

BAB 2 LANDASAN TEORI

2.1 Keamanan Siber (*Cyber Security*)

Keamanan siber atau *cyber security* merupakan perlindungan sistem, jaringan, dan program dari serangan digital [11]. Serangan-serangan ini dilakukan dengan tujuan untuk mengakses dan mengubah informasi sensitif, memeras uang dari pengguna, atau mengganggu proses bisnis normal. Keamanan siber mencakup serangkaian teknologi, proses, dan kontrol yang dirancang untuk melindungi aset digital dari berbagai ancaman siber. Menurut *International Organization for Standardization*, keamanan siber adalah upaya untuk melindungi dunia digital serta aset-aset yang dimiliki pengguna (*user*) dan infrastruktur dari berbagai risiko serangan siber yang mengancam [12].

2.1.1 Ancaman dan Serangan dalam Keamanan Siber

Ancaman dalam dunia siber terus berkembang secara eksponensial seiring dengan kemajuan teknologi. Serangan siber yang paling umum meliputi [13]:

1. *Malicious Software* (Malware): *Software* intrusif yang bersifat berbahaya dan dirancang untuk merusak sistem [14]. Contoh nyata dari ancaman ini ialah banyaknya aplikasi malware yang ada pada playstore, salah satu contoh dari aplikasi ini ialah aplikasi 'SOEX' yang menyamar sebagai aplikasi *chatting* biasa, namun dalam aplikasinya terdeteksi bahwa adanya malware yang disebut "SparkKitty". Malware ini mencuri gambar dari galeri foto perangkat yang terinfeksi. Beberapa versi dari malware ini bahkan menggunakan Google ML Kit OCR untuk mendeteksi dan hanya mengunggah gambar yang mengandung teks [15].
2. **Phishing**: Penipuan untuk mendapatkan informasi sensitif dengan menyamar sebagai pihak tepercaya [16]. Contoh nyata dari ancaman ini ialah serangan terhadap akademisi terkenal dan kritikus Rusia dengan cara para grup peretas mengirimkan beberapa email kepada para korban dengan menyamar sebagai pejabat Departemen Luar Negeri AS [17].
3. *Denial-of-service* (DoS): Serangan yang bertujuan untuk membuat akses jaringan tidak tersedia bagi pengguna yang sah dikarenakan *overload*,

serangan ini dilakukan dengan cara *flooding* data melalui jaringan internet [18]. Contoh nyata dari ancaman ini ialah ketika stasiun TV milik pemerintah Iran (IRIB) diretas dan siaran reguler diinterupsi dan digantikan dengan video-video yang berisi cuplikan unjuk rasa menentang pemerintah Iran [19].

4. *Man-in-the-middle* (MitM): serangan di mana penyerang secara diam-diam mencegat komunikasi antara dua pihak. Contoh nyata dari ancaman ini ialah pada Maret 2019, UC Browser dan UC Browser Mini untuk Android memiliki fitur tersembunyi. Fitur ini memungkinkan perusahaan untuk mengunduh modul atau pustaka baru dari server mereka dan menginstalnya di perangkat pengguna. Masalah utamanya adalah proses pengunduhan ini dilakukan melalui protokol HTTP yang tidak aman, bukan melalui HTTPS yang terenkripsi [20].
5. Hoaks dan Disinformasi: Serangan yang berfokus pada penyebaran informasi palsu atau menyesatkan dengan tujuan untuk menipu, memanipulasi opini publik, atau menciptakan kepanikan. Berbeda dengan *deepfake* yang menggunakan AI untuk menciptakan konten sintetis, hoaks seringkali berupa teks, gambar, atau video yang diedit secara sederhana atau disajikan di luar konteks aslinya [21]. Ancaman ini menjadi sangat signifikan karena kemudahan penyebarannya melalui media sosial dan aplikasi pesan instan, yang dapat menjangkau audiens luas dalam waktu singkat.

Dalam konteks disinformasi dan manipulasi, kemunculan *deepfake* menjadi ancaman serangan siber baru yang signifikan. *Deepfake* adalah konten sintetis yang dihasilkan oleh AI di mana seseorang dalam gambar atau video yang ada digantikan dengan kemiripan orang lain. *Audio deepfake*, secara spesifik, merupakan manipulasi atau pembuatan rekaman audio untuk meniru suara seseorang, yang dapat digunakan untuk tujuan jahat seperti penipuan, pencemaran nama baik, atau penyebaran berita palsu [22].

2.1.2 Peran AI dan Machine Learning dalam Keamanan Siber

Artificial Intelligence (AI) dan *Machine Learning* (ML) memainkan peran krusial dalam memperkuat pertahanan siber. AI dengan menggunakan algoritma ML dapat mendeteksi serangan siber dengan belajar berdasarkan dari pola data lalu lintas jaringan historis mengenai serangan siber sehingga dapat mengidentifikasi ancaman baru dan anomali dalam skala besar [23]. Kemampuan ini menjadikan AI

sebagai alat yang sangat efektif untuk mengotomatisasi respons terhadap insiden keamanan siber dan meningkatkan efisiensi analisis keamanan.

2.1.3 Konsep Dasar Deepfake dan Audio Deepfake

Istilah *deepfake* adalah gabungan dari kata *deep learning* dan *fake*. Ini merujuk pada media sintesis di mana seseorang dalam gambar atau video yang ada digantikan dengan kemiripan orang lain. *Audio deepfake*, yang juga dikenal sebagai *voice cloning* atau *deep voice*, adalah penerapan teknologi serupa untuk menghasilkan audio sintesis [24]. Dengan menggunakan model *deep learning*, sistem dapat dilatih pada sampel suara seseorang untuk kemudian menghasilkan ucapan baru dengan suara orang tersebut. Teknologi ini seringkali mengandalkan arsitektur seperti *autoencoders* atau *Generative Adversarial Networks (GANs)* untuk mencapai tingkat realisme yang tinggi.

2.1.4 Dampak Deepfake dalam Keamanan Siber

Deepfake, khususnya *audio deepfake*, menimbulkan ancaman serius terhadap keamanan siber khususnya pada bidang hoaks dan disinformasi [25]. Teknologi ini memungkinkan pembuatan rekaman suara palsu yang sangat meyakinkan, yang dapat digunakan untuk berbagai tujuan jahat. Dampaknya antara lain:

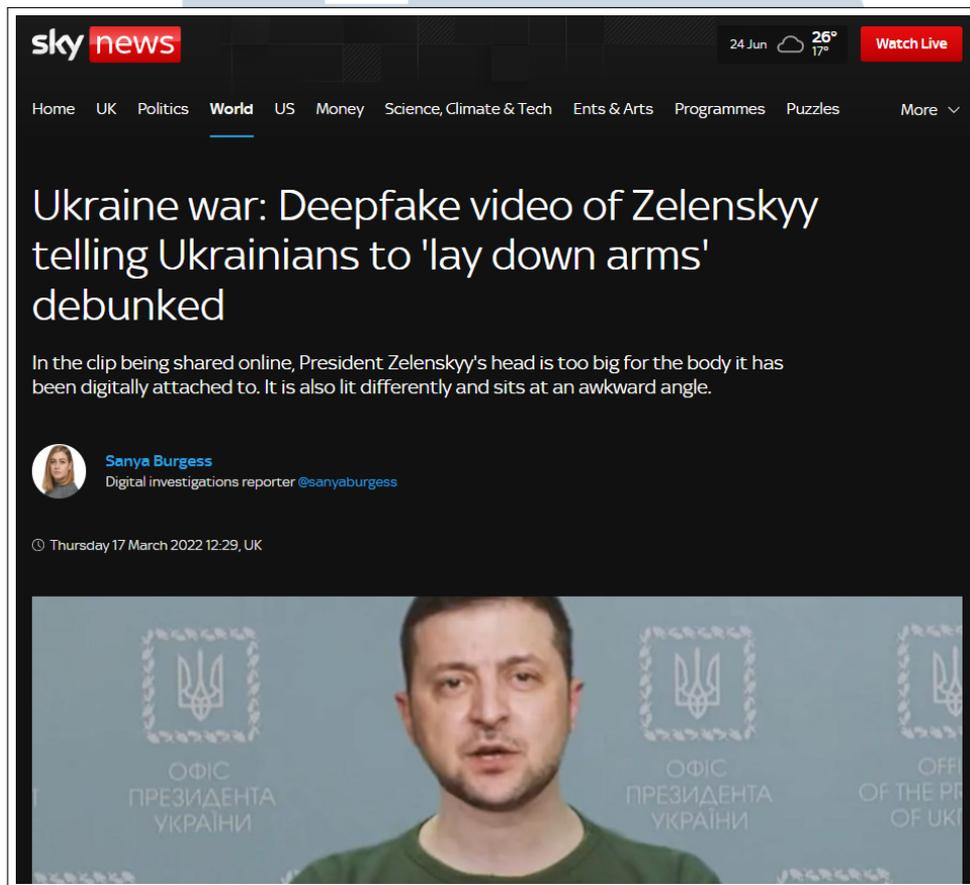
1. Penipuan dan Rekayasa Sosial: Pelaku dapat meniru suara eksekutif perusahaan untuk mengotorisasi transfer dana ilegal.



