Jurnal Teknik Elektro dan Komputasi (ELKOM)
Volume 8 Nomor 1 Maret 2026 ‖ Hal. 102-111
ISSN : 2685-1814 (*Print*)
ISSN : 2685-7677 (*Online*)

Sistem Pendeteksian Cabang Pohon Sengon Berbasis *Artificial Intelligence* Dengan Arsitektur YOLOV4

Ananda Hanif Parikesit[1], Zein Hanni Pradana[1], Wahyu Pamungkas[2*]

[1]Program Studi S1 Teknik Telekomunikasi, Universitas Telkom
[2]Program Studi D3 Teknik Telekomunikasi, Universitas Telkom
Jl. DI Panjaitan No.128, Karangreja, Purwokerto Kidul, Kec. Purwokerto Sel., Kabupaten Banyumas, Jawa Tengah
E-mail: wahyupa@telkomuniversity.ac.id

Naskah Masuk: 10 Juli 2025 ; Diterima: 29 Agustus 2025; Terbit: 31 Maret 2026

ABSTRAK

Abstrak **-** Pohon sengon sebagai komoditas kehutanan bernilai ekonomis tinggi membutuhkan batang yang lurus dan minim percabangan untuk memenuhi standar industri. Namun secara alami, pohon ini cenderung mengalami percabangan lateral berlebihan yang menyebabkan penurunan kualitas kayu akibat terbentuknya mata kayu dan deformasi batang. Teknik pemangkasan manual yang selama ini digunakan sering kali terlambat mendeteksi pertumbuhan cabang, sehingga cabang sudah terlalu besar saat dipangkas dan mengganggu pertumbuhan batang utama. Untuk mengatasi masalah tersebut, penelitian ini mengembangkan sistem pendeteksian cabang berbasis *Artificial Intelligence* (AI) menggunakan arsitektur *You Only Look Once version 4* (YOLOv4). Sistem ini bertujuan untuk mendeteksi pertumbuhan cabang pohon sengon secara otomatis, sehingga proses pemangkasan dapat dilakukan lebih awal. Hasil pengujian menunjukkan bahwa model YOLOv4 dengan konfigurasi *learning rate* 0.01, *batch size* 64, dan *subdivision* 32 mampu mencapai presisi 99% dan *recall* 99%. Dalam uji *real time*, sistem berhasil mendeteksi cabang dengan akurasi sempurna (100%) pada jarak 1–1.5 meter, meskipun akurasinya menurun seiring peningkatan jarak.

Kata kunci: *Artificial Intelligence* (AI), Deteksi Objek, Kayu Sengon, YOLOv4

ABSTRACT

Abstract **-** *The sengon tree as a high economic value forestry commodity requires a straight trunk and minimal branching to meet industry standards. But naturally, this tree tends to experience excessive lateral branching which causes a decrease in wood quality due to the formation of knots and stem deformation. The manual pruning technique that has been used often detects branch growth too late, so that the branches are already too large when pruned and interfere with the growth of the main trunk. To overcome this problem, this research develops an Artificial Intelligence (AI)-based branch detection system using You Only Look Once version 4 (YOLOv4) architecture. This system aims to detect the growth of sengon tree branches automatically, so that the pruning process can be done earlier. The test results show that the YOLOv4 model with a learning rate configuration of 0.01, batch size 64, and subdivision 32 is able to achieve 99% precision and 99% recall. In real-time tests, the system successfully detected branches with perfect accuracy (100%) at a distance of 1-1.5 meters, although the accuracy decreased as the distance increased*.

Keywords: *Artificial Intelligence* (AI), *Object Detection*, Sengon *Tree*, YOLOv4.

1. PENDAHULUAN

Pohon sengon (*Paraserianthes falcataria*) merupakan salah satu komoditas kehutanan bernilai ekonomis tinggi di Indonesia karena pertumbuhannya yang relatif cepat dengan masa panen 5–7 tahun. Jenis ini banyak dibudidayakan untuk memenuhi kebutuhan industri kayu, kayu lapis, dan *furniture*. Namun, kualitas kayu sengon sangat bergantung pada karakteristik pertumbuhan batang yang harus memenuhi standar industri, yaitu tumbuh tegak lurus dengan diameter seragam dan minim percabangan [1]. Secara biologis, pohon sengon cenderung mengembangkan percabangan lateral secara intensif, terutama pada fase pertumbuhan awal. Percabangan yang berlebihan dapat menimbulkan masalah seperti terbentuknya mata kayu dan deformasi batang, sehingga mengurangi nilai ekonomis kayu. Untuk mengatasi hal ini, teknik *pruning* (pemangkasan cabang) diterapkan secara berkala sejak tanaman berumur 6 bulan

hingga 2 tahun dengan interval 6 bulan. Tujuannya adalah menghasilkan batang bebas cabang setinggi 6–8 meter, yang memiliki kualitas lebih tinggi untuk industri perkayuan.

