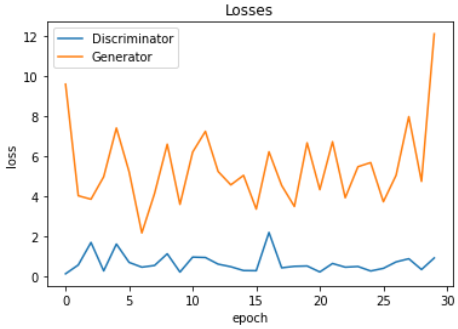
discriminator untuk gambar asli dan yang dihasilkan. Berikut detail *error* maupun skor yang didapat dari data citra baru:

Diketahui:
I = Iterasi
Loss_g = *Error Generator*
Loss_d = *Error Discriminator*
Score_real = Skor Gambar Asli
Score_fake = Skor Gambar Palsu

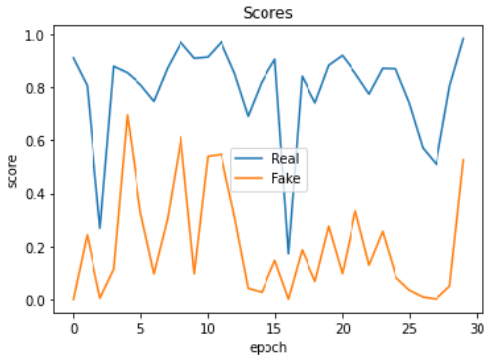
Tabel 1. Detail *training* dataset

I	Loss_g	Loss_d	Score_real	Score_fake
1	4.4686	0.4566	0.7854	0.1543
2	3.0960	0.4977	0.7675	0.1285
3	4.0954	0.3838	0.7928	0.0947
4	4.3336	0.2996	0.7945	0.0227
5	6.3252	0.1351	0.9187	0.0358
6	5.0265	0.1383	0.9616	0.0753
7	6.5721	0.1096	0.9803	0.0822
8	5.9204	0.2172	0.9080	0.0883
9	4.5240	0.1169	0.9309	0.0232
10	5.9092	0.1561	0.9377	0.0446
11	7.0172	0.0704	0.9750	0.0388
12	6.6617	0.1313	0.9406	0.0566
13	5.5148	0.0969	0.9387	0.0098
14	7.8384	0.0337	0.9955	0.0280
15	12.6334	0.0710	0.9963	0.0403
16	6.8767	0.0509	0.9798	0.0235
17	10.1781	0.0916	0.9277	0.0007
18	6.5754	0.0827	0.9719	0.0430
19	7.3811	0.0726	0.9870	0.0539
20	30.9763	0.1034	0.9333	0.0000
21	16.9591	0.3280	0.9921	0.2478
22	6.5336	0.0507	0.9637	0.0083
23	8.2488	0.0786	0.9884	0.0566
24	5.7421	0.1376	0.9375	0.0440
25	15.8789	0.2581	0.9889	0.1973
26	7.9730	0.2318	0.8643	0.0040
27	7.3994	0.0524	0.9697	0.0091
28	7.4678	0.0309	0.9810	0.0095
29	21.4336	0.2567	0.9898	0.1827
30	6.7751	0.0199	0.9867	0.0061
Rata-rata	**8.5445**	**0.1587**	**0.9362**	**0.0603**

Dari tabel data *training* maka didapatkan rata-rata untuk *error generator* adalah 8.5445, *error discriminator* 0.1587, skor gambar asli 0.9362 dan skor gambar palsu 0.0603. Dari tabel data *training*, terlihat *error*/skor dari setiap iterasi dimana *error generator* cukup tinggi dan *error discriminator* yang rendah menandakan *dicrimintor* lebih sering bisa membedakan karakter asli dan palsu. Kemudian dilakukannya pengujian dataset sebanyak 10.000 citra. Visualisasi *error* dan skornya sebagai berikut:



Gambar 9. Visualisai *error* data *testing*



Gambar 10. Visualisasi skor data *testing*

Berikut detail e*rror* maupun skor yang didapat dari data citra baru:

Diketahui:
I = Iterasi
Loss_g = *Error Generator*
Loss_d = *Error Discriminator*
Score_real = Skor Gambar Asli
Score_fake = Skor Gambar Palsu

Tabel 2. Detail *testing* dataset

I	Loss_g	Loss_d	Score_real	Score_fake
1	9.5758	0.1181	0.9113	0.0001
2	4.0119	0.5580	0.8075	0.2436
3	3.8366	1.6805	0.2685	0.0047
4	4.9592	0.2631	0.8803	0.1143
5	7.3913	1.6016	0.8555	0.6961
6	5.2051	0.6856	0.8109	0.3312
7	2.1557	0.4480	0.7474	0.0963
8	4.1233	0.5334	0.8737	0.3053
9	6.5818	1.1151	0.9714	0.6131
10	3.5805	0.2029	0.9103	0.0963
11	6.1917	0.9516	0.9150	0.5400
12	7.2250	0.9303	0.9725	0.5474
13	5.2226	0.6000	0.8534	0.3113
14	4.5524	0.4695	0.6912	0.0417
15	5.0281	0.2799	0.8212	0.0268
16	3.3447	0.2730	0.9067	0.1467
17	6.2043	2.1797	0.1721	0.0007
18	4.5243	0.4152	0.8424	0.1868
19	3.4761	0.4896	0.7418	0.0679
20	6.6502	0.5074	0.8839	0.2759
21	4.3136	0.2081	0.9210	0.0973
22	6.7061	0.6322	0.8527	0.3338
23	3.9123	0.4471	0.7748	0.1294
24	5.4584	0.4819	0.8725	0.2564
25	5.6657	0.2579	0.8704	0.0839
26	3.7127	0.3890	0.7419	0.0345
27	5.0333	0.7106	0.5717	0.0090
28	7.9593	0.8686	0.5091	0.0009
29	4.7272	0.3295	0.8043	0.0498
30	12.0880	0.9068	0.9847	0.5266
Rata-rata	**5.4472**	**0.6511**	**0.7913**	**0.2056**