Gambar 2.1. Contoh Nyata Penipuan dan Rekayasa Sosial

Gambar 2.1 merupakan contoh nyata yang terjadi ketika suara CEO sebuah perusahaan energi yang berbasis di Inggris ditiru menggunakan AI untuk menipu dan mentransfer dana sebesar \$243.000 ke rekening penipu di Hungaria [4].

2. Disinformasi dan Propaganda: Audio palsu dapat dibuat untuk menyebarkan berita bohong dan memanipulasi opini publik.



Gambar 2.2. Contoh Nyata Disinformasi dan Propaganda

Gambar 2.2 merupakan contoh dari disinformasi dalam bentuk video *deepfake* Presiden Ukraina, Zelenskyy, yang muncul pada awal invasi Rusia, di mana ia seolah-olah menyatakan Ukraina menyerah dan kalah dalam perang [26].

3. Pencemaran Nama Baik: Dengan adanya deepfake, individu dapat dijadikan target untuk merusak reputasi mereka.



Gambar 2.3. Contoh Nyata Pencemaran Nama Baik

Gambar 2.3 merupakan contoh sebuah audio *deepfake* dari Walikota London, Sadiq Khan, disebar di media sosial. Audio tersebut meniru suaranya untuk membuat pernyataan kontroversial mengenai acara peringatan perang, yang bertujuan untuk memicu kemarahan publik dan merusak citranya menjelang pemilihan [27].

4. Erosi Kepercayaan: Meluasnya *deepfake* dapat menyebabkan erosi kepercayaan masyarakat terhadap media digital. Fenomena ini menciptakan apa yang disebut sebagai "dividen pembohong" (*liar's dividend*), di mana keberadaan teknologi ini memungkinkan pelaku kejahatan untuk menyangkal bukti audio atau video yang asli dengan mengklaimnya sebagai rekayasa [28].

UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 2.4. Contoh Nyata Erosi Kepercayaan

Gambar 2.4 merupakan contoh nyata dari ancaman ini. Munculnya *robocall* dengan suara Presiden AS Joe Biden yang dihasilkan oleh AI untuk mencegah pemilih memberikan suara dalam pemilihan pendahuluan di New Hampshire [29].

2.2 Artificial Intelligence

Artificial Intelligence (AI) atau Kecerdasan Buatan adalah salah satu cabang ilmu komputer yang berfokus pada pembangunan mesin cerdas yang mampu melakukan tugas-tugas yang pada umumnya memerlukan kecerdasan manusia[30]. Istilah ini pertama kali dicetuskan oleh John McCarthy pada Konferensi Dartmouth, yang menjadi tonggak kelahiran AI sebagai bidang studi formal. Menurut Russell dan Norvig, tujuan utama AI dapat dikategorikan ke dalam empat pendekatan:

1. Sistem yang berpikir seperti manusia.
2. Sistem yang bertindak seperti manusia.
3. Sistem yang berpikir secara rasional.
4. Sistem yang bertindak secara rasional.

Pendekatan terakhir, yaitu agen rasional yang dapat bertindak untuk mencapai hasil terbaik, menjadi definisi standar yang paling dominan dalam studi AI modern [31].

Sebagai sebuah bidang yang luas, AI memiliki berbagai sub-bidang. Sub-bidang yang relevan saat ini ialah Machine Learning (ML), yang memberikan kemampuan pada komputer untuk belajar dari data tanpa diprogram secara eksplisit. Di dalam ML, terdapat lagi sub-bidang yang disebut Deep Learning (DL), yang menggunakan arsitektur jaringan saraf tiruan berlapis-lapis (deep neural networks) untuk mempelajari pola-pola yang sangat kompleks dari data yang berjumlah besar. Keberhasilan Deep Learning dalam tugas seperti pengenalan gambar, pemrosesan bahasa alami, dan sintesis suara telah menjadi motor penggerak revolusi AI saat ini dan merupakan teknologi inti di balik pembuatan serta deteksi deepfake [32].

2.2.1 *Machine Learning*

Machine Learning (ML) atau Pembelajaran Mesin adalah sub-bidang dari Artificial Intelligence yang secara spesifik berfokus pada pengembangan algoritma yang memungkinkan sistem komputer untuk "belajar" dari data. Berbeda dengan pemrograman tradisional yang memerlukan instruksi eksplisit untuk setiap tugas, pendekatan ML adalah membangun model matematis dari data sampel—yang dikenal sebagai "data pelatihan"—untuk membuat prediksi atau keputusan secara otomatis [33]. Esensinya, ML adalah tentang menciptakan program yang kinerjanya dalam suatu tugas dapat meningkat seiring dengan bertambahnya pengalaman atau data. Kemampuan untuk mengidentifikasi pola, bahkan dalam kumpulan data yang sangat besar dan kompleks, menjadikan ML sebagai fondasi bagi banyak aplikasi AI modern, mulai dari sistem rekomendasi hingga kendaraan otonom.

Terdapat tiga paradigma utama dalam Machine Learning yang dibedakan berdasarkan sifat data pelatihan dan mekanisme umpan baliknya [34].

1. *Supervised Learning*: Algoritma belajar dari data yang telah diberi label input-output yang benar, dengan tujuan untuk memetakan input baru ke output yang sesuai. Contohnya adalah klasifikasi email sebagai spam atau bukan spam.
2. *Unsupervised Learning*: Algoritma bekerja dengan data yang tidak berlabel dan mencoba menemukan struktur atau pola tersembunyi di dalamnya, seperti mengelompokkan pelanggan berdasarkan perilaku pembelian.
3. *Reinforcement Learning*: sebuah "agen" belajar untuk membuat keputusan dengan cara berinteraksi dengan lingkungannya. Agen tersebut menerima

imbalan (reward) atau hukuman (punishment) atas tindakannya, dengan tujuan untuk memaksimalkan total imbalan dari waktu ke waktu, seperti dalam melatih AI untuk memainkan sebuah permainan.

2.2.2 *Deep Learning*

Deep Learning (DL) adalah spesialisasi dalam *machine learning* yang menggunakan jaringan saraf tiruan (*Artificial Neural Networks*) dengan banyak lapisan—arsitektur ”dalam” atau *deep*—untuk memodelkan dan memahami pola-pola yang sangat kompleks dalam data. Berbeda dengan model ML konvensional yang mungkin memerlukan rekayasa fitur manual, arsitektur hierarkis DL memungkinkan model untuk secara otomatis mempelajari representasi data dari tingkat yang paling sederhana hingga yang paling abstrak. Setiap lapisan dalam jaringan belajar untuk mengenali fitur pada tingkat yang berbeda, di mana lapisan awal mungkin mendeteksi fitur sederhana seperti tepi atau warna, dan lapisan yang lebih dalam menggabungkannya untuk mengenali objek yang kompleks seperti wajah atau mobil [35].

Kekuatan utama *Deep Learning* terletak pada kemampuannya untuk menangani data tidak terstruktur dalam skala besar, yang menghasilkan terobosan di berbagai bidang. Arsitektur yang paling menonjol termasuk *Convolutional Neural Networks* (CNNs), yang sangat efektif untuk data spasial seperti gambar karena kemampuannya mengenali pola secara lokal, dan *Recurrent Neural Networks* (RNNs), yang dirancang untuk menangani data sekuensial seperti teks atau deret waktu dengan memanfaatkan memori internal. Model-model ini dan variannya telah mencapai kinerja canggih dalam tugas-tugas seperti pengenalan gambar, pemrosesan bahasa alami, dan sintesis audio, menjadikannya teknologi fundamental di balik aplikasi AI modern yang paling transformatif [32].

A **Convolutional Neural Network (CNN)**

Convolutional Neural Network (CNN atau ConvNet) adalah kelas khusus dari jaringan saraf tiruan yang menjadi arsitektur dominan dalam berbagai tugas *computer vision* [36]. Desain CNN terinspirasi oleh korteks visual biologis, di mana neuron individual hanya merespons pada area terbatas yang disebut *Receptive Field*. Keunggulan utama CNN terletak pada kemampuannya untuk secara otomatis dan adaptif mempelajari hierarki fitur dari data input, seperti gambar [37].