Namun, proses pemangkasan saat ini masih mengandalkan pengamatan manual, yang dinilai kurang efisien, memakan waktu, dan rentan terhadap keterlambatan deteksi. Akibatnya, cabang sering kali tumbuh terlalu besar sebelum dipangkas, menghambat pertumbuhan batang utama dan mengurangi efektivitas teknik *pruning* [2]. Oleh karena itu, diperlukan metode yang lebih akurat dan efisien untuk mendeteksi cabang secara dini untuk mengoptimalkan proses pemangkasan.

Salah satu solusi potensial adalah pemanfaatan *Artificial Intelligence* (AI), khususnya pendekatan *computer vision*, yang telah berhasil diaplikasikan di berbagai sektor termasuk kehutanan [3]. *Convolutional Neural Network* (CNN) sebuah arsitektur *deep learning* yang telah terbukti efektif dalam tugas-tugas pengolahan citra digital seperti klasifikasi, deteksi objek, dan segmentasi [4]. Di antara berbagai arsitektur CNN, *You Only Look Once version* 4 (YOLOv4) menonjol karena kemampuannya dalam deteksi objek secara *real time* [5]. YOLOv4 menjadi pilihan ideal karena kemampuannya mendeteksi objek secara *real time* dengan akurasi tinggi, sehingga cocok untuk memantau pertumbuhan cabang di lapangan secara efisien. Dengan pemanfaatan teknologi AI ini diharapkan dapat dilakukan teknik *pruning* lebih awal sehingga mengoptimalkan pertumbuhan batang utama dan akhirnya meningkatkan nilai ekonomis kayu sengon [6].

Penelitian sebelumnya [7] telah berhasil mengembangkan suatu sistem deteksi dengan hasil mencapai tingkat presisi sebesar 95% dan tingkat akurasi sebesar 88%. Temuan ini menunjukkan keberhasilan sistem dalam mengenali buah lada. Keberhasilan penelitian tersebut membuka peluang untuk mengadopsi pendekatan serupa dalam mendeteksi pertumbuhan cabang pohon sengon dengan menggunakan arsitektur YOLO. Berdasarkan permasalahan tersebut, penelitian ini mengembangkan "Sistem Pendeteksian Cabang Pohon Sengon Berbasis *Artificial Intelligence* Dengan Arsitektur YOLOv4". Tujuan sistem ini adalah mengidentifikasi cabang-cabang yang tumbuh secara tak diinginkan, sehingga tindakan pemangkasan dapat diambil sebelum cabang-cabang tersebut tumbuh besar dan mengganggu pertumbuhan batang pohon sengon.

2. KAJIAN PUSTAKA

2.1. *Convolutional Neural Network*

Convolutional Neural Networks (CNN) merupakan kategori spesifik dari *deep neural networks* yang secara khusus memanfaatkan operasi konvolusi untuk memproses data *input* [8]. CNN terdiri dari neuron dengan bobot (*weight*), bias, dan fungsi aktivasi. Keunikan CNN terletak pada lapisan konvolusi (*convolutional layer*) yang memungkinkannya untuk "melihat" dan mempelajari pola spasial dalam gambar. Proses konvolusi pada CNN melibatkan pergeseran *kernel* konvolusi (filter) berukuran tertentu pada seluruh gambar. Setiap posisi *kernel* menghasilkan nilai baru melalui operasi perkalian antara nilai *piksel* pada gambar dengan nilai pada *kernel*. Hasil konvolusi ini merepresentasikan fitur-fitur spesifik pada gambar yang kemudian digunakan oleh CNN untuk mengenali objek, membedakan gambar, dan "belajar" dari data gambar yang diberikan [9]. Arsitektur CNN dibangun atas serangkaian komponen struktural yang terintegrasi secara sistematis, meliputi: lapisan *input*, lapisan konvolusi, lapisan *pooling*, fungsi aktivasi, lapisan *fully connected*, serta lapisan *output* [10].

2.2. *You Only Look Once* (YOLO)

You only look once (YOLO) adalah sebuah algoritma deteksi objek yang berbasis pada *convolutional neural network*. Dalam arsitektur YOLO, terdapat 24 lapisan konvolusi yang berperan dalam mengekstrak fitur dari gambar *input*. Selanjutnya, diikuti oleh 2 lapisan terhubung yang bertujuan untuk memprediksi probabilitas dan koordinat [11]. YOLO dirancang khusus untuk mendeteksi objek secara *real time*. Proses deteksi pada YOLO melibatkan prediksi langsung terhadap koordinat *bounding box* dan probabilitas kelas untuk sebuah gambar dalam satu evaluasi [12].

Dalam pendeteksian objek, *bounding box* (b-*box*) adalah kotak persegi yang digunakan untuk menandai lokasi objek dalam sebuah gambar. B-*box* didefinisikan oleh koordinat pusatnya (P_x, P_y) serta lebar (P_w) dan tinggi (P_h). Fungsi utamanya adalah mengintegrasikan area yang diusulkan untuk pendeteksian objek, yang dikenal sebagai *anchor box*, dengan objek target yang telah diklasifikasikan. *Anchor box* sendiri adalah sekumpulan b-*box* yang telah ditentukan sebelumnya dengan berbagai bentuk dan ukuran, dan berperan sebagai detektor bagian-bagian objek pada skala spesifik. Sistem *anchor box* ini dirancang untuk meningkatkan akurasi dalam penandaan objek [13].

