BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

Penelitian ini berfokus pada pengembangan sistem Klasifikasi Tingkat Serangan otomatis hama wereng coklat (*Nilaparvata lugens*) pada tanaman padi berbasis citra digital menggunakan metode *deep learning*. Sistem ini memanfaatkan pendekatan *transfer learning* pada model *Convolutional Neural Network* (CNN), khususnya arsitektur EfficientNet-B2, RegNetY-800MF, dan ConvNeXt-Tiny. Implementasi sistem diarahkan untuk platform Android dengan pendekatan unggah gambar (non-realtime), yang memungkinkan pengguna (petani atau penyuluh) mengidentifikasi serangan hama secara praktis dan cepat hanya melalui aplikasi mobile. Pada bab ini dijelaskan tentang teori-teori yang mendasari permasalahan dan penyelesaian tesis, baik perangkat keras, perangkat lunak, dan penelitian

2.1.1 Hama Wereng

Padi merupakan tanaman pangan utama yang mendukung ketahanan pangan nasional, terutama di negara agraris seperti Indonesia. Wereng coklat (*Nilaparvata lugens*) adalah serangga penghisap cairan tanaman yang berwarna kecokelatan. Panjang tubuh 2 - 4,4 mm. Serangga dewasa mempunyai 2 bentuk, yaitu bersayap pendek (*brakhiptera*) dan bersayap panjang (*makroptera*) dapat dilihat pada Gambar 2.1. Pada tahap permulaan wereng datang pada pertanaman padi yang sudah mulai tumbuh yaitu pada umur 15 hari setelah tanam atau pada umur 10-20 hari setelah tanam. Di daerah beriklim sedang, pada awalnya populasi wereng coklat rendah, kemudian berkembang dengan cepat. Perkembangan populasi wereng juga tergantung pada inangnya (varietas) padi yang cocok untuk perkembangannya (Iamba & Dono, 2021).

Gambar 2.1 Wereng coklat bersayap (Makroptera)
dan tanpa sayap (Brakhiptera) (Nurbaeti dkk. 2010)

2.1.2 Pengolahan Citra Digital (Image Processing)

Pengolahan citra digital secara umum pengolahan citra digital merupakan langkah-langkah teknik dalam mengestimasi ciri-ciri objek di dalam citra, pengukuran ciri yang berkaitan dengan geometri objek dan menginterpretasi geometri Langkah dalam pengolahan citra digital diawali dari proses penangkapan atau pengambilan citra (image acquisition) menggunakan sensor berupa kamera, alat pemindai,dll. Kemudian dilanjutkan dengan proses persiapan (preprocessing) seperti proses perubahan ukuran (image resizing) atau perbaikan kualitas (image enhancement) sebelum akhirnya digunakan dalam tujuan tertentu (Mumuni & Mumuni, 2022).

2.1.3 Deep Learning

Deep learning adalah suatu kerangka matematis yang digunakan untuk pembelajaran representasi dari suatu data. Saat ini pengembangan *deep learning* sangat pesat dilakukan, salah satunya dalam bidang pertanian. *Deep learning* merupakan teknik pembelajaran mesin yang mengeksploitasi banyak lapisan pemrosesan informasi non-linear untuk ekstraksi dan transformasi fitur supervised maupun unsupervised serta analisis dan klasifikasi pola (Altalak dkk., 2022). Salah satu model *deep learning* yang populer digunakan yaitu Convolutional Neural Network(CNN) yang digunakan untuk memproses data berupa input gambar, memberikan nilai bobot dan dapat membedakan antara objek (Alzubaidi dkk., 2021). Cara ekstraksi fitur otomatis merupakan model pembelajaran yang paling akurat untuk pengolahan citra seperti Klasifikasi Tingkat Serangan objek, klasifikasi, dan pengenalan objek (Yu dkk., 2022).

2.1.4 Teknik Pusposive Sampling

Tahap pengumpulan data, di mana citra tanaman padi diambil langsung dari lahan pertanian di wilayah Siak, Riau. Teknik *Purposive Sampling* digunakan untuk memilih rumpun tanaman padi yang menunjukkan gejala serangan hama wereng coklat. Pengambilan gambar dilakukan menggunakan kamera ponsel dengan jarak ±30 cm dari objek untuk memastikan bahwa detail visual hama dapat tertangkap secara optimal dalam bingkai gambar. Proses ini juga mempertimbangkan variasi pencahayaan dan sudut pandang agar model dapat belajar dari beragam kondisi lapangan (Campbell dkk., 2020).

2.1.5 Pra-pemrosesan data

pra-pemrosesan data yang dilakukan mencakup *resize* gambar ke ukuran input model (misalnya 320x320 piksel), serta augmentasi citra seperti rotasi, *flipping*, dan penyesuaian pencahayaan untuk memperluas keragaman data. Anotasi dilakukan secara manual menggunakan aplikasi LabelImg untuk menandai posisi hama dalam setiap gambar (Lu dkk., 2022).