Arsitektur CNN pada umumnya terdiri dari tiga jenis lapisan utama:

1. Lapisan Konvolusi (*Convolutional Layer*): Ini adalah inti dari CNN. Lapisan ini menerapkan serangkaian filter (atau *kernel*) yang dapat dipelajari ke data input. Setiap filter bergeser (berkonvolusi) di seluruh area input untuk menghasilkan *feature map*. Proses ini memungkinkan filter untuk mendeteksi fitur-fitur lokal tertentu seperti tepi, sudut, tekstur, atau warna. Jaringan belajar nilai-nilai dalam filter ini selama proses pelatihan.
2. Lapisan Pooling (*Pooling Layer*): Lapisan ini biasanya ditempatkan setelah lapisan konvolusi dan berfungsi untuk mengurangi dimensi spasial (*downsampling*) dari *feature map*. Pengurangan ini membuat representasi fitur menjadi lebih ringkas, mengurangi jumlah parameter dan beban komputasi, serta memberikan invariansi terhadap translasi kecil pada input. Jenis yang paling umum adalah *Max Pooling*.
3. Lapisan Terhubung Penuh (*Fully Connected Layer*): Setelah beberapa lapisan konvolusi dan pooling mengekstraksi fitur-fitur tingkat tinggi, hasilnya akan diratakan menjadi vektor satu dimensi dan dimasukkan ke dalam lapisan terhubung penuh. Lapisan ini berfungsi untuk melakukan tugas klasifikasi akhir berdasarkan fitur-fitur yang telah diekstraksi.

Hubungan antara CNN dan GAN menjadi sangat penting dengan munculnya DCGAN (*Deep Convolutional Generative Adversarial Networks*). Sebelum DCGAN, arsitektur Generator dan Diskriminator pada GAN seringkali hanya menggunakan *Multi-Layer Perceptrons* (MLP), yang membatasi kemampuannya untuk bekerja pada data gambar yang kompleks [38]. DCGAN merevolusi ini dengan menggabungkan kekuatan arsitektur CNN ke dalam kerangka kerja GAN:

1. Diskriminator dibangun sebagai CNN klasifikasi standar. Tujuannya adalah untuk mengambil gambar (asli atau palsu) sebagai input dan menghasilkan probabilitas apakah gambar tersebut asli.
2. Generator dibangun sebagai CNN yang dimodifikasi (sering disebut *deconvolutional network* atau jaringan konvolusi terbalik), yang melakukan proses kebalikan dari diskriminator. Ia mengambil vektor noise acak sebagai input dan secara bertahap melakukan *upsampling* melalui lapisan-lapisan konvolusi terbalik (*transposed convolutions*) untuk menghasilkan sebuah gambar baru.

Generator mampu menghasilkan data yang tidak dapat dibedakan dari data asli oleh Diskriminator.

Pemilihan fungsi aktivasi pada kedua jaringan ini bersifat krusial dan berbeda, disesuaikan dengan tujuan spesifik masing-masing. Generator, sebagai "seniman", menggunakan aktivasi ReLU pada lapisan dalamnya untuk efisiensi pembelajaran dan aktivasi Tanh pada lapisan *output*-nya. Fungsi Tanh memastikan data yang dihasilkan berada dalam rentang $[-1, 1]$, sesuai dengan normalisasi data asli. Di sisi lain, Diskriminator, sebagai "kritikus", menggunakan aktivasi LeakyReLU pada lapisan dalamnya. LeakyReLU mencegah masalah "neuron mati" dan memastikan gradien atau *feedback* yang stabil selalu mengalir ke Generator, sehingga proses pelatihan menjadi lebih stabil. Pada lapisan *output*-nya, Diskriminator menggunakan aktivasi Sigmoid untuk menghasilkan nilai probabilitas antara $[0, 1]$, yang merepresentasikan keyakinan apakah sebuah data asli atau palsu [40].

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))] \quad (2.1)$$

Persamaan 2.1 pada dasarnya adalah sebuah fungsi nilai (*value function*) $V(D, G)$ yang mengilustrasikan sebuah permainan kompetitif. Persamaan ini mendefinisikan tujuan dari arsitektur GAN dalam sebuah kerangka permainan min-max. Hubungan antara kedua pemain—yaitu Diskriminator (D) dan Generator (G)—bersifat adversarial dan dapat dianalogikan sebagai *zero-sum game*, di mana keuntungan satu pemain adalah kerugian bagi pemain lainnya.

Tujuan dari permainan ini dijelaskan oleh operator $\min_G \max_D$: Diskriminator (D) berusaha untuk memaksimalkan (*max*) nilai fungsi $V(D, G)$, sementara Generator (G) secara bersamaan berusaha untuk meminimalkan (*min*) nilai fungsi tersebut. Berikut adalah rincian tujuan masing-masing pemain:

A Tujuan Diskriminator (D): Memaksimalkan Fungsi Nilai

Diskriminator dirancang untuk menjadi detektif yang sempurna. Tujuannya adalah membuat nilai $V(D, G)$ sebesar mungkin (nilai optimalnya adalah 0). Hal ini dicapai dengan cara:

1. Untuk data asli (\mathbf{x}): Diskriminator harus sangat yakin bahwa data tersebut

asli. Ini berarti outputnya, $D(x)$, harus mendekati 1. Akibatnya, suku pertama dari persamaan, $\log D(x)$, akan mendekati $\log(1) = 0$.

2. Untuk data palsu ($G(z)$): Diskriminator harus sangat yakin bahwa data tersebut palsu. Ini berarti outputnya, $D(G(z))$, harus mendekati 0. Akibatnya, suku kedua dari persamaan, $\log(1 - D(G(z)))$, akan mendekati $\log(1 - 0) = 0$.

Ketika Diskriminator berada pada performa puncaknya, kedua suku dalam persamaan menjadi 0, dan ia berhasil memaksimalkan fungsi nilai.

B Tujuan Generator (G): Meminimalkan Fungsi Nilai

Generator, sebaliknya, dirancang untuk menjadi pemalsu yang sempurna. Tujuannya adalah membuat nilai $V(D, G)$ sekecil mungkin (menuju negatif tak terhingga). Generator tidak dapat memengaruhi suku pertama persamaan (yang berurusan dengan data asli). Oleh karena itu, ia hanya fokus pada suku kedua dengan cara:

1. Menghasilkan data palsu ($G(z)$) yang sangat realistis sehingga mampu menipu Diskriminator.
2. Tujuannya adalah membuat output Diskriminator, $D(G(z))$, mendekati 1 (Diskriminator salah mengira data palsu sebagai data asli).
3. Ketika $D(G(z))$ mendekati 1, maka nilai $(1 - D(G(z)))$ akan mendekati 0. Akibatnya, suku kedua persamaan, $\log(1 - D(G(z)))$, akan mendekati $\log(0)$, yaitu $-\infty$.

Dengan demikian, keberhasilan Generator dalam menipu Diskriminator akan secara langsung meminimalkan fungsi nilai, sesuai dengan tujuannya.

C Kaitan dengan Arsitektur Penelitian

Dalam konteks penelitian ini, arsitektur Diskriminator DCGAN tidak dilatih secara adversarial melawan sebuah Generator. Sebaliknya, model ini dilatih sebagai sebuah *binary classifier* standar, menggunakan dataset yang telah berisi sampel audio asli ('real') dan audio palsu ('fake').

Pemilihan arsitektur ini didasarkan pada prinsip bahwa struktur Diskriminator DCGAN secara inheren dirancang untuk menjadi ekstraktor

fitur yang sangat efektif dalam membedakan data asli dari data sintesis. Fitur-fitur desainnya, seperti penggunaan *strided convolutions* untuk *downsampling* dan aktivasi LeakyReLU, terbukti andal dalam menangkap pola dan artefak visual pada spektrogram yang seringkali menjadi penanda data buatan.

Dengan kata lain, penelitian ini mengadopsi sebuah arsitektur yang kekuatannya telah terbukti dalam domain generatif, dan menerapkannya secara efektif pada tugas klasifikasi langsung. Kekuatan model ini bukan berasal dari "pengalaman bertarung" dalam penelitian ini, melainkan dari efektivitas desain arsitekturalnya untuk tugas identifikasi anomali pada data visual.

2.2.4 *Deep Convolutional Generative Adversarial Networks (DCGANs)*

Deep Convolutional Generative Adversarial Networks (DCGANs) adalah evolusi dari arsitektur GANs konvensional yang mengintegrasikan *Convolutional Neural Networks (CNNs)* dengan tujuan untuk menambahkan aspek *image recognition* CNNs ke dalam struktur Generator dan Diskriminator [38]. Penggunaan lapisan konvolusional membuat DCGANs sangat efektif dalam tugas-tugas yang melibatkan data spasial seperti gambar, atau representasi visual dari audio seperti spektrogram.

