BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu berisikan kajian literatur yang membahas terkait deteksi kloning suara dan audio deepfake dengan menggunakan *deep learning*, khususnya model berbasis *attention* dan *self-supervised transformer*, penggabungan fitur, serta pengujian lintas bahasa. Kajian ini bertujuan untuk melihat perkembangan performa model, arah pemecahan masalah yang sudah ditempuh, celah yang masih tersisa, serta posisi penelitian yang diusulkan di antara pendekatan-pendekatan tersebut. Kajian dirangkum pada Tabel 2.1 dengan fokus pada hasil metrik evaluasi, konsistensi performa, bentuk arsitektur yang digunakan, dan keterbatasan yang masih tampak pada kondisi lintas domain maupun bahasa non-Inggris.

Model berbasis *attention* menjadi salah satu pendekatan yang kuat karena mampu menyeleksi bagian sinyal yang relevan pada dimensi waktu, frekuensi, maupun hubungan *spectro-temporal*. Model *Dual-Path Time-Frequency Attention Network* (DPTFAN) menunjukkan performa sangat tinggi dengan akurasi 98,94% dan *Equal Error Rate* (EER) 0,35% melalui CNN backbone, *dual-path time-frequency attention*, dan *Pythagorean Hesitant Fuzzy Sets* (PHFS), sehingga relevan sebagai dasar pemodelan artefak akustik pada penelitian ini [14]. Model *Audio Anti-Spoofing using Integrated Spectro-Temporal Graph Attention Networks* (AASIST) juga memperlihatkan kekuatan *graph attention* dengan EER 0,83%, sedangkan versi ringan AASIST-L tetap kompetitif dengan EER 0,99% [10]. Pendekatan *graph attention dual-channel* menghasilkan EER 1,64% pada ASVspoof 2019 dan 6,76% pada ASVspoof 2021 *Logical Access* (LA), yang menunjukkan bahwa relasi antarfitur dapat membantu deteksi, tetapi performanya masih menurun pada kondisi yang lebih sulit [19]. Model 1DResNet dengan *Convolutional Block Attention Module* (CBAM) menghasilkan EER 1,94% pada ASVspoof 2019 dan 4,94% pada *cross-dataset* ASVspoof 2015, sehingga model ringan berbasis attention tetap menjanjikan, tetapi generalisasi lintas dataset belum sepenuhnya stabil [20]. *Enhanced Siamese* CNN dengan *self-attention* dan

StacLoss memperoleh EER 2,95% serta akurasi 98%, sehingga masih kompetitif sebagai pembanding klasifikasi audio palsu [13]. Namun, model Mel dan F0 dengan *cross-attention fusion* memperoleh EER 26,41% pada ASVspoof 2019 dan 28,52% pada ASVspoof 2021, sehingga *attention* atau fusi fitur belum otomatis menghasilkan *robustness* tinggi apabila rancangan model belum cukup mampu menghadapi variasi data [21].

Pendekatan lain yang unggul dalam deteksi kloning suara adalah *self-supervised transformer*, terutama Wav2Vec 2.0. Wav2Vec 2.0 *Xtremely Large-Scale Speech Representation* (XLS-R) yang dipadukan dengan AASIST mencapai EER 0,22%, sehingga menunjukkan bahwa representasi *self-supervised Learning* (SSL) dapat memperkuat sistem *anti-spoofing* ketika digunakan bersama *back-end* yang sesuai [22]. Wav2Vec 2.0 juga memperoleh EER 0,38% pada dataset *Arabic Fake Audio Dataset for Modern Standard Arabic* (AFAD-MSA), tetapi performanya dapat turun drastis pada generator TTS yang tidak terlihat saat pelatihan, sehingga generalisasi terhadap generator baru masih menjadi masalah penting [23]. Kerangka iWAX yang menggabungkan Wav2Vec 2.0, AASIST, dan XGBoost menunjukkan performa kompetitif sekaligus menambahkan sisi interpretabilitas, meskipun kompleksitas *pipeline* menjadi lebih tinggi [24]. Wav2Vec 2.0 menjadi model yang unggul pada jenis *self-supervised transformer* model ini mampu mempelajari representasi ucapan dari data besar dan menangkap informasi temporal, fonetik, serta konteks ucapan yang lebih luas

Temuan-temuan tersebut menunjukkan bahwa performa terbaik umumnya tidak berasal dari satu komponen tunggal, melainkan dari penggabungan beberapa sumber informasi. Model *attention* kuat dalam membaca artefak akustik pada representasi waktu-frekuensi, sedangkan *self-supervised transformer* kuat dalam membaca representasi ucapan yang lebih kontekstual. Dasar ini menjadi alasan penelitian ini menggabungkan DPTFAN dan Wav2Vec 2.0 dalam bentuk model hibrida. DPTFAN digunakan untuk menangkap pola akustik dan artefak pada domain waktu-frekuensi, sedangkan Wav2Vec 2.0 digunakan untuk menangkap pola temporal, fonetik, dan konteks ucapan dari *waveform*. Penggabungan dilakukan pada tingkat skor agar kedua cabang tetap dapat bekerja pada ruang

representasinya masing-masing tanpa harus menyelaraskan fitur internal yang berbeda bentuk dan maknanya.

Strategi penggabungan model harus melihat representasi yang dihasilkan oleh model yang digunakan. Pada konfigurasi dan dataset pengujian yang sama, penggabungan HuBERT dan WavLM dengan berbagai metode fusi menghasilkan nilai metrik yang bervariasi [18]. *Mean score fusion* menghasilkan EER 1,72%, sedangkan varian yang disebut sebagai *more refined score fusion* memperoleh EER 0,63%, lebih rendah dibandingkan feature concatenation sebesar 0,89%, tetapi hanya menyamai performa model tunggal terbaik yang juga memperoleh EER 0,63%. *Attentional Multi-Feature Fusion* memperoleh hasil terbaik dengan EER 0,42%, sehingga hasil tersebut menunjukkan bahwa fusi skor sederhana melalui perataan tidak memberikan peningkatan, sedangkan mekanisme penggabungan skor yang lebih terarah dapat mempertahankan performa model terbaik tanpa menyatukan representasi internal.

Berbagai penelitian memperlihatkan bahwa *robustness* model masih menjadi persoalan utama. Nilai EER rendah pada dataset utama belum selalu bertahan ketika model diuji pada kondisi audio yang dimodifikasi [18], [19], [25]. Oleh karena itu, penelitian ini tidak hanya menggunakan akurasi, tetapi juga EER dan AUC agar evaluasi dapat membaca keseimbangan kesalahan dan kemampuan pemisahan kelas secara lebih menyeluruh. Keterbatasan lain terlihat pada konteks bahasa non-Inggris. Sebagian besar penelitian masih bertumpu pada benchmark internasional seperti ASVspoof, sehingga variasi fonetik, ritme, prosodi, dan karakteristik ujaran dari bahasa lain belum sepenuhnya terwakili. InaSAS menjadi salah satu rujukan utama karena membahas *antispoofing* untuk ujaran Bahasa Indonesia melalui InaSpoof-v1, dengan model AASIST-L menghasilkan EER sebesar 4,19% [11]. Namun hasil ini masih dapat ditingkatkan kembali dengan menerapkan hibrida pada model-model yang sesuai. Hal ini memperkuat posisi penelitian ini, karena deteksi kloning suara Bahasa Indonesia membutuhkan model yang mampu membaca artefak akustik sekaligus struktur ucapan yang sesuai dengan karakteristik bahasa Indonesia.

Tabel 2.1 Tabel Penelitian Terdahulu

Judul	Tahun	Model	Dataset	Hasil	Kelebihan	Kekurangan
Audio Deepfake Detection via a Fuzzy Dual-Path Time-Frequency Attention Network [14]	2025	DPTFAN (CNN *backbone* + *dual-path time-frequency attention* + PHFS)	ASVspoof 2019 *Logical Access* (LA)	Akurasi 98.94%, EER 0.35% (pada dataset ASVspoof 2019 LA	Menggabungkan pemodelan *time–frequency (dual-path attention)* dengan *backbone* CNN sehingga efektif menangkap artefak spektral dan dinamika temporal; performa sangat tinggi di ASVspoof 2019 LA dan FoR, sehingga kuat sebagai referensi arsitektur attention akustik.	Validasi masih didominasi *benchmark* tertentu dan belum diuji ke dataset multilingual atau generator yang benar-benar baru *(out-of-distribution)* di luar pola dataset *benchmark*.
Experimental Study: Enhancing Voice Spoofing Detection Models with Wav2Vec 2.0 [22]	2024	Wav2Vec 2.0 + AASIST	ASVspoof 2019 LA	EER 0.22%, min t-DCF 0.0063	Menunjukkan bahwa representasi SSL Wav2Vec 2.0 bisa meningkatkan *anti-spoofing* secara signifikan; memberi insight praktis tentang pemakaian SSL sebagai front-end.	Penelitian masih belum diuji coba ke bahasa lain. Hasil juga sangat bergantung pada setting eksperimen dan *benchmark* tertentu (ASVspoof 2019 LA) dan belum generalisasi ke domain lain (codec/noise/rerecord)
Dual-Channel Spoofed Speech Detection Based on Graph Attention Networks [19]	2025	Res2NetCA + DHGFM (graph attention, dual-channel)	ASVspoof 2019, ASVspoof 2021 LA	EER 1.64% pada dataset ASVspoof2019 dan EER 6.76% pada dataset ASVspoof2021 LA	Memanfaatkan *graph attention* dan desain *dual-channel* untuk menangkap relasi antarfitur/kanal, dan menunjukkan hasil kompetitif pada ASVspoof 2019 serta tetap berjalan pada ASVspoof 2021 LA (yang lebih sulit).	Pendekatan *dual*-channel tidak selalu cocok untuk skenario dunia nyata yang dominan mono channel; performa turun cukup besar pada ASVspoof 2021 LA, sehingga *robustness* terhadap kondisi yang lebih realistis masih menjadi tantangan.
AASIST: Audio Anti-Spoofing using Integrated Spectro-	2022	AASIST, AASIST-L	ASVspoof 2019 LA	EER 0,83% pada model AASIST dan EER 0,99% pada	Arsitektur *graph-attention* yang mengintegrasikan petunjuk spectro-temporal;	Tetap sensitif pada domain shift (*unseen attacks* / dataset berbeda) sehingga sering perlu strategi

Judul	Tahun	Model	Dataset	Hasil	Kelebihan	Kekurangan
Temporal Graph Attention Networks [10]				model AASIST-L *(lightweight)*	dikenal kuat sebagai baseline/SOTA yang stabil di ASVspoof 2019 LA, dan versi lightweight (AASIST-L) memberi opsi efisiensi.	tambahan (augmentasi, fine-tuning, atau fusion) untuk benar-benar *robust*.
CBAM-ResNet: A Lightweight ResNet Network Focusing on Time Domain Features for End-to-End Deepfake Speech Detection [20]	2025	1DResNet + CBAM	ASVspoof 2019 LA, ASVspoof 2015	EER 1.94% (ASVspoof2019 eval); cross-dataset: EER 4.94% (ASVspoof2015 eval); model sangat ringan (0.36M params)	Model sangat ringan (parameter kecil) dan fokus *time-domain* dengan modul CBAM sehingga cocok untuk skenario *resource* terbatas; menunjukkan hasil yang baik dan menyediakan gambaran *cross-dataset* (ASVspoof 2015).	Ada gap generalisasi yang terlihat dari hasil cross-dataset (EER naik); fokus *time-domain* dapat melewatkan artefak tertentu yang lebih jelas pada representasi waktu-frekuensi, jadi bisa kalah dari pendekatan spectro-temporal/SSL pada serangan tertentu.
Multi-scale Information Aggregation for Spoofing Detection [21]	2024	SincNet + DLA *(multi-scale aggregation)*	ASVspoof 2019 LA, ASVspoof 2021 LA/DF	EER 0.93% (ASVspoof 2019 LA), EER 13.99% (ASVspoof 2021 DF)	Desain multi-scale aggregation membantu menangkap pola spoof pada berbagai resolusi (lokal-global); menunjukkan kinerja baik di ASVspoof 2019 LA dan ada pembahasan *robustness* terhadap variasi.	Performa pada ASVspoof 2021 DF masih relatif berat (EER tinggi), yang menandakan tantangan besar ketika menghadapi *deepfake*/serangan yang lebih modern.
AFAD-MSA: Dataset and Models for Arabic Fake Audio Detection [25]	2026	*Baseline* Wav2Vec 2.0	AFAD-MSA + uji *adhoc* pada XTTS-v2, FishSpeech, SpeechT5, MMS	Pada test set AFAD-MSA: Wav2Vec 2.0 EER 0.38%. Namun pada *unseen/open-source* TTS tertentu, performa bisa turun drastis (EER 46.7-49.2%	Memberikan perspektif "realistic evaluation": detektor bisa sangat bagus in-domain (EER kecil), serta menguji *unseen/open-source* TTS dan menunjukkan masalah generalisasi secara jelas.	Fokus dataset dan bahasa pada Arab. Hasil EER dapat melonjak drastis pada generator yang berbeda *(out-of-domain)*, sehingga menegaskan keterbatasan detektor tunggal tanpa strategi *robustness* (domain augmentasi atau fusi).