ISSN : 2685-1814 (*Print*)
ISSN : 2685-7677 (*Online*)

2.3. *You Only Look Once version* 4 (YOLOv4)

You Only Look Once version 4 (YOLOv4) mengimplementasikan pendekatan komprehensif dalam pemrosesan citra. Sistem ini mampu melakukan deteksi simultan terhadap *multiple* objek dalam satu citra dengan menghasilkan prediksi *bounding box* beserta probabilitas kelasnya [14]. Arsitektur YOLOv4 mengintegrasikan CNN untuk ekstraksi fitur dari citra *input*, yang kemudian diproses melalui serangkaian layer konvolusi dan *fully connected* untuk menghasilkan prediksi probabilitas kelas serta lokasi *bounding box* dari setiap objek terdeteksi. Arsitektur YOLOv4 mengintegrasikan tiga komponen utama yang terdiri dari *backbone* CSP *Darknet*-53, *neck* yang mencakup SPP dan PAN, serta *head* yang mengadaptasi YOLOv3 [15].

2.4. *Confusion Matrix*

Confusion matrix merupakan tabel evaluasi yang digunakan untuk mengukur kinerja model klasifikasi dengan membandingkan hasil prediksi terhadap data aktual. Matriks ini menampilkan jumlah prediksi yang benar (*true positive* dan *true negative*) serta kesalahan klasifikasi (*false positive* dan *false negative*) untuk setiap kelas, sehingga memungkinkan penghitungan matriks evaluasi seperti akurasi, presisi, *recall*, dan F1-*score* [16]. Evaluasi kinerja model dapat dianalisis menggunakan *confusion matrix* yang ditunjukkan pada Tabel 1. Matriks ini mengandung empat komponen evaluasi yang menggambarkan hasil klasifikasi, dengan rincian sebagai berikut:

1. *True Positive* (TP): Deteksi yang benar, di mana *bounding box* yang diprediksi sesuai dengan *ground truth bounding box* berdasarkan kriteria tertentu. Kriteria yang dimaksud adalah ketika *bounding box* hasil prediksi sesuai dengan *ground truth bounding box* pada cabang pohon sengon dengan nilai *Intersection over Union* (IoU) mencapai ambang batas ≥ 50%.
2. *False* Positive (FP): Deteksi yang salah, baik karena mendeteksi objek yang tidak ada atau *bounding box* yang tidak sesuai dengan *ground truth*.
3. *False Negative* (FN): *Ground truth bounding box* yang tidak terdeteksi oleh model.
4. Sedangkan komponen *True Negative* (TN) tidak diperhitungkan dalam konteks deteksi objek karena jumlah *bounding box* yang seharusnya tidak terdeteksi bersifat tak terbatas [17].

Tabel 1. *Confusion matrix* [16]

	Actual Positive	***Actual Negative***
Predicted Positive	*True Positive* (TP)	*False Positive* (FP)
Predicted Negative	*False Negative* (FN)	*True Negative* (TN)

Dalam implementasi YOLO dan berbagai variannya, terdapat dua matriks evaluasi yang berperan penting untuk menilai performa deteksi objek. Matriks-matriks ini telah menjadi standar yang umum digunakan dalam pengembangan dan evaluasi sistem deteksi objek:

1) *Precision* mengukur persentase deteksi yang benar dari seluruh deteksi yang dihasilkan. Formula *precision* dapat ditulis menggunakan persamaan 1.

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{Semua \quad Deteksi} \tag{1}$$

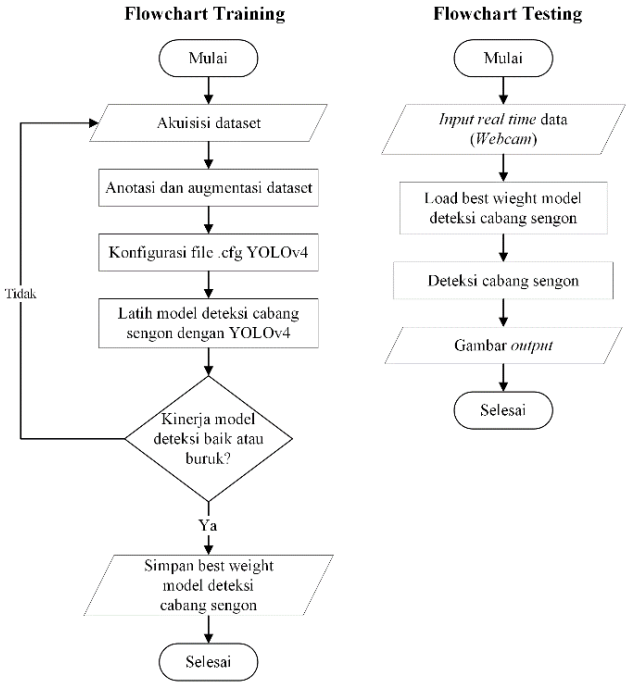
2) *Recall* mengukur persentase *ground truth* yang berhasil dideteksi. Formula *recall* dapat ditulis menggunakan persamaan 2 [18].

