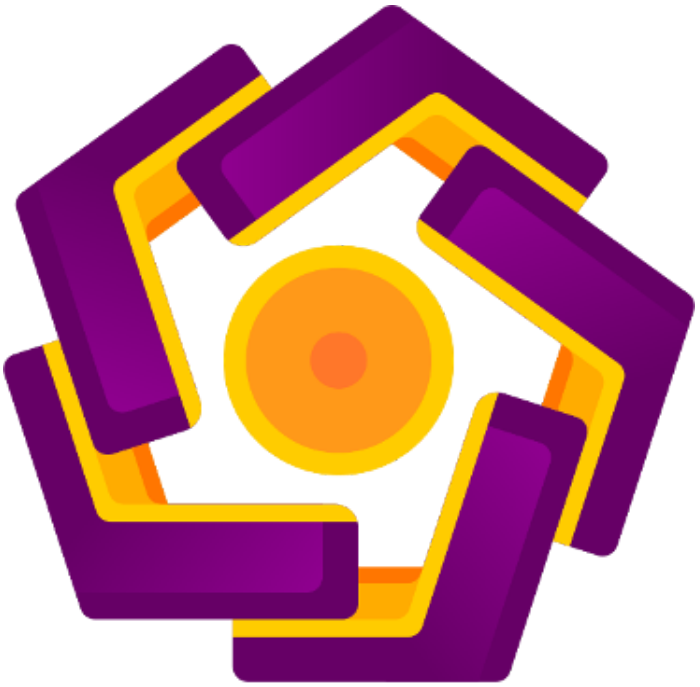
Laporan Final Project
Kecerdasan Buatan (Lanjut)

Pembangkitan Citra Wajah Anime Menggunakan Deep Convolutional Generative Adversarial Networks (DCGAN) dengan Strategi Split Learning Rate



Disusun oleh :

Muhammad Tegar Revolusi Seto	23.11.5743
Redomas Baegy Hardianathan	23.11.5733
Daffa Akmal Ayom Pamungkas	23.11.5728
Muhamad Fikih Rizaldi	23.11.5724

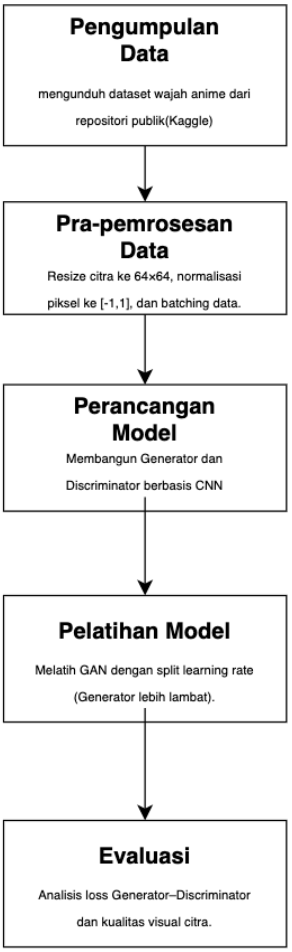
1. Latar Belakang

Dalam industri hiburan dan media digital, permintaan terhadap aset visual karakter animasi (anime) terus meningkat secara eksponensial. Namun, proses pembuatan karakter anime secara manual oleh seniman membutuhkan waktu yang lama dan biaya produksi yang tinggi. Di sisi lain, penggunaan *Deep Learning* dalam visi komputer (*Computer Vision*) sering terkendala oleh keterbatasan dataset dan isu privasi pada penggunaan wajah manusia asli [1]. Oleh karena itu, dibutuhkan sebuah sistem yang mampu membangkitkan citra wajah sintetis secara otomatis untuk mempercepat proses kreatif dan menyediakan dataset tanpa batasan hak cipta.

Generative Adversarial Networks (GAN) yang diperkenalkan oleh Goodfellow et al. (2014) menjadi solusi populer untuk masalah ini. Namun, pelatihan GAN standar sering mengalami ketidakstabilan seperti *vanishing gradient* dan *mode collapse* (menghasilkan gambar yang sama terus-menerus) [2]. Arsitektur *Deep Convolutional GAN* (DCGAN) kemudian diajukan untuk meningkatkan stabilitas menggunakan lapisan konvolusi [3]. Meskipun demikian, tantangan utama dalam DCGAN adalah menyeimbangkan kemampuan antara Generator dan Discriminator; jika Discriminator terlalu dominan, Generator akan gagal belajar. Penelitian ini bertujuan mengimplementasikan DCGAN dengan strategi modifikasi *Split Learning Rate* dan *Label Smoothing* untuk mengatasi dominasi Discriminator dan menghasilkan wajah anime berkualitas tinggi.

2. Metode

A. Alur Pengerjaan (Project Flow)



Tahapan pengerjaan proyek dilakukan sebagai berikut:

Tahapan pengerjaan proyek ini disusun secara bertahap untuk memastikan model dapat dilatih secara stabil dan menghasilkan citra wajah anime yang berkualitas.

