BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Suara dan Karakteristik Suara Hewan

Suara adalah gelombang akustik yang merambat melalui medium seperti udara atau air dan dapat ditangkap oleh indera pendengaran (Rabiner & Schafer, 2011). Dalam bentuk digital, suara direpresentasikan sebagai sinyal waktu diskret yang memiliki berbagai atribut seperti frekuensi, amplitudo, dan durasi (Müller, 2015). Setiap spesies hewan memiliki karakteristik suara yang unik, yang bergantung pada struktur anatomi vokal serta kondisi fisiologis dan emosional hewan tersebut (Cummings & Popper, 2016).

Menurut Yeon dkk. (2006), suara hewan dapat mencerminkan berbagai kondisi seperti lapar, stres, sakit, atau bahkan kehamilan. Misalnya, seekor sapi akan mengeluarkan suara berbeda ketika lapar dibandingkan saat merasa tidak nyaman karena dipisahkan dari anaknya. Oleh karena itu, suara hewan bukan hanya alat komunikasi tetapi juga indikator penting dalam memahami perilaku dan kondisi biologis hewan.

Karakteristik akustik seperti frekuensi dominan, intensitas, dan kontur temporal pada suara hewan dapat digunakan sebagai parameter dalam proses identifikasi dan klasifikasi otomatis. Hal ini menjadi dasar dalam pengembangan sistem pengenalan suara hewan berbasis teknologi deep learning (UCKUN & OZKAN, 2023).

2.1.2 Spectrogram

Spectrogram adalah representasi visual dari sinyal suara dalam domain waktu dan frekuensi, yang menunjukkan perubahan spektrum frekuensi terhadap waktu. Dalam spectrogram, sumbu horizontal merepresentasikan waktu, sumbu vertikal merepresentasikan frekuensi, dan intensitas warna menunjukkan amplitudo atau energi pada frekuensi tertentu.

Spectrogram memungkinkan visualisasi pola suara yang sulit diamati langsung dalam bentuk gelombang waktu (time-domain). Dalam konteks

pengenalan suara hewan, spectrogram sangat efektif karena mampu mengekspresikan informasi frekuensi yang khas dari setiap jenis suara hewan. Spectrogram yang dihasilkan dapat diolah sebagai citra dua dimensi, sehingga cocok dijadikan input untuk algoritma CNN (Thornton, 2019; Uckun & Ozkan, 2023).

2.1.3 Short-Time Fourier Transform (STFT)

Short-Time Fourier *Transform* (STFT) adalah teknik transformasi sinyal waktu-ke-frekuensi yang digunakan untuk menganalisis sinyal non-stasioner (Allen, 1977; Rabiner & Schafer, 2011). STFT membagi sinyal menjadi segmen-segmen pendek (frame), lalu menerapkan Fourier *Transform* pada setiap segmen. Hasilnya adalah distribusi frekuensi dari sinyal yang bervariasi terhadap waktu, dan inilah yang membentuk dasar dari *spectrogram* (Müller, 2015).

STFT efektif digunakan dalam pemrosesan sinyal suara karena mampu menangkap perubahan spektrum frekuensi dalam waktu yang singkat, sehingga cocok untuk menganalisis suara hewan yang sering kali bersifat dinamis dan tidak beraturan (Boashash, 2003)

2.1.4 Mel-Spectrogram

Mel-Spectrogram adalah representasi visual dari sinyal audio yang menggambarkan intensitas frekuensi dalam domain waktu, dengan frekuensi ditransformasikan ke dalam skala Mel yang menyerupai cara manusia mendengar (Müller, 2015; Stevens et al., 1937). Metode ini banyak digunakan dalam sistem pengenalan suara karena mampu mempertahankan informasi temporal dan frekuensi secara lebih komprehensif dibandingkan metode lain seperti MFCC (Lawrence R. Rabiner, 2011; McFee, 2015).