Judul	Tahun	Model	Dataset	Hasil	Kelebihan	Kekurangan
				pada XTTS-v2/FishSpeech, dan 56.8% pada SpeechT5)		
iWAX: interpretable Wav2vec-AASIST-XGBoost framework for voice spoofing detection [23]	2025	Wav2Vec 2.0 + AASIST + XGBoost	ASVspoof 2019 LA	Model iWAX mengungguli AASIST dan w2v2-AASIST dalam EER; dengan ablation LightGBM berada di kisaran Eval EER ~0.275-0.303	Menggabungkan Wav2Vec 2.0 + AASIST dengan *classifier* (XGBoost) sehingga punya sisi interpretabilitas dan tetap kompetitif.	Kompleksitas *pipeline* meningkat (SSL + AASIST + model tabular); Hasil dan kestabilan sensitif terhadap pilihan fitur/ablation dan setup training, perlu pembuktian *robustness* pada dataset yang lebih beragam *(unseen/real-world)*.
Audio Deepfake Detection Using Deep Learning [13]	2025	Enhanced Siamese CNN + *self-attention* + StacLoss	ASVspoof 2019	EER 2.95%, Akurasi 98%, Precision 97%, Recall 96%, F1 96.5%, ROC-AUC 99%	Pendekatan Siamese + *self-attention* membantu mempelajari perbedaan pasangan/kemiripan.	EER masih lebih tinggi dibanding beberapa SOTA modern; evaluasi yang dominan pada ASVspoof 2019 membuat klaim generalisasi perlu hati-hati karena setup bisa menambah beban sampling/pairing saat pelatihan.
Deepfake Voice Detection: An Approach Using End-to-End Transformer with Acoustic Feature Fusion by Cross-Attention [26]	2025	Mel + F0 *cross-attention fusion* + CNN + Transformer	ASVspoof 2019 LA, *cross-test* ke ASVspoof 2021	EER 26.41% (ASVspoof 2019, setelah *silence processing*); EER 28.52% (ASVspoof 2021)	Memberi ide fusi yang jelas dan rapi secara metodologis: Mel + F0 digabung lewat *cross-attention,* lalu diproses CNN + *Transformer*.	Performa EER cukup tinggi dibanding SOTA; *robustness* masih lemah pada dataset yang lebih sulit.

Judul	Tahun	Model	Dataset	Hasil	Kelebihan	Kekurangan
Robust DeepFake Audio Detection via an Improved NeXt-TDNN with Multi-Fused Self-Supervised Learning Features [18]	2025	HuBERT + WavLM + AMFF + Improved NeXt-TDNN (ECA)	ASVspoof 2019 LA/PA, ASVspoof 2021 LA, ASVspoof 5	EER 0.42%, (ASVspoof 2019 LA); EER 1.01% (PA); pooled EER 6.56% (ASVspoof 2021 LA); EER 7.23% (ASVspoof 5)	Menggabungkan HuBERT + WavLM lalu *attentional fusion* (AMFF) + TDNN yang ditingkatkan; hasil sangat bagus di ASVspoof 2019 LA/PA dan tetap diuji pada ASVspoof 2021.	Ada degradasi pada *benchmark* yang lebih baru atau yang lebih sulit, sehingga menunjukkan bahwa meski hibrida SSL-*fusion* kuat, generalisasi terhadap serangan yang sangat berbeda masih harus ditingkatkan; biaya komputasi juga cenderung tinggi.
InaSAS: Benchmarking Indonesian Speech Antispoofing Systems [11]	2025	AASIST-L	InaSpoof-v1 (Indonesia)	Model AASIST-L paling efektif untuk *synthesized speech* dengan EER 4,19%	Memberi *benchmark* khusus Indonesia (InaSpoof-v1).	Karena sifatnya *benchmarking*, fokusnya bukan menawarkan satu arsitektur baru yang paling optimal.

Berdasarkan kajian tersebut, penelitian ini mengadaptasi kekuatan *attention* dari DPTFAN dan kekuatan representasi ucapan dari Wav2Vec 2.0 melalui fusi skor adaptif. Pendekatan ini dipilih untuk memadukan informasi artefak waktu-frekuensi dan representasi kontekstual waveform. Selain itu, pendekatan ini juga diarahkan untuk merespons keterbatasan penelitian terdahulu terkait generalisasi lintas domain dan keterwakilan bahasa non-Inggris.

2.2 Teori yang berkaitan

2.2.1 Kecerdasan Buatan

Kecerdasan Buatan atau *Artificial Intelligence* (AI) merujuk pada kemampuan sistem komputasi untuk meniru proses kognitif manusia seperti belajar, mengenali pola, mengambil keputusan, serta memahami informasi dalam bentuk teks, gambar, maupun suara [27].AI dalam ranah audio telah mendorong kemunculan sistem *text-to-speech* dan *voice conversion* berbasis jaringan saraf dalam yang mampu menghasilkan suara sintetis dengan tingkat kemiripan tinggi terhadap suara manusia asli [28], [29]. Performa tersebut menunjukkan bahwa suara yang dihasilkan oleh AI mampu mereplikasi karakteristik vokal secara presisi layaknya manusia. AI dapat dimanfaatkan sebagai konsumsi positif, seperti *audiobook*, asisten virtual, hingga produksi konten digital, namun di sisi lain, kemampuan meniru suara manusia membuka celah penyalahgunaan dalam bentuk kloning suara untuk tujuan manipulatif [30], [31]. Dalam kondisi tertentu, pendengar manusia tidak mampu membedakan secara akurat antara suara AI dan rekaman manusia asli, sehingga membedakan suara asli dan sintetis menjadi semakin sulit tanpa bantuan sistem otomatis [1]. Hal ini membuktikan bahwa perkembangan AI sangat menuntut inovasi pada sisi generatif, sisi deteksi dan keamanan.

Pendekatan AI mencakup *rule-based systems*, *machine learning*, dan *deep learning* [32]. Pendekatan berbasis pembelajaran data menjadi dominan karena mampu menangkap pola kompleks yang sulit dirumuskan secara eksplisit [33]. Pada domain suara, pola tersebut mencakup karakteristik spektral, dinamika temporal, hingga struktur fonetik [34], [35], [36]. Oleh sebab itu, penelitian mengenai deteksi kloning suara tidak dapat dilepaskan dari kerangka besar

kecerdasan buatan, karena seluruh proses pemodelan, pelatihan, hingga evaluasi sistem dibangun di atas paradigma pembelajaran berbasis data.

2.2.2 Audio *Deepfake*

Audio *deepfake* merujuk pada manipulasi atau pembuatan ulang sinyal suara manusia menggunakan model kecerdasan buatan sehingga menghasilkan ucapan yang terdengar alami seperti oleh penutur asli [37]. Pelaku memanfaatkan model generatif berbasis jaringan saraf dalam untuk membentuk ulang karakteristik vokal secara menyeluruh menjadi audio *deepfake* [38]. Model sintesis modern mampu meniru timbre, intonasi, dinamika energi, hingga pola artikulasi dengan tingkat kemiripan yang sangat tinggi [25]. Audio *deepfake* dapat diklasifikasikan menjadi tiga kategori utama, yakni [37]:

1. *Text-to-speech*

   Speech synthesis atau *text-to-speech* (TTS) adalah teknologi yang mampu menghasilkan ucapan manusia sintetis dari input teks secara natural [39]. Teknologi ini memanfaatkan arsitektur *deep learning* untuk memetakan karakter atau fonem menjadi gelombang suara yang memiliki intonasi dan ritme yang menyerupai suara aslinya.

2. *Voice conversion*

   Voice conversion atau konversi suara merupakan sebuah teknik yang mengubah suara penutur sumber menjadi suara penutur target tanpa mengubah isi linguistiknya [29]. Teknik ini memisahkan konten fonetik dari identitas vokal, kemudian merekonstruksi sinyal baru dengan karakter suara target. Sistem konversi suara modern mampu mempertahankan informasi linguistik dengan baik sambil mengubah karakteristik penutur secara nyata [40].

3. *Replay attack*

   Replay attack adalah pemutaran ulang rekaman suara asli melalui perangkat lain untuk mengecoh sistem verifikasi suara [41]. Meskipun tidak melibatkan sintesis digital, *replay attack* tetap termasuk dalam kategori spoofing karena menghasilkan sinyal tidak langsung yang mengandung distorsi kanal.

Keberadaan berbagai bentuk pemalsuan suara tersebut memperlihatkan bahwa perkembangan teknologi generatif telah mengubah suara menjadi media yang semakin mudah direkayasa dengan tingkat realisme yang tinggi. Situasi ini menimbulkan konsekuensi serius, terutama pada konteks yang menjadikan suara sebagai dasar autentikasi, identifikasi, maupun bukti komunikasi. Manipulasi audio berbasis AI telah menjadi tantangan nyata yang memerlukan sistem deteksi khusus yang mampu mengidentifikasi artefak akustik maupun inkonsistensi pola ucapan.

2.2.3 Kloning Suara

Kloning suara muncul dari dua teknologi yang saling melengkapi, yaitu *text-to-speech* (TTS) berbasis jaringan neural dan *voice conversion* atau konversi suara [42]. Kloning suara menjadi jauh lebih berbahaya ketika sistem mampu melakukan f*ew-shot* atau *zero-shot cloning*, karena target dapat ditiru hanya dari sampel yang sangat singkat [43]. Model generatif modern menunjukkan bahwa sistem *zero-shot multi-speaker text-to-speech* mampu meniru karakteristik suara penutur baru hanya dengan sampel berdurasi singkat, dan tetap mempertahankan kualitas alami dengan skor *Mean Opinion Score* (MOS) di atas 4 pada skala 5 poin [3], [44]. Angka-angka ini menunjukkan bahwa proses peniruan identitas vokal bukan lagi hanya membutuhkan waktu puluhan detik untuk menghasilkan suara yang dinilai sangat mirip oleh pendengar. Sisi lain yang membuat kloning suara mudah dipakai adalah strategi *speaker adaptation* pada TTS dengan pendekatan multimodal yang dapat berjalan dalam mode kloning teks ke suara maupun mode suara ke suara dengan cara mengatur *encoder* mana yang digunakan, lalu melakukan *fine-tuning* terutama pada bagian *decoder* [45].

Walau terdengar natural bagi manusia, suara hasil kloning biasanya meninggalkan jejak yang dapat dimanfaatkan oleh sistem deteksi. Jejak ini muncul karena proses generatif cenderung memproduksi artefak pada domain spektral dan temporal, terutama pada tahap *vocoder* yang mengubah representasi waktu-frekuensi menjadi gelombang audio [46]. Salah satu arah riset yang kuat menunjukkan bahwa artefak khas *vocoder* bisa dijadikan "sidik

jari" untuk membedakan suara sintetis dan suara asli. Namun performa suatu sistem dapat turun drastis saat domain berubah ketika jenis vocoder atau *post-processing* berbeda, di mana artefak generatif dapat dengan mudah tersamarkan oleh kompresi, codec, atau rekaman ulang. Suatu model yang menekankan eksploitasi artefak neural *vocoder* memiliki strategi multitask untuk klasifikasi *bonafide* dan *spoof* sekaligus identifikasi *vocoder*, mampu mencapai nilai *Equal Error Rate* (EER) yang sangat baik, namun nilai tersebut menurun ketika diuji pada dataset dengan kloning suara yang canggih [47].

2.2.4 Sistem Deteksi Audio Kloning Suara

Sistem deteksi audio kloning suara adalah sistem yang dirancang untuk menilai apakah suatu rekaman merupakan suara asli atau suara hasil sintesis dan manipulasi [4]. Tujuan utamanya adalah untuk membaca sifat sinyal audio itu sendiri untuk menemukan tanda bahwa suara tersebut dibentuk oleh proses generatif. Deteksi audio *deepfake* telah ditempatkan sebagai cabang tersendiri dalam keamanan audio, karena suara sintetis kini sudah cukup meyakinkan untuk mengaburkan batas antara ucapan alami dan ucapan hasil model [38]. Area ini berkembang dari kebutuhan untuk menghadapi audio hasil *text-to-speech* dan *voice conversion*, hingga kini bergerak pada penguatan sistem deteksi yang mampu mengenali manipulasi secara langsung dari sinyal suara.

Perkembangan sistem deteksi audio kloning suara pada awalnya banyak bertumpu pada pendekatan *feature engineering* klasik, yaitu mengekstraksi ciri akustik buatan tangan seperti *cepstral* dan fitur waktu-frekuensi, lalu mengklasifikasikannya dengan model statistik seperti *Gaussian Mixture Model* (GMM) [4]. Pola ini kuat pada fase awal *anti-spoofing* karena relatif ringan, cukup interpretatif, dan efektif untuk membedakan ucapan asli dan ucapan sintetis pada skenario yang masih terbatas. Seiring bertambahnya kompleksitas serangan, arah penelitian kemudian bergeser ke *deep learning*, terutama model berbasis CNN yang memproses representasi spektral untuk menangkap artefak lokal pada domain waktu-frekuensi. Setelah itu, penelitian mutakhir mulai banyak memanfaatkan mekanisme *attention* dan *Self-supervised Learning* (SSL) seperti Wav2Vec 2.0, karena representasi pralatih dari model-model

tersebut dinilai lebih mampu menangkap jejak manipulasi yang halus dan lebih adaptif terhadap serangan yang terus berubah dengan kontribusi kecil terhadap klasifikasi [4], [13].