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{Semua \quad GroundTruth} \tag{2}$$

3. METODE PENELITIAN

Dalam rangka mengembangkan sistem deteksi cabang berbasis YOLOv4, model YOLOv4 perlu dilatih menggunakan data visual atau sampel cabang pohon sengon. Alur perancangan sistem dapat dilihat pada Gambar 1.

ISSN : 2685-1814 (*Print*)
ISSN : 2685-7677 (*Online*)



Gambar 1. Flowchart perancangan sistem

Gambar 1 mengilustrasikan dua alur utama dalam perancangan sistem deteksi cabang. Alur pertama menjelaskan proses pelatihan, yang meliputi akuisisi, anotasi, dan augmentasi *dataset*, hingga pelatihan model YOLOv4. Alur kedua menjelaskan proses pengujian secara *real time* menggunakan kamera *webcam*.

3.1 Flowchart Training

Pada tahap ini dilakukan akuisisi *dataset* yang akan digunakan untuk melatih model dan mengevaluasi kinerja sistem dalam deteksi cabang pohon sengon. *Dataset* terdiri dari citra cabang pohon sengon yang diambil menggunakan kamera *smartphone*. Total *dataset* yang berhasil dikumpulkan pada tahap ini berjumlah 430 citra cabang pohon sengon. Gambar 2 merupakan salah satu contoh citra cabang pohon sengon.

Gambar 2. Contoh citra cabang pohon sengon

ISSN : 2685-1814 (*Print*)
ISSN : 2685-7677 (*Online*)

Setelah akuisisi, anotasi citra dilakukan untuk menandai objek cabang pohon sebagai target deteksi. Proses anotasi menggunakan platform *Roboflow* diawali dengan mengunggah citra-citra yang telah dikumpulkan. Selanjutnya, *bounding box* ditandai sesuai dengan bentuk cabang pohon sengon, dan seluruh objek diklasifikasikan dalam satu kelas: "cabang". Proses anotasi dapat dilihat pada gambar 3.

Gambar 3. Proses anotasi dataset

Setelah seluruh *dataset* diberi label, *augmentasi* data dilakukan. *Augmentasi* ini meliputi penyeragaman ukuran citra menjadi 416×416 *piksel* dan pemotongan (*crop*) citra sebesar 20%. Selain itu, penyesuaian kecerahan otomatis (*auto brightness*) dengan variasi ±20% diterapkan. Proses *augmentasi* ini meningkatkan jumlah citra dalam *dataset* dari 430 menjadi 1118.

Tabel 2. Parameter YOLOv4.

Parameter	**Nilai**
Batch size	64 / 32 / 16
Subdivision	32 / 16
Learning Rate	0.01 / 0.001 / 0.0001
Channels	3
Burn In	1000
Max Batches	2000
Steps	1600, 1800
Scales	0.1, 0.1
Filter	18
Classes	1

Sebelum proses *training*, peneliti melakukan konfigurasi parameter sistem pada berkas *.cfg*. Konfigurasi ini bertujuan untuk memastikan model dilatih sesuai dengan parameter yang disesuaikan. Detail parameter yang dikonfigurasi disajikan pada Tabel 2. Dalam penelitian ini, variasi *batch size* (64 dan 32), *subdivision* (16 dan 32), dan *learning rate* (0.01, 0.001, dan 0.0001) digunakan untuk mengidentifikasi konfigurasi optimal yang dapat menghasilkan performa model maksimal selama proses pelatihan. Setelah melalui proses *training*, model dievaluasi untuk mengukur performanya dalam mendeteksi objek cabang. Evaluasi dilakukan menggunakan dua matriks utama: presisi (*precision*) yang mengukur akurasi prediksi positif model, dan *recall* yang mengevaluasi kemampuan model dalam mengidentifikasi seluruh prediksi positif dari data.

3.2 Flowchart Testing

Setelah proses pelatihan, model menghasilkan bobot terbaik (*best weight*) yang berfungsi sebagai "otak" sistem untuk mengenali pola dan ciri khas cabang sengon dari citra masukan. Pada tahap pengujian menggunakan *webcam* ini, proses pelabelan data (pemberian *bounding box* secara manual pada citra untuk dijadikan *ground truth*) sudah tidak dilakukan lagi. Hal ini karena model YOLOv4 telah menyelesaikan proses pembelajaran supervisinya dan telah memahami representasi fitur cabang

ISSN : 2685-1814 (*Print*)
ISSN : 2685-7677 (*Online*)

sengon dari dataset yang telah dilabeli selama masa pelatihan**.** Model yang telah terlatih ini kemudian dimuat dan siap digunakan untuk pengujian data *real time* yang diperoleh dari kamera *webcam*. Hasil deteksi ditampilkan sebagai keluaran dalam bentuk citra dengan penandaan berupa kotak pembatas (*bounding box*) pada setiap lokasi cabang sengon yang terdeteksi. Pengujian dilakukan dengan mengarahkan kamera *webcam* ke pohon sengon pada jarak bervariasi dari 1 meter hingga 3 meter, dengan sampel sebanyak 10 buah pohon sengon berukuran 1-2 meter.