Data citra yang diperoleh kemudian dibagi menjadi data latih dan data uji. Pembagian data ini dilakukan secara random. Besar pembagian data pada penelitian ini yaitu sebesar 70% untuk data latih, 20% untuk data valid, dan 10% untuk data uji. Setelah itu dilakukan pelabelan data untuk mengetahui objek yang akan dikenali dalam sebuah citra, dalam hal ini objek hama wereng yang ada pada tanaman padi. Citra yang didapatkan memiliki berbagai macam objek, sehingga objek hama wereng dapat dikenali dalam sebuah citra pada klasifikasi CNN *transfer learning*. Pelabelan dilakukan dengan menggunakan tool LabelImg (Lauren, n.d.).

2.1.6 Arsitektur CNN (Convolutional Neural Network)

Metode pengolahan data dua dimensi dalam bentuk citra menggunakan multilayer perceptron (MLP) dalam Deep Neural Network mengalami pengembangan menjadi *Convolution Nerural Network* (CNN), metode MLP tetap dapat digunakan dalam melakukan klasifikasi citra hanya saja dianggap memiliki kekurangan karena tidak dapat menyimpan informasi spasial dan menganggap bahwa setiap piksel yang ada dalam citra sebagai fitur yang independen sehingga

mempengaruhi hasil yang diperoleh, penemuan CNN ini pertama kali dilakukan oleh Hubel dan Wiesel. Convolutional Layer merupakan proses utama dalam CNN yang bertugas melakukan operasi konvolusi pada output dari layer sebelumnya.

Layer filter menggunakan ukuran serta stride eksklusif setiap pergeseran akan dipengaruhi oleh jumlah stride yang akan digeser di seluruh area *feature map* atau activation map. Pada penerapannya, pooling layer yang biasa dipergunakan ialah max pooling dan average pooling. menjadi model, bila kita memakai Max Pooling 2x2 dengan Stride 2, maka di setiap pergeseran filter, nilai yang diambil adalah nilai yang terbesar di area 2x2 tadi, Sedangkan Average Pooling akan mengambil nilai rata-rata. Pada proses ini akan dihasilkan transformasi linier sesuai dengan informasi spasial dari data input. Kernel konvolusi pada tahap trainingnya dapat bersifat *fleksible* disesuaikan dengan input citra pada CNN dengan cara merubah nilai atau bobot pada layer untuk menspesifikan kernel konvolusi yang digunakan (Agustien dkk., 2021).

2.1.7 Model CNN Pre-trained

Model CNN yang digunakan adalah EfficientNet-B2, RegNetY-800MF, dan ConvNeXt-Tiny. Ketiga model ini dipilih karena memiliki performa tinggi dalam klasifikasi citra namun tetap efisien untuk dijalankan di perangkat mobile (Hossen dkk., 2025).

2.1.8 Labelling Citra

Anotasi dilakukan dengan memberikan *bounding box* pada objek wereng yang terdapat pada setiap citra sehingga lokasi objek dapat digunakan sebagai informasi *ground truth* dalam proses pelatihan model object detection berbasis YOLOv8. Kualitas anotasi menjadi salah satu aspek penting dalam penyusunan *dataset* object detection, karena *dataset* dapat mengandung berbagai permasalahan seperti inaccurate *bounding box*, objek yang salah kelas, dan objek yang tidak diberi label (missing labels) (Hussain dkk., 2026). Kesalahan anotasi, khususnya kesalahan dalam penentuan lokasi *bounding box*, dapat memengaruhi proses pelatihan dan evaluasi model deteksi. Kesalahan tersebut dapat menghasilkan label noise sehingga informasi supervisi yang digunakan model menjadi tidak konsisten. Hussain dkk. (2026) menjelaskan bahwa kesalahan anotasi dapat memengaruhi

proses training dan evaluasi model object detection, sedangkan Zreiqat dkk. (2026) menunjukkan bahwa missing objects, mislocalized *bounding box*es, dan ketidakkonsistenan kelas dapat membatasi performa model YOLO.