Dari tabel data *testing* maka didapatkan rata-rata untuk *error generator* adalah 5.4472, *error discriminator* 0.6511, skor gambar asli 0.7913 dan skor gambar palsu 0.2056. Dari tabel data *testing*, terlihat *error*/skor dari setiap iterasi dimana *error generator* cukup tinggi namun lebih rendah dari uji

training dan *error discriminator* yang cukup tinggi menandakan generator lebih sering membuat *dicriminator* salah yang berarti *discriminator* sering keliru membedakan gambar palsu menjadi asli.

5. KESIMPULAN

Menciptakan karakter Anime otomatis menggunakan *Generative Adversarial Networks* (GAN) adalah tujuan penelitian ini, GAN merupakan model *generative* yang membuat instance data baru menyerupai data pelatihan kita. Penelitian ini menggunakan sebanyak berjumlah 63.565 citra sebagai data *training* dan 10.000 citra sebagai data *testing* dan menggunakan PyTorch untuk melatih GAN. Berdasarkan hasil penelitian ini, GAN dapat dengan baik menghasilkan karakter Anime baru dengan *error* dari waktu ke waktu yang cukup tinggi untuk data trainingnya memiliki rata-rata 8.5445 untuk *error generator*, rata-rata *error discriminator* yaitu 0.1587, rata-rata skor *discriminator* yaitu 0.9362 dan rata-rata skor *generator* adalah 0.0603. Namun data testingnya memiliki *error* yang lebih rendah dan skor yang lebih tinggi untuk *generator* yaitu menghasilkan nilai rata-rata untuk *error generator* adalah 5.4472, *error discriminator* 0.6511, skor gambar asli 0.7913 dan skor gambar palsu 0.2056. Hal ini menandakan generator lebih sering membuat *dicriminator* salah yang berarti *discriminator* sering keliru membedakan gambar palsu menjadi asli.

REFERENSI

[1] E. P. Simoncelli and B. A. Olshausen, “N i s n r,” 2001.
[2] C. Ledig *et al.*, “Foto-Realistik Super-Resolusi Gambar Tunggal Menggunakan Perlawanan Generatif Jaringan Abstrak,” pp. 4681–4690.
[3] C. Li, K. Xu, J. Zhu, and B. Zhang, “Triple generative adversarial nets,” *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, pp. 4089–4099, 2017.
[4] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, “Improved training of wasserstein GANs,” *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, pp. 5768–5778, 2017.
[5] O. I. Goodfellow *et al.*, “Jaringan Permusuhan Generatif,” vol. 63, no. November, 2020.
[6] Y. Jin, J. Zhang, M. Li, Y. Tian, H. Zhu, and Z. Fang, “Towards the Automatic Anime Characters Creation with Generative Adversarial Networks,” vol. 92, pp. 1–16, 2017, [Online]. Available: http://arxiv.org/abs/1708.05509.
[7] S. Arora, R. Ge, Y. Liang, and Y. Zhang, “Generalisasi dan Ekuilibrium dalam Generative Adversarial Nets (GANs),” 2017.
[8] A. Odena, C. Olah, and J. Shlens, “Sintesis Gambar Bersyarat dengan GAN Pengklasifikasi Tambahan Abstrak,” 2017.
[9] L. Genisa and D. I. Mulyana, “Implementasi Penerapan Metode C4. 5 dan Naïve Bayes Dalam Tingkat Kelulusan Akreditasi Lembaga PAUD Pada Badan Akreditasi Nasional,” *J. Media …*, vol. 5, pp. 1595–1604, 2021, doi: 10.30865/mib.v5i4.3267.
[10] M. Heusel, S. Hochreiter, U. Johannes, and K. Linz, “GAN Dilatih dengan Aturan Pembaruan Dua Skala Waktu Konvergen ke Kesetimbangan Nash Lokal,” no. Nips, 2017.
[11] P. Isola, A. A. Efros, L. Penelitian, A. I. Berkeley, and U. C. Berkeley, “Terjemahan Gambar-ke-Gambar dengan Jaringan Berlawanan Bersyarat Abstrak,” pp. 1125–1134.
[12] D. I. Mulyana, “Optimization of Image Classification Using the Convolutional Neural Network (CNN) Algorithm for Cirebon Batik Image Indonesian,” no. 12, pp. 39–46, 2021.