2.2.5 Karakteristik Fonetik dan Akustik Bahasa Indonesia

Karakteristik fonetik dan akustik Bahasa Indonesia merupakan landasan penting dalam penelitian deteksi kloning suara karena sistem deteksi membaca pola bunyi pada sinyal suara [4]. Pada level fonetik, pengamatan berangkat dari satuan bunyi (vokal-konsonan) dan bagaimana bunyi tersebut terealisasi melalui frekuensi, intensitas, durasi, transisi antarsuku kata, serta pola prosodi. Sistem prosodi Bahasa Indonesia modern kerap diposisikan sebagai bahasa yang tidak memiliki tekanan leksikal yang teratur, sehingga penonjolan bunyi lebih banyak muncul sebagai realisasi akustik pada tingkat ujaran daripada pola tekanan kata yang kaku [48]. Kondisi ini membuat pendekatan yang terbiasa pada bahasa dengan stres leksikal yang kuat tidak selalu langsung selaras ketika diterapkan pada ujaran Bahasa Indonesia. Variasi regional juga memperlebar rentang realisasi prosodi, karena kecenderungan penonjolan suku kata dapat berbeda antarragam, termasuk pergeseran pola penonjolan yang berkaitan dengan latar kebahasaan penutur [49].

Aspek utama unsur suprasegmental pada level akustik adalah *pitch*, intensitas, dan durasi. Ketiganya membentuk ritme bicara sekaligus menjadi area yang kerap terlihat terlalu seragam pada audio hasil kloning. Pengukuran prosodi pada penutur yang dipengaruhi bahasa daerah menunjukkan bahwa frekuensi, intensitas, dan durasi dapat berubah menurut karakter penutur, sehingga pembacaan sinyal yang hanya bertumpu pada satu ciri berisiko tidak stabil. Selain itu, identitas akustik bahasa juga dipengaruhi oleh sistem vokal melalui distribusi forman (F1-F3), dan variasi nilai forman antardaerah memperlihatkan bahwa ruang akustik vokal di Indonesia sangat beragam, sehingga peniruan suara yang tampak meyakinkan dapat tetap menyisakan kekakuan pada perpindahan formant ketika berpindah penutur atau ragam. Pola penekanan suku kata pada beberapa bahasa daerah menunjukkan variasi penekanan yang dapat muncul pada posisi tertentu dengan nilai akustik yang

terukur, sehingga sistem deteksi untuk Bahasa Indonesia perlu dibangun dengan asumsi bahwa variasi penutur asli memang lebar dan tidak selalu mengikuti satu pola standar yang tunggal.

2.2.6 *Data Labeling*

Data labeling merupakan proses pemberian anotasi atau label pada data mentah, agar dapat digunakan dalam pelatihan model *machine learning* [50]. Label menjadi acuan utama yang menghubungkan data masukan dengan kelas keluaran agar model dapat mempelajari pola yang sesuai selama proses pelatihan [51]. Pada domain *machine learning*, *supervised learning* bergantung pada data yang telah diberi anotasi, karena proses pembelajarannya dibangun dari pasangan data dan label yang tersedia [52]. Dalam tugas klasifikasi biner, *data labeling* menempatkan setiap sampel ke dalam salah satu dari dua kelas yang sudah ditetapkan [53]. Pada penelitian deteksi kloning suara, pelabelan diarahkan pada dua kategori utama, yaitu real untuk audio asli dan palsu untuk audio hasil sintesis atau manipulasi [54]. Pemisahan kategori ini dibutuhkan karena model hanya dapat membentuk batas pemisah yang benar apabila definisi kelas disusun secara konsisten sejak awal. Jika audio sintetis atau audio asli diberi label yang tidak sesuai, maka pola yang dipelajari model dapat bergeser dan memengaruhi kualitas generalisasi pada tahap pengujian.

2.2.7 *Data Splitting*

Data splitting adalah proses membagi dataset ke dalam beberapa bagian agar pelatihan, validasi, dan pengujian model dapat dilakukan secara terpisah [55]. Tujuan utamanya adalah untuk mengurangi kebocoran data saat model dilatih dan diuji, sehingga hasil evaluasi lebih mencerminkan kemampuan generalisasi [56]. Pemilihan strategi pembagian data merupakan bagian penting dari validasi model, karena kualitas evaluasi sangat dipengaruhi oleh bagaimana data latih dan data uji dipisahkan. Dalam praktik pembelajaran mesin, pembagian yang paling umum dilakukan adalah ke dalam data latih *(train)*, data validasi *(validation)*, dan data uji *(test)* [57]. Data latih digunakan untuk memperbarui parameter model selama proses pelatihan. Data validasi dipakai untuk memantau performa selama pelatihan, menilai kecenderungan

overfitting, serta membantu pemilihan konfigurasi atau model terbaik. Data uji digunakan setelah pelatihan selesai untuk memberikan penilaian akhir yang lebih objektif.

Dua rasio yang sering digunakan untuk pembagian data latih, validasi, dan uji adalah 70:20:10 dan 80:10:10. Kedua rasio tersebut digunakan bersama rasio lain sebagai bagian dari skenario eksperimen, dan hasil yang dilaporkan menunjukkan bahwa konfigurasi 80:10:10 memberikan performa terbaik [58]. Peningkatan porsi data latih dapat membantu model membangun pola pembelajaran dengan lebih baik, khususnya pada tugas klasifikasi berbasis *deep learning* [59]. Namun, ukuran data validasi tetap penting karena berpengaruh terhadap kualitas estimasi performa model selama pelatihan [60]. Oleh karena itu, tidak ada satu rasio pembagian yang selalu paling optimal untuk semua kondisi. Pada dataset yang lebih besar, rasio 80:10:10 dapat lebih menguntungkan karena memberikan porsi data latih yang lebih besar tanpa menghilangkan fungsi validasi dan pengujian. Sebaliknya, pada dataset yang lebih terbatas atau ketika peneliti memerlukan pemantauan performa yang lebih hati-hati, rasio 70:20:10 dapat dipertimbangkan karena menyediakan porsi validasi yang lebih luas. Perbandingan kedua rasio terdapat pada Tabel 2.2.

Tabel 2.2 Tabel Perbandingan Rasio *Data Splitting*

Aspek	**Rasio**	
	70:20:10	**80:10:10**
Fokus	Memperkuat pemantauan performa	Memaksimalkan pelatihan model
Kelebihan	Evaluasi validasi lebih kuat	Data latih lebih banyak, peluang belajar pola lebih besar
Kekurangan	Data latih lebih sedikit	Validasi lebih terbatas
Kondisi Pemakaian	Dataset lebih terbatas, *tuning* lebih hati-hati	Dataset cukup besar, model *deep learning*

Rasio 80:10:10 memberi porsi data latih yang lebih besar, sehingga lebih menguntungkan ketika model memerlukan lebih banyak contoh untuk mempelajari pola, terutama pada tugas *deep learning* dengan jumlah data yang cukup memadai [61]. Sebaliknya, rasio 70:20:10 memberi ruang validasi yang lebih luas, sehingga pemantauan performa selama pelatihan dapat dilakukan dengan basis data yang lebih besar [62]. Semakin besar porsi data latih, semakin

besar peluang model membangun pola yang lebih baik, tetapi porsi pengujian dan validasi tetap harus cukup representatif agar evaluasi tidak menjadi bias. Maka dari itu, tidak ada satu rasio yang selalu paling baik untuk semua kondisi. Pada dataset yang jumlahnya lebih besar, 80:10:10 sering lebih menguntungkan karena porsi latih bertambah tanpa menghilangkan validasi dan pengujian. Pada dataset yang lebih terbatas atau membutuhkan pemantauan performa yang lebih hati-hati selama pelatihan, rasio 70:20:10 menjadi pilihan yang lebih konservatif.

2.2.8 *Resampling*

Sample rate atau laju sampel adalah jumlah sampel digital yang diambil dari sinyal kontinu dalam setiap detik, dan umumnya dinyatakan dalam satuan Hertz (Hz) [63]. Dalam pemrosesan audio digital, laju sampel menentukan seberapa rapat sinyal suara direpresentasikan dalam bentuk diskret. Perbedaan laju sampel dalam data audio akan memengaruhi kualitas representasi sinyal sekaligus kebutuhan komputasi [64]. Laju sampel yang lebih tinggi dapat menyimpan komponen frekuensi yang lebih luas, namun meningkatkan ukuran data dan beban pemrosesan [78]. Sebaliknya, laju sampel yang lebih rendah membuat data lebih ringan, tetapi rentang frekuensi yang dapat dipertahankan terbatas [65]. Penelitian berbasis audio seringkali menetapkan satu laju sampel yang seragam agar seluruh data berada pada standar representasi yang sama sebelum masuk ke tahap pemodelan [66]. Proses perubahan dari satu laju sampel ke laju sampel lain dijelaskan sebagai *resampling*, yaitu konversi frekuensi sampling waveform ke frekuensi baru menggunakan band-limited interpolation [67].

Resampling adalah proses mengubah laju sampel suatu sinyal audio ke laju sampel lain tanpa mengubah isi dasar sinyal secara sengaja [35]. Tujuan utamanya adalah menyamakan format audio agar seluruh data dapat diproses secara konsisten dalam satu alur [68]. Proses ini diperlukan ketika dataset berasal dari sumber yang tidak seragam, karena model *deep learning* membutuhkan masukan yang konsisten [69]. Perbedaan laju sampel pada data audio akan memengaruhi jumlah sampel per detik, panjang sinyal, serta

karakter representasi yang dihasilkan pada tahap ekstraksi fitur [70]. Proses *resampling* ke satu nilai laju sampel yang sama akan membuat data dapat dinormalisasi ke ruang representasi yang lebih seragam [68]. Pada penelitian deteksi kloning suara, perbedaan laju sampel berpotensi menimbulkan variasi fitur yang tidak sepenuhnya berasal dari karakteristik suara, tetapi juga dari perbedaan format dan kondisi perekaman [71]. Oleh sebab itu, *resampling* dapat dipahami sebagai tahap standarisasi sinyal yang menjaga konsistensi input sebelum proses ekstraksi fitur atau pelatihan model dilakukan.

2.2.9 Ekstraksi Fitur Audio

Ekstraksi fitur audio merupakan tahap yang mengubah gelombang suara mentah menjadi representasi yang lebih terstruktur agar pola pada sinyal akustik dapat dibaca model secara lebih jelas [72]. Suara sintetis modern sering hanya meninggalkan jejak manipulasi yang sangat halus, sehingga representasi yang kurang tepat dapat membuat perbedaan antara suara asli dan suara palsu menjadi semakin sulit dikenali [4]. Fitur untuk deteksi audio *deepfake* terbagi ke dalam dua kelompok besar, yaitu *handcrafted features* dan *learning-based features [13]*. *Handcrafted features* dibangun dari prinsip dasar pemrosesan sinyal dan dirancang untuk menonjolkan informasi tertentu dari domain waktu, frekuensi, atau cepstral [73]. Pada sinyal ucapan yang bersifat nonstasioner, salah satu representasi yang dibutuhkan adalah waktu-frekuensi, karena karakteristik suara berubah terus sepanjang waktu dan tidak dapat dijelaskan hanya dengan satu spektrum yang tetap, sehingga banyak pendekatan deteksi yang masih bertumpu pada fitur berbasis spektral yang mampu memperlihatkan distribusi energi pada frekuensi tertentu dari waktu ke waktu [74], [75], [76]. Representasi ini dianggap penting karena proses sintesis dan manipulasi suara sering meninggalkan jejak pada tekstur spektral, keteraturan harmonik, maupun transisi energi yang tidak sepenuhnya sama dengan ucapan alami, meskipun secara perseptual suara tersebut terdengar meyakinkan [4].

Bentuk representasi spektral yang paling sering dibahas dalam literatur meliputi *Short Time Fourier Transform* (STFT), mel-*spectrogram*, dan log-mel *spectrogram*, karena ketiganya membentuk alur representasi yang saling

berkaitan [77]. STFT digunakan untuk memecah sinyal suara ke dalam potongan waktu pendek, lalu menghitung kandungan frekuensi pada setiap potongan tersebut. Hasil ini kemudian dapat dipetakan menjadi mel-*spectrogram*, yaitu representasi yang menyusun energi spektrum ke dalam skala mel agar lebih selaras dengan sensitivitas pendengaran manusia terhadap frekuensi suara [78]. Setelah itu, transformasi logaritmik menghasilkan log-mel *spectrogram*, yang membuat rentang energi menjadi lebih stabil sekaligus membantu menonjol [79]. Bentuk dua dimensi dari representasi ini juga mempertahankan hubungan antara waktu dan frekuensi, sehingga pola seperti harmonik, forman, dan perubahan energi antarbagian ujaran tetap dapat diamati dengan jelas [76]. Log-mel *spectrogram* banyak dipertahankan dalam penelitian deteksi audio sintetis, terutama pada arsitektur konvolusional, karena struktur spasialnya memungkinkan model mempelajari perbedaan halus antara suara alami dan suara hasil manipulasi [80].

Fitur spektral klasik belum kehilangan relevansinya dalam literatur deteksi audio kloning suara walaupun pendekatan modern telah berkembang ke arah representasi yang dipelajari langsung oleh model [13]. Representasi log-mel *spectrogram* tetap dianggap kuat karena mampu menyajikan informasi akustik secara kaya, stabil, dan tetap cukup mudah diinterpretasikan. Efektivitas deteksi model sangat dipengaruhi oleh kesesuaian antara bentuk representasi masukan dan karakteristik artefak yang ingin diungkap [54].