1. EfficientNet-B2

EfficientNet-B2 merupakan salah satu varian dari keluarga EfficientNet yang dikembangkan oleh Tan dan Le (2019). EfficientNet menggunakan metode *compound scaling* untuk meningkatkan kapasitas model dengan menyeimbangkan tiga dimensi jaringan, yaitu kedalaman (*depth*), lebar (*width*), dan resolusi input (*resolution*) secara bersamaan menggunakan koefisien tertentu. Pendekatan ini berbeda dengan metode penskalaan CNN yang hanya meningkatkan salah satu dimensi jaringan. Arsitektur EfficientNet dibangun dari EfficientNet-B0 sebagai model dasar, kemudian dikembangkan menjadi varian B1 hingga B7 melalui metode *compound scaling*. Struktur dasarnya menggunakan blok *Mobile Inverted Bottleneck Convolution* (MBConv) yang dilengkapi dengan mekanisme *Squeeze-and-Excitation* (SE). Berdasarkan rancangan tersebut, keluarga EfficientNet dikembangkan untuk memperoleh keseimbangan antara performa model dan kebutuhan komputasi dalam tugas klasifikasi citra (Tan & Le, 2020).

```
===== MODEL SUMMARY: EfficientNet-B2 =====
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 efficientnetb2 (Functional)  (None, 10, 10, 1408)     7768569

 global_average_pooling2d (G  (None, 1408)             0
 lobalAveragePooling2D)

 dense (Dense)               (None, 3)                 4227

=================================================================
Total params: 7,772,796
Trainable params: 4,227
Non-trainable params: 7,768,569
_________________________________________________________________
```

Gambar 2.2 Arsitektur EfficientNet-B2 (Sumber: Hasil pengolahan TensorFlow (2026))

Berdasarkan Gambar 2.2, arsitektur EfficientNet-B2 yang digunakan terdiri atas model dasar EfficientNet-B2 sebagai *feature extractor*, lapisan *Global Average Pooling* 2D, dan lapisan Dense sebagai classifier. Model menghasilkan *feature map* berukuran 10 × 10 × 1408 yang kemudian diringkas menjadi vektor fitur berdimensi 1408 sebelum dilakukan proses klasifikasi ke dalam tiga kelas tingkat serangan hama wereng coklat. EfficientNet-B2 memiliki total 7.772.796 parameter, dengan 4.227 parameter *trainable* dan 7.768.569 parameter *non-trainable*. Hal ini

menunjukkan bahwa model menggunakan pendekatan *feature extraction*, di mana bobot *pretrained* EfficientNet-B2 dibekukan dan hanya lapisan klasifikasi yang dilatih menggunakan *dataset* penelitian. Penggunaan EfficientNet-B2 memungkinkan ekstraksi fitur yang efektif dan efisien melalui pemanfaatan bobot *pretrained* dari ImageNet.

2. RegNetY-800MF

RegNet diperoleh melalui pendekatan *network design space*, yaitu perancangan ruang desain yang mencakup sekumpulan arsitektur jaringan, bukan merancang satu arsitektur secara individual. Dari pendekatan tersebut diperoleh struktur jaringan yang sederhana dan teratur, dengan pengaturan lebar (*width*) dan kedalaman (*depth*) jaringan yang mengikuti fungsi linear terkuantisasi. RegNet dikembangkan dalam beberapa konfigurasi dengan kebutuhan komputasi yang berbeda. Varian RegNetY menggunakan blok yang dilengkapi mekanisme *Squeeze-and-Excitation* (SE), sedangkan penamaan 800MF menunjukkan kelas kompleksitas komputasi model yang berada pada kisaran 800 juta FLOPs. Hasil pengujian pada penelitian aslinya menunjukkan bahwa keluarga RegNet mampu memberikan performa yang kompetitif pada berbagai tingkat kebutuhan komputasi (Hossen dkk., 2025).

```
===== MODEL SUMMARY: RegNetY-400MF =====
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 regnety004 (Functional)     (None, 10, 10, 440)       3930296

 global_average_pooling2d_1   (None, 440)              0
 (GlobalAveragePooling2D)

 dense_1 (Dense)             (None, 3)                 1323

=================================================================
Total params: 3,931,619
Trainable params: 1,323
Non-trainable params: 3,930,296
_________________________________________________________________
```

Gambar 2.3 Arsitektur RegNetY-800MF (Sumber: Hasil pengolahan TensorFlow (2026))

Berdasarkan Gambar 2.3, model RegNetY-800MF digunakan sebagai *feature extractor* dengan total parameter sebanyak 3.931.619 parameter. Dari jumlah tersebut, hanya 1.323 parameter yang dapat dilatih (*trainable*), sedangkan 3.930.296 parameter lainnya tidak dilatih (*non-trainable*) karena bobot *pretrained* dibekukan selama proses pelatihan. Strategi ini bertujuan untuk memanfaatkan fitur

yang telah dipelajari sebelumnya sekaligus mengurangi risiko *overfitting* pada *dataset* penelitian yang relatif terbatas. Arsitektur model terdiri atas *backbone* RegNetY-800MF yang menghasilkan *feature map* berukuran 10 × 10 × 440, kemudian dilanjutkan dengan lapisan *Global Average Pooling* 2D untuk mereduksi dimensi fitur menjadi vektor berukuran 440. Pada tahap akhir, digunakan lapisan Dense dengan 3 neuron sebagai classifier untuk mengklasifikasikan citra ke dalam tiga kategori tingkat serangan hama.