Proses pembentukan *Mel-Spectrogram* dimulai dengan pembagian sinyal audio menjadi frame pendek, kemudian setiap frame dianalisis menggunakan Fast Fourier Transform (FFT) untuk mendapatkan spektrum frekuensinya. Hasilnya kemudian dipetakan ke dalam skala Mel menggunakan bank filter, dan ditampilkan dalam bentuk log-amplitudo terhadap waktu. Representasi ini sangat efektif untuk digunakan sebagai input pada model deep learning seperti Convolutional Neural Network (CNN), karena menyerupai citra dua dimensi (gambar) yang bisa diproses

secara visual oleh jaringan.

Mel-Spectrogram sering digunakan dalam berbagai studi pengenalan suara dan klasifikasi suara hewan karena kemampuannya dalam menangkap fitur akustik yang kompleks (Choi et al., 2016; Tokozume & Harada, 2017). Selain itu, teknik ini juga umum digunakan dalam aplikasi real-time yang berbasis mobile karena dapat dikombinasikan dengan model ringan seperti MobileNetV2.

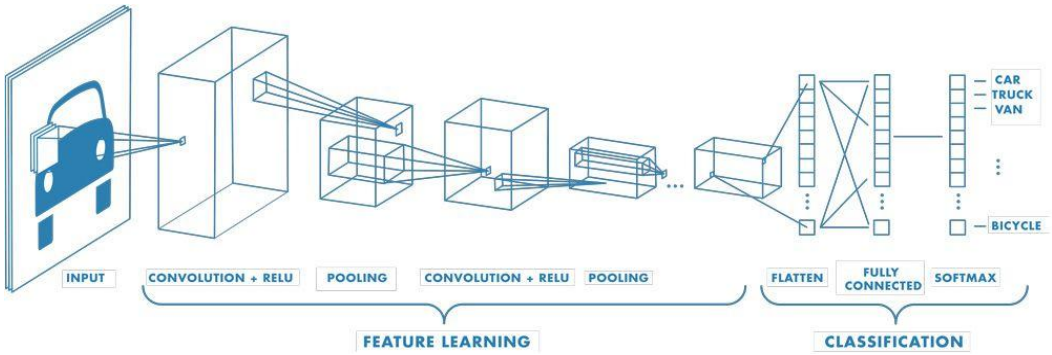
2.1.5 Deep Learning

Deep Learning (DL) merupakan cabang dari machine learning yang menggunakan struktur jaringan saraf tiruan berlapis-lapis (*multi-layered neural networks*) untuk mempelajari representasi data secara otomatis. Jaringan ini terdiri dari neuron-neuron buatan yang diorganisasi dalam lapisan-lapisan, dan memiliki kemampuan untuk mempelajari fitur dari data yang kompleks dan berskala besar. Berbeda dari algoritma pembelajaran tradisional yang membutuhkan rekayasa fitur manual, *deep learning* mampu mengekstraksi fitur secara otomatis melalui proses hierarkis (Alzubaidi dkk., 2021; Schmidhuber, 2014).

Deep Learning telah menjadi metode dominan dalam berbagai bidang seperti pengenalan pola, pengolahan citra, pemrosesan suara, serta pemahaman bahasa alami karena fleksibilitas dan skalabilitasnya dalam menangani data tidak terstruktur (Alzubaidi dkk., 2021; Schmidhuber, 2014).

2.1.6 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) adalah jenis arsitektur *deep learning* yang dirancang untuk mengolah data berbentuk *grid* seperti gambar dua dimensi atau *spectrogram*. CNN terdiri dari tiga bagian utama: *Feature Learning*, Classification, dan Output Prediction. CNN menggunakan lapisan konvolusi (convolutional layers) untuk mendeteksi fitur lokal, *pooling layers* untuk mereduksi dimensi, dan *fully connected layers* untuk klasifikasi akhir (Zhao dkk., 2024).



Gambar 2.1 Tahapan Proses CNN (MatWorks, 2024)

Pada Gambar 2.1 kita bisa melihat pada tahapan proses CNN ini, terdapat dua bagian utama yaitu, *feature learning* dan *classification*.