4. HASIL DAN PEMBAHASAN

4.1 Hasil *training* model YOLOv4

Tabel 3. Hasil *training* model YOLOv4.

Parameter	*Precision*	*Recall*
Learning Rate : 0.01 *Batch size* : 32 *Subdivision* : 16	69%	85%
Learning Rate : 0.01 *Batch size* : 64 *Subdivision* : 16	98%	100%
Learning Rate : 0.01 *Batch size* : 64 *Subdivision* : 32	99%	99%
Learning Rate : 0.001 *Batch size* : 32 *Subdivision* : 16	89%	99%
Learning Rate : 0.001 *Batch size* : 64 *Subdivision* : 16	93%	99%
Learning Rate : 0.001 *Batch size* : 64 *Subdivision* : 32	91%	99%
Learning Rate : 0.0001 *Batch size* : 32 *Subdivision* : 16	69%	86%
Learning Rate : 0.0001 *Batch size* : 64 *Subdivision* : 16	79%	86%
Learning Rate : 0.0001 *Batch size* : 64 *Subdivision* : 32	73%	85%

Berdasarkan data yang ditampilkan dalam Tabel 3 dapat dianalisis bahwa variasi parameter *Learning Rate*, *Batch size*, dan *Subdivision* secara signifikan mempengaruhi performa model dalam hal *precision*, *recall*, dan waktu komputasi. Evaluasi terhadap berbagai konfigurasi *learning rate* menunjukkan pola yang berbeda-beda, dimana *learning rate* 0.01 menghasilkan performa terbaik dengan *precision* 98-99% dan *recall* 99-100% ketika dikombinasikan dengan *batch size* 64. Sebagai contoh, peningkatan *batch size* dari 32 ke 64 dengan *subdivision* 16 meningkatkan *precision* dari 69% ke 98% dan *recall* dari 85% ke 100%, namun durasi komputasi bertambah dari 1 jam 38 menit menjadi 3 jam 4 menit.

Sementara itu, *learning rate* 0.001 menampilkan konsistensi yang baik dengan *precision* 89-93% dan *recall* stabil di 99%. Keunggulan utama konfigurasi ini terletak pada efisiensi waktu, khususnya dengan *batch size* 64 dan *subdivision* 16 yang hanya membutuhkan 1 jam 43 menit, menjadikannya alternatif yang lebih efisien tanpa mengurangi performa secara drastis. Sebaliknya, *learning rate* 0.0001 menunjukkan hasil kurang optimal dengan *precision* 69-79% dan *recall* 85-86%, mengindikasikan bahwa nilai ini tidak sesuai untuk karakteristik *dataset* yang digunakan dalam penelitian ini.

Selain itu, pengaruh *batch size* dan *subdivision* menunjukkan bahwa *batch size* 64 secara konsisten menghasilkan performa superior dibandingkan *batch size* 32, terutama pada *learning rate* 0.01 dan 0.001, meski dengan konsekuensi peningkatan waktu pelatihan. Parameter *subdivision* menunjukkan bahwa nilai 32 memberikan performa sebanding dengan nilai 16, namun memerlukan durasi pelatihan sedikit lebih lama. Temuan ini menggarisbawahi pentingnya keseimbangan antara performa model dan efisiensi komputasi dalam pemilihan *hyperparameter*.

ISSN : 2685-1814 (*Print*)
ISSN : 2685-7677 (*Online*)

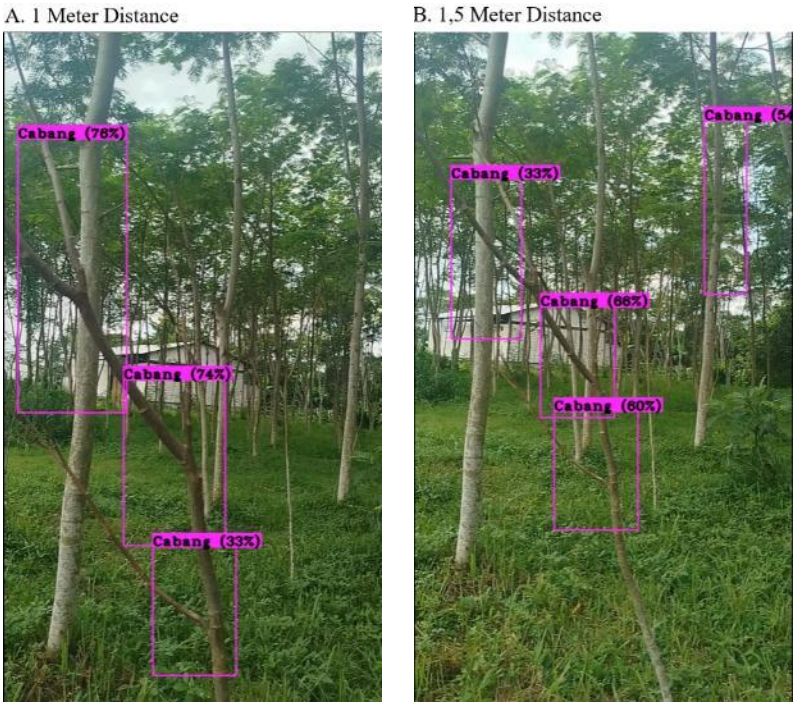
Berdasarkan hasil *training* yang telah dilakukan, konfigurasi optimal untuk model YOLOv4 dalam mendeteksi cabang pohon sengon terdiri dari *learning rate* 0.01, *batch size* 64, dan *subdivision* 16 atau 32. Konfigurasi ini menunjukkan performa terbaik dengan nilai *precision* mencapai 98%, 99% dan *recall* 99%, 100%. Meskipun waktu pelatihan yang diperlukan lebih lama, yaitu 3 jam 4 menit, peningkatan signifikan dalam akurasi membuat pertukaran ini dapat diterima untuk aplikasi sistem deteksi cabang pohon sengon dengan model YOLOv4.

