Chapter 17: Representation Learning dan Generative Learning Menggunakan Autoencoders dan GANs

Pendahuluan

Chapter ini menyelami dua area yang sangat menarik dalam *unsupervised learning*: Representation Learning dan Generative Learning. *Representation learning* bertujuan untuk secara otomatis menemukan cara yang efisien dan bermakna untuk merepresentasikan data (belajar fitur), sedangkan *generative learning* bertujuan untuk melatih model yang mampu menghasilkan data baru yang realistis dan mirip dengan data pelatihan. Kita akan membahas dua keluarga model utama untuk tugas-tugas ini: Autoencoders dan Generative Adversarial Networks (GANs).

1. Autoencoders untuk Representation Learning

Autoencoder (AE) adalah sebuah Jaringan Saraf Tiruan (JST) yang dilatih untuk merekonstruksi inputnya sendiri. Meskipun terdengar sepele, dengan memberikan batasan tertentu pada jaringan, kita dapat memaksanya untuk mempelajari representasi data yang menarik di lapisan tersembunyinya.

Arsitektur Inti:

Sebuah AE terdiri dari dua bagian utama:

1. Encoder: Jaringan saraf yang menerima data input x dan mengubahnya menjadi representasi berdimensi lebih rendah yang disebut *latent representation* atau *coding*, dinotasikan sebagai z. Proses ini dapat ditulis sebagai z = f(x).
2. Decoder: Jaringan saraf yang mengambil *coding* z dan mencoba merekonstruksi data input asli. Proses ini dapat ditulis sebagai x' = g(z).

Tujuan pelatihan adalah untuk membuat output rekonstruksi x' semirip mungkin dengan input asli x. Ini dicapai dengan meminimalkan reconstruction loss, biasanya *Mean Squared Error* (MSE) atau *Binary Cross-Entropy*.

Varian-Varian Autoencoder

- Stacked Autoencoder: Ini adalah AE dalam (deep) dengan beberapa lapisan tersembunyi di dalam encoder dan decoder. Biasanya, arsitekturnya simetris. Lapisan tersembunyi pusatnya (coding layer) memiliki dimensi paling kecil.
- Denoising Autoencoder: Varian ini tidak dilatih pada data asli, melainkan pada versi data yang telah diberi noise (misalnya, dengan menambahkan noise Gaussian atau menghilangkan beberapa piksel secara acak). Namun, target rekonstruksinya tetap data asli yang bersih. Dengan memaksa model untuk membersihkan noise, ia belajar untuk mengekstrak fitur yang lebih robust dan bermakna, bukan hanya sekadar menyalin input.
- Sparse Autoencoder: Varian ini menambahkan sebuah kendala sparsity pada *coding layer*. Tujuannya adalah untuk memaksa agar sebagian besar neuron di *coding layer*

menjadi tidak aktif (outputnya mendekati nol) untuk setiap input. Hal ini dapat dicapai dengan menambahkan *term* regularisasi pada *loss function*, seperti regularisasi L1 pada aktivasi *coding layer*, atau dengan meminimalkan *KL divergence* antara distribusi aktivasi neuron dan distribusi target yang *sparse*. Ini memaksa model untuk merepresentasikan setiap input menggunakan kombinasi kecil dari fitur-fitur khusus.

2. Variational Autoencoders (VAEs) sebagai Model Generatif

Berbeda dengan AE standar yang hanya belajar untuk merekonstruksi, Variational Autoencoder (VAE) adalah model generatif, artinya ia dapat menghasilkan data baru.