Perbedaan utama antara DCGANs dan GANs konvensional terletak pada arsitekturnya. DCGANs memperkenalkan beberapa modifikasi kunci untuk menstabilkan proses pelatihan, antara lain:

1. Mengganti lapisan *pooling* dengan *strided convolutions* (pada Diskriminator) dan *fractional-strided convolutions* (pada Generator).
2. Menggunakan *Batch Normalization* di kedua jaringan untuk menstabilkan pembelajaran.
3. Menggunakan aktivasi ReLU di semua lapisan Generator kecuali lapisan *output* yang menggunakan Tanh, dan aktivasi LeakyReLU di semua lapisan Diskriminator.

Implementasi DCGANs telah terbukti berhasil dalam berbagai bidang, termasuk sintesis gambar berkualitas tinggi [41], dan dalam konteks penelitian ini, berpotensi untuk deteksi audio melalui representasi spektrogramnya.

2.2.5 Spektrogram dan Transformasi Sinyal Audio

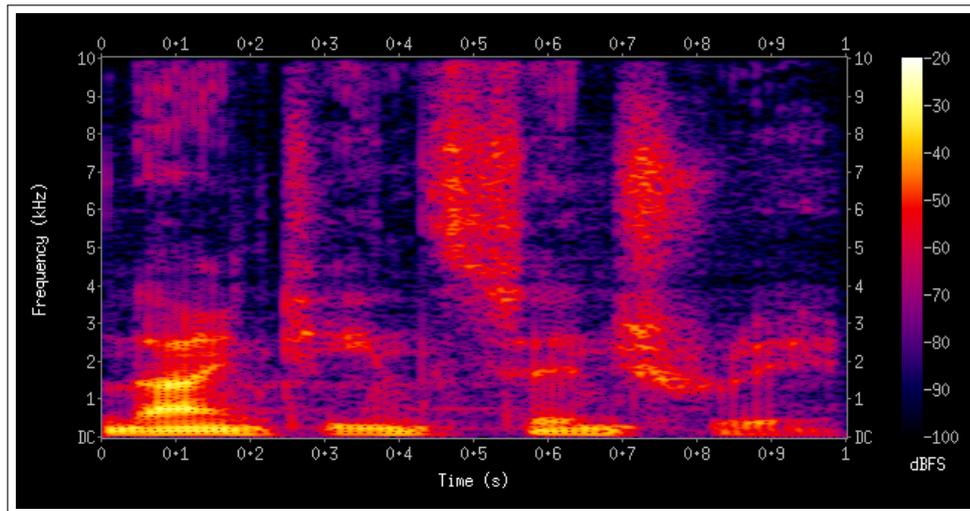
Tantangan fundamental dalam analisis audio menggunakan model *deep learning* modern adalah perbedaan dalam dimensi data. Sinyal audio secara alami berbentuk data deret waktu satu dimensi (1D), yang merepresentasikan perubahan amplitudo seiring waktu. Di sisi lain, arsitektur *Convolutional Neural Networks* (CNNs) yang sangat efektif untuk ekstraksi fitur, dirancang untuk beroperasi pada data dua dimensi (2D) seperti gambar [42]. Oleh karena itu, diperlukan sebuah metode untuk mengubah sinyal 1D audio menjadi representasi 2D yang kaya akan fitur tanpa kehilangan informasi temporal yang krusial.

Metode standar untuk menganalisis kandungan frekuensi dari sebuah sinyal adalah dengan menggunakan *Fourier Transform*. Namun, *Fourier Transform* konvensional memiliki keterbatasan signifikan untuk sinyal audio: ia menganalisis keseluruhan sinyal secara serentak, sehingga menghasilkan spektrum frekuensi rata-rata dan kehilangan semua informasi tentang kapan frekuensi tertentu muncul. Sinyal audio bersifat non-stasioner, artinya properti statistiknya (termasuk frekuensi) berubah seiring waktu. Mengetahui frekuensi apa saja yang ada tanpa tahu kapan frekuensi itu muncul membuat analisis menjadi tidak berguna [43].

Untuk mengatasi masalah ini, digunakanlah *Short-Time Fourier Transform* (STFT). STFT adalah metode fundamental untuk ekstraksi fitur audio yang dapat menjawab "frekuensi apa yang muncul dan kapan itu muncul?". STFT bekerja dengan cara:

1. Memecah sinyal audio yang panjang menjadi segmen-segmen pendek yang saling tumpang tindih (*overlapping frames*).
2. Menerapkan *Fourier Transform* pada setiap segmen pendek tersebut secara individual untuk menganalisis kandungannya.
3. Menggabungkan hasil dari setiap segmen secara berurutan untuk membangun representasi 2D.

Hasil dari proses STFT inilah yang disebut spektrogram.



Gambar 2.6. Representasi audio secara visual (Spektrogram)

Spektrogram adalah representasi visual di mana sumbu horizontal merepresentasikan waktu, sumbu vertikal merepresentasikan frekuensi, dan intensitas warna pada setiap titik merepresentasikan amplitudo atau energi dari frekuensi tertentu pada waktu tertentu [44], yang dimana contoh dari spektrogram dapat dilihat pada gambar 2.6.

Proses STFT ini diatur oleh beberapa parameter kunci:

1. `n_fft`: Merupakan jumlah sampel yang digunakan dalam satu jendela analisis untuk melakukan *Fast Fourier Transform* (FFT). Nilai yang lebih besar akan memberikan resolusi frekuensi yang lebih baik (lebih detail dalam sumbu y), namun dengan mengorbankan resolusi waktu.
2. `hop_length`: Merupakan jumlah sampel antara awal dari dua jendela analisis yang berurutan. Nilai yang lebih kecil akan menghasilkan lebih banyak tumpang tindih antar jendela, yang memberikan resolusi waktu yang lebih baik (lebih detail dalam sumbu x) tetapi dengan biaya komputasi yang lebih tinggi.

Meskipun spektrogram standar sudah menyediakan representasi waktu-frekuensi, sumbu frekuensinya bersifat linear. Skala linear ini tidak sesuai dengan cara pendengaran manusia memproses suara, di mana manusia lebih peka terhadap perubahan pada frekuensi rendah daripada frekuensi tinggi. Untuk menciptakan representasi fitur yang lebih efektif dan relevan secara perseptual untuk tugas-tugas yang berkaitan dengan suara manusia, seperti deteksi deepfake, maka sumbu frekuensi linear dari spektrogram ini diubah menjadi skala Mel.

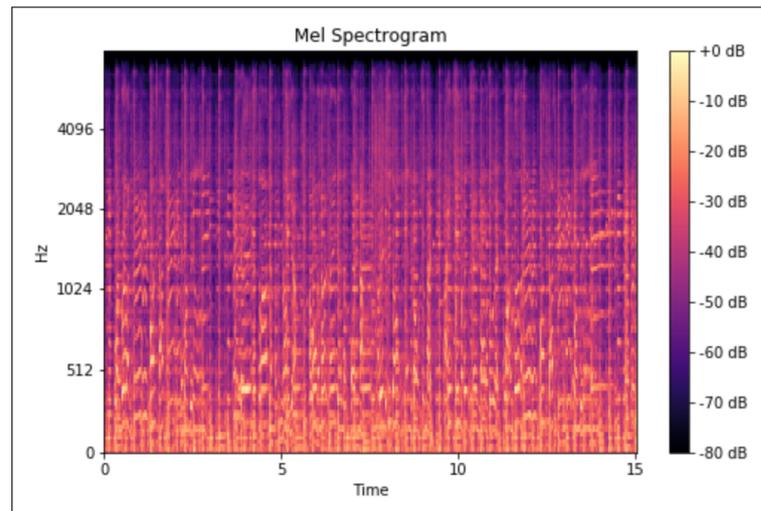
2.3 Mel-Spektrogram

Mel-Spektrogram adalah varian dari spektrogram yang memetakan sumbu frekuensi ke dalam skala Mel. Skala Mel adalah skala perseptual dari (*pitch*) yang didasarkan pada cara telinga manusia memproses suara. Skala ini memberikan penekanan lebih pada perubahan di frekuensi rendah, di mana pendengaran manusia lebih sensitif, daripada di frekuensi tinggi. Penggunaan skala Mel membuat representasi audio lebih ringkas dan fokus pada fitur-fitur yang paling relevan bagi persepsi manusia, yang telah terbukti sangat efektif untuk meningkatkan kinerja model *machine learning* dalam berbagai tugas audio [33]. Proses ini diatur oleh parameter `n_mels`, yang menentukan jumlah pita Mel (*Mel bands*) yang akan digunakan. Jumlah ini secara efektif menentukan tinggi (jumlah baris piksel) dari gambar spektrogram akhir.

$$m = 2595 \log_{10} \left(1 + \frac{f}{700} \right) \quad (2.2)$$

Rumus 2.2 merupakan proses pembuatan Mel-Spektrogram dimulai dengan spektrogram standar yang dihasilkan oleh STFT. Kemudian, spektrum daya tersebut dipetakan ke skala Mel dengan konversi dari frekuensi dalam Hertz (f) ke skala Mel (m). Rumus 2.2 mengubah frekuensi dari skala linier (Hertz) menjadi skala Mel yang bersifat perseptual. Berikut adalah penjelasan komponennya:

1. m : Nilai frekuensi dalam skala Mel.
2. f : Nilai frekuensi dalam Hertz (Hz).
3. 2595 dan 700: Konstanta ini digunakan untuk menyelaraskan skala agar 1000 Hz sama dengan 1000 Mel. Rumus ini meniru respons non-linier dari telinga manusia terhadap frekuensi.