1. **Pengumpulan Data:**

Pada tahap awal, data dikumpulkan dengan mengunduh dataset wajah anime yang tersedia secara terbuka melalui platform Kaggle. Pemilihan Kaggle didasarkan pada kemudahan akses, kelengkapan metadata, serta ketersediaan dataset dalam jumlah besar yang sesuai untuk kebutuhan pelatihan model deep learning.

2. **Pra-pemrosesan:**

Data citra yang telah diperoleh tidak dapat langsung digunakan sehingga perlu dilakukan pra-pemrosesan terlebih dahulu. Setiap citra diubah ukurannya menjadi 64×64 piksel untuk menyeragamkan dimensi input model. Selain itu, nilai piksel dinormalisasi ke rentang [-1, 1] agar distribusi data sesuai dengan fungsi aktivasi yang digunakan pada Generator. Selanjutnya, data dimuat ke dalam DataLoader sehingga proses pelatihan dapat dilakukan secara batch dan lebih efisien.

3. **Perancangan Model:**

Model yang digunakan dalam penelitian ini adalah Generative Adversarial Network (GAN) yang terdiri dari dua jaringan utama, yaitu Generator dan Discriminator. Kedua jaringan dirancang menggunakan pendekatan Convolutional Neural Network (CNN). Generator bertugas menghasilkan citra wajah anime baru dari vektor noise acak, sementara Discriminator berperan sebagai penilai yang membedakan antara citra asli dari dataset dan citra hasil pembangkitan Generator.

4. **Pelatihan Model:**

Proses pelatihan dilakukan dengan melatih Generator dan Discriminator secara bergantian dalam satu siklus training. Untuk menjaga keseimbangan pembelajaran, diterapkan strategi Split Learning Rate, di mana laju pembelajaran Generator dibuat lebih kecil dibandingkan Discriminator. Pendekatan ini bertujuan agar Generator belajar secara lebih bertahap dan tidak mengalami ketidakstabilan selama proses pelatihan.

5. **Evaluasi:**

Tahap akhir adalah evaluasi performa model. Evaluasi dilakukan dengan mengamati metrik klasifikasi pada Discriminator untuk melihat kemampuannya dalam membedakan citra asli dan palsu. Selain itu, proses pelatihan dianalisis melalui visualisasi kurva loss Generator dan Discriminator, serta penilaian kualitas visual citra wajah anime yang dihasilkan oleh Generator.

B. Algoritma & Arsitektur Model

Algoritma yang digunakan adalah DCGAN (Deep Convolutional GAN). Model ini terdiri dari dua jaringan saraf yang saling berkompetisi (Minimax Game):

1. **Generator (G):** Bertugas membangkitkan citra palsu dari vektor acak (*latent vector*).
    - *Input:* Vektor Noise $100 \times 1 \times 1$.
    - *Layer:* 4 lapis *Transposed Convolution* (Upsampling).

- *Aktivasi:* ReLU pada setiap layer, dan Tanh pada layer output (untuk menghasilkan citra RGB).
- *Optimasi:* Adam Optimizer dengan **Learning Rate 0.0001** (Sengaja diperlambat agar belajar lebih detail).

2. **Discriminator (D):** Bertugas membedakan citra asli (dari dataset) dan citra palsu (dari Generator).
    - *Input:* Citra RGB 64×64.
    - *Layer:* 4 lapis *Convolution* (Downsampling) dengan *Strides*.
    - *Aktivasi:* LeakyReLU (slope 0.2) dan Sigmoid pada output akhir.
    - *Optimasi:* Adam Optimizer dengan **Learning Rate 0.0002** (Standar).
    - *Teknik Stabilisasi:* Menggunakan **Label Smoothing** dengan target label 'Real' = **0.98**.

3. Dataset

Dataset yang digunakan dalam penelitian ini bersumber dari **Kaggle** dengan nama *Anime Faces Dataset*. Dataset ini merupakan kumpulan citra yang didapatkan melalui metode *web crawling* dari situs *[tautan mencurigakan telah dihapus]*.

- **Jumlah Data:** ~21.000 citra wajah.
- **Format Asli:** Beragam resolusi.
- **Preprocessing:** Seluruh citra di-*crop* hanya pada bagian wajah, diubah ukurannya menjadi **64x64 piksel**, dan dikonversi ke format tensor PyTorch dengan normalisasi (0.5, 0.5, 0.5).

4. Hasil Pengujian

A. Skenario Pengujian

Pengujian dilakukan menggunakan perangkat keras dengan GPU NVIDIA GeForce RTX 3050.

- **Epoch:** 50 Epoch.
- **Batch Size:** 128.
- **Durasi Pelatihan:** 33 Menit 23 Detik.
- **Hyperparameter:** Label Real = 0.98, Label Fake = 0.0.

B. Hasil Kuantitatif

Evaluasi dilakukan pada akhir epoch ke-50 untuk mengukur performa Discriminator dalam mendeteksi citra hasil bangkitan Generator.