Perbedaan Utama dengan AE Standar:

- Probabilistik: VAE adalah *probabilistic autoencoder*. Alih-alih memetakan input ke satu titik di ruang laten, encoder VAE memetakan input ke sebuah distribusi probabilitas di atas ruang laten. Biasanya, distribusi ini diasumsikan sebagai Gaussian.
- Output Encoder: Encoder tidak menghasilkan vektor z secara langsung. Sebaliknya, ia menghasilkan dua vektor: vektor rata-rata μ (mu) dan vektor log-varians $\log(\sigma^2)$ (log dari sigma kuadrat).
- Sampling dengan Reparameterization Trick: Sebuah titik laten z kemudian diambil sampelnya (*sampled*) dari distribusi Gaussian N(μ, σ). Agar proses *backpropagation* dapat mengalir melalui langkah sampling yang acak ini, digunakanlah reparameterization trick: $z = \mu + \sigma \otimes \varepsilon$ di mana ε (epsilon) adalah tensor acak dari distribusi normal standar, dan ⊗ adalah perkalian per elemen. Dengan cara ini, keacakan dipisahkan dari parameter model, memungkinkan gradien mengalir melalui μ dan σ.

Fungsi Loss VAE:

Fungsi loss VAE terdiri dari dua komponen:

1. Reconstruction Loss: Sama seperti AE biasa, bagian ini mendorong VAE untuk merekonstruksi inputnya dengan baik.
2. Latent Loss (KL Divergence): Ini adalah *term* regularisasi yang memaksa distribusi yang dipelajari oleh encoder agar sedekat mungkin dengan distribusi target (biasanya distribusi normal standar). Hal ini membuat ruang laten menjadi terstruktur dan kontinu, yang sangat penting untuk proses generasi.

Setelah VAE dilatih, kita dapat menghasilkan data baru dengan mengambil sampel vektor z secara acak dari distribusi normal standar dan memberikannya ke decoder.

3. Generative Adversarial Networks (GANs)

GANs adalah kelas model generatif yang sangat kuat dan menghasilkan sampel yang seringkali sangat realistis. GAN didasarkan pada permainan antara dua jaringan saraf yang saling bersaing.

Para Pemain:

1. Generator (G): Tujuannya adalah untuk menghasilkan data palsu yang tidak bisa dibedakan dari data asli. Ia menerima input berupa vektor noise acak dan menghasilkan output yang memiliki struktur sama dengan data asli (misalnya, sebuah gambar).
2. Discriminator (D): Tujuannya adalah untuk membedakan antara data asli dari set pelatihan dan data palsu yang dibuat oleh Generator. Pada dasarnya, ini adalah sebuah *classifier* biner.

Proses Pelatihan Adversarial:

Pelatihan GAN adalah proses dinamis di mana kedua jaringan dilatih secara bergantian:

1. Melatih Discriminator:
    - Generator menghasilkan satu *batch* data palsu.
    - Discriminator dilatih pada *batch* campuran yang berisi data asli (dengan label 1) dan data palsu (dengan label 0). Tujuannya adalah untuk memaksimalkan akurasinya dalam membedakan keduanya.
2. Melatih Generator:
    - Selama fase ini, bobot Discriminator dibekukan.
    - Generator menghasilkan satu *batch* data palsu.
    - Data palsu ini dilewatkan ke Discriminator, tetapi kali ini kita memberinya label target 1 (asli).
    - *Loss* dihitung berdasarkan seberapa baik Generator berhasil "menipu" Discriminator. Gradien dari *loss* ini kemudian disebarkan kembali (*backpropagated*) hanya untuk memperbarui bobot Generator.

Proses ini terus berlanjut. Seiring waktu, Generator menjadi semakin baik dalam membuat data palsu, dan Discriminator menjadi semakin baik dalam mendeteksinya. Titik ekuilibrium tercapai ketika Generator menghasilkan data palsu yang begitu realistis sehingga Discriminator tidak bisa lagi membedakannya dari data asli (akurasi sekitar 50%).

Meskipun sangat kuat, pelatihan GAN terkenal sulit dan tidak stabil, dengan masalah umum seperti *mode collapse* (Generator hanya menghasilkan variasi sampel yang sangat terbatas).