Gambar 2.7. Bentuk Mel-Spektrogram

Gambar 2.7 merupakan contoh dari audio yang telah dikonversi menjadi Mel-Spektrogram. Mel-Spektrogram memberikan bobot lebih pada frekuensi yang lebih penting secara perseptual, sehingga menghasilkan representasi fitur yang lebih kuat untuk analisis audio [45].

2.4 Teknik Deteksi Audio Deepfake

Deteksi *audio deepfake* adalah proses mengidentifikasi apakah sebuah rekaman audio asli atau sintesis (deepfake).

1. Metode Konvensional: Metode awal berfokus pada analisis fitur akustik tingkat rendah seperti frekuensi fundamental, *formants*, dan *Mel-frequency Cepstral Coefficients* (MFCCs). Namun, metode ini seringkali gagal mendeteksi audio yang dihasilkan oleh model sintesis canggih.
2. Metode Berbasis *Deep Learning*: Pendekatan modern memanfaatkan model *deep learning* seperti CNNs dan RNNs untuk belajar fitur-fitur pembeda secara otomatis dari representasi data audio, seperti spektrogram mentah [46]. DCGANs memiliki keuntungan dalam konteks deteksi audio dikarenakan diskriminator dari arsitektur DCGAN yang telah dilatih dapat digunakan sebagai ekstraktor fitur dan sebagai classifier itu sendiri. Diskriminator dilatih untuk menjadi ahli dalam menemukan ke tidak konsisten-an sekecil apa pun yang membedakan data palsu dari data asli, ia dapat menjadi alat deteksi yang sangat sensitif dan efektif.

2.5 Dataset untuk Pelatihan Model DCGANs

Kualitas dan kuantitas dataset sangat menentukan kinerja model *deep learning*. Untuk tugas deteksi *audio deepfake*, dataset yang ideal harus berisi dua jenis data:

1. Audio Asli (*Bonafide*): Rekaman suara manusia asli dari berbagai penutur, kondisi rekaman, dan bahasa.
2. Audio Palsu (*Spoof*): Audio yang dihasilkan oleh berbagai teknik sintesis suara atau *voice conversion*.

Dataset yang digunakan pada penelitian ini ialah 'The Fake-or-Real (FoR) Dataset (deepfake audio)' pada folder *for-original* yang terdiri dari 11916 audio asli dan 21529 audio palsu [47].

Sebelum dimasukkan ke dalam model DCGANs, data audio mentah (*raw audio*) harus melalui tahap *preprocessing*. Proses yang paling umum adalah mengubah sinyal audio satu dimensi menjadi representasi dua dimensi seperti spektrogram atau *mel-spectrogram*. Representasi visual ini memungkinkan model berbasis CNN untuk memproses data audio seolah-olah itu adalah gambar. Tantangan utama dalam pengumpulan dataset adalah kurangnya variasi dalam teknik serangan sintesis dan memastikan data mencakup berbagai kondisi dunia nyata.

2.6 Evaluasi Kinerja Model

Untuk mengevaluasi kinerja model deteksi *audio deepfake*, beberapa metrik dan teknik standar digunakan. Metrik ini mengukur seberapa baik model dapat membedakan antara audio asli dan palsu.

A Metrik Evaluasi

Untuk mengukur kinerja model klasifikasi biner secara kuantitatif, beberapa metrik standar digunakan. Metrik-metrik ini dihitung berdasarkan empat kemungkinan hasil prediksi yang dirangkum dalam *confusion matrix*: *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).

1. *Accuracy*: Mengukur rasio total prediksi yang benar terhadap keseluruhan jumlah data. Ini memberikan gambaran umum tentang seberapa sering model

membuat prediksi yang tepat.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.3)$$

2. *Precision*: Mengukur tingkat ketepatan dari prediksi positif. Metrik ini menjawab pertanyaan: "Dari semua yang diprediksi sebagai *fake*, berapa persen yang benar-benar *fake*?"

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2.4)$$

3. *Recall* (Sensitivity): Mengukur kemampuan model untuk menemukan semua sampel positif. Metrik ini menjawab: "Dari semua audio *fake* yang ada, berapa persen yang berhasil terdeteksi?"

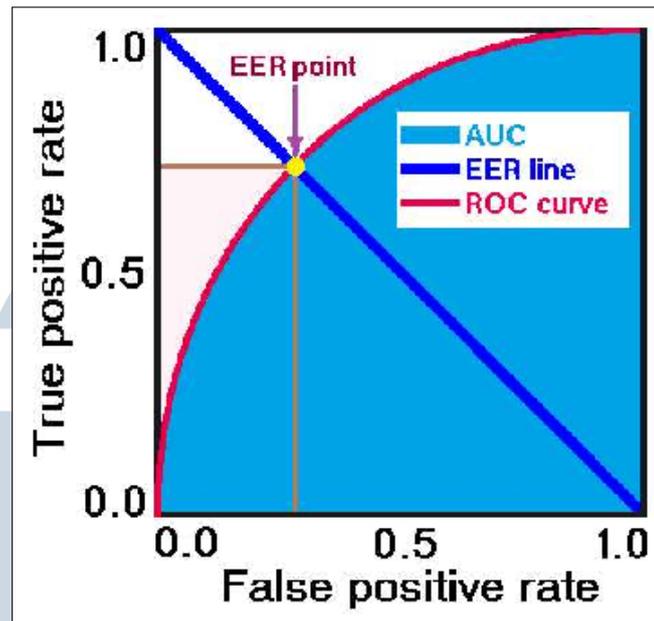
$$\text{Recall} = \frac{TP}{TP + FN} \quad (2.5)$$

4. F1-Score: Merupakan rata-rata harmonik dari *Precision* dan *Recall*. Metrik ini berguna untuk menyeimbangkan kedua metrik tersebut, terutama jika ada ketidakseimbangan kelas atau jika kedua jenis kesalahan (FP dan FN) sama-sama penting.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (2.6)$$

5. *Confusion Matrix*: Sebuah tabel yang memvisualisasikan performa model dengan menunjukkan jumlah prediksi yang benar dan salah (TP, TN, FP, FN) untuk setiap kelas.

6. *Equal Error Rate* (EER): Merupakan sebuah metrik untuk mengukur performa sistem klasifikasi biner pada titik di mana kedua jenis kesalahan memiliki nilai yang sama.



Gambar 2.8. Estimasi EER dari ROC Curve

Berdasarkan metode dari gambar 2.8, EER ditentukan dari kurva ROC (*Receiver Operating Characteristic*) yang memplot *True Positive Rate* (TPR) terhadap *False Positive Rate* (FPR). Dalam konteks ini, FPR sama dengan *False Acceptance Rate* (FAR), sedangkan *False Rejection Rate* (FRR) dihitung sebagai $1 - \text{TPR}$. EER adalah titik di mana nilai FAR sama dengan FRR, yang secara konseptual merupakan titik potong antara kurva FAR dan kurva FRR. Metrik ini umum digunakan dalam sistem verifikasi biometrik dan deteksi *spoofing* untuk menunjukkan titik keseimbangan terbaik dari performa model [48]. Semakin rendah nilai EER, semakin baik kinerja model.

2.7 Teknologi dan Konsep Pendukung

Implementasi penelitian ini didukung oleh serangkaian teknologi, kerangka kerja, dan pustaka perangkat lunak yang menjadi fondasi bagi alur kerja teknis, mulai dari pemrosesan data hingga pelatihan model.