3. ConvNeXt-Tiny

ConvNeXt merupakan arsitektur CNN yang dikembangkan dengan memodernisasi desain ResNet melalui beberapa prinsip desain yang digunakan pada *Vision Transformer*, tetapi tetap mempertahankan operasi konvolusi sebagai komponen utama. Pengembangan tersebut mencakup perubahan pada struktur tahapan jaringan, penggunaan *depthwise convolution* dengan kernel 7 × 7, penerapan *Layer Normalization*, serta penggunaan fungsi aktivasi GELU. ConvNeXt tetap dibangun sepenuhnya menggunakan modul CNN standar tanpa mekanisme *self-attention*. Hasil pengujian pada penelitian aslinya menunjukkan bahwa keluarga ConvNeXt mampu memberikan performa yang kompetitif dengan arsitektur *Transformer* pada klasifikasi ImageNet serta dapat digunakan sebagai *backbone* pada tugas deteksi objek dan segmentasi semantik (Syahputra dkk., 2023).

```
===== MODEL SUMMARY: ConvNeXt-Tiny =====
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 convnext_tiny (Functional)  (None, 10, 10, 768)       27820128

 global_average_pooling2d_8   (None, 768)              0
 (GlobalAveragePooling2D)

 dense_8 (Dense)             (None, 3)                 2307

=================================================================
Total params: 27,822,435
Trainable params: 2,307
Non-trainable params: 27,820,128
```

Gambar 2.4 Arsitektur ConvNeXt-Tiny (Sumber: Hasil pengolahan TensorFlow (2026))

Pada gambar 2.4 Arsitektur model ConvNeXt-Tiny digunakan sebagai representasi arsitektur CNN modern yang mengadopsi prinsip desain dari Vision Transformer, namun tetap mempertahankan struktur konvolusional. Dari ringkasan model, ConvNeXt-Tiny menghasilkan *feature map* berukuran 10×10×768 pada lapisan akhir *backbone*. Lapisan *Global Average Pooling* diterapkan untuk

merangkum informasi spasial menjadi vektor fitur berdimensi 768, sebelum diteruskan ke lapisan *Dense* dengan tiga neuron sebagai classifier. Struktur classifier yang sederhana ini bertujuan untuk memastikan bahwa perbedaan performa yang dihasilkan lebih dipengaruhi oleh kemampuan *backbone* dalam mengekstraksi fitur, bukan oleh kompleksitas classifier. ConvNeXt-Tiny memiliki total parameter sebanyak 27.822.435 parameter, dengan 2.307 parameter yang dapat dilatih. Jumlah parameter *non-trainable* yang besar menunjukkan bahwa model ini sangat bergantung pada fitur hasil pretraining ImageNet. Kompleksitas arsitektur ConvNeXt-Tiny memungkinkan model ini menangkap pola visual yang lebih kompleks, namun juga berpotensi memerlukan strategi pelatihan yang lebih stabil, seperti *two-phase training*, untuk mencegah *overfitting*.

2.1.9 Arsitektur Model

Pembuatan model CNN dalam penelitian Klasifikasi Tingkat Serangan hama wereng dilakukan melalui beberapa tahapan utama. Tahapan pertama adalah inisiasi arsitektur model, di mana struktur jaringan dibangun berdasarkan model *pretrained* seperti EfficientNet-B2, RegNetY-800MF, atau ConvNeXt-Tiny. Lapisan-lapisan dasar dari model *pretrained* dipertahankan, sementara bagian akhir (fully connected layers) disesuaikan dengan jumlah kelas dalam *dataset* lokal, yaitu tingkat serangan dengan 3 kelas yaitu ringan, sedang dan berat

2.1.10 Pengembangan Model

Pengembangan model dilakukan menggunakan tiga arsitektur CNN *pre-trained*, yaitu ConvNeXt-Tiny, RegNetY-800MF, dan EfficientNet-B2. Setiap model disesuaikan dengan tiga kelas tingkat serangan, yaitu ringan, sedang, dan berat. Selanjutnya, beberapa *hyperparameter* seperti *learning rate*, *weight decay*, *batch size*, dan *optimizer* diatur untuk memperoleh konfigurasi pelatihan yang sesuai dengan karakteristik *dataset* penelitian. Chu dkk. (2023).

1. *Batch size*

Batch size adalah variabel yang menentukan banyaknya citra atau data latih yang dimasukkan saat proses pelatihan. Semakin kecil nilai ukuran batch yang digunakan, maka proses pelatihan data akan semakin cepat. Jika semakin besar ukuran batch, proses pelatihan pun akan memakan waktu yang cukup lama. Hal

tersebut terjadi karena saat proses pelatihan data, kapasitas penyimpanan yang dibutuhkan semakin besar juga. Hal ini juga turut mempengaruhi akurasi pada sistem. Jika nilai batch size yang digunakan semakin besar, maka akurasinya pun akan semakin tinggi, karena akan semakin banyak fitur yang dipelajari oleh sistem Kim dkk. (2022).