2.1.6.1 Feature Learning

Feature learning merupakan proses utama dalam CNN yang bertujuan untuk mengekstraksi fitur penting dari data input. Proses ini terdiri dari dua komponen penting yaitu convolutional layer dan pooling layer. Lapisan-lapisan ini bekerja secara bertahap untuk menangkap representasi spasial dari gambar atau citra *spectrogram.*

1) Convolutional Layer

Convolutional layer bertugas untuk mengekstraksi fitur lokal dari data input. Operasi konvolusi dilakukan menggunakan kernel (filter) yang bergerak melintasi gambar input. Hasil dari operasi ini disebut feature map, yang menggambarkan area-area penting dalam gambar. Setiap filter memiliki kemampuan untuk mendeteksi pola seperti garis, tepi, atau tekstur. (LeCun et al., 2015)

2) Pooling Layer

Pooling layer digunakan untuk mengurangi dimensi dari *feature map.* Terdapat dua jenis pooling yang umum digunakan, yaitu *max pooling* dan *average pooling*. *Max pooling* mengambil nilai maksimum dari patch tertentu, sedangkan *average pooling* mengambil rata-rata nilai. Penggunaan *pooling layer* membantu mengurangi kompleksitas komputasi dan mencegah *overfitting*. (Szegedy et al., 2016)

3) Classification

Setelah fitur diekstraksi melalui feature learning, CNN melanjutkan ke tahap klasifikasi. Data dari feature map yang telah diringkas oleh pooling akan diratakan melalui proses flattening dan dilanjutkan ke fully connected layer.

4) Flattening

Flattening adalah proses meratakan hasil dari feature map menjadi vektor satu dimensi agar dapat digunakan sebagai input untuk lapisan-lapisan dense. Tahapan ini menjadi jembatan antara bagian ekstraksi fitur dan klasifikasi. (Yamashita et al., 2018)

5) Fully Connected Layer

Fully connected layer berfungsi menggabungkan fitur-fitur yang telah diekstraksi untuk menghasilkan keputusan klasifikasi akhir dalam CNN (Khan et al., 2020)

6) SoftMax

Softmax adalah fungsi aktivasi yang digunakan pada lapisan output dalam klasifikasi multikelas. Fungsi ini mengubah output dari layer sebelumnya menjadi probabilitas untuk setiap kelas, dan nilai tertingginya digunakan sebagai prediksi akhir. (Yamashita et al., 2018)

2.1.7 Preprocessing Audio

Preprocessing audio adalah tahap awal penting dalam pengolahan sinyal suara yang bertujuan untuk meningkatkan kualitas dan konsistensi data sebelum dilakukan ekstraksi fitur atau klasifikasi. Menurut Müller (2015) dan Rabiner & Schafer (2011), *preprocessing* berperan dalam menghilangkan *noise*, menstabilkan amplitudo sinyal, serta menyeragamkan durasi dan format data suara agar sesuai sebagai input ke dalam sistem pembelajaran mesin.

Beberapa tahapan umum dalam preprocessing audio meliputi:

1. Penghapusan *Noise*

   Dilakukan dengan menerapkan band-pass *filter* pada rentang frekuensi 100 Hz hingga 10 kHz untuk mempertahankan komponen suara hewan, sekaligus mengurangi gangguan suara latar seperti angin atau air. Teknik pengurangan *noise* berbasis spectral subtraction juga sering digunakan dalam sistem pengenalan suara (Xie & Zhu, 2023).

2. Normalisasi Amplitudo

   Untuk menghindari bias akibat perbedaan volume rekaman, sinyal audio dinormalisasi ke rentang [-1, 1]. Proses ini memastikan model tidak terpengaruh oleh variasi level suara yang tidak relevan secara semantik (LeBien et al., 2020).

3. Segmentasi

   Audio dipotong menjadi segmen berdurasi tetap (misalnya 3 detik) untuk memastikan keseragaman input pada saat dikonversi menjadi representasi visual seperti spectrogram (Xie et al., 2020). Segmen tanpa sinyal dominan akan dibuang berdasarkan ambang amplitudo.