2.7.1 Bahasa Pemrograman dan Lingkungan

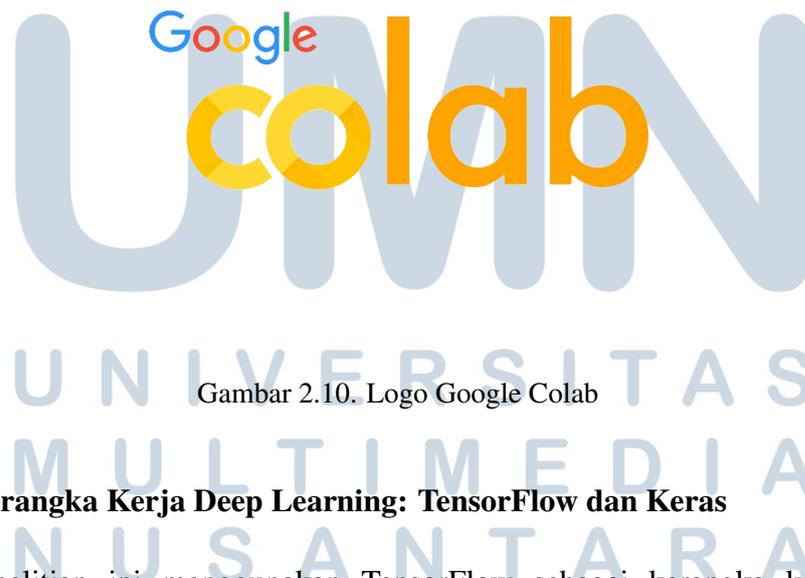
1. Python: Seluruh kode dalam penelitian ini dibangun menggunakan bahasa pemrograman Python. Python dipilih karena merupakan standar industri

untuk pengembangan kecerdasan buatan, didukung oleh ekosistem yang matang dan ketersediaan pustaka ilmiah yang sangat luas [49]. Gambar 2.9 merupakan logo dari bahasa pemrograman python.



Gambar 2.9. Logo Python

2. Google Colaboratory: Eksperimen dijalankan pada platform Google Colab untuk memanfaatkan sumber daya komputasi berbasis cloud. Keunggulan utamanya adalah akses gratis ke unit pemrosesan grafis (GPU), yang secara signifikan mempercepat waktu pelatihan model *deep learning*. Gambar 2.10 merupakan logo dari lingkungan yang digunakan, google colab.



Gambar 2.10. Logo Google Colab

2.7.2 Kerangka Kerja Deep Learning: TensorFlow dan Keras

Penelitian ini menggunakan TensorFlow sebagai kerangka kerja *deep learning* utama yang dimana logonya terdapat pada gambar 2.11. TensorFlow adalah platform *open-source* yang dikembangkan oleh Google untuk komputasi numerik berperforma tinggi. Ia menyediakan ekosistem komprehensif untuk

membangun dan menerapkan aplikasi *machine learning* [50]. Di atas TensorFlow, penelitian ini memanfaatkan Keras, yaitu API tingkat tinggi yang terintegrasi erat yang dimana logonya terdapat pada gambar 2.12. Keras menyederhanakan proses pembangunan model dengan menyediakan antarmuka yang intuitif dan modular, seperti `Sequential` API untuk mendefinisikan lapisan-lapisan model secara berurutan, serta fungsi-fungsi seperti `compile()` dan `fit()` untuk mengonfigurasi dan menjalankan proses pelatihan dengan mudah.



Gambar 2.11. Logo Tensorflow



Gambar 2.12. Logo Keras

2.7.3 Pustaka Pendukung Utama

Beberapa pustaka Python lainnya memainkan peran penting dalam alur kerja penelitian ini:

1. Librosa: Pustaka khusus untuk analisis audio. Dalam penelitian ini, Librosa sangat krusial untuk: (a) memuat file audio dari disk ke dalam larik NumPy (`librosa.load`), (b) mengubah sinyal audio 1D menjadi representasi 2D melalui *Short-Time Fourier Transform* (STFT), dan (c) memetakan sumbu frekuensi STFT ke skala Mel untuk menghasilkan Mel-spektrogram (`librosa.feature.melspectrogram`). Logo dari pustaka librosa dapat dilihat pada gambar 2.13.



Gambar 2.13. Logo Librosa

2. NumPy: Merupakan pustaka fundamental untuk komputasi numerik. Spektrogram yang dihasilkan pada dasarnya adalah larik NumPy. Operasi seperti normalisasi, penskalaan, dan manipulasi data sebelum dimasukkan ke model sangat bergantung pada kecepatan dan efisiensi NumPy. Logo dari pustaka NumPy dapat dilihat pada gambar 2.14.



Gambar 2.14. Logo NumPy

3. Scikit-learn: Pustaka ini menyediakan alat-alat penting untuk alur kerja *machine learning* standar. `train_test_split` digunakan untuk membagi dataset secara terstratifikasi (menjaga rasio kelas). Setelah pelatihan, `classification_report` digunakan untuk menghasilkan metrik evaluasi yang komprehensif (presisi, recall, f1-score), dan `roc_curve` digunakan untuk menghitung Tingkat Kesalahan Setara (EER). Logo dari pustaka Scikit-learn dapat dilihat pada gambar 2.15.



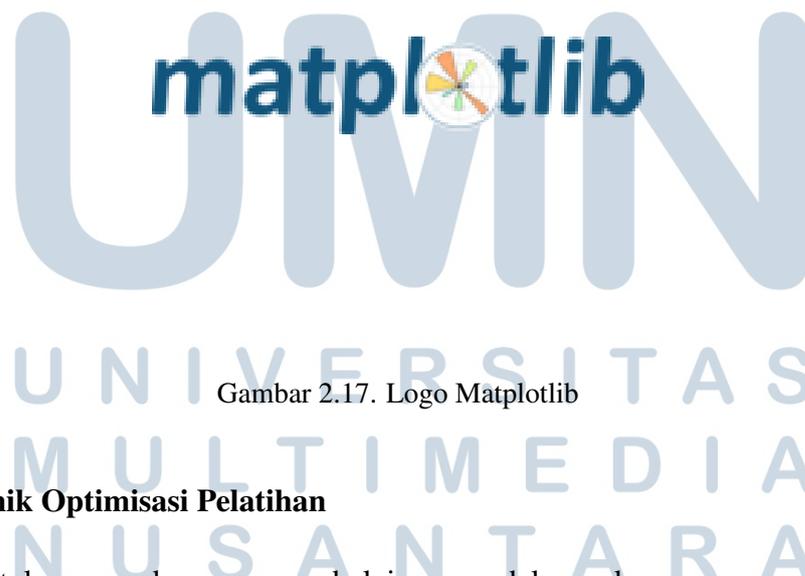
Gambar 2.15. Logo Scikit-learn

4. OpenCV (`cv2`): Berfokus pada pemrosesan gambar, peran utama OpenCV di sini adalah untuk memanipulasi spektrogram seolah-olah itu adalah gambar. Fungsi `cv2.resize` digunakan untuk memastikan bahwa setiap spektrogram, terlepas dari variasi kecil dalam dimensinya, diubah ukurannya secara seragam menjadi 128x128 piksel agar sesuai dengan ukuran input model. Logo dari pustaka OpenCV dapat dilihat pada gambar 2.16.



Gambar 2.16. Logo OpenCV

5. Matplotlib: Pustaka ini digunakan untuk visualisasi hasil setelah proses pelatihan selesai. Secara spesifik, `matplotlib.pyplot` digunakan untuk mem-plot grafik akurasi dan *loss* dari data histori pelatihan, yang memungkinkan analisis visual terhadap fenomena seperti *overfitting*. Logo dari pustaka `librosa` dapat dilihat pada gambar 2.17.



Gambar 2.17. Logo Matplotlib

2.8 Teknik Optimisasi Pelatihan

Untuk memandu proses pembelajaran model agar konvergen secara efisien dan mampu melakukan generalisasi dengan baik, beberapa teknik optimisasi diterapkan.

1. Optimizer AdamW: Optimizer adalah algoritma yang bertugas untuk memodifikasi atribut dari jaringan saraf, seperti bobot (*weights*) dan laju pembelajaran (*learning rate*), guna mengurangi nilai *loss* secara keseluruhan. Penelitian ini menggunakan AdamW, sebuah varian dari optimizer Adam yang populer. Keunggulan utama AdamW terletak pada caranya menangani *weight decay* (regularisasi L2). AdamW memisahkan (*decouple*) pembaruan *weight decay* dari pembaruan gradien, yang terbukti lebih efektif dan stabil dalam membantu model melakukan generalisasi dan mencegah *overfitting* dibandingkan dengan implementasi pada Adam standar [51].
2. Callbacks: Selama proses pelatihan, beberapa teknik *callback* Keras digunakan untuk memantau dan mengontrol perilaku pelatihan secara dinamis. *Callback* adalah fungsi yang dieksekusi pada berbagai tahap dalam proses pelatihan (misalnya, di akhir setiap *epoch*) [52].
 - (a) **EarlyStopping**: Sebuah *callback* yang menghentikan proses pelatihan secara otomatis jika metrik yang dipantau (misalnya, *validation loss*) tidak menunjukkan perbaikan setelah sejumlah *epoch* tertentu (*patience*). Ini membantu mencegah *overfitting* yang parah dan menghemat waktu komputasi.
 - (b) **ReduceLRonPlateau**: *Callback* ini mengurangi laju pembelajaran (*learning rate*) ketika sebuah metrik berhenti membaik. Menurunkan laju pembelajaran memungkinkan model untuk melakukan penyesuaian yang lebih halus pada bobotnya, yang dapat membantu menemukan minimum yang lebih baik dan keluar dari *plateau* (stagnasi) performa.