2.2.10 Pooling

Pooling adalah operasi agregasi yang digunakan untuk merangkum sekumpulan representasi lokal menjadi representasi yang lebih ringkas [91]. Keluaran model pada tingkat frame dalam jaringan yang memproses sinyal masih berbentuk deret fitur yang panjang, sehingga perlu adanya mekanisme yang mengubah representasi *frame-level* menjadi satu vektor agar tetap dapat digunakan untuk klasifikasi pada tingkat ujaran [92]. *Pooling* yang sering

digunakan dalam model representasi ucapan adalah *statistical pooling*, yaitu agregasi yang merangkum distribusi fitur melalui rata-rata *(mean)* dan simpangan baku [93]. Bentuk ini banyak digunakan pada tugas representasi ucapan karena menangkap kecenderungan pusat dari fitur dan menyimpan informasi variasi antar-*frame* [94]. *Statistics pooling layer* yang umum dipakai adalah konkatenasi *mean* dan standar deviasi dari aktivasi *frame-level* yang membentuk representasi ujaran pada tingkat segmen [81]. *Mean* gambaran kecenderungan umum dari aktivasi fitur, sedangkan simpangan baku menyimpan informasi tentang variasi atau penyebaran aktivasi antar*frame*. Kombinasi keduanya membuat representasi akhir dapat menangkap kestabilan atau perubahan respons tersebut sepanjang sinyal. Secara matematis, keluaran *frame-level* dinyatakan sebagai X pada Persamaan 2.1 dengan T sebagai jumlah *frame* dan d sebagai dimensi fitur. Perhitungan *mean pooling* dan *pooling* simpangan baku didapatkan menggunakan Persamaan 2.2 dan Persamaan 2.3, yang kemudian keduanya digabungkan dengan Persamaan 2.4.

$$X = \{x_1, x_2, \dots, x_T\}, x_t \in \mathbb{R}^d \tag{2.1}$$

$$\mu = \frac{1}{T}\sum_{t=1}^{T} x_t \tag{2.2}$$

$$\sigma = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(x_t - \mu)^2} \tag{2.3}$$

$$z = [\mu \parallel \sigma] \tag{2.4}$$

Rumus 2.1 Perhitungan *Statistical Pooling*
Sumber: [82]

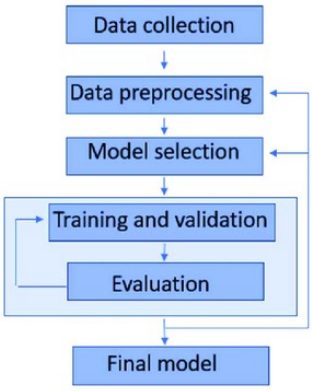
2.2.11 Augmentasi Data

Augmentasi data adalah proses membentuk variasi baru dari data latih dengan tetap mempertahankan identitas kelas utamanya, sehingga model memperoleh paparan terhadap kondisi masukan yang lebih beragam selama pelatihan [83]. Tujuan augmentasi data pada *machine learning* berbasis audio adalah menambah keragaman sinyal, mengurangi kecenderungan *overfitting*, dan membantu model menjadi lebih tahan terhadap perubahan kondisi rekaman

[84]. Proses augmentasi audio dapat dilakukan pada domain gelombang mentah *(raw waveform)* maupun pada domain representasi fitur [85]. Pendekatan berbasis domain waktu dapat mencakup penambahan *noise*, kompresi, *reverb*, dan *time stretching*, dan sebagainya. [86]. Augmentasi tidak terbatas pada satu teknik saja, tetapi merupakan kelompok strategi yang dipilih sesuai karakter data dan tujuan model.

2.2.12 *Machine Learning*

Machine learning atau pembelajaran mesin merupakan cabang kecerdasan buatan yang berfokus pada kemampuan sistem untuk mempelajari pola dari data dan menggunakan pola tersebut untuk membuat prediksi atau keputusan tanpa harus diprogram dengan aturan yang sepenuhnya eksplisit [87]. Inti dari pendekatan ini terletak pada proses pembelajaran dari contoh, sehingga sistem tidak lagi bergantung pada logika yang kaku, melainkan membangun fungsi pemetaan dari hubungan antara data masukan dan keluaran yang diamati [88]. *Machine learning* diposisikan sebagai fondasi pemrosesan berbasis data untuk tugas seperti klasifikasi, regresi, dan pengelompokan, dengan tujuan utama menemukan pola yang dapat digeneralisasi ke data baru [89]. *Machine learning* terdiri atas beberapa paradigma pembelajaran yang mencakup *supervised learning*, *unsupervised learning*, dan *reinforcement learning* [90]. Dari ketiga paradigma tersebut, pendekatan yang paling relevan untuk penelitian deteksi kloning suara adalah *supervised learning*, karena model dilatih untuk membedakan dua kelas yang sudah didefinisikan, yaitu audio asli dan audio hasil kloning.



Gambar 2.1 Alur Kerja *Machine Learning*
Sumber: [91]

Alur kerja *machine learning* dimulai dari pengumpulan data, dilanjutkan dengan praproses data, pemilihan model, pelatihan dan validasi, kemudian evaluasi hingga menghasilkan model akhir [91]. Pada Gambar 2.1, rangkaian ini ditunjukkan sebagai proses yang tidak sepenuhnya linear, karena tahap evaluasi dapat mengarahkan penyesuaian kembali pada proses pelatihan, pemilihan model, bahkan praproses data. Alur ini menunjukkan bahwa pengembangan model *machine learning* pada dasarnya bersifat iteratif, di mana hasil evaluasi digunakan untuk memperbaiki representasi data maupun konfigurasi model sampai diperoleh performa yang paling sesuai. Pada domain audio, alur tersebut menjadi lebih menantang karena data yang diolah tidak berbentuk tabel statis, melainkan sinyal yang berubah terus sepanjang waktu dan memuat hubungan kompleks antarfrekuensi, energi, serta struktur temporal [92], [93]. Kondisi ini membuat tahap *preprocessing* dan representasi data menjadi sangat penting, karena kualitas informasi yang masuk ke model akan sangat menentukan kualitas pola yang dapat dipelajari selama pelatihan [94]. Setelah itu, model yang dipilih akan membentuk batas keputusan berdasarkan pola pada data latih, lalu kinerjanya diperiksa kembali melalui validasi dan evaluasi untuk melihat sejauh mana model mampu melakukan generalisasi pada data yang tidak digunakan saat pelatihan [95]. Dalam tugas klasifikasi, hasil akhirnya adalah keputusan apakah suatu sampel baru lebih dekat ke satu kelas atau kelas lainnya, berdasarkan pola yang telah dipelajari dari keseluruhan alur tersebut [96].

Meskipun banyak penelitian mutakhir bergerak ke arah *deep learning*, *machine learning* tetap banyak digunakan karena menjadi dasar logika pembelajaran yang digunakan oleh seluruh model modern, termasuk jaringan saraf dalam [89]. *Deep learning* pada dasarnya berkembang dari prinsip yang sama, tetapi menggunakan arsitektur berlapis untuk mempelajari representasi yang jauh lebih kompleks dan lebih kaya [97]. Oleh karena itu, pemahaman mengenai *machine learning* tetap diperlukan agar hubungan antara data, fitur, label, proses pelatihan, dan kemampuan generalisasi sistem dapat dijelaskan

secara runtut sebelum masuk ke pembahasan *deep learning*. Dalam deteksi kloning suara, *machine learning* menjadi kerangka dasar yang menghubungkan sinyal audio yang telah direpresentasikan ke dalam fitur tertentu dengan keputusan klasifikasi akhir, sehingga sistem dapat membedakan suara asli dan suara sintetis secara sistematis.

2.2.13 *Deep Learning*

Deep learning merupakan cabang dari *machine learning* yang menggunakan jaringan saraf berlapis banyak untuk mempelajari representasi data secara bertingkat, mulai dari pola yang sederhana hingga pola yang lebih abstrak [98]. *Deep learning* sangat sesuai untuk data yang kompleks dan tidak mudah dirumuskan secara manual seperti gambar, teks, dan terutama suara karena mampu mempelajari sebagian besar representasi langsung dari data selama proses pelatihan [99]. Penggunaan *deep learning* pada domain suara atau audio berkembang pesat karena sinyal suara memiliki karakter yang dinamis, berubah terus sepanjang waktu, dan memuat hubungan yang saling terkait antara frekuensi, energi, ritme, fase, serta struktur fonetik. Hal ini mengakibatkan terjadinya kebutuhan pendekatan yang lebih kompleks untuk membaca keseluruhan karakter suara secara utuh, sehingga berbagai arsitektur *deep learning* digunakan sesuai sifat informasi yang ingin ditangkap. Arsitektur jenis CNN banyak dimanfaatkan untuk membaca pola lokal pada representasi waktu-frekuensi, seperti tekstur spektral, harmonik, perubahan energi, dan artefak kecil pada area frekuensi tertentu [100]. RNN dan turunannya banyak digunakan untuk memodelkan ketergantungan temporal, sedangkan pendekatan berbasis attention dan transformer semakin menonjol karena mampu menangkap hubungan yang lebih panjang dan lebih luas pada sinyal ucapan [101].

Deep learning banyak digunakan dalam tugas deteksi kloning suara karena unggul dalam membedakan suara asli dan suara sintetis berdasarkan pola akustik dan fonetik yang sering sangat halus [5]. Tantangan utama dalam mendeteksi kloning suara adalah mencapai performa tinggi pada data yang serupa dengan data pelatihan dan memastikan bahwa representasi yang

dipelajari tetap berguna ketika model menghadapi generator baru, kualitas rekaman yang berbeda, atau kondisi dunia nyata yang lebih bervariasi [4]. D*eep learning* dipandang sebagai pendekatan yang penting dalam deteksi audio kloning suara karena mampu menggabungkan pembacaan artefak akustik lokal dengan pemahaman pola ucapan yang lebih tinggi dalam satu kerangka pembelajaran.

2.2.14 Mekanisme *Attention*

Mekanisme *Attention* merupakan mekanisme dalam *deep learning* untuk memberi bobot berbeda pada bagian input sesuai tingkat kepentingannya saat membentuk representasi [102]. Mekanisme ini digunakan untuk memusatkan fokus pada informasi yang paling relevan bagi tugas prediksi pada model [103]. Setiap bagian input tidak selalu membawa kontribusi yang sama, sehingga model perlu belajar menonjolkan komponen yang paling informatif dan menurunkan pengaruh komponen yang kurang penting. Dalam perkembangan *deep learning* modern, *attention* dipandang sebagai konsep penting karena mampu meningkatkan kualitas representasi melalui penekanan selektif pada fitur yang paling berkontribusi terhadap hasil prediksi [102]. Attention berperan sebagai mekanisme penyaringan terhadap hasil ekstraksi fitur, sehingga informasi yang diteruskan ke tahap berikutnya menjadi lebih terarah dan lebih bernilai bagi proses pengambilan keputusan [13].

Attention memiliki beberapa bentuk sesuai kebutuhan representasi data. Sebagian *attention* bekerja pada hubungan antarfitur lokal, sebagian memodelkan keterkaitan antarlangkah waktu, dan sebagian lain menghitung hubungan antarbagian input secara menyeluruh melalui *self-attention [104]*. Pada pemrosesan ujaran, mekanisme ini penting karena informasi pada sinyal suara tidak tersebar merata sepanjang durasi audio. Beberapa bagian memuat kandungan fonetik dan akustik yang kuat, sementara bagian lain lebih banyak berisi keheningan, transisi lemah, atau unsur dengan kontribusi kecil terhadap klasifikasi [4]. Kondisi tersebut membuat *attention* membantu jaringan memilih bagian sinyal dengan nilai informasi tertinggi, sehingga pembacaan model menjadi lebih selektif, lebih terfokus, dan lebih sesuai dengan karakter ucapan

yang diproses. *Attention* membantu model menangkap ketidakwajaran halus pada berbagai bagian sinyal, baik dalam bentuk artefak akustik, perubahan spektral yang janggal, maupun pola ucapan yang terlalu seragam [13].

2.2.15 *Pythagorean Hesitant Fuzzy Sets*

Pythagorean Hesitant Fuzzy Sets (PHFS) merupakan pengembangan dari teori himpunan *fuzzy* yang digunakan untuk merepresentasikan kondisi ketika suatu elemen tidak dapat dinilai hanya dengan satu derajat keanggotaan yang pasti [105]. Pada situasi tertentu, penilaian dapat bersifat ragu, sehingga terdapat lebih dari satu kemungkinan nilai keanggotaan maupun non-keanggotaan yang dianggap masuk akal [106]. PHFS dibangun dari penggabungan dua gagasan, yaitu *hesitant fuzzy set* yang mengakomodasi beberapa nilai kemungkinan secara bersamaan, dan *Pythagorean fuzzy set* yang memberi ruang representasi lebih luas melalui pembatasan jumlah kuadrat derajat keanggotaan dan non-keanggotaan [107]. Dalam lingkungan PHFS, keraguan dinyatakan sebagai sekumpulan kemungkinan nilai yang tetap berada dalam batas valid secara matematis.

