# BAB I

# GENERATIVE ADVERSIAL NETWORK (GAN)

G

enerative Adversarial Networks (GAN) adalah salah satu inovasi paling signifikan dalam bidang kecerdasan buatan dan pembelajaran mendalam (deep learning). Dengan dua jaringan yang saling bersaing—Generator dan Discriminator—GAN membuka peluang baru dalam generasi data sintetis, penciptaan gambar realistis, dan berbagai aplikasi lainnya yang sebelumnya tidak mungkin dicapai dengan metode konvensional. Bab ini membahas tentang konsep dasar GAN, bagaimana GAN berbeda dari algoritma pembelajaran mendalam lainnya, dan komponen utama dari jaringan GAN.

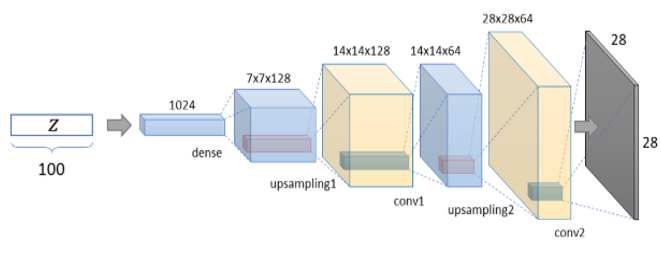
## Konsep Gan

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 .

### Generator

Bentuk jaringan Generator dapat dilihat berkebalikan dengan struktur jaringan saraf pada umumnya. Jaringan Generator menerima input sebuah vektor angka z, kemudian mengubahnya menjadi output gambar tiga dimensi. Vektor input z umumnya dibangkitkan secara acak, lalu dari angka sembarang tersebut Generator membangkitkan gambar yang juga sembarang.

Tugas Generator adalah dilatih untuk bisa membuat sekumpulan gambar palsu, yang saat dilihat oleh Discriminator, Discriminator tidak bisa membedakan antara asli dan palsunya. Gambar 2 adalah contoh model jaringan generator. Input Generator adalah sebuah array yang dibangkitkan secara random, disebut “seed”, pada gambar seed adalah 1x100. Generator akan mengubahnya menjadi sebuah gambar berukuran pixel NxN, pada gambar adalah 28x28 menggunakan Convolutional Neural Network.



Gambar 1. 1 Model Jaringan Generator

1. Arsitektur dan Komponen Generator

Arsitektur Generator dalam Generative Adversarial Networks (GAN) berperan penting dalam menghasilkan data yang realistis. Generator bertindak sebagai model pembuat, yang bertujuan untuk menciptakan data baru yang menyerupai data asli. Untuk mencapai hal ini, arsitektur Generator sering kali menggunakan lapisan-lapisan jaringan saraf yang bertujuan untuk mengubah vektor acak (sering disebut *latent vector* atau *noise*) menjadi keluaran yang diinginkan, seperti gambar atau data lain. Lapisan-lapisan ini membantu dalam melakukan *upsampling* pada vektor noise untuk menghasilkan sebuah gambar. Biasanya, generator terdiri dari beberapa komponen berikut:

1. **Lapisan Input**: Generator dimulai dengan lapisan input yang menerima sebuah vektor noise acak, yang biasanya diambil dari distribusi normal atau seragam. Vektor ini berfungsi sebagai *seed* atau titik awal untuk proses generasi data.
2. **Lapisan Fully Connected**: Pada tahap awal jaringan, lapisan *fully connected* digunakan untuk mentransformasikan vektor noise input menjadi bentuk yang sesuai untuk pemrosesan lebih lanjut.
3. **Batch Normalization**: Teknik ini sering digunakan antara lapisan-lapisan untuk menstabilkan proses pembelajaran dengan menormalisasi output dari lapisan sebelumnya. Ini membantu dalam mengatasi masalah seperti *mode collapse* dan mempercepat konvergensi.
4. **Fungsi Aktivasi**: ReLU (Rectified Linear Unit) atau Leaky ReLU adalah pilihan umum untuk fungsi aktivasi di generator. Fungsi-fungsi ini memperkenalkan non-linearitas ke dalam model, memungkinkan model untuk menghasilkan data yang lebih kompleks.
5. **Lapisan Konvolusi Transposisi**: Juga dikenal sebagai lapisan dekonvolusi (meskipun ini kurang tepat), lapisan ini sangat penting dalam generator. Mereka melakukan *upsampling* pada input dari lapisan sebelumnya ke dimensi spasial yang lebih tinggi, yang secara efektif melakukan kebalikan dari apa yang dilakukan lapisan konvolusi dalam sebuah CNN.
6. **Lapisan Reshaping:** Lapisan ini digunakan untuk membentuk ulang data menjadi format output yang diinginkan, terutama ketika menghasilkan gambar.
7. **Lapisan Output**: Lapisan terakhir biasanya menggunakan fungsi aktivasi *tanh* atau *sigmoid*, tergantung pada jenis data yang dihasilkan. Untuk generasi gambar, fungsi *tanh* sering digunakan untuk mengeluarkan nilai piksel dalam rentang yang dinormalisasi.