2. *Learning rate*

Learning rate merupakan penentu seberapa banyak weight yang diperbarui dalam proses backpropagation. *Learning rate* juga menentukan kecepatan pada iterasi sehingga dapat mencapai *loss function* minimum. Proses training berjalan semakin cepat jika nilai *learning rate* semakin tinggi Chu dkk. (2023). Namun, nilai *learning rate* yang terlalu tinggi juga dapat menyebabkan nilai *loss function* turun-naik tidak menentu, sehingga dibutuhkan beberapa kali percobaan untuk mendapatkan nilai *learning rate* yang optimal (Konar dkk. 2021).

2.1.11 Pelatihan Model (training)

Tahap ketiga adalah pelatihan model (*training*), di mana model dilatih menggunakan data latih dalam beberapa *epoch*. Selama proses ini, bobot pada jaringan diperbarui secara iteratif menggunakan algoritma backpropagation berdasarkan hasil loss dan optimasi. Proses ini terus berlangsung hingga model mencapai konvergensi atau memenuhi kriteria *early stopping* untuk mencegah *overfitting*.

2.1.12 Evaluasi Model

Untuk menguji seberapa baik kinerja model terhadap data yang diuji, maka dapat dilakukan dengan pendekatan *Confusion matrix*. Adapun proses evaluasi menggunakan *Confusion matrix* dapat dilihat pada Tabel 2. 1 (Hasibuan dkk., 2025).

Tabel 2.1 *Confusion matrix*

Prediksi		
Aktual	Yes	No
Yes	TP	FN
No	FP	TN

True Positive (TP) : Data kelas positif dan nilai prediksi positif.
False Positive (FP) : Data kelas negatif dan nilai prediksi positif.
False Negative (FN) : Data kelas positif dan nilai prediksi negatif.
True Negative (TN) : Data kelas negatif dan nilai prediksi negatif.

Nilai prediksi adalah keluaran dari program dimana nilainya positif atau negatif. Nilai aktual adalah nilai sebenarnya di mana nilainya benar atau salah. Nilai *precision, recall* dan *F1-score* pada *Confusion matrix* dapat dilihat pada persamaan berikut (Hasibuan dkk., 2025):

$$Precision \qquad \frac{TP}{TP + FP} \qquad (2\text{-}1)$$

$$Recall \qquad \frac{TP}{TP + FN} \qquad (2\text{-}2)$$

$$F1 - Score \qquad 2\,\frac{Precision \times Recall}{Precision + Recall} \qquad (2\text{-}3)$$

Keterangan :

Precision : Kemampuan model dalam menghasilkan prediksi positif yang benar.

Recall : Kemampuan model dalam menemukan seluruh data yang benar-benar positif.

F1-score : Nilai harmonis antara *Precision* dan *Recall* yang digunakan untuk mengevaluasi keseimbangan performa model.

2.1.13 *Transfer learning*

Transfer learning (TL) merupakan salah satu pendekatan dalam *deep learning* (DL) yang saat ini semakin banyak digunakan untuk mengatasi keterbatasan data berlabel dalam berbagai bidang, termasuk pertanian. Dalam konteks TL, model yang sudah dilatih sebelumnya (*pre-trained* model) pada domain atau tugas tertentu dapat dimanfaatkan kembali untuk menyelesaikan tugas baru yang memiliki kemiripan, namun dengan data terbatas. Hal ini sangat relevan bagi sektor pertanian yang seringkali kekurangan *dataset* berkualitas (Hossen dkk., 2025).

Transfer learning (TL) dapat didefinisikan dari berbagai sudut pandang. Menurut(Hosna dkk., 2022). TL secara konseptual terdiri dari tiga elemen utama, yaitu "Task" (Tugas), "Domain" (Ranah), dan kombinasi keduanya, yaitu "Task and Domain". Ketiga konsep ini dijelaskan secara terpisah sebagai berikut:

1. Definisi Domain

Domain terdiri dari dua elemen: ruang fitur dan distribusi probabilitas marginal. Sebagai contoh, jika tugas pembelajaran adalah memprediksi pertumbuhan tanaman, maka mencakup variabel-variabel seperti iklim, jenis tanah, dan pemupukan, sedangkan mewakili nilai dari setiap variabel tersebut.

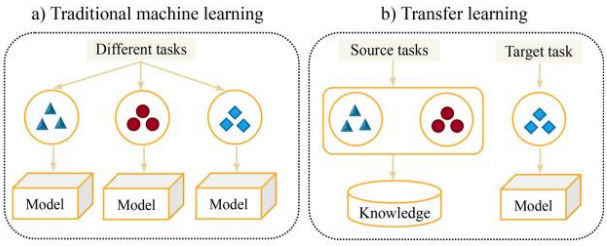
2. Definisi Task

Task terdiri dari ruang label dan fungsi prediksi yang dipelajari dari data pelatihan. Fungsi ini digunakan untuk memperkirakan probabilitas bersyarat yaitu untuk memprediksi label suatu data baru.