4. Penanganan Ketidakseimbangan Data (*Imbalanced Data*)

   Dalam pengembangan model machine learning, imbalanced data atau ketidakseimbangan kelas terjadi ketika jumlah sampel pada satu atau beberapa kategori jauh melebihi kategori lainnya. Kondisi ini dapat menyebabkan model menjadi bias, di mana model akan cenderung memprediksi kelas mayoritas dan mengabaikan kelas minoritas secara statistik. Untuk mengatasi masalah ini, diterapkan teknik Undersampling, yaitu metode pengurangan jumlah data pada kelas mayoritas agar setara dengan jumlah data pada kelas minoritas. Dalam pemrosesan sinyal audio, pengurangan data secara acak (*Random Undersampling*) berisiko membuang informasi akustik yang penting. Oleh karena itu, digunakan pendekatan *Quality-Based Heuristic Undersampling* berdasarkan metrik kualitas suara, yaitu *Root Mean Square* (RMS) *Energy*.

5. Transformasi ke *Spectrogram*

   Setelah *preprocessing* dasar, audio diubah ke bentuk visual menggunakan metode STFT (*Short-Time Fourier Transform*) atau *Mel-Spectrogram.* STFT menangkap representasi frekuensi secara lokal, sedangkan *Mel-Spectrogram* mencerminkan persepsi manusia terhadap frekuensi (Choi et al., 2016; Stevens et al., 1937).

2.1.8 Root Mean Square (RMS) Energy

RMS *Energy* adalah metrik yang digunakan untuk mengukur tingkat kenyaringan atau kekuatan rata-rata dari sebuah sinyal audio. Metrik ini sangat

efektif untuk membedakan antara sinyal yang mengandung suara target (seperti vokalisasi hewan) dengan sinyal yang didominasi oleh keheningan (silence) atau derau latar (background noise). Nilai RMS dari sebuah sinyal digital dengan panjang N dapat dihitung menggunakan persamaan berikut:

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} \tag{2.1}$$

$\boldsymbol{N}$ = Jumlah total sampel dalam sinyal audio.

$\boldsymbol{x_i}$ = Nilai amplitudo pada sampel ke-i.

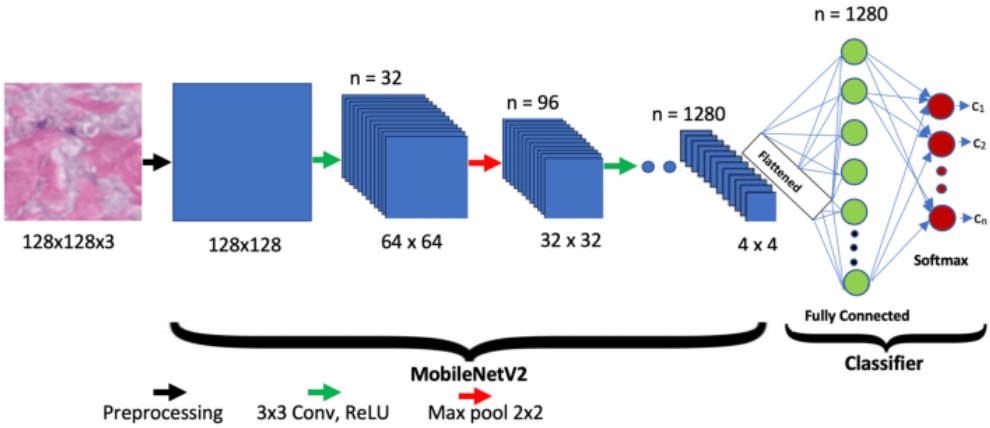
Dengan menggunakan metrik RMS *Energy*, file rekaman audio diurutkan berdasarkan kekuatan sinyalnya dari yang tertinggi hingga terendah. Data yang kemudian dipilih untuk dipertahankan adalah data dengan nilai RMS tertinggi yang diasumsikan memiliki suara hewan paling jelas dan tajam. Sebaliknya, data dengan nilai RMS terendah yang dibuang adalah data yang didominasi oleh *noise* atau keheningan. Teknik ini memastikan bahwa proses penyeimbangan dataset tidak mengorbankan kualitas informasi fitur yang akan diekstraksi oleh model DL.