4.2 Pengujian model deteksi cabang pohon sengon secara *real time*

Dalam penelitian ini, pengujian dilakukan secara *real time* dengan menerapkan model yang memiliki nilai *precision* dan *recall* tertinggi berdasarkan hasil pelatihan. Pengujian dilakukan dengan mengarahkan kamera *webcam* pada jarak bervariasi, yaitu 1 meter hingga 3 meter, terhadap sampel sebanyak 10 pohon sengon (tinggi 1–2 m). Tujuan pengujian adalah untuk menganalisis performa dan akurasi model deteksi objek dalam berbagai kondisi jarak.

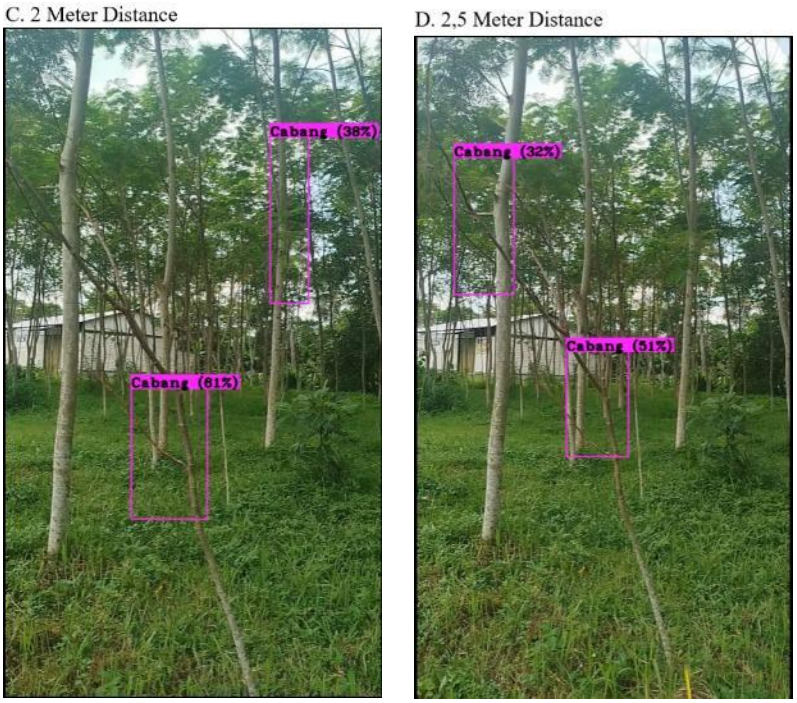
Gambar 4 memperlihatkan hasil deteksi cabang secara *real time* menggunakan *webcam*. Pada jarak 1 meter, model berhasil mendeteksi tiga *bounding box* dengan *confidence score* masing-masing sebesar 76%, 74%, dan 33%, yang seluruhnya dikategorikan sebagai *True Positive* (TP). Selanjutnya, pada jarak 1,5 meter, model mendeteksi empat *bounding box* dengan *confidence score* 66%, 60%, 33%, dan 54%. Dari hasil tersebut, dua *bounding box* (66% dan 60%) tergolong TP, sedangkan dua lainnya (33% dan 54%) diklasifikasikan sebagai *False Positive* (FP) karena terdeteksi pada area yang bukan merupakan cabang.

Gambar 5 menunjukkan hasil deteksi cabang pada jarak 2–2,5 meter. Pada pengujian ini, model berhasil mendeteksi dua *bounding box* dengan *confidence score* 61% dan 51% yang merupakan TP, serta dua *bounding box* lainnya dengan *confidence score* 38% dan 32% yang tergolong FP. Selain itu, pada jarak ini terjadi dua *False Negative* (FN) akibat kegagalan model dalam mendeteksi cabang yang seharusnya teridentifikasi. Gambar 6 menunjukkan kondisi di mana model sepenuhnya gagal mendeteksi cabang pada jarak 3 meter, dengan hasil deteksi pada jarak tersebut dikategorikan sebagai tiga buah FN.