3. Definisi Task dan Domain

Dalam *Transfer learning*, diberikan sebuah domain sumber, dan domain target, maka tujuan dari TL adalah meningkatkan kinerja prediksi pada domain target dengan memanfaatkan pengetahuan dari domain sumber, meskipun domain dan tugasnya berbeda.

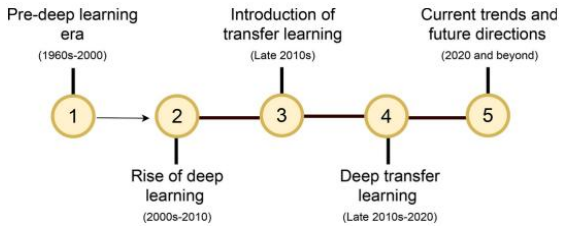
2.1.14 Perbedaan Transfer learning dan Machine Learning Tradisional



Gambar 2.5 Perbedaan *Transfer learning* dengan Machine Learning Tradisional (Hossen dkk. 2025)

Berbeda dari pembelajaran mesin tradisional yang memerlukan pelatihan dari nol untuk setiap tugas, TL memanfaatkan pengetahuan dari satu tugas atau domain (source) untuk meningkatkan kinerja pada tugas lain (target). Keuntungan utamanya adalah penghematan waktu pelatihan dan berkurangnya kebutuhan data berlabel dalam jumlah besar (Hosna dkk., 2022).

2.1.15 Evolusi Transfer learning



Gambar 2.6 Roadmap Evolusi *Transfer learning* (Hossen dkk. 2025)

Menurut (Hosna dkk., 2022) TL mulai berkembang sejak era pra-DL dengan algoritma seperti *decision tree* dan *SVM*, dan mengalami lonjakan besar pada awal 2010-an dengan hadirnya CNN seperti AlexNet. Sejak saat itu, berbagai arsitektur seperti VGG, ResNet, dan EfficientNet-B2 menjadi landasan utama dalam pengembangan TL, termasuk di bidang pertanian.

2.1.16 Jenis-Jenis Transfer learning

TL dapat diklasifikasikan ke dalam empat jenis:

1. Model-based TL: Menggunakan parameter dari model sebelumnya, sering kali dilakukan dengan *fine-tuning*.
2. Instance-based TL: Mentransfer data dengan memberikan bobot berbeda pada setiap instance.
3. Relational-based TL: Mengalihkan pengetahuan berbasis hubungan logis antara domain.
4. Feature-based TL: Menyelaraskan fitur dari domain asal dan target menggunakan transformasi statistik atau geometris.

2.1.17 Penerapan Transfer learning dalam Bidang Pertanian

Pertanian menghadapi banyak tantangan seperti kelangkaan data, variabilitas lingkungan, dan kesulitan dalam pengumpulan data. Oleh karena itu, TL menjadi solusi penting untuk Menghemat waktu pelatihan, Mengurangi kebutuhan data berlabel dan Meningkatkan akurasi model dalam lingkungan yang kompleks.

2.1.18 Tensorflow lite

Tensorflow lite adalah *framework* dari tensorflow yang dibuat khusus untuk mengeksekusi model-model kompleks yang dapat dikembangkan di android (Aqil dkk., 2021).*Tensorflow lite* dapat berjalan tanpa adanya koneksi internet dan lebih

fleksibel terutama ketika menggunakan model custom untuk machine learning model (Taufiq and Pratama, 2021).

Setelah proses pelatihan dan evaluasi selesai, model dengan performa terbaik dikonversi ke format *Tensorflow lite* (.tflite) agar dapat dijalankan pada perangkat Android . Model hasil konversi kemudian diintegrasikan ke dalam aplikasi menggunakan *Tensorflow lite* Interpreter. Pada tahap inferensi, citra yang dipilih pengguna diproses sesuai ukuran input model, kemudian hasil prediksi digunakan untuk menentukan kelas tingkat serangan wereng, yaitu ringan, sedang, atau berat. Proses ini dilakukan secara lokal pada perangkat sehingga aplikasi dapat digunakan tanpa koneksi internet. (Putra, 2020).