2.1.9 MobileNetV2

MobileNetV2 merupakan arsitektur CNN yang dirancang khusus untuk aplikasi *mobile* dan sistem dengan sumber daya yang terbatas. MobileNetV2 merupakan pengembangan dari MobileNetV1, yang mengutamakan efisiensi dalam hal penggunaan sumber daya komputasi dan memori tanpa mengorbankan akurasi. MobileNetV2 memperkenalkan dua fitur baru yaitu *inverted residuals* dan *linear bottlenecks. Inverted Residuals* adalah desain blok residu terbalik yang menggunakan koneksi *shortcut* langsung antara lapisan *bottleneck* yang tipis, memungkinkan propagasi gradien yang lebih efisien sambil mempertahankan informasi penting pada representasi fitur. Sementara itu, *linear bottlenecks* menghilangkan fungsi non-linearitas pada lapisan *bottleneck* untuk mengurangi hilangnya informasi penting di sub ruang berdimensi rendah, sehingga memperkuat kapasitas representasi model. (Sandler et al., 2018).

Arsitektur MobileNetV2 juga mengadopsi *depthwise separable convolutions*,

sebuah teknik yang membagi operasi konvolusi menjadi dua tahap: *depthwise convolution* untuk menyaring fitur pada setiap kanal secara independen, dan *pointwise convolution* untuk menggabungkan informasi antar kanal. Pendekatan ini secara signifikan mengurangi kebutuhan komputasi hingga 8-9 kali dibandingkan dengan konvolusi standar, tanpa mengorbankan akurasi. MobileNetV2 juga memanfaatkan ekspansi lapisan sementara untuk memungkinkan jaringan mengekstraksi fitur yang kompleks sebelum memproyeksikan kembali fitur-fitur tersebut ke representasi berdimensi rendah.



Gambar 2.2 Arsitektur MobileNetV2 (Akay et al., 2021)

Gambar 2.2 menunjukkan arsitektur MobileNetV2 yang memanfaatkan *depthwise separable convolution*. Proses dimulai dengan memproses gambar *input* berdimensi 224×224×3 melalui lapisan *convolutional* pertama untuk menghasilkan representasi fitur awal. Operasi *depthwise separable convolution* kemudian diterapkan secara bertahap untuk meningkatkan jumlah kanal (misalnya dari 32 menjadi 640) dan mengurangi resolusi spasial (contohnya dari 112×112 menjadi 56×56) menggunakan operasi dengan *stride*. Pada tahap akhir, fitur dengan dimensi 7×7 dipadatkan menggunakan *Global Average Pooling* untuk menghasilkan representasi berdimensi rendah. Representasi ini diproses lebih lanjut melalui beberapa lapisan *fully connected* (FC) yang dilengkapi *Batch Normalization* untuk meningkatkan stabilitas pelatihan dan mempercepat konvergensi. Akhirnya, lapisan *output* menghasilkan probabilitas untuk setiap kelas target sesuai dengan tugas klasifikasi.

2.1.10 TensorFlow

TensorFlow adalah *framework open-source* yang digunakan untuk membangun dan menjalankan model *machine learning* dan *deep learning*. Framework ini mendukung pelatihan dan *deployment* model pada berbagai *platform,* termasuk *mobile* dan *edge computing*. TensorFlow menyediakan API yang fleksibel dan alat bantu seperti TensorBoard untuk visualisasi proses pelatihan. (TensorFlow, 2024)