Gambar 4. Hasil deteksi cabang dengan jarak 1 dan 1,5 meter

ISSN : 2685-1814 (*Print*)
ISSN : 2685-7677 (*Online*)



Gambar 5. Hasil deteksi cabang dengan jarak 2 dan 2,5 meter

Gambar 6. Hasil deteksi cabang dengan jarak 3 meter

Tabel 4. Hasil Deteksi cabang menggunakan *webcam* dengan jarak 1-3m.

Percobaan ke	**Jarak**				
	1m	**1,5m**	**2m**	**2,5m**	**3m**
1	Berhasil	Berhasil	Berhasil	Berhasil	Berhasil
2	Berhasil	Berhasil	Tidak	Tidak	Tidak
3	Berhasil	Berhasil	Berhasil	Berhasil	tidak
4	Berhasil	Berhasil	Berhasil	Berhasil	Tidak
5	Berhasil	Berhasil	Tidak	Tidak	Tidak
6	Berhasil	Berhasil	Berhasil	Berhasil	Berhasil

ISSN : 2685-1814 (*Print*)
ISSN : 2685-7677 (*Online*)

Percobaan ke	Jarak				
	1m	1,5m	2m	2,5m	3m
7	Berhasil	Berhasil	Tidak	Berhasil	Tidak
8	Berhasil	Berhasil	Berhasil	Berhasil	Tidak
9	Berhasil	Berhasil	Berhasil	Berhasil	Berhasil
10	Berhasil	Berhasil	Berhasil	Tidak	Tidak

Berdasarkan Tabel 4, hasil pengujian menunjukkan bahwa pada jarak 1 meter dan 1.5 meter model mencapai tingkat keberhasilan deteksi yang sempurna, yaitu 100%, dalam mendeteksi cabang pohon sengon di seluruh percobaan. Hal ini mengindikasikan bahwa pada rentang jarak tersebut, *webcam* mampu menangkap citra cabang pohon sengon dengan sangat optimal. Namun, ketika jarak ditingkatkan menjadi 2 meter, terjadi penurunan tingkat keberhasilan deteksi menjadi 70%. Dari 10 kali percobaan, terdapat 3 kali kegagalan deteksi yang terjadi pada percobaan ke-2, ke-5 dan ke-7.

Pada jarak 2.5 meter, model masih dapat mempertahankan tingkat keberhasilan 70%, meskipun kegagalan deteksi terjadi pada percobaan ke-2, ke-5 dan ke-5. Sementara itu, pada jarak 3 meter, tingkat keberhasilan model menurun secara signifikan menjadi hanya 30%, dengan keberhasilan deteksi tercatat hanya pada percobaan ke-1, ke-6, dan ke-9. Dari hasil pengujian ini, dapat disimpulkan bahwa jarak optimal untuk deteksi cabang pohon sengon menggunakan *webcam* berada pada rentang 1 hingga 1.5 meter.

Hal ini juga sejalan dengan kondisi di kebun sengon, di mana jarak antara satu pohon sengon dengan pohon sengon lainnya berkisar pada 3 meter. Dengan kondisi jarak antar pohon sengon sebesar 3 meter, maka penempatan kamera *webcam* dengan jarak 1.5 meter menjadi optimal, sehingga dapat meningkatkan efektivitas deteksi dan memaksimalkan hasil pendeteksian cabang sengon.

Tabel 5. *Confusion matrix* pengujian *real time*

Predicted Value	***Actual Value***		
		Positive	***Negative***
	Positive	TP = 64	FP = 21
	Negative	FN = 13	TN = 0

Berdasarkan Tabel 5, hasil pengujian deteksi cabang menggunakan kamera *webcam* menunjukkan bahwa TP mencapai angka 64, yang berarti model berhasil mendeteksi cabang dengan benar sebanyak 64 kali. Sementara itu FP bernilai 21, mengindikasikan bahwa model salah mengklasifikasikan objek yang bukan cabang sebagai cabang sebanyak 21 kali. Selain itu, FN bernilai 13, yang menunjukkan bahwa model gagal mendeteksi cabang yang sebenarnya ada sebanyak 13 kali.

Berdasarkan nilai-nilai tersebut, dapat dihitung beberapa matriks evaluasi kinerja sistem:

1. Presisi

$$Precision = \frac{TP}{TP + FP} = \frac{64}{64 + 21} = 0.75 \text{ atau } 75\%$$

2. *Recall*

$$Recall = \frac{TP}{TP + FN} = \frac{64}{64 + 13} = 0.83 \text{ atau } 83\%$$

Hasil perhitungan menunjukkan bahwa sistem memiliki model yang cukup baik dalam hal *recall*, yang mencapai 83%. Ini berarti kemampuan sistem untuk mendeteksi cabang yang sebenarnya cukup tinggi. Namun, tingkat presisi yang lebih rendah, yaitu 75%, mengindikasikan model memiliki tantangan dalam membedakan antara cabang dan objek lain yang menyerupai cabang.