2.2 Penelitian Terkait

Penelitian ini merupakan studi literatur yang merujuk pada beberapa penelitian terdahulu di antaranya adalah sebagai berikut:

Penelitian (Virgantara Putra dkk., 2023) melakukan klasifikasi penyakit dan hama pada tanaman padi menggunakan pendekatan *transfer learning*. Beberapa arsitektur dibandingkan, yaitu MobileNetV2, NasNet Mobile, EfficientNet-B7, InceptionV3, VGG16, dan CNN sederhana. *Dataset* terlebih dahulu melalui tahap *preprocessing* dan augmentasi sebelum digunakan dalam proses pelatihan. Hasil pengujian menunjukkan bahwa MobileNetV2 memberikan performa terbaik dengan *accuracy*, *precision*, *recall*, dan *F1-score* sebesar 96%. Penelitian tersebut menunjukkan bahwa perbandingan beberapa model *pre-trained* dapat digunakan untuk menentukan arsitektur yang sesuai dengan karakteristik citra penyakit dan hama padi. Namun, penelitian tersebut berfokus pada klasifikasi jenis penyakit dan hama, sedangkan penelitian ini berfokus pada klasifikasi tingkat serangan hama wereng coklat menjadi ringan, sedang, dan berat.

Pada penelitian (Chen dkk., 2021) mengembangkan sistem untuk menentukan kepadatan serangan hama wereng berdasarkan citra yang diambil secara langsung menggunakan *smartphone*. Tingkat serangan dibagi menjadi empat kategori berdasarkan jumlah wereng, yaitu rendah (0–10), sedang (11–20), tinggi (21–30), dan sangat tinggi (>30). Proses identifikasi dilakukan secara bertahap menggunakan AdaBoost dengan fitur Haar, Support Vector Machine

(SVM) dengan fitur Histogram of Oriented Gradient (HOG), serta penambahan nilai ambang (*threshold*). Hasil pengujian menunjukkan akurasi sebesar 85,2% dengan *false detection* sebesar 9,6%. Kelebihan penelitian ini terletak pada penggunaan citra yang diperoleh langsung menggunakan *smartphone* serta penentuan tingkat serangan berdasarkan jumlah hama, sehingga memiliki kedekatan dengan kondisi lapangan. Namun, metode yang digunakan masih bergantung pada ekstraksi fitur dan penentuan *threshold* secara manual. Penelitian ini menjadi salah satu rujukan karena memiliki objek yang sama, yaitu wereng, tetapi berbeda dengan penelitian ini yang memanfaatkan CNN berbasis *transfer learning* untuk mempelajari karakteristik citra secara otomatis dan mengklasifikasikan tingkat serangan menjadi ringan, sedang, dan berat.

Penelitian oleh (Gu dkk., 2021) menunjukkan keberhasilan penggunaan model CNN pra-latih seperti VGG16, VGG19, dan ConvNeXt-Tiny dalam Proses yang dilakukan mencakup pemotongan citra pada bagian penting, ekstraksi fitur mendalam (*Deep Feature*), dan penggunaan algoritma *k-nearest neighbors* (kNN) untuk proses klasifikasi. Hasil penelitian menunjukkan akurasi hingga 99,61% pada klasifikasi hama serta adanya peningkatan performa setelah dilakukan *fine-tuning*. Kelebihan penelitian tersebut adalah pemanfaatan pengetahuan dari model yang telah dilatih sebelumnya sehingga model dapat disesuaikan dengan data pertanian yang digunakan. Namun, objek penelitian berfokus pada penyakit dan hama tanaman cabai serta menggunakan skema klasifikasi yang berbeda dengan penelitian ini. Oleh karena itu, hasil akurasi yang diperoleh tidak dapat dibandingkan secara langsung. Meskipun demikian, penelitian tersebut memberikan gambaran bahwa *transfer learning* dan *fine-tuning* dapat digunakan untuk menyesuaikan model CNN terhadap karakteristik *dataset* pertanian.

Penelitian oleh (Syahputra dkk., 2023) membandingkan beberapa arsitektur CNN berbasis *transfer learning*, yaitu Xception, RegNetY-800MF, DenseNet201, dan InceptionV3, untuk melakukan klasifikasi 102 jenis hama pada *dataset* IP102. Hasil penelitian menunjukkan bahwa DenseNet201 memperoleh performa terbaik dengan akurasi rata-rata sekitar 70%. Kelebihan penelitian ini adalah penggunaan beberapa arsitektur *pre-trained* pada *dataset* dengan jumlah kelas yang besar, sehingga dapat menunjukkan bahwa setiap arsitektur memiliki kemampuan yang

berbeda dalam mengenali karakteristik visual hama. Namun, penelitian tersebut berfokus pada klasifikasi jenis hama, bukan tingkat serangan dari satu jenis hama tertentu. *Dataset* yang digunakan juga merupakan *dataset* IP102, sehingga karakteristiknya berbeda dengan citra lapangan yang digunakan dalam penelitian ini. Penelitian Syahputra dkk. tetap relevan sebagai dasar penggunaan RegNetY dan pendekatan perbandingan beberapa arsitektur CNN untuk menentukan model yang paling sesuai dengan karakteristik *dataset*.