2.1.11 Confusion Matrix

Confusion Matrix adalah metode evaluasi model klasifikasi yang menggambarkan hasil prediksi dibandingkan dengan data aktual. Komponen utamanya meliputi *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). Dari matrix ini dapat dihitung metrik seperti akurasi, presisi, *recall*, dan F1-score. (Saito & Rehmsmeier, 2015). Untuk kasus *multi-class* (lebih dari dua kelas), confusion matrix berbentuk matriks N × N *times* N × N, dengan N adalah jumlah kelas. Setiap baris mewakili kelas sebenarnya (*actual class*) dan setiap kolom mewakili kelas hasil prediksi (*predicted class*) (Markoulidakis et al., 2021)

Tabel 2.2 *Confusion Matrix*

Kelas Sebenarnya / Prediksi Kelas	Class 1	Class 2	Class 3	Class 4
Class 1	c1,1	c1,2	c1,3	c1,4
Class 2	c2,1	c2,2	c2,3	c2,4
Class 3	c3,1	c3,2	c3,3	c3,4
Class 4	c4,1	c4,2	c4,3	c4,4

Confusion Matrix dapat dijelaskan sebagai berikut:

1) *True Positive* (TP_i): jumlah sampel kelas *i* yang diprediksi benar sebagai kelas *i.*
2) *False Positive* (FP_i): jumlah sampel dari kelas lain yang salah diprediksi sebagai kelas *i.*
3) *True Negative* (TN_i): jumlah sampel dari kelas lain yang diprediksi benar bukan sebagai kelas *i.*

4) *False Negative* (FN_i): : jumlah sampel kelas i yang salah diprediksi ke kelas lain.

Accuracy dihitung untuk mengevaluasi kinerja sistem dengan mempertimbangkan jumlah total prediksi yang benar yang dibuat oleh pengklasifikasi. Rumus untuk menghitung akurasi adalah sebagai berikut:

$$Accuracy = \frac{\sum_{i=1}^{N} TP(C_i)}{\sum_{i=1}^{N} \sum_{j=1}^{N} C_{i,j}} \quad (2.2)$$

Recall dihitung berdasarkan proporsi *input* positif yang teridentifikasi dengan benar. Ini mengukur tingkat TP dan dihitung pada persamaan 2.2 dengan rumus berikut:

$$Recall = \frac{TP(C_i)}{TP(C_i) + FN(C_i)} \quad (2.3)$$

Precision mengukur jumlah kasus positif yang diprediksi dengan benar oleh pengklasifikasi. Rumus untuk menghitung *precision* pada persamaan 2.3 adalah sebagai berikut:

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

F1-Score merupakan ukuran akurasi yang menggabungkan *precision* dan *recall*. Rumus untuk mengihitung F1 Score dihitung pada persamaan 2.4 dengan rumus berikut:

$$F1\ Score = 2\ x \left(\frac{TPR(C_i)\ x\ PPV(C_i)}{TPR(C_i)\ x\ PPV(C_i)}\right) \quad (2.5)$$

2.1.12 Black Box Testing

Black Box Testing merupakan salah satu metode pengujian perangkat lunak yang sering digunakan, dengan penekanan pada pengujian fungsi sistem tanpa perlu mengetahui detail struktur internal, arsitektur, atau kode programnya. Pendekatan ini bertujuan untuk memeriksa apakah sistem sudah beroperasi sesuai spesifikasi dan kebutuhan yang telah ditetapkan, tanpa perlu memahami secara teknis cara kerja atau rancangan di baliknya. Dalam metode ini, penguji hanya berfokus pada fitur yang tersedia, lalu memberikan input tertentu untuk mengecek apakah output

yang dihasilkan sesuai dengan hasil yang diharapkan. (Dika Pratama & Noviarsyah Dadaprawira, 2023)

2.1.13 Usability Testing

Usability testing merupakan metode evaluasi perangkat lunak yang berfokus pada sejauh mana suatu sistem dapat digunakan oleh pengguna secara efektif, efisien, dan memberikan kepuasan sesuai konteks penggunaannya. Brooke (1996)menyatakan bahwa usability dapat diukur dengan melibatkan responden untuk mencoba sistem secara langsung, kemudian memberikan penilaian melalui instrumen kuesioner berbasis skala Likert. Penilaian ini memungkinkan peneliti memperoleh gambaran kuantitatif mengenai pengalaman pengguna serta mengidentifikasi aspek yang perlu diperbaiki pada sistem agar lebih mudah, cepat, dan nyaman digunakan.