PHFS untuk suatu elemen x pada semesta penuturan X dapat dinyatakan sebagai P pada Persamaan 2.5, dengan $H_P(x)$ yang didefinisikan pada Persamaan 2.6 sebagai himpunan pasangan nilai kemungkinan derajat keanggotaan dan non-keanggotaan [123]. Pada himpunan tersebut, μ_i merepresentasikan kemungkinan derajat keanggotaan, sedangkan ν_i merepresentasikan kemungkinan derajat non-keanggotaan, dengan batas bahwa keduanya berada pada rentang 0 sampai 1 dan tetap memenuhi kendala *Pythagorean* [122]. Berdasarkan pasangan nilai tersebut, derajat keraguan atau *hesitation degree* kemudian dapat dihitung menggunakan Persamaan 2.7 melalui π_i, yang menyatakan bagian ketidakpastian yang belum teralokasi ke keanggotaan maupun non-keanggotaan [108]. Semakin besar nilai π_i, semakin besar pula tingkat ambiguitas atau ketidakpastian yang melekat pada penilaian terhadap elemen tersebut.

$$P = \{\langle x, H_P(x)\rangle \mid x \in X\} \tag{2.5}$$

$$H_P(x) = \{(\mu_i, \nu_i) \mid 0 \leq \mu_i, \nu_i \leq 1,\ \mu_i^2 + \nu_i^2 \leq 1\} \tag{2.6}$$

$$\pi_i = \sqrt{1 - \mu_i^2 - \nu_i^2} \tag{2.7}$$

Rumus 2.2 Perhitungan *Pythagorean Hesitant Fuzzy Sets*

PHFS digunakan dalam deteksi kloning suara karena keluaran model deteksi tidak selalu berada pada kondisi yang sepenuhnya tegas [14]. Skor model dapat memuat ambiguitas pada beberapa sampel audio sehingga keputusan klasifikasi memerlukan representasi ketidakpastian yang lebih fleksibel [109]. Ruang ketidakpastian tersebut dapat dimodelkan dalam bentuk pasangan kemungkinan keanggotaan dan non-keanggotaan yang tetap valid secara matematis, sehingga proses pengambilan keputusan tidak hanya bergantung pada satu skor pasti [110].

2.2.16 *Convolutional Attention Network*

Convolutional Attention Network (CAN) merupakan keluarga arsitektur CNN yang diperkaya dengan modul *attention* untuk melakukan pembobotan ulang terhadap fitur yang telah diekstraksi [111]. *Attention* pada CAN diposisikan sebagai cara untuk menilai bagian mana dari peta fitur yang perlu ditekankan sebelum diteruskan ke lapisan berikutnya tanpa menggunakan mekanisme relasi global berbasis *self-attention* seperti pada transformer [112]. Bentuk ini terlihat pada modul *attention* untuk CNN, yakni *Squeeze-and-Excitation* (SE) yang melakukan *channel-wise recalibration* dan CBAM yang menambahkan perhatian kanal serta spasial untuk memperhalus representasi fitur melalui pembobotan ulang adaptif [113].

CAN digunakan dalam domain audio karena representasi waktu-frekuensi seperti spektrogram atau log-mel dapat diperlakukan sebagai peta fitur 2D, sehingga CNN mengekstraksi pola lokal sementara *attention* melakukan re-weighting pada bagian yang paling informatif [114]. Pada mekanisme *attention* waktu-frekuensi, bobot temporal dan frekuensi dapat dibentuk secara terpisah, dinormalisasi dengan Softmax, lalu dikalikan kembali dengan representasi spektrogram untuk menghasilkan fitur yang lebih terarah [14]. Mekanisme ini

pada praktiknya sering diwujudkan sebagai *soft attention* berbasis normalisasi di sepanjang sumbu tertentu, sehingga bobot pada waktu atau frekuensi menjadi terdistribusi dan lebih mudah ditafsir sebagai tingkat penekanan relatif.

2.2.17 *Self-Supervised Learning* pada Audio

Self-supervised learning pada audio merupakan pendekatan pembelajaran yang memanfaatkan data suara tanpa label untuk membentuk representasi yang berguna sebelum model digunakan pada tugas akhir tertentu [15]. Pendekatan ini mampu mempelajari struktur dasar sinyal ucapan langsung dari data, tanpa bergantung pada anotasi manual sejak awal [115]. Model diarahkan untuk mengenali pola fonetik, hubungan antarsegmen, dan susunan internal sinyal suara melalui tugas bantu yang dibentuk dari data itu sendiri. Pada domain audio, data suara jauh lebih mudah dikumpulkan dibandingkan data berlabel, sementara proses pelabelan sering memerlukan waktu, biaya, dan keahlian tambahan. *Self-supervised learning* dipahami sebagai cara untuk membekali model dengan pengetahuan awal mengenai karakteristik umum ucapan, sehingga pembelajaran pada tugas spesifik dapat dilakukan di atas representasi yang sudah kaya informasi [116].

Self-supervised learning bekerja melalui dua tahap utama, yaitu *pre-training* dan *fine-tuning* [15]. Pada tahap *pre-training*, model mempelajari pola umum dari sejumlah besar audio tanpa label dengan cara menebak bagian sinyal yang disembunyikan, membedakan representasi yang benar dari yang salah, atau membangun kembali hubungan antarbagian audio [117]. Pada tahap *fine-tuning,* representasi yang telah dipelajari digunakan sebagai dasar untuk menyelesaikan tugas spesifik seperti klasifikasi audio, pengenalan ujaran, atau deteksi audio sintetis [18]. Sinyal ucapan yang diolah model *self-supervised learning* menyimpan informasi pada banyak lapisan representasi sekaligus. Representasi hasil *self-supervised learning* bernilai tinggi karena mampu menangkap hubungan antarkomponen suara secara lebih mendalam, sehingga pembacaan sinyal tidak berhenti pada ciri permukaan [116].

2.2.18 Fungsi *Loss*

Fungsi *loss* adalah fungsi objektif yang digunakan untuk mengukur selisih antara keluaran prediksi model dan nilai target yang seharusnya [118]. Nilai *loss* menjadi acuan utama untuk menilai apakah model sudah bergerak ke arah yang benar atau masih menghasilkan kesalahan yang besar [119]. Semakin kecil nilai *loss*, semakin dekat hasil prediksi model terhadap target, sehingga fungsi ini berperan sebagai indikator kuantitatif dari kualitas pembelajaran model pada setiap iterasi. Fungsi *loss* juga menentukan arah pembaruan parameter karena gradien dihitung dari fungsi ini, sehingga karakter fungsi loss akan memengaruhi bagaimana model belajar [120]. Pemilihan fungsi *loss* selalu disesuaikan dengan jenis tugas yang dikerjakan. Pada tugas klasifikasi, fungsi *loss* dirancang untuk menilai seberapa jauh probabilitas atau skor prediksi model menyimpang dari label kelas yang benar [121]. Untuk klasifikasi biner, bentuk yang umum digunakan berasal dari binary *cross entropy*, yaitu fungsi yang menghukum prediksi yang semakin jauh dari label target [118]. Fungsi *loss* dapat dipahami sebagai jembatan antara hasil prediksi dan proses optimasi karena kualitas pembelajaran model sangat bergantung pada bagaimana kesalahan didefinisikan dan dihitung [122].

2.2.19 *Optimizer*

Optimizer adalah mekanisme yang digunakan untuk memperbarui parameter model berdasarkan gradien dari fungsi *loss* [123]. Setelah nilai kesalahan dihitung, model perlu menentukan seberapa besar dan ke arah mana bobot harus diubah agar kesalahan tersebut berkurang. *Optimizer* merupakan bagian penting yang memengaruhi kecepatan konvergensi, kestabilan pelatihan, dan kemampuan model mencapai solusi yang baik [124].

Proses optimasi dilakukan dengan memanfaatkan gradien dari fungsi *loss* terhadap parameter model [125]. Bentuk paling dasar dari pembaruan parameter dituliskan pada Persamaan 2.8 dengan θ sebagai parameter model, η sebagai learning rate, dan $\nabla\theta L$ sebagai gradien fungsi *loss*. Persamaan tersebut menunjukkan bahwa parameter diperbarui ke arah yang berlawanan dengan gradien agar nilai loss menurun. Dari prinsip dasar ini, kemudian berkembang berbagai jenis *optimizer* yang menambahkan mekanisme tambahan seperti

momentum, adaptasi laju belajar, dan regularisasi untuk membuat proses pelatihan lebih efisien dan lebih stabil. *Optimizer* dapat dipahami sebagai aturan pembelajaran yang menghubungkan informasi gradien dengan perubahan parameter model [126]. Karena tiap *optimizer* memiliki karakter yang berbeda, pemilihannya perlu disesuaikan dengan sifat model dan data yang digunakan.

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L} \tag{2.8}$$

Rumus 2.3 Perhitungan Dasar *Optimizer* [127]

2.2.20 Metode Fusi dalam Deteksi Kloning Suara

Metode fusi merupakan pendekatan penggabungan informasi dari dua atau lebih sumber representasi agar sistem deteksi memiliki dasar keputusan yang lebih kuat dibandingkan penggunaan satu model tunggal [4]. Pada deteksi kloning suara, kebutuhan terhadap fusi muncul karena artefak audio palsu tidak selalu terlihat pada satu bentuk representasi [128]. Sebagian artefak dapat muncul pada pola spektral, sedangkan bagian lain dapat terlihat melalui ketidakwajaran transisi fonem, ritme ujaran, prosodi, atau hubungan temporal antarbagian audio, sehingga penelitian audio *deepfake* banyak mengembangkan model berbasis waktu-frekuensi, representasi *self-supervised*, serta penggabungan beberapa sumber fitur [18]. Oleh karena itu, sistem deteksi modern sering memanfaatkan beberapa bentuk representasi audio, mulai dari fitur buatan tangan, fitur waktu-frekuensi, sampai representasi *self-supervised learning*.

Berdasarkan titik penggabungannya, metode fusi pada deteksi kloning suara dapat dibedakan menjadi fusi fitur, fusi konkatenasi, fusi berbasis *attention*, dan fusi skor [4]. Fusi fitur menggabungkan representasi sebelum klasifikasi, sedangkan fusi *embedding* menggabungkan vektor laten yang sudah dibentuk oleh model [129]. Fusi berbasis *attention* menambahkan pembobotan agar sistem dapat memilih bagian representasi yang lebih penting, sementara, fusi skor dilakukan setelah setiap model menghasilkan keluaran prediksi, sehingga representasi internal tiap cabang tidak perlu disatukan sejak awal [130]. Perbedaan ini penting karena fusi fitur, *embedding*, dan *attention* umumnya

memerlukan penanganan tambahan terhadap bentuk representasi, seperti penyelarasan dimensi, penyesuaian skala atau distribusi fitur, serta pelatihan modul fusi, sedangkan fusi skor lebih sederhana ketika rancangan hibrida ingin mempertahankan kemandirian tiap model penyusunnya [18].

Sebuah penelitian menggunakan model HuBERT dan WavLM, menunjukkan bahwa penggabungan beberapa representasi tidak selalu meningkatkan performa [28]. Kedua model masing-masing menghasilkan EER 0,63% dan 0,64%. Fitur konkatenasi justru meningkatkan EER menjadi 0,89%, sedangkan fusi skor *mean* menghasilkan EER 1,72%. Varian fusi skor yang dikembangkan lebih lanjut memperoleh EER 0,63%, lebih baik daripada kedua pendekatan tersebut. Model HuBERT dan WavLM sama-sama merupakan model *self-supervised* berbasis *waveform* dan *transformer*. Meskipun representasinya relatif berdekatan, konkatenasi tetap tidak meningkatkan performa tanpa mekanisme penyelarasan yang dipelajari. Pada penelitian ini, perbedaannya lebih besar karena DPTFAN membentuk representasi waktu-frekuensi dari log-Mel spektrogram dan *side feature*, sedangkan Wav2Vec 2.0 membentuk representasi temporal kontekstual dari *raw waveform* [131], [132]. Penggabungan kedua cabang pada level fitur atau *embedding* akan membutuhkan penentuan layer yang digunakan, penyamaan panjang temporal, pemetaan dimensi, normalisasi distribusi, serta pelatihan *projection layer*, mekanisme *attention*, dan *classifier* baru sehingga dapat mempersulit hasil. Fusi skor dipilih karena penelitian ini bertujuan mempertahankan DPTFAN dan Wav2Vec 2.0 sebagai model yang dikembangkan dan diuji secara mandiri. Skor kedua model dikalibrasi, dinormalisasi, dan digabungkan melalui pembobotan adaptif berdasarkan *confidence* [17], [133]. Pendekatan ini menghindari pelatihan ulang kedua *backbone*, membatasi kompleksitas arsitektur tambahan, dan memungkinkan perubahan performa setelah fusi dibandingkan secara langsung dengan masing-masing model tunggal [134].