5. KESIMPULAN

Berdasarkan hasil penerapan model YOLOv4 dalam mendeteksi cabang pohon sengon, dapat disimpulkan bahwa konfigurasi optimal model dicapai dengan parameter *learning rate* 0.01, *batch size* 64, dan *subdivision* 32. Konfigurasi ini menghasilkan performa terbaik dengan *precision* dan *recall* masing-masing sebesar 99%. Dalam pengujian deteksi objek secara *real time*, sistem menunjukkan tingkat keberhasilan 100% pada jarak 1–1,5 meter. Namun, pada jarak 2 meter, akurasinya turun menjadi 70%, dan pada jarak 3 meter hanya mencapai 30%. Dengan demikian, jarak optimal untuk deteksi menggunakan *webcam* adalah 1–1,5 meter, yang selaras dengan kondisi lapangan di kebun sengon yang umumnya memiliki jarak tanam sekitar 3 meter. Hasil ini menunjukkan bahwa penerapan YOLOv4 dapat

meningkatkan efektivitas deteksi cabang sengon, terutama jika digunakan dalam rentang jarak yang direkomendasikan.

REFERENSI

[1] H. Krisnawati, E. Varis, M. Kallio, and M. Kanninen, *Paraserianthes falcataria (L.) Nielsen: Ekologi, silvikultur dan produktivitas*. Bogor: Center for International Forestry Research, 2011.
[2] Budiadi, Wiyono, L. D. Lestari, M. Sofiyulloh, and Suyanto, *Tumpangsari dan Hutan Rakyat, Dinamika Budidaya Kayu dan Pangan Petani Jawa*. Yogyakarta: World Agroforestry (ICRAF), 2023.
[3] N. Kühl, M. Goutier, R. Hirt, and G. Satzger, "Machine Learning in Artificial Intelligence: Towards a Common Understanding," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019, p. 10.
[4] E. Putra and W. Suartika, "Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101," *J. Tek. ITS*, vol. 5, no. 1, p. 191064, Mar. 2016.
[5] Q. Aini, N. Lutfiani, H. Kusumah, and M. S. Zahran, "Deteksi dan Pengenalan Objek Dengan Model Machine Learning: Model Yolo," *CESS (Journal Comput. Eng. Syst. Sci.*, vol. 6, no. 2, p. 192, Jul. 2021.
[6] D. Iskandar Mulyana and M. A. Rofik, "Implementasi Deteksi Real Time Klasifikasi Jenis Kendaraan Di Indonesia Menggunakan Metode YOLOV5," *J. Pendidik. Tambusai*, vol. 6, no. 3, pp. 13971–13982, Jul. 2022.
[7] A. Rohim and R. M. Harmie, "Sistem Pendeteksi Buah Lada Berbasis Convolutional Neural Network (CNN)," Politeknik Manufaktur Negeri Bangka Belitung, 2021.
[8] M. Vakalopoulou, S. Christodoulidis, N. Burgos, O. Colliot, and V. Lepetit, "Deep Learning: Basics and Convolutional Neural Networks (CNNs)," in *Machine Learning for Brain Disorders*, 2023, pp. 77–115.
[9] R. M. Awangga, R. Andarsyah, and E. C. Putro, *Object Detection People With Faster region-Based Convolutional Neural Network(Faster R-CNN)*, 1st ed. Bandung: Kreatif lndustri Nusantara, 2020.
[10] M. M. Taye, "Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions," *Computation*, vol. 11, no. 3, p. 52, Mar. 2023.
[11] C. R. Gunawan, N. Nurdin, and F. Fajriana, "Design of A Real-Time Object Detection Prototype System with YOLOv3 (You Only Look Once)," *Int. J. Eng. Sci. Inf. Technol.*, vol. 2, no. 3, pp. 96–99, Oct. 2022.
[12] Z. S. Jannah and F. A. Sutanto, "Implementasi Algoritma YOLO (You Only Look Once) Untuk Deteksi Rias Adat Nusantara," *J. Ilm. Univ. Batanghari Jambi*, vol. 22, no. 3, p. 1490, Oct. 2022.
[13] H.-M. Park and J.-H. Park, "YOLO Network with a Circular Bounding Box to Classify the Flowering Degree of Chrysanthemum," *AgriEngineering*, vol. 5, no. 3, pp. 1530–1543, Aug. 2023.
[14] F. H. Awad, M. M. Hamad, and L. Alzubaidi, "Robust Classification and Detection of Big Medical Data Using Advanced Parallel K-Means Clustering, YOLOv4, and Logistic Regression," *Life*, vol. 13, no. 3, p. 691, Mar. 2023.
[15] M. Sozzi, S. Cantalamessa, A. Cogato, A. Kayad, and F. Marinello, "Automatic Bunch Detection in White Grape Varieties Using YOLOv3, YOLOv4, and YOLOv5 Deep Learning Algorithms," *Agronomy*, vol. 12, no. 2, p. 319, Jan. 2022.
[16] K. M. Ting, "Confusion Matrix," in *Encyclopedia of Machine Learning*, Boston, MA: Springer US, 2011, pp. 209–209.
[17] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, "A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit," *Electronics*, vol. 10, no. 3, p. 279, Jan. 2021.
[18] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A Survey on Performance Metrics for Object-Detection Algorithms," in *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2020, pp. 237–242.