2.2 Penelitian Terdahulu

Penelitian relevan dilakukan oleh UCKUN & OZKAN (2023) dalam International Conference on New Trends in Applied Sciences (ICONTAS'23) dengan judul "Spectrogram Based Classification of Animal Sounds with Deep Learning and Machine Learning Methods". Studi ini menggunakan data suara burung, kucing, dan anjing yang dikonversi menjadi citra *spectrogram* dengan MATLAB, kemudian diekstraksi fiturnya menggunakan SqueezeNet dan Inception v3, serta diklasifikasikan melalui algoritma SVM, K-NN, ANN, Logistic Regression, dan Random Forest. Hasil penelitian menunjukkan kombinasi SqueezeNet + SVM serta Inception v3 + Logistic Regression mampu mencapai akurasi tertinggi, yaitu 99,1%, sehingga membuktikan efektivitas penggabungan deep learning sebagai feature extractor dengan machine learning sebagai classifier dalam mengenali suara hewan.

Penelitian kedua dilakukan oleh Mega Risondang et al. (2019) dalam Journal of Informatics, Information System, Software Engineering and Applications (INISTA) dengan judul "Aplikasi Identifikasi Suara Hewan Menggunakan Metode MFCC". Sistem ini memanfaatkan ekstraksi fitur MFCC yang diklasifikasikan dengan HMM, ditambah penghapusan noise dan normalisasi sinyal. Hasil pengujian menunjukkan metode ini cukup efektif dengan akurasi tinggi pada suara

hewan terisolasi, meskipun kinerjanya dipengaruhi kualitas rekaman dan kebisingan latar.

Penelitian ketiga dilakukan oleh LeBien et al. (2020) dalam Ecological Informatics berjudul "A Pipeline for Automated Species Identification from Acoustic Data Using Deep Learning". Penelitian ini mengembangkan sistem pengenalan suara hewan tropis dengan mengubah audio menjadi spektrogram menggunakan STFT dan mengklasifikasikannya memakai CNN ResNet-50. Sistem ini mencapai akurasi hingga 92% dalam klasifikasi multi-spesies, meskipun menghadapi kendala noise lingkungan dan tumpang tindih frekuensi antarspesies.

Penelitian keempat dilakukan oleh Xie et al. (2020) dalam jurnal Applied Acoustics dengan judul Bioacoustic Signal Classification Using MobileNetV2 and Spectrogram Features. Penelitian ini mengeksplorasi penggunaan arsitektur MobileNetV2 untuk klasifikasi suara hewan berbasis spektrogram. Dataset yang digunakan berasal dari suara burung pada platform Xeno-Canto, yang dikonversi menjadi *Mel-Spektrogram* dan STFT untuk perbandingan. Prosesnya mencakup preprocessing sinyal audio, konversi ke spektrogram, pelatihan model MobileNetV2, dan evaluasi performa menggunakan metrik akurasi, presisi, dan recall. Hasil penelitian menunjukkan bahwa *Mel-Spektrogram* menghasilkan akurasi lebih tinggi (95%) dibandingkan STFT (92%) untuk klasifikasi.

Penelitian kelimat dilakukan oleh Smith dkk. (2021) dalam *Deep Learning-Based Insect Sound Classification Using Wavelet-Based Features* (Journal of Eco-Acoustics). Berbeda dari *spectrogram*, studi ini menggunakan *Continuous Wavelet Transform* (CWT) untuk ekstraksi fitur suara serangga, dilanjutkan dengan klasifikasi menggunakan CNN. Dataset mencakup beberapa spesies serangga tropis dengan hasil akurasi 88% dan F1-score 0,87. Pendekatan ini menyoroti alternatif metode representasi waktu-frekuensi selain STFT/*Mel-Spectrogram*.