2.2.21 Metode Evaluasi Sistem Deteksi

Metrik evaluasi dipakai untuk menilai seberapa baik model membedakan audio asli dan audio hasil kloning [4]. Karena keluaran model pada akhirnya

berupa keputusan klasifikasi biner, evaluasi tidak cukup hanya melihat apakah prediksi benar atau salah, tetapi juga perlu melihat bagaimana perilaku skor model saat ambang keputusan diubah [135]. Metrik yang digunakan dalam penelitian ini, di antaranya:

1. *Equal Error Rate*

   Equal Error Rate (EER) adalah titik ketika *False Acceptance Rate* (FAR) sama dengan *False Rejection* Rate (FRR) [136]. FAR dapat dipahami sebagai kondisi ketika sampel dari kelas yang seharusnya ditolak justru diterima oleh sistem, sedangkan FRR adalah kondisi ketika sampel dari kelas yang seharusnya diterima justru ditolak [137]. Karena kedua jenis kesalahan ini saling dipengaruhi oleh ambang keputusan yang digunakan, EER dipakai untuk menunjukkan titik keseimbangan yang paling representatif antara keduanya. Semakin kecil nilai EER, semakin baik kemampuan sistem dalam membedakan dua kelas yang diuji. Nilai FAR dan FRR dapat dihitung menggunakan Persamaan 2.9 dan Persamaan 2.10, dengan *True Positive* (TP) menyatakan jumlah sampel positif yang berhasil diprediksi sebagai positif, *True Negative* (TN) menyatakan jumlah sampel negatif yang berhasil diprediksi sebagai negatif, *False Positive* (FP) menyatakan jumlah sampel negatif yang salah diprediksi sebagai positif, dan *False Negative* (FN) menyatakan jumlah sampel positif yang salah diprediksi sebagai negatif. Nilai EER selanjutnya ditentukan pada kondisi yang dinyatakan dalam Persamaan 2.12, yaitu saat FAR dan FRR berada pada titik yang sama sebagai representasi keseimbangan antara kesalahan penerimaan dan kesalahan penolakan.

$$FAR = \frac{FP}{FP+T} \tag{2.9}$$

$$FRR = \frac{FN}{FN+TP} \tag{2.10}$$

$$EER = FAR = FRR \tag{2.11}$$

Rumus 2.4 Perhitungan Metrik *Equal Error Rate* [138]

Nilai EER pada praktik evaluasi diperoleh dengan menggeser ambang keputusan hingga ditemukan titik saat FAR dan FRR berpotongan atau berada pada selisih paling kecil. EER dipakai sebagai ukuran standar untuk menilai performa sistem autentikasi dan deteksi berbasis sinyal karena metrik ini langsung merepresentasikan keseimbangan dua kesalahan utama pada satu titik operasi [139].

2. *Area Under the Curve*

   Area Under the Curve (AUC) adalah luas area di bawah kurva *Receiver Operating Characteristic* (ROC) [135]. AUC menilai kemampuan model untuk memisahkan dua kelas secara menyeluruh di seluruh rentang ambang. Perhitungan AUC didasarkan pada kurva ROC yang dibentuk dari hubungan antara *True Positive Rate* (TPR) dan *False Positive Rate* (FPR) pada berbagai nilai *threshold*. Pada Persamaan 2.12, TPR menyatakan proporsi sampel positif yang berhasil dikenali dengan benar, sedangkan pada Persamaan 2.13, FPR menyatakan proporsi sampel negatif yang keliru diprediksi sebagai positif [140]. Selanjutnya, AUC dihitung menggunakan Persamaan 2.14, yang menyatakan luas area di bawah kurva ROC pada rentang FPR dari 0 hingga 1. Nilai yang mendekati 1 menunjukkan bahwa model memiliki kemampuan pemisahan kelas yang sangat baik, sedangkan nilai 0.5 menunjukkan performa yang mendekati tebakan acak [141]. Metrik ini dipandang penting karena tidak terikat pada satu *threshold* dan lebih stabil untuk membaca kualitas ranking skor model pada klasifikasi biner.

$$TPR = \frac{TP}{TP+FN} \tag{2.12}$$

$$FPR = \frac{FP}{FP+TN} \tag{2.13}$$

$$AUC = \int_0^1 TPR(FPR)\, d(FPR) \tag{2.14}$$

Rumus 2.5 Perhitungan Metrik *Area Under the Curve* [142]

3. Akurasi

   Akurasi adalah proporsi seluruh prediksi yang benar dibandingkan dengan jumlah seluruh data yang dievaluasi [143]. Metrik ini mudah dipahami karena secara langsung menunjukkan seberapa banyak keputusan model yang sesuai dengan label sebenarnya. Dalam klasifikasi biner, akurasi dihitung dari gabungan prediksi benar pada kelas positif dan kelas negatif terhadap seluruh sampel yang diuji. Nilai akurasi dihitung menggunakan Persamaan 2.15 dengan menjumlahkan TP dan TN, lalu dibagi dengan seluruh jumlah prediksi, yaitu TP, TN, FP, dan FN [144]. Nilai akurasi yang tinggi menunjukkan bahwa sebagian besar prediksi model sesuai dengan kondisi sebenarnya, tetapi metrik ini tidak selalu cukup untuk menggambarkan kemampuan model dalam membedakan kelas secara lebih rinci pada berbagai kondisi keputusan [145].

$$Akurasi = \frac{TP+}{TP+TN+FP+FN} \quad (2.15)$$

Rumus 2.6 Perhitungan Metrik Akurasi [144]

Ketiga metrik ini saling melengkapi dan memberikan gambaran menyeluruh mengenai kinerja model deteksi kloning suara. Pertimbangan terhadap EER, AUC, dan akurasi sangat penting dalam mengevaluasi dan membandingkan performa berbagai model deteksi kloning suara.

2.2.22 *User Acceptance Test*

User Acceptance Testing (UAT) merupakan tahap pengujian yang dilakukan untuk menilai apakah sistem yang telah dikembangkan dapat diterima oleh pengguna berdasarkan kebutuhan dan tujuan penggunaan yang telah ditentukan [146]. Pengujian ini berfokus pada sudut pandang pengguna akhir, sehingga aspek yang dinilai berkaitan dengan fungsi sistem yang berjalan, kemudahan alur penggunaan, kejelasan tampilan, pemahaman terhadap keluaran sistem, serta kesesuaian fitur dengan tugas yang harus dilakukan pengguna. Pengujian penerimaan pengguna dapat dilakukan dengan pendekatan berbasis skenario, di mana pengguna diminta menjalankan serangkaian aktivitas yang mewakili alur penggunaan utama sistem [147]. Setiap skenario

disusun berdasarkan fitur yang ingin diuji, hasil yang diharapkan, dan kriteria keberhasilan yang telah ditetapkan [148]. Hasil UAT menunjukkan bagian sistem yang sudah dapat diterima pengguna serta bagian yang masih memerlukan perbaikan [146].

2.3 *Framework*/Algoritma yang digunakan

Penelitian deteksi kloning suara membutuhkan tahapan kerja yang tertata agar pengolahan data, pemodelan, dan evaluasi berjalan konsisten dari awal sampai akhir. Kerangka kerja digunakan untuk menjaga urutan aktivitas penelitian tetap jelas, dari penetapan tujuan, penyiapan data, pengembangan model, sampai evaluasi hasil. Algoritma dipilih untuk menjawab kebutuhan teknis pada klasifikasi audio *real* dan *fake*. Dalam penelitian berbasis *data mining* dan *machine learning*, pendekatan yang sistematis digunakan agar keputusan teknis dapat ditelusuri kembali secara runtut dan dapat dipertanggungjawabkan

2.3.1 *Cross-Industry Standard Process for Data Mining*

Cross-Industry Standard Process for Data Mining (CRISP-DM) adalah *framework* yang sering digunakan untuk menjalankan proyek *data mining* dan membantu mengelola seluruh siklus hidup proyek tersebut [149]. CRISP-DM dikembangkan oleh perusahaan konsultan SPSS pada tahun 1999 sebagai respons atas kebutuhan akan metodologi standar untuk proyek-proyek *data mining* yang semakin kompleks dan beragam [150]. *Framework* CRISP-DM terstruktur dalam enam tahapan utama yang saling berkaitan dan bersifat iteratif. Gambar 2.2 menunjukkan keenam tahapan CRISP-DM [151]:

1. Pemahaman Bisnis

   Tahapan ini berfokus pada memahami permasalahan bisnis yang ingin diselesaikan, menentukan tujuan proyek, dan mengidentifikasi kebutuhan data. Hasil dari tahapan ini adalah definisi yang jelas mengenai permasalahan dan tujuan analisis data.
2. Pemahaman Data

   Pada tahapan ini dilakukan pengumpulan dan eksplorasi awal data untuk memahami karakteristik data, mengidentifikasi pola awal, dan mengatasi

permasalahan kualitas data seperti data yang hilang atau anomali. Outputnya adalah pemahaman menyeluruh atas kualitas dan karakteristik data yang ada.

3. Persiapan Data

   Tahapan ini merupakan langkah krusial yang berfokus pada pembersihan dan transformasi data untuk mempersiapkan data agar dapat digunakan dalam pemodelan. Aktivitas ini dapat mencakup handling missing values, pengubahan tipe data, dan pengurangan dimensi. Hasilnya adalah data yang bersih, konsisten dan siap untuk digunakan dalam proses selanjutnya.

4. Pemodelan

   Tahapan ini melibatkan pemilihan dan penerapan algoritma data mining yang sesuai untuk menghasilkan model prediksi. Tahapan ini dapat melibatkan eksplorasi beberapa model dan algoritma. Hasilnya berupa model yang akan digunakan untuk proses prediksi.

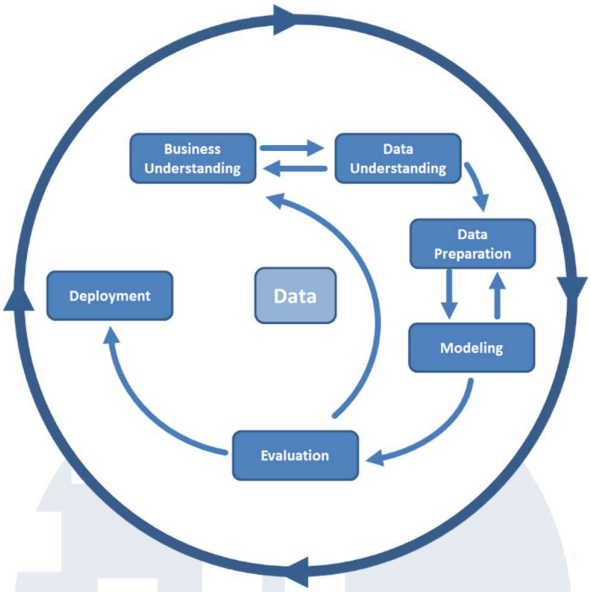
5. Evaluasi

   Tahapan evaluasi bertujuan untuk mengukur performa model yang telah dibangun dan mengevaluasi apakah model tersebut memenuhi kebutuhan bisnis yang telah ditentukan sebelumnya. Tahap ini sangat penting dan dapat membawa proses ke tahap sebelumnya jika performa model belum memuaskan.

6. *Deployment*

   Tahapan ini melibatkan penerapan model yang sudah dievaluasi ke dalam sistem bisnis untuk menghasilkan output yang memberikan manfaat bisnis yang nyata.

Signifikansi penggunaan CRISP-DM terletak pada karakteristiknya yang komprehensif, terstruktur, serta bersifat iteratif. Hal ini membantu manajemen proyek data mining agar lebih terorganisir, mengurangi risiko kegagalan, dan memastikan bahwa model yang dihasilkan dapat memenuhi kebutuhan bisnis. Dalam penelitian ini, CRISP-DM digunakan sebagai kerangka kerja untuk mengelola dan menjalankan seluruh proses penelitian.



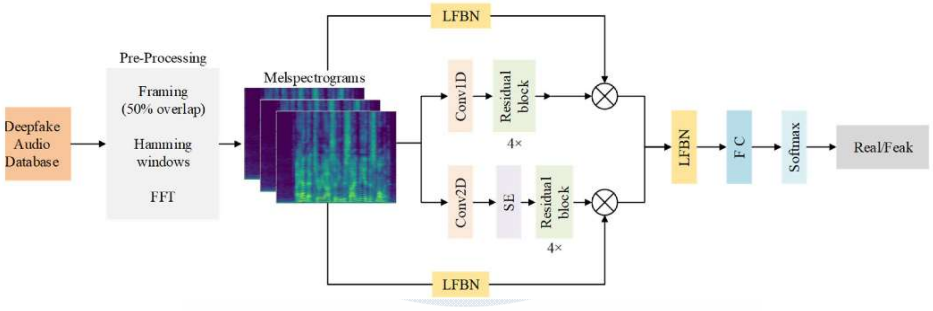
Gambar 2.2 Tahapan Kerangka Kerja CRISM-DM
Sumber: [152]

2.3.2 *Dual-Path Time-Frequency Attention Network*

Dual-Path Time-Frequency Attention Network (DPTFAN) merupakan arsitektur deteksi audio yang memisahkan pembacaan fitur ke dua jalur paralel, jalur waktu dan jalur frekuensi, agar jejak manipulasi dapat ditangkap dari dua sudut yang berbeda [14]. DPTFAN bekerja pada representasi *time frequency* sehingga pola perubahan antarframe dan pola spektral berada dalam satu peta fitur. DPTFAN dipadukan dengan *Pythagorean Hesitant Fuzzy Sets* (PHFS) membawa mekanisme pemodelan ketidakpastian agar fitur yang ragu atau ambigu tetap terwakili secara eksplisit sebelum keputusan akhir dibentuk.