Metrik Evaluasi	**Hasil (%)**
Akurasi (*Accuracy*)	**85.82%**

Metrik Evaluasi	Hasil (%)
Presisi (*Precision*)	**94.48%**
Recall	**76.08%**
F1-Score	**84.29%**

5. Analisa Hasil

Berdasarkan data hasil pengujian di atas, dapat dianalisis beberapa poin penting:

1. **Keseimbangan Model (*Nash Equilibrium*):** Nilai akurasi **85.82%** menunjukkan kondisi pelatihan yang sangat sehat. Dalam GAN, akurasi 100% menandakan kegagalan (Generator gagal menipu), sedangkan 50% menandakan Discriminator bingung. Angka 85% membuktikan bahwa Discriminator cukup pintar untuk membedakan, namun Generator juga cukup kompeten untuk menipu Discriminator sebesar ~15-20% dari waktu pengujian.
2. **Efektivitas *Split Learning Rate*:** Strategi menurunkan *Learning Rate* Generator menjadi 0.0001 (setengah dari Discriminator) terbukti sukses meningkatkan Presisi hingga **94.48%**. Hal ini memaksa Generator untuk belajar fitur wajah yang lebih "meyakinkan" agar bisa lolos dari deteksi Discriminator yang ketat.
3. **Dampak *Label Smoothing*:** Penggunaan label **0.98** berhasil mencegah *overfitting*. Grafik *Loss* (lihat lampiran) menunjukkan fluktuasi yang dinamis tanpa adanya saturasi gradien (Loss tidak pernah *stuck* di 0 atau 100).
4. **Kualitas Visual:** Secara kualitatif, gambar yang dihasilkan pada Epoch 50 memiliki struktur wajah anime yang jelas (mata, mulut, rambut) dan tidak mengalami *mode collapse*.

6. Kesimpulan

Implementasi DCGAN untuk pembangkitan wajah anime telah berhasil dilakukan. Penerapan strategi **Split Learning Rate** dan **Label Smoothing (0.98)** terbukti efektif dalam menjaga stabilitas pelatihan, yang dibuktikan dengan capaian akurasi model sebesar **85.82%** dan F1-Score **84.29%** dalam waktu komputasi yang efisien (33 menit). Model mampu menghasilkan variasi wajah anime sintetis yang menyerupai karakteristik dataset asli tanpa mengalami kegagalan pelatihan.

7. Referensi

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 27, 2014, pp. 2672–2680. [Online]. Available: https://papers.nips.cc/paper/5423-generative-adversarial-nets

[3] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2016. [Online]. Available: https://arxiv.org/abs/1511.06434

[4] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Advances in Neural Information Processing Systems*, vol. 29, 2016, pp. 2234–2242. [Online]. Available: https://arxiv.org/abs/1606.03498

[5] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 6626–6637. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf

[6] S. Rakshit, "Anime Face Dataset," *Kaggle*, 2019. [Online]. Available: https://www.kaggle.com/datasets/soumikrakshit/anime-faces

8. Kontribusi & Distribusi Anggota Kelompok

- **[Daffa Akmal Ayom Pamungkas]:**

  Langkah 1: Konfigurasi dan set up (Menjelaskan library PyTorch yang dipakai).

  .Langkah 2: Persiapan data ekstrak dan load

  Menjelaskan codingan DataLoader.

  Menjelaskan proses Resize((64, 64)) dan Normalize (sesuai jobdesc "Preprocessing").

  Menampilkan contoh data batch pertama

- **[Muhammad Tegar Revolusi Seto]:**

  Langkah 3: Arsitektur model generator dan diskriminator (Menjelaskan kode implementasi NetG dan NetD sesuai jobdesc "Implementasi kode utama").

  Langkah 4: Inisialisasi dan optimizer (Menjelaskan bagian Split Learning Rate dan Label Smoothing sesuai jobdesc "Tuning Hyperparameter").

- **[Muhamad Fikih Rizaldi]:**

  Langkah 5: Training (Menjelaskan *looping* epoch, karena jobdesc-nya "Menjalankan eksperimen").

  Langkah 7: Simpan model dan tes ulang (Bagian teknis output eksperimen).

  Langkah 8: Simpan gambar ke folder result (Sesuai jobdesc "Mendokumentasikan hasil").

- **[Redomas Baegy Hardiantahan]:**

  Langkah 6: Evaluasi hasil dan grafik (Membaca makna grafik Biru/Oranye, bukan sekadar menampilkannya, tapi menganalisisnya sesuai jobdesc "Analisis hasil").

  Langkah 9: Rekap akurasi dll (Menyimpulkan angka Akurasi 85.82% dan Presisi 94% menjadi sebuah kesimpulan akhir apakah project berhasil atau tidak).

LAMPIRAN

Video :

https://drive.google.com/drive/folders/1xlqtYEBJuZObfUCNXYqZ0HvSNsKzdU6Z?usp=sharing

Github:

https://github.com/bagoongyoo/Artificial_Intellegence.git

ZeroGpt :