Berdasarkan penelitian-penelitian terdahulu, dapat disimpulkan bahwa pendekatan berbasis spektrogram, baik menggunakan STFT maupun *Mel-Spektrogram*, efektif untuk klasifikasi suara hewan, dengan akurasi yang bervariasi tergantung pada dataset, metode ekstraksi fitur, dan model klasifikasi yang digunakan. Penelitian Uckun & Ozkan (2023) menunjukkan keunggulan kombinasi deep learning dan machine learning untuk akurasi tinggi, sementara Risondang et

al. (2019) membuktikan efektivitas metode klasik seperti MFCC untuk perangkat dengan sumber daya terbatas. LeBien et al. (2020) mendemonstrasikan keberhasilan STFT dalam lingkungan tropis yang kompleks, dan Xie et al. (2021) menegaskan efisiensi MobileNetV2 dengan *Mel-Spektrogram*. Penelitian ini berbeda dari penelitian terdahulu karena berfokus pada penggunaan dataset iNaturalist Sounds yang mencakup beragam spesies hewan, membandingkan performa STFT dan *Mel-Spectrogram* secara langsung dalam satu sistem, dan mengimplementasikan model MobileNetV2 pada perangkat Android untuk aplikasi real-time di lapangan. Selain itu, penelitian ini menekankan preprocessing sinyal suara untuk mengatasi tantangan seperti kebisingan lingkungan dan tumpang tindih frekuensi, serta menggunakan metrik evaluasi seperti akurasi, presisi, recall, dan F1-score untuk analisis performa yang komprehensif. Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi baru dalam pengembangan sistem pengenalan suara hewan yang praktis dan efisien untuk keperluan pemantauan ekologi.

Dari kelima penelitian yang telah diuraikan, terlihat bahwa pendekatan pengenalan suara berbasis *spectrogram* dan deep learning memiliki potensi besar dalam klasifikasi suara hewan. Masing-masing penelitian memberikan kontribusi berbeda: dari efisiensi model ringan, penggunaan metode klasik, hingga pengujian dalam kondisi kompleks seperti hutan tropis. Namun, masih terdapat research gap, yaitu belum adanya studi yang secara langsung membandingkan performa dua metode ekstraksi fitur dalam satu sistem yang diimplementasikan untuk perangkat Android. Oleh karena itu, penelitian ini hadir untuk menjawab gap tersebut dengan menyusun sistem klasifikasi suara hewan berbasis MobileNetV2 yang mampu berjalan secara real-time di perangkat Android dan membandingkan efektivitas STFT serta *Mel-Spectrogram* dari segi akurasi, presisi, *recall*, dan F1-score.

Berikut adalah tabel perbandingan penelitian terdahulu berdasarkan metode, dataset, dan hasil:

Tabel 2.1 Penelitian Terdahulu

No.	**Penelitian**	**Metode Ekstraksi Fitur**	**Dataset**	**Hasil (Akurasi)**
1	UCKUN & OZKAN (2023)	*Spectrogram* (MATLAB)	Suara burung, kucing, anjing	99,1%
2	Mega Risondang et al. (2019)	MFCC	Suara hewan lokal	tidak disebutkan secara spesifik
3	LeBien et al. (2020)	STFT	Rekaman hutan tropis	92%
4	Xie & Zhu (2023)	*Mel-Spectrogram*, STFT	Xeno-Canto (suara burung)	Mel-Spec 95%; STFT 92%; plus presisi & recall
5	Smith dkk. (2021)	*Continuous Wavelet Transform* (CWT) + CNN	Suara serangga tropis	88%; F1-score 0,87
6	Jimmy Tan. (2025)	Pengenalan suara hewan berbasis Android menggunakan MobileNetV2 dengan perbandingan STFT dan *Mel-Spectrogram*	Suara Amphibia, Aves, Mammalia, Reptilia, Insecta	93.5%