Gambar 2.3 menunjukkan konsep arsitektur DPTFAN berbasis PHFS yang dimulai dengan memasukkan *input* berupa peta fitur *time frequency* seperti log mel, kemudian *backbone* konvolusional membentuk fitur menengah F yang merangkum pola lokal dari *input*. Setelah itu, modul *dual-path attention* membentuk dua peta bobot perhatian. Bobot perhatian waktu menekankan frame yang paling berpengaruh untuk keputusan real dan fake, sedangkan bobot perhatian frekuensi menekankan pita spektral yang paling sensitif terhadap jejak sintesis. Dua peta bobot tersebut digunakan untuk membobotkan ulang fitur menengah sehingga bagian yang relevan memiliki kontribusi lebih besar pada

representasi akhir. Di sisi temporal, pembobotan diarahkan untuk menangkap keteraturan perubahan antarkerangka dan kestabilan pola temporal, karena audio hasil sintesis sering menyisakan ketidakwajaran pada transisi antarbunyi, pola jeda, ritme, dan perubahan energi antarsegmen. Di sisi frekuensi, pembobotan diarahkan pada distribusi energi antarpita, struktur harmonik, detail resonansi, dan perubahan spektral antarkerangka, karena sintesis suara dapat memunculkan harmonik yang terlalu rapi, tekstur spektral yang terasa datar, atau relasi antarpita yang tidak konsisten dengan ucapan manusia. Pada tahap berikutnya, keluaran dari jalur waktu dan jalur frekuensi dipertemukan kembali melalui proses penggabungan representasi sebelum skor keluaran dibentuk [21].



Gambar 2.3 Arsitektur Model DPTFAN
Sumber: [14]

Mekanisme *attention* pada DPTFAN dapat dijelaskan sebagai pembobotan ulang terhadap peta fitur hasil ekstraksi awal. Pada Persamaan 2.16 dan Persamaan 2.17, F menyatakan fitur menengah hasil backbone, At menyatakan bobot perhatian pada sumbu waktu, dan Af menyatakan bobot perhatian pada sumbu frekuensi, sedangkan simbol $\odot$ menyatakan perkalian elemen per elemen. Hasil perihitungan dari kedua persamaan tersebut dinyatakan sebagai Ft untuk jalur waktu dan Ff untuk jalur frekuensi.

$$Ft = At \odot F \tag{2.16}$$

$$Ff = Af \odot F \tag{2.17}$$

$$F\text{fusion} = \lambda Ft + (1 - \lambda)Ff \tag{2.18}$$

Rumus 2.7 Perhitungan Mekanisme *Attention* pada DPTFAN [114]

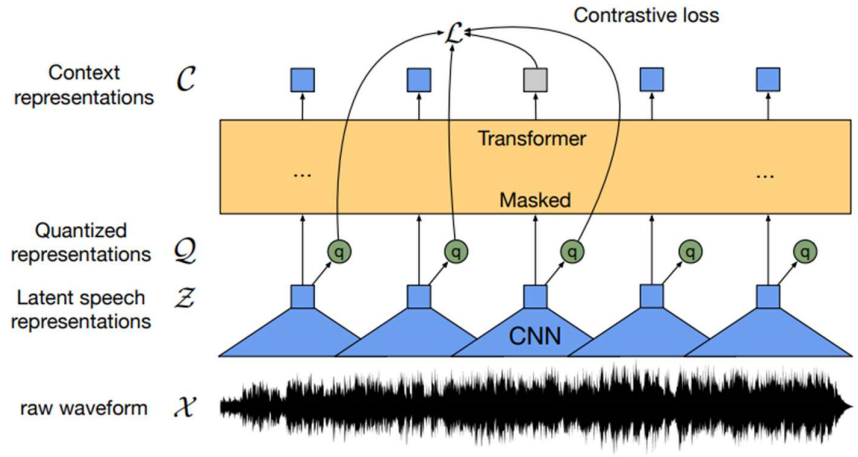
Penggabungan dua jalur dapat dituliskan secara konseptual pada Persamaan 2.18, dengan λ sebagai koefisien penggabungan yang mengatur kontribusi dua jalur dalam membentuk representasi gabungan. Bentuk ini merupakan formulasi konseptual dari proses fusi dua jalur, bukan persamaan baku tunggal yang harus persis sama pada setiap implementasi. Pembacaan waktu dan frekuensi tidak dibiarkan berdiri sendiri sampai keluaran akhir, tetapi dipertemukan kembali agar model memperoleh satu representasi yang sudah tersaring dari dua sudut analisis.

PHFS merepresentasikan kondisi fitur melalui tiga komponen, yaitu membership degree μ, *non-membership degree* ν, dan *hesitancy degree* π. Ketika bukti pada fitur berada di wilayah ragu, informasi tersebut tetap dapat dicatat sebagai derajat keraguan dan tidak langsung dipaksa menjadi keputusan yang sepenuhnya tegas sejak awal [114]. Fitur dengan nilai keraguan π yang lebih tinggi merepresentasikan area sinyal yang lebih samar, sehingga cabang *fuzzy* dapat dipakai untuk menonjolkan atau mengoreksi area tersebut sebelum keputusan akhir dibentuk. DPTFAN berbasis PHFS dirancang untuk menghadapi dua persoalan yang paling sulit pada kloning suara, yakni jejak manipulasi yang tersebar di dua dimensi sinyal dan ketidakpastian fitur yang muncul ketika perbedaan antara audio asli dan audio palsu sangat tipis. *Dual-path attention* membuat model lebih peka terhadap artefak lokal dan lebih siap menghadapi sampel yang berada di area batas keputusan.

2.3.3 Wav2Vec 2.0

Wav2Vec 2.0 merupakan model representasi ucapan berbasis *self-supervised learning* yang membentuk *embedding* langsung dari gelombang audio mentah tanpa memerlukan label [15]. Gelombang audio diproses terlebih dahulu menjadi representasi laten, lalu representasi tersebut diperkaya dengan konteks urutan menggunakan transformer sehingga model dapat membaca ciri akustik permukaan serta menyerap struktur ucapan yang lebih dalam, termasuk hubungan temporal antarbagiannya. Wav2Vec 2.0 dijelaskan sebagai kerangka yang menyembunyikan input di ruang laten dan menyelesaikan tugas kontrasif yang didefinisikan atas kuantisasi representasi laten, sehingga pembelajaran

dilakukan dari struktur internal sinyal ucapan tanpa memerlukan label pada tahap awal. Arsitektur Wav2Vec 2.0 pada Gambar 2.4 tersusun atas rangkaian bertingkat yang memisahkan pembentukan fitur awal, pemodelan konteks, dan target pembelajaran [132]. Dalam praktiknya, arsitektur Wav2Vec 2.0 terdiri atas tiga komponen utama, di antaranya:



Gambar 2.4 Arsitektur Model Wav2Vec 2.0
Sumber: [132]

1. *Feature Encoder*

   Bagian ini berupa rangkaian konvolusi temporal yang mengubah gelombang audio mentah x menjadi representasi laten menggunakan Persamaan 2.19 dengan $f(\cdot)$ sebagai *convolutional feature encoder [153]*. Representasi laten ini dibentuk sebelum masking dan menjadi dasar bagi seluruh proses berikutnya. Fungsi utama dari *feature encoder* adalah memadatkan sinyal mentah menjadi deret fitur yang tetap mempertahankan informasi lokal penting dari audio.

$$z_1, z_2, \ldots, z_T = f(x) \tag{2.19}$$

Rumus 2.8 Perhitungan *Feature Encoder* Wav2Vec 2.0 [132]

2. *Context Network*

   Representasi laten kemudian diproses oleh *transformer* untuk membentuk representasi konteks menggunakan Persamaan 2.20, dengan $g(\cdot)$ sebagai *context network*. Bagian ini berfungsi membaca hubungan antarlangkah waktu, sehingga tiap vektor konteks memuat informasi lokal dan bagian lain

dalam sekuens ucapan [23]. Karena itulah Wav2Vec 2.0 dapat menangkap pola ucapan yang lebih luas daripada model yang hanya membaca *frame* secara lokal.

$$c_1, c_2, \dots, c_T = g(z_1, z_2, \dots, z_T) \tag{2.20}$$

Rumus 2.9 Perhitungan *Context Network* Wav2Vec 2.0 [132]

3. Modul Kuantisasi

   Pada tahap pra-latih, representasi laten dipetakan ke bentuk diskret melalui modul kuantisasi pada Persamaan 2.21, dengan $Q(\cdot)$ sebagai fungsi kuantisasi. Keluaran diskret ini dipakai sebagai target dalam tugas kontrastif. Peran modul ini adah sebagai penyedia sasaran pembelajaran agar model dapat membedakan representasi target yang benar dari kandidat yang salah selama pra-latih [15].

$$q_t = Q(z_t) \tag{2.21}$$

Rumus 2.10 Perhitungan Modul Kuantisasi Wav2Vec 2.0 [132]

Saat tahap pra-latih, sebagian posisi pada representasi laten disamarkan *(masked)*, lalu model diminta mengenali target kuantisasi yang benar di antara sejumlah pengalih. Tujuannya adalah untuk mempelajari struktur internal sinyal ucapan [115]. Fungsi objektif kontrastif yang digunakan dapat dihitung menggunakan Persamaan 2.22, dengan ct sebagai representasi konteks pada waktu ke-t, qt sebagai target kuantisasi yang benar, Qt sebagai himpunan kandidat yang berisi target benar dan pengalih, sim(·) sebagai ukuran kemiripan, dan κ sebagai parameter temperatur. Bentuk ini menunjukkan bahwa model dilatih untuk memberi skor kemiripan paling tinggi pada target yang benar dibanding kandidat lainnya.

$$L_m = -\log \frac{\exp\left(\mathrm{sim}(c_t, q_t)/\kappa\right)}{\sum_{\tilde{q} \in Q_t} \exp\left(\mathrm{sim}(c_t, \tilde{q})/\kappa\right)} \tag{2.22}$$

Rumus 2.11 Perhitungan *Contrastive Loss* [132]

$$L = L_m + \alpha L_d \tag{2.23}$$

Rumus 2.12 Perhitungan *Diversity Loss* [132]

Wav2Vec 2.0 juga memakai *diversity loss* untuk mendorong pemakaian kode kuantisasi yang lebih bervariasi pada Persamaan 2.23 dengan Ld sebagai *diversity loss* dan α sebagai bobot pengali [132]. Kombinasi dua fungsi ini membuat model dapat membedakan target yang benar dan menjaga keberagaman representasi diskret yang digunakan selama pelatihan. Keluaran dari model Wav2Vec 2.0 menghasilkan representasi yang bersifat bertingkat, sehingga tiap lapisan dapat membawa jenis informasi yang berbeda. Lapisan yang lebih dekat ke *encoder* cenderung lebih kaya pada detail akustik lokal, sedangkan lapisan yang lebih dalam pada *transformer* membawa konteks ucapan yang lebih luas [154]. Karena itu, Wav2Vec 2.0 sering dipakai sebagai front-end representasi, sementara keputusan klasifikasi akhir dibebankan pada modul lain yang ditempatkan di belakangnya. Pemilihan subset lapisan transformer dan *fine-tuning* parsial sangat diperhatikan karena tidak semua lapisan menyimpan kontribusi yang sama untuk tugas deteksi suara palsu [128].

Wav2Vec 2.0 dipakai untuk deteksi kloning suara karena model ini dapat membawa sisi representasi yang tidak selalu tampak jelas pada fitur spektral biasa. Wav2Vec 2.0 bekerja dari gelombang mentah untuk membangun embedding yang memuat pola fonetik, karakter ucapan, dan konteks sekuens yang lebih dalam. Keluaran model ini kemudian dapat diteruskan ke *back-end* klasifikasi atau digabungkan dengan model lain agar sistem memiliki dua sudut baca yang saling melengkapi [23]. Atas dasar tersebut, Wav2Vec 2.0 pada penelitian ini dapat diposisikan sebagai komponen yang memperkuat pembacaan sinyal dari sisi representasi ucapan tingkat tinggi, sehingga deteksi real dan fake tidak hanya bergantung pada artefak spektral, tetapi juga pada struktur ucapan yang lebih kaya.

2.3.4 Fusi Skor

Fusi skor atau fusi level skor adalah pendekatan penggabungan yang dilakukan setelah masing-masing model menyelesaikan proses inferensinya dan menghasilkan skor keluaran [155]. Pada jenis fusi ini, setiap model tetap bekerja secara mandiri sesuai representasi yang dipelajarinya, lalu skor prediksi dari beberapa model digabungkan menjadi satu nilai akhir yang dipakai untuk

mengambil keputusan klasifikasi [134]. Posisi fusi skor berada di antara fusi level fitur dan fusi level keputusan, yakni representasi internal tidak dicampur sejak awal, namun keputusan akhir juga belum diambil secara terpisah per model [156]. Karena yang digabungkan adalah skor, fusi ini menjaga kemandirian tiap cabang sambil tetap memanfaatkan kekuatan yang saling melengkapi. Dalam sistem deteksi kloning suara, fusi skor menjadi relevan ketika dua model membaca sinyal dari sudut yang berbeda [18]. Ketika keluaran kedua model sama-sama dapat dinyatakan sebagai skor keyakinan terhadap kelas *real* atau *fake*, penggabungan pada level skor memberi jalan yang lebih bersih untuk menyatukan kontribusi keduanya tanpa harus memaksa representasi internal yang berbeda menjadi satu vektor sejak awal. Fusi skor juga dapat berperan sebagai teknik yang menggabungkan skor keluaran dari model atau sistem berbeda untuk membentuk keputusan yang lebih kuat daripada satu sumber tunggal [134].