2.9 Studi Literatur dan Riset Terkait

2.9.1 C. Gohen, G. Raja (2022): Generative Adversarial Networks

Generative Adversarial Networks (GANs) adalah salah satu metode terkemuka dalam pembelajaran mesin yang digunakan untuk mempelajari distribusi data dan menghasilkan contoh data baru dengan kualitas tinggi. Teknologi ini telah mengalami perkembangan pesat sejak pertama kali diperkenalkan dan telah diterapkan dalam berbagai bidang seperti pengolahan citra, sintesis teks, dan peningkatan data (data augmentation).

GANs terdiri dari dua komponen utama, yaitu generator dan discriminator, yang bekerja secara adversarial (saling berlawanan). Generator berusaha menciptakan data yang menyerupai data asli, sementara discriminator bertugas membedakan antara data asli dan data yang dihasilkan oleh generator. Proses ini membentuk permainan min-max yang memungkinkan GANs untuk menghasilkan data sintesis dengan karakteristik yang semakin realistis.

Meskipun memiliki keunggulan signifikan, GANs menghadapi beberapa tantangan utama dalam proses pelatihannya, antara lain:

1. Mode Collapse – Generator cenderung menghasilkan variasi data yang terbatas, tidak mencerminkan seluruh distribusi data asli.
2. Vanishing Gradients – Apabila discriminator terlalu kuat, generator akan menerima gradien yang sangat kecil, sehingga sulit untuk meningkatkan kinerjanya.
3. Instabilitas Pelatihan – Parameter model sering kali tidak stabil, menyebabkan kualitas hasil yang bervariasi.

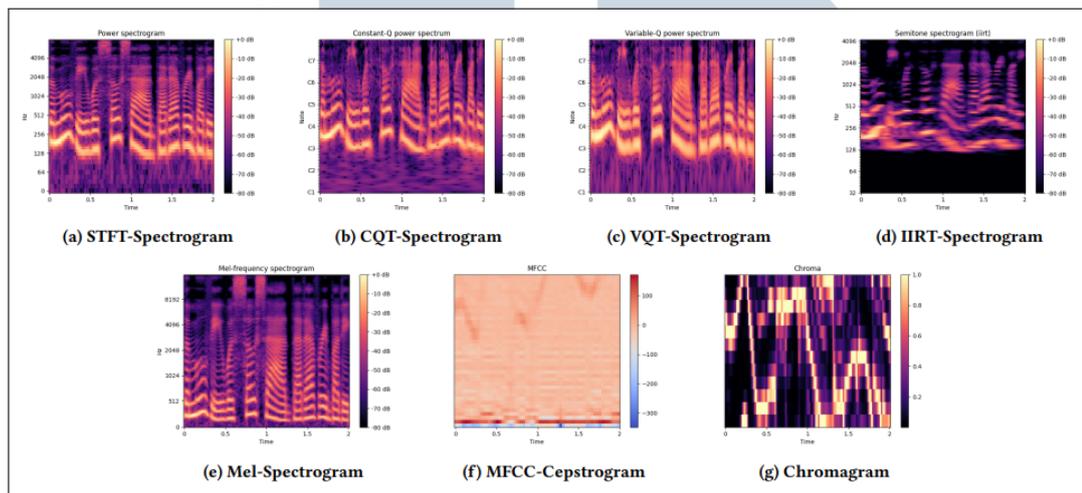
Sejumlah penelitian telah dilakukan untuk mengatasi masalah ini melalui pengembangan berbagai varian arsitektur dan fungsi loss, seperti:

1. Semi-supervised GAN (SGAN) dan Conditional GAN (CGAN) yang memanfaatkan informasi label dalam proses pelatihan.
2. Deep Convolutional GAN (DCGAN) yang menggantikan lapisan fully connected dengan jaringan konvolusi untuk meningkatkan stabilitas dan kualitas hasil.
3. Progressive GAN (PROGAN) yang menggunakan pendekatan pelatihan bertahap untuk menghasilkan gambar dengan resolusi tinggi.
4. BigGAN dan StyleGAN, yang memperkenalkan arsitektur lebih kompleks guna meningkatkan variasi dan kualitas gambar.

2.9.2 F. Anton, M. Kamil, H. Petr (2024): Deepfake Speech Detection: A Spectrogram Analysis

Penelitian ini berfokus pada deteksi audio speech deepfake yang berfokus pada penggunaan spektrogram sebagai input untuk jaringan saraf dalam untuk

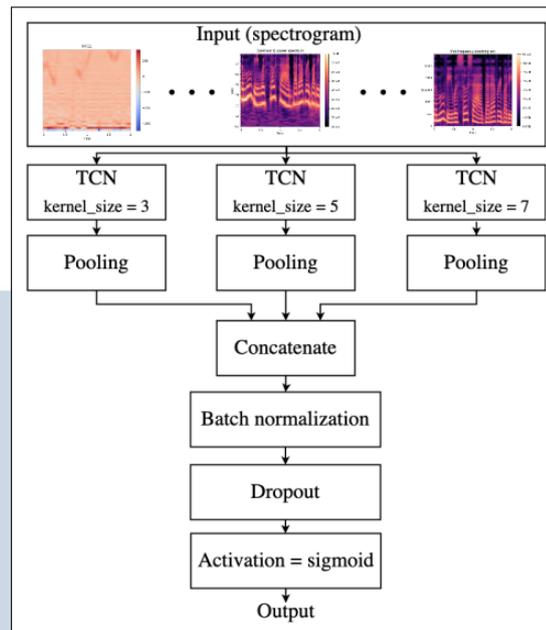
mendeteksi ucapan deepfake. Studi ini bertujuan untuk menganalisis kelayakan berbagai spektrogram untuk deteksi ucapan deepfake dengan membandingkan kinerja, persyaratan perangkat keras, dan kecepatannya. Penelitian ini menggunakan 7 spektrogram yang berbeda, yang dapat dilihat pada gambar 2.18:



Gambar 2.18. 7 Spektrograms yang dipakai

Arsitektur detektor berbasis *Temporal Convolutional Network* (TCN), sebuah jenis arsitektur jaringan saraf yang efektif untuk data sekuensial, dijelaskan dengan menggambarkan penggunaan AveragePooling dan parameter kunci untuk pelatihan jaringan, pengoptimalan, dan evaluasi kinerja. Detektor adalah istilah umum untuk model atau sistem yang dirancang untuk mengidentifikasi keberadaan pola atau sinyal tertentu dalam data, dalam hal ini, untuk mendeteksi audio *deepfake*.

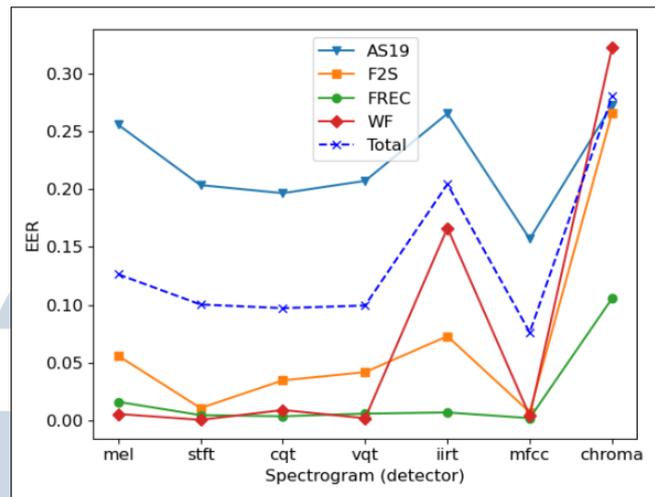
U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



Gambar 2.19. Arsitektur Detektor

Gambar 2.19 merupakan arsitektur dari detektor yang melakukan *pooling* atas beberapa input spektrogram, lalu setelahnya akan dilakukang penggabungan output dari pooling menjadi satu vektor fitur. Setelah digabungkan, vektor fitur tersebut akan di normalisasi dan kemudian diterapkan lapisan akhir, dropout, untuk mencegah *overfitting* dan juga diaktivasi menggunakan sigmoid sehingga menghasilkan *output* akhir. Detektor tersebut diimplementasikan di dalam Python dengan menggunakan TensorFlow framework. Sehingga hasil dari evaluasi dengan Equal Error Rates dari penggunaan spektrogram berdasarkan datasets yang telah digunakan ditampilkan dalam graf berikut:

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



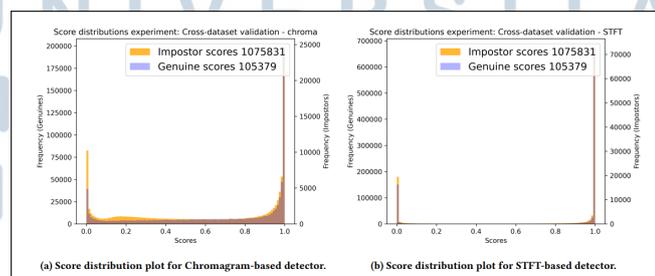
Gambar 2.20. Graph Equal Error Rate

Gambar 2.20 mengindikasikan Equal Error Rate (sumbu y) dari bermacam spektrogram yang digunakan (sumbu x) dengan menggunakan dataset yang direpresentasikan oleh bentuk garis. Lalu dilakukannya gabungan skor antar semua dataset sehingga menghasilkan tabel berikut:

<i>EER (%)</i>						
<i>Mel</i>	<i>STFT</i>	<i>CQT</i>	<i>VQT</i>	<i>IIRT</i>	<i>MFCC</i>	<i>Chroma</i>
45.85	46.25	46.43	45.88	49.05	48.23	46.58

Gambar 2.21. Tabel ERR Spektrograms

Lalu disimpulkannya bahwa detector dengan basis STFT-Spektrograms merupakan salah satu detector dengan performa terbaik dikarenakan performanya terhadap data yang familiar maupun yang tidak familiar sehingga bisa dibilang sebagai detector berbasis spektrogram yang paling optimal.