Penelitian sebelumnya oleh (Khalifa, dkk. 2021) melakukan klasifikasi hama serangga menggunakan pendekatan *transfer learning* dan augmentasi data pada *dataset* IP102 yang terdiri dari sekitar 27.500 citra. Model yang dibandingkan meliputi EfficientNet-B2, GoogLeNet, dan SqueezeNet. Evaluasi dilakukan menggunakan *accuracy*, *precision*, *recall*, dan *F1-score*, dengan EfficientNet-B2 memperoleh akurasi tertinggi sebesar 89,33%. Kelebihan penelitian tersebut terletak pada penggunaan augmentasi dan perbandingan beberapa model *pre-trained*, sehingga dapat diketahui model yang memberikan performa terbaik pada *dataset* yang sama. Namun, penelitian masih berfokus pada klasifikasi jenis hama dalam *dataset* IP102 dan belum membahas tingkat serangan suatu hama secara khusus. Penelitian ini relevan karena menunjukkan kemampuan EfficientNet-B2 dalam klasifikasi citra hama dan menjadi salah satu pertimbangan penggunaan EfficientNet-B2 sebagai model pembanding dalam penelitian ini.

Penelitian oleh (Aristan & Kusuma, 2023) mengevaluasi beberapa arsitektur CNN ringan berbasis *transfer learning* untuk klasifikasi penyakit tanaman dan implementasinya pada perangkat Android . Penelitian tersebut menggunakan lebih dari 104.000 citra yang mencakup 21 jenis tanaman dan 79 kelas. Model yang dibandingkan antara lain MobileNetV3, EfficientNet, ShuffleNetV2, dan Mason model dengan mempertimbangkan akurasi, ukuran model, waktu inferensi, serta kebutuhan sumber daya pada *smartphone*. Hasil pengujian menunjukkan bahwa Mason model memperoleh akurasi 90,54% dengan ukuran model yang kecil, sedangkan MobileNetV3 menunjukkan waktu prediksi sekitar 60,31 ms per citra. Kelebihan penelitian ini adalah evaluasi tidak hanya dilakukan berdasarkan akurasi, tetapi juga mempertimbangkan efisiensi model ketika dijalankan pada perangkat bergerak. Namun, objek yang diklasifikasikan merupakan penyakit tanaman dan

jumlah data yang digunakan jauh lebih besar dibandingkan *dataset* penelitian ini. Penelitian tersebut tetap relevan karena menunjukkan bahwa pemilihan model untuk aplikasi Android perlu mempertimbangkan performa klasifikasi sekaligus ukuran model dan waktu inferensi.

Penelitian sebelumnya oleh Loris Nanni, dkk. (2022) mengeksplorasi pendekatan CNN yang dikombinasikan dengan metode *saliency* untuk klasifikasi hama serangga pada beberapa *dataset*, termasuk IP102. Hasil pengujian menunjukkan bahwa akurasi tertinggi pada *dataset* IP102 mencapai 61,93%. Kelebihan penelitian ini terletak pada penggunaan informasi *saliency* untuk membantu model menekankan bagian citra yang dianggap penting dalam proses klasifikasi. Namun, hasil pada IP102 menunjukkan bahwa klasifikasi hama dengan jumlah kelas yang banyak masih menjadi permasalahan yang cukup sulit. Selain itu, penelitian tersebut berfokus pada klasifikasi jenis hama dan belum membahas tingkat serangan maupun implementasi pada perangkat bergerak. Temuan tersebut menunjukkan bahwa karakteristik *dataset* dan kemiripan visual antarobjek dapat memengaruhi kemampuan model dalam melakukan klasifikasi.

Tabel 2.2 Penelitian Terdahulu

Peneliti	**Metode**	***Dataset***	**Akurasi**	**Kelebihan**	**Kelemahan**
Chen dkk. (2021)	AdaBoost + Haar + SVM	Citra lapangan	85.2%	Pengujian langsung di sawah	Klasifikasi Tingkat Serangan masih banyak *false positive*, bergantung pencahayaan
(Gu dkk., 2021)	ConvNeXt-Tiny	Hot Pepper Disease *Dataset*	99.61%	Akurasi sangat tinggi	Bukan domain wereng/padi
(Syahputra dkk., 2023)	Xception, RegNetY-800MF, DenseNet201	IP-102 *Dataset*	70%	Efektif pada data kecil	Tidak diuji di lapangan
(Aristan & Kusuma, 2023)	CNN ringan (MobileNetV3, EfficientNet-B2)	104.000 citra multi-tanaman	90.54%	Efisien di Android	Tidak fokus pada hama spesifik
Penelitian ini (2026)	EfficientNet-B2, RegNetY-800MF, ConvNeXt-Tiny + TFLite	*Dataset* Wereng Lapangan (Siak, Riau)	Akan diuji	*Dataset* lokal & *offline*	Akan diuji lebih lanjut