Alasan utama penggunaan fusi skor adalah karena sifatnya yang lebih sederhana dan lebih fleksibel dibanding penggabungan di tingkat fitur [155]. Ketika dua model dibangun dari jenis representasi yang berbeda, penggabungan di tingkat fitur sering memerlukan penyesuaian dimensi, penyelarasan distribusi, dan rancangan ulang ruang representasi. Fusi skor tidak menghadapi persoalan tersebut, karena yang dipadukan hanya keluaran akhir yang sudah berada pada ruang keputusan. Implementasinya juga relatif ringan, mudah disesuaikan, dan tetap mampu memanfaatkan kekuatan komplementer dari beberapa sumber, terutama ketika kualitas tiap sumber tidak selalu stabil pada semua kondisi. Sebelum penggabungan, skor perlu diperlakukan secara hati-hati karena nilai keluaran dari dua model dapat memiliki rentang, distribusi, dan tingkat keyakinan yang berbeda [133]. Beberapa cara untuk meningkatkan hasil fusi skor yakni:

1. Kalibrasi skor

   Kalibrasi skor digunakan untuk membuat skor keluaran model lebih selaras dengan makna keputusan yang diwakilinya [157]. Skor mentah dari model deteksi suara belum tentu langsung memiliki arti probabilistik yang setara,

terutama ketika model dibangun dengan arsitektur atau fungsi pembelajaran yang berbeda.

2. Normalisasi skor

   Normalisasi skor digunakan untuk menyamakan skala dan sebaran skor dari beberapa model. Normalisasi dapat menstabilkan skor model dan menurunkan EER sebanyak 2% [158].

3. Pembentukan nilai *confidence*

   Nilai *confidence* digunakan untuk membaca seberapa tegas skor yang diberikan oleh suatu model. Skor yang dekat dengan area netral menunjukkan keputusan yang kurang kuat, sedangkan skor yang jauh dari area netral menunjukkan keyakinan yang lebih tinggi. Tahap ini berguna karena dua model dapat memberikan tingkat keyakinan yang berbeda terhadap audio yang sama.

4. Pembobotan adaptif

   Pembobotan adaptif digunakan untuk menentukan kontribusi tiap model berdasarkan confidence atau reliabilitas skor. Model yang menunjukkan keyakinan lebih tinggi dapat diberi bobot lebih besar, sedangkan model yang lebih ragu diberi bobot lebih kecil.

Secara matematis, bentuk paling umum dari fusi skor adalah penggabungan berbobot yang dihitung menggunakan Persamaan 2.24. Jika skor dari model pertama dinyatakan sebagai S_1 dan skor dari model kedua sebagai S_2, maka skor akhir dapat dirumuskan sebagai Nilai α menyatakan besarnya kontribusi model pertama, sedangkan $(1-\alpha)$ menyatakan kontribusi model kedua, dengan nilai α berada pada rentang 0 dan 1. Ketika bobot dipilih secara tepat, skor akhir akan lebih merepresentasikan kekuatan gabungan kedua model daripada hanya mengikuti salah satu sumber secara dominan.

$$S_{fused} = \alpha S_1 + (1-\alpha) S_2 \tag{2.24}$$

Rumus 2.13 Perhitungan Fusi Skor Berbobot [159]

Fusi skor dipilih karena tujuan utamanya adalah menggabungkan dua pembacaan yang berbeda terhadap audio yang sama tanpa mengubah struktur

dasar masing-masing model. DPTFAN berperan sebagai cabang yang memproses representasi log-Mel spectrogram dan side feature, dengan jalur waktu dan frekuensi yang diarahkan untuk membentuk skor berdasarkan pola lokal, perubahan temporal, serta distribusi energi dan spektral audio. Sementara itu, Wav2Vec 2.0 menerima *waveform* secara langsung dan membentuk representasi ujaran melalui *feature encoder* dan *contextual transformer*, sehingga skor prediksinya merepresentasikan pola fonetik dan hubungan temporal dalam sinyal audio. Pendekatan fusi skor mempertahankan karakter DPTFAN sebagai pembaca representasi akustik waktu-frekuensi dan Wav2Vec 2.0 sebagai pembentuk representasi ujaran kontekstual, kemudian mempertemukan keduanya pada tahap yang paling langsung berkaitan dengan keputusan klasifikasi. Fusi skor menyatukan dua sumber keyakinan menjadi satu skor akhir untuk menentukan apakah audio tergolong *real* atau *fake*, tanpa mencampurkan representasi internal kedua model sejak tahap awal.

2.3.5 BCEWithLogitsLoss

Fungsi *loss* yang digunakan pada penelitian ini adalah BCEWithLogitsLoss karena tugas yang dikerjakan berupa klasifikasi biner, yaitu membedakan audio asli dan palsu [4]. Fungsi *loss* berperan sebagai penghubung antara skor keluaran model dan proses optimasi parameter, terutama untuk keluaran model yang masih berbentuk logit atau skor mentah, sehingga model tidak perlu menerapkan sigmoid secara terpisah sebelum perhitungan loss [118]. BCEWithLogitsLoss menggabungkan Sigmoid dan Binary Cross Entropy dalam satu perhitungan, serta dibuat lebih stabil secara numerik dibandingkan penggunaan sigmoid yang dipisahkan dari BCELoss karena memanfaatkan log-sum-exp trick [160].

$$\mathcal{L}(x, y) = -[y\log\ \sigma(x) + (1 - y)\log\,(1 - \sigma(x))] \tag{2.25}$$

Rumus 2.14 Perhitungan Fungsi *Loss* BCEWithLogitsLoss [161]

Perhitungan *loss* dihitung dengan menggunakan Persamanaan (2.25) dengan x sebagai logit, y sebagai label biner, dan $\sigma(x)$ sebagai fungsi sigmoid [162]. Pada formulasi ini, nilai loss akan mengecil ketika skor prediksi semakin

sesuai dengan label sebenarnya, dan akan membesar ketika model memberi keyakinan tinggi pada kelas yang salah. Karena keluaran akhir model pada penelitian ini adalah skor kecenderungan terhadap kelas tertentu, BCEWithLogitsLoss menjadi pilihan yang tepat untuk mengukur kesalahan prediksi selama proses pelatihan [163]. Penggunaan BCEWithLogitsLoss juga mendukung alur klasifikasi yang lebih langsung, karena model cukup menghasilkan satu skor untuk setiap sampel audio. Skor tersebut kemudian digunakan dalam proses pembelajaran untuk memperbaiki parameter model, sementara keputusan akhir asli atau palsu baru ditentukan pada tahap evaluasi melalui ambang tertentu.

2.3.6 AdamW

Optimizer yang digunakan pada penelitian ini adalah AdamW. Pemilihan ini didasarkan pada kemampuannya melakukan pembaruan parameter secara adaptif, sekaligus menerapkan *weight decay* secara terpisah dari pembaruan gradien utama [164]. *Optimizer* ini memiliki regularisasi bobot bekerja lebih terkontrol dibandingkan formulasi Adam yang mencampurkan penalti bobot ke dalam pembaruan gradien. AdamW tetap mempertahankan inti pembelajaran adaptif dari Adam melalui estimasi momen pertama dan momen kedua gradien, tetapi menambahkan langkah *weight decay* secara terpisah.

$$\theta_t \leftarrow \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} - \eta\lambda\theta_{t-1} \tag{2.26}$$

Rumus 2.15 Perhitungan *Optimizer* AdamW [165]

Bentuk pembaruan parameternya dapat disederhanakan menjadi Persamaan 2.26 dengan θ_t sebagai parameter pada iterasi ke-t, η sebagai *learning rate*, $\hat{m}_t$ sebagai estimasi momen pertama, $\hat{v}_t$ sebagai estimasi momen kedua, ϵ sebagai konstanta stabilisasi, dan λ sebagai koefisien *weight decay* [165]

2.4 Tools/software yang digunakan

Pelaksanaan penelitian ini didukung oleh beberapa alat dan perangkat lunak yang memiliki peran berbeda pada setiap tahap pekerjaan, mulai dari penulisan kode, pelatihan model, pengujian, hingga penyajian hasil. Pemilihan alat dan perangkat lunak didasarkan pada kesesuaiannya dengan kebutuhan pengembangan

sistem deteksi kloning suara yang melibatkan pemrosesan audio, pelatihan model *deep learning*, dan penyajian aplikasi dalam bentuk antarmuka yang lebih mudah digunakan. Perangkat yang digunakan dibahas sesuai fungsinya dalam proses penelitian, sehingga hubungan antara lingkungan pengembangan, kerangka komputasi, dan sarana *deployment* dapat terlihat dengan lebih jelas.

2.4.1 Python

Python adalah bahasa pemrograman tingkat tinggi yang dirancang untuk mendukung penulisan program secara jelas dan mudah dibaca, membuat Python banyak digunakan pada pengolahan data, pengembangan aplikasi, dan pembelajaran mesin [166]. Keunggulan Python terletak pada sintaks yang ringkas, struktur bahasa yang relatif mudah dipahami, serta dukungan pustaka standar yang besar [167]. Python memperkenalkan konsep dasar bahasa dan sistemnya secara langsung, sehingga pengguna dapat membangun program dari logika yang sederhana sampai struktur yang lebih kompleks. Python dipilih sebagai salah satu alat yang digunakan pada penelitian ini karena mampu menjadi fondasi bagi seluruh alur kerja, mulai dari pengolahan data audio, implementasi model, evaluasi, hingga integrasi ke aplikasi [168]. Setiap tahapan dapat dibangun dalam kerangka yang konsisten dengan satu bahasa yang sama.

2.4.2 PyTorch

PyTorch adalah kerangka kerja pembelajaran mesin dan *deep learning* yang menggabungkan representasi data dalam bentuk tensor dengan kemampuan komputasi numerik untuk pelatihan model jaringan saraf [169]. Fokus utama PyTorch berada pada pembangunan model, pengolahan tensor, diferensiasi otomatis, dan optimasi parameter [170]. PyTorch bertumpu pada dua komponen yang sangat penting. Pertama, tensor sebagai struktur data utama untuk merepresentasikan masukan, parameter, dan keluaran model. Kedua, autograd sebagai mesin diferensiasi otomatis yang menghitung gradien saat proses pelatihan berlangsung [171]. PyTorch juga menyediakan mekanisme matematis agar parameter model dapat diperbarui berdasarkan gradien dari fungsi *loss*. PyTorch menjadi kerangka utama untuk mengimplementasikan arsitektur

model deteksi kloning suara dalam penelitian ini karena menyediakan struktur tensor, modul jaringan saraf, proses *forward*, *backpropagation*, dan optimasi parameter dalam satu lingkungan komputasi yang mendukung model *deep learning*.

2.4.3 Antigravity

Antigravity berperan sebagai lingkungan pengembangan, yaitu ruang kerja tempat kode ditulis, dijalankan, dan diuji selama proses penelitian berlangsung [172]. Dalam pengembangan sistem, lingkungan pengembangan memiliki fungsi yang berbeda dari bahasa pemrograman atau kerangka kerja [173]. Bahasa pemrograman menyediakan sintaks dan logika komputasi, sedangkan lingkungan pengembangan menyediakan tempat agar seluruh proses penulisan, pengujian, dan perbaikan kode dapat dilakukan secara lebih tertata [174]. Lingkungan pengembangan adalah sarana yang membantu peneliti mengelola proses eksperimen secara iteratif [175]. Penelitian yang melibatkan banyak percobaan model, perubahan parameter, serta pengujian berulang memerlukan ruang kerja yang memudahkan peninjauan ulang hasil, modifikasi kode, dan pelaksanaan ulang sebagian pipeline tanpa harus membangun ulang seluruh alur dari awal . Dengan fungsi tersebut, lingkungan pengembangan berperan dalam menjaga efisiensi, keterlacakan proses, dan kerapian eksperimen. Antigravity digunakan untuk mendukung proses pengembangan model dan pengujian kode secara bertahap.

2.4.4 Streamlit

Streamlit adalah kerangka kerja Python *open-source* yang dirancang untuk membangun aplikasi data interaktif hanya dengan beberapa baris kode [176]. Streamlit merupakan sebuah perangkat lunak yang berfokus pada penyajian antarmuka aplikasi [177]. Perangkat ini bekerja dengan menambahkan perintah-perintah tertentu ke dalam skrip Python biasa, lalu menjalankannya sebagai aplikasi yang tampil melalui peramban. Streamlit dapat mengubah skrip Python menjadi antarmuka interaktif yang dapat memuat teks, tabel, grafik, serta elemen masukan dari pengguna. Karena itulah Streamlit sering digunakan

ketika model yang sudah selesai dibangun ingin disajikan dalam bentuk aplikasi yang lebih mudah diakses.

Streamlit digunakan pada tahap *deployment* dalam penelitian ini. Perannya adalah menjembatani model deteksi kloning suara dengan pengguna melalui antarmuka yang lebih aplikatif. Sebagai framework presentasi dan interaksi berbasis Python, Streamlit mampu membuat keluaran model ditampilkan dalam bentuk aplikasi sederhana. Streamlit berperan sebagai sarana untuk menyajikan hasil implementasi dalam bentuk yang dapat digunakan secara langsung [178].