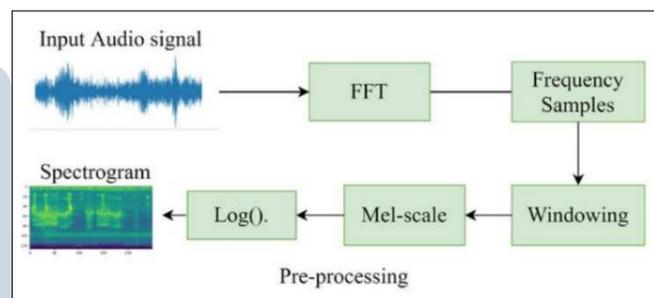
Gambar 2.22. Chroma vs STFT Spektrogram

Gambar 2.22 merupakan perbandingan antara detector yang menggunakan chroma spektrogram dengan detector yang menggunakan STFT spektrogram.

2.9.3 L. Jovelin, G. Bobby, M. Ruji (2024): Spectrogram-Based Analysis and Detection of Deepfake Audio Using Enhanced DCGANs for Secure Content Distribution

Penelitian ini berfokus pada deteksi audio deepfake dengan menggunakan spektrogram dan juga salah satu dari generative adversarial networks (GANs) yaitu DCGANs.

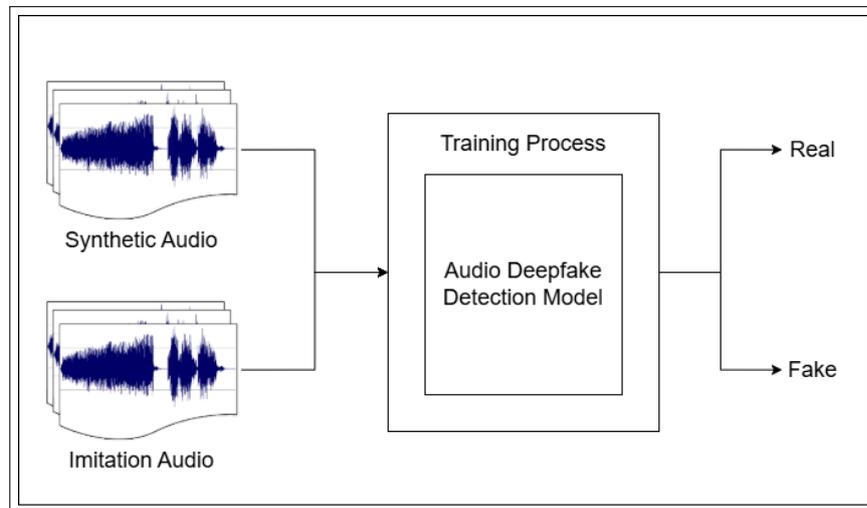
Metode yang digunakan para peneliti ialah memasukkan normalisasi batch ke dalam komponen generator dan diskriminator dari model DCGAN untuk meningkatkan stabilitas dalam membedakan audio nyata dan palsu. Studi ini menggunakan dua kumpulan dataset: satu berisi ucapan manusia asli (didapatkan melalui kaggle) dan yang lain berisi audio deepfake yang dihasilkan melalui Retrieval-based Voice Conversion (RVC). Kumpulan data diproses sebelumnya menggunakan Audacity dan Sonic Visualizer, dan sampel audio dikategorikan ke dalam kelas *real* dan *fake* (FoR). Lalu dilakukannya konversi audio ke spektrogram dengan cara sebagai berikut:



Gambar 2.23. Proses konversi audio ke spektrogram

Proses dimulai dengan pembacaan file ke dalam time-domain signal (sinyal waveforms yang di rekam beriring waktu) dan meng-apply Fast Fourier Transform (FFT) untuk konversi ke bentuk frequency-domain yang dimana magnitude dari sampel menggambarkan amplitudo dari komponen frekuensi sinyal. Setelah itu sinyal dibagikan menjadi bagian tumpang tindih yang dimana setiap bagian dikalikan oleh window function untuk mengurangi kebocoran spektral dan di map ke skala Mel. Akhirnya dilakukan logaritmik transformasi berdasarkan frekuensi Mel yang ada sehingga menghasilkan Mel-Spektrogram.

Berikut merupakan gambaran proses dari audio deepfake detection.



Gambar 2.24. Proses Audio Deepfake Detection

Gambar 2.24 merupakan proses atau cara kerja dari model audio deepfake detection yang dimana model di train dengan menggunakan dua dataset, satu dataset asli dan satu dataset deepfake/imitasi lalu model akan melakukan klasifikasi real (class 1) atau fake (class 0).



2.10 Analisis Keterkaitan dan *Research Gap*

Tabel 2.1. Ringkasan Studi Literatur Terkait

No.	Penelitian (Author, Tahun)	Topik Penelitian (Metode/Model)	Hasil / Kesimpulan
1	C. Gohen & G. Raja (2022)	Tinjauan Arsitektur Generative Adversarial Networks (GANs)	Hasil terbaik untuk klasifikasi dicapai oleh DCGAN (22.48% EER).
2	F. Anton et al. (2024)	Deteksi Audio Deepfake (Analisis Spektrogram dengan TCN)	Hasil terbaik dicapai dengan spektrogram STFT (46.25% EER).
3	L. Jovelin et al. (2024)	Deteksi Audio Deepfake dengan DCGAN	Akurasi 92.86%

Dari tabel 2.1 yang terdiri dari kesimpulan ketiga studi tersebut, dapat diidentifikasi keterkaitan serta celah penelitian (*research gap*) yang menjadi landasan penelitian ini. Penelitian oleh Gohen dan Raja (2022) menyediakan fondasi teoretis mengenai keunggulan arsitektur DCGAN untuk data citra, yang mendukung keputusan metodologis untuk menggunakan spektrogram. Selanjutnya, studi oleh Anton et al. (2024) memberikan validasi empiris bahwa representasi spektrogram berbasis STFT merupakan salah satu fitur paling optimal untuk deteksi audio deepfake, yang memperkuat pemilihan metode pra-pemrosesan dalam penelitian ini.

Penelitian oleh Jovelin et al. (2024) menjadi pembanding metodologis utama karena menggunakan pendekatan serupa. Namun, dari sinilah *research gap* utama diidentifikasi. Meskipun Jovelin et al. telah menggunakan DCGAN, model tersebut belum diimplementasikannya optimisasi regularisasi yang ekstensif untuk mengatasi *overfitting* pada dataset yang berbeda.

Oleh karena itu, penelitian ini berkontribusi dengan mengisi celah tersebut, yaitu dengan menerapkan arsitektur Diskriminator DCGAN yang telah dioptimalkan dengan teknik regularisasi berlapis (kombinasi Dropout, L2, dan AdamW) pada dataset standar *Fake-or-Real* (FoR) dan menganalisis hasilnya secara komprehensif menggunakan metrik akurasi dan *Equal Error Rate* (EER).

2.11 Kesimpulan

Bab ini telah memaparkan landasan teoretis yang diperlukan untuk memahami penelitian ini, mulai dari domain umum keamanan siber hingga konsep teknis spesifik dari *Deep Convolutional Generative Adversarial Networks* (DCGANs). Ancaman yang ditimbulkan oleh *audio deepfake* memerlukan pengembangan metode deteksi yang canggih dan andal. Teori mengenai arsitektur, cara kerja, dan proses pelatihan DCGANs, serta metodologi evaluasi kinerja dan teknologi pendukung, akan menjadi landasan utama dalam perancangan dan implementasi model deteksi *audio deepfake* yang akan dibahas secara rinci pada bab-bab selanjutnya. Keterkaitan antara konsep-konsep ini membentuk dasar yang kuat untuk menjawab rumusan masalah penelitian.

